Providing Reliable Route Guidance: Phase II

Prepared by

Yu (Marco) Nie, Xing Wu, Joseph Zissman, Chanhan Lee
Department of Civil & Environmental Engineering
Northwestern University
Michael Haynes, Chicago Transit Authority

December 20, 2010

Contents

List of Tables

List of Figures

Acknowledgement

This work was funded by the Center for the Commercialization of Innovative Transportation Technology at Northwestern University, a University Transportation Center Program of the Research and Innovative Technology Administration of USDOT through support from the Safe, Accountable, Flexible, Efficient Transportation Equity Act (SAFETEA-LU).

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Summary

The overarching goal of the project is to enhance travel reliability of highway users by providing them with reliable route guidance produced from newly developed routing algorithms that are validated and implemented with real traffic data. To these ends, the project further develops the reliable routing algorithms, tests the idea of using the Chicago Transit Authority (CTA) bus data to estimate travel times on arterial and local roads, and conducts a survey designed to understand travelers' reaction and attitude to travel reliability and their opinions about reliable route guidance products.

Algorithm development First, a new convolution method based on Discrete Fourier Transform (DFT) is proposed, which is more difficult to implement but has an attractive theoretical complexity. While the new method can produce highly precise solutions, it significantly increases computation overhead. Compared with the DFT-based method, the direct convolution scheme based on adaptive discretization seems to strike a better balance between accuracy and efficiency, particulary for large-scale applications. Second, the higher order stochastic dominance (SD) rules are employed to solve the reliable routing problem. Numerical experiments show that the higher order SD rules can reduce the number of non-dominated paths, hence promising improvements in computational efficiency. Unlike the existing heuristic methods, this approach guarantees that all generated paths are admissible under first-order SD, or FSD. The optimal solutions solved with the second-order SD (SSD) are found to be close to those solved with FSD, while the CPU time required to solve the reliable shortest path problem is roughly reduced by a factor of two when the SSD rule is employed.

CTA bus data A prototype PL/SQL code is developed to extract travel speed information from AVL bus data and to match them to links in the Chicago regional network. In total, a sample of one month of all matched bus data (about 28 million records) is generated and

analyzed in the study. The results indicate that (1) on freeways and expressways, the bus travel time data have strong correlations with those obtained from loop detectors; (2) the data on arterial and locals streets typically contain larger noises; (3) the arterial data quality is better on longer streets than shorter streets, and better on streets located in suburbs than those in the downtown area; and (4) the bus data tend to overestimate travel time under free flow or light congestion conditions, but better represent the reality in the presence of heavy congestion. Many of these findings are expected. Note that the noises in the arterial data are likely due to the impacts of bus stops and intersections, which are clearly larger when the link is short and/or located in the downtown area. The lack of sufficient data coverage is another drawback of bus data. Results indicate that many arterial links, although served by CTA bus routes, only have a few observations each day, which are far from enough to create a reliable daily travel time profile. Finally, even when they are not disrupted by stops, intersections and congestion, buses and cars are likely to travel at quite different speeds on average. Our results indicate that under free flow conditions the speed difference between buses and cars is likely to be well over 5 mile/hour in the study area.

Travel reliability survey With the help of a professional web survey company, we obtained a sample of 220 valid responses from students and workers in the Chicago Metropolitan area. We found that travel reliability is the second most important decision variable in route choice, next only to travel time. The survey results also confirm that commuters do budget a buffer time to ensure a high probability of on-time arrival for important trips, and that they perceive this as the most important approach to improving reliability. However, to be attractive to most travelers, a reliable route guidance has to generate at least 10 - 20% time savings. Finally, the most desirable forms to commercialize the reliable routing technology include integrating it into the existing Internet-based map services (such as Google/Yahoo Maps) and developing smart-phone-based applications.

Chapter 1

Introduction

1.1 Background

Travel time reliability is an important dimension of transportation service. Travel reliability enables travelers to make better use of their own time; shippers and freight carriers need predictable travel times to fulfill on-time arrival deliveries and other commitments in order to keep competitive. Abdel-Aty, Kitamura & Jovanis (1997) and Liu, Recker & Chen (2004) found that travel time reliability is one of the most important factors that affect route choice. Small, Noland, Chu & Lewis (1999) found that both travelers and freight carries tend to avoid schedule mismatching. However, urban transportation systems are affected by uncertainties of various sorts, ranging from disastrous events such as terrorist attacks or earthquakes to minor incidents such as bad weather conditions, highway maintenance or accidents. The U.S. Federal Highway Administration (FHWA) (2000) estimates that 50-60% of congestion delays are non-recurrent, and the percentage is even higher in smaller urban areas. Taken individually or in combination, these factors could adversely affect and perturb the quality of transportation services. Travel behavior researchers have revealed that the unexpected delay resulted from these disruptions produces far greater frustration among motorists than those expected ones (FHWA 2000). The lack of travel reliability information often encourages overly conservative risk-averse behavior or leads to uncomfortable, sometimes disastrous, disruptions. In order to help users hedge against uncertainty, it is important to develop new decision-supporting tools that are capable of making best use of

various sources of data to (1) reveal and document random patterns of travel times in highway networks and (2) provide real-time route guidance that accounts for uncertainty. The objective of this research is to deploy reliable algorithms and to explore the potential to commercialize these techniques. The current project is the second phase of this effort. The results from Phase I, which was completed in 2009, are briefly summarized as follows. The reader is referred to Nie, Wu, Nelson & Dillenburg (2009) for more details.

- We combine the GIS and traffic data from the Gary-Chicago-Milwaukee (GCM) traveler information system with travel planning data obtained from Chicago Metropolitan Agency for Planning (CMAP) to generate travel time distributions as the inputs for reliable route guidance models. Figure 1.1 shows the scope of the CMAP network. GCM traffic data include two parts: loop detector data (speed, volume and occupancy), and I-PASS transponder data (travel time between toll booths). In the Chicago area, the system has 827 loop detectors and 309 I-PASS OD pairs and has operated since October of 2004. In total, there are approximately 500 million records from 2004 - 2008 in the traffic database. A Windows console application named GCM-Gateway was developed to download these traffic data and convert them into a MYSQL database on a workstation at Northwestern University. With these data, the distributions of travel times were constructed on the freeway and toll road links. The travel time distributions on other roads were estimated from both observed traffic data and CMAP planning data, see Nie et al. (2009).

- The reliable routing algorithm was implemented using Visual C++ and tested on the the CMAP network. The experiments indicate that best paths do vary substantially with the reliability requirement, measured by the probability of arriving on-time or earlier. It is also shown that given the sheer size of the case study, the reported computational performance is satisfactory.

- Phase I develops a software tool named Chicago Test-bed for Reliable Routing (CTR), which provides an integrated environment to: 1) visualize and analyze traffic data, 2) construct

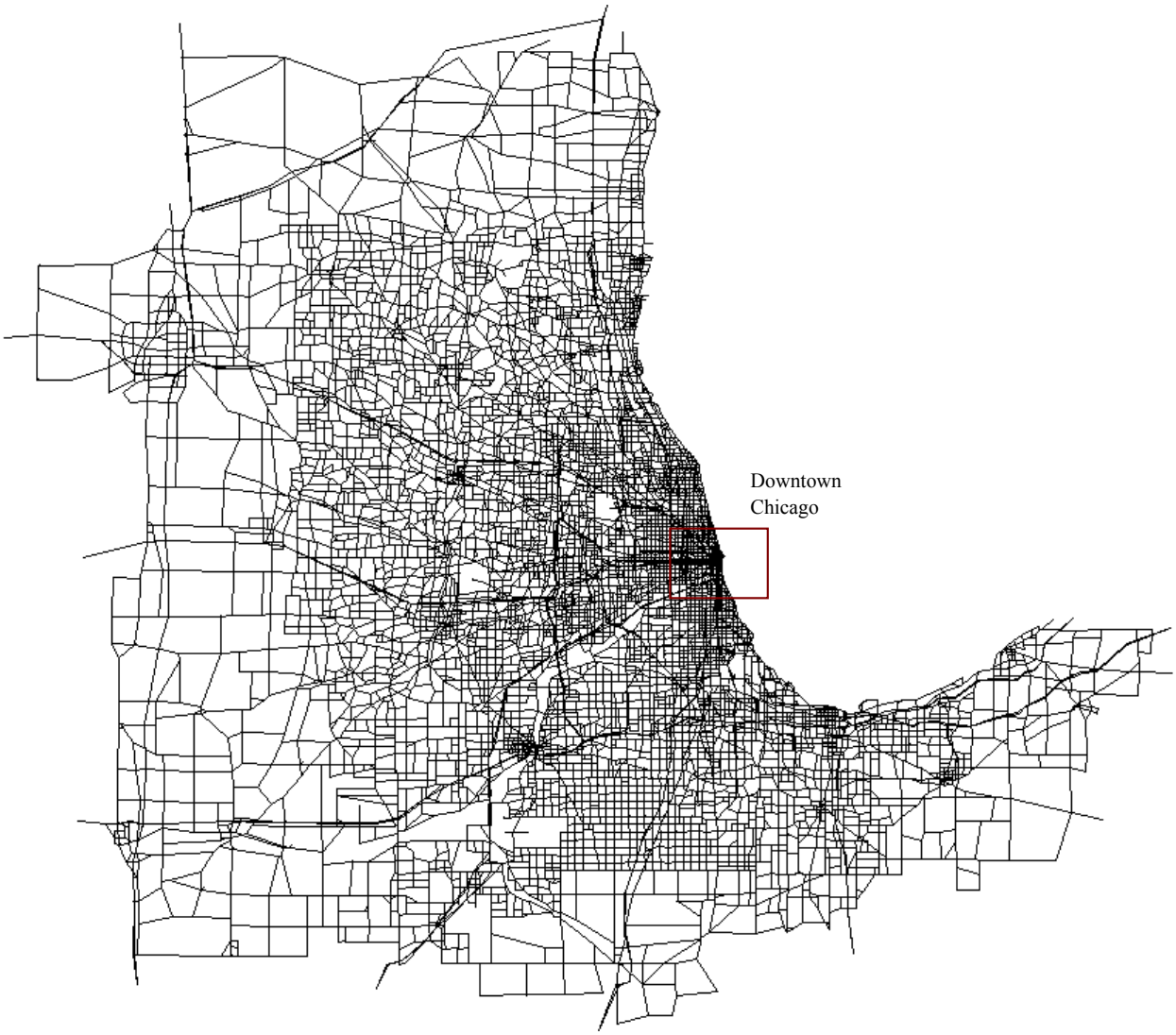


Figure 1.1: Topology of planning network from Chicago Metropolitan Planning Agency (CMAP)

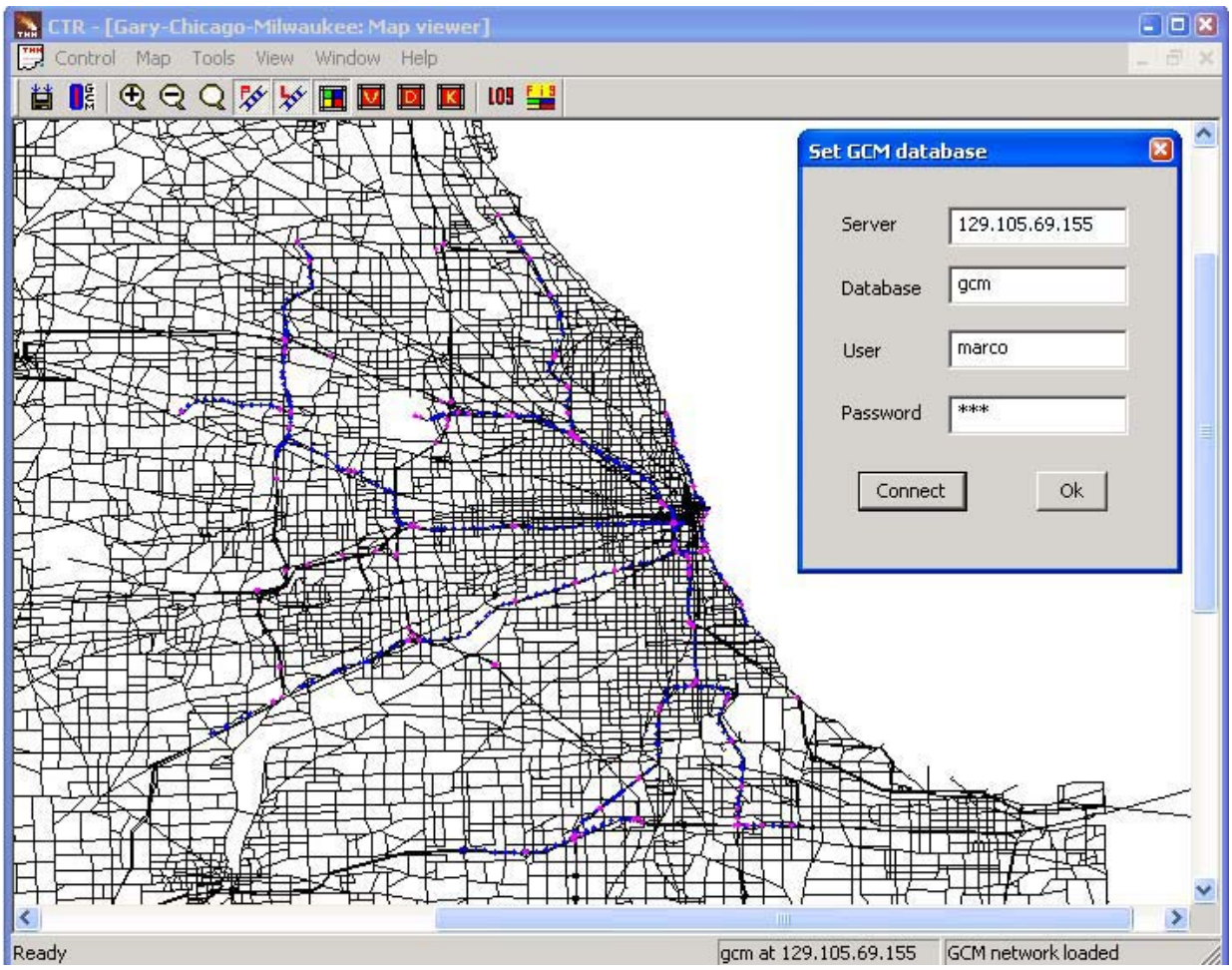


Figure 1.2: Main graphical user interface (GUI) of Chicago test-bed for reliable routing (CTR)

and display travel reliability measures for the Chicago network, and 3) provide reliable route guidance and compare it with conventional routing algorithms. Figure 1.2 shows the main GUI for CTR with the CMAP network loaded.

1.2 Research objectives

The first objective in Phase II is to improve the computational performance of the reliable routing algorithm in two aspects: the convolution method that produces path travel time distributions and the stochastic dominance rules adopted to generate non-dominated paths. In Phase I, we developed a convolution method based on an adaptive discretization scheme, which were

implemented and tested in the case study. Phase II examines a new convolution method based on Discrete Fourier Transform (DFT), which promises much better computational efficiency in theory. This new method is tested extensively and compared with the method developed in Phase I. We also implement and test the idea of using higher order stochastic dominance (SD) rules in reliable route generation. The higher-order SD rules promise better computational performance because the set of non-dominated paths corresponding to higher order SD rules are often much smaller. The overall performance of this approach, particulary regarding how well it trades off accuracy with efficiency, is evaluated with numerical experiments.

Our second objective is to explore the use of AVL bus data provided by Chicago Transit Authority (CTA) for travel time estimation on local roads. Recall that Phase I is focused on the data collected from loop detectors and I-PASS transponders on interstate highways and expressways (see the links marked as blue and pink in Figure 1.2). Since these sensors cover only a small portion of all network links, the travel time distributions on local roads have to be estimated using travel planning data through statistical methods. In Phase II, a sample of AVL bus data (one month) are extracted from the CTA bus database and the processed street speed data are mapped onto the CMAP network for further analysis. We then analyzed how these bus data represent real traffic conditions on different roads and discussed the potential to use them in reliable route guidance.

The third objective of this project is to collect and understand users' opinions about travel reliability and corresponding route guidance products. This is regarded as a necessary step to evaluating the potential value of reliable routing techniques as a commercial product. To this end, a web-based survey was conducted through uSamp Inc. and a sample of responses from more than two hundred motorists in the Chicago Metropolitan area was obtained. A preliminary analysis of the survey results is provided.

1.3 Organization

The rest of this report is organized as follows. Chapter 2 briefly reviews the literature. Chapter 3 revisits the reliable routing problem and its solution algorithms, and proposes techniques that aim at improving the computational performance. These solution techniques are tested and compared using numerical experiments in Chapter 4. Chapter 5 first discusses the methods used to extract travel time data from archived CTA AVL bus database and then analyzes the quality of these data. Chapter 6 describes the survey and analyzes the survey results. Chapter 7 concludes the study with a summary of findings.

Chapter 2

Literature Review

In this chapter, we briefly review the literature of the reliable routing models and the stochastic dominance theory, and introduce the notation necessary to describing them.

2.1 Notations

Consider a directed and connected network $G(\mathcal{N}, \mathcal{A}, \mathcal{P})$ consisting of a set of nodes $\mathcal{N}$ with the number of nodes $|\mathcal{N}| = n$, a set of links $\mathcal{A}$ with the number of links ($|\mathcal{A}| = m$), and a probability distribution $\mathcal{P}$ describing the statistics of link travel times. Let the destination of routing be node s. The travel time on any link ij (denoted as c_{ij}) is assumed to follow a random distribution with a probability density function $p_{ij}(\cdot)$ and $F_{ij}(\cdot)$ denotes the cumulative density function (CDF) of c_{ij}. For brevity, other notations used in the report are summarized in Table 2.1.

2.2 Stochastic routing problem

A routing problem concerned here aims to direct vehicles from an origin to a destination along a path that are considered "optimal" one way or another. Depending on whether or not the guidance is coordinated by a central control unit, the problem can be classified as "centralized" or "decentralized". They can also be labeled as "adaptive" or "*a prior*", according to whether or not en-route re-routing is allowed. Two other factors that are often used in classification are dynamics (i.e. travel time varies over time) and uncertainties (i.e. travel time is random). This

Table 2.1: Notations

Network:	
$\mathcal{A}$	set of links
$\mathcal{N}$	set of nodes
n	number of nodes
m	number of links
s	the destination nodes
K^{rs}	set of paths between r and s
k^{rs}	path k between r and s
c_{ij}	random link traversal time on link ij
$\Gamma_1^{rs}, \Gamma_2^{rs}, \Gamma_3^{rs}$	set of FSD / SSD / TSD-admissible paths between r and s
$\Omega_1^{rs}, \Omega_2^{rs}, \Omega_3^{rs}$	set of FSD / SSD / TSD-optimal paths between r and s
π_k^{rs}	random traversal time on path k^{rs}
δ_{ijk}^{rs}	1 if path k^{rs} uses link ij and 0 otherwise
$k^{js} \diamond ij$	extension of path k^{js} along link ij
q^{rs}	demand between OD pair rs
Probability:	
$p_{ij}(\cdot)$	probability density function for travel time on link ij
$P_{ij}(\cdot)$	probability mass function corresponding to $p_{ij}(\cdot)$
$P_{ij}^t(\cdot)$	probability mass function corresponding to the entry time t
$F_{ij}(\cdot)$	cumulative density function for travel time on link ij
$u_k^{rs}(b)$	maximum probability of arriving on time following path k^{rs} at a time budget b
$v_k^{rs}(\alpha)$	minimum travel time that ensures on-time arrival with a reliability α following path k^{rs}
$u^{rs}(b)$	maximum probability of arriving on time following an optimal routing policy for OD pair (r,s), given time budget b
$v^{rs}(\alpha)$	minimum travel time following an optimal routing policy for OD pair (r,s), given desired on-time arrival probability α
$u^{rs}(t,b)$	maximum probability of arriving on time following an optimal policy for OD pair (r,s), given time budget b, departing at time t
$\bar{k}^{is}(b)$	optimal path corresponding to $u^{is}(b)$
Algorithm:	
ϕ	length of time budget intervals
ϵ	accuracy of path comparison
L	number of time units in the analysis
φ	length of departure time intervals
T	number of departure time intervals in the analysis
Other:	
$U(\cdot)$	utility functions
$V(\cdot)$	disutility functions

Table 2.2: Various definitions of stochastic optimal paths

	LET-based	Reliability-based
Adaptive	Cheung (1998),Fu (2001) Miller-Hooks (2001),Provan (2003) Waller & Ziliaskopoulos (2002) Polychronopoulos & Tsitsiklis (1996) Gao & Chabini (2006)	Bander & White (2002) Fan, Kalaba & Moore (2005*a*)
A priori	Hall (1986),Fu & Rilett (1998) Miller-Hooks & Mahmassani (2000) Fan et al. (2005*b*)	Frank (1969), Sigal, Alan, Pritsker & Solberg (1980) Loui (1983), Sivakumar & Batta (1994) Yu & Yang (1998) Montemanni & Gambardella (2004) Bard & Bennett (1991), Miller-Hooks (1997) Miller-Hooks & Mahmassani (2003) Miller-Hooks & Mahmassani (1998)

research considers decentralized, *a prior* routing problem on stochastic networks [1]. The focus is to incorporate travel reliability as an integrated objective of routing.

When uncertainties are concerned, "optimal" routing, either adaptive or *a priori,* has many different definitions. A classic one considers a routing strategy optimal if it incurs the least expected travel time (LET) (Hall 1986, Polychronopoulos & Tsitsiklis 1996, Fu & Rilett 1998, Cheung 1998, Miller-Hooks & Mahmassani 2000, Miller-Hooks 2001, Fu 2001, Waller & Ziliaskopoulos 2002, Provan 2003, Gao & Chabini 2006, Fan, Kalaba & Moore 2005*b*). Clearly, the LET path may not properly weigh in travel time reliability since it overlooks travel time variances. This concern gives rise to the reliability-based routing problems.

Table 2.2 classifies stochastic routing problems in four categories, using two criteria. Our focus is the right bottom cell, i.e., reliability-based *a priori* routing problem, which is further discussed in what follows.

Frank (1969) defined the optimal path as the one that maximizes the probability of the travel

[1]While the routing model studied herein can address time-varying stochasticity, dynamics is not the focus of this project.

time equal to or less than a given threshold t_0, i.e.,

$$\overline{k}^{rs} = \arg\left(\max_{\forall k^{rs} \in K^{rs}} \{P(\pi_k^{rs} \leq t_0)\}\right) \tag{2.1}$$

Note that the distributions of all link travel times are continuous. Let $\mathcal{C} = (c_{12}, \cdots, c_{ij}, \cdots)$ and $\Pi^{rs} = (\pi_1^{rs}, \cdots, \pi_k^{rs}, \cdots)$ be vectors of link travel times and travel times of paths between an original-destination (OD) pair rs, respectively. Since the joint distribution of $\mathcal{C}$ is assumed to be given, the joint distribution of Π^{rs} can be obtained using the characteristic functions of $\mathcal{C}$ and Π^{rs}, and then the optimal path can be identified based on (2.1). However, this method is not practicable because it requires both multi-dimensional integration and path enumeration.

Mirchandani (1976) presented a recursive algorithm to solve a discrete version of Equation (2.1). However, the algorithm is suitable only for small problems since it requires to enumerate not only all paths but also all travel time possibilities for each path through a network expansion.

Sigal et al. (1980) suggested using the probability of being the shortest path as an index to define the optimality. For path l, the optimality index R_l is defined as:

$$R_l = P(\pi_l \leq \pi_k), \forall k \neq l \tag{2.2}$$

To calculate R_l, a multi-dimensional integral has to be evaluated. To evaluate the integral, it is necessary to obtain the set of links that are used by more than one path, and the joint distribution of these links. Determining such a set too requires enumerating all paths.

The expected utility theory (von Neumann & Morgenstern 1967) has also been used to define path optimality. Loui (1983) showed that the Bellman's principle of optimality can be used to find optimal path when utility functions are affine or exponential. This restriction was also noticed in Eiger, Mirchandani & Soroush (1985). For a general polynomial and monotonic utility function, Loui's expected-utility problem can be reduced to a bi-criterion (mean and variance) shortest path problem. In effect, a traveler is allowed to tradeoff the expected value and variance using generalized dynamic program (GDP) (see, e.g., Carraway, Morin & Moskowitz 1990) based on Pareto optimality (or non-dominance relationship). More general nonlinear utility functions

may be approximated by piecewise linear functions (see Murthy & Sarkar 1996, Murthy & Sarkar 1998). They (Murthy & Sarkar 1996, Murthy & Sarkar 1998) also proposed a few efficient solution procedures based on relaxation. The mean-variance tradeoff has been treated in other ways. For example, Sivakumar & Batta (1994) added an extra constraint into the shortest path problem to ensure that the identified LET paths have a variance smaller than a benchmark. In Sen, Pillai, Joshi & Rathi (2001), the objective function of stochastic routing is a parametric linear combination of mean and variance. In either case, GDP cannot be applied. Instead, nonlinear programming solution techniques must be used.

Stochastic routing has also been discussed in the context of robust optimization, that is, a path is optimal if its worst-case travel time is the minimum (Yu & Yang 1998, Montemanni & Gambardella 2004, Bertsimas & Sim 2003). Depending on the setting, such robust routing problems are either NP-hard (Yu & Yang 1998, Montemanni & Gambardella 2004) or solvable in polynomial time (Bertsimas & Sim 2003).

Bard & Bennett (1991) defined the optimal path as the one that maximizes the expected utility in a stochastic acyclic network. Compared with the study of Loui (1983) where utility functions have to be polynomial and monotonic, Bard & Bennett (1991) only required that the utility functions to be non-linear and monotonic. In order to find the global optimal path, all paths have to be enumerated. To improve the computational efficiency, they attempted to reduce the network size by using the theory of the first order stochastic dominance (see Section 2.3.2 for details). Specifically, if a path is dominated, the links used by this path are elected for possible elimination. In their paper, the first order stochastic dominance restraint is relaxed as ”post median stochastic dominance”, i.e., only the points on the tail of CDF (after the median) are checked, in order to eliminate more paths. It is shown that through reduction, 90% of paths in an acyclic network are eliminated, which makes path enumeration practicably feasible. This approach, however, is only applicable on acyclic networks.

Miller-Hooks (1997) and Miller-Hooks & Mahmassani (2003) also employed the first order

stochastic dominance to define path optimality. Besides the first order stochastic dominance, they also defined two types of path dominance: (1) deterministic dominance, and (2) expected value dominance. Label-correcting algorithms were proposed to find non-dominant paths under the path dominance rules. Recognizing that the exact algorithm does not have a polynomial bound, heuristics are considered in Miller-Hooks (1997) which attempt to limit the size of the retained non-dominant paths by a predetermined number. As noted in Miller-Hooks (1997), however, these heuristics may not identify any non-dominant paths.

Reliability has also been defined using the concept of connectivity (Chen, Bell, Wang & Bogenberger 2006, Kaparias, Bell, Chen & Bogenberger 2007). This approach models reliability as the probability that the travel time on a link is greater than a threshold. Accordingly, the reliability on a path is the product of the reliability of links used by this path (assuming independent distributions). A software tool known as ICNavS was developed based on this approach (Kaparias et al. 2007).

2.3 Stochastic dominance

2.3.1 Classic theory of stochastic dominance

The stochastic dominance (SD) theory has been extensively used in finance and economics to rank random variable when their distributions are known (Hanoch & Levy 1969, Hadar & Russell 1971, Rothschild & Stiglitz 1970, Whitmore 1970, Bawa, Bondurtham, Rao & Suri 1983, Muller & Stoyan 2002, Dentcheva & Ruszczynski 2003). This concept has been exploited to study the stochastic routing problems by several authors (Bard & Bennett 1991, Miller-Hooks 1997, Miller-Hooks & Mahmassani 2003, Nie & Wu 2009). We now formally introduce the SD theory in the following.

Definition 2.1 (FSD $\succeq^1$) *A random variable X dominates another random variable Y in the first order, denoted as $X \succ^1 Y$ if $F_X(t) \leq F_Y(t)$, $\forall t$ and $F_X(t) < F_Y(t)$ for some t, where F_X is the cumulative density function (CDF) of random variable X.*

Definition 2.2 (SSD $\succeq^2$) *A random variable X dominates another random variable Y in the second order, denoted as $X \succ^2 Y$ if $\int_{-\infty}^{t} F_X(w)dw \leq \int_{-\infty}^{t} F_Y(w)dw, \forall t$ and $\int_{-\infty}^{t} F_X(w)dw < \int_{-\infty}^{t} F_Y(w)dw$ for some t.*

Let $F_X^{(2)}(t) = \int_{-\infty}^{t} F_X(w)dw$, and the third-order stochastic dominance can be defined as follows.

Definition 2.3 (TSD $\succeq^3$) *A random variable X dominates another random variable Y in the third order, denoted as $X \succ^3 Y$ if $\int_{-\infty}^{t} F_X^{(2)}(w)dw \leq \int_{-\infty}^{t} F_Y^{(2)}(w)dw, \forall t$ and $\int_{-\infty}^{t} F_X^{(2)}(w)dw < \int_{-\infty}^{t} F_Y^{(2)}(w)dw$ for some t.*

According to the random utility theory, random variable X is preferred to Y implies that X has a higher expected utility. The stochastic dominance theory not only provides a tool to rank random variables, but also explains the ranking within the framework of utility theory, corresponding to utility function $U(\cdot)$ (see e.g. Levy & Hanoch 1970)

Theorem 2.1 *$X \succeq^1 Y$ if and only if $E[U(X)] > E[U(Y)]$, for any nondecreasing utility function $U(\cdot)$, i.e., $U' \geq 0$.*

Theorem 2.2 *$X \succeq^2 Y$ if and only if $E[U(X)] > E[U(Y)]$, for any nondecreasing and concave utility function $U(\cdot)$, i.e, $U' \geq 0$ and $U'' \leq 0$.*

Theorem 2.3 *$X \succeq^3 Y$ if and only if $E[U(X)] > E[U(Y)]$, for any utility functions $U(\cdot)$ such that $U' \geq 0, U'' \leq 0$ and $U''' \geq 0$.*

The reader is referred to Bawa (1975) and Heyer (2001) for proofs of the above classical results.

Theorem 2.1 implies that any *nonsatiable* decision maker *who is always better off with more quantities of interest* ($U' > 0$), prefers X to Y if $X \succ^1 Y$. Theorem 2.2 is for non-satiable and **risk-averse** decision makers, because any concave utility function ($U'' < 0$) implies risk-aversion (Friedman & Savage 1948). Theorem 2.3 states that the third order stochastic dominance corresponds to

ruin-aversion (Heyer 2001, Ullrich 2009). According to Heyer (2001) and Ullrich (2009), ruin-averse decision makers are "willing to accept a small, almost certain loss in exchange for the remote possibility of large returns", and on the other hand, "unwilling to accept a small, almost certain gain in exchange for the remote possibility of ruin". It is shown that ruin aversion is captured by positively skewed probability density functions.

2.3.2 Stochastic dominance for decreasing utility functions

Definitions 2.1, 2.2 and 2.3 are all based on the circumstance that decision-makers are never worse off off with more quantities of the random variable of interest. That is, their utility function is non-decreasing with respect to the quantity. However, as travel time is concerned, travelers usually prefer shorter travel times to longer ones. That is, if the utility depends on travel time, the utility function is decreasing, i.e., $U' < 0$. In this case, the stochastic dominance has to be re-defined.

Definition 2.4 (FSD for decreasing utility functions $\succ_1$) *A random variable X dominates another random variable Y in the first order under decreasing utility functions, denoted as $X \succ_1 Y$, if $F_X(t) \geq F_Y(t), \forall t$ and $F_X(t) > F_Y(t)$ for some t.*

When stochastic dominance is defined in Definition 2.4, Theorem 2.1 becomes

Theorem 2.4 *$X \succ_1 Y$ in the first order if and only if $E[U(X)] > E[U(Y)]$, for any decreasing utility function $U(\cdot)$, i.e., $U' < 0$.*

We shall address the second and third order stochastic dominance for decreasing utility functions in Chapter 3.

Chapter 3

Formulation and Solution Algorithm

In this chapter, we introduce the underlying mathematical formulation and solution algorithm for reliable route guidance. The focus is to discuss the two proposed improvements in the solution algorithm: the convolution method based on discrete Fourier transform and the approximation based on higher-order stochastic dominance.

3.1 Formulation

The reliable *a priori* shortest path (RASP) problem aims to find, starting from any node $i \neq s$, *a priori* path which requires least time budget to ensure a specified probability of arriving at the destination s on time or earlier. Nie and Wu (2009) showed that this problem is equivalent to finding all non-dominated paths under the first-order stochastic dominance (FSD), cf. Definition 2.4. The non-dominated paths under FSD, called FSD-admissible, is defined as follows.

Definition 3.1 (FSD-admissible path) *A path l^{rs} is FSD-admissible if $\exists$ no path k^{rs} such that $\pi_k^{rs} \succ_1 \pi_l^{rs}$.*

Denote $\Gamma_{\mathrm{FSD}}^{rs}$ as the FSD-admissible path set. Accordingly, the FSD Pareto-optimal frontier can be identified from the admissible paths:

$$u_{\mathrm{FSD}}^{rs}(b) = \max\left\{u_k^{rs}(b), \forall k^{rs} \in \Gamma_{\mathrm{FSD}}^{rs}\right\}, \forall b \geq 0 \tag{3.1}$$

The Pareto-optimal frontier guarantees optimality, that is, for any on-time arrival probability, the path that gives the least time budget (i.e., the reliable shortest path) must be an admissible

path. Once the Pareto-optimal frontier is known, an optimal path $\bar{k}^{rs}$ can be identified such that $u_k^{rs}(b) = u_{\text{FSD}}^{rs}(b)$. The Pareto-optimal frontier can be also represented by the inverse CDF, as follows

$$v_{\text{FSD}}^{rs}(\alpha) = \min\left\{v_k^{rs}(\alpha), \forall k^{rs} \in \Gamma_{\text{FSD}}^{rs}\right\}, \forall 0 \le \alpha \le 1 \tag{3.2}$$

In Nie & Wu (2009), the RASP problem is formulated as a general dynamic programming (GDP) problem:

$$\begin{aligned}&\text{Find } \Gamma_{\text{FSD}}^{is}\ \forall i \in \mathcal{N} \text{ such that}\\ &\Gamma_{\text{FSD}}^{is} = \gamma_{\succ_1}(k^{is} = k^{js} \diamondsuit ij \,|k^{js} \in \Gamma_{\text{FSD}}^{js}, \quad \forall ij \in \mathcal{A}), \forall i \ne s; \Gamma_{\text{FSD}}^{ss} = \{0^{ss}\}\end{aligned} \tag{3.3}$$

where $k^{js} \diamond ij$ extends path k^{js} along link ij (therefore, the distribution of the travel time on path k^{js} is generated by convolving the distributions of the travel times on path k^{js} and link ij); $\gamma_{\succ_1}(K)$ represents the operation which retrieves FSD-admissible paths from a set K using Definition 3.1; 0^{ss} is a dummy path representing the boundary condition. Problem (3.3) can be solved using label correcting algorithms such as proposed in Miller-Hooks (1997) and our report for Phase I (Nie et al. 2009). For brevity we do not reiterate the structure of the algorithm here. The reader is referred to the above references.

3.2 Direct discrete convolution

A key component in the RASP algorithm is the calculation of the path travel time distributions u_k^{rs} by convolving the travel travel time distributions of its member links. If the link travel time distribution is continuous, the path travel time distribution can be calculated recursively from convolution as follows:

$$u_k^{is}(b) = \int_0^b u_k^{js}(b-w)p_{ij}(w)dw, \ \forall b \in [0, T], \tag{3.4}$$

where T is the maximum possible travel time in the network, path k^{is} is constructed by extending path k^{js} along link ij to node i, i.e., $k^{is} = k^{js} \diamond ij$.

In most cases, it is more convenient to approximate the continuous distributions with their discrete counterparts. Let T be divided into L uniform discrete intervals with $L\phi = T$, then the probability mass of interval l is calculated as

$$P_{ij}(l) = \int_{(l-1)\phi}^{l\phi} p_{ij}(w)dw,\ l = 1, ..., L. \tag{3.5}$$

Accordingly, Equation (3.4) can be discretized as

$$U_k^{is}(l) = \sum_{l'=1}^{l} U_k^{js}(l - l')P_{ij}(l'),\ l = 1, \cdots, L. \tag{3.6}$$

where $U_k^{is}(l)$ denote the cumulative probability on path $k \in K^{is}$ up to the lth discrete interval. Intuitively, the quality of convolution results is controlled by the resolution of the discretization L: a larger L generally leads to higher accuracy.

While straightforward, the discrete convolution of (3.6) is not satisfactory for a number of reasons, of which the most important is probably the requirement for identical T and L for all distributions [1]. Wu & Nie (2009) proposed to divide the support such that each discrete interval has the same predetermined probability mass in order to avoid estimating T a priori. Since the discrete interval of different random variables may not be of uniform length, a different method was proposed to perform convolution. Nevertheless, neither (3.6) nor the method proposed in Wu & Nie (2009) recognizes the possible unevenly concentrations of probability mass. As a result, they could over-represent the flat portions on the probability density function, while under-representing sharp changes. In Phase I, Nie et al. (2009) designed an adaptive discretization approach (ADA) that precisely addresses this issue. For expository convenience, we first review this method in what follows.

ADA starts by dividing the support of each random variable into L intervals with a uniform length ϕ, and then computes the probability mass functions using Equation (3.5). The key difference here, comparing to (3.6), is that ϕ may vary from one random variable to another, depending

[1] Note that the maximum possible travel time in a stochastic network is not readily available. Even if it is, such a range is likely to be too large to represent individual distributions efficiently. See Nie et al. (2009) for more discussions.

on the range of the support [2]. Moreover, a procedure termed *Consolidation* is introduced to merge consecutive intervals together such that no more than one interval has a probability mass smaller than $1/L$. "Merging" two consecutive discrete intervals means removing the boundary between them and assigning the sum of their probability masses to the new interval. Consolidation produces a set of effective support intervals (ESI), whose size is often much smaller than the initial discretization resolution L.

Consider two random variables X and Y, which, after discretization and consolidation, can be represented by a set of discrete support points, W_x and W_y, and associated probability mass vectors, Q_x and Q_y. Let the number of effective support points for X and Y be L_x and L_y, respectively. We have

$$W_x = [w_1^x, ..., w_{L_x}^x], W_y = [w_1^y, ..., w_{L_y}^y], Q_x = [q_1^x, ..., q_{L_x}^x], Q_y = [q_1^y, ..., q_{L_y}^y]$$

The following procedure can be used to convolve X and Y, i.e, to obtain $Z = X \otimes Y$.

ADA-based Direct Convolution Scheme

Inputs: W_x, W_y, Q_x, Q_y.

Outputs: W_z, Q_z, i.e., the vectors of discrete support points and probability mass for $Z = X \otimes Y$.

Step 0 Obtain the range of support for Z. Set $b_0^z = 1.5(w_1^x + w_1^y) - 0.5(w_2^x + w_2^y), b_L^z = 1.5(w_{L_x}^x + w_{L_y}^y) - 0.5(w_{L_x-1}^x + w_{L_y-1}^y)$. Divide $[b_0^z, b_L^z]$ into L intervals of uniform length, and compute $\phi = (b_L^z - b_0^z)/L$. Initialize $p_l^z = 0, \forall l = 1, \cdots, L$.

Step 1 for $i = 1, 2, \cdots, L_x$,

for $j = 1, 2, \cdots, L_y$,

Calculate $ts = w_i^x + w_j^y$ and $tp = q_i^x q_j^y$. Define $l = \left[\frac{ts - b_0^z}{\phi}\right]_-$, and set $p_l^z = p_l^z + tp$

end for

[2] Discrete random variables often have a well-defined, finite support range. For continuously distributed random variables that have infinite support ranges, the upper (lower) bound of the support can be taken at a $100(1-\epsilon)$ (ϵ) percentile value, where ϵ is a small positive number selected by the modeler. The value used in this project is 0.001.

end for

Step 2 Call Procedure Consolidation to obtain W_z, Q_z with inputs $\{b_0^z, b_1^z, ..., b_L^z\}, \{p_1^z, ..., p_L^z\}$.

Consolidation

Inputs: Vectors of initial discrete intervals and probability $\{b_0, b_1, ..., b_L\}, \{p_1, ..., p_L\}$.

Outputs: $W = \{w_1, ..., w_{L_0}\}, Q = \{q_1, ..., q_{L_0}\}$, where L_0 is the number of effective support points.

Step 0 Initialization. Set $l = 1, i = 1, \phi = 1/L$. Set $tp = 0, ts = 0$.

Step 1 if $i > L$; stop, set $L_0 = l$; otherwise, $tp = p_i, ts = 0.5(b_{i-1} + b_i)p_i$, go to Step 2.

Step 2 If $tp > \phi$, $q_l = tp, w_l = ts/tp$, set $i = i + 1, l = l + 1$, go to Step 1; otherwise, go to Step 3.

Step 3 Set $i = i + 1, tp = tp + p_i; ts = ts + 0.5(b_{i-1} + b_i)p_i$, go to Step 2.

Nie et al. (2009) observed that consolidation reduces the number of support points required to represent a distribution by a factor of 3, while producing PDF and CDF very similar to those of the original distribution (represented by L support points). While this observation indicates that consolidation may bring out significant computational benefits without sacrificing much numerical accuracy, it is unclear how the small errors introduced by consolidation would accumulate during the course of convolutions. Numerical experiments presented later will address this question.

As pointed out in the report for Phase I (Nie et al. 2009) , the main shortcoming of the ADA-based direct convolution scheme is its quadratic complexity. We next examine the convolution method based on the discrete Fourier transform, which provides a much better theoretical complexity.

3.3 Convolution based on discrete Fourier transform

The Fourier transform is often used to convert a complex-valued function from the time domain to the frequency domain (Bracewell 2000). Specifically, let $\hat{f}(s)$ denote the Fourier transform

of function $f(x)$, then

$$\hat{f}(s) = \int_{-\infty}^{\infty} f(t)e^{-2\pi its}dt. \tag{3.7}$$

The Fourier transform simplifies convolution. That is, if $h(t)$ is the convolution of $f(t)$ and $g(t)$, written as $f(t) \otimes g(t)$, then

$$\hat{h}(s) = \hat{f}(s) \cdot \hat{g}(s) \tag{3.8}$$

Note that i in Equation (3.7) is the imaginary unit, i.e., $i = \sqrt{-1}$. In the discrete case, the sequence of L numbers $[p_1, ..., p_L]$ (in the time domain) can be transformed into the sequence of complex numbers $[\hat{p}_1, ..., \hat{p}_L]$ (in the frequency domain) using the following formula:

$$\hat{p}_k = \sum_{l=1}^{L} p_l \exp\left(\frac{-2\pi i}{L}(k-1)(l-1)\right), k = 1, ..., L \tag{3.9}$$

Equation (3.9) is called *discrete Fourier transform* (DFT). On the other hand, the *inverse discrete Fourier transform* (IDFT) reads

$$p_l = \sum_{k=1}^{L} \hat{p}_k \exp\left(\frac{2\pi i}{L}(k-1)(l-1)\right), l = 1, ..., L \tag{3.10}$$

Since DFT can be computed in time $O(L \log L)$ using the *fast Fourier transform* (FFT) algorithm (Cormen, Leiserson, Rivest & Stein 2001), the convolution based on DFT has a theoretical complexity of $O(L \log L)$ (it involves one FFT, one inverse FFT (IFFT), and a point-wise multiplication). Note that to apply FFT in convolution requires that the discrete intervals of both distributions have the same length. The following procedure describes how to convolve two distributions through FFT.

Procedure: **Basic DFT-based Convolution**

Inputs: Support point vectors $W_x = \left(w_1^x, w_1^x + \phi, w_1^x + 2\phi, \cdots, w_1^x + (L_x - 1)\phi\right)$ and $W_y = \left(w_1^y, w_1^y + \phi, w_1^y + \phi, \cdots, w_1^y + (L_y - 1)\phi\right)$, and the corresponding probability mass vectors Q_x and Q_y.

Outputs: W_z and Q_z for $Z = X \otimes Y$.

Step 0 Find an integer m such that $2^m \geq L_x + L_y - 1$ and $2^{m-1} < L_x + L_y - 1$.

Step 1 Extend Q_x and Q_y so that each has 2^m elements. All newly added elements are zeros.

Step 2 Transform Q_x and Q_y to $\hat{Q}_x$ and $\hat{Q}_y$, respectively, using Equation (3.9).

Step 3 Perform a point-wise multiplication of $\hat{Q}_x$ and $\hat{Q}_y$ to get $\hat{Q}_z$.

Step 4 Transform $\hat{Q}_z$ to Q_z, using Equation (3.10).

Step 5 Truncate Q_z so that it only keeps the first $L_x + L_y - 1$ elements. Let $L_z = L_x + L_y - 1$. Then, corresponding to Q_z, the discrete support points are $W_z = \left(w_1^x + w_1^y, w_1^x + w_1^y + \phi, w_1^x + w_1^y + 2\phi, \cdots, w_1^x + w_1^y + (L_z - 1)\phi\right)$.

The Cooley-Tukey algorithm is employed to conduct FFT and IFFT (Cooley & Tukey 1965) in Steps 2 and 4. More specifically, a radix-2 decimation-in-time algorithm is implemented. Note that Q_x and Q_y have to be extended to 2^m in Step 0 because the algorithm requires the length of the probability mass vector to be a power of two. It is also worth noting that the DFT-based convolution requires both distributions have discrete intervals of equal length (i.e., ϕ). If the input discrete distributions do not satisfy this condition, they have to be re-adjusted according to an identical ϕ.

3.4 Convolution of multiple random variables

3.4.1 All-at-once versus one-at-a-time

When convolution involves $n > 2$ distributions, the ideal choice is taking an all-at-once scheme (Ng & Waller 2010). That is, all distributions are transformed to the frequency domain and convolved all-at-once in that domain (through pointwise multiplication) until the resulting distribution is transformed back to the time domain. This scheme only requires n FFT and one IFFT operations. However, it requires that the probability mass vectors of all member links be extended to the same length. The details of the procedure are described below.

Procedure: All-at-once DFT-based convolution (for multiple random variables)

Inputs: Random variables $X_1, X_2, \cdots, X_n$, whose vectors of support points are $W_1, W_2, \cdots, W_n$, with the corresponding probability mass vectors $Q_1, Q_2, \cdots, Q_n$, respectively. The length of W_k is $L_k, k = 1, 2, \cdots, n$.

Outputs: Vectors of support points and probability mass for Z, W_z and Q_z respectively, where $Z = X_1 \otimes X_2 \otimes \cdots \otimes X_n$.

Step 0 Set the total number of the support points of Z as $\sum_{i=1}^{n} L_i - n + 1$.

Step 1 Find an integer m such that $2^m \geq \sum_{i=1}^{n} L_i - n + 1$ and $2^{m-1} < \sum_{i=1}^{n} L_i - n + 1$.

Step 2 Extend the lengths of $Q_1, Q_2, \cdots, Q_n$ to 2^m by adding zeros at the end of vectors.

Step 3 Transform $Q_1, Q_2, \cdots, Q_n$ to $\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_n$ using Equation (3.9).

step 4 Compute the convolution of distributions in the frequency domain: $\hat{Q}_z = \prod_{i=1}^{n} \hat{Q}_i$.

step 5 Transform $\hat{Q}'_z$ to Q_z using Equation (3.10). .

step 6 Truncate Q_z such that it only keeps the first $\sum_{i=1}^{n} L_i - n + 1$ elements. The discrete support points are $W_z = \left(\sum_{i=1}^{n} w^i, \sum_{i=1}^{n} w^i + \phi, \sum_{i=1}^{n} w^i + 2\phi, \cdots, \sum_{i=1}^{n} w^i + (L_z - 1)\phi\right)$, where $L_z = \sum_{i=1}^{n} L_i - n + 1$.

A main problem associated with the above convolution scheme is that it cannot be directly applied in the label-correcting algorithm for the RASP problem. For one thing, because the probability mass vector of each link has to be extended to a specific length which depends on the total number of support points of all member links, the method implicitly requires generating an entire path before calculating its distribution. However, since paths and their travel time distributions are generated concurrently in a label-correcting algorithm, not all member links of a path are known until the algorithm is terminated. Moreover, the all-at-once scheme makes it

very difficult to identify dominated paths, as the stochastic dominance is defined in the time-domain. Without the ability of eliminating dominated sub-paths, the label-correcting algorithm basically equals path enumeration, which is computationally prohibitive for large problems.

The above problem can be resolved using a one-at-a-time convolution scheme. The main idea is to perform one convolution each time in the frequency domain, and then to transform the intermediate result back to the time domain so that the dominance relationship can be properly checked. The procedure is given as follows (note that the inputs and outputs are ignored because they are the same to the all-at-once scheme).

Procedure: One-at-a-time DFT-based convolution (for multiple random variables)

Step 0 Set $Q_z = Q_1, W_z = W_1$.

Step 1 For $i = 2$ to n, call Basic DFT-based Convolution with inputs W_z, W_i, Q_z, Q_i to obtain W'_z, Q'_z. Then update $W_z = W'_z, Q_z = Q'_z$.

Note that the one-at-a-time scheme requires approximately $2n$ FFT and n IFFT operations (since each convolution involves two FFT and one IFFT operations). This triples the amount of total FFT operations compared to the all-at-once scheme. However, the good news is that most convolutions would operate on shorter vectors. Suppose that each distribution has L support points. Conducing FFT for each link requires approximately $O(nL\log(nL))$ steps in the all-at-once scheme, where nL is an approximated length of each probability mass vector. In total, the number of the required steps is in the order of $(n^2L\log(nL) + n(n-1)L) \simeq (n^2L\log(nL)$ [3]. For the one-at-a-time scheme, the number of the required steps is in the order of

$$\begin{aligned} &3\times 2L\log(2L) + 2L + 3\times 3L\log(3L) + 3L + \cdots + 3\times nL\log(nL) + nL \\ \leq &1.5n(n-1)L\log(nL) + \frac{n(n-1)L}{2} \end{aligned}$$

Note that $iL\log(iL)$ is the steps required by one FFT (or IFFT) in the $(i-1)$th convolution, and each convolution in the one-at-a-time scheme requires two FFT and one IFFT operations. It can

[3] The first term in the parenthesis is the cost of FFT and IFFT and the second accounts for point-wise multiplication.

be seen from the above analysis that the one-at-a-time scheme has a complexity comparable with the all-at-once scheme thanks to the savings from operating on shorter vectors. Also, this saving is more significant when n is larger, because the upper bound $1.5n(n-1)L\log(nL) + \frac{n(n-1)L}{2}$ becomes looser. More importantly, the one-at-a-time scheme can be easily integrated with the label-correcting algorithm because the intermediate convolution results are always available in the time domain for the dominance check. Furthermore, the one-at-a-time scheme makes it possible to reduce the number of ineffective support points generated in convolution, which could further improve the computation efficiency. To the elaboration of the last point we now turn.

3.4.2 Ineffective support points in DFT-based convolution

We first highlight the issue of ineffective support points in the DFT-based convolution through a numerical example. It is worth reiterating that this problem can only be addressed if a one-at-a-time scheme is employed. Consider four paths which consist of, respectively, 2, 3, 4 and 14 links. All link travel times follow the Gamma distribution, with the same parameter setting as in Section 4 of Wu & Nie (2009). The length of discrete travel time interval ϕ is set to 0.1. The path travel time distributions are obtained using the one-at-a-time convolution scheme. Table 3.1 shows the numbers of support points of these distributions corresponding to different cumulative probability ranges. Take the 2-link path as an example. In the range between 0.99 to 1 located 293 discrete points, which means that these support points together contributes to about 1% of all probability masses. Storing and manipulating those 293 discrete support points are clearly ineffective from the computational point of view because they contribute little to shaping the distribution. A less obvious yet more severe problem created by these ineffective support points is that their presence may generate many non-dominated paths which would be otherwise dominated. These “extra” non-dominated paths could bring about significantly higher memory requirements and computational overhead, while making little difference on the Pareto optimality frontier (and hence on route choice). We note that these problems can become still worse as

Table 3.1: Number of support points used to present the travel time distributions of four paths in different intervals of cumulative probability α

	$0.01 \leq \alpha \leq 0.99$	$0 \leq \alpha < 0.01$	$0.99 < \alpha < 1$	$0.999 < \alpha < 1$
2-link	113	12	293	247
3-link	113	34	728	539
4-link	324	88	1049	935
14-link	396	323	3089	2977

the number of links increases. For the 14-link path, 98% of all probability mass are contributed by less than 9% of the support points. On the other hand, more than 80% of all support points are located between 0.99 and 1.

In light of the above observations, we propose to re-adjust the discrete distribution generated from the basic DFT-based convolution by removing the ineffective support points. A support point is labeled as *ineffective* if it is located in the interval of $\alpha < \epsilon$ or $\alpha > 1 - \epsilon$, where α is the cumulative probability and ϵ is a pre-determined accuracy index. Note that ϵ controls how many support points would be labeled as "ineffective" and therefore being excluded from the vector of support points. Thus, the larger the index is, the more support points would be removed, and larger approximation errors would be introduced by the procedure. We describe in the following a revised DFT-basic convolution scheme which includes a post-process to remove the support points ineffective with respect to ϵ. For convenience, this procedure is called *approximated DFT-based convolution*.

Procedure: Approximated DFT-based convolution

Inputs: Inputs for the basic DFT-based convolution, and the accuracy index ϵ.

Outputs: Vector of effective support points $\tilde{W}_z$ and the corresponding probability mass vector $\tilde{Q}_z$.

Step 0 Call the basic DFT-based convolution to obtain the original probability mass vector Q_z and the corresponding support point vector W_z.

Step 1 Calculate the cumulative probabilities $F_z = (f_1^z, f_2^z, \cdots, f_{L_z}^z)$ based on Q_z.

Step 2 If $f_i^z < \epsilon$ and $f_{i+1}^z > \epsilon$, let $\tilde{q}_1^z = f_i^z$ and $\tilde{w}_1^z = w_i^z$; if $f_{j-1}^z < 1-\epsilon$ and $f_j^z > 1-\epsilon$, let $\tilde{q}_{j-i+1}^z = 1 - f_{j-1}^z$, and $\tilde{w}_{j-i+1}^z = w_j^z$. For any $i < l < j$, set $\tilde{q}_{l-i+1}^z = q_l^z$, and $\tilde{w}_{l-i+1}^z = w_l^z$. Set $\tilde{Q}_z = \{\tilde{q}_1^z, \cdots, \tilde{q}_{i-j+1}^z\}, \tilde{W}_z = \{\tilde{w}_1^z, \cdots, \tilde{w}_{i-j+1}^z\}$

The post-process in Step 2 simply concentrates ϵ probability mass on the first and last support points, which would change the range of the entire support, as well as the overall shape of the distribution. It is thus important to examine how the convolution results might be affected by the approximation errors. We address this question in the next chapter through numerical experiments.

3.5 Higher-order stochastic dominance

3.5.1 Definitions

In Chapter 2, we have reviewed the theory of stochastic dominance. Especially, we discussed the first order stochastic dominance when the utility function is non-increasing. According to the theory, the FSD rule is used by non-satiable travelers whose utility always decreases as travel time increases. Essentially, it can be shown that if $X \succ_1 Y$, then $\mathrm{E}[U(X)] > \mathrm{E}[U(Y)]$ for any utility function U such that $U' < 0$ (Theorem 2.4). Higher order stochastic dominance can be used to address decision-makers' risk-taking preference. Let us first define the second-order SD (SSD) and third-order SD (TSD) as follows.

Definition 3.2 (SSD for decreasing utility functions, $\succ_2$) *A random variable X dominates another random variable Y in the second order, denoted as $X \succ_2 Y$, if*

$$\int_t^T F_X(w)dw \geq \int_t^T F_Y(w)dw, \forall t \text{ and } \int_t^T F_X(w)dw > \int_t^T F_Y(w)dw \text{ for at least one } t$$

Definition 3.3 (TSD for decreasing utility functions, $\succ_3$) *A random variable X dominates another random variable Y in the third order, denoted as $X \succ_3 Y$, if*

$$\int_t^T F_X^{(2d)}(w)dw \geq \int_t^T F_Y^{(2d)}(w)dw, \forall t \text{ and } \int_t^T F_X^{(2d)}(w)dw > \int_t^T F_Y^{(2d)}(w)dw \text{ for at least one } t$$

where $F_X^{(2d)}(t) = \int_t^T F_X(w)dw$.

$X \succ_2 Y$ implies that $\mathrm{E}[U(X)] > \mathrm{E}[U(Y)]$ for any utility function U such that $U' < 0$ (decreasing) and $U'' < 0$ (concave); $X \succ_3 Y$ implies that $\mathrm{E}[U(X)] > \mathrm{E}[U(Y)]$ for any U such that $U' < 0, U'' < 0$ and $U''' < 0$. Note that the concave utility function (i.e. $U'' < 0$) can be interpreted as risk aversion in the sense that an individual with such a utility function would always prefer the expectation of a random variable to the random variable itself (Friedman & Savage 1948). On the other hand, $U''' < 0$ can be interpreted as ruin aversion. A ruin-averse individual would accept "a small, almost certain loss in exchange for the remote possibility of large returns" (Heyer 2001, Ullrich 2009).

Similarly, we can define SSD- and TSD-admissible paths as follows.

Definition 3.4 (SSD/TSD-admissible path) *A path* l^{rs} *is SSD/TSD-admissible if* $\exists$ *no path* k^{rs} *such that* $k^{rs} \succ_2 / \succ_3 l^{rs}$.

SSD- and TSD-admissible path set are denoted as $\Gamma^{rs}_{\mathrm{SSD}}$ and $\Gamma^{rs}_{\mathrm{TSD}}$ respectively, and the corresponding Pareto-optimal frontiers are

$$u^{rs}_{\mathrm{SSD}}(b) = \max\left\{u^{rs}_k(b), \forall k^{rs} \in \Gamma^{rs}_{\mathrm{SSD}}\right\}, \forall b \geq 0,\ u^{rs}_{\mathrm{TSD}}(b) = \max\left\{u^{rs}_k(b), \forall k^{rs} \in \Gamma^{rs}_{\mathrm{TSD}}\right\}, \forall b \geq 0 \tag{3.11}$$

Furthermore, the following relationship between the admissible path sets under different SD rules can be derived from the SD definitions.

Proposition 3.1 $\Gamma^{rs}_{TSD} \subseteq \Gamma^{rs}_{SSD} \subseteq \Gamma^{rs}_{FSD}$.

Proof: cf. Wu & Nie (tentatively accepted). □

To further clarify the above results, Figure 3.1 plots the CDFs of the random travel times on three paths. The discrete probability mass functions of these random variables, as well as their mean, variance and skewness are specified in the table next to the figure. Using Definition 2.4, one can verify that all the three paths are FSD-admissible. According to Definition 3.2, paths 2 and 3 are SSD-admissible, and they both dominate path 1 in the second order. Finally, using

Time	Path 1	Path 2	Path 3
0	0.00	0.00	0.00
1	0.10	0.05	0.15
2	0.19	0.20	0.20
3	0.30	0.38	0.25
4	0.39	0.40	0.35
5	0.50	0.42	0.63
6	0.60	0.50	0.69
7	0.70	0.81	0.75
8	0.80	0.95	0.97
9	0.90	0.99	0.99
10	1.00	1.00	1.00
Mean	5.00	5.30	5.02
Var.	11.00	6.27	6.00
Skew.	0.00	-0.12	-0.19

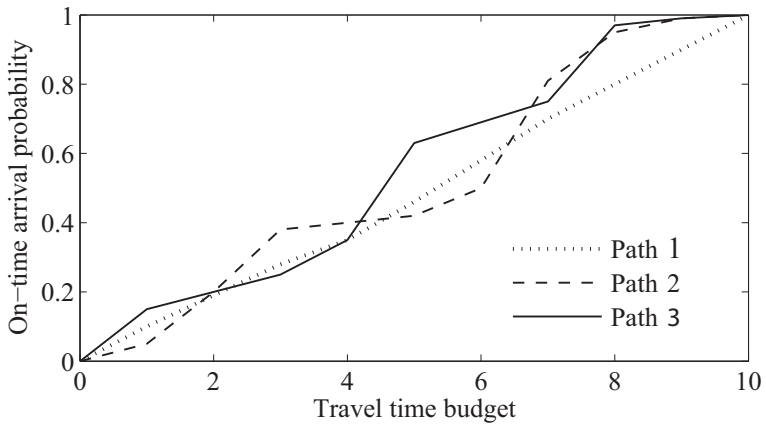


Figure 3.1: Cumulative density functions of travel times of three paths

Definition 3.3 one can show that path 3 dominates paths 2 and 1 in the third order, and hence path 3 is the only TSD-admissible path. Interestingly, path 1 has higher variance comparing to the other two paths, which explains why it is not preferred by any risk-averse traveler. Also note that the TSD-admissible path has the most negative skewness among all paths, which is related to the ruin-averse behavior. However, path 2 is not TSD-admissible even though it has a negative skewness. Thus, negative skewness does not ensure TSD-admissibility.

Finally, path 1 gives an example where admissibility does not warrant optimality. Note that while path 1 is FSD-admissible, it does not give the least time budget for any cumulative probability. Any non-satiable traveler who is interested in minimizing travel time would not choose path 1, regardless of their desired probability of arriving on time.

3.5.2 Finding SSD/TSD-admissible paths

We have shown that FSD-admissible paths can be found by checking the stochastic dominance relationship between any pair of paths and eliminating those that are dominated. SSD/TSD-admissible paths can be found in the same way. SSD/TSD-admissible paths have the following properties:

Proposition 3.2 *SSD/TSD-admissible paths have the following properties: (1) They must be acyclic; (2)*

Subpaths of any SSD/TSD-admissible paths must be also SSD/TSD-admissible.

Proof: cf. Wu & Nie (tentatively accepted). □

According to Proposition 3.2, the same formulation and solution algorithm for finding FSD-admissible path can also be used to find higher-order admissible paths. The only difference is that the operator $\gamma_{\succ_1}$ in the formulation (3.3) should be replaced with $\gamma_{\succ_2}$ and $\gamma_{\succ_3}$ for SSD and TSD, respectively.

Since Γ^{rs}_{TSD} and Γ^{rs}_{SSD} are smaller than Γ^{rs}_{FSD} (Proposition 3.1), these higher-order admissible sets are potentially easier to enumerate. More important, Proposition 3.1 ensures that the higher-order SD-admissible paths must also be FSD-admissible. However, since they lead to different admissible path sets, the higher order SD rules also produce different Pareto-optimal frontiers. In fact, one can show that

$$u^{rs}_{\text{FSD}}(b) \geq u^{rs}_{\text{SSD}}(b) \geq u^{rs}_{\text{TSD}}(b), \forall b$$

That is to say, for the same time budget, the higher-order SD frontiers always generate a more conservative path, which has an equal or lower on-time arrival probability comparing to the FSD counterpart.

Chapter 4

Experiment Results

Numerical experiments are conducted in this chapter to evaluate the performance of various numerical convolution methods, including the newly developed DFT-based schemes, and approximation methods deriving from higher-order stochastic dominance. All algorithms were coded using C++ and tested on a Windows-XP (64) workstation with two 3.00 GHz Xeon CPUs and 4G RAM.

4.1 Evaluation of numerical convolution methods

Results of four experiments are presented in this section. The first three are designed to examine the computational performance of various methods in computing convolutions along two sample paths, each consisting of 20 and 10 links respectively. The link travel time distributions are assumed to follow the Gamma distributions, with the same parameter setting as in Section 4 of Wu & Nie (2009). The first experiment is focused on the impacts of discretization and consolidation on the ADA-based direct convolution scheme. The second experiment compares the convolution results obtained from the ADA-based direct convolution with those from the DFT-based convolution. The third experiment compares the all-at-once and one-at-a-time DFT-based schemes, and also examines the performance of the approximated DFT-based convolution scheme. The last experiment actually solves the RASP problem on a medium-size network (Chicago sketch network), using two convolution schemes: the ADA-based direct convolution

scheme and the approximated DFT-based convolution scheme. By default, the DFT-based convolution is implemented using the one-at-a-time scheme.

4.1.1 ADA-based direct convolution

Recall that the ADA-based direct convolution includes a consolidation procedure to merge support points whose probability mass is less than $1/L$, where L is a predetermined upper bound for the number of support points and can be viewed as a global parameter that controls the solution quality. After consolidation, the actual number of support points used to represent a distribution is typically much smaller than L. While this helps reducing computational cost, it is important to know how much it compromises the solution quality. Another interesting question is how sensitive the convolution results are to the global control parameter L.

To answer the above questions, we generate and compare six distributions for each path using the ADA-based method, each corresponding to $L = 100, 500, 1000$ and with or without consolidation. The CDFs of these distributions are plotted together in Figure 4.1-(a). Clearly, the differences between these distributions are hardly visible at this scale for both paths, which may be an indication that the most computationally efficient choice ($L = 100$ with consolidation) does provide a reasonable solution.

To better gauge the differences, the following error metrics are introduced.

$$\lambda_{\mathrm{A}} = \max_{\forall \alpha \in \Lambda} \left(\lambda_{\mathrm{A}}(\alpha)\right) \quad ; \quad \bar{\lambda}_{\mathrm{A}} = \sum_{\forall \alpha \in \Lambda} \left(\lambda_{\mathrm{A}}(\alpha)\right) / |\Lambda|, \text{ where} \tag{4.1}$$

$$\lambda_{\mathrm{A}}(\alpha) = \frac{|v_{\mathrm{B}}(\alpha) - v_{\mathrm{A}}(\alpha)|}{v_{\mathrm{B}}(\alpha)} \quad ; \quad \Lambda = \{0.01, 0.02, \cdots, 0.99\}, \tag{4.2}$$

and $v_{\mathrm{A}}^{rs}(\cdot)$ and $v_{\mathrm{B}}^{rs}(\cdot)$ are the inverse CDFs of the test and benchmark distributions. Note that we ignore the portions of CDF when $\alpha < 0.01$ and $\alpha > 0.99$ because the differences in these ranges are deemed practically insignificant.

For both paths, the distributions obtained with the highest resolution ($L = 1000$) and without consolidation are used as the benchmark. The CPU time required to obtain each distribution and their error metrics relative to the benchmark distribution are given in Table 4.1. We first

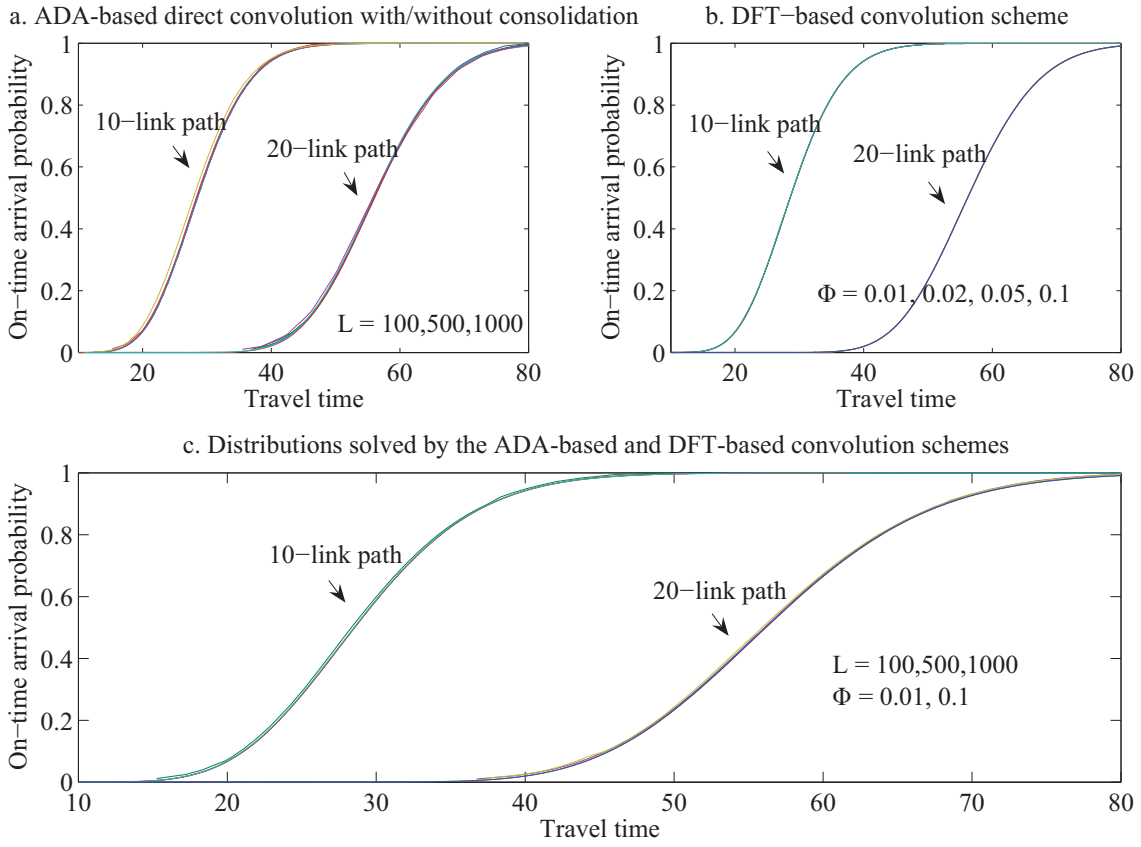


Figure 4.1: Distributions of path travel times solved by different convolution methods, under different setting of length of intervals (ϕ) and number of discrete intervals (L)

Table 4.1: Computational performance of the ADA-based direct convolution

Path	Performance Index	With Consolidation			Without Consolidation		
		$L = 100$	$L = 500$	$L = 1000$	$L = 100$	$L = 500$	$L = 1000$
10-link	Max Error λ_A	1.54%	0.12%	0.10%	0.39%	0.05%	–
	Ave Error $\bar{\lambda}_A$	0.13%	0.02%	0.02%	0.16%	0.03%	–
	CPU time(second)	< 0.001	0.006	0.040	0.009	0.056	0.254
20-link	Max Error λ_A	2.64%	1.22%	0.90%	2.43%	0.12%	–
	Ave Error $\bar{\lambda}_A$	0.12%	0.11%	0.06%	0.09%	0.04%	–
	CPU time(second)	< 0.001	0.012	0.078	0.016	0.109	0.502

Note: the case where $L = 1000$ (highest resolution) and without consolidation is used as the benchmark case.

note that maximum and average errors in all cases are relatively small (the largest deviation from the benchmark is still smaller than 3%). As expected, when L increases, both average and maximum errors decreases while more CPU time is required for the calculation. It seems that the improvement of the solution quality is outpaced by the degrading computational performance. For instance, when L is increased from 100 to 1000 for the 20-link path (with consolidation), the average errors was reduced by 50% while the CPU time was 78 times higher.

In most cases, the errors in the cases with consolidation are larger than those without [1]. However, the gains in the computational performance by consolation seems to make up for the loss in solution quality. For example, when $L = 500$ (the 20-link path), consolidation increases the average error by a factor of 3, but reduces the CPU time by a factor of 9. Because the above results suggest that consolidation is an effective strategy, the ADA-based direct convolution is always implemented with consolidation in other experiments.

4.1.2 ADA-based direct convolution vs. DFT-based convolution

We first reveal how the solution obtained by the one-at-a-time DFT-based convolution scheme converges as the resolution ϕ decreases. For this purpose, four different values of ϕ, 0.1, 0.05, 0.02 and 0.01, are tested. Figure 4.2-a shows that, when ϕ decreases from 0.1 to 0.01, the CPU

[1] Interestingly, for the 10-link path, when $L = 100, 500$, consolidation led to slightly smaller average errors. The mechanism that leads to this seemingly counter-intuitive result remains unclear. However, we note that these incidents are likely to have low occurrence and insignificant consequences.

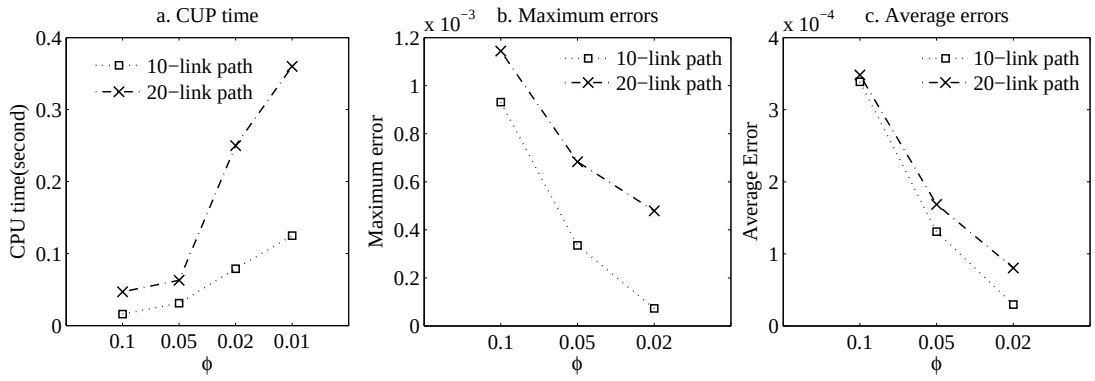


Figure 4.2: Comparison of the distributions generated by the DFT-based convolution with different resolutions. $\phi = 0.1, 0.05, 0.02$ and 0.1) ($\phi = 0.01$ is used as a benchmark to measure the errors of other cases

time required to compute convolution increases from 0.016 seconds to 0.125 seconds for the 10-link path, and from 0.047 seconds to 0.359 seconds for the 20-link path. Figure 4.2-b and 4.2-c show that the average and maximum errors (as defined in Equation (4.1)) of the distributions generated with different ϕ, using the case with $\phi = 0.01$ as the benchmark. Clearly, both error indexes decrease quickly with ϕ. The maximum error for $\phi = 0.1$ is 0.11% and 0.09% for the 20-link path and 10-link path, respectively, and the maximum errors of $\phi = 0.02$ is reduced to 0.05% and 0.01%.

Figure 4.1-(c) compares the distributions solved using the ADA-based direct convolution (where $L = 100, 500, 1000$), with those solved by the DFT-based convolution (where $\phi = 0.01$). Table 4.2 reports the error indexes using the DFT-based solution as a benchmark, which shows that the relative errors increase as L decreases. However, even when $L = 100$, the maximum and average errors are only 1.23% and 0.14%, respectively, for the 20-link path. On the other hand, the distribution of the 20-link path solved from the DFT-based convolution uses 42,412 support points, and it took about 0.688 seconds CPU time to perform the convolution. As a comparison, the ADA-based direct convolution ($L = 100$) only generates 55 support points for the final distribution, and the entire convolution process takes less 0.001 seconds CPU time. This suggests that the ADA-based direct method is able to strike a better balance between efficiency and accuracy, compared to the DFT-based method.

Table 4.2: Comparison of the distributions generated by the ADA-based direct convolution ($L = 100, 500, 1000$) and by the DFT-based convolution ($\phi = 0.01$), which is set as the benchmark

Performance Index	10-link path			20-link path		
	$L = 1000$	$L = 500$	$L = 100$	$L = 1000$	$L = 500$	$L = 100$
Maximum error λ_A	0.11%	0.44%	1.36%	0.10%	0.34%	1.23%
Average error $\bar{\lambda}_A$	0.01%	0.01%	0.14%	0.01%	0.02%	0.14%

4.1.3 Comparison of DFT-based convolution schemes

This section compares three DFT-based convolution schemes: the all-at-once convolution scheme (DFT-I), the one-at-a-time convolution scheme (DFT-II), and the approximated DFT-based convolution scheme (DFT-III). We still set $\phi = 0.1$, 0.05, 0.02 and 0.01 in this experiment. For DFT-III, two accuracy indexes are tested, namely $\epsilon = 0.01$ and 0.001.

The CPU times used to calculate the travel time distributions of two paths are reported in Table 4.3, and the error indexes are reported in Table 4.4 (the benchmark distributions are those solved by DFT-II with $\phi = 0.01$). These results indicate that removing the ineffective support points leads to significant gains in the computational efficiency, at the expense of modest loss in accuracy. When $\phi = 0.01, \epsilon = 0.01$, for example, DFT-III only needs about 1/6 of the CPU time used by DFT-II for the 20-link path. Yet the average error is only 0.53%.

Table 4.4 shows that the one-at-a-time and all-at-once convolution schemes produce almost identical convolution results. From Table 4.3 we can see that DFT-I is slightly faster than DFT-II for the 10-link path, but mostly slower for the 20-link path. This observation agrees with the complexity analysis given in Section 3.4.2, which suggests that the two schemes have similar complexity but DFT-I is likely to become relatively slower as the path contains more links. For further clarification, a 5-link path is tested and the results obtained from the two schemes are reported again in Table 4.3. As expected, DFT-I is consistently faster than DFT-II in all cases and the differences of CPU times are significant. While by no means conclusive, these results seem to indicate that the ideal all-at-once convolution may perform well only when paths are short.

Table 4.3: Computation time (in seconds) of three DFT-based convolution schemes

Reso. ϕ	20-link path				10-link path				5-link path	
	DFT I	DFT II	DFT-III $\epsilon = 0.01$	DFT-III $\epsilon = 0.001$	DFT I	DFT II	DFT-III $\epsilon = 0.01$	DFT-III $\epsilon = 0.001$	DFT I	DFT II
0.1	0.047	0.047	0.015	0.016	0.015	0.016	< 0.001	< 0.001	< 0.001	0.015
0.05	0.110	0.063	0.016	0.016	0.015	0.031	< 0.001	< 0.001	< 0.001	0.015
0.02	0.204	0.251	0.031	0.063	0.062	0.079	0.015	0.032	0.016	0.032
0.01	0.453	0.359	0.078	0.141	0.125	0.125	0.031	0.047	0.016	0.047

Note: DFT-I refers to the all-at-once convolution scheme; DFT-II refers to the one-at-a-time convolution scheme; and DFT-III refers to approximated DFT-based convolution scheme.

Table 4.4: Comparison between the distributions generated by the three DFT-based schemes, the benchmark distributions are those solved by DFT-II with $\phi = 0.01$

Scheme	Index \ Reso.ϕ	20-link path				10-link path			
		0.1	0.05	0.02	0.01	0.1	0.05	0.02	0.01
DFT-III $\epsilon = 0.01$	Max. err. λ_A	6.32%	3.48%	3.49%	6.26%	4.19%	4.09%	4.06%	4.05%
	Ave. error $\bar{\lambda}_A$	0.74%	0.61%	0.61%	0.53%	0.92%	0.85%	0.82%	0.81%
DFT-III $\epsilon = 0.001$	Max. err. λ_A	3.02%	2.30%	2.28%	0.65%	3.57%	3.52%	3.48%	3.47%
	Ave. err. $\bar{\lambda}_A$	0.75%	0.65%	0.62%	0.05%	1.18%	1.08%	1.03%	1.01%
DFT-I	Max. err. λ_A	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Ave. err. $\bar{\lambda}_A$	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

4.1.4 Impacts of convolutions schemes on reliable path finding

The previous results have shown the both ADA-based and approximated DFT-based convolution schemes provide satisfactory trade-off between efficiency and accuracy. We now further examine these two schemes by applying them to solve the RASP problem on the Chicago sketch network, which has 933 nodes and 2950 links (Wu & Nie 2009). All link travel times are assumed to follow Gamma distributions with the same parameter setting in Section 4 of Wu & Nie (2009). For the approximated DFT-based convolution scheme, the travel time resolution ϕ is set to 0.1, 0.2 and 0.5, and the accuracy index ϵ is set to 0.01 and 0.001. For the ADA-based direct convolution scheme, L is set to 100.

In this experiment, the FSD-Pareto-optimal frontiers are used to measure the quality of the solutions obtained by different convolution schemes. The solution obtained by the ADA-based direct convolution scheme is used as the benchmark to check the relative errors of the various approximated DFT-based convolution schemes (with different ϕ and ϵ). Two frontiers are compared in three different intervals of cumulative probability α: $\Xi_1 = [0.05, 0.06, \cdots, 0.98, 0.99], \Xi_2 = [0.10, 0.11, \cdots, 0.94, 0.95], \Xi_3 = [0.10, 0.11, \cdots, 0.89, 0.90]$. Then we define extreme maximum error and average maximum error for each Ξ as follows:

$$\text{Sup.}\lambda^{s}_{\Xi} = \sup_{\forall i \in \mathcal{N}} \left\{ \max_{\forall \alpha \in \Xi} \left[\lambda^{is}(\alpha) \right] \right\} \tag{4.3}$$

$$\text{Ave.}\lambda^{s}_{\Xi} = \frac{1}{|\mathcal{N}|} \left\{ \max_{\forall \alpha \in \Xi} \left[\lambda^{is}(\alpha) \right] \right\} \tag{4.4}$$

where $\mathcal{N}$ is the set of all nodes in the network. To mitigate the effects of network topology, the RASP problem is solved by each convolution scheme ten times, each for a randomly selected destination, and the above indexes are averaged over the the ten runs and reported in Tables 4.5 and 4.6.

The results show that the ADA-based convolution runs faster than the approximated DFT-based convolution in most cases, with the exception of the case where $\phi = 0.5$ and $\epsilon = 0.01$. The errors of the approximated DFT-based convolution decreases with ϕ and ϵ, suggesting that

Table 4.5: Error indexes of the approximated DFT-based convolution scheme (using the ADA-based direct convolution scheme with $L = 100$ as benchmark)

Accuracy ϵ	Resolution of Travel time ϕ	Ave.λ^s $\alpha \in [0.05, 0.99]$	Sup.λ^s $\alpha \in [0.05, 0.99]$	Ave.λ^s $\alpha \in [0.1, 0.95]$	Sup.λ^s $\alpha \in [0.1, 0.95]$	Ave.λ^s $\alpha \in [0.1, 0.9]$	Sup.λ^s $\alpha \in [0.1, 0.9]$
0.001	0.1	1.15%	8.80%	0.55%	4.20%	0.43%	4.20%
	0.2	2.76%	52.37%	1.30%	11.18%	0.97%	10.39%
	0.5	6.67%	43.80%	2.97%	32.28%	2.08%	19.19%
0.01	0.1	7.75%	10.72%	3.47%	5.07%	2.20%	4.16%
	0.2	7.31%	27.05%	3.11%	10.78%	2.02%	6.13%
	0.5	6.23%	120.70%	3.21%	66.39%	2.14%	19.36%

Table 4.6: Computational performance of the approximated DFT-based convolution scheme with different ϕ and ϵ. The performance of the benchmark case (the ADA-based direct convolution with $L = 100$: CPU time - 0.90 seconds, Ave. $|\Gamma^{is}_{\text{FSD}}|$ - 2.86 and Max. $|\Gamma^{is}_{\text{FSD}}|$ - 16.6)

Resolution of Travel time ϕ	CPU time (second)	Ave. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$	Max. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$	CPU time (second)	Ave. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$	Max. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$
	Accuracy $\epsilon = 0.01$			Accuracy $\epsilon = 0.001$		
0.1	2.69	1.91	7.43	3.71	2.04	7.00
0.2	1.27	1.97	7.86	1.92	2.25	7.71
0.5	0.71	1.98	7.86	1.64	2.59	10.14

the choice of the benchmark is appropriate. In most cases, the distributions generated by two schemes are close enough in the interval of $\alpha \in [0.1, 0.9]$, with the average errors less than 3%. The errors are larger in the interval where α is close to 1 or 0, clearly due to the fact that the ineffective support points are removed in these intervals. Overall, the approximated DFT-based convolution does not outperform its ADA-based counterpart, despite it has a much better theoretical complexity.

We close this section by commenting on the choice of ϕ in the DFT-based method. A close look at the results reveals that those large extreme maximum errors reported in Table 4.5 may be caused by the inappropriate choice of ϕ. Specifically, $\phi = 0.5$ or 0.2 may be too large for the effective support range on some links in the network, which in turn leads to poorly represented distributions (with a very few support points). To avoid this problem would require selecting a ϕ small enough for all links. Unfortunately, this strategy is likely to be inefficient in a large network where the support ranges on links vary significantly.

4.2 Pareto-optimal frontiers generated by higher order SD rules

This section tests the label-correcting algorithm implemented with different SD rules, namely, FSD, SSD and TSD. Both computational performance and the solution quality (measured by the generated Pareto frontiers) are compared. For convenience, we name the solution algorithm corresponding to FSD, SSD and TSD rules as FSD-LC, SSD-LC and TSD-LC, where LC stands for label-correcting. These algorithms are tested on four networks, the Chicago Sketch network (933 nodes and 2950 links, cf. Nie & Wu (2009)), and three grid networks: 40×40 (i.e. 1600 nodes and 6240 links), 50×50 (i.e. 2500 nodes and 9800 links), and 60×60 grid network (i.e. 3600 nodes and 14160 links). Two different types of link travel time distributions are considered here: Gamma distributions and Uniform distribution. The parameter setting of Gamma distributions is exactly the same setting as in Wu & Nie (2009). For uniform distributions $p_{ij}(x) = 1/(U - L)$, L is fixed at 0 and U is randomly drawn from $[3.5, 10]$.

Table 4.7: The gaps between FSD, SSD and TSD Pareto-optimal solutions

	FSD-SSD				FSD-TSD			
network	Ave.λ^s	Sup.λ^s	Ave.λ^s	Sup.λ^s	Ave.λ^s	Sup.λ^s	Ave.λ^s	Sup.λ^s
	$\alpha \in [0.01, 0.3]$		$\alpha \in [0.3, 0.99]$		$\alpha \in [0.01, 0.3]$		$\alpha \in [0.3, 0.99]$	
Gamma distribution								
CK	0.60%	17.15%	0.04%	2.00%	0.79%	15.11%	1.00%	13.45%
40×40	0.57%	20.29%	0.07%	2.03%	0.67%	15.08%	0.86%	20.04%
50×50	0.32%	14.68%	0.04%	1.71%	0.65%	14.27%	1.00%	16.11%
60×60	0.33%	22.35%	0.04%	2.15%	0.60%	18.03%	0.82%	19.10%
Uniform distribution								
CK	0.36%	15.97%	0.01%	0.85%	0.38%	8.90%	0.72%	12.05%
40×40	0.18%	8.48%	0.02%	0.88%	0.38%	8.90%	0.35%	7.83%
50×50	0.13%	18.02%	0.02%	1.12%	0.12%	4.31%	0.20%	6.86%
60×60	0.10%	7.32%	0.02%	0.71%	0.13%	4.87%	0.19%	7.42%

First, the maximum difference between the FSD Pareto-optimal frontier and the SSD and TSD frontiers for any $\alpha \in \Xi$ are obtained for any node $i \neq s$ as $\lambda^{is}(\alpha)$. Then, the so-called extreme maximum error and average maximum error are defined for each Ξ as follows:

$$\text{Sup.}\lambda^s_\Xi = \sup_{\forall i \in \mathcal{N}} \left\{ \max_{\forall \alpha \in \Xi} \left[\lambda^{is}(\alpha) \right] \right\} \tag{4.5}$$

$$\text{Ave.}\lambda^s_\Xi = \frac{1}{|\mathcal{N}|} \left\{ \max_{\forall \alpha \in \Xi} \left[\lambda^{is}(\alpha) \right] \right\} \tag{4.6}$$

To avoid the fluctuations associated with network topology, all algorithms are run ten times, each for a randomly selected destination for each network, and the above indexes are averaged over the the ten runs and reported in Tables 4.7. Note that two values of Ξ are considered in the table: $\Xi_1 = [0.01, 0.3]$ and $\Xi_2 = [0.3, 0.99]$.

The results reported in Table 4.7 show that the gap between FSD and SSD optimal solutions are small, especially when $\alpha \in [0.01, 0.3]$. Within this range, the average maximum error is less than 0.1% and the extreme maximum error is smaller than 3%. The discrepancy between FSD and SSD frontiers is larger for smaller α. This is expected, because the SSD Pareto-optimal solutions impose risk averse behavior. Thus, when travelers are more risk-prone (e.g. they are willing to accept a small on-time arrival probability in exchange for a smaller time budget), the reliable

Table 4.8: Computational performance when using FSD, SSD and TSD to identify non-dominated paths

	CPU time (second)	Ave. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$	Max. $\lvert\Gamma^{is}_{\text{FSD}}\rvert$	CPU time (second)	Ave. $\lvert\Gamma^{is}_{\text{SSD}}\rvert$	Max. $\lvert\Gamma^{is}_{\text{SSD}}\rvert$	CPU time (second)	Ave. $\lvert\Gamma^{is}_{\text{TSD}}\rvert$	Max. $\lvert\Gamma^{is}_{\text{TSD}}\rvert$
	Gamma Distribution								
CK	0.90	2.86	16.6	0.50	1.59	5	0.28	1.44	2
40×40	4.13	4.16	31.7	1.75	1.85	6.2	0.98	1.61	2
50×50	7.02	4.57	40.1	2.15	1.77	6.4	1.26	1.62	2
60×60	20.21	5.95	61	4.51	1.88	6.8	2.79	1.66	2
	Uniform Distribution								
CK	1.50	2.30	17.9	0.89	1.50	4.7	0.64	1.48	2
40×40	4.20	3.19	30.2	2.12	1.62	5.1	1.63	1.61	2
50×50	7.98	3.90	42.9	3.14	1.63	5.1	2.59	1.61	2
60×60	11.29	3.7	53.4	4.69	1.63	5.1	3.87	1.61	2

route generated by SSD tends to be more conservative. The difference between FSD and TSD Pareto-optimal solutions is still larger, with the extreme maximum error is as large as 20% even when $\alpha > 0.3$. This indicates that the TSD frontier might be too conservative as an approximation to FSD.

Table 4.8 shows SSD-LC is about twice as fast as FSD-LC on average. In some cases, the SSD-LC is almost five times faster. The main reason is that the number of SSD-admissible paths is less than that of FSD-admissible paths. Also, the speed of TSD-LC is slightly better than SSD-LC, but the difference is barely noticeable in most cases. This is not a surprise given that the numbers of average admissible paths generated by TSD and SSD are very similar.

The above results suggest that SSD-LC is a desirable approximation algorithm to finding most reliable paths. It consistently reduces the CPU time by more than 50%, and the resulted approximation errors are relatively small, especially for larger α. Also, SSD-LC tends to over-estimate the time budget for a given on-time arrival probability, which is acceptable to risk-averse travelers. In contrast, TSD-LC results in much larger approximation errors, while offering little gains in computational efficiency.

4.3 Summary

The findings from the numerical experiments are summarized as follows:

1. The basic DFT-based convolution scheme can produce highly precise solutions yet has high computation overhead. The approximated DFT-based convolution scheme proposed in this project improves the computational performance by removing "ineffective" support points, which makes it more suitable to large applications.

2. The conventional all-at-once DFT-based convolution scheme does not necessarily outperform the one-at-a-time scheme proposed in this project, especially when paths have a large number of member links. Moreover, it is difficult to apply the all-at-once convolution scheme in reliable path finding without path enumeration.

3. Compared with all DFT-based convolution schemes, the ADA-based direct convolution scheme strikes a better balance between accuracy and efficiency.

4. Reliable shortest paths generated by the SSD rule are good approximations to those based on the FSD rule, especially when the desired on-time arrival probability is high. The CPU time required to solve the reliable shortest path problem is reduced by a factor of two when the SSD-rule is used. The TSD rule offers similar computational performance but leads to much larger approximation errors.

Chapter 5

CTA Bus Data: Acquisition and Analysis

This chapter summarizes the efforts to extract useful travel time information from existing CTA AVL bus database, and analyzes the usefulness of these data for the purpose of travel time estimation on different roads.

5.1 Background

The technical hardware for the bus probe analysis is a pre-existing bus status reporting system developed by CleverDevices for the CTA technology department. Each of the 2000 operative buses in the CTA fleet is equipped with an onboard computer and mobile networking suite. The ”brain” of the system is the Intelligent Vehicle Network (IVN) computer, originally developed to automate the public address system on each bus.

To perceive which stop the bus is approaching, the IVN is equipped with a GPS locator. In an extension of this existing functionality, the CTA in late 2008 experimented with adding a mobile access router to each bus with the ability to transmit small-scale data reports to centralized servers [1] in real time. Since the system went live in May 2009 (and for approximately six months prior), every CTA bus has transmitted its location in geographic coordinates and instantaneous speed to the CTA approximately every 30 seconds. In addition, operator, route, bus, and pattern

[1] Note: The CTA stores its regularly-collected BusTracker data in a centralized archive accessible only by CTA personnel. The following sections give the procedure for pulling selected data from the archive, performing analytic functions on that data, and mapping it for visual analysis. All code was developed using PL/SQL developer for use on Oracle Server 10g, with the Oracle Spatial feature. Spatial columns, which cannot be exported to non-GIS data types, are noted where used.

identification numbers are retrieved for each record by a series of archive database triggers (they are reported only when they change). Currently, the data is archived by the CTA technology department indefinitely.

Because the location of the bus and the time between reports (technically referred to as ”polls”) is known continuously as a bus progresses on a route, the average speed of the bus between two contiguous polls can be calculated as the distance traveled divided by elapsed time. The segment between the two polls over which the speed is calculated is referred to as the poll segment. Filtered and corrected to allow for peculiarities of bus traffic, the poll segment speeds have been shown in limited validation testing to be a reasonable approximation of the traffic speed on that segment of road.

The individual poll segments themselves can be mapped and analyzed, but for our purpose, they must be matched to the nearest link in the CMAP network. Many poll segments may match to the same link, in which case each poll segement is considered as an independent observation of the speed on the matched link.

5.2 Data extraction

5.2.1 Data collected

The CTA archives the following information from each bus in the process of normal polling:

- The poll time in both date and time. This value is referred to as ”AVL_Event_Time”, with AVL standing for Automatic Vehicle Locator. An AVL Event is essentially a poll of a bus location.
- Identification numbers for individual poll, bus, operator, route, run, trip, pattern (a route can be run in either direction, or sometimes with planned spurs) and garage of origin.
- Instantaneous speed - Experimentation (through riding CTA buses with handheld GPS and a networked computer showing short-term polling) has demonstrated that the instantaneous speed is significantly inaccurate. The average poll segment speed calculated later

has been demonstrated accurate and all future references to ”speed” will refer to segment average speed.

- Latitude, Longitude, and Heading (standard GPS WGS ’83 format, north taken as 0 degrees) at the moment the poll was collected - Although headings taken Chicago streets, which largely conform to cardinal grid typically fall near 90, 180, 270, and 360/0 degrees, a bus turning into a stop can generate a heading nearly perpendicular to its overall direction of travel.

- A reading of whether the bus was on-route - A true/false response to whether a bus is beyond set threshold of distance from the last route logged in.

An archive exists of this information for all polls dating from approximately January 1, 2009. This archive table is ordinary Oracle Database 10g, without any Spatial columns, meaning that this data can be directly exported to text and comma delineated formats.

5.2.2 Programming structure

The task of first calculating, then matching average bus speeds to network links is accomplished by a BUS_SPEED package (see Appendix A). In PL/SQL parlance, a package is a collection of functions and procedures held together for convenience. A package consists of a declaration, which lists the contents, and a body, which contains the full coding for each sub-program in the package.

A package cannot itself be executed. Rather, it is a vessel for holding other executable code. Therefore, the package BUS_SPEED contains a procedure BUS_SPEED.MAIN_GO which serves to call all the other sub-programs in the package in a specific order.

A query calls BUS_SPEED.MAIN_GO with three pieces of input. First, the user inputs either ’SRA’ or ’NU’ to indicate which network the poll segments should be matched to. ’SRA’ indicates the City of Chicago Transportation Department’s Strategic Arterial Network, which was used for testing for simplicity. Future networks might include ’CTA’ for network based around CTA bus

lines.

```
BEGIN BUS_SPEED.MAIN_GO('NU'); END;

– This query calls for matching to the CMAP network for the last full 15-minute period.
```

```
BEGIN BUS_SPEED.MAIN_GO('SRA'); END;

– This query calls for matching to the strategic arterial network for the last full 15-minute period.
```

The user can input the start time and end time for the query in SQL form. If no dates are entered, the function FN_LAST_15 will find the last full 15-minute period (at the current time) and assign the beginning and end times of that period as the ends of the query. For example, if the procedure was called at 12:17, the query would begin at 12:00 and end at 12:15. A 15-minute period has been demonstrated as the shortest period in which bus coverage of the network can be maximized, any shorter and the number of unmatched segments increases.

```
BEGIN BUS_SPEED.MAIN_GO('NU',to_date('09-11-2009 17:00', 'mm-dd-yyyy hh24:mi'),to_date('09-11-2009 18:00', 'mm-dd-yyyy hh24:mi')); END;

– This query calls for matching to the CMAP network for the period between 5:00 and 6:00 p.m. on Friday, September 11, 2009.
```

- BUS_SPEED.MAIN_GO calls five other procedures, though no query will require it to call all of them:
- BUS_SPEED.BUILD_SPEED_TABLE - Constructs a table POLL_SPEED with speed, geography, and identification for each poll segment.
- BUS_SPEED.CLEAN_SPEED_TABLE - Removes irrelevant or inaccurate polls.
- BUS_SPEED.NU_MATCH - Uses a function FN_NU_MATCH to attempt to retrieve a matching CMAP link ID number for every poll segment.
- BUS_SPEED.SRA_MATCH - Does the same as the previous procedure for the strategic arterial segments.

- BUS_SPEED.CONGESTION_SCORE - Assigns a congestion score based on segment averages. This additional functionality allows for easy identification of atypical momentary congestion but removes the ability to compare the performance of different segments.

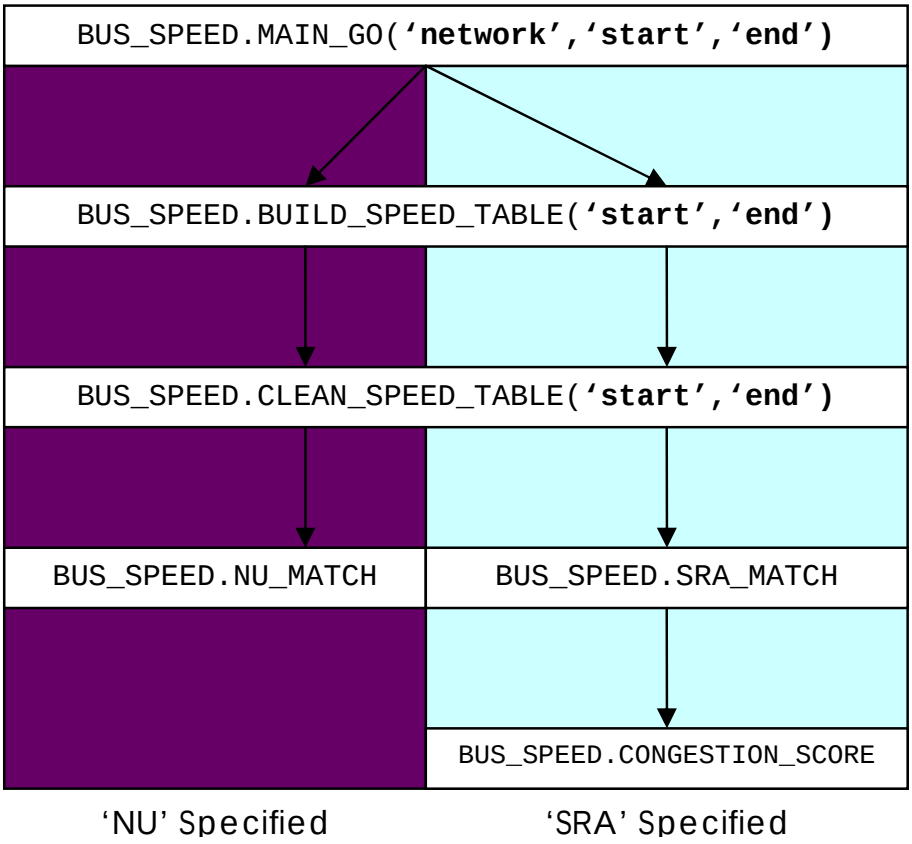


Figure 5.1: The progression of BUS_SPEED based on input variables

At peak hour, one 15-minute query using the BUS_SPEED package takes approximately 2 minutes to run. A stepwise log is kept for each run using the procedure BUS_SPEED.APP_LOG_INSERT and can be accessed even while the program is executing.

5.2.3 Constructing POLL_SPEED

BUS_SPEED.BUILD_SPEED_TABLE constructs POLL_SPEED in two phases. First, data are pulled from the archive into an intermediate table POLL_DATA_INNER. Then, the speed calculations are performed, and the results are recorded in POLL_SPEED. POLL_DATA_INNER has no

Spatial columns. The POLL_SPEED table has a Spatial column POLLSEG which records the poll segments as line segment geometries.

The POLL_DATA_INNER table contains the following data extracted from the archive:

- Identification numbers for bus, operator, route and pattern, direction and deviation of the bus
- Latitude, longitude, and heading at the poll
- Latitude, longitude, and heading at the previous poll - Because Oracle cannot perform operations up and down in the table (one row against another row), but only across a row from column to column, the location and heading information is lagged by one row and reported in separate columns LAST_LATITUDE, LAST_LONGITUDE, and LAST_HEADING [2].
- Time of poll and the time of the last poll.
- The time difference between the two polls, the elapsed time used in calculating poll segment average speed.

LAG(AVL_EVENT_TIME, 1) OVER (PARTITION BY BUS_ID ORDER BY AVL_EVENT_TIME) LAST_AVL_EVENT_TIME

– This excerpt from BUILD_SPEED_TABLE calls for a new column called LAST_AVL_EVENT time consisting of the column AVL_EVENT_TIME partitioned by BUS_ID then ordered chronologically, shifted downward by 1 row, so that each row of LAST_AVL_EVENT_TIME contains the contents of AVL_EVENT_TIME from the row directly above (see below).

POLL_SPEED consists of all of the data in POLL_DATA inner with the following additions:

- A Spatial function computes the distance between the current and last latitude and longitude, the distance the bus traveled in the elapsed time calculated above, in units of feet.

[2] Note: Lagging is a powerful tool within Oracle, and one that is somewhat unique to the software. Lagging, or the built-in LAG function in PL/SQL, takes an entire specified column and shifts it down (or up, using the LEAD function) by a specified number of rows. This can then be renamed as a separate column.

- The speed is then calculated by calculating the ratio of distance to speed, using units of miles per hour.

- The segment is saved as a Spatial line segment geometry. These columns cannot be exported in text or comma-delineated formats, meaning the table cannot be exported without first having these columns removed. The Spatial columns record the form of the segment geometrically in Spatial format, which can be matched to network links recorded in the same manner.

It is important to note that the POLL_SPEED table includes only segments between 5 and 70 seconds long (to remove inaccurate records from faulty equipment) as well as selecting out polls recorded at zero latitude and longitude, which sometimes occur when the bus first starts.

5.2.4 Cleaning the speed table

When built, the POLL_SPEED table initially contains approximately 50,000 records per 15 minutes of query, each representing a poll segment (there are the same number of poll segments as polls, each segment is identified by its endpoint - see below). The remaining steps, cleaning and matching, remove a large number of those records either because they are considered irrelevant, because they are inaccurate, or because they cannot be matched to a link in the applicable network.

The cleaning step is accomplished by the procedure BUS_SPEED.CLEAN_SPEED_TABLE, which deletes records from POLL_SPEED meeting the following criteria:

- The first 90 seconds of data - When the AVL_EVENT_TIME column was lagged during the creation of POLL_DATA_INNER, the first poll of a time period (noon in the example query 12:00 to 12:15) would not have an associated previous poll and therefore could not be included in a segment. This would cause that poll segment to be missed in the survey altogether. To correct for this, BUS_SPEED.MAIN_GO actually asks for 90 seconds of data before the requested start of the query. Because the package is built to run repeatedly every

15 minutes, that 90 seconds would be double-counted (once by the prior 15 minute period it is actually in, and once by the next one). To correct this, the first 90 seconds of data are deleted from POLL_SPEED.

- Polls where the segment average speed fell above 65 miles per hour - CTA buses are speed governed to approximately 60 mph (depending on the model of the bus). There are situations where a bus could exceed this speed, specifically downhill runs on expressways under good conditions, but a bus exceeding 65 mph is assumed an error. This does not necessarily mean the bus is moving that fast; it is more likely either that polls were taken too close together in time (partially corrected by the 5-second requirement above) or that the GPS incorrectly located the bus by some small degree.

- Polls where the bus did not move for approximately 90 seconds - While it is possible that a bus remaining in precisely the same location for three or more polls is stuck in traffic, vehicles in heavy traffic typically creep slightly over that period. The movement of even a small creep can be registered by the on-board GPS, so a bus which is recorded as not having moved for 90 seconds is assumed to be stationary, usually parked. This typically happens for one of several reasons:

  - The bus is at a terminus and is waiting to begin the return trip (typically takes 3-5 minutes)
  - The bus is engaged in a long relief stop and is waiting for a new driver at a shift change (can take up to 10 minutes)
  - The bus is trapped in a stop by traffic, the effects of which would not be felt by a car at the same location and moment.
  - The bus is parked at a garage.

At the conclusion of these steps, which usually amount to approximately one fifth of the initial total number of polls, the POLL_SPEED table is complete. It contains the balance of the

relevant poll segments over the queried time period.

5.2.5 Matching poll segments with network links

At this point SpatialConsole (and potentially ArcMap) can simply plot the poll segments. They follow the street grid and could theoretically be selected for metadata such as average speed. Such a map, however, is both complex (since overlapping speeds may vary widely) and imprecise (since outliers are reported equally with ”typical” results). To make a usable color-coded speed map, the poll segments must be aggregated by matching with a simpler network.

BUS_SPEED is currently capable of matching the poll segments with two sets of network links. One is the City of Chicago’s Strategic Arterial Network, also referred to as SRA segments. Because the SRA network does not include many major streets (Belmont Avenue, for example), it does not have the breadth to show the much specific information. It is, however, simple enough to experiment with and includes all relevant metadata (street name, endpoints, direction, number of bus stops, etc.)

The second network is the CMAP network used by our previous project. Maps are possible for this network as well. However, the lack of street name and endpoints for the segments prevents full efficacy of this network. Prior difficulties obtaining link direction for the CMAP network were addressed with the creation of the function FN_SEG_DIR to assign angle, bearing, and direction to line segments (see Appendix B). Samples of both maps are provided on in the following page.

Figures 5.2 and 5.3 are sample maps of each network to which BUS_SPEED can match its poll segments. The time period used for each map is the same, from 5:00 to 5:15 p.m. on Wednesday, August 12, 2009.

This time period is representative of typical weekday evening peak hour conditions, and took place during the three-day weekday period used to determine base average speeds for congestion score assignment.

Grey indicates that no poll segments were matched to a link. Red indicates congested flow

and green indicates free flow, with yellow and orange in between. (see Section **??**)

In addition to the CMAP network, it is also possible to match the poll segments to a smaller subset of Chicago streets identified by the City of Chicago Transportation Department as strategic arterial roads.

While this network is sparser and leaves out many important arterial roads citywide, it is simple enough for experimentation and easier to read to identify citywide traffic trends.

This network was used for all testing of the BUS_SPEED package.

Both networks are matched by the same process. A pair of functions in BUS_SPEED, FN_SRA_MATCH and FN_NU_MATCH, are passed the location of a poll segment by matching procedures, BUS_SPEED.SRA_MAT and BUS_SPEED.NU_MATCH. The function receives the location of a poll segment and attempts to return the number of a matching link, based on the following criteria:

- The closest link - The Spatial function SDO_NN finds the nearest neighbor in Spatial terms, using the Spatial columns in both the POLL_SPEED table and the network link table. It is believed that this is done by distance from midpoint-to-midpoint, though the documentation is not clear in this regard and the function is built into Spatial. SDO_NN actually finds a list of closest neighbors, which is then pared down to the closest which meets the other criteria.

- The link is within 100 meters (300 feet) of the poll segment - This prevents poll segments in outlying locations off the network from being assigned randomly to border segments.

- The current and previous headings are consistent with the direction [3] of the link - This is a crucial step which serves two important purposes. First, each street in a spatial network is represented by two parallel lines, one for each direction on that street. Because congestion can differ across the directions, poll segments must be matched to a specific side of the

[3] Note: The directional matrix used to match the poll segments is defined so that North and South are very narrow (20 degrees wide) while East and West are very wide (approximately 70 degrees wide) with the remainder given to the non-cardinal directions). The reason for this is that diagonal streets in Chicago are annotated either as East-West, Northeast-Southwest, or Northwest-Southeast depending on their angle, but none is annotated as North-South. To ensure that the maximum number of diagonal segments are matched, those directional ranges are atypically wide.

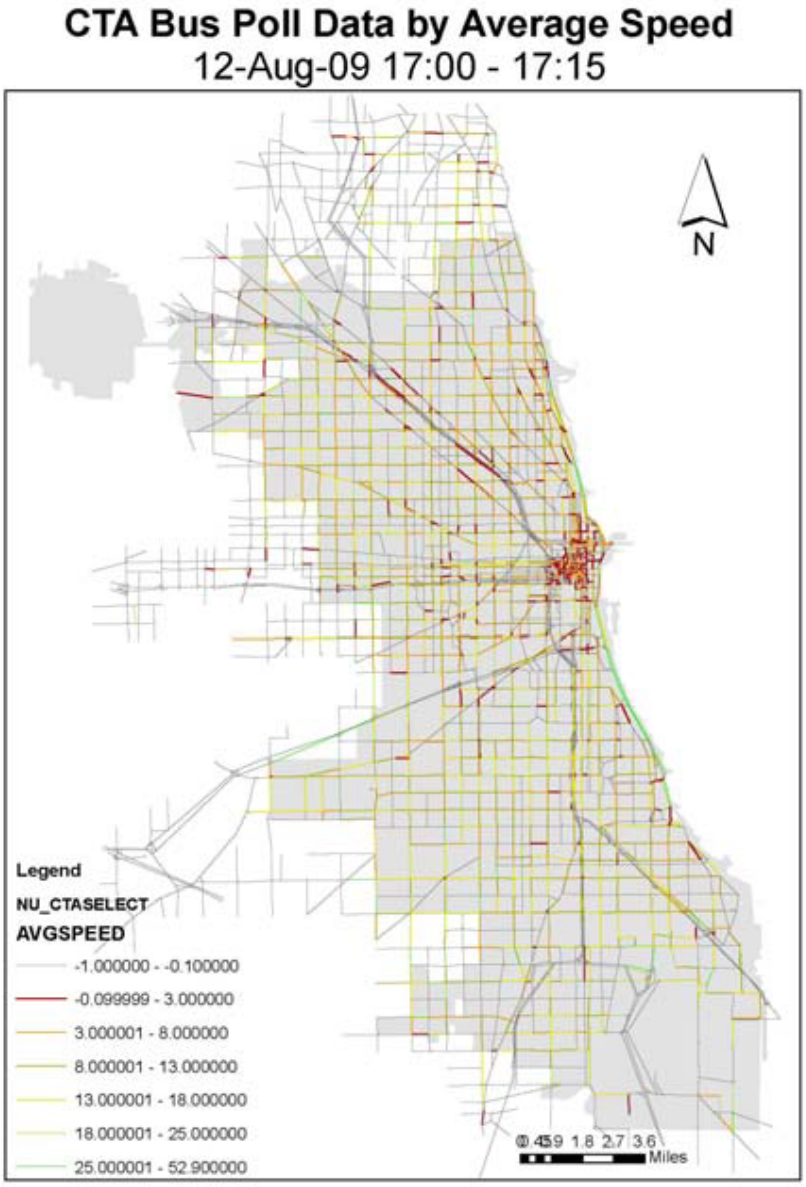


Figure 5.2: NU CMAP sample map

street by heading. Also, buses cross intersections and sometimes turn. A bus which reports in an intersection could be matched to a segment on the perpendicular street by mistake. A bus which turns is not truly reporting conditions on either street, and must be ignored. Checking both the instantaneous and previous headings with the link directions ensures that these conditions are met.

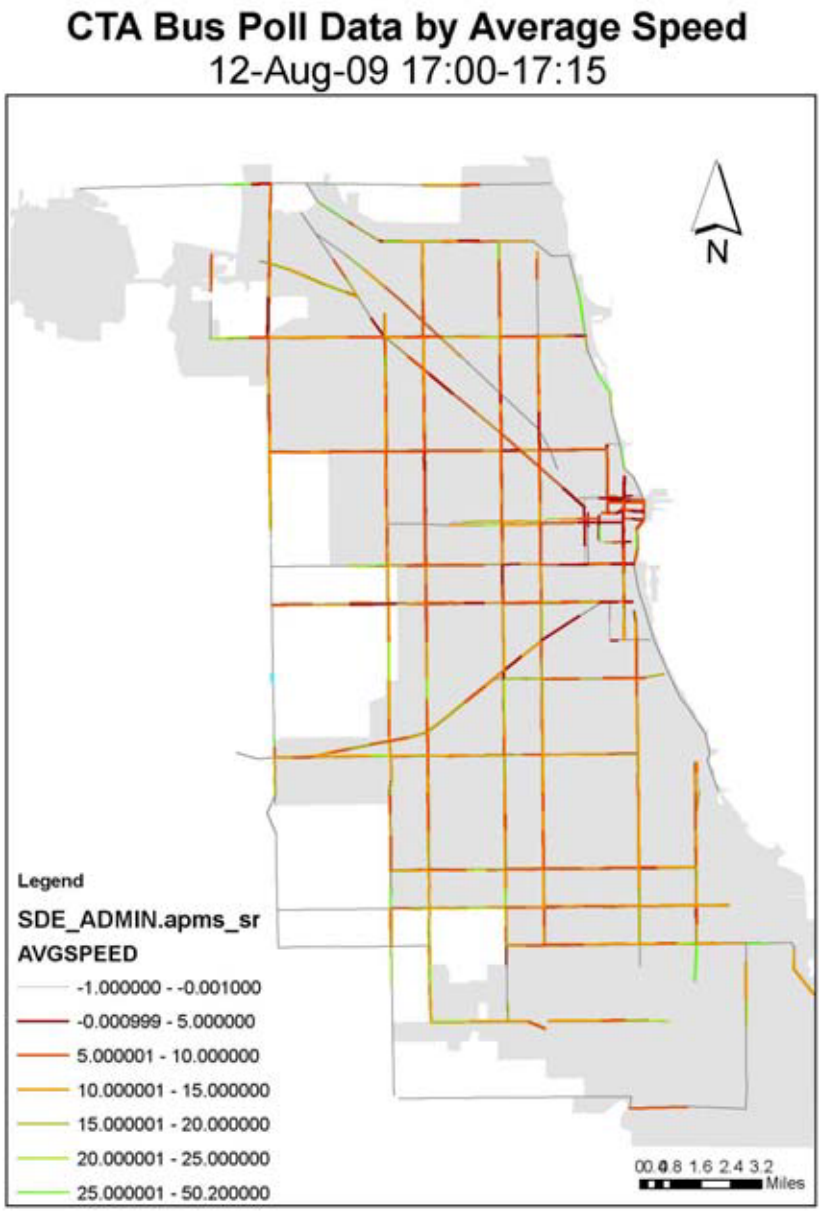


Figure 5.3: SRA sample map

At the conclusion of matching, a MATCH_ID column in POLL_SPEED contains either the ID number of a matching network link (from either network) or a NULL if the segment cannot be matched. Approximately 50% of the initial poll segments successfully match (to the CMAP network, 25% match to the less-comprehensive SRA network). BUS_SPEED.MAIN_GO and BUS_SPEED itself are now complete.

5.2.6 Converted to link travel time

Up to now we have described how to generate bus poll segments and match them with CMAP links. Effectively, each bus poll segment, when matched with a link in the CMAP network, provides an observation of the average speed on that link at the polling time. To facilitate a pilot study of these data, CTA incorporated the above extraction procedure in their daily database operation between 10/09/2009 and 11/13/2009, which produces a bus poll segment speed table with about 28 million records. The speed data in this table are then converted to travel times using the following procedures:

1. Increase the observed average bus speed by 5 mph to make up for the difference between bus and car, based on the recommendation in the literature (Pu 2008).

2. Take the average of the poll start and end time as the nominal time of measurement.

3. Calculate the average travel time by dividing the length of the link by the inflated bus poll speed.

The processed link travel time data are stored in a MYSQL database that also hosts the GCM data.

Bus data usually have poorer temporal coverage compared to the loop detector data. A location may only have a few valid bus speed observations for a given day, while the loop detectors usually record travel speeds once per 5 minutes (288 data points per day). To resolve this issue, a postprocess is introduced to generate the entire travel time profile at each location that have bus observations. We first divide a day into 288 5-minute time periods. For each time period i, if bus travel time observations exist, then the average of all observations are used as the nominal travel time for the period. If not, the travel time is obtained by $t_i = \lambda t_{i-1} + (1-\lambda)t0$. where t_i is the nominal travel time for period i and $t0$ is the free flow travel time. In this study, λ is set to 0.9.

5.3 Travel time data collected by CTA bus probes

We now analyze the link travel time data generated from the above procedure. The analysis is focused on a specific weekday, October 15, 2009.

5.3.1 Travel times on freeway and expressways

CTA bus routes cover sections of freeways and expressways in the area, such as Lake Shore Drive and I-290 (near downtown). For these roads, travel time data from both the loop detectors and CTA bus probes are available. In total, there are over three hundred such road links, from which we selected a small sample, including three links from Lake Shore Drive and two from I-290, to verify the quality of the bus data. The locations of these five links are shown in Figure 5.4.

The link travel times recorded by loop detectors and bus probes on a specific weekday (October, 15, 2009) are compared in Figures 5.5 - 5.9. From these figures, we can see that in most cases both sets of travel time data reflect the morning peak traffic congestion, although bus data evidently have much more noises. For the two links on Lake Shore Drive (17842 and 13210), the recorded travel times from loop detectors and bus probes are remarkably consistent during the peak periods. For links 1969 and 3025, bus data did capture peak periods but significant differences exist in the overall travel time profile between the two data sets. The bus data on link 7081 seem problematic. First note that the bus coverage is poor at this location. Second, for those limited readings, the data suggest that the bus consistently runs 10 times slower than the traffic during much of the off periods. We postulate that this problem is likely to be caused by link mismatch (matching a non-freeway link to a freeway link).

When there is no traffic congestion, the travel times recorded by loop detectors are smaller than those recorded by bus probes (remember we have inflated the observed bus speeds by 5 mph when we generated our bus travel time database). This observation suggests that (1) buses usually run at speeds close to or lower than the free flow travel speed (speed limit) but other

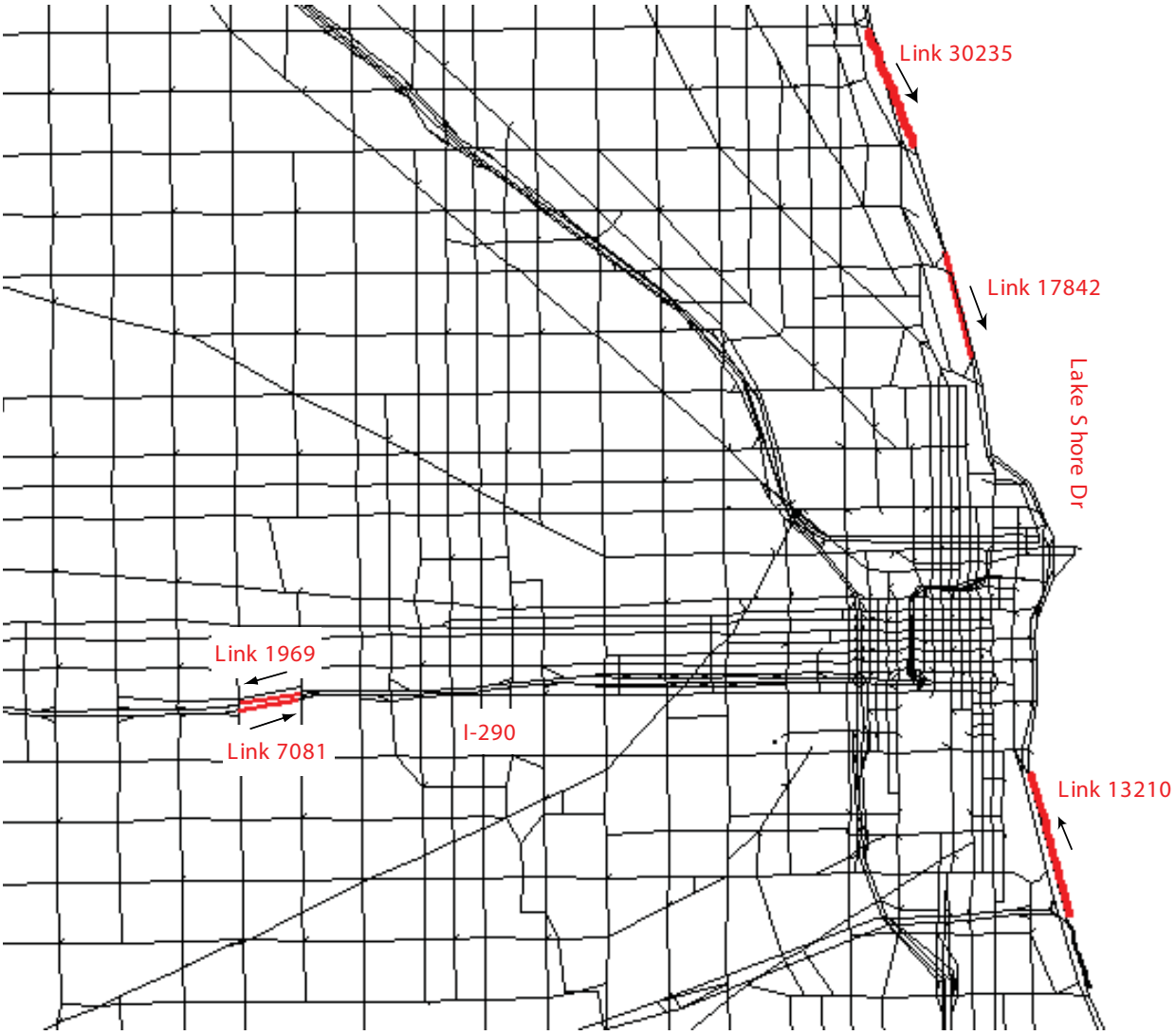


Figure 5.4: Locations of the selected links belonging to Lake Shore Drive and I-290

vehicles travel at speeds higher than the speed limit, when there is no traffic congestion; and (2) the speed difference between bus and car is likely to be more than 5 mph in the free flow traffic. When the road is congested, however, every vehicle, including buses, is delayed and almost travels at the same speed, as shown in Figures 5.7 and 5.5. In summary, with sufficient coverage, bus data may capture heavy congestion on freeway and expressway but proper adjustments must be made in the other conditions.

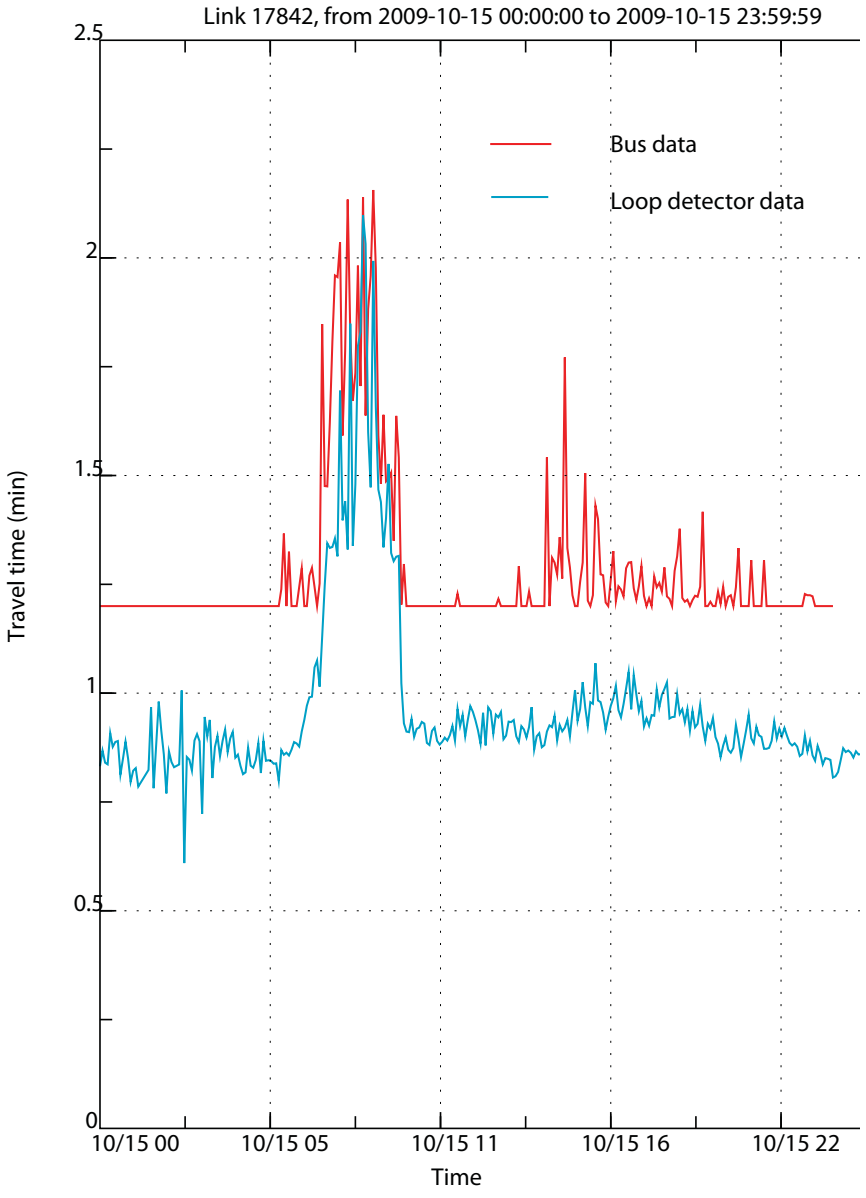


Figure 5.5: Travel times of link 17842 located in Lake Shore Drive on 10/15/2009, recorded by loop detectors and bus probes

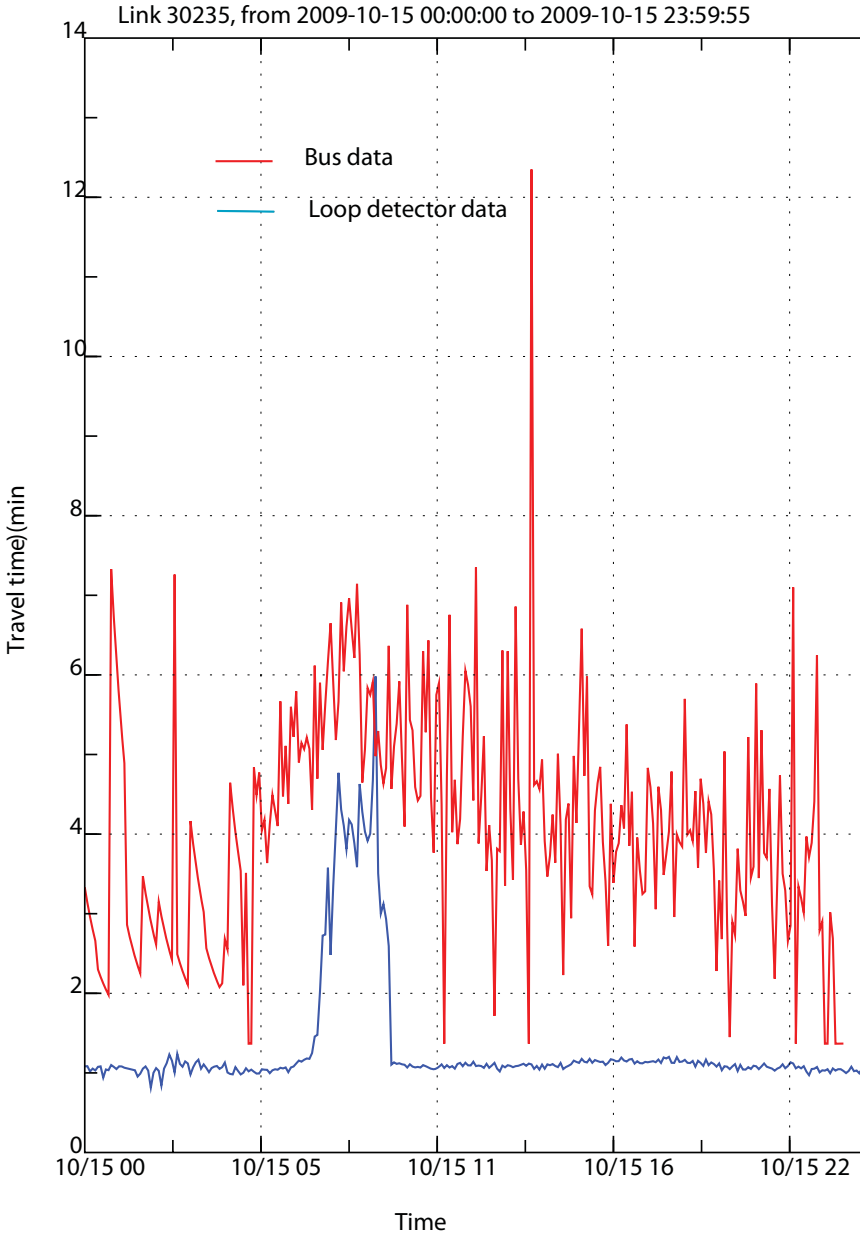


Figure 5.6: Travel times of link 30235 located in Lake Shore Drive on 10/15/2009, recorded by loop detectors and bus probes

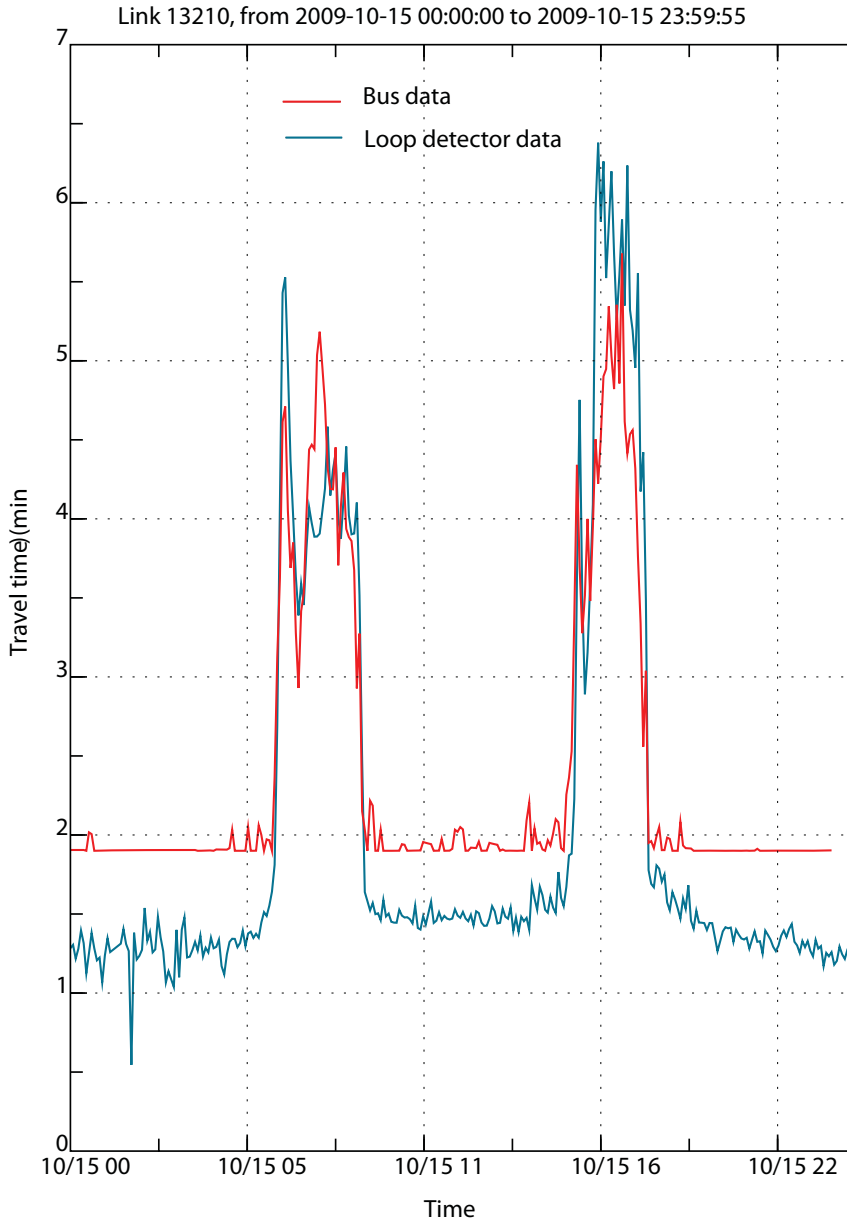


Figure 5.7: Travel times of link 13210 located in Lake Shore Drive on 10/15/2009, recorded by loop detectors and bus probes

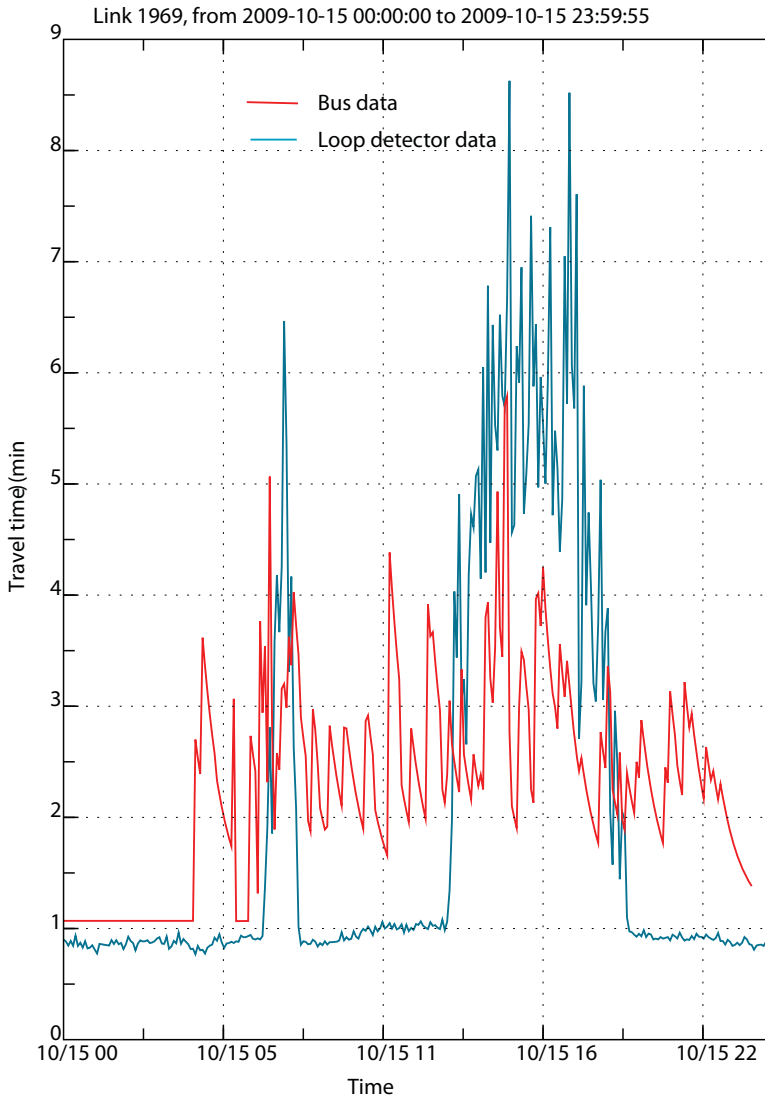


Figure 5.8: Travel times of link 1969 located in I-290 on 10/15/2009, recorded by loop detectors and bus probes

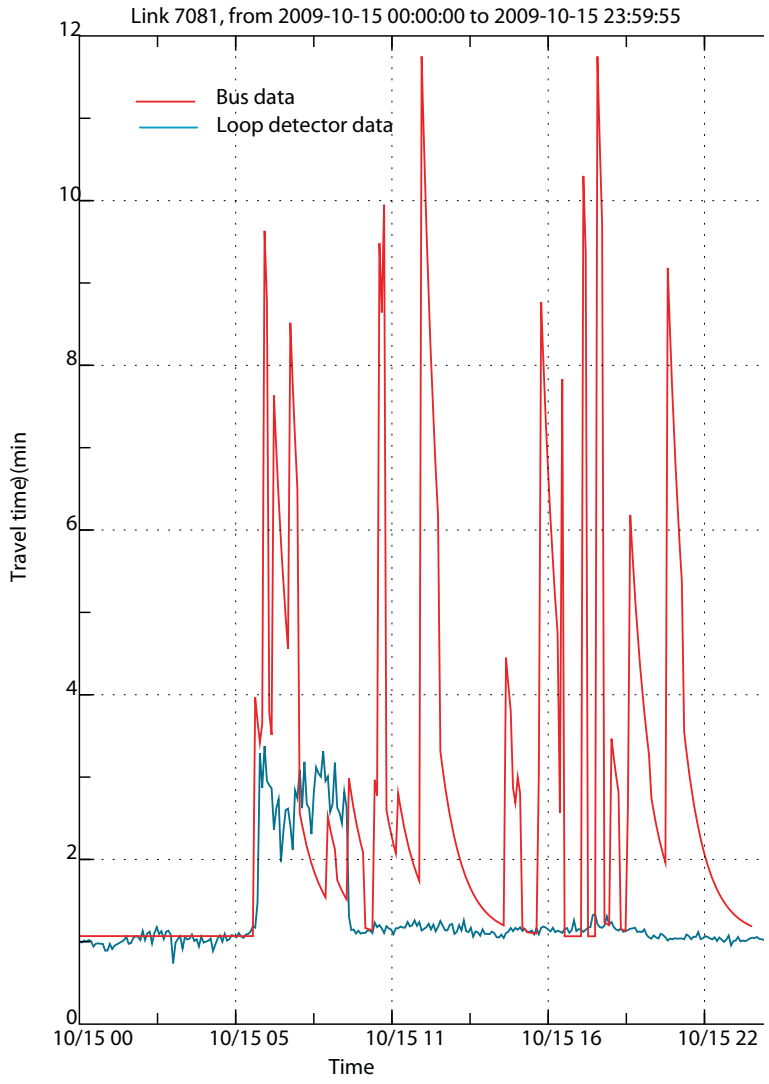


Figure 5.9: Travel times of link 7081 located in I-290 on 10/15/2009, recorded by loop detectors and bus probes

5.3.2 Travel times on arterial and local streets located in suburbs

In this section, 12 links on six arterial roads in the north, west and south suburbs of Chicago are examined. For each arterial road, a pair of links (in opposite directions) are selected. In the north, we select West Montrose Avenue and West Lawrence (Figure 5.10(a)). In the west, West Madison Street and West Roosevelt Road are selected, both of which are parallel to the links of I-290 studied in the last section (Figures 5.11(a)). In the south, West 47th Street and South Halsted Street are selected (Figure 5.12(a)). The lengths of the selected links are no shorter than half miles in order to reduce the impacts of bus stops and signals.

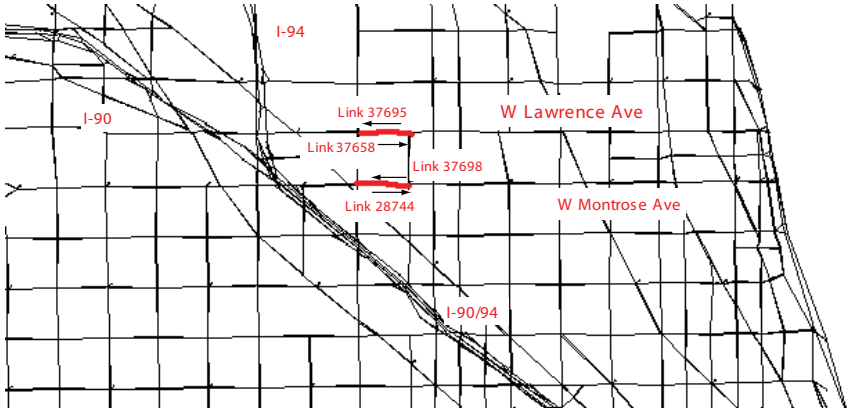
The travel time profiles are given in Figures 5.10(b), 5.11(b) and 5.12(b). Distinctive morning peak periods can be observed from the plots for links 37698, 37685, 34137, 40767, 22269, 39545 and 41457. We can also see evening peak periods from the data on links 40767, 10907 and 39545, and the mid-of-day peak period from the data of links 37698, 39545, 41455 and 18036. However, no typical pattern emerge from the data for links 28744, 37695 and 37658.

For the purpose of comparison, four shorter links on local streets in these suburbs are also examined. The locations of links are shown in Figure 5.13(a), and the recorded travel times are reported in Figure 5.13(b). As expected, while some plots do reveal the peak-time pattern (e.g. link 1891), the quality of the data degrades in general as the noises and lack of temporal coverage become worse due to the short link length.

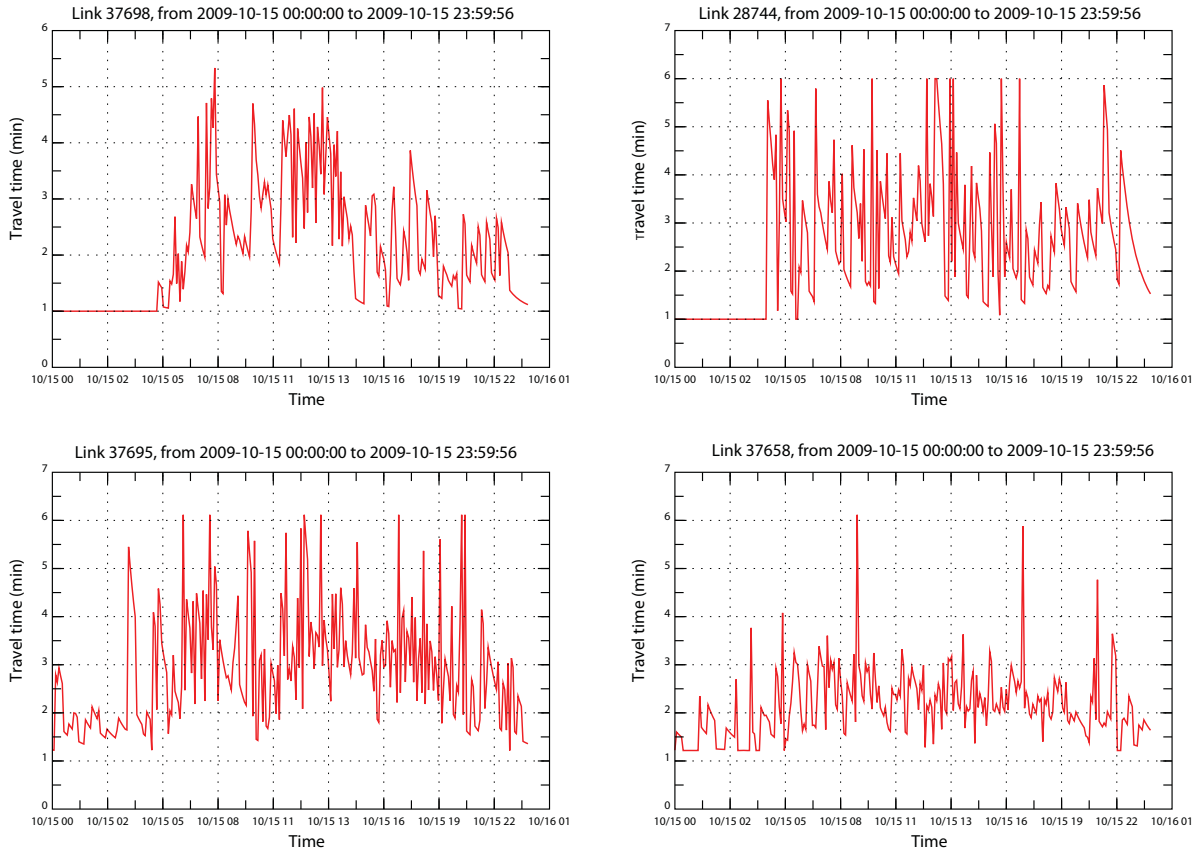
To summarize, for long links located in suburbs, the data from bus probes reasonably reflect congestion patterns. This is expected because buses can travel a relatively long distance on such links without disruptions from stops and intersections. Also, note that the demand for bus services in the suburb area is relatively low compared to the downtown area, which helps reduce delays associated with stops (note that these delays cannot be excluded from the data).

5.3.3 Travel times on arterial and local streets in Downtown Chicago

We examine a sample of streets located in Downtown Chicago. In total four links on Michigan Avenue (a major arterial street) and two links from two minor local one-way streets (South



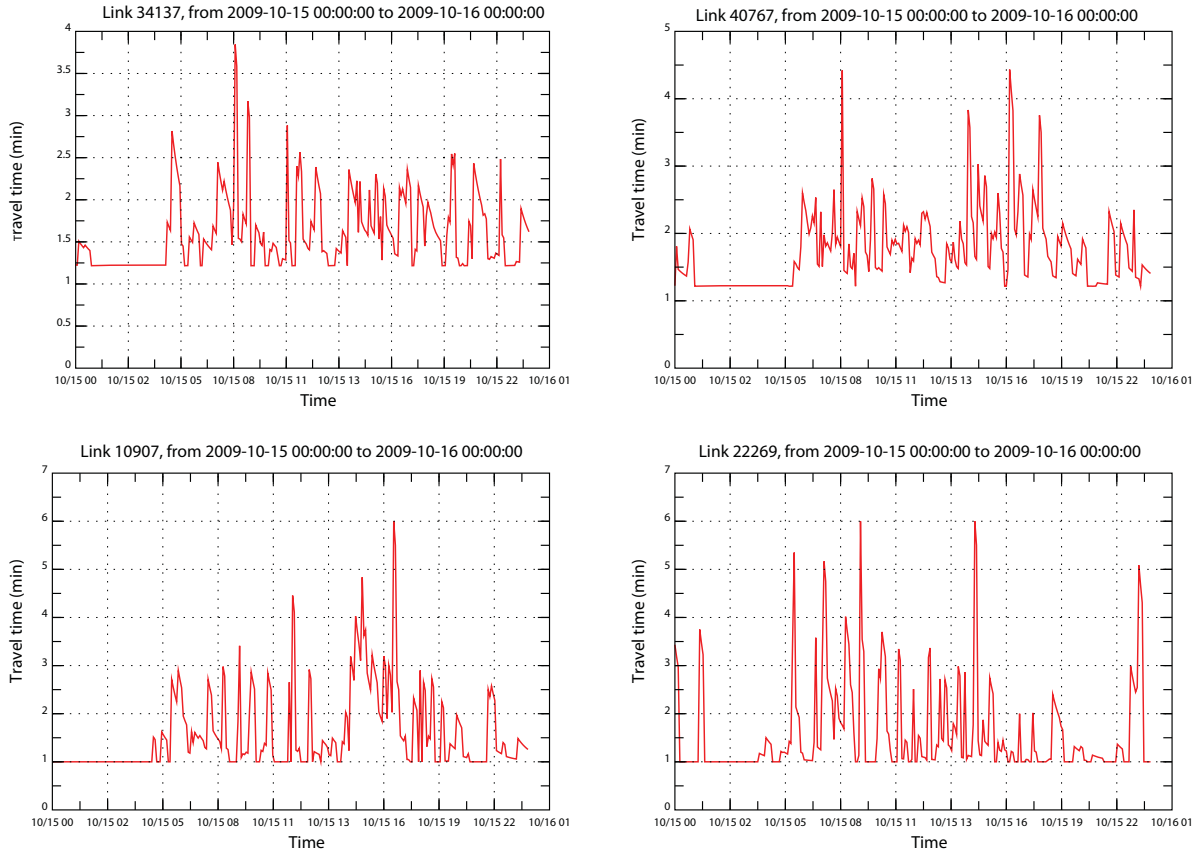
(a) Locations of selected links



(b) Recorded link travel times

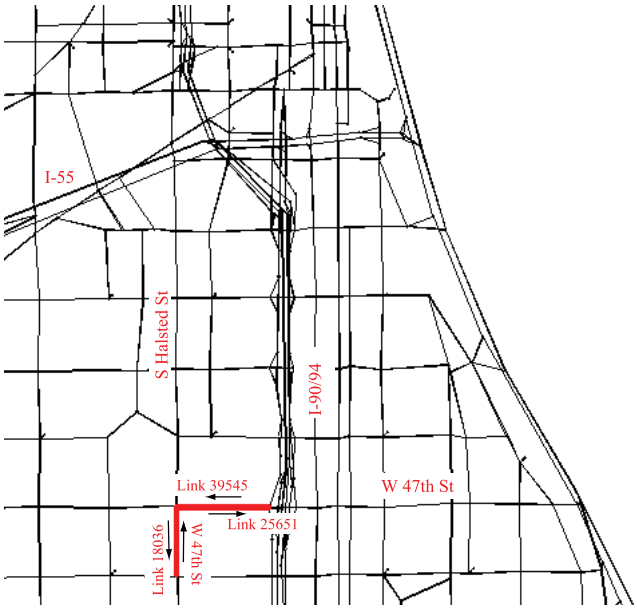
Figure 5.10: Locations and travel times of links 37695, 37658, 37698 and 28744 located on W. Montrose Ave. and W. Lawrence Ave. on 10/15/2009

(a) Locations of selected links

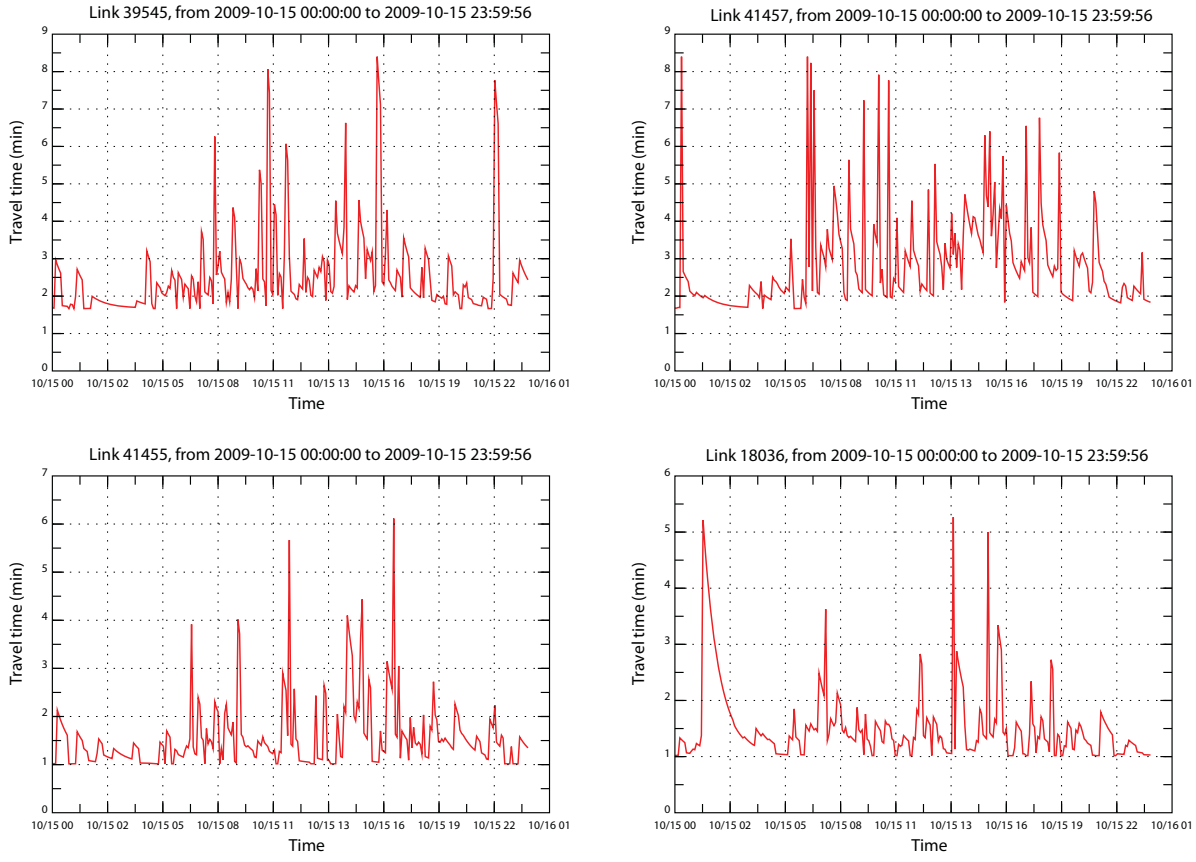


(b) Recorded link travel times

Figure 5.11: Locations and travel times of links 34137, 40767, 7060 and 25458 located on W. Madison St. and W. Roosevelt Rd. on 10/15/2009

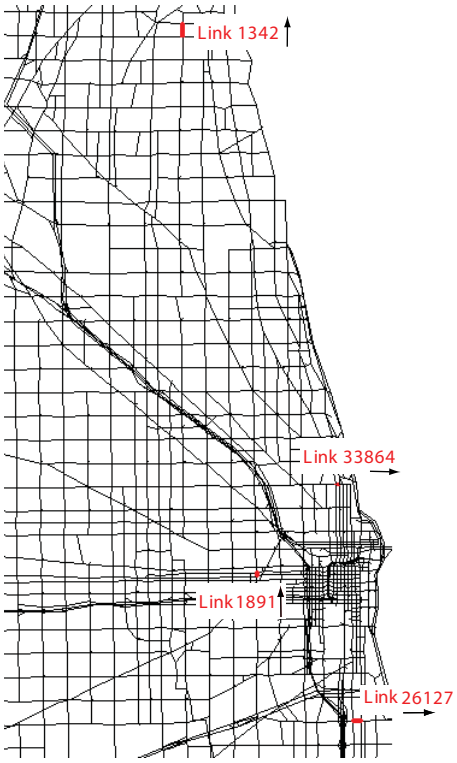


(a) Locations of selected links

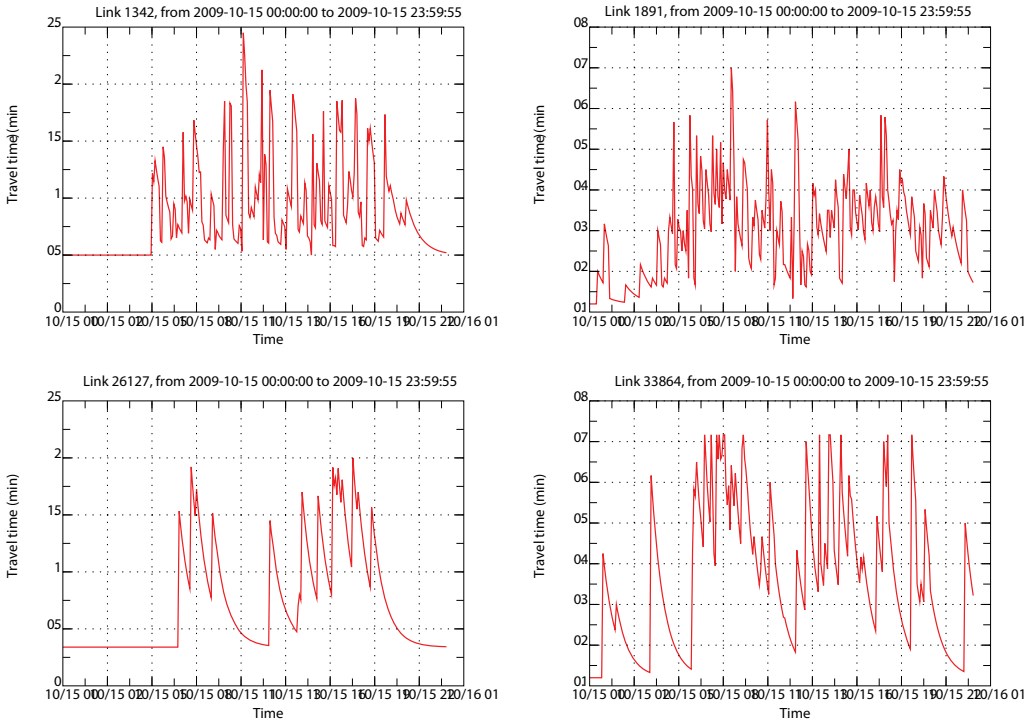


(b) Recorded link travel times

Figure 5.12: Locations and travel times of links 39545, 25651, 41457 and 41455 located on W. 47st St. and S. Halsted Rd. on 10/15/2009



(a) Locations of selected links



(b) Recorded link travel times

Figure 5.13: Locations and travel times of links 1342, 1891, 26127 and 33864 on 10 / 15 / 2009

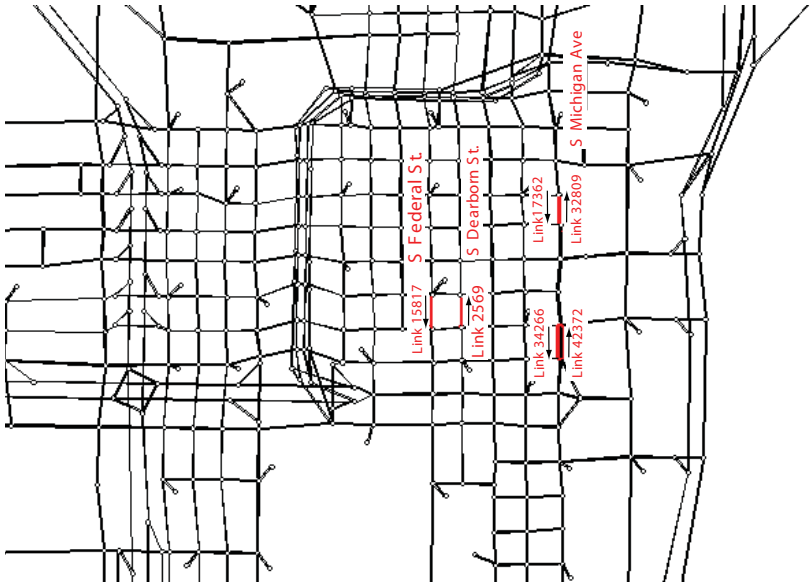


Figure 5.14: Locations of selected links in Chicago Downtown

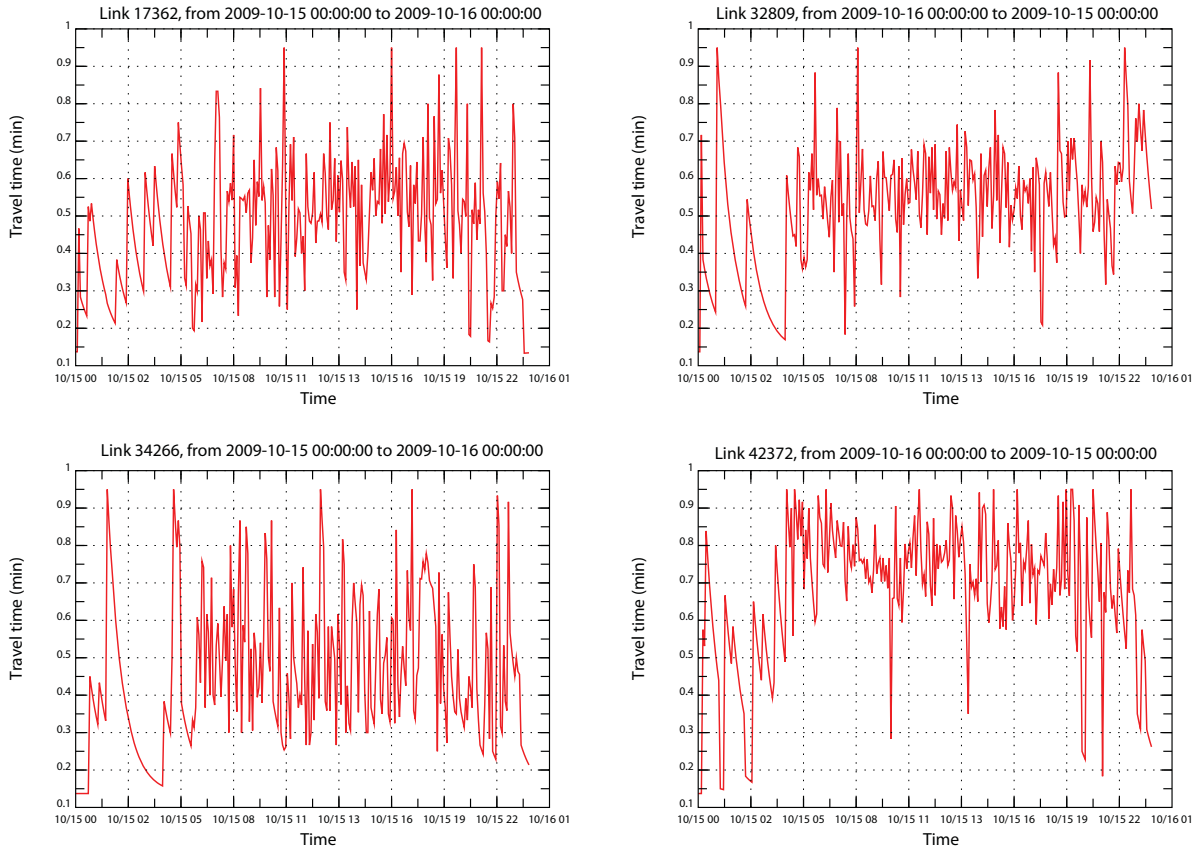
Federal Street and South Dearborn Street) are selected. The location of selected links are shown in Figure 5.14. The recorded travel times are reported in Figures 5.15(a) and 5.15(b).

Two observations can be made from Figures 5.15(a) and 5.15(b). First, travel times experienced by buses are roughly 3 - 4 times higher than the free flow travel time on these links throughout the day. This is likely due to frequent bus stops and intersections that prevent buses from operating at normal speeds. Second, the noises in the data are so strong that it is very difficult to distinguish peak and off-peak periods. Hence, using these data for the purpose of travel time estimation may be inappropriate.

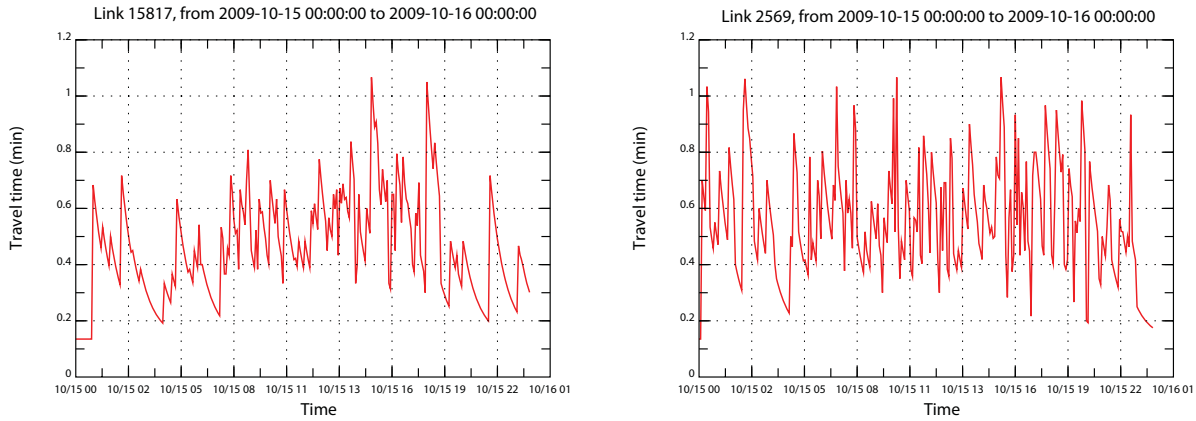
5.4 Summary

In this chapter, we present and test the methodology used to extract travel time data from archived bus AVL data. We use bus poll segments as "virtual speed detectors" for the matched links in the CMAP network. Link travel times are then obtained through a simple postprocess. Our main findings from examining a one-month sample of CTA bus travel time data are:

- The bus travel time data have strong correlations with those obtained from loop detectors on



(a) Recorded link travel times of links located in Michigan Avenue



(b) Recorded link travel times of links located in minor local streets

Figure 5.15: Travel times of links located in S. Michigan Ave., S. Dearborn St. and S. Federal St. on 10/15/2009

freeways and expressways. While the speeds of buses and cars do seem to have significant discrepancies under free-flow / light-congested conditions, they tend to agree with each other much better when congestion is heavy.

- The data on arterial and locals street cannot be directly validated. However, they typically contain larger noises, likely resulted from the disruptions of bus stops and intersections.
- In general, the data quality is better on longer streets than shorter streets, and better on streets located in suburbs than those in the downtown area. On the long streets in suburbs, the bus data are able to reveal distinctive peak and off-peak periods in many cases.

Chapter 6

Travel Reliability Survey

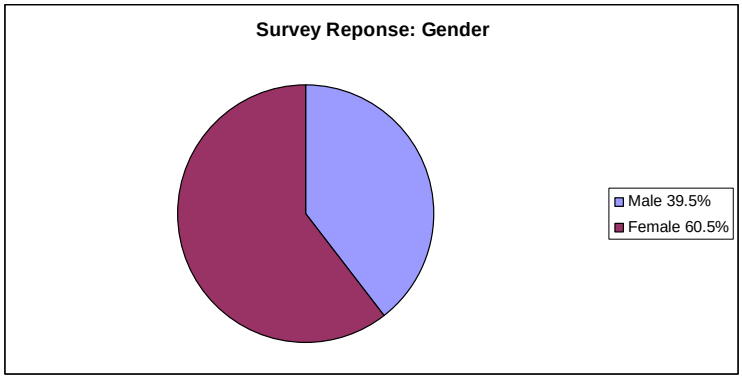
The travel reliability survey, which contains 22 questions (see Appendix), intends to collect three types of data: demographical and basic travel-related information, reaction and attitude to travel reliability, and opinions about potential reliable route guidance products. The target population is students and workers living in the Chicago Metropolitan area. We exclude non-workers because they are deemed as having more flexible schedules and hence being less concerned about travel reliability. To ensure high response rate, a professional web survey company uSample inc. was hired to recruit responders. The research team designed the survey using Survey Monkey and the link to the survey was provided to uSample. The company then reached out to responders from various online sources, including social media, hand-picked affiliate sites, and web advertisements. It took less than a week to obtain a sample of 228 filled surveys, of which 220 have consistently answered all the questions, while 8 dropped out after question 2 primarily because they do not drive. In the following, the survey results from each of the three categories are presented and analyzed.

6.1 Demographical and basic travel-related information

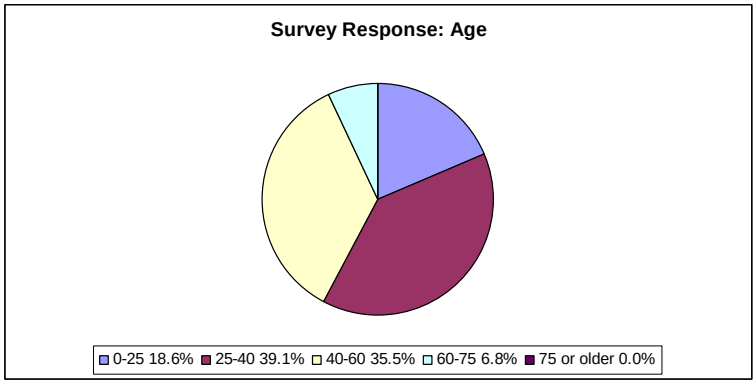
6.1.1 Demographical information

Overall, the majority of the people who have filled the survey were in the age range from 25 to 60. Specifically, 39% were in the age range 25-40, and 35.5% were in the range 40-60. About 60.5% of responders are females (see Figures 6.1(a) and 6.1(b)). Of all the responders about 12.7%

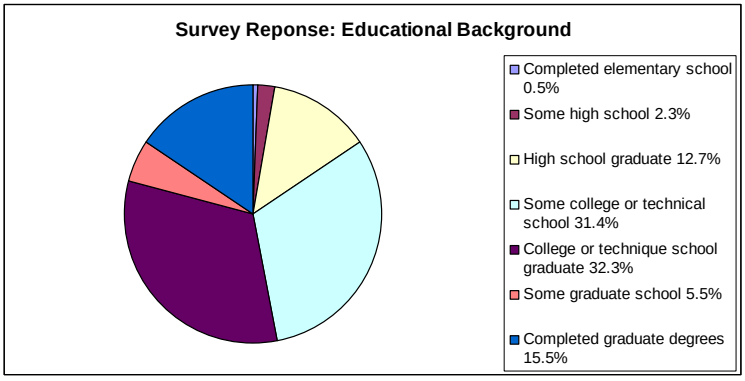
have only completed high school, 32.3% have completed college or technical school and 15.5% were shown to have completed graduate school (see Figure 6.1(c)).



(a)



(b)



(c)

Figure 6.1: Distributions of gender, age and education background of the motorists who participated in our survey

6.1.2 Basic travel-related information

97.3% indicated that there is a public transportation system that runs through their neighborhood (see Figure 6.2(a)). However, less than a half (46.7%) of the people felt that the public transportation system is reliable, and only 37.1% of responders use public transportation for commute.

6.2 Reaction and attitude to travel reliability

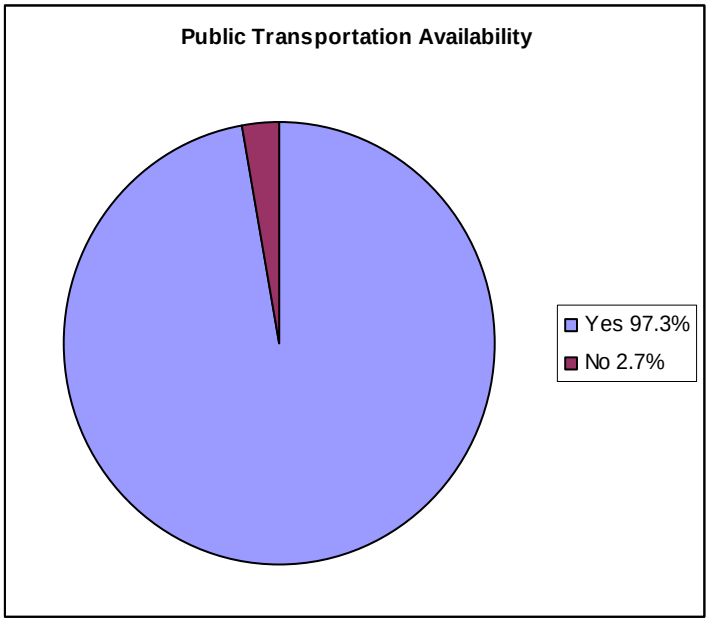
6.2.1 Current methods used by commuters

Our responders heavily rely on either the past experience or the internet for route guidance (see Figure 6.3): 57.3% have indicated that they obtain driving direction from Google / Yahoo, and 58% said that they make route choice based on their own experience. In addition, 30.5% of the responders use traffic reports provided by radio / TV, and only 23.6% said they rely on the in-vehicle navigation system. We note that the majority of in-vehicle navigation users are from the age group 25-40, which is expected as the technology is relatively new. Moreover, the majority of the responders seem to be happy with their current routing methods (see Figure 6.4).

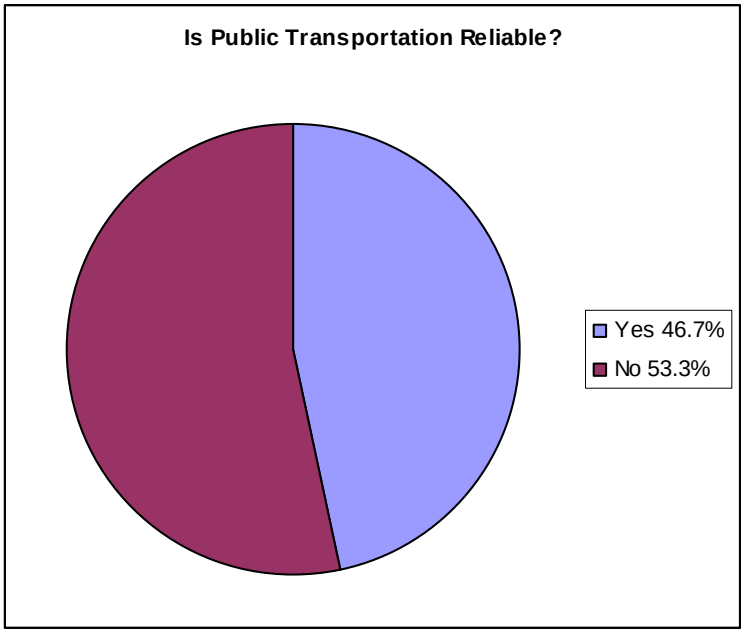
6.2.2 Importance of travel time reliability

It is somewhat surprising that the responders perceive “arriving on time” as important for almost all hypothetical trips given in the question, see Figure 6.5. The lowest rating is for “attending a party”, which still receive an average importance rating of 3.5 / 5. Maybe “being on time ” is perceived as such good etiquette that the responders tend to inflate their stated importance. Nevertheless, significant differences do exist between different trips. The most important trip that has received highest ratings is job interview (4.74). Going to work or airport for a flight are the second most important events, both with rating slightly over 4.5.

An overwhelming majorit of the responders, over 80%, said that time is the most important factor in their route choice. At the second place is travel reliability, which influences route choices among more than 40% of all responders. The other variables are cost (31.5%), comfort (24.7%),



(a)



(b)

Figure 6.2: Access to public transportation and its reliability

safety (21.0%), and emission (1.4%), see Figure 6.6. It is interesting to notice that more responders concern comfort than safety, which seems counter-intuitive. A possible explanation is that life-threatening accidents are rare events whereas comfort is a daily experience.

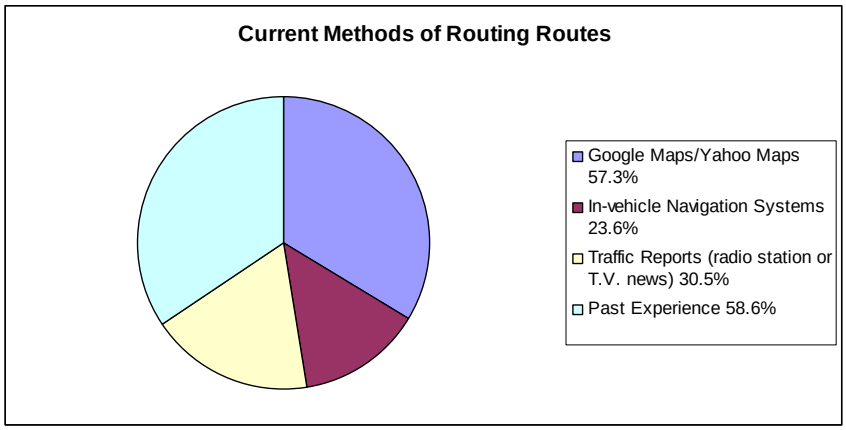


Figure 6.3: Distribution of the current methods of routing employed by commuters

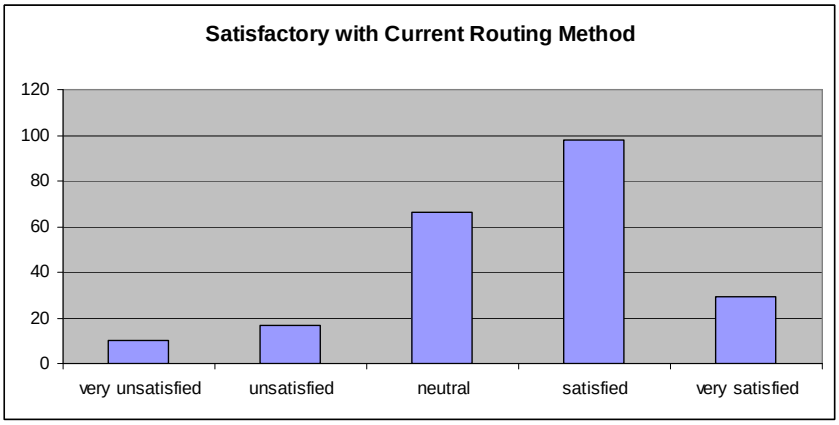


Figure 6.4: Distribution of the satisfactory of commuters with current routing method

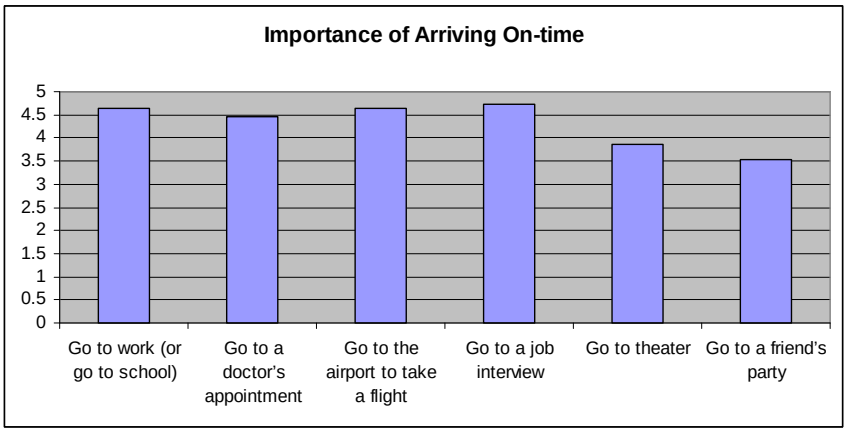


Figure 6.5: Importance of arriving on-time for various trips with different purposes

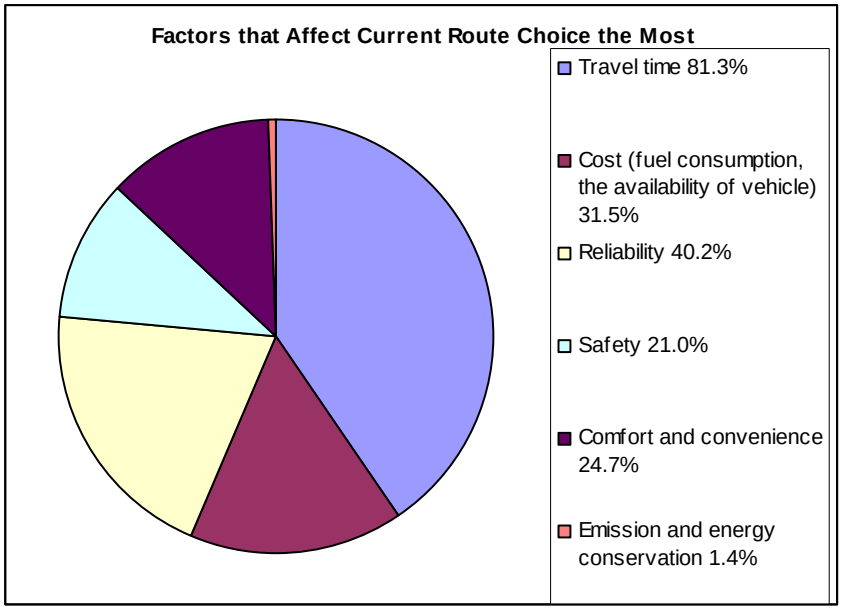


Figure 6.6: Distribution of the factors which impact the route choice most

6.2.3 Travel time budget

Most commuters indicated that they would plan more time than normally required for important trips. When the commuters were faced with a question on how much time they would budget for a trip that would take on average of 30 minutes and 60 minutes with traffic to the airport for an 8:30 AM flight, 68.5% indicated that they will budget more than 60 minutes for the trip. Overall, 94.5% indicated that they will budget at least more than 35 minutes for the trip. When given question on how much they will budget for daily trips to work that would take 30 minutes on average (Question 15), 85.3% indicated that they will budget at least 35 minutes or more for the trip, while 49.8% indicated that they will budget at least 45 minutes or more, i.e., 50% longer than the mean travel time. 12.8% answered that they will budget more than an hour for the trip (see Figure 6.7). These responses clearly verify the behavior assumption made in our reliable routing model: that people would budget a buffer time to ensure a high probability of on-time arrival.

Moreover, 39.7% of all responders indicated that the issue of travel time reliability concerns

them almost every day. When asked about what type of reliability information they prefer, 37.4% chose the average trip time and the standard deviation, and 20.4% chose the histogram plot of travel times.

Finally, when the commuters were asked about "what do you do to ensure you arrive on time?", the highest average rating (3.89 out of 5) was given to "reserving more time than needed" and the second highest rating (3.24 out of 5) was given to "getting real time traffic information".

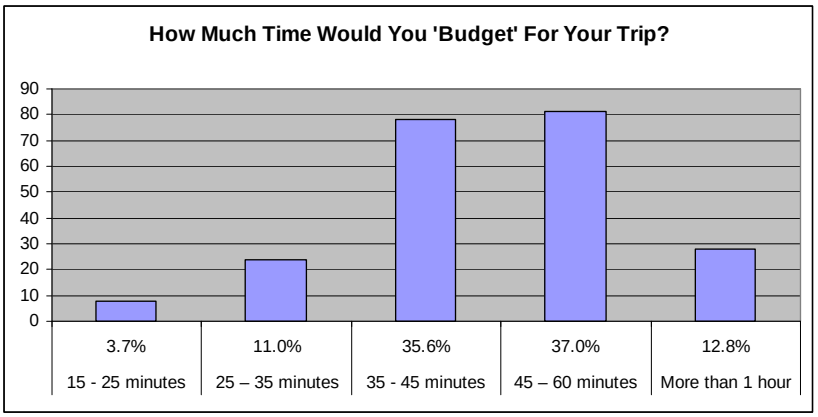


Figure 6.7: Time budget for trips

6.3 Opinions about reliable route guidance products

In the survey, the responders are given a hypothetical scenario where they have access to a reliable route guidance product, CTR, which may suggest a "reliable" but unfamiliar route for their work trips. The responders are then asked how much time savings produced by CTR would induce them to switch routes. As shown in Figure 6.8, about 40% of all responders require 10-20% or less time savings to take the new reliable route. About 35% of commuters require 30% or more. Most users would not care the difference if the time saving is no greater than 10%.

When asked about the running time of CTR, most people indicated that they are either neutral or satisfied with the fact that CTR needs about 30 seconds to 60 seconds to generate a reliable route guidance. We note that a commercialized product would achieve a much better computation speed through various computing technologies.

Finally, many people would like to receive a reliable route guidance product as an added-on function in Google or Yahoo maps (48.7%). Many others would like to have it on their smart phones (37.7%) or in-vehicle navigation systems (30.7%). The details are given in Figure 6.10.

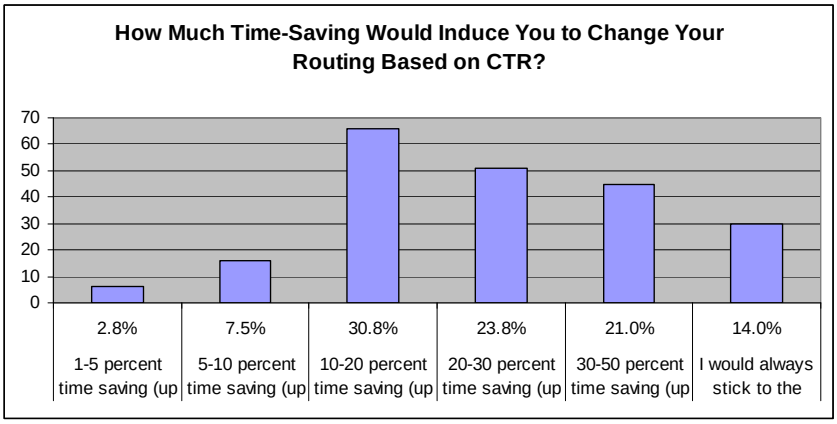


Figure 6.8: Expectation of time saving by using CTR in route guidance

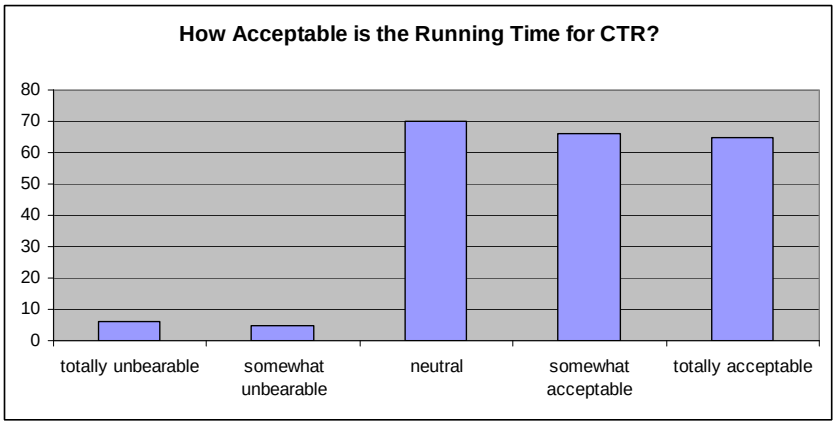


Figure 6.9: Evaluation of CTR running time

6.4 Summary

The preliminary findings from the survey are summarized as follows:

- Travel reliability is found to be the second most important decision variable in route choice, next only to travel time. Over 40% of all responders believe that their route choices are affected by travel reliability.

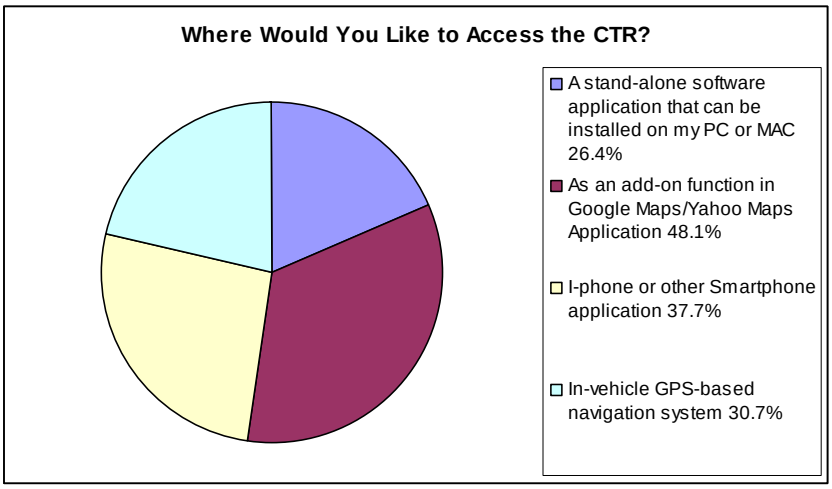


Figure 6.10: Expectation of CTR in the future application

- The survey results confirm that commuters do budget a buffer time to ensure a high probability of on-time arrival for important trips, and that they perceive this as the most important approach to improving reliability.
- Reliable route guidance products need to generate more than 10% time savings in order to be attractive.
- The most desirable commercialization forms are to integrate the technology into the existing Internet-based map services (such as Google / Yahoo Maps) and to develop application for smart phones.

Chapter 7

Conclusions

We first summarize what have been accomplished in this project, and then set possible directions for further research.

7.1 Main results

This project further develops the reliable routing algorithm in two aspects. First, we proposed a new convolution method based on Discrete Fourier Transform (DFT). While the DFT-based convolution method can produce highly precise solutions, it significantly increases computation overhead. Compared with DFT-based convolution schemes, the direct convolution scheme based on adaptive discretization seem to strike a better balance between accuracy and efficiency, particulary for large-scale applications. Second, we employ the higher order stochastic dominance (SD) rules to solve the reliable routing problem. We show that the higher order SD rules can reduce the number of non-dominant paths, which promises to improve the computational efficiency. Unlike the existing heuristic methods, the proposed approach guarantees that all enumerated paths are FSD-admissible. Our results indicate that the optimal solutions solved with the second-order stochastic dominance (SSD) are close to those solved with first-order SD, or FSD. On the other hand, the CPU time required to solve the reliable shortest path problem is roughly reduced a factor of two when the SSD-rule is employed.

The project tests the idea of using the Chicago Transit Authority (CTA) buses as traffic probes.

A prototype PL/SQL code was developed to extract travel speed information from AVL bus data and to match them to links in the CMAP network. In total, a sample of about one month of all matched bus data (about 28 million records) is generated and analyzed in the study. The results indicate that (1) the bus travel time data have strong correlations with those obtained from loop detectors on freeways; (2) the data on arterial and locals street typically contain larger noises; (3) the arterial data quality is better on longer streets than shorter streets, and better on streets located in suburbs than those in the downtown area; and (4) the bus data tend to overestimate travel time under free flow or light congestion conditions, but better represent the reality in the presence of heavy congestion. Many of these findings are expected. Note that the noises in the arterial data are likely due to the impacts of bus stops and intersections, which are clearly larger when the link is short and/or located in the downtown area. The lack of sufficient data coverage represents another obstacle. Results indicate that many arterial links, although served by CTA bus routes, only have a few observations each day, which is far from enough to create a reliable daily travel time profile. Finally, even when they are not disrupted by stops, intersections and congestion, buses and cars are likely to travel at quite different speeds on average. Our results indicate that under free flow condition the speed difference between buses and cars is likely to be well over 5 mile/hour in the area.

We conduct a survey designed to understand travelers' reaction and attitude to travel reliability and their opinion about reliable route guidance products. Through a professional web survey company, we obtained a sample of 220 valid responses from students and workers in the Chicago Metropolitan area. We found that travel reliability is the second most important decision variable in route choice, next only to travel time. The survey results also confirm that commuters do budget a buffer time to ensure a high probability of on-time arrival for important trips, and that they perceive this as the most important approach to improving reliability. However, to be attractive to most travelers, a reliable route guidance has to generate at least 10 - 20% time savings. Finally, the most desirable forms to commercialize the reliable routing technology include

integrating it into the existing Internet-based map services (such as Google / Yahoo Maps) and developing smart-phones-based applications.

7.2 Future work

Our analysis and experiments suggest that the DFT-based convolution method is not as computationally competitive as its theoretical complexity promises due to several implementation issues. The first issue was discussed in Section 3.3, which has to do with the requirement of equal discrete support intervals. Because our problem often involves convoluting two distributions with quite different ranges, this requirement dramatically increases the computational burden. The second and more important issue is that checking stochastic dominance is not as straightforward in the frequency domain as in the time domain. Consequently, the distributions have to be transformed to the frequency domain to compute convolution and then transformed back the time-domain to check dominance. Note that these transformations will take place whenever a path is augmented by one link, which is highly inefficient. The ideal strategy would be always performing the convolution in the frequency domains, and only transform back to the time domain after all path travel time distributions are obtained. To avoid path enumeration, however, it requires developing the stochastic dominance relationship in the frequency domain. Fully realizing the potential of DFT-based method is an interesting direction for further research.

The use of CTA bus data in support of reliable routing is worth of further investigations. Note that our current analysis only considers time-of-day travel time profiles. To be used in the reliable routing, these data have to be aggregated to generate the travel time distributions for each period (e.g, morning rush hour, mid-of-day, etc). We expect that the data aggregation is likely to reduce random noises and the relationship between bus and loop detectors can be more easily constructed in the form of travel time distributions (instead of daily travel time profile). Furthermore, validation studies are needed for arterial links that are short and / or located in the downtown area. The key is to obtain data on real traffic conditions through other sources, such

as GPS-equipped passenger cars and trucks.

This study provides only a preliminary analysis of the survey results. It is interesting to develop some econometric models in order to better understand how different factors (gender, education, age, risk-aversion) might affect travelers' attitudes to travel reliability.

Bibliography

Abdel-Aty, M., Kitamura, R. & Jovanis, P. (1997), 'Using state preference data for studying the effect of advanced traffic information on drivers' route choice.', *Transportation Reseach* **5C**, 39–50.

Bander, J. L. & White, C. C. (2002), 'A heuristic search approach for a nonstationary stochastic shortest path problem with terminal cost', *Transportation Science* **36**(2), 218–230.

Bard, J. & Bennett, J. (1991), 'Arc reduction and path preference in stochastic acyclic networks', *Management Science* **37(2)**, 198–215.

Bawa, V. S. (1975), 'Optimal rules for ordering uncertian prospects', *Financial Economics* **2**, 95–121.

Bawa, V. S., Bondurtham, J., Rao, R. & Suri, H. L. (1983), 'On determination of stochastic dominance optimal set', *Journal of Finance* **40**, 417–431.

Bertsimas, D. & Sim, M. (2003), 'Robust discrete optimization and network flow', *Mathematical Programming Series B* **98**, 49–71.

Bracewell, R. (2000), *The Fourier Transform and Its Applications*, 3rd edn, McGraw-Hill, Boston.

Carraway, R. L., Morin, T. L. & Moskowitz, H. (1990), 'Generalized dynamic programming for multicriteria optimization', *European Journal of Operational Research* **44**(1), 95–104.

Chen, Y., Bell, M. G. H., Wang, D. & Bogenberger, K. (2006), 'Risk-averse time-dependent route guidance by constrained dynamic a* search in decentralized system architecture', *Transportation Research Record* **1944**, 51–57.

Cheung, R. K. (1998), 'Iterative methods for dynamic stochastic shortest path problems', *Naval Research Logistics* **45**(8), 769–789.

Cooley, J. W. & Tukey, J. W. (1965), 'An algorithm for the machine calculation of complex fourier series.', *Mathematics of Computation* **19**, 297 – 301.

Cormen, T., Leiserson, C., Rivest, R. & Stein, C. (2001), *Introduction to Algorithms*, MIT Press, Cambridge, MA.

Dentcheva, D. & Ruszczynski, A. (2003), 'Optimization with stochastic dominance constraints', *SIAM Journal of Optimization* **14**(2), 548–566.

Eiger, A., Mirchandani, P. B. & Soroush, H. (1985), 'Path preferences and optimal paths in probabilistic networks', *Transportation Science* **19**(1), 75–84.

Fan, Y., Kalaba, R. & Moore, J. (2005*a*), 'Arriving on time', *Journal of Optimization Theory and Applications* **127**(3), 497–513.

Fan, Y., Kalaba, R. & Moore, J. (2005*b*), 'Shortest paths in stochastic networks with correlated link costs', *Computers and Mathematics with Applications* **49**(9-10), 1549–1564.

FHWA (2000), *Traffic Incident Management Handbook*, Federal Highway Administration.

Frank, H. (1969), 'Shortest paths in probabilistic graphs', *Operations Research* **17**(4), 583–599.

Friedman, M. & Savage, L. (1948), 'The utility analysis of choices involving risk', *Journal of Political Economy* **56**, 279–304.

Fu, L. (2001), 'An adaptive routing algorithm for in-vehicle route guidance systems with real-time information', *Transportation Research Part B* **35**(8), 749–765.

Fu, L. & Rilett, L. R. (1998), 'Expected shortest paths in dynamic and stochastic traffic networks', *Transportation Research Part B* **32**(7), 499–516.

Gao, S. & Chabini, I. (2006), 'Optimal routing policy problems in stochastic time-dependent networks', *Transportation Research Part B* **40**(2), 93–122.

Hadar, J. & Russell, W. R. (1971), 'Stochastic dominance and diversification', *Journal of Economic Theory* **3**, 288–305.

Hall, R. W. (1986), 'The fastest path through a network with random time-dependent travel time', *Transportation Science* **20**(3), 182–188.

Hanoch, G. & Levy, H. (1969), 'The efficiency analysis of choices involving risk', *Review of Economics studies* **36**, 335–346.

Heyer, D. D. (2001), Stochastic dominance: a tool for evaluating reinsurance alternatives, *in* 'Casualty Actuarial Society Forum'.

Kaparias, I., Bell, M., Chen, Y. & Bogenberger, K. (2007), 'Icnavs: a tool for reliable dynamic route guidance', *IET Intellegent Transportation Systems* **1**(4), 225–253.

Levy, H. & Hanoch, G. (1970), 'Relative effectiveness of efficiency criteria for portfolio selection', *Journal of Financial and Quantitative Analysis* **5**, 63–76.

Liu, H. X., Recker, W. & Chen, A. (2004), 'Uncovering the contribution of travel time reliability to dynamic route choice using real-time loop data', *Transportation Research Part A* **38**(6), 435–453.

Loui, R. P. (1983), 'Optimal paths in graphs with stochastic or multidimensional weights', *Communications of the ACM* **26**(9), 670–676.

Miller-Hooks, E. (1997), Optimal Routing in Time-Varying, Stochastic Networks: Algorithms and Implementations, PhD thesis, Department of Civil Engineering, University of Texas at Austin.

Miller-Hooks, E. D. (2001), 'Adaptive least-expected time paths in stochastic, time-varying transportation and data networks', *Networks* **37**(1), 35–52.

Miller-Hooks, E. D. & Mahmassani, H. S. (2000), 'Least expected time paths in stochastic, time-varying transportation networks', *Transportation Science* **34**(2), 198–215.

Miller-Hooks, E. D. & Mahmassani, H. S. (2003), 'Path comparisons for a priori and time-adaptive decisions in stochastic, time-varying networks', *European Journal of Operational Research* **146**(2), 67–82.

Miller-Hooks, E. & Mahmassani, H. (1998), 'Optimal routing of hazardous materials in stochastic, time-varying transportation networks', *Transportation Research Record* **1645**, 143–151.

Mirchandani, P. B. (1976), 'Shortest distance and reliability of probabilistic networks', *Computers and Operations Research* **3**(4), 347–355.

Montemanni, R. & Gambardella, L. (2004), 'An exact algorithm for the robust shortest path problem with interval data', *Computers and Operations Research* **31**(10), 1667–1680.

Muller, A. & Stoyan, D. (2002), *Comparison Methods for Stochastic Models and Risks*, John Wiley & Sons, Chichester, UK.

Murthy, I. & Sarkar, S. (1996), 'A relaxation-based pruning technique for a class of stochastic shortest path problems', *Transportation Science* **30**(3), 220–236.

Murthy, I. & Sarkar, S. (1998), 'Stochastic shortest path problems with piecewise linear concave linear functions', *Management Science* **44**(11), 125–136.

Ng, M. & Waller, T. (2010), 'A computationally efficient methodolody to characterize travel time reliability using the fast fourier transform', *Transportation Research Part B* **44**, 1202–1219.

Nie, Y. & Wu, X. (2009), 'Shortest path problem considering on-time arrival probability', *Transportation Research Part B* **43**, 597–613.

Nie, Y., Wu, X., Nelson, P. & Dillenburg, J. (2009), Providing reliable route guidance using Chicago data, Technical Report 2009-001 1, Center for the Commercialization of the Innovative Transportation Technology, Northwestern University, Evanston, IL.

Polychronopoulos, G. H. & Tsitsiklis, J. N. (1996), 'Stochastic shortest path problems with recourse', *Networks* **27**(2), 133–143.

Provan, J. S. (2003), 'A polynomial-time algorithm to find shortest paths with recourse', *Networks* **41**(2), 115–125.

Pu, W. (2008), Bus Probe Based Urban Travel Time Prediction, PhD thesis, University of Illinois, Chicago.

Rothschild, M. & Stiglitz, J. E. (1970), 'Increasing risk. I. a definition', *Journal of Economic Theory* **2**, 225–243.

Sen, S., Pillai, R., Joshi, S. & Rathi, A. (2001), 'A mean-variance model for route guidance in advanced traveler information systems', *Transportation Science* **35**(1), 37–49.

Sigal, C. E., Alan, A., Pritsker, B. & Solberg, J. J. (1980), 'The stochastic shortest route problem', *Operations Research* **28**(5), 1122–1129.

Sivakumar, R. & Batta, R. (1994), 'The variance-constrained shortest path problem', *Transportation Science* **28**(4), 309–316.

Small, K. A., Noland, R., Chu, X. & Lewis, D. (1999), Valuation of travel-time saving and predictability in congested conditions for highway user-cost estimation., Technical report, NCHRP Report No.431, Transportation Research Board, National Research Council, USA.

Ullrich, C. (2009), *Forecasting and Hedging in the Foreign Exchange Markets*, Springer Berlin Heidelberg, chapter Preferences over Probability Distributions, pp. 117–131.

von Neumann, J. & Morgenstern, O. (1967), *Theory of Games and Economic Behavior*, Wiley, New York.

Waller, S. T. & Ziliaskopoulos, A. K. (2002), 'On the online shortest path problem with limited arc cost dependencies', *Networks* **40**(4), 216–227.

Whitmore, G. A. (1970), 'Third degree stochastic dominance', *American Economic Review* **60**, 457–459.

Wu, X. & Nie, Y. (2009), 'Implementation issues in approximate algorithms for reliable a priori shortest path problem', *Journal of the Transportation Research Board* **2091**, 51–60.

Wu, X. & Nie, Y. M. (tentatively accepted), 'Modeling heterogeneous risk-taking behavior in route choice: A stochastic dominance approach', *Transportaton Research Part B* .

Yu, G. & Yang, J. (1998), 'On the robust shortest path problem', *Computers and Operations Research* **25**(6), 457–468.

Appendix A

Survey Questions

A.1 Personal characteristics: travel-related

1. What is your current employment status?

◇ Self-employed ◇ Full-time ◇ Part-time ◇ Non-employed student
◇ Homemaker ◇ Retired ◇ Unemployed

If you checked 1, 2 or 3 in question 1, please answer questions 2 and 3 and 4

2. How do you usually get to your work place (or school) from home?

◇ Drive a vehicle alone ◇ Carpool with family member(s)
◇ Carpool with others (not family members) ◇ Take transit
◇ Mixed car-transit transportation ◇ Bicycle or walk
◇ Others, can you specify _____

3. If you checked 1, 2 or 3 in question 2, how do you choose your route?

◇ Only one reasonable route is available to me
◇ I have several route choices but I use one of them everyday because it is fast and convenient
◇ I have several route choices and I mix use them according to real-time traffic reports and other conditions

4. How would you describe the day-to-day fluctuation of your commuting time (the time you spend in traveling between home and work)?

◇ Negligible
◇ Significant disruption happens occasionally (approximately once a month)
◇ Significant disruptions happen once or twice a week
◇ Largely unpredictable (disruptions in various forms happen three times a week or more)

5. Do you have accessibility to public transportation in your neighborhood?

◇ Yes ◇ No

If the answer is ”yes”, do you think the service is more reliable than driving?

◇ Yes ◇ No

6. How old are you?

◇ 0 - 25 ◇ 25 - 40 ◇ 40 - 60 ◇ 60 - 75 ◇ 75 or older

7. What is your gender?

◇ Male ◇ Female

8. What is your education background?

◇ Completed elementary school ◇ Some high school
◇ High school graduate ◇ Some college or technical school
◇ College or technique school graduate ◇ Some graduate school
◇ Completed graduate degrees

A.2 Reliable Routing behavioral and attitude

9. What methods, if any, are you currently using to plan your routes? (Please check all that apply)

◇ Google Maps / Yahoo Maps ◇ In-vehicle Navigation Systems
◇ Traffic Reports (radio station or T.V. news) ◇ Past Experience
◇ Other:___ (Please specify)

10. Please indicate how satisfied you are with the current routing method? Please use the scale from 1 to 5 (1 indicates ”very unsatisfied” and 5 indicates ”very satisfied”) to specify: _____

11. Please use a scale from 1 to 5 to indicate how important arriving on-time is for you. (1 represents ”not important at all” and 5 represents ”extremely important”)

◇ Go to work (or go to school): _____
◇ Go to a doctor’s appointment: _____
◇ Go to the airport to take a flight: _____
◇ Go to a job interview: _____
◇ Go to theater: _____
◇ Go to a friend’s party: _____

12. Which TWO of the following factors that affect YOUR CURRENT route choice behavior MOST?

◇ Travel time ◇ Cost (fuel consumption, the availability of vehicle)
◇ Reliability ◇ Safety ◇ Comfort and convenience
◇ Emission and energy conservation

13. Which TWO of the following factors that affect YOUR CURRENT route choice behavior LEAST?

◇Travel time ◇ Cost (fuel consumption, the availability of vehicle)
◇ Reliability ◇Safety
◇ Comfort and convenience ◇ Emission and energy conservation

14. Suppose you plan to arrive at the airport to take a flight at 8:30am on a weekday. Google Map tells you that "the trip takes 30 minutes on average and 60 minutes in traffic". How much time you would "budget" for that trip?

◇15 - 25 minutes ◇ 25 - 35 minutes ◇35 - 45 minutes
◇45 - 60 minutes hour ◇ More than 1 hour

15. Suppose your daily work trip takes place during morning rush hour and takes approximately 30 minutes on average. How much time you would usually "budget" for that trip?

◇15 - 25 minutes ◇ 25 - 35 minutes ◇ 35 - 45 minutes
◇45 - 60 minutes hour ◇ More than 1 hour

16. Generally, how often does travel time reliability become a concern to your travel decision?

◇Everyday ◇ Once a few days ◇ Once a few weeks
◇Once a few months ◇ A couple of time a year ◇ Almost never.

17. If you are concerned about the reliability of your work trip, which of the following information you find most useful in making your routing decision?

◇ The average trip travel time and the standard deviation. Roughly speaking, your trip time is likely to be lower than the sum of average plus standard deviation with a probability of 6 out of 7.

◇A histogram plot of trip travel times, from which you may find out that one in 100 times the trip takes 29 - 30 minutes, 3 out of 100 times the trip take 30 - 32 minutes, and so on.

◇Number of times you are late for work per week , per month or per year if you always depart

at the same time.

◇The departure time that ensures you are late for work no more than once per week, per month or per year.

18. When travel reliability is important to you, what do you do to ensure arriving on time? Please rank the following options using 1 - 5 (1 is the most unfavorable and 5 is the most favorable).

__ Reserve more time for travel

__ Detour, always avoid using freeway / expressway

__ Detour, try to stay on freeway / expressway as much as possible

__ Take transit

__ Get real time traffic informationReliable route guidance software

A.3 Evaluation of CTR

19. Suppose you want to ensure that you are not late for work more than once a month (that is a probability of 1 out 22 approximately). In order to meet the goal, CTR may suggest that you are better off by switching to a relatively unfamiliar route which consists of less highway and more local streets. CTR would tell you how much time you would save over the long run by changing the route. How much time saving do you think would induce you to adopt the recommended route (assume that travel time is your only concern)?

◇ 1-5 percent time saving (up to 1.5 minutes out of a 30 minutes trip)
◇ 5-10 percent time saving (up to 3 minutes out of a 30 minute trip)
◇ 10-20 percent time saving (up to 6 minutes out of 30 minutes trip)
◇ 20-30 percent time saving (up to 9 minutes out of 30 minutes trip)
◇ 30-50 percent time saving (up to 15 minutes out of 30 minutes trip)
◇ I would always stick to the familiar route

20. Currently, it takes a half minute to one minute for CTR to generate the reliable routes. However, since reliable routes change slowly over the time, you need to run the program for any particular trip no more than once a month. Do you think this running time is acceptable?

Please use the scale from 1 to 5 (1 indicates ”totally unbearable” and 5 indicates ”totally acceptable”) to specify: ____

21. If your scale for Question 20 is lower than 3, what is the longest time you are willing to wait to get a travel route and departure time recommendation? ____

22. Suppose that you are interested in receiving reliable route guidance. Where you would like to access it?

◇ A stand-alone software application that can be installed on my PC or MAC
◇ As an add-on function in Google Maps/Yahoo Maps Application
◇ I-phone or other Smartphone application
◇ In-vehicle GPS-based navigation system
◇ Other:____ (Please specify)