Algorithms for Routing Problems in Stochastic Time-Dependent Networks

by

Seong-Cheol Kang

Submitted to the Department of Civil and Environmental Engineering
and the Operations Research Center
in partial fulfillment of the requirements for the degrees of

Master of Science in Transportation

and

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2002

Author .....
Department of Civil and Environmental Engineering
and Operations Research Center
September 29, 2001

Certified by .....
Ismail Chabini
Assistant Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by .....
James B. Orlin
Edward Pennell Brooks Professor of Operations Research
Codirector, Operations Research Center

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

Algorithms for Routing Problems in Stochastic Time-Dependent Networks

by

Seong-Cheol Kang

Submitted to the Department of Civil and Environmental Engineering
and the Operations Research Center
on September 29, 2001, in partial fulfillment of the
requirements for the degrees of
Master of Science in Transportation
and
Master of Science in Operations Research

Abstract

This thesis considers routing problems in stochastic time-dependent networks where link travel times are modeled as time-dependent discrete random variables. Among various routing problems that arise in stochastic time-dependent networks, we focus on the following three problem classes: the minimum possible travel time path problem, the minimum expected travel time next-arc hyperpath problem, and the minimum expected travel time path problem.

In the first problem class, we study the all-to-one minimum possible travel time paths problem, the all-to-one minimum possible travel cost paths problem, the all-to-one $k-$minimum path travel time realizations problem, and the all-to-one $k-$dynamic shortest paths problem. As routing problems in the second problem class, we discuss the all-to-one minimum expected travel time next-arc hyperpaths problem, the all-to-one minimum expected travel cost next-arc hyperpaths problem, the all-to-one minimum expected travel time next-arc hyperpaths problem in signalized networks, and the all-to-one minimum expected travel time next-arc hyperpaths problem in multimodal networks. Finally, we explore the all-to-one minimum expected travel time paths problem that belongs to the third problem class.

We drive optimality conditions for each routing problem. We develop efficient solution algorithms for the routing problems in the first two problem classes. We prove that the algorithms developed in the thesis have theoretically better worst-case running time complexities than the existing algorithms in the literature. Computational tests show that our algorithms outperform the existing algorithms in practice as well.

Thesis Supervisor: Ismail Chabini
Title: Assistant Professor of Civil and Environmental Engineering

Acknowledgments

First and foremost, I would like to thank my family–my parents, brother, sisters, nephews, and nieces for their love, blessing, and support. Had it not been for their constant encouragement, this work could not have been done. I love them all and dedicate this thesis to them.

I wish to thank Professor Ismail Chabini for being my research advisor. I also would like to thank Professor Cynthia Barnhart, Professor Moshe Ben-Akiva, and Professor Nigel Wilson for their constructive advice.

I am deeply grateful to Professor Robert Freund and Professor Dimitris Bertsimas for their unconditional support. Their encouragement, guidance, and help can never be repaid fully.

Special thanks go to Professor Ikki Kim at Hanyang University in Korea. His heartfelt counsel cheered me up when I was frustrated.

Finally, I would like to thank my friends at the Center for Transportation Studies, at the Operations Research Center, and elsewhere at MIT for their help and friendship.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Research Background

Routing people, vehicles, goods, messages, or other commodities over transportation or communication networks has been an important issue in satisfying economical and societal needs. When commodities are sent over a network, there usually exist one or more criteria by which routes for the commodities are determined. For instance, individuals on their way to work in the morning may want to take routes with the shortest travel times. A telecommunication company may want to route calls over a communication network in such a way that it accommodates as many calls as possible and also maintains a delay for each call below a certain level. Among various criteria, travel time between an origin and a destination serves as the primary concern in many routing applications.

Traditionally, routing problems were considered over *deterministic static networks* where link travel times are deterministic and time-invariant, i.e. they are fixed quantities. Many routing problems in deterministic static networks, whose objectives were mainly to find shortest paths, were successfully investigated and many efficient solution algorithms were developed as a result.

Researchers and practitioners then began to consider dynamism associated with link travel times and to use *deterministic time-dependent networks*, where link travel times change time-dependently, as base networks for various routing problems. With the advent of the Intelligent Transportation Systems (ITS), deterministic time-dependent networks became more important especially in transportation area because time-dependent travel time information was required to successfully deploy several subsystems of ITS such as

the Advanced Traveler Information Systems (ATIS) and the Advanced Traffic Management Systems (ATMS).

In deterministic time-dependent networks, it is assumed that the travel time on a link varies according to the entry time to the link. It is also assumed that for a given link entry time, the link travel time is a constant that is known a priori with certainty. Although deterministic time-dependent networks approximate well real transportation or communication networks where network characteristics change according to the time of day and therefore have been extensively used for various routing problems for the last decade, a couple of drawbacks of deterministic time-dependent networks especially for transportation networks have been pointed out.

Firstly, we know from daily experience that even for the same link entry time, traversing a link may take different amounts of time in congested transportation networks, because of traffic volume change, incidents, meteorological condition, or other factors. Secondly, we do not know the travel time on a link in advance until we complete traversing the link. Therefore one of the assumptions of deterministic time-dependent networks, which is the travel time on a link is an a priori known constant with certainty for a given link entry time, is not valid in congested transportation networks.

In order to obtain more realistic routing solutions by modeling actual transportation networks better, it is imperative to take into consideration the variability and uncertainty of a link travel time for a given link entry time. A natural approach to capture the variability and uncertainty of a link travel time would be to consider a link travel time for a given link entry time as a random variable. This approach results in *stochastic time-dependent networks* where link travel times are random variables that have different probability distributions according to link entry times.

Consideration of both stochasticity and time-dependency of link travel times makes routing problems much harder to solve. Presumably this is one of the reasons that active research on routing problems in stochastic time-dependent networks has begun recently and only a small number of research papers have been published. However, since stochasticity and dynamism are becoming increasingly important issues that cannot be ignored in many routing applications in transportation or communication networks, we expect stochastic time-dependent networks to take the place of deterministic time-dependent networks as base networks for various routing problems in the very near future.

This thesis studies several routing problems in stochastic time-dependent networks from a transportation application perspective. Particularly we focus on the development of fast solution algorithms for the routing problems with the intention of utilizing the algorithms for real time ITS applications.

1.2 Network Types

We may think of four network types depending on which of stochasticity and time-dependency of link travel times is considered. We illustrate the network types with examples in this section.

A deterministic static network has fixed link travel times as exemplified in Figure 1-1. No stochasticity and time-dependency of link travel times exist. If another attribute of a link other than travel time is used, the value of the link attribute can be negative.

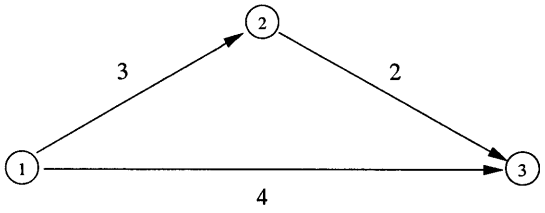

Figure 1-1: An Example of Deterministic Static Network

Figure 1-2 shows an example of stochastic static network. In networks of this type, the travel time on a link is a random variable. For instance, the travel time on link $(2, 3)$ follows a normal distribution with the mean of 4 and the variance of 1. Link travel time random variables may be discrete or continuous. There is no time-dependency of link travel times, so only one probability distribution is associated with each link. It is well known that we

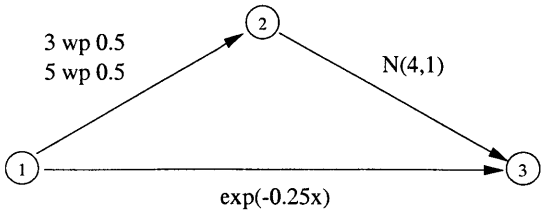

Figure 1-2: An Example of Stochastic Static Network

can find a path with the minimum expected travel time from an origin to a destination by setting each link travel time to its expected value and then applying a shortest path algorithm (Sigal et al. [24], Eiger et al. [9]). Stochastic static networks have been widely used in the reliability analysis of communication or manufacturing networks and in the PERT-type network analysis for project management.

An example of deterministic time-dependent network is depicted in Figure 1-3. The travel time on a link has different values according to the link entry time. For instance, the travel time on link $(1,2)$ is 3 units of time when a traveler enters the link at time 0 and 5 units of time when he enters the link at time 1. As mentioned in the previous section, a link travel time is constant for a given link entry time. Link travel times and link entry times can be modeled in discrete time or in continuous time. Figure 1-3 shows a deterministic discrete time-dependent network.

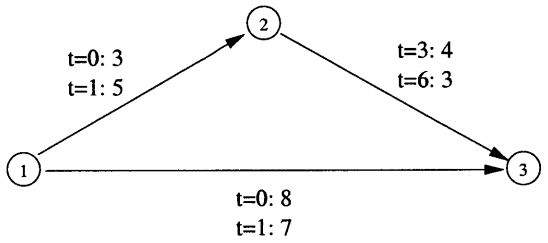

Figure 1-3: An Example of Deterministic Time-Dependent Network

Finally, Figure 1-4 illustrates an example of stochastic time-dependent network. For a given link entry time, the travel time on a link is obtained from a probability distribution that may be either discrete or continuous. For instance, the travel time on link $(1,3)$ is exponentially distributed, but the distribution has different parameters according to the

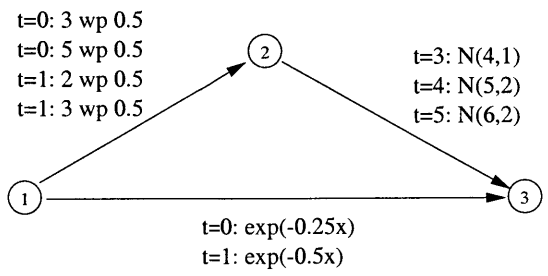

Figure 1-4: An Example of Stochastic Time-Dependent Network

link entry time. Note that stochastic time-dependent networks can be thought the most general network type because they can reduce to networks of other types if one or both of stochasticity and time-dependency of link travel times is ignored. In this thesis, we consider stochastic time-dependent networks where probability distributions are discrete and link travel times as well as link entry times are also discrete.

1.3 Problem Variants

Unlike deterministic static or deterministic time-dependent networks, stochastic time-dependent networks may allow more than one path to have some positive probability of being shortest for any origin-destination pair. Therefore the shortest travel time is not a well-defined criterion for selecting routes in stochastic time-dependent networks.

We may employ other criteria by which routing decisions can be made unambiguously in stochastic time-dependent networks, such as the lowest possible travel time on a path, the longest possible travel time on a path, the expected travel time on a path, the variance of the travel time on a path, etc. In this thesis, we consider the following two criteria among others:

- Lowest Possible Travel Time
- Expected Travel Time

In routing problems with the lowest possible travel time criterion, we want to find a path whose lowest possible travel time (minimum travel time) is smaller than that of any other path for an origin-destination pair at a given departure time. In the literature, routing problems of dispatching emergency vehicles in urban areas adopt this criterion (Miller-Hooks [16], Miller-Hooks and Mahmassani [17]).

The expected travel time criterion is associated with another class of routing problems where we want to find a path with the minimum expected travel time for an origin-destination pair at a given departure time.

It turns out that the expected travel time criterion results in a different class of routing problems if travelers are allowed to change their paths adaptively during their trips in stochastic time-dependent networks. This class of routing problems, called the minimum expected travel time next-arc hyperpath problem, is also studied in this thesis.

To recapitulate, the following three classes of routing problems are discussed in the thesis:

- Minimum Possible Travel Time Path Problem
- Minimum Expected Travel Time Path Problem
- Minimum Expected Travel Time Next-Arc Hyperpath Problem

Routing problems are further classified by the number of origins, the number of destinations, and the number of departure times. In the context of ITS applications, routing problems concerning all origins, a single destination, and all departure times (so-called all-to-one problems) are of particular interest. Therefore we will study various all-to-one routing problems in each problem class in this thesis.

1.4 Contributions

The contributions of the thesis are as follows:

- For each routing problem, we derive a set of optimality conditions that is a dynamic programming formulation of the problem.
- For several routing problems in the first and the third problem classes (the minimum possible travel time path problem and the minimum expected travel time next-arc hyperpath problem), we develop solution algorithms with worst-case optimal running time complexities.
- For other routing problems, we develop efficient solution algorithms that have better worst-case running time complexities than the existing algorithms in the literature.

1.5 Outline of the Thesis

The thesis is organized in eight chapters. This first chapter mentions the research background, the network types, the routing problem variants to be studied, and the contributions of the thesis. In Chapter 2, the related literature is reviewed. In Chapter 3, we present some preliminary material for the thesis, which includes basic notation, terminologies, assumptions, the concept of time-space network, and the concept of next-arc hyperpath. We study

several routing problems that belong to the minimum possible travel time path problem class in Chapter 4. We discuss the characteristics of routing problems of this class. For each routing problem studied, we derive a set of optimality conditions and then develop an efficient solution algorithm from it. In Chapter 5, we first introduce two routing policies in stochastic time-dependent networks. We then distinguish between the minimum expected travel time next-arc hyperpath problem and the minimum expected travel time path problem, based on routing policy. Several routing problems in the former problem class are studied in the chapter. For each routing problem, we derive a set of optimality conditions and develop an efficient solution algorithm. In Chapter 6, we discuss one problem in the minimum expected travel time path problem class, the all-to-one minimum expected travel time paths problem. We explain why this problem is inherently difficult to solve and discuss ideas of possible solution algorithms. In Chapter 7, we report on the results of the computational tests on several algorithms presented in the preceding chapters. Finally, the conclusions and the future research directions are given in Chapter 8.

Chapter 2

Literature Review

In this chapter, we review the literature pertaining to routing problems in stochastic static networks, in deterministic time-dependent networks, and in stochastic time-dependent networks. We do not survey the research work on routing problems in deterministic static networks because several textbooks summarizing the work, for instance Ahuja et al. [1] and Bertsekas [2], are readily available.

2.1 Routing Problems in Stochastic Static Networks

Most papers credit Frank [10] with the first published work on the "shortest" path problem in stochastic static networks. He considers the problem of computing the probability that the minimum travel time from an origin node to a destination node is less than some value when link travel times have continuous probability distributions. Notice that this problem is essentially equivalent to determining the probability distribution of the minimum travel time. To avoid the complications arising from the computation of multiple integrals, he approximates the computation by Monte Carlo simulation. He also studies the comparison of any two disjoint paths from an origin node to a destination node based on the probability that the travel time on each path is greater than a given threshold. He recognizes that the computation of this probability involves a convolution of link travel time random variables, which is formidable when each path contains many links. He overcomes this difficulty by using the central limit theorem. In addition, he discusses hypothesis tests on the probability that the minimum travel time from an origin node to destination node is greater than some specified value.

Sigal et al. [24] consider the selection of the "shortest" path from an origin node to a destination node when link travel time random variables are independent of each other. As a performance measure of a path to be used in determining the "shortest" path, they introduce the concept of a path optimality index. The optimality index of a path is defined as the probability that the path is shorter than all other paths. They recognize that the optimality index of a path is difficult to compute, because links may belong to more than one paths and therefore path travel times are not independent random variables. They resolve the statistical dependence among path travel times by using the concept of uniformly directed cutsets and present an analytical procedure to compute the optimality index of a path. The analytical procedure, however, involves multiple integrals that are difficult to evaluate in practice.

Eiger et al. [9] study the problem of finding an optimal path from an origin node to a destination node when a traveler uses a utility function to evaluate each path. The utility function is a nonincreasing function of the travel time on a path. An optimal path is defined as one with the maximum expected utility. They show that when link travel times are independent random variables and the utility function is linear or exponential, an efficient Dijkstra-type algorithm can solve the problem.

Mirchandani and Soroush [19] extend the work of Eiger et al. [9] to the problem with a quadratic utility function. In this case, Dijkstra-type algorithms may not find an optimal path. They propose an algorithm that depends on only the first and second moments of the travel time on a path, but it has an exponential running complexity in the worst case.

Kulkarni [14] considers networks where link travel times are independent and exponentially distributed random variables. From the network, he constructs a continuous time Markov chain (CTMC) with a single absorbing state such that the time until absorption into the absorbing state starting from the initial state is equal to the minimum travel time from a given origin node to a given destination node in the network. Using the Markov chain, he develops methods for computing the distribution of the minimum travel time, the moments of the minimum travel time, and the optimality index of a path. His approach has a limitation that the state space of the Markov chain may grow exponentially with the network size. Hence this approach is not suitable for large dense networks.

Extending the work of Kulkarni [14], Corea and Kulkarni [6] present methods for computing the distribution and moments of the minimum travel time in networks where link

travel times are independent, nonnegative, and integer valued random variables. They use a discrete time Markov chain (DTMC) with a finite state space and a single absorbing state. The same drawback as in Kulkarni [14] exists in this approach too.

2.2 Routing Problems in Deterministic Time-Dependent Networks

Perhaps the earliest paper dealing with routing problems (shortest path problems) in deterministic time-dependent networks can be attributed to Cooke and Halsey [5]. They present a recursive functional form (a set of optimality conditions) that gives the shortest paths from all nodes to one destination node for all discrete departure times.

Dreyfus [8] suggests a label-setting approach which generalizes Dijkstra's algorithm to determine the shortest path between two nodes for a given departure time.

Kaufman and Smith [13] show a counterexample for which Dreyfus' approach fails to detect the shortest path. They establish a consistency condition under which Dijkstra-type algorithms (Dreyfus' approach) are guaranteed to find the shortest path with the same computational complexity as that of the static shortest path problem. This consistency condition is thought of as the first-in first-out (FIFO) condition.

Ziliaskopoulos and Mahmassani [26] propose a label-correcting algorithm that determines the shortest paths from all nodes to one destination node for all discrete departure times. The algorithm does not require the FIFO condition to hold in the network. It can handle the case where an attribute of a link other than travel time, which can have a negative value, is used, as long as the network does not have negative "cost" cycles.

Chabini [3] considers three routing problems: the shortest paths from one origin node to all other nodes for a given departure time (called the one-to-all shortest paths problem), the shortest paths from all nodes to one destination node for all discrete departure times (called the all-to-one shortest paths problem), and the minimum cost paths from all nodes to one destination node for all discrete departure times (called the all-to-one minimum cost paths problem). He reviews optimality conditions for each problem. He then develops decreasing order of departure time (DOT) algorithms for the all-to-one shortest paths problem and the all-to-one minimum cost paths problem. He shows that the two algorithms have the optimal running time complexities, and that consequently no algorithms with better running

time complexities can be found for the problems. The ideas underlying those decreasing order of departure time algorithms will be extended to routing problems in stochastic time-dependent networks in this thesis.

Chabini and Dean [4] present a deeper analysis of waiting at nodes. They develop a decreasing order of departure time algorithm for the all-to-one shortest paths problem and an increasing order of departure time (IOT) algorithm for the one-to-all shortest paths problem, when waiting at nodes is allowed in the network.

2.3 Routing Problems in Stochastic Time-Dependent Networks

Hall [12] appears to be a seminal paper about routing problems in stochastic time-dependent networks. He shows that static shortest path algorithms such as Dijkstra's algorithm may not find the minimum expected travel time path between two nodes for a given departure time in stochastic time-dependent networks. He proposes an algorithm that finds the minimum expected travel time path from one origin node to one destination node for a given departure time. The algorithm combines a branch-and-bound technique and a $k-$shortest paths algorithm. Although the algorithm is exact, it does not explain how to compute the expected travel time on a given path. The algorithm will be reviewed in Chapter 6 of this thesis. He also realizes that the best route from any intermediate node to the destination in terms of expected travel time depends on the arrival time at that intermediate node. Therefore the best route can be found by deferring the choice of the next link to take until the intermediate node is reached. He calls this method the time-adaptive route choice. The result obtained from the time-adaptive route choice is generally not a simple path, but it is referred to as an optimal adaptive decision rule (it is called the minimum expected travel time next-arc hyperpath in this thesis). He applies dynamic programming to the problem of finding an optimal adaptive decision rule from one origin node to one destination node for a given departure time. We will study related problems in Chapter 5.

Kaufman and Smith [13] briefly mention a consistency condition for the determination of the minimum expected travel time path in stochastic time-dependent networks. However they do not propose any solution algorithm.

Miller-Hooks [16] studies various routing problems in stochastic time-dependent net-

works where link travel times are independent discrete random variables. She proposes several efficient solution algorithms, some of which will be revisited in this thesis.

Miller-Hooks and Mahmassani [17] discuss the problem of finding a path with the minimum possible travel time from all nodes to one destination node for all discrete departure times when link travel times are independent discrete random variables. They extend the problem to selecting several paths from all nodes to one destination node for all discrete departure times. These two problems will be studied in detail in Chapter 4.

Fu and Rilett [11] study the problem of finding the minimum expected travel time path from one origin node to one destination node for a given departure time when link travel times are modeled as continuous time stochastic processes, i.e. link travel times have continuous probability distributions. They claim that even if the link travel times are independent random variables, the probability distribution of the travel time on a path is very hard to obtain from the probability distributions of the travel times on the links constituting the path. Hence they study how to estimate the expected travel time on a path by using the means and variances of the travel times on the constituent links of the path. They present a heuristic algorithm for the problem, which relies on a $k-$shortest paths algorithm as well as the estimated expected travel time on a path.

Miller-Hooks and Mahmassani [18] propose an efficient algorithm for the problem of finding the minimum expected travel time next-arc hyperpaths from all nodes to one destination node for all discrete departure times. They also present a non-polynomial algorithm for the problem of finding the minimum expected travel time paths from all nodes to one destination node for all discrete departure times. We will revisit these algorithms in Chapters 5 and 6.

Yang and Miller-Hooks [25] study the problem of finding the minimum expected travel time next-arc hyperpaths from all nodes to one destination node for all discrete departure times in signalized networks. This problem will be discussed in Chapter 5.

Opasanon and Miller-Hooks [21] consider stochastic time-dependent multimodal networks. They study the problem of finding the minimum expected travel time next-arc hyperpaths from all nodes to one destination node for all discrete departure times in such networks. This problem will be also discussed in Chapter 5.

Chapter 3

Preliminaries

In this chapter, we present preliminary material for this thesis. In Section 3.1, we introduce basic notation, terminologies, and assumptions that will be used throughout the thesis. Other problem-specific notation, terminologies, and assumptions will be introduced where appropriate in the thesis. The comprehensive set of notation is summarized in Appendix A. We also introduce the concepts of time-space network and next-arc hyperpath in Section 3.2 and in Section 3.3, respectively.

3.1 Notation, Terminologies, and Assumptions

3.1.1 Network

Consider a network consisting of a finite number of nodes and a finite number of directed links. We denote such a network by $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of n nodes and $\mathcal{A}$ is the set of m directed links.

Let d denote a given destination node. The network is assumed to have at least one directed path from every node to the destination node d. It is assumed that no parallel links exist between any two nodes, thus we have $m \leq n(n-1)$.

We denote by $\mathcal{O}(i)$ the set of end nodes of outgoing links from node i, i.e. $\mathcal{O}(i) = \{j \mid (i,j) \in \mathcal{A}\}$. Similarly, we denote by $\mathcal{I}(i)$ the set of start nodes of incoming links to node i, i.e. $\mathcal{I}(i) = \{j \mid (j,i) \in \mathcal{A}\}$. Note that $\sum_{i \in \mathcal{N}} |\mathcal{O}(i)| = \sum_{i \in \mathcal{N}} |\mathcal{I}(i)| = m$.

3.1.2 Time Period of Interest

Let $\mathcal{H}^* = [t_l, t_u]$ be a continuous time period of interest, which is generally a peak period of the network. We discretize $\mathcal{H}^*$ into small time intervals. Let $\mathcal{H} = \{t_l, t_l + \Delta t, t_l + 2\Delta t, \ldots, t_l + (H-1)\Delta t\}$ be a discretized time period of interest, where Δt is the length of each small time interval and $t_l + (H-1)\Delta t = t_u$. Δt should be chosen such that it is no greater than any link travel time value. This allows us to avoid the case where one would arrive at a next node at the same time interval as the departure time from a node, which is not realistic from a practical point of view.

For the sake of brevity of exposition hereafter, without loss of generality, we assume that $t_l = 0$ and $\Delta t = 1$. Thus the discretized time period of interest becomes the set $\mathcal{H} = \{0, \ldots, H-1\}$.

3.1.3 Link Travel Times

We assume that link travel times are time-dependent random variables whose probability distributions vary according to the times that the links are entered.

For each link $(i,j) \in \mathcal{A}$ and each link entry time $t \in \mathcal{H}$, let $T_{ij}(t)$ be a random variable denoting the travel time on link (i,j) when one enters the link at time t. We assume that the probability distribution of $T_{ij}(t)$ is discrete. The probability mass function (PMF) of $T_{ij}(t)$ is denoted by $p_{T_{ij}(t)}$.

We define a *realization* of $T_{ij}(t)$ as a couple of a possible travel time value and its probability. Let $T_{ij}(t)$ have $r_{ij}(t)$ realizations denoted by $(\tau_{ij}^r(t), p_{ij}^r(t))$, $r \in \mathcal{R}_{ij}(t) = \{1, \ldots, r_{ij}(t)\}$. $\tau_{ij}^r(t)$ is the travel time value of the r^{th} realization of $T_{ij}(t)$. $p_{ij}^r(t)$ is the probability of the r^{th} realization of $T_{ij}(t)$ (the probability that $\tau_{ij}^r(t)$ occurs, i.e. $P[T_{ij}(t) = \tau_{ij}^r(t)] = p_{ij}^r(t)$). We assume that $\tau_{ij}^r(t)$, $\forall r \in \mathcal{R}_{ij}(t)$ are distinct. We represent the PMF of $T_{ij}(t)$ by the set of realizations as follows:

$$p_{T_{ij}(t)} = \left\{ (\tau_{ij}^r(t), p_{ij}^r(t)) \mid r \in \mathcal{R}_{ij}(t) \right\}, \tag{3.1}$$

where $\sum_{r \in \mathcal{R}_{ij}(t)} p_{ij}^r(t) = 1$.

We assume that the link travel time values $\tau_{ij}^r(t)$, $\forall (i,j) \in \mathcal{A}$, $\forall t \in \mathcal{H}$, $\forall r \in \mathcal{R}_{ij}(t)$ are positive integers. We also assume that after the peak period, i.e. $t > H-1$, the network is no longer time-dependent, but still stochastic. The travel time distribution of link (i,j)

at a link entry time $t > H-1$ is assumed to be the same as that of link (i,j) at the link entry time $t = H-1$. Mathematically, $p_{T_{ij}(t)} = p_{T_{ij}(H-1)}$, $\forall\,(i,j) \in \mathcal{A}$, $\forall\, t > H-1$.

We denote by $\widetilde{m}$ the total number of link travel time realizations in the network during the peak period. $\widetilde{m}$ is given by

$$\widetilde{m} = \sum_{(i,j)\in\mathcal{A}} \sum_{t\in\mathcal{H}} |\mathcal{R}_{ij}(t)|. \tag{3.2}$$

As will be explained in Section 3.2.2, $\widetilde{m}$ is the same as the number of links in the time-space network of a stochastic time-dependent network.

3.1.4 Path Travel Times

In stochastic time-dependent networks, the travel time on a path is also a time-dependent random variable because it is a function of the travel times of the links constituting the path, which are time-dependent random variables.

Let $L_i^c(t)$ be a random variable denoting the travel time on path c from node i to the destination node d at departure time t. Like a realization of a link travel time, we define a *realization* of $L_i^c(t)$ as a couple of a possible travel time value and its probability. The realizations of $L_i^c(t)$ are not given data, but computed from the travel time realizations of the links on the path.

We denote the realizations of $L_i^c(t)$ by $(l_i^{c^k}(t), p_i^{c^k}(t))$, $k = 1, 2, \ldots, k_i^c(t)$, where $k_i^c(t)$ is the number of realizations. $l_i^{c^k}(t)$ is the travel time value of the k^{th} realization of $L_i^c(t)$ and $p_i^{c^k}(t)$ is the probability that $l_i^{c^k}(t)$ occurs, i.e. $P[L_i^c(t) = l_i^{c^k}(t)] = p_i^{c^k}(t)$. The PMF of $L_i^c(t)$ is then represented by

$$p_{L_i^c(t)} = \left\{(l_i^{c^k}(t), p_i^{c^k}(t)) \mid k = 1, 2, \ldots, k_i^c(t)\right\}. \tag{3.3}$$

Let us illustrate how to construct the PMF of $L_i^c(t)$ from the link travel time PMFs when the link travel times are independent random variables. Consider a trip from node 1 to node 3 at departure time 0 in the stochastic time-dependent network depicted in Figure 3-1 and Table 3.1. Let path a be $1 \to 2 \to 3$ and path b be $1 \to 3$. Assuming that all link travel time random variables are independent of each other, we obtain all travel time realizations of each path for this trip as follows:

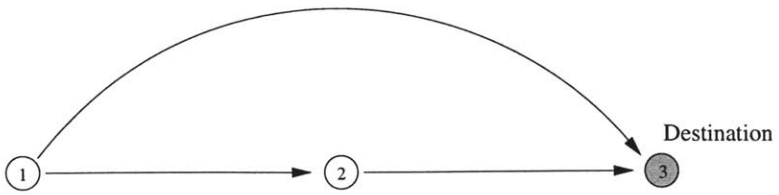

Figure 3-1: An Example Network

Table 3.1: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$	$(1,3)$	$(2,3)$	
Departure Time (t)	0	0	1	2
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (2, 0.5)	(4, 0.5) (6, 0.5)	(2, 0.4) (3, 0.6)	(1, 0.8) (3, 0.2)

- For path a:

$$l_1^{a^1}(0) = \tau_{12}^1(0) + \tau_{23}^1(0 + \tau_{12}^1(0)) = 1 + \tau_{23}^1(1) = 1 + 2 = 3$$

$$p_1^{a^1}(0) = p_{12}^1(0) \times p_{23}^1(0 + \tau_{12}^1(0)) = 0.5 \times p_{23}^1(1) = 0.5 \times 0.4 = 0.2$$

$$l_1^{a^2}(0) = \tau_{12}^1(0) + \tau_{23}^2(0 + \tau_{12}^1(0)) = 1 + \tau_{23}^2(1) = 1 + 3 = 4$$

$$p_1^{a^2}(0) = p_{12}^1(0) \times p_{23}^2(0 + \tau_{12}^1(0)) = 0.5 \times p_{23}^2(1) = 0.5 \times 0.6 = 0.3$$

$$l_1^{a^3}(0) = \tau_{12}^2(0) + \tau_{23}^1(0 + \tau_{12}^2(0)) = 2 + \tau_{23}^1(2) = 2 + 1 = 3$$

$$p_1^{a^3}(0) = p_{12}^2(0) \times p_{23}^1(0 + \tau_{12}^2(0)) = 0.5 \times p_{23}^1(2) = 0.5 \times 0.8 = 0.4$$

$$l_1^{a^4}(0) = \tau_{12}^2(0) + \tau_{23}^2(0 + \tau_{12}^2(0)) = 2 + \tau_{23}^2(2) = 2 + 3 = 5$$

$$p_1^{a^4}(0) = p_{12}^2(0) \times p_{23}^2(0 + \tau_{12}^2(0)) = 0.5 \times p_{23}^2(2) = 0.5 \times 0.2 = 0.1$$

- For path b:

$$l_1^{b^1}(0) = \tau_{13}^1(0) = 4$$

$$p_1^{b^1}(0) = p_{13}^1(0) = 0.5$$

$$l_1^{b^2}(0) = \tau_{13}^2(0) = 6$$

$$p_1^{b^2}(0) = p_{13}^2(0) = 0.5$$

The PMFs of $L_1^a(0)$ and $L_1^b(0)$ are therefore given by

$$p_{L_1^a(0)} = \{(3, 0.2), (4, 0.3), (3, 0.4), (5, 0.1)\}, \tag{3.4}$$

$$p_{L_1^b(0)} = \{(4, 0.5), (6, 0.5)\}. \tag{3.5}$$

Note that the first and the third realizations of $L_1^a(0)$ have the same travel time value, i.e. $l_1^{a^1}(0) = l_1^{a^3}(0)$. The two path travel time realizations, however, are obtained by different combinations of the link travel time realizations. The first realization is obtained by $(\tau_{12}^1(0), p_{12}^1(0))$ and $(\tau_{23}^1(1), p_{23}^1(1))$, whereas the third realization is obtained by $(\tau_{12}^2(0), p_{12}^2(0))$ and $(\tau_{23}^1(2), p_{23}^1(2))$. In the minimum possible travel time path problem that will be discussed in Chapter 4, we will treat each travel time realization of a path obtained by a different combination of the link travel time realizations on the path differently, regardless of its travel time value. Hence the first and the third travel time realizations of path a in the above example are considered to be different realizations although they have the same travel time value. Consequently, we allow allow a PMF of path travel time to have more than one realization with the same travel time value as exemplified in (3.4).

Now we introduce several terminologies related to path travel time. For a given node i, path c, and departure time t, $\min_k\{l_i^{c^k}(t)\}$ is referred to as the *minimum travel time* on path c from node i to node d at departure time t. A travel time realization of path c from node i to node d at departure time t, whose travel time value equals to the minimum travel time is called a *minimum travel time realization.* Since we differentiate between travel time realizations of a path, which have the same travel time value, but are obtained by different combinations of the link travel time realizations on the path, there may exist several minimum travel time realizations on path c from node i to node d at departure time t.

In the above example, the minimum travel time on path a from node 1 to node 3 at departure time 0 is 3 units of time. $(3, 0.2)$ and $(3, 0.4)$ are the minimum travel time realizations of path a at departure time 0. As for path b from node 1 to node 3, the minimum travel time at departure time 0 is 4 units of time and $(4, 0.5)$ is the minimum travel time realization at departure time 0.

For a given node i and departure time t, we refer to $\min_c\{\min_k\{l_i^{c^k}(t)\}\}$ as the *minimum possible travel time* from node i to node d at departure time t. We call a path travel time

realization from node i to node d at departure time t, whose travel time value equals to the minimum possible travel time a *minimum possible travel time realization.* More than one minimum possible travel time realization may exist from node i to node d at departure time t. Some of those realizations may belong to the same path.

The minimum possible travel time from node 1 to node 3 at departure time 0 in the above example is 3 units of time. $(3, 0.2)$ and $(3, 0.4)$ are the minimum possible travel time realizations from node 1 to node 3 at departure time 0. Both of the minimum possible travel time realizations belong to path a.

Note that the minimum travel time and a minimum travel time realization are the terminologies associated with each (origin node, path, departure time) triplet, while the minimum possible travel time and a minimum possible travel time realization are the terminologies associated with each (origin node, departure time) pair.

3.1.5 Other Notation and Assumptions

Stochastic time-dependent networks are said to have the FIFO property if the following conditions hold [17].

$$P[s + T_{ij}(s) \leq t + T_{ij}(t)] = 1, \ \ \forall (i,j) \in \mathcal{A}, \ \forall t \in \mathcal{H}, \ \forall s < t, \ s \in \mathcal{H}. \tag{3.6}$$

If the networks have the FIFO property, solution algorithms for some routing problems could be developed fairly easily. For instance, Dijkstra-type algorithms can solve the minimum possible travel time path problem. However it is a property that is not always satisfied in real transportation networks. In this thesis, we do not impose the FIFO property on the networks.

Concerning waiting at nodes, we assume that no waiting is allowed at all nodes in most part of the thesis. However we will provide short discussions about the case where waiting is allowed at all nodes in Chapters 4 and 5.

While developing solution algorithms for various routing problems in stochastic time-dependent networks in the subsequent chapters, we often need to solve an all-to-one shortest paths problem in deterministic static networks as an initialization step of the algorithms (details will be given in relevant parts of the thesis). We denote by $\Theta(f(n,m))$ the lowest possible running time to solve an all-to-one static shortest paths problem with n nodes and

m links.

3.2 Time-Space Network

3.2.1 Time-Space Network of a Deterministic Time-Dependent Network

When a deterministic time-dependent network is graphically represented without explicitly incorporating time as a dimension, a vector of numbers which indicates time-dependent travel times is associated with each link as exemplified in Figure 3-2. In this figure, for instance, $[1, 2, 1, 2, 2]$ on link $(1, 2)$ means that the traversal time of link $(1, 2)$ is 1 unit of time at departure time 0, 2 at departure time 1, and so on.

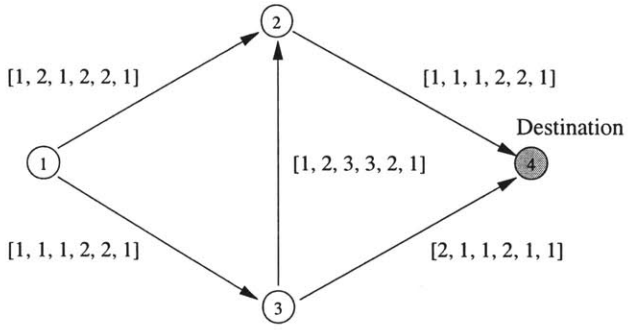

Figure 3-2: An Example of Deterministic Time-Dependent Network

This network representation, however, is not convenient for visualization of the network and algorithm development. Especially for the later purpose, we use the concept of *time-space network*. A time-space network is a deterministic static network constructed by expanding the original network in the time dimension. The following shows how to construct the time-space network of a deterministic time-dependent network.

For each node i of the original network, we expand node i by creating node-time pairs (i, t) for all $t \in \mathcal{H}$. We also create an additional node-time pair (i, H) which plays a role of all node-time pairs (i, t), $\forall t > H - 1$. Node-time pairs $(i, t) \in \mathcal{N} \times \mathcal{H} \cup \{H\}$ are the set of nodes of the time-space network. We introduce a link between two node-time pairs (i, t) and (j, s), if $(i, j) \in \mathcal{A}$ and the travel time on link (i, j) at time t is $s - t$.

Let us denote the set of nodes and the set of links of the time-space network by $\widehat{\mathcal{N}}$ and $\widehat{\mathcal{A}}$ respectively. They are mathematically expressed as follows:

$$\widehat{\mathcal{N}} = \{(i,t) \mid i \in \mathcal{N},\ t \in \mathcal{H} \cup \{H\}\}, \tag{3.7}$$

$$\widehat{\mathcal{A}} = \{((i,t),(j,s)) \mid (i,j) \in \mathcal{A},\ t + \tau_{ij}(t) = s,\ t \in \mathcal{H},\ s \in \mathcal{H} \cup \{H\}\}, \tag{3.8}$$

where $\tau_{ij}(t)$ is the deterministic travel time on link (i,j) of the original network at link entry time t.

Figure 3-3 shows the time-space network of the deterministic time-dependent network depicted in Figure 3-2. Here are several observations about the time-space network of a deterministic time-dependent network.

1. The time-space network is a deterministic static network. Therefore any deterministic static shortest path algorithm can be directly applied to the time-space network.

2. There exist a one-to-one correspondence between all paths in the time-space network and all paths in the original network.

3. If all link travel times of the original network are positive, then the corresponding time-space network is acyclic.

4. By construction, no parallel links exist between any two nodes in the time-space network.

5. If waiting is allowed at node i of the original network from time t to time s $(s > t)$, then there exist links $((i,t),(i,t+1))$, $((i,t+1),(i,t+2))$, $\ldots$, $((i,s-1),(i,s))$ in the corresponding time-space network.[1]

6. The number of nodes in the time-space network is $|\mathcal{N}| \times |\mathcal{H} \cup \{H\}| = n(H+1)$.

7. The number of links in the time-space network is $|\mathcal{A}| \times |\mathcal{H}| = mH$. If waiting is allowed at nodes, it is bound from above by $(|\mathcal{A}| + |\mathcal{N}|) \times |\mathcal{H}| = (m+n)H$.[2]

8. Let $\widehat{\mathcal{O}}((i,t))$ be the set of end nodes of outgoing links from node (i,t) in the time-space network. If $\widetilde{\mathcal{O}}((i,t)) = \{(j_1,s_1),(j_2,s_2),\ldots,(j_p,s_p)\}$, then $j_1 \neq j_2 \neq \cdots \neq j_p$.

[1] This is valid only if the cost of waiting is additive (Chabini and Dean [4]).

[2] This is valid only if the length of waiting is unlimited and the cost of waiting is additive (Chabini and Dean [4]).

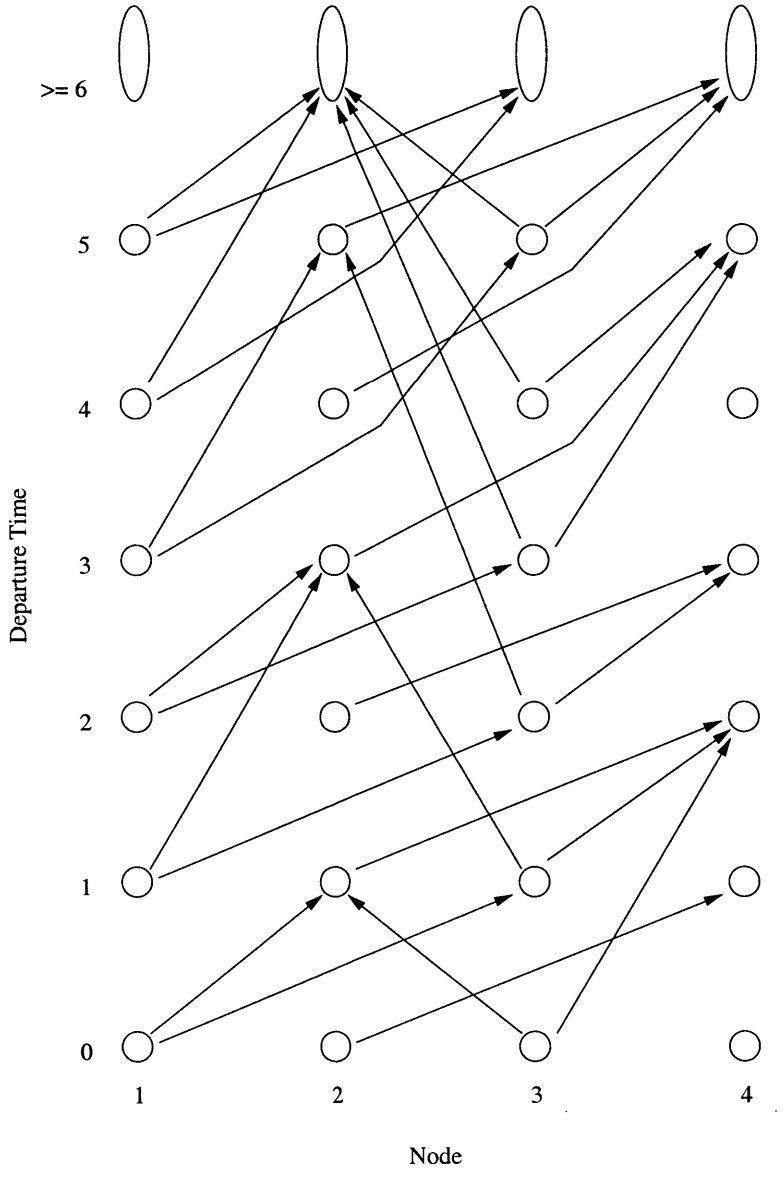

Figure 3-3: Time-Space Network of the Deterministic Time-Dependent Network in Figure 3-2, $H = 6$

We can also draw a time-space network using a three-dimensional diagram where one dimension represents departure time and the other two dimensions describe the spatial layout of the nodes in the original network (see Figure 3-4).

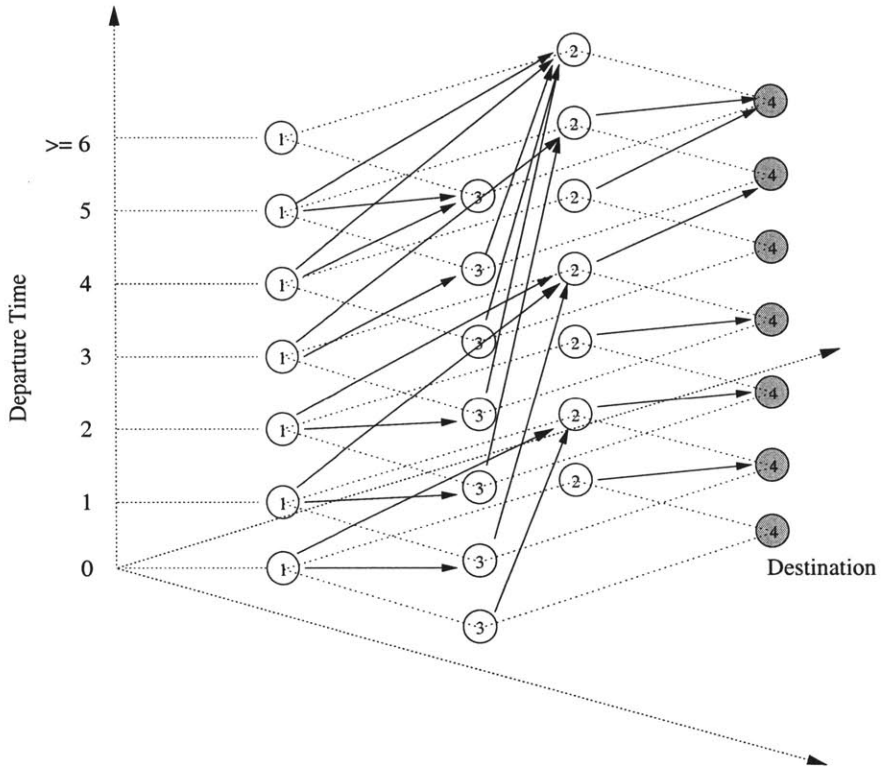

Figure 3-4: An Example of 3-Dimensional Time-Space Network

3.2.2 Time-Space Network of a Stochastic Time-Dependent Network

Similar to deterministic time-dependent networks, we can construct the time-space network of a stochastic time-dependent network. In this case, the set of nodes of the time-space network is the same as $\widehat{\mathcal{N}}$, but the time-space network has much more links because the travel time on link (i, j) of the original network may have several realizations for a given departure time.

Let $\widetilde{\mathcal{N}}$ and $\widetilde{\mathcal{A}}$ be the set of nodes and the set of links of the time-space network of a stochastic time-dependent network, respectively. Then

$$\widetilde{\mathcal{N}} = \{(i,t) \mid i \in \mathcal{N},\ t \in \mathcal{H} \cup \{H\}\}, \tag{3.9}$$

$$\widetilde{\mathcal{A}} = \left\{((i,t),(j,s)) \mid (i,j) \in \mathcal{A},\ t + \tau_{ij}^r(t) = s,\ r \in \mathcal{R}_{ij}(t),\ t \in \mathcal{H},\ s \in \mathcal{H} \cup \{H\}\right\}. \quad (3.10)$$

Several observations about the time-space network of a stochastic time-dependent network are as follows:

1. The time-space network is a deterministic static network. Therefore any deterministic static shortest path algorithm can be directly applied to the time-space network.
2. Since we assume that all $\tau_{ij}^r(t)$ are positive, the time-space network is acyclic.
3. By construction, no parallel links exist between any two nodes in the time-space network.
4. If waiting is allowed at node i of the original network from time t to time s ($s > t$), then there exist links $((i,t),(i,t+1))$, $((i,t+1),(i,t+2))$, $\ldots$, $((i,s-1),(i,s))$ in the corresponding time-space network.[3]
5. The number of nodes in the time-space network is $|\mathcal{N}| \times |\mathcal{H} \cup \{H\}| = n(H+1)$.
6. The number of links in the time-space network is $\widetilde{m} = \sum_{(i,j)\in\mathcal{A}} \sum_{t\in\mathcal{H}} |\mathcal{R}_{ij}(t)|$. If waiting is allowed at nodes, it is bound from above by $\widetilde{m} + nH$.[4]
7. Let $\widetilde{\mathcal{O}}((i,t))$ be the set of end nodes of outgoing links from node (i,t) in the time-space network. If $\widetilde{\mathcal{O}}((i,t)) = \{(j_1,s_1),(j_2,s_2),\ldots,(j_p,s_p)\}$, then some of $j_1, j_2, \cdots, j_p$ can be identical.

Suppose there exist several paths between two nodes in the time-space network of a stochastic time-dependent network. From the last observation above, we can deduce that some of the topological paths corresponding to those paths could be identical. Note that all topological paths corresponding to paths between two nodes in the time-space network of a deterministic time-dependent network are distinct if waiting at nodes is not allowed in the deterministic time-dependent network.

The acyclic property of the time-space network of a stochastic time-dependent network will play a key role when we develop efficient algorithms for routing problems in stochastic time-dependent networks later in this thesis.

[3]This is valid only if the cost of waiting is additive (Chabini and Dean [4]).

[4]This is valid only if the length of waiting is unlimited and the cost of waiting is additive (Chabini and Dean [4]).

Figure 3-5 shows an example of stochastic time-dependent network. For instance, $[1, 2, 1, 2, 2, 1]$ wp 0.5 and $[2, 1, 3, 3, 3, 2]$ wp 0.5 on link $(1, 2)$ mean that $P[T_{12}(0) = 1] = 0.5$, $P[T_{12}(0) = 2] = 0.5$; $P[T_{12}(1) = 2] = 0.5$, $P[T_{12}(1) = 1] = 0.5$; and so on. Its time-space network is drawn in Figure 3-6. The solid lines correspond to the first realizations and the dotted lines correspond to the second realizations.

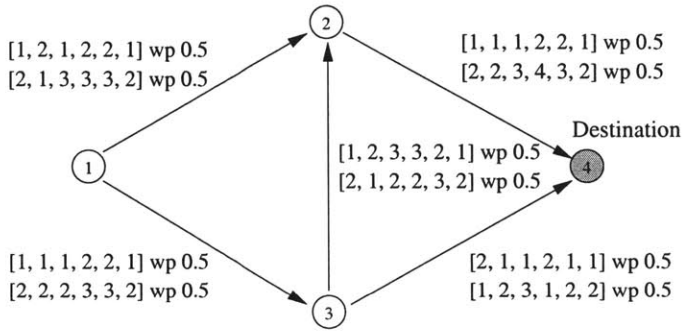

Figure 3-5: An Example of Stochastic Time-Dependent Network

3.3 Hyperpath and Next-Arc Hyperpath

Finding an optimal simple path from an origin node to a destination node based on a priori network state information is of primary interest in many routing problems. The determination of shortest paths in deterministic static networks is one example.

However, if the network state changes during the trip from an origin node to a destination node, the simple path chosen beforehand might be no longer optimal. In this case, a traveler might need to deviate from the chosen path in order to move to a better path by utilizing new network state information en route. Since different routing decisions can be made according to new network state information available during the trip, an optimal path cannot be described as a simple path. Instead, all different routing decisions collectively form an acyclic network often called a *hyperpath*, which connects the origin node and the destination node.

Generally, a hyperpath is a collection of more than one simple path from an origin node to a destination node. All simple paths on the hyperpath are "candidates" that a traveler can choose, and the actual selection of a path is dependent upon the network state en route.

The concept of hyperpath has played an important role in transit networks with over-

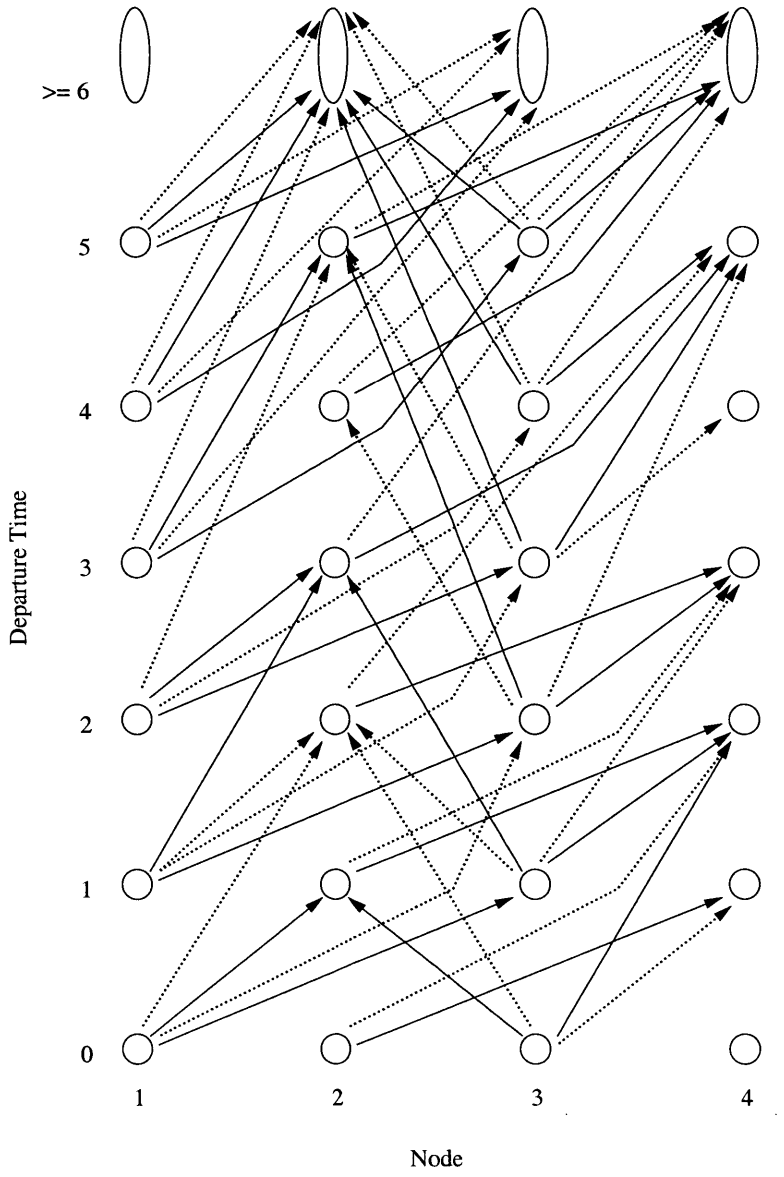

Figure 3-6: Time-Space Network of the Stochastic Time-Dependent Network in Figure 3-5, $H = 6$

lapping routes. Consider bus services in transportation networks. In general, passengers do not know the exact arrival times of bus lines at bus stops, but they have knowledge on the frequency of each bus line. If there exist several ways (combinations of bus lines) to go from an origin node to a destination node, the actual path, i.e. a sequence of bus lines that a passenger takes, will vary according to the arrivals of buses at the bus stops. The collection of bus lines that a passenger may take at the bus stops forms a hyperpath (Pallottino and Scutellà [22]).

Transit routing in transportation networks is not the only application for which the hyperpath concept can be usefully exploited. In fact, there are other routing problems where an optimal itinerary of a traveler is not completely defined a priori, because an optimal itinerary may change as new network state information reveals during the trip. In these problems too, hyperpaths can model the traveler's adaptive routing decisions that depend on the en route network state.

We define a *next-arc hyperpath* as a constrained hyperpath such that at every node of the hyperpath, only one link among outgoing links from the node is used at each departure time. We will use the concept of next-arc hyperpath in Chapter 5 to solve expected travel time-based routing problems in stochastic time-dependent networks when travelers are allowed to select links adaptively en route depending on the arrival times at nodes in the network.

Figure 3-7 shows a next-arc hyperpath example for a trip from node 1 to node 5. In this example, path $1 \rightarrow 2 \rightarrow 3 \rightarrow 5$ is optimal if a traveler arrives at node 2 before 9 AM. But if he arrives at node 2 after 9 AM, it is better to take path $1 \rightarrow 2 \rightarrow 4 \rightarrow 5$. Note that if the exact traversal time of link $(1, 2)$ is not known a priori, we cannot specify a simple optimal path from node 1 to node 5 because either path can be optimal depending on the network state after he leaves node 1. Notice that according to the arrival time at node 2,

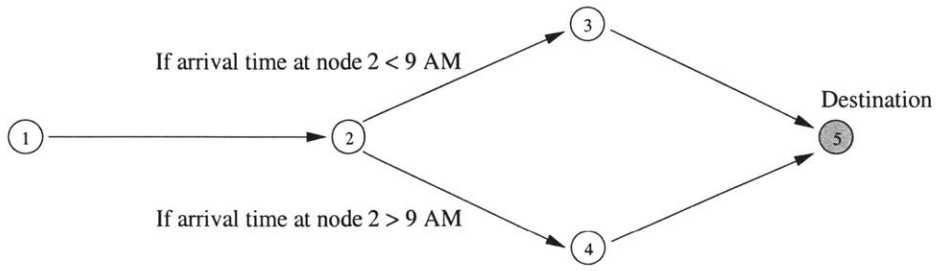

Figure 3-7: A Next-Arc Hyperpath Example

one of the two outgoing links from node 2 is used, i.e. either link $(2, 3)$ or link $(2, 4)$. Using both links at a given departure time from node 2 is not allowed in a next-arc hyperpath.

For the mathematical definition of hyperpath and related discussions, we refer the reader to Nguyen and Pallottino [20] and Pretolani [23].

Chapter 4

Minimum Possible Travel Time Paths

In this chapter, we discuss the *minimum possible travel time path (MPTTP) problem* that relies on path travel time realizations. As mentioned in subsection 3.1.4, in stochastic time-dependent networks, the travel time on a path for a given departure time is a random variable that may have several realizations. In the minimum possible travel time path problem, we compare all path travel time realizations between an origin node and a destination node for a given departure time and select a path that has a minimum possible travel time realization. A path with a minimum possible travel time realization is referred to as a *minimum possible travel time path.*

We first study the all-to-one minimum possible travel time paths problem which is one variant of the minimum possible travel time path problem in Sections 4.1–4.5. In Section 4.6, we consider the exact probability that the minimum possible travel time occurs on a path. We discuss several extensions of the all-to-one minimum possible travel time paths problem in Section 4.7. Finally, we touch briefly on the case when waiting is allowed at all nodes in the network in Section 4.8.

4.1 Introduction

We define the *all-to-one minimum possible travel time paths problem* as follows: *Given a stochastic time-dependent network, the problem is to find a path with a minimum possible travel time realization from each node to a given destination node for each departure time.*

If more than one path have a minimum possible travel time realization, a path that has the highest probability associated with a minimum possible travel time realization is selected.

The link travel time random variables are assumed to be independent of each other in this problem.

4.1.1 An Illustrative Example

Let us illustrate the problem with the stochastic time-dependent network depicted in Figure 4-1 and Table 4.1.

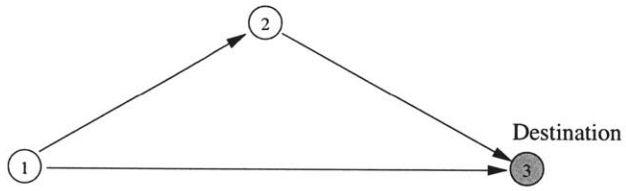

Figure 4-1: An Example Network

Table 4.1: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$	$(2,3)$		$(1,3)$
Departure Time (t)	0	1	4	0
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (4, 0.5)	(1, 0.5) (5, 0.5)	(6, 0.5) (8, 0.5)	(2, 0.1) (4, 0.9)

Given the assumption that all link travel times are independent random variables, we can easily compute all possible travel time values and their probabilities for each origin-destination pair and each departure time (refer to Section 3.1.4). Table 4.2 shows all path travel time realizations for origin-destination pairs 1–3 and 2–3.

Consider a trip from node 1 at time 0 to the destination node 3. There exist two paths, namely path a: $1 \rightarrow 2 \rightarrow 3$ and path b: $1 \rightarrow 3$ for this trip. The minimum possible travel time for this trip is 2 units of time. Note that both path a and path b are minimum possible travel time paths because they both have a minimum possible travel time realization. However path a is selected for this trip because the probability that the minimum possible travel time occurs on path a is higher than that of path b ($0.25 > 0.1$).

Table 4.2: Path Travel Time Realizations

Origin Node	Departure Time	Path	Travel Time	Probability
1	0	$1 \rightarrow 2 \rightarrow 3$	2	0.25
1	0	$1 \rightarrow 2 \rightarrow 3$	6	0.25
1	0	$1 \rightarrow 2 \rightarrow 3$	10	0.25
1	0	$1 \rightarrow 2 \rightarrow 3$	12	0.25
1	0	$1 \rightarrow 3$	2	0.1
1	0	$1 \rightarrow 3$	4	0.9
2	1	$2 \rightarrow 3$	1	0.5
2	1	$2 \rightarrow 3$	5	0.5
2	4	$2 \rightarrow 3$	6	0.5
2	4	$2 \rightarrow 3$	8	0.5

Table 4.3 summarizes the minimum possible travel time, its probability, and the selected path for each origin node for each departure time.

Table 4.3: Minimum Possible Travel Times, Their Probabilities, and Selected Paths

Origin Node	Departure Time	Minimum Possible Travel Time	Probability	Selected Path
1	0	2	0.25	$1 \rightarrow 2 \rightarrow 3$
2	1	1	0.5	$2 \rightarrow 3$
2	4	6	0.5	$2 \rightarrow 3$

It is interesting to note that although path a is a minimum possible travel time path for a trip from node 1 to node 3 at departure time 0, it has 0.25 probability of being the longest travel time path for the trip. This shows that a minimum possible travel time path may actually result in a longer travel time than other paths. This is a typical characteristic of the minimum possible travel time path problem because we are considering only minimum travel time realizations of paths.

4.1.2 Reasons to Study the Problem

As seen in the above example, a minimum possible travel time path may also have some probability of resulting in a longer travel time than other paths. In some cases, the probability that the minimum possible travel time occurs on that path is so small that it is rarely achieved. From these points of view, the minimum possible travel time path problem might

be thought to be of little importance. There are, however, several reasons to study this problem.

First, this problem is easy to formulate and to solve. Hence it helps us understand the characteristics of routing problems in stochastic time-dependent networks and provides us with basic building blocks for more complex routing problems. Second, the travel time distributions of minimum possible travel time paths often provide useful information for route planning purposes. Third, minimum possible travel time paths can aid in determining minimum expected travel time paths. As will be explained in Chapter 6, the currently known algorithms for the all-to-one minimum expected travel time paths problem have non-polynomial worst-case running time complexities. The information on minimum possible travel time paths may allow us to develop better algorithms or to speed up the existing algorithms for the all-to-one minimum expected travel time paths problem. Fourth, in the literature this problem is considered useful in dispatching urban emergency vehicles where the earliest possible arrival times at scenes are of paramount importance and the emergency vehicles have exclusive right of way.

4.2 Optimality Conditions

In order to solve the all-to-one minimum possible travel time paths problem with a real-size stochastic time-dependent network quickly, we need an efficient solution algorithm that exploits the characteristics of the problem and of the network. In this section, we formally derive optimality conditions[1] for the all-to-one minimum possible travel time paths problem, from which such an algorithm is developed. By optimality conditions, we mean algebraic conditions that the decision variables of the problem should satisfy. After deriving the optimality conditions, we make several useful observations.

We begin by defining additional notation needed for the derivation of the optimality conditions. Let $\overline{T}_{ij}^{c}(t)$ denote the travel time from node i to the destination node d when one leaves node i at time t by taking link (i,j) and then follows path c from node j to node d. $\overline{T}_{ij}^{c}(t)$ can be expressed as the sum of two random variables:

[1]In the literature, a set of optimality conditions for a problem is also referred to as a mathematical formulation of the problem.

$$\begin{aligned}\overline{T}_{ij}^{c}(t) &= \text{traversal time on link } (i,j) + \\ &\quad \text{travel time on path } c \text{ from node } j \text{ to node } d \\ &= T_{ij}(t) + L_j^c(t + T_{ij}(t)), \end{aligned} \tag{4.1}$$

where $L_j^c(t + T_{ij}(t))$ is a random variable denoting the travel time on path c from node j to node d at departure time $t + T_{ij}(t)$.

Let us denote by $\lambda_i(t)$ the minimum possible travel time from node i to node d when one leaves node i at time t, which is a decision variable of the problem. We can obtain $\lambda_i(t)$ by

$$\lambda_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \text{travel time values of } \overline{T}_{ij}^{c}(t) \right\} \right\}. \tag{4.2}$$

Using (4.1), we can rewrite (4.2) as

$$\lambda_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^{r}(t) + \text{travel time values of } L_j^c(t + \tau_{ij}^{r}(t)) \right\} \right\} \right\}. \tag{4.3}$$

Since random variable $L_j^c(t + \tau_{ij}^{r}(t))$ may take $l_j^{c^k}(t + \tau_{ij}^{r}(t))$, $k = 1, 2, \ldots, k_j^c(t)$ travel time values (refer to 3.1.4), (4.3) is the same as

$$\begin{aligned}\lambda_i(t) &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^{r}(t) + \min_{k} \left\{ l_j^{c^k}(t + \tau_{ij}^{r}(t)) \right\} \right\} \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^{r}(t) + \min_{c} \left\{ \min_{k} \left\{ l_j^{c^k}(t + \tau_{ij}^{r}(t)) \right\} \right\} \right\} \right\}. \end{aligned} \tag{4.4}$$

By the definition of $\lambda_j(t + \tau_{ij}^{r}(t))$, we have $\lambda_j(t + \tau_{ij}^{r}(t)) = \min_c \left\{ \min_k \left\{ l_j^{c^k}(t + \tau_{ij}^{r}(t)) \right\} \right\}$. Therefore (4.4) becomes

$$\lambda_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^{r}(t) + \lambda_j(t + \tau_{ij}^{r}(t)) \right\} \right\}. \tag{4.5}$$

Since $\lambda_d(t) = 0$ for all $t \in \mathcal{H}$, we have the following optimality conditions that $\lambda_i(t)$ for all $i \in \mathcal{N}$ and all $t \in \mathcal{H}$ should satisfy:

$$\lambda_i(t) = \begin{cases} \min_{j \in \mathcal{O}(i)} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^{r}(t) + \lambda_j(t + \tau_{ij}^{r}(t)) \right\} \right\} & \forall\, i \neq d,\ \forall\, t \in \mathcal{H}, \\ 0 & i = d,\ \forall\, t \in \mathcal{H}. \end{cases} \tag{4.6}$$

When departure time t is greater than $H-1$, it is assumed that the network becomes stochastic static and that $p_{T_{ij}(t)} = p_{T_{ij}(H-1)}$. Hence for $t > H-1$, (4.6) reduces to the following functional form:

$$\lambda_i(t) = \lambda_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \min\limits_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) + \lambda_j(t + \tau_{ij}^r(H)) \right\} \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H} \\ 0 & i = d,\ \forall t \notin \mathcal{H} \end{cases}$$

$$= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \min\limits_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) + \lambda_j(H) \right\} \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{4.7}$$

Note that in (4.7) we use the fact that for $t > H-1$, $\lambda_i(t)$ and $\lambda_j(t+\tau_{ij}^r(H))$ are respectively equal to $\lambda_i(H)$ and $\lambda_j(H)$ because no time-dependency exist when $t > H-1$.

In fact, without following the derivation steps we have just shown, we can also obtain (4.5) directly from the following argument: Since all link travel time values are positive, $\lambda_i(t)$ is equal to the minimum among all possible sums of *a value of the link travel time to an adjacent node j* ($\tau_{ij}^r(t)$) and *the minimum possible travel time from node j to node d when one leaves node j at time $t + \tau_{ij}^r(t)$* ($\lambda_j(t + \tau_{ij}(t))$). The mathematical formulation of this argument is exactly the same as (4.5).

It is possible that from node i at departure time t, more than one path result in the minimum possible travel time to node d, i.e. $\lambda_i(t)$. If this is the case, we assume that a traveler selects a path that has the highest probability associated with $\lambda_i(t)$. We need additional optimality conditions to take into account this case.

For each node j such that $j \in \mathcal{O}(i)$, we define set $\Omega_{ij}(t)$ as follows:

$$\Omega_{ij}(t) = \{r \mid r \in \mathcal{R}_{ij}(t),\ \tau_{ij}^r(t) + \lambda_j(t + \tau_{ij}^r(t)) = \lambda_i(t)\}. \tag{4.8}$$

$\Omega_{ij}(t)$ contains the indexes of the link travel time realizations of link (i,j) at departure time t, which engender the minimum possible travel time from node i to node d via link (i,j). If $\Omega_{ij}(t) = \emptyset$, it means that the travel time realizations of link (i,j) do not contribute to any of the minimum possible travel time realizations from node i to node d at departure time t. In other words, if $\Omega_{ij}(t) = \emptyset$, link (i,j) does not belong to any minimum possible travel time path from node i to node d at departure time t.

Let $q \in \Omega_{ij}(t)$. The probability of a minimum possible travel time realization obtained

by $(\tau_{ij}^q(t), p_{ij}^q(t))$ and a travel time realization of path c from node j to node d is computed as follows:

$$\begin{aligned}
&P[\overline{T}_{ij}^c(t) = \lambda_i(t) \text{ through } \tau_{ij}^q(t)] \\
&\quad = P[T_{ij}(t) = \tau_{ij}^q(t),\ L_j^c(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t))] \\
&\quad = P[T_{ij}(t) = \tau_{ij}^q(t)] \times P[L_j^c(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t)) \mid T_{ij}(t) = \tau_{ij}^q(t)] \\
&\quad = P[T_{ij}(t) = \tau_{ij}^q(t)] \times P[L_j^c(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t))] \\
&\quad = p_{ij}^q(t) \times P[L_j^c(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t))], \qquad (4.9)
\end{aligned}$$

where the third equality follows from the independence assumption on the link travel time random variables.

Let us denote by $\gamma_i(t)$ the highest probability that $\lambda_i(t)$ occurs, i.e. the highest value among the probabilities of all minimum possible travel time realizations from node i to node d at departure time t. $\gamma_i(t)$, which is also a decision variable of the problem, can be obtained by

$$\begin{aligned}
\gamma_i(t) &= \max_{j \in O(i)} \left\{ \max_c \left\{ P[\overline{T}_{ij}^c(t) = \lambda_i(t)] \right\} \right\} \\
&= \max_{j \in O(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ P[\overline{T}_{ij}^c(t) = \lambda_i(t) \text{ through } \tau_{ij}^q(t)] \right\} \right\} \right\} \\
&= \max_{j \in O(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times P[L_j^c(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t))] \right\} \right\} \right\}. \qquad (4.10)
\end{aligned}$$

Note that the travel time values of several realizations of $L_j^c(t + \tau_{ij}^q(t))$ can be equal to $\lambda_j(t + \tau_{ij}^q(t))$. Let $(l_j^{c^{k_s}}(t + \tau_{ij}^q(t)),\ p_j^{c^{k_s}}(t + \tau_{ij}^q(t)))$, $s = 1, 2, \ldots$ be such realizations, i.e. $l_j^{c^{k_s}}(t + \tau_{ij}^q(t)) = \lambda_j(t + \tau_{ij}^q(t))$, $s = 1, 2, \ldots$. Then we can rewrite (4.10) as follows:

$$\begin{aligned}
\gamma_i(t) &= \max_{j \in O(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \max_s \left\{ P[L_j^c(t + \tau_{ij}^q(t)) = l_j^{c^{k_s}}(t + \tau_{ij}^q(t))] \right\} \right\} \right\} \right\} \\
&= \max_{j \in O(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \max_s \left\{ p_j^{c^{k_s}}(t + \tau_{ij}^q(t)) \right\} \right\} \right\} \right\} \\
&= \max_{j \in O(i)} \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \max_c \left\{ \max_s \left\{ p_j^{c^{k_s}}(t + \tau_{ij}^q(t)) \right\} \right\} \right\} \right\} \\
&= \max_{j \in O(i)} \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\} \right\}, \qquad (4.11)
\end{aligned}$$

where we use $\gamma_j(t+\tau_{ij}^q(t)) = \max_c \left\{ \max_s \left\{ p_j^{c^{k_s}}(t+\tau_{ij}^q(t)) \right\} \right\}$ by the definition of $\gamma_j(t+\tau_{ij}^q(t))$.

By convention, we set $\gamma_d(t)$ to 1 for all $t \in \mathcal{H}$. The optimality conditions that $\gamma_i(t)$, $\forall i \in \mathcal{N}$, $\forall t \in \mathcal{H}$ should satisfy are therefore given by

$$\gamma_i(t) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ 1 & i = d,\ \forall t \in \mathcal{H}. \end{cases} \tag{4.12}$$

For $t > H - 1$, due to the static property of the network, we have

$$\gamma_i(t) = \gamma_i(H) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{q \in \Omega_{ij}(H)} \left\{ p_{ij}^q(H) \times \gamma_j(H) \right\} \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 1 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{4.13}$$

We associate each node (i, t) in the time-space network with a pair of labels $(\lambda_i(t), \gamma_i(t))$. Then solving the all-to-one minimum possible travel time paths problem is equivalent to determining all $(\lambda_i(t), \gamma_i(t))$ that satisfy the optimality conditions (4.6), (4.7), (4.12), and (4.13).

The design of an efficient solution algorithm for the all-to-one minimum possible travel time paths problem depends on how fast we can determine all label pairs $(\lambda_i(t), \gamma_i(t))$ that satisfy the optimality conditions. Now we introduce an important proposition regarding this issue.

Proposition 1. *All label pairs $(\lambda_i(t), \gamma_i(t))$ satisfying the optimality conditions (4.6), (4.7), (4.12), and (4.13) can be determined in a decreasing order of departure time in a single pass.*

Proof. Since all $\tau_{ij}^r(t)$ are assumed to be positive, the optimality conditions (4.6) and (4.12) indicate that only $(\lambda_j(\tilde{t}), \gamma_j(\tilde{t}))$ such that $\tilde{t} > t$, can influence $(\lambda_i(t), \gamma_i(t))$. Hence starting from $t = H$ and proceeding to $t = 0$, we can determine $(\lambda_i(t), \gamma_i(t))$ for all $i \in \mathcal{N}$ in a decreasing order of departure time. In order to prove that $(\lambda_i(t), \gamma_i(t))$ can be computed in a single pass, we argue that once $(\lambda_i(t), \gamma_i(t))$ are determined, they do not change. We prove this by contradiction. Suppose that we compute the value of $(\lambda_i(t), \gamma_i(t))$ for some i, t and then modify it later. It means that some $(\lambda_j(\tilde{t}), \gamma_j(\tilde{t}))$ for $j \in \mathcal{O}(i)$ and $\tilde{t} > t$ changes from its previous value (either $\lambda_j(\tilde{t})$ decreases or $\gamma_j(\tilde{t})$ increases). This in turn indicates that some $(\lambda_k(\hat{t}), \gamma_k(\hat{t}))$ for $k \in \mathcal{O}(j)$ and $\hat{t} > \tilde{t}$ changes and so on, until we arrive at the static domain $(t > H - 1)$. In the static domain, however, all $(\lambda_i(t), \gamma_i(t))$, $i \in \mathcal{N}$, $t > H - 1$ do not change once they are set to $(\lambda_i(H), \gamma_i(H))$ by (4.7) and (4.13). This means that all

$(\lambda_i(t), \gamma_i(t))$ at $t = H-1$ do not change after they are initially determined by (4.6). We can repeat this argument in a decreasing order of departure time until we arrive at departure time t. Therefore $(\lambda_i(t), \gamma_i(t))$ for some i, t does not change once it is determined, which contradicts the assumption. This concludes that all $(\lambda_i(t), \gamma_i(t))$ can be determined in a decreasing order of departure time in a single pass. □

Note that Proposition 1 does not hold if some of $\tau_{ij}^r(t)$ have negative values. In that case, the time-space network is no longer guaranteed to be acyclic.

Since we want to find a path with the minimum possible travel time and the highest probability of its occurrence, one might view the problem as a multi-objective problem. However this is not correct because the travel time value is the primary objective we try to minimize no matter what the associated probability would be. Only when there exist more than one path that result in the minimum possible travel time, we compare the associated probabilities to select one of the paths. Therefore there is no trade-off situation that happens in a multi-objective problem.

4.3 Minimum Possible Travel Time Paths in the Static Domain

In Proposition 1, we have shown that the label pairs $(\lambda_i(t), \gamma_i(t))$ can be determined in a decreasing order of departure time. This implies that we need to first compute $(\lambda_i(H), \gamma_i(H))$ in order to calculate other label pairs $(\lambda_i(t), \gamma_i(t))$, $t \leq H-1$. We discuss this issue in this section.

Consider the optimality conditions (4.7) again.

$$\lambda_i(t) = \lambda_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \min\limits_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) + \lambda_j(H) \right\} \right\} & \forall\, i \neq d,\ \forall\, t \notin \mathcal{H}, \\ 0 & i = d,\ \forall\, t \notin \mathcal{H}. \end{cases} \tag{4.14}$$

Since $\lambda_j(H)$ is irrelevant to $r \in \mathcal{R}_{ij}(H)$, we have

$$\min_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) + \lambda_j(H) \right\} = \min_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) \right\} + \lambda_j(H). \tag{4.15}$$

Let $z = \arg\min_{r \in \mathcal{R}_{ij}(H)} \left\{ \tau_{ij}^r(H) \right\}$. Then (4.14) is the same as

$$\lambda_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \tau_{ij}^z(H) + \lambda_j(H) \right\} & \forall i \neq d, \\ 0 & i = d. \end{cases} \tag{4.16}$$

Notice that (4.16) is a set of optimality conditions for the all-to-one shortest paths problem in a deterministic static network where the travel time on link (i,j) is equal to $\tau_{ij}^z(H)$ (It is a variant of Bellman's equation for the one-to-all static shortest paths problem.). This means that solving the all-to-one minimum possible travel time paths problem in the static domain is equivalent to solving an all-to-one static shortest paths problem where link travel times are set to their minimum values at departure time H.

Once all shortest paths are found by solving an all-to-one static shortest paths problem, i.e. once all $\lambda_i(H)$ are determined, $\gamma_i(H)$ are computed by

$$\gamma_i(H) = \prod_{(v,w) \in P_{i \rightsquigarrow d}} p_{vw}^z(H), \tag{4.17}$$

where $P_{i \rightsquigarrow d}$ is the shortest path from node i to the destination node d and $z = \arg\min_{r \in \mathcal{R}_{vw}(H)} \left\{ \tau_{vw}^r(H) \right\}$.

Computing $\gamma_i(H)$ by (4.17) is essentially identical to using (4.13). To see this, we rewrite (4.13) for $i \neq d$ as

$$\gamma_i(H) = \max_{j \in \mathcal{O}(i)} \left\{ \max_{q \in \Omega_{ij}(H)} \left\{ p_{ij}^q(H) \right\} \times \gamma_j(H) \right\}. \tag{4.18}$$

If link (i,j) belongs to the shortest path from node i to node d, then $\Omega_{ij}(H) = \{z\}$ where $z = \arg\min_{r \in \mathcal{R}_{ij}(H)} \{\tau_{ij}^r(H)\}$. Otherwise, $\Omega_{ij}(H) = \emptyset$. Letting $\delta_{ij} = 1$ if link (i,j) is on the shortest path and $\delta_{ij} = 0$ otherwise, (4.18) can be rewritten as

$$\gamma_i(H) = \max_{j \in \mathcal{O}(i)} \left\{ \delta_{ij} \times p_{ij}^z(H) \times \gamma_j(H) \right\} = p_{iw}^z(H) \times \gamma_w(H), \tag{4.19}$$

where w is the successor of node i on the shortest path. Notice that (4.17) can be obtained from the recursive formula (4.19).

4.4 Paths Construction

A label pair $(\lambda_i(t), \gamma_i(t))$ tells us only the minimum possible travel time and the probability

of its occurrence for the corresponding origin node and departure time. In order to construct the minimum possible travel time path, we need to record the nodes constituting the path.

Let us denote by $s_i(t)$ the successor of node i on the minimum possible travel time path when one leaves node i at time t. For $t \in \mathcal{H}$, the minimum possible travel time path from node i to node d at departure time t is finally constructed by (4.12), not by (4.6). Therefore when $t \in \mathcal{H}$, $s_i(t)$ should be determined by the following functional form where we define $s_d(t) = d$:

$$s_i(t) = \begin{cases} \arg\max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ d & i = d,\ \forall t \in \mathcal{H}. \end{cases} \tag{4.20}$$

In the static domain ($t > H - 1$), the minimum possible travel time path from node i to node d is just the shortest path from node i to node d obtained by (4.16). Hence $s_i(t)$ are determined by the following functional form:

$$s_i(t) = s_i(H) = \begin{cases} \arg\min\limits_{j \in \mathcal{O}(i)} \left\{ \tau_{ij}^z(H) + \lambda_j(H) \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ d & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{4.21}$$

Sometimes it is useful to record the arrival time at the successor node. Let $\pi_i(t)$ denote the arrival time at node $s_i(t)$ when one leaves node i at time $t \in \mathcal{H}$. We first determine k such that

$$k = \arg\max_{q \in \Omega_{is_i(t)}(t)} \left\{ p_{is_i(t)}^q(t) \times \gamma_{s_i(t)}(t + \tau_{is_i(t)}^q(t)) \right\}. \tag{4.22}$$

Then $\pi_i(t)$ is obtained by $\pi_i(t) = t + \tau_{is_i(t)}^k(t)$. When $t > H - 1$, $\pi_i(t) = H$ always.

4.5 Solution Algorithms

In this section, we present solution algorithms for the all-to-one minimum possible travel time paths problem and determine their theoretical running time complexities.

We derived the optimality conditions for the all-to-one minimum possible travel time paths problem in Section 4.2. The solution algorithms presented here are based on the optimality conditions. However depending on how to update the label pairs $(\lambda_i(t), \gamma_i(t))$,

their theoretical running time complexities are quite different.

4.5.1 Algorithm MPTTP

The first algorithm we present is developed from the optimality conditions and Proposition 1 discussed in Section 4.2. We call this algorithm *Algorithm MPTTP*. The algorithm is described in Algorithm 1.

It starts by initializing the label quadruplets $(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t))$ in Step 1. In Step 2, $(\lambda_i(H), \gamma_i(H), s_i(H), \pi_i(H))$ are determined. As discussed in Section 4.3 and Section 4.4, this is nothing but solving an all-to-one shortest paths problem in a deterministic static network. Any all-to-one static shortest paths algorithm (for example, a backward application of Dijkstra's algorithm) can be used. Notice that we are passing to an all-to-one static shortest paths algorithm the set of nodes, the set of links, the set of minimum link travel time values and their probabilities at departure time H, and the destination node as input. In Step 3, we determine all label quadruplets for $t \in \mathcal{H}$. We are using Proposition 1 for this step, so the outmost For loop is executed in a decreasing order of departure time. Because the labels do not change once they are determined, Algorithm MPTTP can be viewed as a label-setting algorithm. Algorithm MPTTP terminates after a finite number of steps because each For loop in Step 3 is executed a finite number of times.

Now let us analyze the theoretical worst-case running complexity of Algorithm MPTTP. It takes $\Theta(nH)$ time to initialize all label quadruplets in Step 1. Let $SP(n,m)$ denote the running time of an algorithm used in Step 2 to solve an all-to-one static shortest paths problem. If the original implementation of Dijkstra's algorithm is used, for instance, $SP(n,m) = O(n^2)$. Let us assume that an optimal algorithm in the running time sense is available, whose running time is $SP(n,m) = \Theta(f(n,m))$. To compute the running time complexity of Step 3, we count the total number of elementary operations performed in Step 3. The total number of elementary operations is the following quantity multiplied by some constant:

$$W = \sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{N}\setminus\{d\}} \Big(\sum_{j\in\mathcal{O}(i)} |\mathcal{R}_{ij}(t)| + \sum_{j\in\mathcal{O}(i)} |\mathcal{R}_{ij}(t)| + \sum_{j\in\mathcal{O}(i)} |\mathcal{Q}_{ij}(t)| + \sum_{j\in\mathcal{O}(i)} |\mathcal{Q}_{ij}(t)| + |\mathcal{Q}_{is_i(t)}(t)| + 1 \Big).$$

Algorithm 1 Algorithm MPTTP

Step 1: Initialization

$(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t)) \leftarrow (\infty, 0, \infty, \infty), \quad \forall\, i \neq d,\ \forall\, t \in \mathcal{H}$

$(\lambda_d(t), \gamma_d(t), s_d(t), \pi_d(t)) \leftarrow (0, 1, d, t), \quad \forall\, t \in \mathcal{H}$

Step 2: Minimum Possible Travel Time Paths in Static Domain

$(\lambda_i(H), \gamma_i(H), s_i(H), \pi_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{(\tau_{ij}^z(H),\ p_{ij}^z(H))\}, d)$

Step 3: Minimum Possible Travel Time Paths in Time-Dependent Domain

For $t \leftarrow H - 1$ down to 0 **do**

  For $i \in \mathcal{N} \setminus \{d\}$ **do**

    $\lambda_i(t) \leftarrow \min_{j \in \mathcal{O}(i)} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^r(t) + \lambda_j(t + \tau_{ij}^r(t)) \right\} \right\}$

    For $j \in \mathcal{O}(i)$ **do**

      $\Omega_{ij}(t) \leftarrow \left\{ r \mid r \in \mathcal{R}_{ij}(t),\ \tau_{ij}^r(t) + \lambda_j(t + \tau_{ij}^r(t)) = \lambda_i(t) \right\}$

    End (For)

    $\gamma_i(t) \leftarrow \max_{j \in \mathcal{O}(i)} \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\} \right\}$

    $s_i(t) \leftarrow \arg\max_{j \in \mathcal{O}(i)} \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\} \right\}$

    $k \leftarrow \arg\max_{q \in \Omega_{is_i(t)}(t)} \left\{ p_{is_i(t)}^q(t) \times \gamma_{s_i(t)}(t + \tau_{is_i(t)}^q(t)) \right\}$

    $\pi_i(t) \leftarrow t + \tau_{is_i(t)}^k(t)$

  End (For)

End (For)

Since $|\mathcal{R}_{ij}(t)| \geq |\mathcal{Q}_{ij}(t)|$ for all $(i,j) \in \mathcal{A}$ and all $t \in \mathcal{H}$, we have

$$\begin{aligned} W &\leq \sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{N}\setminus\{d\}} \Big(4 \sum_{j\in\mathcal{O}(i)} |\mathcal{R}_{ij}(t)| + |\mathcal{R}_{is_i(t)}(t)| + 1\Big) \\ &\leq 6 \sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{N}\setminus\{d\}} \sum_{j\in\mathcal{O}(i)} |\mathcal{R}_{ij}(t)| \\ &\leq 6 \sum_{t\in\mathcal{H}} \sum_{(i,j)\in\mathcal{A}} |\mathcal{R}_{ij}(t)| \\ &= 6\widetilde{m}, \end{aligned}$$

where $\widetilde{m} = \sum_{t\in\mathcal{H}} \sum_{(i,j)\in\mathcal{A}} |\mathcal{R}_{ij}(t)|$ which is the total number of links in the time-space network. Therefore the running time complexity of Step 3 is $O(\widetilde{m})$. We can also show that it is $\Omega(\widetilde{m})$. These together imply that it takes $\Theta(\widetilde{m})$ time to complete Step 3.

To conclude, the theoretical worst-case running time complexity of Algorithm MPTTP is $\Theta(nH + f(n,m) + \widetilde{m}) = \Theta(\max(f(n,m), \widetilde{m}))$. We record this as a proposition.

Proposition 2. *The worst-case running time complexity of Algorithm MPTTP is* $\Theta(\max(f(n,m), \widetilde{m}))$ *where* $\Theta(f(n,m))$ *is the lowest possible time to solve an all-to-one static shortest paths problem in Step 2.*

Now let us see how efficient Algorithm MPTTP is. Every solution algorithm for the all-to-one minimum possible travel time paths problem should perform the following tasks at least:

- Initialize all label quadruplets $(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t))$
- Determine $(\lambda_i(H), \gamma_i(H), s_i(H), \pi_i(H))$
- Examine each link travel time realization at least once

The first task requires $\Theta(nH)$ time. For the second task, we need $\Theta(f(n,m))$ time at minimum. It requires $\Omega(\widetilde{m})$ time to examine each link travel time realization at least once, because it is the same as examining all links in the time-space network at least once. Hence $\Omega(nH + f(n,m) + \widetilde{m}) = \Omega(\max(f(n,m), \widetilde{m}))$ is the lowest possible running time to solve the problem.

Proposition 3. *Algorithm MPTTP is optimal in terms of running time. No algorithm with a better running time complexity can be found.*

Proof. The running time of Algorithm MPTTP is the same as the lowest possible running time to solve the problem. Therefore Algorithm MPTTP is optimal in the running time sense. □

When we implement Algorithm MPTTP in computer, we need to create a set $\Omega_{ij}(t)$ for each link (i, j) and departure time t. It is possible to dynamically allocate and de-allocate memory for $\Omega_{ij}(t)$, but Algorithm MPTTP is not ideal in terms of memory management and coding convenience. In Algorithm 2, we present *Algorithm MPTTP2* that is essentially the same algorithm as Algorithm MPTTP, but described in a different way to make a computer implementation easier. It is not difficult to see that the worst-case running time complexity of Algorithm MPTTP2 is also $\Theta(\max(f(n, m), \widetilde{m}))$.

There is another reason that Algorithm MPTTP2 is preferred to Algorithm MPTTP. Algorithm MPTTP first accesses a node (Line 8) and then examines the outgoing links from the node (Line 9). Algorithm MPTTP2 examines all links directly from the link set $\mathcal{A}$ (Line 8). Although the theoretical asymptotic worst-case running time complexities of the two algorithms are the same when $n \leq m$, Algorithm MPTTP requires more time than Algorithm MPTTP2 in practice because of the node access times. Algorithm MPTTP, however, directly incorporates the optimality conditions. Hence it might be more understandable than Algorithm MPTTP2. From now on, we describe most solution algorithms developed in the thesis as we described Algorithm MPTTP, i.e. first access a node and then examine the outgoing links from the node. However, for computational tests on the algorithms in Chapter 7, we implement the algorithms as shown in Algorithm MPTTP2, i.e. examine all links directly from the link set $\mathcal{A}$.

In Appendix B, we will demonstrate how Algorithm MPTTP2 works using an example.

4.5.2 Algorithm LEAST

Miller-Hooks [16] and Miller-Hooks and Mahmassani [17] propose another efficient solution algorithm called *Algorithm LEAST* for the all-to-one minimum possible travel time paths problem. We restate the algorithm in Algorithm 3 in order to compare it with Algorithm MPTTP.

The original version of Algorithm LEAST does not have Step 2. We add Step 2 to the original version because firstly it is a necessary component and secondly it allows us to

Algorithm 2 Algorithm MPTTP2

Step 1: Initialization

$(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t)) \leftarrow (\infty, 0, \infty, \infty), \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

$(\lambda_d(t), \gamma_d(t), s_d(t), \pi_d(t)) \leftarrow (0, 1, d, t), \quad \forall t \in \mathcal{H}$

Step 2: Minimum Possible Travel Time Paths in Static Domain

$(\lambda_i(H), \gamma_i(H), s_i(H), \pi_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{(\tau^z_{ij}(H),\ p^z_{ij}(H))\}, d)$

Step 3: Minimum Possible Travel Time Paths in Time-Dependent Domain

For $t \leftarrow H - 1$ down to 0 **do**

  For $(i, j) \in \mathcal{A}$ **do**

    For $r \in \mathcal{R}_{ij}(t)$ **do**

      $\mu \leftarrow \tau^r_{ij}(t) + \lambda_j(t + \tau^r_{ij}(t))$

      $\nu \leftarrow p^r_{ij}(t) \times \gamma_j(t + \tau^r_{ij}(t))$

      If $(\mu < \lambda_i(t)$ or $(\mu = \lambda_i(t)$ and $\nu > \gamma_i(t)))$ **then**

        $\lambda_i(t) \leftarrow \mu$

        $\gamma_i(t) \leftarrow \nu$

        $s_i(t) \leftarrow j$

        $\pi_i(t) \leftarrow t + \tau^r_{ij(t)}(t)$

      End (If)

    End (For)

  End (For)

End (For)

Algorithm 3 Algorithm LEAST

Step 1: Initialization

$(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t)) \leftarrow (\infty, 0, \infty, \infty), \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

$(\lambda_d(t), \gamma_d(t), s_d(t), \pi_d(t)) \leftarrow (0, 1, d, t), \quad \forall t \in \mathcal{H}$

Create a queue S

Enqueue d to S; flag $= 0$

Step 2: Minimum Possible Travel Time Paths in Static Domain

$(\lambda_i(H), \gamma_i(H), s_i(H), \pi_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{(\tau_{ij}^z(H),\ p_{ij}^z(H))\}, d)$

Step 3: Choose Current Node

If $S = \emptyset$ **then** stop

Else dequeue a node (called j) from S

Step 4: Update Labels

For $i \in \mathcal{I}(j)$ **do**

For $t \leftarrow 0$ to $H - 1$ **do**

$\mu \leftarrow \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \tau_{ij}^r(t) + \lambda_j(t + \tau_{ij}^r(t)) \right\}$

$\mathcal{Q} \leftarrow \{r \mid r \in \mathcal{R}_{ij}(t),\ \tau_{ij}^r(t) + \lambda_j(t + \tau_{ij}^r(t)) = \mu\}$

$k \leftarrow \arg\max_{q \in \mathcal{Q}} \left\{ p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right\}$

$\nu \leftarrow p_{ij}^k(t) \times \gamma_j(t + \tau_{ij}^k(t))$

If ($\mu < \lambda_i(t)$ or ($\mu = \lambda_i(t)$ and $\nu > \gamma_i(t)$)) **then**

$\lambda_i(t) \leftarrow \mu$

$\gamma_i(t) \leftarrow \nu$

$s_i(t) \leftarrow j$

$\pi_i(t) \leftarrow t + \tau_{ij(t)}^k(t)$

flag $\leftarrow 1$

End (If)

End (For)

If ($i \notin S$ and flag $= 1$) **then**

Enqueue i to S; flag $= 0$

End (If)

End (For)

Go to **Step 3**

compare Algorithm LEAST with Algorithm MPTTP with the same yardstick.

4.5.3 Comparison of the Algorithms

It is shown in Miller-Hooks [16] and in Miller-Hooks and Mahmassani [17] that when $|\mathcal{R}_{ij}(t)| = R$ for all $(i,j) \in \mathcal{A}$ and all $t \in \mathcal{H}$, the worst-case running time complexity of Algorithm LEAST is $O(\max(f(n,m), n^3H^2R))$. Under the same assumption, the worst-case running time complexity of Algorithm MPTTP becomes $\Theta(\max(f(n,m), mHR))$.

For networks under consideration in this thesis, $m \leq n(n-1) < n^2$. Therefore, $\max(f(n,m), n^3H^2R) \geq \max(f(n,m), mHR)$ for any values of n, m, H, and R. This concludes that theoretically, the running time of Algorithm MPTTP is always better than or equal to the running time of Algorithm LEAST in the worst-case. This is an obvious result because the running time complexity of Algorithm MPTTP is optimal. In Chapter 7, we will compare the practical running times of the two algorithms.

It is interesting to note that Step 4 of Algorithm LEAST and Step 3 of Algorithim MPTTP2 look alike. The main difference is that the outmost For loop in Step 4 of Algorithm LEAST (Line 12) is performed in terms of nodes, whereas that of Algorithm MPTTP2 (Line 7) is executed with regard to depature times in a decreasing order. Specifically, Algorithm LEAST fixes a node and then updates the label quadruplets of the node for all departure times, i.e. $(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t))$, $\forall\, t \in \mathcal{H}$. On the contrary, Algorithm MPTTP fixes a departure time and then updates the label quadruplets for all nodes at the departure time, i.e. $(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t))$, $\forall\, i \in \mathcal{N}$. This implies that the recognition of Proposition 1 is a key factor in designing a theoretically fastest algorithm.

4.6 Exact Probability of the Minimum Possible Travel Time

Suppose that there exist several minimum possible travel time realizations from node i to node d at departure time t. Then a minimum possible travel time realization with the highest probability and a path that has the realization are selected. It is possible that each of the minimum possible travel time realizations belongs to a different path. It is also possible that some of the minimum possible travel time realizations belong to the same path. If the former is the case, the probability that the minimum possible travel time occurs on the selected path is higher than that of any other path. If the latter is the case, however,

the probability that the minimum possible travel time occurs on the selected path may be actually lower than that of some other path.

This happens because we differentiate between all travel time realizations of a path even if some of those have the same travel time value and the optimality conditions are obtained based on a single path travel time realization. To illustrate this concretely, let us condsider the network example shown in Figure 4-2 and Table 4.4.

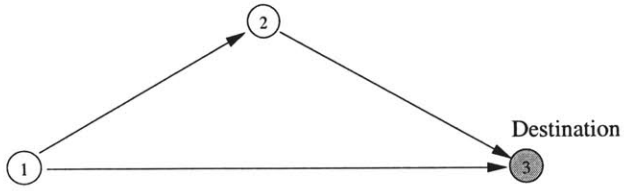

Figure 4-2: An Example Network

Table 4.4: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$	$(2,3)$		$(1,3)$
Depature Time (t)	0	1	2	0
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (2, 0.5)	(2, 0.4) (5, 0.6)	(1, 0.5) (8, 0.5)	(3, 0.3) (4, 0.7)

Table 4.5 shows all path travel time realizations of this example. Let us consider a trip from node 1 to node 3 at departure time 0. Notice that the two paths, namely path a: $1 \rightarrow 2 \rightarrow 3$ and path b: $1 \rightarrow 3$, result in the same minimum possible travel time, which is 3 units of time. If the solution algorithms discussed in Section 4.5 are used to determine the minimum possible travel time path from node 1 to node 3 at departure time 0, path b is selected because it has a minimum possible travel time realization with the highest probability (0.3).

However path a has two minimum possible travel time realizations, so the likelihood that the minimum possible travel time occurs on path a is higher than that of path b $(0.2 + 0.25 > 0.3)$. Therefore it is better to take path a.

The exact probability that the minimum possible travel time is realized on a path is obatined by summing up the probabilities of all minimum possible travel time realizations on the path. The exact probabilities that the minimum possible travel time occurs on path

Table 4.5: Path Travel Time Realizations

Origin Node	Departure Time	Path	Travel Time	Probability
1	0	$1 \rightarrow 2 \rightarrow 3$	**3**	**0.2**
1	0	$1 \rightarrow 2 \rightarrow 3$	6	0.3
1	0	$1 \rightarrow 2 \rightarrow 3$	**3**	**0.25**
1	0	$1 \rightarrow 2 \rightarrow 3$	10	0.25
1	0	$1 \rightarrow 3$	**3**	**0.3**
1	0	$1 \rightarrow 3$	4	0.7
2	1	$2 \rightarrow 3$	2	0.4
2	1	$2 \rightarrow 3$	5	0.6
2	2	$2 \rightarrow 3$	1	0.5
2	2	$2 \rightarrow 3$	8	0.5

a and on path b in the above example are 0.45 and 0.3, respectively.

$\gamma_i(t)$ defined in Section 4.2 is actually a lower bound on the exact probability that $\lambda_i(t)$ occurs on a path. Hence the solution algorithms presented in Section 4.5 may fail to find a path whose exact probability of having the minimum possible travel time is higher than any other path.

To compute the exact probabilities that the minimum travel times occur, one may attempt to change the optimality conditions (4.12) as follows:

$$\gamma_i(t) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{k \in \mathcal{N} \setminus \{i,j\}} \left\{ \sum\limits_{q \in \Omega_{ij}^k(t)} \left(p_{ij}^q(t) \times \gamma_j(t + \tau_{ij}^q(t)) \right) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ 1 & i = d,\ \forall t \in \mathcal{H}, \end{cases} \tag{4.23}$$

where $\Omega_{ij}^k(t)$ for $k \in \mathcal{N} \setminus \{i,j\}$ is defined as

$$\Omega_{ij}^k(t) = \{q \mid q \in \Omega_{ij}(t),\ s_j(t + \tau_{ij}^q(t)) = k\}. \tag{4.24}$$

The intention of this modification is to add up the probabilities of minimum possible travel time realizations that belong to the same path from node i to node d. This modification, however, does not work. The minimum possible travel time realizations obtained by the indexes in set $\Omega_{ij}^k(t)$ are clearly associated with ("going through") the same subpath, $i \rightarrow j \rightarrow k$. But there is no guarantee that those realizations are also associated with the

same subpath from node k to node d.

Let us illustrate this with the example network depicted in Figure 4-3 and Table 4.6. Consider a trip from node 1 to node 5 at departure time 0. As shown in Table 4.7, three paths, $1 \to 2 \to 3 \to 5$, $1 \to 2 \to 3 \to 4 \to 5$, and $1 \to 5$, provide the same minimum possible travel time for the trip, which is 5 units of time. Note that only one minimum possible travel time realization exists on each path. Therefore the associated probabilities (0.24, 0.1, and 0.3) are the exact probabilities that the minimum possible travel time occurs on the three paths respectively. Since path $1 \to 5$ has the highest probability, it is chosen for the trip.

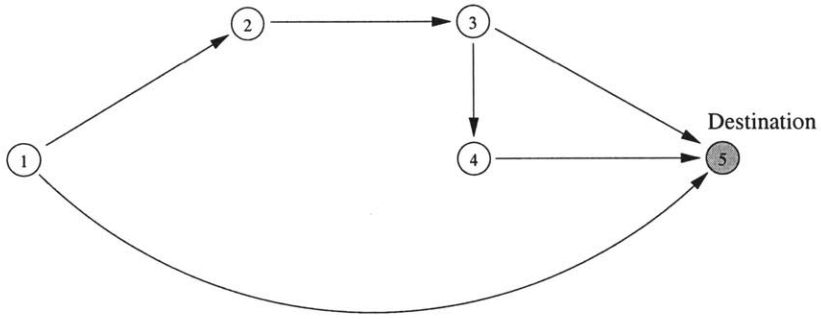

Figure 4-3: An Example Network

Table 4.6: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$	$(1,5)$	$(2,3)$		$(3,4)$		
Departure Time (t)	0	0	1	2	2	3	4
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (2, 0.5)	(5, 0.3) (7, 0.7)	(1, 0.8) (3, 0.2)	(1, 0.8) (2, 0.2)	(1, 0.8) (4, 0.2)	(1, 0.5) (2, 0.5)	(2, 0.5) (4, 0.5)

Link (i,j)	$(3,5)$			$(4,5)$		
Departure Time (t)	2	3	4	3	4	5
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(3, 0.6) (4, 0.4)	(4, 0.5) (5, 0.5)	(4, 0.5) (6, 0.5)	(3, 0.5) (4, 0.5)	(1, 0.5) (2, 0.5)	(2, 0.5) (3, 0.5)

Now let us apply (4.23) to this trip. Note that $\Omega_{15}(0) = \{1\}$ and $\Omega_{12}(0) = \{1, 2\}$. Since $s_2(0+\tau_{12}^1(0)) = s_2(1) = 3$ and $s_2(0+\tau_{12}^2(0)) = s_2(2) = 3$, we have $\Omega_{12}^3(0) = \{1, 2\}$. Therefore $\gamma_1(0)$ is computed by

Table 4.7: Path Travel Time Realizations from Node 1 to Node 5 at Departure Time 0

Origin Node	Departure Time	Path	Travel Time	Probability
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	**5**	**0.24**
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	6	0.16
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	8	0.05
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	10	0.05
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	7	0.2
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	8	0.2
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	8	0.05
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 5$	10	0.05
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	6	0.16
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	7	0.16
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	8	0.04
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	9	0.04
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	8	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	9	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	10	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	11	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	**5**	**0.1**
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	6	0.1
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	7	0.1
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	8	0.1
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	8	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	9	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	10	0.025
1	0	$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$	11	0.025
1	0	$1 \rightarrow 5$	**5**	**0.3**
1	0	$1 \rightarrow 5$	7	0.7

$$\begin{aligned}\gamma_1(0) &= \max\left\{p_{15}^1(0) \times \gamma_5(0+\tau_{15}^1(0)),\ p_{12}^1(0) \times \gamma_2(0+\tau_{12}^1(0)) + p_{12}^2(0) \times \gamma_2(0+\tau_{12}^2(0))\right\} \\ &= \max\left\{p_{15}^1(0) \times \gamma_5(5),\ p_{12}^1(0) \times \gamma_2(1) + p_{12}^2(0) \times \gamma_2(2)\right\} \\ &= \max\left\{0.3 \times 1,\ 0.5 \times 0.48 + 0.5 \times 0.2\right\} \\ &= 0.34.\end{aligned}$$

According to this computation, the minimum possible travel time path includes link $(1,2)$, and the probability that the minimum possible travel time occurs on the path is 0.34. Clearly, this is not correct. We explain the reason using the time-space network of the example network.

In Figure 4-4[2], the two thick solid lines connecting node 1 at time 0 and node 5 at time 5 indicate the minimum possible travel time realizations from node 1 to node 5 through node 2 at departure time 0. Because the two realizations are associated with the same subpath, $1 \rightarrow 2 \rightarrow 3$, their probabilities are added together in (4.23). However, the realizations are associated with different subpaths from node 3 to node 5, indicating that they belong to different paths from node 1 to node 5 and their probabilities cannot be summed up. The minimum possible travel time realization with the highest probability from node 1 to node 5 at departure time 0 is the thick dashed line, and it is on path $1 \rightarrow 5$.

The above example suggests that in order to correctly compute the exact probability that the minimum possible travel time occurs on a path from node i to node d at departure time t, for every minimum possible travel time realization, we have to examine all successor nodes, i.e. all nodes between node i and node d. Obviously, this requires an exponential running time. Hence any solution algorithm, if possible, for the all-to-one minimum possible travel time paths problem, which guarantees to compute the exact probabilities of the minimum possible travel times, has an exponential worst-case running time complexity.

4.7 Extensions

In this section, we study three extensions of the all-to-one minimum possible travel time paths problem. First, we consider the *all-to-one minimum possible travel cost paths problem* where link travel costs are also time-dependent discrete random variables and the minimum

[2]Links corresponding to $T_{35}(3)$, $T_{35}(4)$, and $T_{45}(5)$ are omitted.

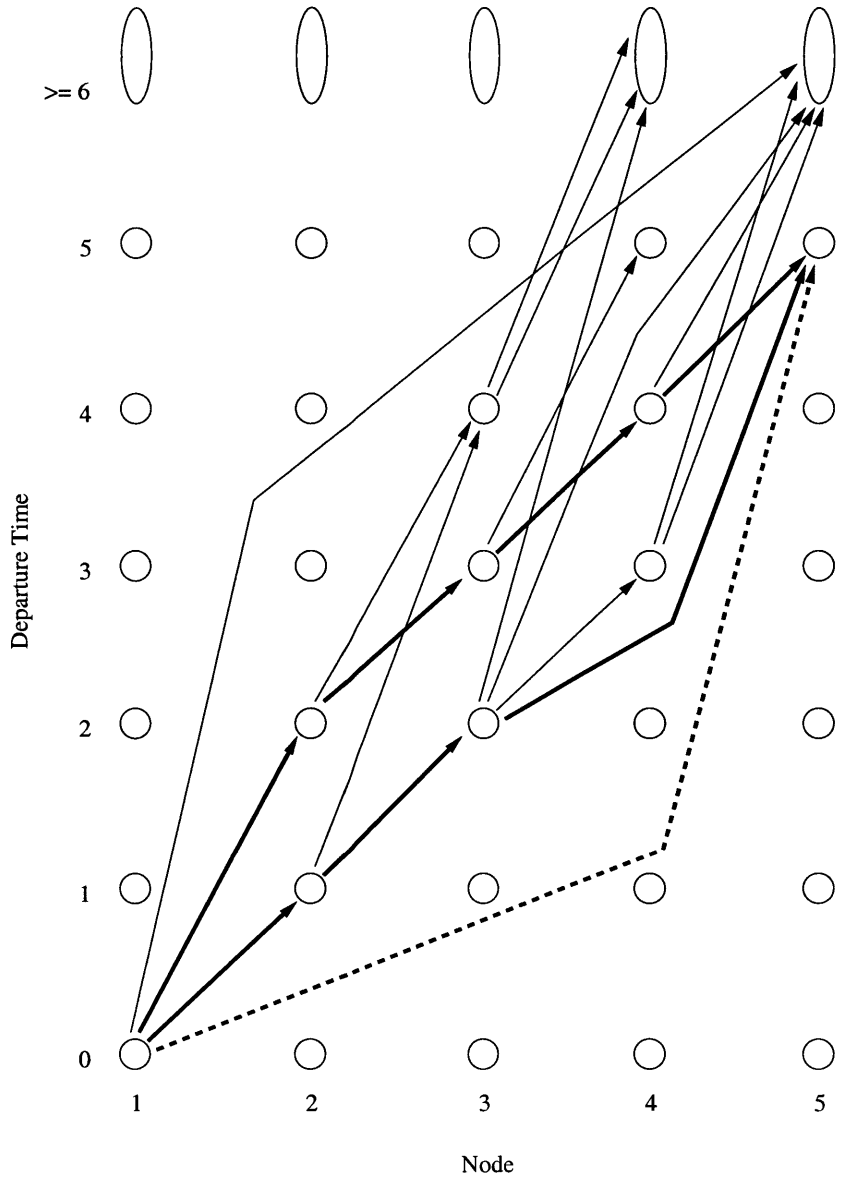

Figure 4-4: Time-Space Network of the Network in Figure 4-3, $H = 6$

possible travel cost is used as a path selection criterion. Second, we discuss how to select several paths from each node to the destination node for each departure time, based on minimum travel time realizations. We call this problem the *all-to-one $k-$minimum path travel time realizations problem.* Third, we introduce an algorithm for the problem of determining all-to-one k shortest paths in deterministic time-dependent networks by suppressing stochasticity in the all-to-one $k-$minimum path travel time realizations problem. The third problem is referred to as the *all-to-one $k-$dynamic shortest paths problem.*

4.7.1 All-to-One Minimum Possible Travel Cost Paths Problem

In this problem, we consider a stochastic time-dependent network where both link travel times and link travel costs are given in the form of time-dependent discrete random variables. The objective of the problem is to find a path with a minimum possible travel cost realization from each node to a given destination node for each departure time.

Additional Notation and Assumptions

Let $C_{ij}(t)$ be a random variable denoting the travel cost on link (i,j) at link entry time t. Assume that $C_{ij}(t)$ has a finite number of realizations, and denote the PMF of $C_{ij}(t)$ for $t \in \mathcal{H}$ by

$$p_{C_{ij}(t)} = \left\{(\zeta_{ij}^x(t),\, g_{ij}^x(t)) \mid x \in \mathcal{X}_{ij}(t)\right\}, \tag{4.25}$$

where $\zeta_{ij}^x(t) \in \Re$, $\forall\, x \in \mathcal{X}_{ij}(t)$ and $\sum_{x \in \mathcal{X}_{ij}(t)} g_{ij}^x(t) = 1$. Note that unlike the link travel times whose values are assumed to be positive integers, the link travel costs do not have any restriction imposed on their values. For all $t > H-1$, we assume that $p_{C_{ij}(t)} = p_{C_{ij}(H-1)}$. It is also assumed that $C_{ij}(t)$ are independent of each other and of $T_{ij}(t)$.

Optimality Conditions

Optimality conditions for the all-to-one minimum possible travel cost paths problem can be derived by following a procedure similar to one used in Section 4.2.

Let $\overline{C}_{ij}^c(t)$ be the travel cost from node i to node d when one leaves node i at time t via link (i,j) and then takes path c from node j to node d. $\overline{C}_{ij}^c(t)$ can be expressed as the sum of two random variables.

$$\begin{aligned}\overline{C}_{ij}^{c}(t) &= \text{travel cost on link } (i,j) + \\ &\quad \text{travel cost on path } c \text{ from node } j \text{ to node } d \\ &= C_{ij}(t) + Y_j^c(t + T_{ij}(t)), \end{aligned} \tag{4.26}$$

where $Y_j^c(t+T_{ij}(t))$ is a random variable denoting the travel cost from node j to node d at departure time $t+T_{ij}(t)$.

Let us denote by $\xi_i(t)$ the minimum possible travel cost from node i to node d when one leaves node i at time t. $\xi_i(t)$ is obtained by

$$\xi_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \text{travel cost values of } \overline{C}_{ij}^{c}(t) \right\} \right\}. \tag{4.27}$$

Using the realizations of $T_{ij}(t)$ and $C_{ij}(t)$, we rewrite (4.27) as

$$\xi_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \min_{x \in \mathcal{X}_{ij}(t)} \left\{ \zeta_{ij}^{x}(t) + \text{travel cost values of } Y_j^c(t + \tau_{ij}^r(t)) \right\} \right\} \right\} \right\}. \tag{4.28}$$

Note that $\zeta_{ij}^{x}(t)$ do not depend upon c and $r \in \mathcal{R}_{ij}(t)$. Letting $z = \arg\min_{x \in \mathcal{X}_{ij}(t)} \left\{ \zeta_{ij}^{x}(t) \right\}$, we can rewrite (4.28) as follows:

$$\begin{aligned}\xi_i(t) &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \min_{x \in \mathcal{X}_{ij}(t)} \left\{ \zeta_{ij}^{x}(t) \right\} + \text{travel cost values of } Y_j^c(t + \tau_{ij}^r(t)) \right\} \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{c} \left\{ \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \zeta_{ij}^{z}(t) + \text{travel cost values of } Y_j^c(t + \tau_{ij}^r(t)) \right\} \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^{z}(t) + \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \min_{c} \left\{ \text{travel cost values of } Y_j^c(t + \tau_{ij}^r(t)) \right\} \right\} \right\}. \end{aligned} \tag{4.29}$$

Let $(y_j^{c^k}(t + \tau_{ij}^r(t)),\ g_j^{c^k}(t + \tau_{ij}^r(t)))$, $k = 1, 2, \ldots, k_j^c(t)$ be the realizations of random variable $Y_j^c(t+\tau_{ij}^r(t))$, where $y_j^{c^k}(t+\tau_{ij}^r(t))$ is the travel cost value of the k^{th} realization and $g_j^{c^k}(t + \tau_{ij}^r(t))$ is its probability. Then (4.29) is equal to

$$\begin{aligned}\xi_i(t) &= \min_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^{z}(t) + \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \min_{c} \left\{ \min_{k} \left\{ y_j^{c^k}(t + \tau_{ij}^r(t)) \right\} \right\} \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^{z}(t) + \min_{r \in \mathcal{R}_{ij}(t)} \left\{ \xi_j(t + \tau_{ij}^r(t)) \right\} \right\}, \end{aligned} \tag{4.30}$$

since $\xi_j(t + \tau_{ij}^r(t)) = \min_c \left\{ \min_k \left\{ y_j^{c^k}(t + \tau_{ij}^r(t)) \right\} \right\}$. Therefore we have the following opti-

mality conditions that $\xi_i(t)$, $\forall i \in \mathcal{N}$, $\forall t$ should satisfy:

$$\xi_i(t) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^z(t) + \min\limits_{r \in \mathcal{R}_{ij}(t)} \left\{ \xi_j(t + \tau_{ij}^r(t)) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ 0 & i = d,\ \forall t \in \mathcal{H}, \end{cases} \tag{4.31}$$

$$\begin{aligned} \xi_i(t) = \xi_i(H) &= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^z(H) + \min\limits_{r \in \mathcal{R}_{ij}(H)} \{ \xi_j(H) \} \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H} \\ 0 & i = d,\ \forall t \notin \mathcal{H} \end{cases} \\ &= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^z(H) + \xi_j(H) \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \end{aligned} \tag{4.32}$$

Let $\varphi_i(t)$ be the highest probability that $\xi_i(t)$ occurs among all minimum possible travel cost realizations from node i to node d at departure time t. It is obtained by

$$\begin{aligned} \varphi_i(t) &= \max_{j \in \mathcal{O}(i)} \left\{ \max_c \left\{ P[\overline{C}_{ij}^c(t) = \xi_i(t)] \right\} \right\} \\ &= \max_{j \in \mathcal{O}(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ P[\overline{C}_{ij}^c(t) = \xi_i(t) \text{ through } \tau_{ij}^q(t)] \right\} \right\} \right\}, \end{aligned} \tag{4.33}$$

where $\Omega_{ij}(t) = \{ r \mid r \in \mathcal{R}_{ij}(t),\ \zeta_{ij}^z(t) + \xi_j(t + \tau_{ij}^r(t)) = \xi_i(t) \}$.

Due to the independence assumptions on $C_{ij}(t)$ and $T_{ij}(t)$, $P[\overline{C}_{ij}^c(t) = \xi_i(t) \text{ through } \tau_{ij}^q(t)]$ is computed by

$$\begin{aligned} &P[\overline{C}_{ij}^c(t) = \xi_i(t) \text{ through } \tau_{ij}^q(t)] \\ &\quad = P[C_{ij}(t) = \zeta_{ij}^z(t),\ T_{ij}(t) = \tau_{ij}^q(t),\ Y_j^c(t + \tau_{ij}^q(t)) = \xi_j(t + \tau_{ij}^q(t))] \\ &\quad = P[C_{ij}(t) = \zeta_{ij}^z(t)] \times P[T_{ij}(t) = \tau_{ij}^q(t)] \times P[Y_j^c(t + \tau_{ij}^q(t)) = \xi_j(t + \tau_{ij}^q(t))] \\ &\quad = g_{ij}^z(t) \times p_{ij}^q(t) \times P[Y_j^c(t + \tau_{ij}^q(t)) = \xi_j(t + \tau_{ij}^q(t))]. \end{aligned} \tag{4.34}$$

Plugging (4.34) into (4.33), we obtain

$$\begin{aligned} \varphi_i(t) &= \max_{j \in \mathcal{O}(i)} \left\{ \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ g_{ij}^z(t) \times p_{ij}^q(t) \times P[Y_j^c(t + \tau_{ij}^q(t)) = \xi_j(t + \tau_{ij}^q(t))] \right\} \right\} \right\} \\ &= \max_{j \in \mathcal{O}(i)} \left\{ g_{ij}^z(t) \times \max_c \left\{ \max_{q \in \Omega_{ij}(t)} \left\{ p_{ij}^q(t) \times P[Y_j^c(t + \tau_{ij}^q(t)) = \xi_j(t + \tau_{ij}^q(t))] \right\} \right\} \right\}. \end{aligned} \tag{4.35}$$

Let $(y_j^{c^{k_s}}(t+\tau_{ij}^q(t)),\ g_j^{c^{k_s}}(t+\tau_{ij}^q(t)))$, $s=1,2,\ldots$ be the realizations of $Y_j^c(t+\tau_{ij}^q(t))$ such that $y_j^{c^{k_s}}(t+\tau_{ij}^q(t))=\xi_j(t+\tau_{ij}^q(t))$. Then we can rewrite (4.35) as follows:

$$\begin{aligned}
\varphi_i(t) &= \max_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(t)\times\max_c\left\{\max_{q\in\Omega_{ij}(t)}\left\{p_{ij}^q(t)\times\max_s\left\{P[Y_j^c(t+\tau_{ij}^q(t))=y_j^{c^{k_s}}(t+\tau_{ij}^q(t))]\right\}\right\}\right\}\right\}\\
&= \max_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(t)\times\max_c\left\{\max_{q\in\Omega_{ij}(t)}\left\{p_{ij}^q(t)\times\max_s\left\{g_j^{c^{k_s}}(t+\tau_{ij}^q(t))\right\}\right\}\right\}\right\}\\
&= \max_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(t)\times\max_{q\in\Omega_{ij}(t)}\left\{p_{ij}^q(t)\times\max_c\left\{\max_s\left\{g_j^{c^{k_s}}(t+\tau_{ij}^q(t))\right\}\right\}\right\}\right\}\\
&= \max_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(t)\times\max_{q\in\Omega_{ij}(t)}\left\{p_{ij}^q(t)\times\varphi_j(t+\tau_{ij}^q(t))\right\}\right\},
\end{aligned}\tag{4.36}$$

where $\varphi_j(t+\tau_{ij}^q(t))=\max_c\left\{\max_s\left\{g_j^{c^{k_s}}(t+\tau_{ij}^q(t))\right\}\right\}$ by the definition of $\varphi_j(t+\tau_{ij}^q(t))$. Therefore optimality conditions for $\varphi_i(t)$ are given by

$$\varphi_i(t)=\begin{cases}\max\limits_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(t)\times\max\limits_{q\in\Omega_{ij}(t)}\left\{p_{ij}^q(t)\times\varphi_j(t+\tau_{ij}^q(t))\right\}\right\} & \forall\, i\neq d,\ \forall\, t\in\mathcal{H},\\ 1 & i=d,\ \forall\, t\in\mathcal{H},\end{cases}\tag{4.37}$$

$$\begin{aligned}
\varphi_i(t)=\varphi_i(H) &= \begin{cases}\max\limits_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(H)\times\max\limits_{q\in\Omega_{ij}(H)}\left\{p_{ij}^q(H)\times\varphi_j(H)\right\}\right\} & \forall\, i\neq d,\ \forall\, t\notin\mathcal{H}\\ 1 & i=d,\ \forall\, t\notin\mathcal{H}\end{cases}\\
&= \begin{cases}\max\limits_{j\in\mathcal{O}(i)}\left\{g_{ij}^z(H)\times\max\limits_{q\in\Omega_{ij}(H)}\left\{p_{ij}^q(H)\right\}\times\varphi_j(H)\right\} & \forall\, i\neq d,\ \forall\, t\notin\mathcal{H},\\ 1 & i=d,\ \forall\, t\notin\mathcal{H}.\end{cases}
\end{aligned}\tag{4.38}$$

Similar to the all-to-one minimum possible travel time paths problem, label pairs $(\xi_i(t),\ \varphi_i(t))$ satisfying the optimality conditions (4.31), (4.32), (4.37), and (4.38) can be determined in a decreasing order of departure time in a single pass. A proof of this is similar to one given in Proposition 1, so it is omitted.

Proposition 4. *All label pairs $(\xi_i(t),\ \varphi_i(t))$ satisfying the optimality conditions (4.31), (4.32), (4.37), and (4.38) can be determined in a decreasing order of departure time in a single pass.*

From (4.32), we can see that $\xi_i(H)$ can be determined by solving an all-to-one minimum cost (cheapest) paths problem in a deterministic static network where the link travel costs are set to $\zeta_{ij}^z(H)$. Once $\xi_i(H)$ are determined, $\varphi_i(H)$ are computed using (4.38). Note that

if link (i,j) belongs to the minimum cost path from node i to node d, $\Omega_{ij}(H) = \Re_{ij}(H)$ because $\zeta_{ij}^z(H) + \xi_j(H)$ is equal to $\xi_i(H)$ no matter what value $T_{ij}(H)$ takes. On the other hand, if link (i,j) is not on the minimum cost path, then $\Omega_{ij}(H) = \emptyset$. Hence for $i \neq d$, we can rewrite (4.38) as

$$\begin{aligned}\varphi_i(H) &= \max_{j \in \mathcal{O}(i)} \left\{ g_{ij}^z(H) \times \left(\delta_{ij} \times p_{ij}^u(H) \right) \times \varphi_j(H) \right\} \\ &= g_{iw}^z(H) \times p_{iw}^u(H) \times \varphi_w(H), \end{aligned} \tag{4.39}$$

where $u = \arg\max_{r \in \Re_{ij}(H)} \{p_{ij}^r(H)\}$, $\delta_{ij} = 1$ if (i,j) is on the minimum cost path and 0 otherwise, and w is the successor of node i on the minimum cost path. (4.39) may also be expressed as

$$\varphi_i(H) = \prod_{(v,w) \in P_{i \rightsquigarrow d}} \left(g_{vw}^z(H) \times p_{vw}^u(H) \right), \tag{4.40}$$

where $P_{i \rightsquigarrow d}$ is the minimum cost path from node i to node d.

Solution Algorithm

Algorithm MPTCP shown in Algorithm 4 is an extension of Algorithm MPTTP to the all-to-one minimum possible travel cost paths problem. Notice that we determine and save $z_{ij}(t)$ (denoted by z above), the index of the link travel cost realization with the smallest value for each link for each departure time, in the initialization step because they will not change in the remainder of the algorithm. It is possible to determine $z_{ij}(t)$ on the fly in Step 3 in order to save memory. For this, we may extend Algorithm MPTTP2 instead.

Proposition 5. *Let $\widetilde{c}$ denote the total number of link travel cost realizations, i.e. $\widetilde{c} = \sum_{(i,j) \in \mathcal{A}} \sum_{t \in \mathcal{H}} |\mathcal{X}_{ij}(t)|$. Then the worst-case running time complexity of Algorithm MPTCP is $\Theta(\max(\widetilde{c}, f(n,m), \widetilde{m}))$ and it is optimal.*

Proof. It takes $\Theta(nH + \widetilde{c})$ time to perform Step 1. The running time for Step 3 is the same as that of Algorithm MPTTP, which is $\Theta(\widetilde{m})$. Therefore the running time complexity of Algorithm MPTCP is $\Theta(nH + \widetilde{c} + f(n,m) + \widetilde{m}) = \Theta(\max(\widetilde{c}, f(n,m), \widetilde{m}))$. To solve the all-to-one minimum possible cost paths problem, we should determine $(\xi_i(H), \varphi_i(H))$ and examine all link travel cost realizations and all link travel time realizations at least once.

Algorithm 4 Algorithm MPTCP

Step 1: Initialization

$(\xi_i(t), \varphi_i(t), s_i(t), \pi_i(t)) \leftarrow (\infty, 0, \infty, \infty), \quad \forall\, i \neq d,\ \forall\, t \in \mathcal{H}$

$(\xi_d(t), \varphi_d(t), s_d(t), \pi_d(t)) \leftarrow (0, 1, d, t), \quad \forall\, t \in \mathcal{H}$

$z_{ij}(t) \leftarrow \arg\min_{x \in \mathcal{X}_{ij}(t)} \left\{\zeta_{ij}^{x}(t)\right\}, \quad \forall\, (i,j) \in \mathcal{A},\ \forall\, t \in \mathcal{H}$

Step 2: Minimum Possible Travel Cost Paths in Static Domain

$(\xi_i(H), \varphi_i(H), s_i(H), \pi_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{(\zeta_{ij}^{z}(H),\ g_{ij}^{z}(H))\}, \{p_{ij}^{u}(H)\}, d)$

Step 3: Minimum Possible Travel Cost Paths in Time-Dependent Domain

For $t \leftarrow H-1$ down to 0 **do**

  For $i \in \mathcal{N} \setminus \{d\}$ **do**

    $\xi_i(t) \leftarrow \min_{j \in \mathcal{O}(i)} \left\{\zeta_{ij}^{z_{ij}(t)}(t) + \min_{r \in \mathcal{R}_{ij}(t)} \left\{\xi_j(t + \tau_{ij}^{r}(t))\right\}\right\}$

    For $j \in \mathcal{O}(i)$ **do**

      $\Omega_{ij}(t) \leftarrow \{r \mid r \in \mathcal{R}_{ij}(t),\ \zeta_{ij}^{z_{ij}(t)}(t) + \xi_j(t + \tau_{ij}^{r}(t)) = \xi_i(t)\}$

    End (For)

    $\varphi_i(t) \leftarrow \max_{j \in \mathcal{O}(i)} \left\{g_{ij}^{z_{ij}(t)}(t) \times \max_{q \in \Omega_{ij}(t)} \left\{p_{ij}^{q}(t) \times \varphi_j(t + \tau_{ij}^{q}(t))\right\}\right\}$

    $s_i(t) \leftarrow \arg\max_{j \in \mathcal{O}(i)} \left\{g_{ij}^{z_{ij}(t)}(t) \times \max_{q \in \Omega_{ij}(t)} \left\{p_{ij}^{q}(t) \times \varphi_j(t + \tau_{ij}^{q}(t))\right\}\right\}$

    $k \leftarrow \arg\max_{q \in \Omega_{is_i(t)}(t)} \left\{p_{is_i(t)}^{q}(t) \times \varphi_{s_i(t)}(t + \tau_{is_i(t)}^{q}(t))\right\}$

    $\pi_i(t) \leftarrow t + \tau_{is_i(t)}^{k}(t)$

  End (For)

End (For)

This implies that $\Omega(f(n,m)+\widetilde{c}+\widetilde{m}) = \Omega(\max(f(n,m),\widetilde{c},\widetilde{m}))$ is the lowest possible running time to solve the problem. Since Algorithm MPTCP attains this lowest possible running time, its running time complexity is optimal. □

4.7.2 All-to-One $k-$Minimum Path Travel Time Realizations Problem

Selecting several paths between an origin node and a destination node is frequently required in many routing problems for various reasons. One reason could be that travelers have different path selection criteria such as travel time, travel cost, familiarity, scenery, etc. and sometimes they choose paths based on a combination of these. Since an optimal path in terms of one criterion may not be optimal for other criteria, finding a single path would not be sufficient in this case.

In stochastic time-dependent networks, travelers might consider not only the minimum travel time on a path but also the variance of travel time on a path as a path selection criterion. In this case, a path whose minimum travel time is slightly greater than the minimum possible travel time, but whose travel time variance is very small, might be preferred to a minimum possible travel time path with a very large travel time variance.

We may solve such a routing problem using a solution algorithm that simultaneously considers the minimum travel time and the variance of travel time with equal weight. However, if the minimum travel time is a primary criterion for choosing a path and the variance of travel time is a secondary criterion, it might be adequate that we first find several paths based on their minimum travel times and then select one of them according to a certain trade-off between the minimum travel time and the variance of travel time. This method can also serve as a proxy for a solution algorithm that takes into account both the minimum travel time and the variance of travel time simultaneously, when such an algorithm is difficult to develop.

In this section, we discuss how to find k minimum path travel time realizations efficiently from each node to a given destination node for each departure time in stochastic time-dependent networks. It should be emphasized that k minimum path travel time realizations are not necessarily associated with k different paths. In an extreme case, a single path may contain all k minimum path travel time realizations. However, for a sufficiently large value of k, we may expect that k minimum path travel time realizations produce several distinct paths. Note that the all-to-one $k-$minimum path travel time realizations problem is a direct

extension of the all-to-one minimum possible travel time paths problem.

Optimality Conditions

Let us denote by $\lambda_i^n(t)$ and $\gamma_i^n(t)$ the travel time value and the probability of the n^{th} minimum path travel time realization from node i to the destination node d at departure time t, respectively. Let $\min_{i\in\mathcal{S}}^n\{x_i\}$ be the operator that returns the n^{th} smallest element from the set $\{x_i \mid i \in \mathcal{S}\}$, where $\mathcal{S}$ is some set. We can obtain $\lambda_i^n(t)$ by

$$\begin{aligned}\lambda_i^n(t) &= \min_{j\in\mathcal{O}(i),\,c}^n \left\{\text{travel time values of } \overline{T}_{ij}^c(t)\right\} \\ &= \min_{j\in\mathcal{O}(i),\,c,\,r\in\mathcal{R}_{ij}(t)}^n \left\{\tau_{ij}^r(t) + \text{travel time values of } L_j^c(t+\tau_{ij}^r(t))\right\} \\ &= \min_{j\in\mathcal{O}(i),\,c,\,r\in\mathcal{R}_{ij}(t),\,m}^n \left\{\tau_{ij}^r(t) + l_j^{c^m}(t+\tau_{ij}^r(t))\right\}. \end{aligned} \tag{4.41}$$

Note that when we compute $\lambda_i^n(t),\ n = 1, 2, \ldots, k$ by (4.41), only the k smallest values from the set $\{l_j^{c^m}(t+\tau_{ij}^r(t)) \mid \forall c,\ \forall m\}$ need to be considered for a given $j \in \mathcal{O}(i)$ and $r \in \mathcal{R}_{ij}(t)$. This can be easily proved by contradiction. These k smallest values can be denoted by $\lambda_j^h(t+\tau_{ij}^r(t)),\ h = 1, 2, \ldots, k$ because

$$\lambda_j^h(t+\tau_{ij}^r(t)) = \min^h\{l_j^{c^m}(t+\tau_{ij}^r(t)) \mid \forall c,\ \forall m\},\quad h = 1, 2, \ldots, k,$$

by the definition of $\lambda_j^h(t+\tau_{ij}^r(t))$. Hence (4.41) is equivalent to the following formula:

$$\lambda_i^n(t) = \min_{j\in\mathcal{O}(i),\,r\in\mathcal{R}_{ij}(t),\,h\in\{1,2,\ldots,k\}}^n \left\{\tau_{ij}^r(t) + \lambda_j^h(t+\tau_{ij}^r(t))\right\}. \tag{4.42}$$

Letting $\mathcal{K} = \{1, 2 \ldots, k\}$, optimality conditions that $\lambda_i^n(t)$ should satisfy are thus given by

$$\lambda_i^n(t) = \begin{cases} \min_{j\in\mathcal{O}(i),\,r\in\mathcal{R}_{ij}(t),\,h\in\mathcal{K}}^n \left\{\tau_{ij}^r(t) + \lambda_j^h(t+\tau_{ij}^r(t))\right\} & \forall i \neq d,\ \forall n \in \mathcal{K},\ \forall t \in \mathcal{H}, \\ 0 & i = d,\ \forall n \in \mathcal{K},\ \forall t \in \mathcal{H}, \end{cases} \tag{4.43}$$

$$\lambda_i^n(t) = \lambda_i^n(H) = \begin{cases} \min_{j\in\mathcal{O}(i),\,r\in\mathcal{R}_{ij}(H),\,h\in\mathcal{K}}^n \left\{\tau_{ij}^r(H) + \lambda_j^h(H)\right\} & \forall i \neq d,\ \forall n \in \mathcal{K},\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall n \in \mathcal{K},\ \forall t \notin \mathcal{H}. \end{cases} \tag{4.44}$$

Since all link travel time values are positive, $\lambda_i^n(t)$ can be determined in a decreasing

order of departure time. In addition, once $\lambda_i^n(t)$ are set, they do not change afterward. A proof of this argument is similar to that of Proposition 1.

It should be noted that (4.44) is not equivalent to the following functional form, where $z = \arg\min_{r \in \mathcal{R}_{ij}(H)} \{\tau_{ij}^r(t)\}$:

$$\lambda_i^n(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i),\, h \in \mathcal{K}}^{n} \left\{ \tau_{ij}^z(H) + \lambda_j^h(H) \right\} & \forall\, i \neq d,\ \forall\, n \in \mathcal{K}, \\ 0 & i = d,\ \forall\, n \in \mathcal{K}. \end{cases} \tag{4.45}$$

(4.45) is a set of optimality conditions for the all-to-one $k-$shortest paths problem in a deterministic static network, where the travel time on link (i,j) is set to $\tau_{ij}^z(H)$. Therefore we must consider all link travel time values at departure time H to determine $\lambda_i^n(H)$ by (4.44). This indicates that $\lambda_i^n(H)$ are obtained by solving an all-to-one $k-$minimum path travel time realizations problem in a stochastic static network, not by solving an all-to-one $k-$shortest paths problem in a deterministic static network.

Let us consider optimality conditions for $\gamma_i^n(t)$. It is possible that more than one path travel time realization from node i to node d at departure time t have the same travel time value, $\lambda_i^n(t)$. Define set $\mathcal{S}_i^n(t)$ as follows:

$$\mathcal{S}_i^n(t) = \left\{ (j,r,h) \mid j \in \mathcal{O}(i),\ r \in \mathcal{R}_{ij}(t),\ h \in \mathcal{K},\ \tau_{ij}^r(t) + \lambda_j^h(t + \tau_{ij}^r(t)) = \lambda_i^n(t) \right\}. \tag{4.46}$$

Suppose we have $\mathcal{S}_i^n(t) = \{(j_1,r_1,h_1),(j_2,r_2,h_2),\ldots,(j_q,r_q,h_q)\}$. If $q \leq (k-n+1)$, all q path travel time realizations obtained by the elements in $\mathcal{S}_i^n(t)$ are used as the n^{th}, the $(n+1)^{\text{th}}$, $\ldots$, and the $(n+q-1)^{\text{th}}$ minimum path travel time realizations from node i to node d at departure time t, respectively. In this case, the probabilities $\gamma_i^n(t)$, $\gamma_i^{n+1}(t)$, $\ldots$, $\gamma_i^{n+q-1}(t)$ are computed from each $(j,r,h) \in \mathcal{S}_i^n(t)$. For instance, $\gamma_i^n(t) = p_{ij_1}^{r_1}(t) \times \gamma_{j_1}^{h_1}(t + \tau_{ij_1}^{r_1}(t))$, $\gamma_i^{n+1}(t) = p_{ij_2}^{r_2}(t) \times \gamma_{j_2}^{h_2}(t + \tau_{ij_2}^{r_2}(t))$, etc. It is not necessary to make $\gamma_i^n(t) \geq \gamma_i^{n+1}(t) \geq \cdots \geq \gamma_i^{n+q-1}(t)$. All we make sure is that both $\lambda_i^n(t)$ and $\gamma_i^n(t)$ are associated with the same $(j,r,h) \in \mathcal{S}_i^n(t)$.

If $q > (k-n+1)$, we discard $(q-k+n-1)$ realizations among q path travel time realizations, whose probabilities are smaller than the probability of any of the remaining realizations to be used as the n^{th}, the $(n+1)^{\text{th}}$, $\ldots$, and the k^{th} minimum path travel time realizations.

It is notationally cumbersome to write down these optimality conditions for $\gamma_i^n(t)$ in

a functional form. We will incorporate the optimality conditions for $\gamma_i^n(t)$ into a solution algorithm in a more convenient way.

Solution Algorithm

We define additional notation to describe a solution algorithm. $s_i^n(t)$ and $\pi_i^n(t)$ respectively denote the successor of node i and the arrival time at node $s_i^c(t)$ on the n^{th} minimum path travel time realization from node i to node d at departure time t. $\kappa_i^n(t)$ indicates the rank of the path travel time realization from node $s_i^n(t)$ to node d at departure time $\pi_i^n(t)$, which one should follow in order to achieve the n^{th} minimum path travel time realization from node i to node d at departure time t.

A label-setting algorithm, named *Algorithm $k-MPTTR$*, is described in Algorithm 5. Notice that it does not implement the optimality conditions for $\lambda_i^n(t)$ as they are stated in (4.43). This is simply for the sake of coding convenience. The optimality conditions for $\gamma_i^n(t)$ are also implemented indirectly. In these respects, Algorithm $k-$MPTTR can be viewed as an extension of Algorithm MPTTP2 to the all-to-one $k-$minimum path travel time realizations problem.

Several remarks about the algorithm are as follows: $c_i(t)$ is a counter that counts the number of minimum path travel time realizations from node i to node d at departure time t during the execution of the algorithm. In Step 2, we solve the all-to-one $k-$minimum path travel time realizations problem in a stochastic static network. Line 19 and 20 select the worst path travel time realization among k minimum path travel time realizations found so far. Set $\mathcal{S}$ at Line 19 contains all $n \in \mathcal{K}$ whose $\lambda_i^n(t)$ are the maximum. Line 20 chooses an index w from $\mathcal{S}$ whose $\gamma_i^w(t)$ is the minimum. Finally, $\lambda_i^n(t)$ obtained from this algorithm do not necessarily satisfy $\lambda_i^1(t) \leq \lambda_i^2(t) \leq \cdots \leq \lambda_i^k(t)$. If this is required, it can be done by sorting $\lambda_i^n(t)$ after the termination of the algorithm.

Proposition 6. *Let $R = \max_{(i,j)\in\mathcal{A}} |\mathcal{R}_{ij}(H)|$, and let $O(g(n,m,R,k))$ be the running time to solve an all-to-one $k-$minimum path travel time realizations problem in a stochastic static network in Step 2. Then the worst-case running time complexity of Algorithm $k-MPTTR$ is $O(\max(g(n,m,R,k), k^2\widetilde{m}))$.*

Proof. We need $\Theta(nHk)$ time to complete Step 1. In Step 3, it is apparent that Line 19 is a bottleneck operation (we assume that the value of k is not so large that Line 19

Algorithm 5 Algorithm $k-$MPTTR

Step 1: Initialization

$(\lambda_i^n(t), \gamma_i^n(t), s_i^n(t), \pi_i^n(t), \kappa_i^n(t)) \leftarrow (\infty, 0, \infty, \infty, \infty), \quad \forall i \neq d, \ \forall n \in \mathcal{K}, \ \forall t \in \mathcal{H}$

$(\lambda_d^n(t), \gamma_d^n(t), s_d^n(t), \pi_d^n(t), \kappa_d^n(t)) \leftarrow (0, 1, d, t, 0), \quad \forall n \in \mathcal{K}, \ \forall t \in \mathcal{H}$

$c_i(t) \leftarrow 0, \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

Step 2: k Minimum Path Travel Time Realizations in Static Domain

$(\lambda_i^n(H), \gamma_i^n(H), s_i^n(H), \pi_i^n(H), \kappa_i^n(H)) \leftarrow$ All-to-One $k-$SMPTTR$(\mathcal{N}, \mathcal{A}, \{T_{ij}(H)\}, d)$

Step 3: k Minimum Path Travel Time Realizations in Time-Dependent Domain

For $t \leftarrow H-1$ down to 0 **do**

  For $(i,j) \in \mathcal{A}$ **do**

    For $r \in \mathcal{R}_{ij}(t)$ **do**

      For $h \in \mathcal{K}$ **do**

        $\mu \leftarrow \tau_{ij}^r(t) + \lambda_j^h(t + \tau_{ij}^r(t))$

        $\nu \leftarrow p_{ij}^r(t) \times \gamma_j^h(t + \tau_{ij}^r(t))$

        If $(c_i(t) < k)$ **then**

          $c_i(t) \leftarrow c_i(t) + 1$

          $\lambda_i^{c_i(t)}(t) \leftarrow \mu; \ \gamma_i^{c_i(t)}(t) \leftarrow \nu$

          $s_i^{c_i(t)}(t) \leftarrow j; \ \pi_i^{c_i(t)}(t) \leftarrow t + \tau_{ij}^r(t); \ \kappa_i^{c_i(t)}(t) \leftarrow h$

        Else

          $\mathcal{S} \leftarrow \arg\operatorname{set}\max_{n \in \mathcal{K}} \{\lambda_i^n(t)\}$

          $w \leftarrow \arg\min_{s \in \mathcal{S}} \{\gamma_i^s(t)\}$

          If $(\mu < \lambda_i^w(t)$ or $(\mu = \lambda_i^w(t)$ and $\nu > \gamma_i^w(t)))$ **then**

            $\lambda_i^w(t) \leftarrow \mu; \ \gamma_i^w(t) \leftarrow \nu$

            $s_i^w(t) \leftarrow j; \ \pi_i^w(t) \leftarrow t + \tau_{ij}^r(t); \ \kappa_i^w(t) \leftarrow h$

          End (If)

        End (If)

      End (For)

    End (For)

  End (For)

End (For)

is executed). Whenever it is visited, $\Theta(|\mathcal{K}|) = \Theta(k)$ running time is needed. Since it can be visited at most $|\mathcal{K}| \sum_{t\in\mathcal{H}} \sum_{(i,j)\in\mathcal{A}} |\mathcal{R}_{ij}(t)| = k\widetilde{m}$ times, the running time complexity of Step 3 is $O(k^2\widetilde{m})$. Therefore the running time complexity of Algorithm $k-$MPTTR is $O(nHk + g(n,m,R,k) + k^2\widetilde{m}) = O(\max(g(n,m,R,k), k^2\widetilde{m}))$. □

Miller-Hooks [16] and Miller-Hooks and Mahmassani [17] propose a label-correcting algorithm for this problem, whose worst-case running time is claimed to be $O(\max(g(n,m,R,k), kn^3H^2\max(R,k)))$, where $R = |\mathcal{R}_{ij}(t)|, \forall\,(i,j) \in \mathcal{A}, \forall\, t \in \mathcal{H}$. It is an extension of Algorithm LEAST described in Section 4.5.2. We refer the reader to Miller-Hooks [16] or Miller-Hooks and Mahmassani [17] for the details of the algorithm.

4.7.3 All-to-One $k-$Dynamic Shortest Paths Problem

The $k-$shortest paths problem in deterministic static networks has been extensively investigated and many research papers have been published. To the best of the author's knowledge, however, no research work on the $k-$shortest paths problem in deterministic time-dependent networks has been published. In this section, we introduce an efficient algorithm that solves the all-to-one $k-$shortest paths problem in deterministic time-dependent networks.

Let us ignore stochasticity from the all-to-one $k-$minimum path travel time realizations problem discussed in Section 4.7.2. It is easy to see that in that case, the problem simply reduces to the all-to-one $k-$shortest paths problem in deterministic time-dependent networks.

Optimality Conditions

Let $\lambda_i^n(t)$ denote the travel time on the n^{th} shortest path from node i to node d at departure time t. Optimality conditions for $\lambda_i^n(t), \forall\, i \in \mathcal{N}, \forall\, t$ are obtained by eliminating terms related to stochasticity from (4.43) and (4.44).

$$\lambda_i^n(t) = \begin{cases} \min\limits_{j\in\mathcal{O}(i),\, h\in\mathcal{K}}^{n} \left\{\tau_{ij}(t) + \lambda_j^h(t+\tau_{ij}(t))\right\} & \forall\, i \neq d,\ \forall\, n \in \mathcal{K},\ \forall\, t \in \mathcal{H}, \\ 0 & i = d,\ \forall\, n \in \mathcal{K},\ \forall\, t \in \mathcal{H}, \end{cases} \tag{4.47}$$

$$\lambda_i^n(t) = \lambda_i^n(H) = \begin{cases} \min\limits_{j\in\mathcal{O}(i),\, h\in\mathcal{K}}^{n} \left\{\tau_{ij}(H) + \lambda_j^h(H)\right\} & \forall\, i \neq d,\ \forall\, h \in \mathcal{K},\ \forall\, t \notin \mathcal{H}, \\ 0 & i = d,\ \forall\, h \in \mathcal{K},\ \forall\, t \notin \mathcal{H}. \end{cases} \tag{4.48}$$

Let $s_i^n(t)$ denote the successor of node i on the n^{th} shortest path from node i to node d at departure time t. Let $\pi_i^n(t)$ denote the arrival time at node $s_i^n(t)$, which is equal to $t+\tau_{is_i^n(t)}(t)$. Let $\kappa_i^n(t)$ indicates the rank of the path from node $s_i^n(t)$ to node d at departure time $\pi_i^n(t)$, which one should follow in order to achieve the n^{th} shortest path from node i to node d at departure time t. $s_i^c(t)$ and $\kappa_i^c(t)$ are obtained by the following functional forms:

$$(s_i^n(t), \kappa_i^n(t)) = \begin{cases} \arg\min^n_{(j,h)\in\mathcal{O}(i)\times\mathcal{K}} \left\{\tau_{ij}(t) + \lambda_j^h(t+\tau_{ij}(t))\right\} & \forall i \neq d,\ \forall d \in \mathcal{K},\ \forall t \in \mathcal{H}, \\ (d,0) & i = d,\ \forall n \in \mathcal{K},\ \forall t \in \mathcal{H}, \end{cases} \tag{4.49}$$

$$(s_i^n(t), \kappa_i^n(t)) = (s_i^n(H), \kappa_i^n(H)) = \begin{cases} \arg\min^n_{(j,h)\in\mathcal{O}(i)\times\mathcal{K}} \left\{\tau_{ij}(H) + \lambda_j^h(H)\right\} & \forall i \neq d,\ \forall n \in \mathcal{K},\ \forall t \notin \mathcal{H}, \\ (d,0) & i = d,\ \forall n \in \mathcal{K},\ \forall t \notin \mathcal{H}. \end{cases} \tag{4.50}$$

Solution Algorithm

A solution algorithm named *Algorithm k−DSP* is shown in Algorithm 6. For each n, the execution of Line 10 requires $|\mathcal{O}(i)| \times |\mathcal{K}| = k|\mathcal{O}(i)|$ comparison operations because we already know $\lambda_i^{n-1}(t)$ ($\lambda_i^0(t)$ is assumed to be $-\infty$). Hence the total number of comparison operations at Line 10 until the termination of the algorithm is bounded from above by k^2Hm as shown below:

$$\sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{N}\setminus\{d\}} k \times k|\mathcal{O}(i)| \leq k^2 \sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{N}} |\mathcal{O}(i)| = k^2Hm.$$

Obviously Line 10 is the computational bottleneck of the algorithm. Hence we have the following proposition about the worst-case running time complexity of Algorithm k−DSP.

Proposition 7. *Let $O(g(n,m,k))$ be the running time to solve an all-to-one $k-$shortest paths problem in a deterministic static network in Step 2. Then the worst-case running time complexity of Algorithm $k-$DSP is $O(\max(g(n,m,k), k^2Hm))$.*

4.8 Waiting at Nodes

So far, we have assumed that no waiting is allowed at all nodes in the network. In this

Algorithm 6 Algorithm $k-$DSP

Step 1: Initialization

$(\lambda_i^n(t), s_i^n(t), \pi_i^n(t), \kappa_i^n(t)) \leftarrow (\infty, \infty, \infty, \infty), \quad \forall\, i \neq d,\ \forall\, n \in \mathcal{K},\ \forall\, t \in \mathcal{H}$

$(\lambda_d^n(t), s_d^n(t), \pi_i^n(t), \kappa_d^n(t)) \leftarrow (0, d, t, 0), \quad \forall\, n \in \mathcal{K},\ \forall\, t \in \mathcal{H}$

Step 2: k Shortest Paths in Static Domain

$(\lambda_i^n(H), s_i^n(H), \pi_i^n(H), \kappa_i^n(H)) \leftarrow$ All-to-One $k-$SP$(\mathcal{N}, \mathcal{A}, \{\tau_{ij}(H)\}, d)$

Step 3: k Shortest Paths in Time-Dependent Domain

For $t \leftarrow H-1$ down to 0 **do**

  For $i \in \mathcal{N} \setminus \{d\}$ **do**

    For $n \leftarrow 1$ to k **do**

      $\lambda_i^n(t) \leftarrow \min^n{}_{j \in \mathcal{O}(i),\, h \in \mathcal{K}} \left\{ \tau_{ij}(t) + \lambda_j^h(t + \tau_{ij}(t)) \right\}$

      $(s_i^n(t), \kappa_i^n(t)) \leftarrow \arg\min^n{}_{(j,h) \in \mathcal{O}(i) \times \mathcal{K}} \left\{ \tau_{ij}(t) + \lambda_j^h(t + \tau_{ij}(t)) \right\}$

      $\pi_i^n(t) = t + \tau_{is_i^n(t)}(t)$

    End (For)

  End (For)

End (For)

section, we briefly revisit the routing problems discussed in the preceding sections for the case where waiting is allowed at all nodes. We assume that the length of waiting is unlimited and the cost of waiting is additive. We only provide the modified optimality conditions for each routing problem. Modifications of solution algorithms are easy, so they are not included here.

4.8.1 All-to-One Minimum Possible Travel Time Paths Problem

When waiting is allowed at all nodes, the optimality conditions for $\lambda_i(t)$ become as follows:

$$\lambda_i(t) = \begin{cases} \min\limits_{j\in\mathcal{O}(i)} \left\{ \min\limits_{t\le\hat{t}\le H-1} \left\{ \min\limits_{r\in\mathcal{R}_{ij}(\hat{t})} \{\hat{t}-t+\tau_{ij}^r(\hat{t})+\lambda_j(\hat{t}+\tau_{ij}^r(\hat{t}))\} \right\} \right\} & \forall i\neq d,\ \forall t\in\mathcal{H}, \\ 0 & i=d,\ \forall t\in\mathcal{H}, \end{cases} \tag{4.51}$$

$$\lambda_i(t) = \lambda_i(H) = \begin{cases} \min\limits_{j\in\mathcal{O}(i)} \{\tau_{ij}^z(H)+\lambda_j(H)\} & \forall i\neq d,\ \forall t\notin\mathcal{H}, \\ 0 & i=d,\ \forall t\notin\mathcal{H}. \end{cases} \tag{4.52}$$

Notice that (4.52) is identical to (4.16) because waiting at nodes after $t = H-1$ is not beneficial at all.

The definition of set $\Omega_{ij}(t)$ should be modified to take account of possible waiting at nodes. For $t\in\mathcal{H}$, we define $\Omega_{ij}(t)$ as follows:

$$\Omega_{ij}(t) = \{(\hat{t},r) \mid t\le\hat{t}\le H-1,\ r\in\mathcal{R}_{ij}(\hat{t}),\ \hat{t}-t+\tau_{ij}^r(\hat{t})+\lambda_j(\hat{t}+\tau_{ij}^r(\hat{t}))=\lambda_i(t)\}. \tag{4.53}$$

For $t > H-1$, the definition is the same as before.

$$\Omega_{ij}(t) = \Omega_{ij}(H) = \{r \mid r\in\mathcal{R}_{ij}(H),\ \tau_{ij}^r(H)+\lambda_j(H)=\lambda_i(H)\}. \tag{4.54}$$

Then optimality conditions for $\gamma_i(t)$ are given by

$$\gamma_i(t) = \begin{cases} \max\limits_{j\in\mathcal{O}(i)} \left\{ \max\limits_{(\hat{t},r)\in\Omega_{ij}(t)} \{p_{ij}^r(\hat{t})\times\gamma_j(\hat{t}+\tau_{ij}^r(\hat{t}))\} \right\} & \forall i\neq d,\ \forall t\in\mathcal{H}, \\ 1 & i=d,\ \forall t\in\mathcal{H}, \end{cases} \tag{4.55}$$

$$\gamma_i(t) = \gamma_i(H) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{r \in \Omega_{ij}(H)} \left\{ p_{ij}^r(H) \times \gamma_j(H) \right\} \right\} & \forall\, i \neq d,\ \forall\, t \notin \mathcal{H}, \\ 1 & i = d,\ \forall\, t \notin \mathcal{H}. \end{cases} \tag{4.56}$$

4.8.2 All-to-One Minimum Possible Travel Cost Paths Problem

When waiting is allowed at all nodes, the optimality conditions for $\xi_i(t)$ change as follows:

$$\xi_i(t) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \min\limits_{t \leq \hat{t} \leq H-1} \left\{ \zeta_{ij}^z(\hat{t}) + \min\limits_{r \in \mathcal{R}_{ij}(\hat{t})} \left\{ \xi_j(\hat{t} + \tau_{ij}^r(\hat{t})) \right\} \right\} \right\} & \forall\, i \neq d,\ \forall\, t \in \mathcal{H}, \\ 0 & i = d,\ \forall\, t \in \mathcal{H}, \end{cases} \tag{4.57}$$

$$\xi_i(t) = \xi_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \zeta_{ij}^z(H) + \xi_j(H) \right\} & \forall\, i \neq d,\ \forall\, t \notin \mathcal{H}, \\ 0 & i = d,\ \forall\, t \notin \mathcal{H}. \end{cases} \tag{4.58}$$

For $t \in \mathcal{H}$, we define $\Omega_{ij}(t)$ as

$$\Omega_{ij}(t) = \{(\hat{t}, r) \mid t \leq \hat{t} \leq H-1,\ r \in \mathcal{R}_{ij}(\hat{t}),\ \zeta_{ij}^z(\hat{t}) + \xi_j(\hat{t} + \tau_{ij}^r(\hat{t})) = \xi_i(t)\}. \tag{4.59}$$

For $t > H-1$, $\Omega_{ij}(t)$ is defined as before.

$$\Omega_{ij}(t) = \Omega_{ij}(H) = \{r \mid r \in \mathcal{R}_{ij}(H),\ \zeta_{ij}^z(H) + \xi_j(H) = \xi_i(H)\}. \tag{4.60}$$

Then optimality conditions for $\varphi_i(t)$ are given by the following functional forms:

$$\varphi_i(t) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ \max\limits_{(\hat{t}, r) \in \Omega_{ij}(t)} \left\{ g_{ij}^z(\hat{t}) \times p_{ij}^r(\hat{t}) \times \varphi_j(\hat{t} + \tau_{ij}^r(\hat{t})) \right\} \right\} & \forall\, i \neq d,\ \forall\, t \in \mathcal{H}, \\ 1 & i = d,\ \forall\, t \in \mathcal{H}, \end{cases} \tag{4.61}$$

$$\varphi_i(t) = \varphi_i(H) = \begin{cases} \max\limits_{j \in \mathcal{O}(i)} \left\{ g_{ij}^z(H) \times \max\limits_{r \in \Omega_{ij}(H)} \left\{ p_{ij}^r(H) \right\} \times \varphi_j(H) \right\} & \forall\, i \neq d,\ \forall\, t \notin \mathcal{H}, \\ 1 & i = d,\ \forall\, t \notin \mathcal{H}. \end{cases} \tag{4.62}$$

4.8.3 All-to-One k–Minimum Path Travel Time Realizations Problem

When waiting is allowed at all nodes, the optimality conditions for $\lambda_i^n(t)$ are modified as follows:

$$\lambda_i^n(t) =$$

$$\begin{cases} \min^n_{j\in\mathcal{O}(i),\, t\le\hat{t}\le H-1,\, r\in\mathcal{R}_{ij}(t),\, h\in\mathcal{K}} \left\{\hat{t}-t+\tau_{ij}^r(\hat{t})+\lambda_j^h(\hat{t}+\tau_{ij}^r(\hat{t}))\right\} & \forall i\neq d,\ \forall n\in\mathcal{K},\ \forall t\in\mathcal{H}, \\ 0 & i=d,\ \forall n\in\mathcal{K},\ \forall t\in\mathcal{H}, \end{cases} \tag{4.63}$$

$$\lambda_i^n(t)=\lambda_i^n(H)=\begin{cases} \min^n_{j\in\mathcal{O}(i),\, r\in\mathcal{R}_{ij}(H),\, h\in\mathcal{K}} \left\{\tau_{ij}^r(H)+\lambda_j^h(H)\right\} & \forall i\neq d,\ \forall n\in\mathcal{K},\ \forall t\notin\mathcal{H}, \\ 0 & i=d,\ \forall n\in\mathcal{K},\ \forall t\notin\mathcal{H}. \end{cases} \tag{4.64}$$

4.8.4 All-to-One $k-$Dynamic Shortest Paths Problem

Finally, the following functional forms are the modified optimality conditions for $\lambda_i^n(t)$ for the all-to-one $k-$dynamic shortest paths problem when waiting is allowed at all nodes:

$$\lambda_i^n(t)=\begin{cases} \min^n_{j\in\mathcal{O}(i),\, t\le\hat{t}\le H-1,\, h\in\mathcal{K}} \left\{\hat{t}-t+\tau_{ij}(\hat{t})+\lambda_j^h(\hat{t}+\tau_{ij}(\hat{t}))\right\} & \forall i\neq d,\ \forall n\in\mathcal{K},\ \forall t\in\mathcal{H}, \\ 0 & i=d,\ \forall n\in\mathcal{K},\ \forall t\in\mathcal{H}, \end{cases} \tag{4.65}$$

$$\lambda_i^n(t)=\lambda_i^n(H)=\begin{cases} \min^n_{j\in\mathcal{O}(i),\, h\in\mathcal{K}} \left\{\tau_{ij}(H)+\lambda_j^h(H)\right\} & \forall i\neq d,\ \forall n\in\mathcal{K},\ \forall t\notin\mathcal{H}, \\ 0 & i=d,\ \forall n\in\mathcal{K},\ \forall t\notin\mathcal{H}. \end{cases} \tag{4.66}$$

Chapter 5

Minimum Expected Travel Time Next-Arc Hyperpaths

When networks are stochastic, expected travel time would be one of the criteria most frequently used for route selection in many applications of a repetitive nature. In this and the next chapters, we study routing problems based on expected travel time in stochastic time-dependent networks.

In Section 5.1, we introduce two routing policies in stochastic time-dependent networks and define two expected travel time-based routing problems from these routing policies. We explain the meaning of the wait-and-see expected minimum travel time and distinguish it from the minimum expected travel time in Section 5.2. We study the all-to-one minimum expected travel time next-arc hyperpaths problem in Sections 5.3–5.7. In Section 5.8, we discuss extensions of the all-to-one minimum expected travel time next-arc hyperpaths problem.

5.1 Routing Policies and Expected Travel Time-based Routing Problems

In most routing problems based on expected travel time, the objective would be to find an optimal routing solution that leads to the minimum expected travel time. In stochastic time-dependent networks, this optimal routing solution may have different forms according to routing policy. Two routing policies have been considered in the literature.

Consider the case where a traveler is capable of obtaining and utilizing network information while he is traveling. In this case, upon arrival at every intermediate node, he can optimally select a link to traverse next based on the travel time information gathered so far. We refer to this routing policy as the *adaptive routing policy* because the traveler can change his path adaptively en route. We call the expected travel time-based routing problem with the adaptive routing policy the *minimum expected travel time next-arc hyperpath (METTH) problem*, because an optimal routing solution is generally a next-arc hyperpath corresponding to the traveler's choices made en route, not a simple path. This problem arises in the study of route guidance systems within the context of ATIS when travelers are equipped with in-vehicle communication devices. We discuss this problem in this chapter.

Consider the other case where a traveler is required to follow a simple path that is determined before he leaves the origin. It is not allowed for the traveler to deviate from the path while he is traveling. In this case, an optimal routing solution should be a simple path. We refer to this routing policy as the *non-adaptive routing policy*. Under the non-adaptive routing policy, our interest is to find an a priori simple path that results in the minimum expected travel time. We call this routing problem the *minimum expected travel time path (METTP) problem*. This problem will be discussed in Chapter 6.

5.1.1 An Illustrative Example and Observations

To illustrate how the two problems are different, consider the stochastic time-dependent network example depicted in Figure 5-1 and Table 5.1[1].

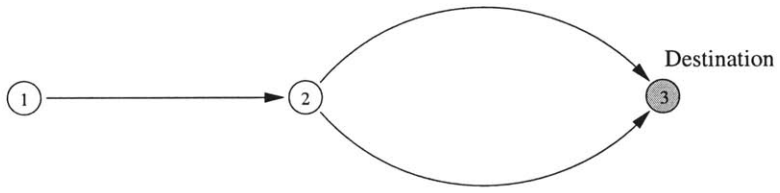

Figure 5-1: An Example Network

Let path a consist of links $(1,2)$ and $(2,3)_{\text{upper}}$ and let path b be comprised of links $(1,2)$ and $(2,3)_{\text{lower}}$. Suppose that a traveler makes a trip from node 1 to node 3 at time 0.

[1]To simplify the computation in this example, we allow parallel links between node 2 and node 3.

Table 5.1: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$	$(2,3)_{\text{upper}}$		$(2,3)_{\text{lower}}$	
Departure Time (t)	0	1	3	1	3
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (3, 0.5)	(1, 0.5) (3, 0.5)	(6, 0.5) (8, 0.5)	(6, 0.5) (8, 0.5)	(2, 0.5) (4, 0.5)

The expected travel time on path a for this trip is computed as follows:

$$\sum_{r=1}^{2}\left(\tau_{12}^r(0)+\sum_{q=1}^{2}\tau_{23_{\text{upper}}}^q(\tau_{12}^r(0))\times p_{23_{\text{upper}}}^q(\tau_{12}^r(0))\right)\times p_{12}^r(0)=$$

$$(1+(1\times 0.5+3\times 0.5))\times 0.5+(3+(6\times 0.5+8\times 0.5))\times 0.5=6.5.$$

Likewise, the expected travel time on path b for the trip is 7 units of time. Therefore path a is the simple path that results in the minimum expected travel time for the trip, i.e. path a is the optimal solution to the minimum expected travel time path problem for the trip. The minimum expected travel time from node 1 to node 3 at departure time 0 under the non-adaptive routing policy is 6.5 units of time.

Figure 5-2: Minimum Expected Travel Time Path

Now suppose that the traveler does not decide on which link to take from node 2 at the outset of the trip, but chooses either link $(2,3)_{\text{upper}}$ or link $(2,3)_{\text{lower}}$ after he arrives at node 2. If he arrives at node 2 at $t=1$, then he will select link $(2,3)_{\text{upper}}$ because at the entry time of 1, it has a lower expected travel time to the destination than link $(2,3)_{\text{lower}}$. On the other hand, if his arrival time at node 2 is $t=3$, then it is better to take link $(2,3)_{\text{lower}}$. The expected travel time of this strategy (solution) is

$$(1+(1\times 0.5+3\times 0.5))\times 0.5+(3+(2\times 0.5+4\times 0.5))\times 0.5=4.5.$$

When the adaptive routing policy is used, four routing strategies are available for the

trip as shown in Table 5.2. Since the routing strategy just described gives the minimum expected travel time, it is the optimal solution to the minimum expected travel time next-arc hyperpath problem for the trip. The hyperpath representing the optimal routing strategy is shown in Figure 5-3.

Table 5.2: Routing Strategies

Routing Strategy	Expected Travel Time
$t = 1$: take link $(2,3)_{\text{upper}}$ $t = 3$: take link $(2,3)_{\text{lower}}$	4.5
$t = 1$: take link $(2,3)_{\text{lower}}$ $t = 3$: take link $(2,3)_{\text{upper}}$	9.0
$t = 1$: take link $(2,3)_{\text{upper}}$ $t = 3$: take link $(2,3)_{\text{upper}}$	6.5
$t = 1$: take link $(2,3)_{\text{lower}}$ $t = 3$: take link $(2,3)_{\text{lower}}$	7.0

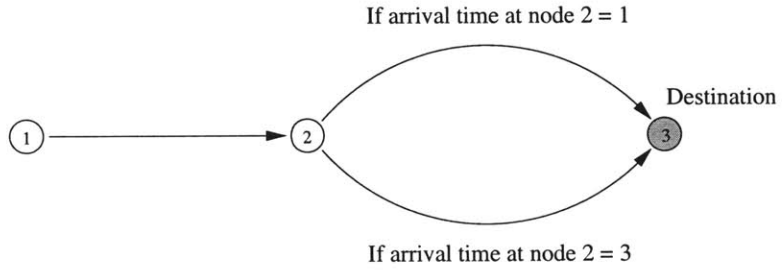

Figure 5-3: Minimum Expected Travel Time Next-Arc Hyperpath

Notice that the third and the fourth routing strategies in Table 5.2 are equivalent to taking path a and path b, respectively. Therefore taking any simple path is also a possible, but not optimal, routing solution when the traveler is allowed to make routing decisions adaptively en route in this example. The other two routing strategies in Table 5.2 do not correspond to simple paths. This implies that all feasible solutions to the minimum expected travel time path problem are also feasible to the minimum expected travel time next-arc hyperpath problem. The converse, however, is not true in general.

It follows from the above observation that for a given origin-destination pair and departure time, the minimum expected travel time obtained from the minimum expected travel time path problem is always greater than or equal to that obtained from the minimum

expected travel time next-arc hyperpath problem. Therefore the latter is a lower bound on the former.

In the minimum expected travel time path problem, i.e. under the non-adaptive routing policy, for a given origin-destination pair and departure time, all travelers follow the same simple path regardless of their actual arrival times at intermediate nodes. However in the minimum expected travel time next-arc hyperpath problem, i.e. under the adaptive routing policy, travelers may take different simple paths depending on their arrival times at intermediate nodes. The hyperpath is the set of those different simple paths.

If there is no time-dependency in the network, i.e. if the network is stochastic static, no matter when a traveler arrives at an intermediate node, the travel time distributions of the outgoing links from the node are the same. Therefore there is no incentive to defer making the decision on which link to take from the node until he arrives at the node. In this case, the minimum expected travel time next-arc hyperpath problem becomes identical to the minimum expected travel time path problem. As mentioned in Chapter 1 and will be shown mathematically in this chapter, if the network is stochastic static, a path with the minimum expected travel time can be found by setting each link travel time to its expected value and then applying a static shortest path algorithm.

5.1.2 Problem Type to Study

In this chapter, we study one type of the minimum expected travel time next-arc hyperpath problem, the *all-to-one minimum expected travel time next-arc hyperpaths problem* defined as follows: *Given a stochastic time-dependent network, from each node to a given destination node for each departure time, the problem is to find a next-arc hyperpath (a routing strategy), which results in the minimum expected travel time when travelers are allowed to change their paths adaptively en route.*

5.2 The Wait-and-See Expected Minimum Travel Time

In this section, we digress from the main topic of this chapter and explain the meaning of the wait-and-see expected minimum travel time. The reader may skip this section without loss of continuity.

Consider a trip that starts from node 1 at time 0 to the destination node 3 in the

following stochastic time-dependent network[2].

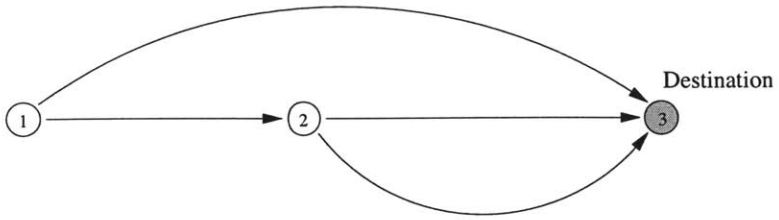

Figure 5-4: An Example Network

Table 5.3: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,3)$	$(1,2)$	$(2,3)_{\text{upper}}$		$(2,3)_{\text{lower}}$	
Departure Time (t)	0	0	2	4	2	4
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(6, 0.5) (8, 0.5)	(2, 0.5) (4, 0.5)	(1, 0.5) (3, 0.5)	(7, 0.5) (9, 0.5)	(6, 0.5) (8, 0.5)	(3, 0.5) (5, 0.5)

A *network state* is defined as a collection of link travel time values that may be experienced during the trip. For instance, the 5^{th} network state in Table 5.4 (s denotes network state number) corresponds to the following link travel time values:

$$\tau_{13}^1(0) = 6,\ \tau_{12}^2(0) = 4,\ \tau_{23_{\text{upper}}}^1(4) = 7,\ \tau_{23_{\text{lower}}}^1(4) = 3.$$

The probability of a network state is defined as the probability that the link travel time values associated with the network state are simultaneously realized. If we assume that the link travel time random variables are independent of each other, the probability of a network state is the product of the probabilities that the associated link travel time values occur. The probability of the 5^{th} network state is therefore given by

$$P_5 = p_{13}^1(0) \times p_{12}^2(0) \times p_{23_{\text{upper}}}^1(4) \times p_{23_{\text{lower}}}^1(4) = 0.0625.$$

For each network state, we can find the minimum travel time path for the trip. Notice that determining this is equivalent to computing a shortest path in a deterministic static network. The minimum travel time path for the 5^{th} network state is link $(1,3)$ that gives 6

[2]To simplify the computation in this example, we allow parallel links between node 2 and node 3.

units of time.

Table 5.4 shows all possible network states for a trip from node 1 to node 3 at departure time 0, their probabilities, the minimum travel time path and the minimum travel time for each network state.

Table 5.4: Network States and Minimum Travel Times

s	$\tau^r_{13}(0)$	$\tau^r_{12}(0)$	$\tau^r_{23_{\text{upper}}}(\cdot)$	$\tau^r_{23_{\text{lower}}}(\cdot)$	P_s	Min. Path	Min. Time
1	6	2	1	6	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	3
2	6	2	1	8	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	3
3	6	2	3	6	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	5
4	6	2	3	8	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	5
5	6	4	7	3	0.0625	$(1,3)$	6
6	6	4	7	5	0.0625	$(1,3)$	6
7	6	4	9	3	0.0625	$(1,3)$	6
8	6	4	9	5	0.0625	$(1,3)$	6
9	8	2	1	6	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	3
10	8	2	1	8	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	3
11	8	2	3	6	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	5
12	8	2	3	8	0.0625	$(1,2) \rightarrow (2,3)_{\text{upper}}$	5
13	8	4	7	3	0.0625	$(1,2) \rightarrow (2,3)_{\text{lower}}$	7
14	8	4	7	5	0.0625	$(1,3)$	8
15	8	4	9	3	0.0625	$(1,2) \rightarrow (2,3)_{\text{lower}}$	7
16	8	4	9	5	0.0625	$(1,3)$	8

The *wait-and-see expected minimum travel time* for a trip from node i to the destination node at departure time t, denoted by $\underline{e}_i(t)$, is defined as

$$\underline{e}_i(t) = \sum_{\text{all states } s} P_s \times \text{minimum travel time for state } s. \tag{5.1}$$

In this example, the wait-and-see expected minimum travel time from node 1 to node 3 at departure time 0 is computed by

$$\underline{e}_1(0) = 3 \times 0.0625 + 3 \times 0.0625 + \cdots + 8 \times 0.0625 = 5.375.$$

Suppose we observe the network for many days and record the observed minimum travel time from node i to the destination node at departure time t every day. The wait-and-see expected minimum travel time is then also obtained by

$$\underline{e}_i(t) = \lim_{n\to\infty} \frac{\sum_{k=1}^{n} \text{observed minimum travel time of day } k}{n}. \tag{5.2}$$

Note that (5.1) and (5.2) are both the same as

$$\underline{e}_i(t) = E[\min_c L_i^c(t)]. \tag{5.3}$$

It should be noted that the wait-and-see expected minimum travel time is generally associated with neither a simple path nor a hyperpath. In other words, it is not a travel time that can be achieved by taking a simple path or a hyperpath in general. Let us call the problem of computing the wait-and-see expected minimum travel time the *wait-and-see expected minimum travel time problem.*

For the example network in Figure 5-4 and Table 5.3, we can verify that the minimum expected travel time path for a trip from node 1 to node 3 at departure time 0 is link $(1, 3)$, whose expected travel time is 7 units of time. The minimum expected travel time next-arc hyperpath for this trip is shown in Figure 5-5 and its expected travel time is 6 units of time.

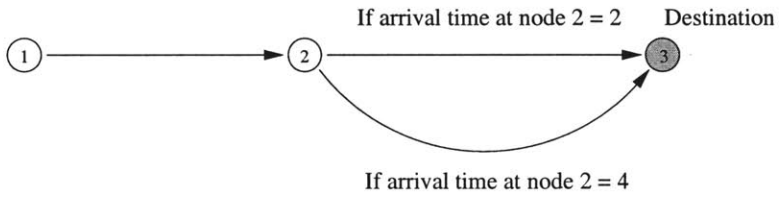

Figure 5-5: Minimum Expected Travel Time Next-Arc Hyperpath

For this trip, the wait-and-see expected minimum travel time is less than the expected travel time of the minimum expected travel time next-arc hyperpath ($5.375 < 6$). This is a general result, and we provide a rough justification for this below.

In the minimum expected travel time next-arc hyperpath problem, there is a constraint such that a solution should be a next-arc hyperpath. The wait-and-see expected minimum travel time problem, however, does not have such a constraint. Hence the latter problem is less constrained than the former problem. This implies that the wait-and-see expected minimum travel time for a given trip is a lower bound on the expected travel time of the minimum expected travel time next-arc hyperpath for the same trip.

In the previous section, we have seen that the expected travel time of the minimum

expected travel time next-arc hyperpath is a lower bound on that of the minimum expected travel time path. This is also justifiable because the minimum expected travel time path problem has a more restrictive constraint, which is a solution should be a simple path, than the minimum expected travel time next-arc hyperpath problem.

5.3 Optimality Conditions

In this section, we drive optimality conditions for the all-to-one minimum expected travel time next-arc hyperpaths problem.

Let $\overline{T}_{ij}(t)$ denote the travel time from node i to the destination node d when a traveler leaves node i at time t via link (i,j) and then takes an optimal next-arc hyperpath from node j to node d. $\overline{T}_{ij}(t)$ is written as the sum of two random variables,

$$\begin{aligned}\overline{T}_{ij}(t) &= \text{traversal time on link } (i,j) + \\ &\quad \text{travel time on an optimal next-arc hyperpath from node } j \text{ to node } d \\ &= T_{ij}(t) + L_j(t + T_{ij}(t)), \end{aligned} \tag{5.4}$$

where $L_j(t + T_{ij}(t))$ is a random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d at departure time $t + T_{ij}(t)$.

The expected value of $\overline{T}_{ij}(t)$ is interpreted as the expected travel time to node d when one leaves node i through link (i,j) at time t and selects all downstream links optimally according to his arrival times at intermediate nodes.

$$E[\overline{T}_{ij}(t)] = E[T_{ij}(t) + L_j(t + T_{ij}(t))] = E[T_{ij}(t)] + E[L_j(t + T_{ij}(t))]. \tag{5.5}$$

Let $e_i(t)$ denote the minimum expected travel time to node d under the adaptive routing policy when one leaves node i at time t. In other words, it is the expected travel time on the minimum expected travel time next-arc hyperpath from node i to node d at departure time t. $e_i(t)$ is obtained by

$$e_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ E[\overline{T}_{ij}(t)] \right\} = \min_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(t)] + E[L_j(t + T_{ij}(t))] \right\}. \tag{5.6}$$

$E[T_{ij}(t)] = \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t)$. By the total expectation theorem, $E[L_j(t + T_{ij}(t))]$ is

computed by

$$E[L_j(t+T_{ij}(t))] = E[E[L_j(t+T_{ij}(t)) \mid T_{ij}(t)]] = \sum_{r\in\mathcal{R}_{ij}(t)} E[L_j(t+\tau_{ij}^r(t))]\, p_{ij}^r(t). \quad (5.7)$$

Therefore we can rewrite (5.6) using the realizations of $T_{ij}(t)$ as follows:

$$e_i(t) = \min_{j\in\mathcal{O}(i)} \left\{ \sum_{r\in\mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \sum_{r\in\mathcal{R}_{ij}(t)} E[L_j(t+\tau_{ij}^r(t))]\, p_{ij}^r(t) \right\}. \quad (5.8)$$

By the definitions of $L_j(t+\tau_{ij}^r(t))$ and $e_j(t+\tau_{ij}^r(t))$, we have $e_j(t+\tau_{ij}^r(t)) = E[L_j(t+\tau_{ij}^r(t))]$. Hence (5.8) becomes

$$\begin{aligned} e_i(t) &= \min_{j\in\mathcal{O}(i)} \left\{ \sum_{r\in\mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \sum_{r\in\mathcal{R}_{ij}(t)} e_j(t+\tau_{ij}^r(t))\, p_{ij}^r(t) \right\} \\ &= \min_{j\in\mathcal{O}(i)} \left\{ \sum_{r\in\mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + e_j(t+\tau_{ij}^r(t))\right)\, p_{ij}^r(t) \right\}. \end{aligned} \quad (5.9)$$

Since $e_d(t) = 0$, optimality conditions for $e_i(t)$, $\forall\, i \in \mathcal{N}$, $\forall\, t \in \mathcal{H}$ are given by

$$e_i(t) = \begin{cases} \min\limits_{j\in\mathcal{O}(i)} \left\{ \sum\limits_{r\in\mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + e_j(t+\tau_{ij}^r(t))\right)\, p_{ij}^r(t) \right\} & \forall\, i \neq d,\ \forall\, t \in \mathcal{H}, \\ 0 & i = d,\ \forall\, t \in \mathcal{H}. \end{cases} \quad (5.10)$$

For $t > H-1$, (5.10) reduces to the following formula due to the no time-dependency in the network when $t > H-1$.

$$e_i(t) = e_i(H) = \begin{cases} \min\limits_{j\in\mathcal{O}(i)} \left\{ \sum\limits_{r\in\mathcal{R}_{ij}(H)} \left(\tau_{ij}^r(H) + e_j(H)\right)\, p_{ij}^r(H) \right\} & \forall\, i \neq d,\ \forall\, t \notin \mathcal{H}, \\ 0 & i = d,\ \forall\, t \notin \mathcal{H}. \end{cases} \quad (5.11)$$

Proposition 8. *All labels $e_i(t)$ satisfying the optimality conditions (5.10) and (5.11) can be determined in a decreasing order of departure time in a single pass.*

Proof. From (5.10), it is apparent that $e_i(t)$ is determined only by $e_j(s)$ such that $s > t$ because all link travel time values, $\tau_{ij}^r(t)$, are positive. Therefore if we first compute $e_i(H)$ as a base case by (5.11), we can compute $e_i(H-1), e_i(H-2), \cdots, e_i(0)$ by (5.10) in a decreasing order of departure time. In order to prove that $e_i(t)$ can be computed in a

single pass, we argue that once $e_i(t)$ are determined, they do not change. We prove this by contradiction. Suppose that we compute the value of $e_i(t)$ for some i, t and then modify it later. It means that some $e_j(s)$ for $j \in \mathcal{O}(i)$ and $s > t$ changes from its previous value (it decreases). This in turn indicates that some $e_k(q)$ for $k \in \mathcal{O}(j)$ and $q > s$ changes and so on, until we arrive at the static domain ($t > H - 1$). In the static domain, however, all $e_i(t)$, $i \in \mathcal{N}$, $t > H - 1$ do not change once they are set to $e_i(H)$ by (5.11). This means that all $e_i(t)$ at $t = H - 1$ do not change after they are initially determined by (5.10). We can repeat this argument in a decreasing order of departure time until we arrive at departure time t. Therefore $e_i(t)$ for some i, t does not change once it is determined, which contradicts the assumption. This concludes that all $e_i(t)$ can be determined in a decreasing order of departure time in a single pass. □

5.4 Minimum Expected Travel Time Next-Arc Hyperpaths in the Static Domain

Let us consider the computation of $e_i(H)$ by (5.11), that is, the computation of the all-to-one minimum expected travel time next-arc hyperpaths in the static domain. We can simplify (5.11) as follows:

$$
\begin{aligned}
e_i(H) &= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \sum\limits_{r \in \mathcal{R}_{ij}(H)} \tau_{ij}^r(H)\, p_{ij}^r(H) + e_j(H) \sum\limits_{r \in \mathcal{R}_{ij}(H)} p_{ij}^r(H) \right\} & \forall\, i \neq d \\ 0 & i = d \end{cases} \\
&= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(H)] + e_j(H) \right\} & \forall\, i \neq d\,, \\ 0 & i = d\,. \end{cases}
\end{aligned} \tag{5.12}
$$

Note that (5.12) is the set of optimality conditions for the all-to-one static shortest paths problem. This indicates that $e_i(H)$ can be found by solving an all-to-one shortest paths problem in a deterministic static network where the link travel times are set to their expected values at departure time H, $E[T_{ij}(H)]$.

Therefore the solutions to the all-to-one minimum expected travel time next-arc hyperpaths problem in the static domain are simple paths. This proves the argument we made in Section 5.1.1 that the minimum expected travel time next-arc hyperpath problem is equiv-

alent to the minimum expected travel time path problem, if there is no time-dependency in the network.

5.5 Hyperpaths Construction

In order to construct the minimum expected travel time next-arc hyperpaths, we maintain $s_i(t)$, the successor of node i on the minimum expected travel time next-arc hyperpath from node i to node d at departure time t. $s_i(t)$, $\forall i \in \mathcal{N}$, $\forall t$ are obtained by the following functional forms, where we define $s_d(t) = d$ for all t:

$$s_i(t) = \begin{cases} \arg\min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + e_j(t + \tau_{ij}^r(t))\right) p_{ij}^r(t) \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ d & i = d,\ \forall t \in \mathcal{H}, \end{cases} \tag{5.13}$$

$$s_i(t) = s_i(H) = \begin{cases} \arg\min_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(H)] + e_j(H) \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{5.14}$$

The minimum expected travel time next-arc hyperpath from node i to node d at departure time t is an acyclic subnetwork of the original network. Finding the links that constitute the subnetwork is rather involved compared to finding the links consituting an optimal simple path. We provide *Algorithm Hyperpath* to find the links of a minimum expected travel time next-arc hyperpath in Algorithm 7. The links are stored in set L when the algorithm is terminated.

5.6 Solution Algorithms

5.6.1 Algorithm METTH

The discussions in the preceding sections allow us to develop an efficient algorithm to solve the all-to-one minimum expected travel time next-arc hyperpaths problem. A label-setting algorithm called *Algorithm METTH* is described in Algorithm 8. In Appendix C, we give an example that illustrates how Algorithm METTH works.

Let us analyze the worst-case running time complexity of Algorithm METTH. It takes $\Theta(nH)$ time to complete Step 1. The total number of elementary operations performed

Algorithm 7 Algorithm Hyperpath

Create a set L

Create a queue S

Insert link $(i, s_i(t))$ into L

If $s_i(t) \neq d$ **then**

    For $r \in \mathcal{R}_{is_i(t)}(t)$ **do**

        Enqueue $(s_i(t), t + \tau^r_{is_i(t)}(t))$ into S

    End (For)

End (If)

While $S \neq \emptyset$ **do**

    Dequeue a (node, departure time) pair, called (i, t), from S

    If L does not contain link $(i, s_i(t))$ **then**

        Insert link $(i, s_i(t))$ into L

    End (If)

    If $s_i(t) \neq d$ **then**

        For $r \in \mathcal{R}_{is_i(t)}(t)$ **do**

            Enqueue $(s_i(t), t + \tau^r_{is_i(t)}(t))$ into S

        End (For)

    End (If)

End (While)

Algorithm 8 Algorithm METTH

1: **Step 1: Initialization**

2: $(e_i(t), s_i(t)) \leftarrow (\infty, \infty), \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

3: $(e_d(t), s_d(t)) \leftarrow (0, d), \quad \forall t \in \mathcal{H}$

4: **Step 2: Minimum Expected Time Hyperpaths in Static Domain**

5: $(e_i(H), s_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{E[T_{ij}(H)]\}, d)$

6: **Step 3: Minimum Expected Time Hyperpaths in Time-Dependent Domain**

7: **For** $t \leftarrow H - 1$ down to 0 **do**

8: **For** $i \in \mathcal{N} \setminus \{d\}$ **do**

9: $e_i(t) \leftarrow \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + e_j(t + \tau_{ij}^r(t)) \right) p_{ij}^r(t) \right\}$

10: $s_i(t) \leftarrow \arg\min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + e_j(t + \tau_{ij}^r(t)) \right) p_{ij}^r(t) \right\}$

11: **End (For)**

12: **End (For)**

in Step 3 is given by $W = \Theta\big(\sum_{t\in\mathcal{H}}\sum_{i\in\mathcal{N}\setminus\{d\}}\sum_{j\in\mathcal{O}(i)}|\mathcal{R}_{ij}(t)|\big)$. Hence the running time of Step 3 is $\Theta(\widetilde{m})$. The total running time complexity of Algorithm METTH is therefore $\Theta(nH + f(n,m) + \widetilde{m}) = \Theta(\max(f(n,m),\widetilde{m}))$.

Proposition 9. *The worst-case running time complexity of Algorithm METTH is* $\Theta(\max(f(n,m),\widetilde{m}))$ *and it is optimal.*

Proof. We only need to show that the running time is optimal. Any solution algorithm for the all-to-one minimum expected travel time next-arc hyperpaths problem must examine all link travel time realizations at least once, because any unexamined link travel time realizations may lead to lower expected travel times. To do so, it requires $\Omega(\widetilde{m})$ time. Any solution algorithm should also compute $e_i(H)$ somehow, and we assume that the lowest possible time to compute $e_i(H)$ is $\Theta(f(n,m))$. Therefore $\Omega(\widetilde{m} + f(n,m)) = \Omega(\max(f(n,m),\widetilde{m}))$ is the lowest possible running time of any solution algorithm. The worst-case running time of Algorithm METTH attains this bound, so it is optimal. □

5.6.2 Algorithm ELB

The all-to-one minimum expected travel time next-arc hyperpaths problem was also studied in Miller-Hooks [16] and in Miller-Hooks and Mahmassani [18], and a modified label-correcting algorithm called *Algorithm ELB* was proposed as a solution algorithm. We restate Algorithm ELB in Algorithm 9 for the reader's reference. Note that the original version of Algorithm ELB does not have Step 2.

When $R = \max_{(i,j)\in\mathcal{A},\, t\in\mathcal{H}}\{|\mathcal{R}_{ij}(t)|\}$, the worst-case running time complexity of Algorithm ELB is $O(\max(f(n,m), n^3H^2R))$ (Miller-Hook and Mahmassani [18]). This running time is clearly inferior to the worst-case running time of Algorithm METTH theoretically because $\max(f(n,m), n^3H^2R) \geq \max(f(n,m),\widetilde{m})$. The practical running time of Algorithm ELB will be compared with that of Algorithm METTH in Chapter 7.

Like Algorithm LEAST for the all-to-one minimum possible travel time paths problem, Algorithm ELB does not exploit the fact that the labels can be set in a decreasing order of departure time in a single pass.

5.7 Waiting at Nodes

If waiting is allowed at all nodes in the network, a traveler may shorten the expected travel

Algorithm 9 Algorithm ELB

Step 1: Initialization

$(e_i(t), s_i(t)) \leftarrow (\infty, \infty), \quad \forall\, i \neq d,\ \forall\, t \in \mathcal{H}$

$(e_d(t), s_d(t)) \leftarrow (0, d), \quad \forall\, t \in \mathcal{H}$

Create a queue S

Enqueue d into S

Step 2: Minimum Expected Time Hyperpaths in Static Domain

$(e_i(H), s_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{E[T_{ij}(H)]\}, d)$

Step 3: Choose Current Node

If $S = \emptyset$ **then** stop

Else dequeue a node (called j) from S

Step 4: Update Labels

For $i \in \mathcal{I}(j)$ **do**

For $t \leftarrow 0$ to $H - 1$ **do**

$\eta \leftarrow \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau^r_{ij}(t) + e_j(t + \tau^r_{ij}(t))\right) p^r_{ij}(t)$

If $\eta < e_i(t)$ **then**

$e_i(t) \leftarrow \eta$

$s_i(t) \leftarrow j$

If $i \notin S$ **then**

Enqueue i into S

End (If)

End (If)

End (For)

End (For)

Go to **Step 3**

time by waiting at some appropriate nodes during his trip. Below we provide optimality conditions for $e_i(t)$ when waiting is allowed at all nodes.

$$e_i(t) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \min\limits_{t \le \hat{t} \le H-1} \left\{ \hat{t} - t + \sum\limits_{r \in \mathcal{R}_{ij}(\hat{t})} \left(\tau_{ij}^r(\hat{t}) + e_j(\hat{t} + \tau_{ij}^r(\hat{t})) \right) p_{ij}^r(\hat{t}) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ 0 & i = d,\ \forall t \in \mathcal{H}, \end{cases} \tag{5.15}$$

$$e_i(t) = e_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \sum\limits_{r \in \mathcal{R}_{ij}(H)} \left(\tau_{ij}^r(H) + e_j(H) \right) p_{ij}^r(H) \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{5.16}$$

Algorithm METTH can be modified to take account of the optimality conditions (5.15) and (5.16). The worst-case running time complexity of the resulting algorithm is $O(\max(f(n,m), H\widetilde{m}))$.

5.8 Extensions

In this section, we extend the all-to-one minimum expected travel time next-arc hyperpaths problem to: (1) when travel cost is of primary concern for route selection, (2) when signals are present in the network, and (3) when several travel modes are available in the network.

5.8.1 Minimum Expected Travel Cost Next-Arc Hyperpaths

Algorithm METTH can be extended for determining all-to-one minimum expected travel cost next-arc hyperpaths in networks where the link travel times and the link travel costs are both time-dependent random variables.

Additional Notation and Assumptions

Recollect the random variable $C_{ij}(t)$ denoting the travel cost on link (i,j) when one enters the link at time t, which we defined in Section 4.7.1. In the all-to-one minimum expected travel cost next-arc hyperpaths problem, however, $C_{ij}(t)$ does not have to be a discrete random variable.

Let a function $f_{C_{ij}(t)}(c) : \Re \longmapsto \Re_+$ such that $\int_c f_{C_{ij}(t)}(c)\, dc = 1$ be the probability

density function (PDF) of $C_{ij}(t)$ for $t \in \mathcal{H}$. For $t > H-1$, it is assumed that $f_{C_{ij}(t)}(c) = f_{C_{ij}(H-1)}(c)$ for all $(i,j) \in \mathcal{A}$. We also assume that no waiting is allowed at all nodes.

Optimality Conditions

Let $\overline{C}_{ij}(t)$ denote the travel cost from node i to the destination node d when one leaves node i at time t via link (i,j) and then takes an optimal next-arc hyperpath from node j to node d. $\overline{C}_{ij}(t)$ is expressed by

$$\begin{aligned} \overline{C}_{ij}(t) &= \text{travel cost over link } (i,j) + \\ &\quad \text{travel cost on an optimal next-arc hyperpath from node } j \text{ to node } d \\ &= C_{ij}(t) + Y_j(t + T_{ij}(t)), \end{aligned} \tag{5.17}$$

where $Y_j(t + T_{ij}(t))$ is a random variable denoting the travel cost on an optimal next-arc hyperpath from node j to node d when one departs from node j at time $t + T_{ij}(t)$. The expected value of $\overline{C}_{ij}(t)$ is given by

$$E[\overline{C}_{ij}(t)] = E[C_{ij}(t) + Y_j(t + T_{ij}(t))] = E[C_{ij}(t)] + E[Y_j(t + T_{ij}(t))]. \tag{5.18}$$

Let $c_i(t)$ be the minimum expected travel cost from node i to node d under the adaptive routing policy when one departs from node i at time t. $c_i(t)$ is obtained by

$$c_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ E[\overline{C}_{ij}(t)] \right\} = \min_{j \in \mathcal{O}(i)} \left\{ E[C_{ij}(t)] + E[Y_j(t + T_{ij}(t))] \right\}. \tag{5.19}$$

Using the total expectation theorem,

$$c_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \int_c f_{C_{ij}(t)}(c)\, dc + \sum_{r \in \mathcal{R}_{ij}(t)} E[Y_j(t + \tau_{ij}^r(t))]\; p_{ij}^r(t) \right\}. \tag{5.20}$$

Since $c_j(t + \tau_{ij}^r(t)) = E[Y_j(t + \tau_{ij}^r(t))]$ by the definitions of $c_j(t + \tau_{ij}^r(t))$ and $Y_j(t + \tau_{ij}^r(t))$, optimality conditions for $c_i(t)$, $\forall\, i \in \mathcal{N}$, $\forall\, t \in \mathcal{H}$ are given by

$$c_i(t) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \int_c f_{C_{ij}(t)}(c)\, dc + \sum\limits_{r \in \mathcal{R}_{ij}(t)} c_j(t + \tau_{ij}^r(t))\; p_{ij}^r(t) \right\} & \forall\, i \neq d,\ \forall\, t \in \mathcal{H}, \\ 0 & i = d,\ \forall\, t \in \mathcal{H}. \end{cases} \tag{5.21}$$

For $t > H - 1$, (5.21) reduces to

$$c_i(t) = c_i(H) = \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \left\{ \int_c f_{C_{ij}(H)}(c)\, dc + \sum\limits_{r \in \mathcal{R}_{ij}(H)} c_j(H)\, p_{ij}^r(H) \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H} \\ 0 & i = d,\ \forall t \notin \mathcal{H} \end{cases}$$

$$= \begin{cases} \min\limits_{j \in \mathcal{O}(i)} \{E[C_{ij}(H)] + c_j(H)\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}, \end{cases} \tag{5.22}$$

which can be solved by an all-to-one static shortest paths algorithm.

It should be noted that the individual probabilities of link travel cost realizations are not used in the optimality conditions. We only need to know the expected link travel costs at discrete departure times, $E[C_{ij}(t)]$, to solve the all-to-one minimum expected travel cost next-arc hyperpaths problem. We, however, should know the PMFs of the link travel times to solve the problem.

Proposition 10. *All labels $c_i(t)$ satisfying the optimality conditions (5.21) and (5.22) can be determined in a decreasing order of departure time in a single pass.*

Proof. A proof is similar to the proof of Proposition 8. □

Solution Algorithm

A solution algorithm called *Algorithm METCH* for the all-to-one minimum expected travel cost next-arc hyperpaths problem is described in Algorithm 10.

Proposition 11. *Suppose we approximate $f_{C_{ij}(t)}(c)$ by the following PMF:*

$$p_{C_{ij}(t)} = \left\{ (\zeta_{ij}^x(t), g_{ij}^x(t)),\ x \in \mathcal{X}_{ij}(t) \right\},$$

where $\mathcal{X}_{ij}(t) = \{1, 2, \ldots, x_{ij}(t)\}$. Let $\widetilde{c} = \sum_{(i,j) \in \mathcal{A}} \sum_{t \in \mathcal{H}} |\mathcal{X}_{ij}(t)|$. The worst-case running time complexity of Algorithm METCH in this case is $\Theta(\max(f(n,m), \widetilde{c}, \widetilde{m}))$ and it is optimal.

Proof. It takes $\Theta(nH)$ time to complete Step 1. Since the approximation converts Line 9 into the following,

Algorithm 10 Algorithm METCH

1: **Step 1: Initialization**

2: $(c_i(t), s_i(t)) \leftarrow (\infty, \infty), \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

3: $(c_d(t), s_d(t)) \leftarrow (0, d), \quad \forall t \in \mathcal{H}$

4: **Step 2: Minimum Expected Cost Hyperpaths in Static Domain**

5: $(c_i(H), s_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{E[C_{ij}(H)]\}, d)$

6: **Step 3: Minimum Expected Cost Hyperpaths in Time-Dependent Domain**

7: **For** $t \leftarrow H - 1$ down to 0 **do**

8: **For** $i \in \mathcal{N} \setminus \{d\}$ **do**

9: $c_i(t) \leftarrow \min_{j \in \mathcal{O}(i)} \left\{ \int_c f_{C_{ij}(t)}(c)\, dc + \sum_{r \in \mathcal{R}_{ij}(t)} c_j(t + \tau_{ij}^r(t))\ p_{ij}^r(t) \right\}$

10: $s_i(t) \leftarrow \arg\min_{j \in \mathcal{O}(i)} \left\{ \int_c f_{C_{ij}(t)}(c)\, dc + \sum_{r \in \mathcal{R}_{ij}(t)} c_j(t + \tau_{ij}^r(t))\ p_{ij}^r(t) \right\}$

11: **End (For)**

12: **End (For)**

$$c_i(t) \leftarrow \min_{j \in \mathcal{O}(i)} \left\{ \sum_{x \in \mathcal{X}_{ij}(t)} \zeta_{ij}^x(t)\, g_{ij}^x(t) + \sum_{r \in \mathcal{R}_{ij}(t)} c_j(t + \tau_{ij}^r(t))\, p_{ij}^r(t) \right\},$$

the total number of elementary operations needed in Step 3 is given by $W = \Theta\big(\sum_{t \in \mathcal{H}} \sum_{i \in \mathcal{N} \setminus \{d\}} \sum_{j \in \mathcal{O}(i)} (|\mathcal{X}_{ij}(t)| + |\mathcal{R}_{ij}(t)|)\big)$. Hence the running time of Step 3 is $\Theta(\widetilde{c} + \widetilde{m})$. The worst-case running time complexity of Algorithm METCH is therefore $\Theta(nH + f(n,m) + \widetilde{c} + \widetilde{m}) = \Theta(\max(f(n,m), \widetilde{c}, \widetilde{m}))$. To solve the problem, we need to compute $c_i(H)$ and to examine all link travel cost realizations and all link travel time realizations at least once. Hence $\Omega(f(n,m) + \widetilde{c} + \widetilde{m}) = \Omega(\max(f(n,m), \widetilde{c}, \widetilde{m}))$ is the lowest possible running time for the problem. This indicates that the running time complexity of Algorithm METCH is optimal. □

5.8.2 Signalized Networks

If a network is signalized, waiting due to signal control at intersections often accounts for a large proportion of the total travel time, and therefore, cannot be ignored when routing decisions are made. In this section, we discuss the all-to-one minimum expected travel time next-arc hyperpaths problem when signals are present in the network.

Problem Overview, Additional Notation and Assumptions

A signal cycle at a signalized intersection consists of signal phases, each of which defines a set of movements allowed within its duration. Generally, a particular movement is only allowed within a particular phase of a signal cycle.

When a traveler arrives at a signalized intersection, if his desired movement is not permitted in the current signal phase, he should wait until the beginning of the next phase in which the movement is allowed. The time between the moment of his arrival at the intersection and the beginning of the next phase that permits his intended movement is called the *penalty* for the movement (Yang and Miller-Hooks [25]).

For a given movement, the value of the penalty obviously depends on the arrival time at the intersection. If the network is equipped with pre-timed signal control devices, the lengths and the sequence of the signal phases at each signalized intersection are fixed and known in advance. In this case, it is possible to compute the penalties for movements at each signalized intersection for each arrival time before making a trip.

For an example, Figure 5-6 illustrates a signalized intersection that has 12 possible movements. Table 5.5 shows the allowed movements and the duration of each of four phases at the intersection. Let us assume that the signal consists of only red and green time. Amber time is not considered. We also assume that there is no spillback effect from downstream intersections and that a driver does not have to wait for more than one signal cycle at the intersection. In other words, we assume that intersections are not saturated. Table 5.6 summarizes the penalties for the movements assuming that Phase I starts at time 0. Note that the penalties repeat every 10 units of time (the cycle length). Therefore, we do not have to compute the penalties for all arrival times. Computing the penalties for the first signal cycle suffices.

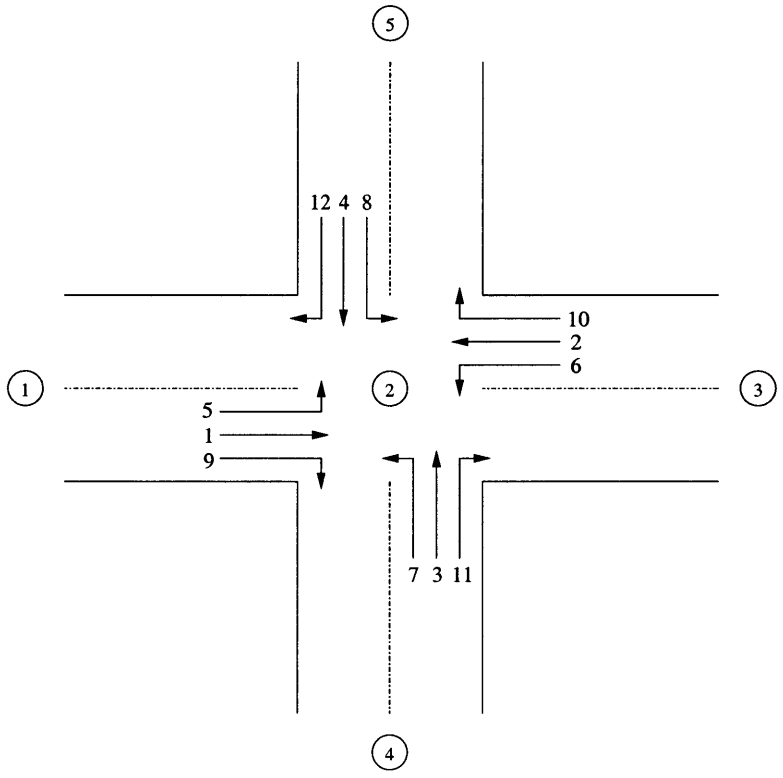

Figure 5-6: Movements at a Signalized Intersection

Let $w_{hij}(t)$ be the penalty (the amount of waiting time) when one arrives at node i at time t from node h and then wishes to take link (i, j). Once we construct the penalty table such as Table 5.6, we can easily obtain the value of $w_{hij}(t)$ from it. For instance, $w_{123}(3) = 7$, $w_{125}(0) = 6$ in this example. For trips commencing at node i at time t, we

Table 5.5: Signal Phases

	Phase I	Phase II	Phase III	Phase IV
Allowed Movements	1, 2, 9, 10	3, 4, 11, 12	5, 6	7, 8
Duration (units of time)	3	3	2	2

Table 5.6: Penalties (units of time)

Arrival Time (t)	Signal Phase	Penalty for Phase I Movements	Penalty for Phase II Movements	Penalty for Phase III Movements	Penalty for Phase IV Movements
0	I	0	3	6	8
1	I	0	2	5	7
2	I	0	1	4	6
3	II	7	0	3	5
4	II	6	0	2	4
5	II	5	0	1	3
6	III	4	7	0	2
7	III	3	6	0	1
8	IV	2	5	8	0
9	IV	1	4	7	0
10	I	0	3	6	8
11	I	0	2	5	7
⋮	⋮	⋮	⋮	⋮	⋮

define $w_{iij}(t) = 0$ for all $j \in \mathcal{O}(i)$.

Below we derive optimality conditions for the all-to-one minimum expected travel time next-arc hyperpaths problem in a signalized network. As usual, we assume that waiting at a node is not allowed unless it is required due to the red signal.

Optimality Conditions

We denote by $\overline{T}_{hij}(t)$ the travel time from node i to the destination node d when a traveler arrives at node i at time t from node h and then wants to take link (i,j) and an optimal next-arc hyperpath from node j to node d. $\overline{T}_{hij}(t)$ is the sum of three quantities as follows:

$$\begin{aligned}\overline{T}_{hij}(t) &= \text{waiting time for the signal at node } i + \\ &\quad \text{traversal time on link } (i,j) + \\ &\quad \text{travel time on an optimal next-arc hyperpath from node } j \text{ node } d \\ &= w_{hij}(t) + T_{ij}(t + w_{hij}(t)) + L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t))), \qquad (5.23)\end{aligned}$$

where $L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t)))$ is a random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d when the traveler arrives at node j at time $t + w_{hij}(t) + T_{ij}(t + w_{hij}(t))$ from node i.

Since $w_{hij}(t)$ is a constant for a given h, i, j and t, the expected value of $\overline{T}_{hij}(t)$ is

$$\begin{aligned}E[\overline{T}_{hij}(t)] &= E[w_{hij}(t) + T_{ij}(t + w_{hij}(t)) + L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t)))] \\ &= E[w_{hij}(t)] + E[T_{ij}(t + w_{hij}(t))] + E[L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t)))] \\ &= w_{hij}(t) + E[T_{ij}(t + w_{hij}(t))] + E[L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t)))]. \qquad (5.24)\end{aligned}$$

Let $e_{hi}(t)$ be the minimum expected travel time from node i to node d under the adaptive routing policy when the traveler arrives at node i at time t from node h. $e_{hi}(t)$ is obtained by

$$\begin{aligned}e_{hi}(t) &= \min_{j \in \mathcal{O}(i)} \left\{ E[\overline{T}_{hij}(t)] \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ w_{hij}(t) + E[T_{ij}(t + w_{hij}(t))] + E[L_{ij}(t + w_{hij}(t) + T_{ij}(t + w_{hij}(t)))] \right\}. \qquad (5.25)\end{aligned}$$

$E[T_{ij}(t+w_{hij}(t))] = \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} \tau_{ij}^r(t+w_{hij}(t))\, p_{ij}^r(t+w_{hij}(t))$ and $E[L_{ij}(t+w_{hij}(t)+ T_{ij}(t+w_{hij}(t)))]$ is computed by

$$\begin{aligned} &E[L_{ij}(t+w_{hij}(t)+T_{ij}(t+w_{hij}(t)))] \\ &\quad = E[E[L_{ij}(t+w_{hij}(t)+T_{ij}(t+w_{hij}(t))) \mid T_{ij}(t+w_{hij}(t))]] \\ &\quad = \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} E[L_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))]\, p_{ij}^r(t+w_{hij}(t)). \end{aligned} \tag{5.26}$$

Therefore we can rewrite (5.25) as follows.

$$\begin{aligned} e_{hi}(t) = \min_{j\in\mathcal{O}(i)} \Bigg\{ & w_{hij}(t) + \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} \tau_{ij}^r(t+w_{hij}(t))\, p_{ij}^r(t+w_{hij}(t)) + \\ & \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} E[L_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))]\, p_{ij}^r(t+w_{hij}(t)) \Bigg\}. \end{aligned} \tag{5.27}$$

Since $e_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t))) = E[L_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))]$ by the definitions of $e_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))$ and $L_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))$, (5.27) is rewritten as

$$\begin{aligned} e_{hi}(t) &= \min_{j\in\mathcal{O}(i)} \Bigg\{ w_{hij}(t) + \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} \tau_{ij}^r(t+w_{hij}(t))\, p_{ij}^r(t+w_{hij}(t)) + \\ &\qquad \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} e_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t)))\, p_{ij}^r(t+w_{hij}(t)) \Bigg\} \\ &= \min_{j\in\mathcal{O}(i)} \Bigg\{ \sum_{r\in\mathcal{R}_{ij}(t+w_{hij}(t))} \Big(w_{hij}(t) + \tau_{ij}^r(t+w_{hij}(t)) + \\ &\qquad e_{ij}(t+w_{hij}(t)+\tau_{ij}^r(t+w_{hij}(t))) \Big)\, p_{ij}^r(t+w_{hij}(t)) \Bigg\}. \end{aligned} \tag{5.28}$$

When $i = d$, we define $e_{hd}(t) = 0$ for all $h \in \mathcal{I}(d) \cup \{d\}$ and all t. Once all $e_{hi}(t)$ are determined, the minimum expected travel time of a trip starting from node i at time t is given by $e_{ii}(t)$.

We assume that no signal effects prevail when $t > H-1$. Therefore $w_{hij}(t) = 0$ for all $t > H-1$ and (5.28) reduces to

$$e_{hi}(t) = e_{hi}(H) = \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(H)} \left(\tau_{ij}^r(H) + e_{ij}(H)\right)\ p_{ij}^r(H) \right\}$$
$$= \min_{j \in \mathcal{O}(i)} \{E[T_{ij}(H)] + e_{ij}(H)\}. \tag{5.29}$$

Since no signal exists, which node a traveler arrives at node i from, i.e. the predecessor of node i, is not relevant when $t > H - 1$. Hence (5.29) can be further reduced to

$$e_i(t) = e_i(H) = \min_{j \in \mathcal{O}(i)} \{E[T_{ij}(H)] + e_j(H)\}. \tag{5.30}$$

As in Section 5.4, $e_i(H)$ satisfying (5.30) can be obtained by solving an all-to-one static shortest paths problem using $E[T_{ij}(H)]$ as link travel times. Once $e_i(H)$ are determined, we set $e_{hi}(H) = e_i(H)$ for all $h \in \mathcal{I}(i) \cup \{i\}$. Then $e_{hi}(t)$ for $t \in \mathcal{H}$ can be determined by (5.28) in a decreasing order of departure time in a single pass.

Proposition 12. *All labels $e_i(t)$ satisfying the optimality conditions (5.28) and (5.29) can be determined in a decreasing order of departure time in a single pass.*

Proof. A detailed proof is similar to the proof of Proposition 8. □

Solution Algorithm

We present a solution algorithm called *Algorithm METTH–Signal* in Algorithm 11. $s_{hi}(t)$ denotes the successor of node i on the minimum expected travel time next-arc hyperpath from node i to node d when one arrives at node i at time t from node h.

Proposition 13. *The worst-case running time complexity of Algorithm METTH–Signal is* $O(\max(f(n,m), n\widetilde{m}))$.

Proof. It takes $\Theta(nH + mH)$ time to initialize $(e_{hi}(t), s_{hi}(t))$ in Step 1. The computation of $w_{hij}(t)$ in Step 1 requires $O(nmH)$ time. In Step 3, we need to perform

$$\Theta\Big(\sum_{t \in \mathcal{H}} \sum_{i \in \mathcal{N} \setminus \{d\}} \sum_{h \in \mathcal{I}(i) \cup \{i\}} \sum_{j \in \mathcal{O}(i)} |\mathcal{R}_{ij}(t + w_{hij}(t))|\Big) \subset O(n\widetilde{m})$$

operations. Hence the worst-case running time complexity of Algorithm METTH–Signal is $O(nH + mH + nmH + f(n,m) + n\widetilde{m}) = O(\max(f(n,m), n\widetilde{m}))$. □

Algorithm 11 Algorithm METTH–Signal

Step 1: Initialization

$(e_{hi}(t), s_{hi}(t)) \leftarrow (\infty, \infty), \quad \forall i \in \mathcal{N} \setminus \{d\}, \ \forall h \in \mathcal{I}(i) \cup \{i\}, \ \forall t \in \mathcal{H}$

$(e_{hd}(t), s_{hd}(t)) \leftarrow (0, d), \quad \forall h \in \mathcal{I}(d) \cup \{d\}, \ \forall t \in \mathcal{H}$

Compute $w_{hij}(t), \quad \forall i \in \mathcal{N}, \ \forall h \in \mathcal{I}(i) \cup \{i\}, \ \forall j \in \mathcal{O}(i), \ \forall t \in \mathcal{H}$

Step 2: Minimum Expected Time Hyperpaths in Static Domain

$(e_{hi}(H), s_{hi}(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{E[T_{ij}(H)]\}, d)$

Step 3: Minimum Expected Time Hyperpaths in Time-Dependent Domain

For $t \leftarrow H - 1$ down to 0 **do**

For $i \in \mathcal{N} \setminus \{d\}$ **do**

For $h \in \mathcal{I}(i) \cup \{i\}$ **do**

$$e_{hi}(t) \leftarrow \min_{j \in \mathcal{O}(i)} \Big\{ \sum_{r \in \mathcal{R}_{ij}(t + w_{hij}(t))} \big(w_{hij}(t) + \tau_{ij}^r(t + w_{hij}(t)) + e_{ij}(t + w_{hij}(t) + \tau_{ij}^r(t + w_{hij}(t))) \big) \, p_{ij}^r(t + w_{hij}(t)) \Big\}$$

$$s_{hi}(t) \leftarrow \arg\min_{j \in \mathcal{O}(i)} \Big\{ \sum_{r \in \mathcal{R}_{ij}(t + w_{hij}(t))} \big(w_{hij}(t) + \tau_{ij}^r(t + w_{hij}(t)) + e_{ij}(t + w_{hij}(t) + \tau_{ij}^r(t + w_{hij}(t))) \big) \, p_{ij}^r(t + w_{hij}(t)) \Big\}$$

End (For)

End (For)

End (For)

Let $R = \max_{(i,j)\in\mathcal{A},\, t\in\mathcal{H}} \{|\mathcal{R}_{ij}(t)|\}$. Then the worst-case running time complexity of Algorithm METTH–Signal is bounded from above by $O(\max(f(n,m), nmHR)$. Note that the worst-case running time complexity of the solution algorithm for the same problem proposed in Yang and Miller-Hooks [25] is $O(\max(f(n,m), n^4H^2R))$. Since $m < n^2$, Algorithm METTH–Signal has a theoretically better worst-case running time than Yang and Miller-Hooks' algorithm.

5.8.3 Multimodal Networks

If there exist several travel modes in a network, traveling via a single mode for the entire trip from an origin node to a destination node might not necessarily result in the minimum expected travel time. By changing travel modes judiciously along the trip, a traveler may reduce his expected travel time. We discuss here how to find all-to-one minimum expected travel time next-arc hyperpaths in stochastic time-dependent multimodal networks. Let us first introduce additional notation and assumptions for this problem.

Additional Notation and Assumptions

Let $\mathcal{M} = \{1, \cdots, M\}$ be the set of travel modes available in the network. Some modes may not be available on some links. We denote by $\mathcal{M}_{ij} \subseteq \mathcal{M}$ the set of modes available on link (i,j).

For each link $(i,j) \in \mathcal{A}$ and each mode $b \in \mathcal{M}_{ij}$, let random variable $T_{ij}^b(t)$ denote the travel time on link (i,j) via mode b when one enters the link at time t. We assume that $T_{ij}^b(t)$ is a discrete random variable that takes $r_{ij}^b(t)$ distinct values denoted by $\tau_{ij}^{br}(t)$, $r \in \mathcal{R}_{ij}^b(t) = \{1, \cdots, r_{ij}^b(t)\}$. The probability that $\tau_{ij}^{br}(t)$ occurs is denoted by $p_{ij}^{br}(t)$. The PMF of $T_{ij}^b(t)$ is then represented by

$$p_{T_{ij}^b(t)} = \left\{(\tau_{ij}^{br}(t),\, p_{ij}^{br}(t)) \mid r \in \mathcal{R}_{ij}^b(t)\right\}. \tag{5.31}$$

When a traveler changes modes at nodes, mode transfer delays arise. In order to take into consideration the variability of these delays, we introduce another time-dependent random variable, $D_{hij}^{ab}(t)$, where $(h,i) \in \mathcal{A}, (i,j) \in \mathcal{A}$ and $a \in \mathcal{M}_{hi}$, $b \in \mathcal{M}_{ij}$. This random variable indicates the mode transfer delay when one arrives at node i at time t from node h via mode a and then takes mode b to go to node j. We assume that $D_{hij}^{ab}(t)$ is also a

discrete random variable with the following PMF:

$$p_{D^{ab}_{hij}(t)} = \left\{(\delta^{ab^s}_{hij}(t),\ q^{ab^s}_{hij}(t)) \mid s \in \mathcal{R}^{ab}_{hij}(t)\right\}, \tag{5.32}$$

where $\delta^{ab^s}_{hij}(t)$ is the mode transfer delay value of the s^{th} realization of $D^{ab}_{hij}(t)$, $q^{ab^s}_{hij}(t)$ is the probability that $\delta^{ab^s}_{hij}(t)$ occurs, and $\mathcal{R}^{ab}_{hij}(t) = \left\{1, \cdots, r^{ab}_{hij}(t)\right\}$ is the set of indexes of realizations of $D^{ab}_{hij}(t)$.

If $b = a$, i.e. if one travels from node h to node j through node i using the same mode, no mode transfer delay is incurred at node i. Therefore the PMF of $D^{aa}_{hij}(t)$ is given by

$$p_{D^{aa}_{hij}(t)} = \left\{(0, 1.0)\right\}. \tag{5.33}$$

When $h = i$, i.e. when a traveler initiates his trip at node i, we assume that there are no mode transfer delays at node i. Hence the PMF of $D^{1b}_{iij}(t)$ is

$$p_{D^{1b}_{iij}(t)} = \left\{(0, 1.0)\right\}, \tag{5.34}$$

where we define, by convention, the fictitious mode set $\mathcal{M}_{ii}$ as $\{1\}$, $\forall\, i \in \mathcal{N}$.

We assume that all $\tau^{b^r}_{ij}(t)$ are positive integers and that $\delta^{ab^s}_{hij}(t)$ are likewise if $a \neq b$ and $h \neq i$. We also assume that for all $t > H - 1$, $p_{T^b_{ij}(t)} = p_{T^b_{ij}(H-1)}$ and $p_{D^{ab}_{hij}(t)} = p_{D^{ab}_{hij}(H-1)}$. It is assumed that waiting at nodes is not allowed unless it is required for mode transfer.

Optimality Conditions

Let $\overline{T}^{ab}_{hij}(t)$ denote the travel time from node i to the destination node d when one arrives at node i at time t from node h via mode a and takes link (i, j) using mode b and then follows an optimal next-arc hyperpath from node j to node d. $\overline{T}^{ab}_{hij}(t)$ can be expressed as the sum of three random variables as follows:

$$\begin{aligned}
\overline{T}^{ab}_{hij}(t) &= \text{mode transfer delay at node } i + \\
&\quad \text{traversal time on link } (i, j) + \\
&\quad \text{travel time on an optimal next-arc hyperpath from node } j \text{ to node } d \\
&= D^{ab}_{hij}(t) + T^b_{ij}(t + D^{ab}_{hij}(t)) + L^b_{ij}(t + D^{ab}_{hij}(t) + T^b_{ij}(t + D^{ab}_{hij}(t))),
\end{aligned} \tag{5.35}$$

where $L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)))$ is a random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d when one arrives at node j at time $t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t))$ from node i via mode b.

The expected value of $\overline{T}_{hij}^{ab}(t)$ is

$$\begin{aligned} E[\overline{T}_{hij}^{ab}(t)] &= E[D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)) + L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}(t + D_{hij}^{ab}(t)))] \\ &= E[D_{hij}^{ab}(t)] + E[T_{ij}^{b}(t + D_{hij}^{ab}(t))] + E[L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)))]. \end{aligned} \tag{5.36}$$

Let $e_{hi}^{a}(t)$ be the minimum expected travel time from node i to node d under the adaptive routing policy when one arrives at node i at time t from node h via mode a. $e_{hi}^{a}(t)$ is obtained by

$$\begin{aligned} e_{hi}^{a}(t) &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{b \in \mathcal{M}_{ij}} \left\{ E[\overline{T}_{hij}^{ab}(t)] \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \min_{b \in \mathcal{M}_{ij}} \left\{ E[D_{hij}^{ab}(t)] + E[T_{ij}^{b}(t + D_{hij}^{ab}(t))] + \right. \right. \\ &\qquad \left. \left. E[L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)))] \right\} \right\}. \end{aligned} \tag{5.37}$$

$E[D_{hij}^{ab}(t)] = \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \delta_{hij}^{ab^s}(t)\, q_{hij}^{ab^s}(t)$ and $E[T_{ij}^{b}(t + D_{hij}^{ab}(t))]$ is computed as follows:

$$\begin{aligned} E[T_{ij}^{b}(t + D_{hij}^{ab}(t))] &= E[E[T_{ij}^{b}(t + D_{hij}^{ab}(t)) \mid D_{hij}^{ab}(t)]] \\ &= \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} E[T_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))]\, q_{hij}^{ab^s}(t) \\ &= \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t))\, p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t). \end{aligned} \tag{5.38}$$

By using the total expectation theorem twice, $E[L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)))]$ is computed by

$$\begin{aligned} &E[L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t)))] \\ &= E[E[L_{ij}^{b}(t + D_{hij}^{ab}(t) + T_{ij}^{b}(t + D_{hij}^{ab}(t))) \mid D_{hij}^{ab}(t)]] \\ &= \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} E[L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + T_{ij}^{b}(t + \delta_{hij}^{ab^s}(t)))]\, q_{hij}^{ab^s}(t) \end{aligned}$$

$$= \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} E[E[L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + T_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))) \mid T_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))]]\; q_{hij}^{ab^s}(t)$$

$$= \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} E[L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))]\; p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t). \tag{5.39}$$

Using (5.38) and (5.39), we rewrite (5.37) as

$$e_{hi}^{a}(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{b \in \mathcal{M}_{ij}} \left\{ \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \delta_{hij}^{ab^s}(t)\; q_{hij}^{ab^s}(t) + \right.\right.$$

$$\sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t))\; p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t) +$$

$$\left.\left. \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} E[L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))]\; p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t) \right\} \right\}. \tag{5.40}$$

By the definitions of $L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))$ and $e_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))$, we have $e_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t))) = E[L_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))]$. Hence (5.40) is the same as

$$e_{hi}^{a}(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_{b \in \mathcal{M}_{ij}} \left\{ \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \delta_{hij}^{ab^s}(t)\; q_{hij}^{ab^s}(t) + \right.\right.$$

$$\sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t))\; p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t) +$$

$$\left.\left. \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} e_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)))\; p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t) \right\} \right\}$$

$$= \min_{j \in \mathcal{O}(i)} \left\{ \min_{b \in \mathcal{M}_{ij}} \left\{ \sum_{s \in \mathcal{R}_{hij}^{ab}(t)} \left[\sum_{r \in \mathcal{R}_{ij}^{b}(t + \delta_{hij}^{ab^s}(t))} \left(\delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) + \right.\right.\right.\right.$$

$$\left.\left.\left.\left. e_{ij}^{b}(t + \delta_{hij}^{ab^s}(t) + \tau_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t))) \right) p_{ij}^{b^r}(t + \delta_{hij}^{ab^s}(t)) \right] q_{hij}^{ab^s}(t) \right\} \right\}. \tag{5.41}$$

For $t > H - 1$, (5.41) reduces to the following functional form:

$$e^a_{hi}(t) = e^a_{hi}(H) =$$
$$\min_{j\in\mathcal{O}(i)}\left\{\min_{b\in\mathcal{M}_{ij}}\left\{\sum_{s\in\mathcal{R}^{ab}_{hij}(H)}\left[\sum_{r\in\mathcal{R}^{b}_{ij}(H)}\left(\delta^{ab^s}_{hij}(H)+\tau^{b^r}_{ij}(H)+e^b_{ij}(H)\right)p^{b^r}_{ij}(H)\right]q^{ab^s}_{hij}(H)\right\}\right\}. \tag{5.42}$$

(5.41) and (5.42) together with $e^a_{hd}(t)=0$, $\forall\, h\in\mathcal{I}(d)\cup\{d\}$, $\forall\, a\in\mathcal{M}_{hd}$, $\forall\, t$ are optimality conditions for the all-to-one minimum expected travel time next-arc hyperpaths problem in stochastic time-dependent multimodal networks. Once all $e^a_{hi}(t)$ are computed, $e^1_{ii}(t)$ contains the minimum expected travel time for a trip commencing at node i at time t.

Note that in (5.41), $e^a_{hi}(t)$ depends only on $e^b_{ij}(s)$, where $s>t$. Hence all $e^a_{hi}(t)$ can be set in a decreasing order of departure time. A detailed proof of the following proposition is similar to that of Proposition 8, so we do not include it here.

Proposition 14. *All labels $e^a_{hi}(t)$ satisfying the optimality conditions (5.41) and (5.42) can be determined in a decreasing order of departure time in a single pass.*

Before the determination of $e^a_{hi}(t)$ for $t\in\mathcal{H}$, we need to compute $e^a_{hi}(H)$ by (5.42) as a base case. We can rewrite (5.42) as follows:

$$\begin{aligned}
e^a_{hi}(H) &= \min_{j\in\mathcal{O}(i)}\left\{\min_{b\in\mathcal{M}_{ij}}\left\{\sum_{s\in\mathcal{R}^{ab}_{hij}(H)}\left[\delta^{ab^s}_{hij}(H)\sum_{r\in\mathcal{R}^{b}_{ij}(H)}p^{b^r}_{ij}(H)+\right.\right.\right.\\
&\qquad\left.\left.\left.\sum_{r\in\mathcal{R}^{b}_{ij}(H)}\tau^{b^r}_{ij}(H)\,p^{b^r}_{ij}(H)+e^b_{ij}(H)\sum_{r\in\mathcal{R}^{b}_{ij}(H)}p^{b^r}_{ij}(H)\right]q^{ab^s}_{hij}(H)\right\}\right\}\\
&= \min_{j\in\mathcal{O}(i)}\left\{\min_{b\in\mathcal{M}_{ij}}\left\{\sum_{s\in\mathcal{R}^{ab}_{hij}(H)}\left[\delta^{ab^s}_{hij}(H)+E[T^b_{ij}(H)]+e^b_{ij}(H)\right]q^{ab^s}_{hij}(H)\right\}\right\}\\
&= \min_{j\in\mathcal{O}(i)}\left\{\min_{b\in\mathcal{M}_{ij}}\left\{E[D^{ab}_{hij}(H)]+E[T^b_{ij}(H)]+e^b_{ij}(H)\right\}\right\}.
\end{aligned} \tag{5.43}$$

This implies that $e^a_{hi}(H)$ are determined by solving an all-to-one multimodal static shortest paths problem.

Solution Algorithm

Let $s^a_{hi}(t), m^a_{hi}(t)$ respectively denote the successor of node i and the travel mode for link $(i, s^a_{hi}(t))$ on the minimum expected travel time next-arc hyperpath from node i to node

d when a traveler arrives at node i at time t from node h via mode a. In Algorithm 12, we propose a label-setting algorithm called *Algorithm METTH–Multimodal* that solves the all-to-one minimum expected travel time next-arc hyperpaths problem in stochastic time-dependent multimodal networks.

Proposition 15. *Let* $D = \max_{(h,i)\in\mathcal{A},(i,j)\in\mathcal{A},a\in\mathcal{M}_{hi},b\in\mathcal{M}_{ij},t\in\mathcal{H}} \{|\mathcal{R}^{ab}_{hij}(t)|\}$, *and let* $O(g(n,m,M))$ *be the running time to solve an all-to-one multimodal static shortest paths problem in Step 2. Then the worst-case running time complexity of Algorithm METTH–Multimodal is* $O(\max(g(n,m,M), nM^2D\widetilde{m}))$.

Proof. It takes $O(mHD)$ time to initialize $(e^a_{hi}(t), s^a_{hi}(t), m^a_{hi}(t))$ in Step 1. In Step 3, we need to perform

$$\Theta\Big(\sum_{t\in\mathcal{H}}\sum_{i\in\mathcal{N}\setminus\{d\}}\sum_{h\in\mathcal{I}(i)\cup\{i\}}\sum_{a\in\mathcal{M}_{hi}}\sum_{j\in\mathcal{O}(i)}\sum_{b\in\mathcal{M}_{ij}}\sum_{s\in\mathcal{R}^{ab}_{hij}(t)}|\mathcal{R}^{b}_{ij}(t+\delta^{abs}_{hij}(t))|\Big) \subset O(nM^2D\widetilde{m})$$

operations. Hence the worst-case running time complexity of Algorithm METTH–Multimodal is $O(mHD + g(n,m,M) + nM^2D\widetilde{m}) = O(\max(g(n,m,M), nM^2D\widetilde{m}))$. □

Opasanon and Miller-Hooks [21] propose a label-correcting algorithm for the same problem, whose worst-case running time complexity is $O(\max(g(n,m,M), n^5H^2M^3DR))$, where $R = \max_{(i,j)\in\mathcal{A},b\in\mathcal{M}_{ij},t\in\mathcal{H}} \{|\mathcal{R}^{b}_{ij}(t)|\}$. Since the worst-case running time complexity of Algorithm METTH–Multimodal is bounded from above by $O(\max(g(n,m,M), nM^2DmHR))$, Algorithm METTH–Multimodal has a better worst-case running time complexity than Opasanon and Miller-Hooks' algorithm.

Algorithm 12 Algorithm METTH–Multimodal

1: **Step 1: Initialization**

2: $(e^a_{hi}(t), s^a_{hi}(t), m^a_{hi}(t)) \leftarrow (\infty, \infty, \infty), \quad \forall i \in \mathcal{N} \setminus \{d\}, \ \forall h \in \mathcal{I}(i) \cup \{i\}, \ \forall a \in \mathcal{M}_{hi}, \ \forall t \in \mathcal{H}$

3: $(e^a_{hd}(t), s^a_{hd}(t), m^a_{hd}(t)) \leftarrow (0, d, 0), \quad \forall h \in \mathcal{I}(d) \cup \{d\}, \ \forall a \in \mathcal{M}_{hd}, \ \forall t \in \mathcal{H}$

4: **Step 2: Minimum Expected Time Multimodal Hyperpaths in Static Domain**

5: $(e^a_{hi}(H), s^a_{hi}(H), m^a_{hi}(H)) \leftarrow$ All-to-One MSP$(\mathcal{N}, \mathcal{A}, \mathcal{M}, \{E[T^a_{ij}(H)]\}, \{E[D^{ab}_{hij}(H)]\}, d)$

6: **Step 3: Minimum Expected Time Multimodal Hyperpaths in Time-Dependent Domain**

7: **For** $t \leftarrow H - 1$ down to 0 **do**

8: **For** $i \in \mathcal{N} \setminus \{d\}$ **do**

9: **For** $h \in \mathcal{I}(i) \cup \{i\}$ **do**

10: **For** $a \in \mathcal{M}_{hi}$ **do**

11: $$e^a_{hi}(t) \leftarrow \min_{j \in \mathcal{O}(i)} \Big\{ \min_{b \in \mathcal{M}_{ij}} \Big\{ \sum_{s \in \mathcal{R}^{ab}_{hij}(t)} \Big[\sum_{r \in \mathcal{R}^b_{ij}(t + \delta^{ab^s}_{hij}(t))} \big(\delta^{ab^s}_{hij}(t) + \tau^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)) + e^b_{ij}(t + \delta^{ab^s}_{hij}(t) + \tau^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)))\big) \, p^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)) \Big] \, q^{ab^s}_{hij}(t) \Big\} \Big\}$$

12: $$(s^a_{hi}(t), m^a_{hi}(t)) \leftarrow \arg\min_{(j,b) \in \mathcal{O}(i) \times \mathcal{M}_{ij}} \Big\{ \sum_{s \in \mathcal{R}^{ab}_{hij}(t)} \Big[\sum_{r \in \mathcal{R}^b_{ij}(t + \delta^{ab^s}_{hij}(t))} \big(\delta^{ab^s}_{hij}(t) + \tau^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)) + e^b_{ij}(t + \delta^{ab^s}_{hij}(t) + \tau^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)))\big) \, p^{b^r}_{ij}(t + \delta^{ab^s}_{hij}(t)) \Big] \, q^{ab^s}_{hij}(t) \Big\}$$

13: **End (For)**

14: **End (For)**

15: **End (For)**

16: **End (For)**

Chapter 6

Minimum Expected Travel Time Paths

In this chapter, we explore another class of expected travel time-based routing problems in stochastic time-dependent networks, called the *minimum expected travel time path (METTP) problem.* Recall that this problem adopts the non-adaptive routing policy, so recourse en route is not allowed.

Minimum expected travel time paths are important to travelers who do not have appropriate in-vehicle route guidance equipments and therefore rely on a priori minimum expected travel time paths information for their routing planning. This problem may also appear as a subproblem in other routing applications where a simple path rather than a next-arc hyperpath is sought as an optimal solution.

In this chapter, we focus on the *all-to-one minimum expected travel time paths problem,* one variant of the minimum expected travel time path problem, which is defined as follows: *Given a stochastic time-dependent network, the problem is to find a simple path with the minimum expected travel time from each node to a given destination node for each departure time.*

We discuss optimality conditions for the all-to-one minimum expected travel time paths problem in Section 6.1. In Section 6.2, we explain why this problem is difficult to solve efficiently. We present ideas of possible solution algorithms for the problem together with a known algorithm in the literature in Section 6.3.

6.1 Optimality Conditions

To derive optimality conditions for the all-to-one minimum expected travel time paths problem, we use the notation used in Chapter 4. Recall that $\overline{T}_{ij}^{c}(t)$, the travel time from node i to the destination node d when a traveler leaves node i at time t by taking link (i,j) and then follows path c from node j to node d, can be expressed as follows:

$$\begin{aligned}\overline{T}_{ij}^{c}(t) &= \text{traversal time on link } (i,j) + \\ &\quad \text{travel time on path } c \text{ from node } j \text{ to node } d \\ &= T_{ij}(t) + L_j^c(t + T_{ij}(t)). \end{aligned} \tag{6.1}$$

The expected value of $\overline{T}_{ij}^{c}(t)$ is

$$E[\overline{T}_{ij}^{c}(t)] = E[T_{ij}(t) + L_j^c(t + T_{ij}(t))] = E[T_{ij}(t)] + E[L_j^c(t + T_{ij}(t))]. \tag{6.2}$$

Let us denote by $\overline{e}_i(t)$ the minimum expected travel time from node i to node d at departure time t under the non-adaptive routing policy. In other words, it is the expected travel time on the minimum expected travel time path from node i to node d at departure time t. $\overline{e}_i(t)$ is obtained by

$$\overline{e}_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \min_c \left\{ E[\overline{T}_{ij}^{c}(t)] \right\} \right\} = \min_{j \in \mathcal{O}(i)} \left\{ \min_c \left\{ E[T_{ij}(t)] + E[L_j^c(t + T_{ij}(t))] \right\} \right\}. \tag{6.3}$$

Rewriting (6.3) using the realizations of $T_{ij}(t)$, we have

$$\begin{aligned}\overline{e}_i(t) &= \min_{j \in \mathcal{O}(i)} \left\{ \min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \sum_{r \in \mathcal{R}_{ij}(t)} E[L_j^c(t + \tau_{ij}^r(t))]\, p_{ij}^r(t) \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} E[L_j^c(t + \tau_{ij}^r(t))]\, p_{ij}^r(t) \right\} \right\}. \end{aligned} \tag{6.4}$$

Unlike other problems we have seen so far, we cannot simplify (6.4) any further. Note that attempting to rewrite (6.4) as the following functional form is not correct.

$$\overline{e}_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \sum_{r \in \mathcal{R}_{ij}(t)} \overline{e}_j(t + \tau_{ij}^r(t))\, p_{ij}^r(t) \right\}. \tag{6.5}$$

The reason is as follows. Let $z = \arg\min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} E[L_j^c(t + \tau_{ij}^r(t))]\ p_{ij}^r(t) \right\}$. Then it is not guaranteed that $E[L_j^z(t + \tau_{ij}^r(t))] = \overline{e}_j(t + \tau_{ij}^r(t))$ for all $r \in \mathcal{R}_{ij}(t)$. For some $q \in \mathcal{R}_{ij}(t)$, some path (called w) other than z can be the minimum expected travel time path, i.e. $E[L_j^z(t + \tau_{ij}^q(t))] > \overline{e}_j(t + \tau_{ij}^q(t)) = E[L_j^w(t + \tau_{ij}^q(t))]$.

To put it in other words, since $\overline{e}_j(t + \tau_{ij}^r(t)) = \min_c \left\{ E[L_j^c(t + \tau_{ij}^r(t))] \right\}$ by the definition of $\overline{e}_j(t + \tau_{ij}^r(t))$, (6.5) is equivalent to the following formula which is clearly different from (6.4).

$$\overline{e}_i(t) = \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\ p_{ij}^r(t) + \sum_{r \in \mathcal{R}_{ij}(t)} \min_c \left\{ E[L_j^c(t + \tau_{ij}^r(t))] \right\} p_{ij}^r(t) \right\}. \tag{6.6}$$

Nonetheless, (6.4) gives the following optimality conditions for the all-to-one minimum expected travel time paths problem:

$$\overline{e}_i(t) = \begin{cases} \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\ p_{ij}^r(t) + \min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} E[L_j^c(t + \tau_{ij}^r(t))]\ p_{ij}^r(t) \right\} \right\} & \forall i \neq d,\ \forall t \in \mathcal{H}, \\ 0 & i = d,\ \forall t \in \mathcal{H}, \end{cases} \tag{6.7}$$

$$\overline{e}_i(t) = \overline{e}_i(H) = \begin{cases} \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(H)} \tau_{ij}^r(H)\ p_{ij}^r(H) + \min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(H)} E[L_j^c(H)]\ p_{ij}^r(H) \right\} \right\} & \forall i \neq d,\ \forall t \notin \mathcal{H}, \\ 0 & i = d,\ \forall t \notin \mathcal{H}. \end{cases} \tag{6.8}$$

Note that we can simplify (6.8) as follows:

$$\begin{aligned} \overline{e}_i(H) &= \min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(H)} \tau_{ij}^r(H)\ p_{ij}^r(H) + \min_c \left\{ E[L_j^c(H)] \sum_{r \in \mathcal{R}_{ij}(H)} p_{ij}^r(H) \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(H)] + \min_c \left\{ E[L_j^c(H)] \right\} \right\} \\ &= \min_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(H)] + \overline{e}_j(H) \right\}, \quad \forall i \neq d. \end{aligned} \tag{6.9}$$

(6.9) is the set of optimality conditions for the all-to-one shortest paths problem in a deterministic static network where link travel time values are set to $E[T_{ij}(H)]$.

Let $s_i(t)$ be the successor of node i on the minimum expected travel time path from node i to node d at departure time t. Let $\kappa_i(t)$ denote the index of the path from node $s_i(t)$ to node d that is a subpath of the minimum expected travel time path from node i to node d at departure time t. For $i \in \mathcal{N} \setminus \{d\}$ and $t \in \mathcal{H}$, $s_i(t)$ and $\kappa_i(t)$ are obtained by

$$s_i(t) = \arg\min_{j \in \mathcal{O}(i)} \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} \tau_{ij}^r(t)\, p_{ij}^r(t) + \min_c \left\{ \sum_{r \in \mathcal{R}_{ij}(t)} E[L_j^c(t + \tau_{ij}^r(t))]\, p_{ij}^r(t) \right\} \right\}, \quad (6.10)$$

$$\kappa_i(t) = \arg\min_c \left\{ \sum_{r \in \mathcal{R}_{is_i(t)}(t)} E[L_{s_i(t)}^c(t + \tau_{is_i(t)}^r(t))]\, p_{is_i(t)}^r(t) \right\}. \quad (6.11)$$

When $t > H - 1$,

$$s_i(t) = s_i(H) = \arg\min_{j \in \mathcal{O}(i)} \left\{ E[T_{ij}(H)] + \overline{e}_j(H) \right\}. \quad (6.12)$$

Proposition 16. *All labels $\overline{e}_i(t)$ can be determined in a decreasing order of departure time in a single pass.*

Proof. Since $\overline{e}_j(t + \tau_{ij}^r(t)) = \min_c \left\{ E[L_j^c(t + \tau_{ij}^r(t))] \right\}$, $\overline{e}_i(t)$ is dependent upon $\overline{e}_j(t + \tau_{ij}^r(t))$ and other $E[L_j^c(t + \tau_{ij}^r(t))]$ values in (6.7). Since all $\tau_{ij}^r(t)$ are positive, no $\overline{e}_j(s)$ where $s < t$ can affect the computation of $\overline{e}_i(t)$. This concludes that all $\overline{e}_i(t)$ can be determined in a decreasing order of departure time, starting from the determination of $\overline{e}_i(H)$. To prove that $\overline{e}_i(t)$ can be determined in a single pass, we have to show that once $\overline{e}_i(t)$ and $E[L_i^c(t)]$ are computed at departure time t, they do not change later. A proof of this is similar to one given in Proposition 8, so we omit it. □

The optimality conditions (6.7), however, reflect a difficulty in solving the problem since all simple paths from node j to node d should be considered in order to compute $\overline{e}_i(t)$. If we do not consider all paths from node j to node d, Proposition 16 does not hold and any single pass decreasing order of departure time algorithm results in suboptimal solutions. Since the enumeration of all paths is a very time-consuming task even for moderate size networks, single pass decreasing order of departure time algorithms are not applicable to the all-to-one minimum expected travel time paths problem.

6.2 Inherent Difficulty of the Problem

It should be noted that we need to keep not only $\overline{e}_i(t)$ but also all other $E[L_i^c(t)]$ at node i at departure time t because some $E[L_i^c(t)]$ might be necessary for the determination of some $\overline{e}_j(s)$ for $s < t$. The number of simple paths from a node to the destination node increases exponentially as the size of the network increases. Accordingly the number of $E[L_i^c(t)]$ to be kept at node i at departure time t also grows exponentially. This implies that the all-to-one minimum expected travel time paths problem is inherently difficult to solve efficiently.

Let us expand on this inherent difficulty. We have seen that each node maintains H labels, i.e. $e_i(t)$, $\forall t \in \mathcal{H}$, in the all-to-one minimum expected travel time next-arc hyperpaths problem. Each label is the expected travel time on the optimal next-arc hyperpath for the corresponding departure time. In the all-to-one minimum expected travel time paths problem, however, each node may have to keep much more than H labels. We illustrate this with an example after we introduce two dominance concepts for path elimination (Miller-Hooks [16]).

Consider all simple paths between node i and node d. *Pairwise-dominance* and *group-dominance* are respectively defined as follows:

Definition 1. *Path c is pairwise-dominated if there exists a path q such that $E[L_i^q(t)] \leq E[L_i^c(t)]$, $\forall t \in \mathcal{H}$ and $E[L_i^q(t)] < E[L_i^c(t)]$ for some t. Otherwise, path c is pairwise-non-dominated.*

Definition 2. *Path c is group-dominated if for each departure time t, there exists a path q such that $E[L_i^q(t)] < E[L_i^c(t)]$. Otherwise, path c is group-nondominated.*

For the pairwise-dominance, we compare the expected travel time on path c with the expected travel time on another path for all departure times. For the group-dominance, however, we check, for each departure time, if there always exists a path whose expected travel time is smaller than that of path c. Hence as the name indicates, we compare path c with a group of paths, not with a single path.

Consider the network example in Figure 6-1. There are three paths (a, b, and c) from node 2 to the destination node 3. The expected travel times on path a, b, and c at departure time 1 and 2 are shown below. The PMF of the travel time on link $(1, 2)$ at the link entry time 0 is also given.

Link $(1,2)$: $p_{T_{12}(0)} = \{(1, 0.5), (2, 0.5)\}$

Path a: $E[L_2^a(1)] = 5, \quad E[L_2^a(2)] = 10$

Path b: $E[L_2^b(1)] = 10, \quad E[L_2^b(2)] = 5$

Path c: $E[L_2^c(1)] = 7, \quad E[L_2^c(2)] = 7$

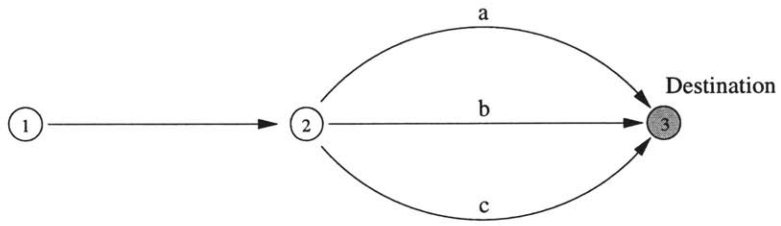

Figure 6-1: An Example Network

Notice that path c is group-dominated by paths a and b, but not pairwise-dominated by any other path. For trips that originate from node 2 at time 1 or 2, path c would be never used since there always exists another path with smaller expected travel time for each departure time. Now let us consider trips commencing at node 1 at time 0. The expected travel times on three paths from node 1 to node 3 are computed as follows:

$$E[L_1^{1\to 2\overset{a}{\rightsquigarrow}3}(0)] = \big(1 + E[L_2^a(1)]\big) \times 0.5 + \big(2 + E[L_2^a(2)]\big) \times 0.5 = 9.$$
$$E[L_1^{1\to 2\overset{b}{\rightsquigarrow}3}(0)] = \big(1 + E[L_2^b(1)]\big) \times 0.5 + \big(2 + E[L_2^b(2)]\big) \times 0.5 = 9.$$
$$E[L_1^{1\to 2\overset{c}{\rightsquigarrow}3}(0)] = \big(1 + E[L_2^c(1)]\big) \times 0.5 + \big(2 + E[L_2^c(2)]\big) \times 0.5 = 8.5.$$

Since path $1 \to 2 \overset{c}{\rightsquigarrow} 3$ gives the minimum expected travel time, trips commencing at node 1 at time 0 will use it. This implies that removing path c by the group-dominance at node 2 would result in a sub-optimal path selection from node 1. Hence we cannot eliminate paths by the group-dominance in the minimum expected travel time path problem. Note that if the group-dominance were applicable, each node would maintain at most H different paths and therefore would keep at most H^2 labels, indicating that a polynomial running time solution algorithm could be viable.

If a path is pairwise-dominated, we can safely eliminate it because it can never be used. However even if we reduce the number of paths associated with each node by the pairwise-dominance, the number of remaining paths (therefore labels) at each node can still

be enormous. This implies that the worst-case running time complexity of any algorithm for the all-to-one minimum expected travel time paths problem is non-polynomial.

6.3 Solution Algorithms

6.3.1 Algorithm METTH–based–METTP

In the all-to-one minimum expected travel time next-arc hyperpaths problem, it is possible that an optimal next-arc hyperpath from node i to node d at departure time t is a simple path. *Algorithm METTH–based–METTP* to be presented in this section is motivated by this possibility. The algorithm uses k minimum expected travel time next-arc hyperpaths to find the minimum expected travel time path from each node for each departure time.

We did not discuss the all-to-one $k-$minimum expected travel time next-arc hyperpaths problem in this thesis. However it is an easy extension of the all-to-one minimum expected travel time next-arc hyperpaths problem, just as we extended the all-to-one minimum possible travel time paths problem to the all-to-one $k-$minimum path travel time realizations problem.

Let us explain Algorithm METTH–based–METTP. In Step 1, the algorithm creates a set S and sets $P_i(t)$. Set S contains node-departure time pairs for which the minimum expected travel time paths have not been found yet. Set $P_i(t)$ possesses the indexes of next-arc hyperpaths that are simple paths from node i at departure time t. K is the number of next-arc hyperpaths to be found initially, which is a user input value.

In Step 2, k minimum expected travel time next-arc hyperpaths are determined for each $(i,t) \in S$.

In Step 3, the algorithm examines the k minimum expected travel time next-arc hyperpaths for each $(i,t) \in S$ to see whether any of those hyperpaths are actually simple paths. The indexes of hyperpaths that are simple paths are collected into set $P_i(t)$. If $P_i(t)$ is empty, it means that none of the current k minimum expected travel time next-arc hyperpaths are simple paths. Otherwise, we select a hyperpath, which is a simple path, with the minimum expected travel time from $P_i(t)$ and then remove (i,t) from S because we have found the minimum expected travel time path for that node-departure time pair. If S becomes empty, the algorithm is terminated. Otherwise, we increase the value of k by K and go to Step 2.

Algorithm 13 Algorithm METTH–based–METTP

Step 1: Initialization

Create a set $S \leftarrow \{(i,t) \mid \forall i \in \mathcal{N} \setminus \{d\},\ \forall t \in \mathcal{H}\}$

Create sets $P_i(t) \leftarrow \emptyset,\ \ \forall i \in \mathcal{N} \setminus \{d\},\ \forall t \in \mathcal{H}$

$k \leftarrow K$

Step 2: Determination of k Minimum Expected Travel Time Next-Arc Hyperpaths

Determine k minimum expected travel time next-arc hyperpaths from each $(i,t) \in S$ to the destination node d

Step 3: Checking for Simple Paths

For $(i,t) \in S$ **do**

  For $c \leftarrow 1$ to k **do**

    If the c^{th} minimum expected travel time next-arc hyperpath is a simple path **then**

      insert c into $P_i(t)$

    End (If)

  End (For)

  If $P_i(t) \neq \emptyset$ **then**

    $\overline{e}_i(t) = \min_{c \in P_i(t)} \{e_i^c(t)\}$

    Remove (i,t) from S

  End (If)

End (For)

If $S = \emptyset$ **then** stop

Else

  $k \leftarrow k + K$

  Go to **Step 2**

End (If)

$e_i^c(t)$ at Line 15 denotes the expected travel time on the c^{th} minimum expected travel time next-arc hyperpath from node i at departure time t.

6.3.2 Algorithm METTP–B&B

We propose another solution algorithm called *Algorithm METTP–B&B*, which employs a branch-and-bound method based on minimum possible travel times on paths. The algorithm is described in Algorithm 14.

In Step 1, we create a set S that contains node-departure time pairs for which the minimum expected travel time paths have not been identified yet. We then determine the minimum possible travel time paths from all nodes to the destination node d for all departure times. Algorithm MPTTP presented in Section 4.5.1 can be used for this purpose. Once the minimum possible travel time path from node i at departure time t is determined, we compute the expected travel time on that path. This expected travel time is obtained by using the following formulae recursively from node d until we get $e_i^*(t)$:

$$e_d^*(t) = 0, \quad \forall t \in \mathcal{H}, \tag{6.13}$$

$$e_j^*(t) = \sum_{r \in \mathcal{R}_{jk}(t)} \left(\tau_{jk}^r(t) + e_k^*(t + \tau_{jk}^r(t))\right)\ p_{jk}^r(t), \quad \forall t \in \mathcal{H}, \tag{6.14}$$

where node j is the precessor of node k on the path. This expected travel time is an upper bound, denoted by $e_i^U(t)$, on the minimum expected travel time from node i to node d for departure time t .

In Step 2, we determine the next minimum possible travel time path for each $(i,t) \in S$. The minimum travel time on this path is stored in $\lambda_i^k(t)$. Note that $\lambda_i^k(t)$ is a lower bound on the expected travel time of the new path just found and on the expected travel times of all other paths not yet considered from node i to node d for departure time t.

In Step 3, we compare $\lambda_i^k(t)$ with $e_i^U(t)$, i.e. the minimum travel time on the new path found in Step 2 with the current upper bound on the minimum expected travel time from node i to node d for departure time t.

If $\lambda_i^k(t)$ is not smaller than $e_i^U(t)$, then the expected travel time of the new path and the expected travel times of all other paths not yet considered cannot be smaller than $e_i^U(t)$. Therefore $e_i^U(t)$ is the minimum expected travel time from node i to node d for departure time t and the associated path is the minimum expected travel time path. Since we have

Algorithm 14 Algorithm METTP–B&B

Step 1: Initialization

Create a set $S \leftarrow \{(i,t) \mid \forall i \in \mathcal{N} \setminus \{d\},\ \forall t \in \mathcal{H}\}$

Determine the minimum possible travel time paths for all $(i,t) \in S$

Compute the expected travel times $e_i^*(t)$ on the minimum possible travel time paths

$e_i^U(t) = e_i^*(t), \quad \forall (i,t) \in S$

$k \leftarrow 1$

Step 2: Branching

$k \leftarrow k+1$

$\lambda_i^k(t) \leftarrow$ the minimum travel time on the k^{th} minimum possible travel time path, $\forall (i,t) \in S$

Step 3: Bound Checking and Pruning

For $(i,t) \in S$ **do**

If $e_i^U(t) \leq \lambda_i^k(t)$ **then**

The path associated with $e_i^U(t)$ is the minimum expected travel time path from node i to node d for departure time t

$\overline{e}_i(t) \leftarrow e_i^U(t)$

Remove (i,t) from S

Else

Compute the expected travel time $e_i^*(t)$ on the k^{th} minimum possible travel time path

If $e_i^U(t) > e_i^*(t)$ **then**

$e_i^U(t) \leftarrow e_i^*(t)$

End (If)

End (If)

End (For)

If $S = \emptyset$ **then** stop

Else go to **Step 2**

End (If)

found the minimum expected travel time path for (i,t), we remove (i,t) from set S.

If $\lambda_i^k(t)$ is smaller than $e_i^U(t)$, we compute the expected travel time on the new path. If this expected travel is smaller than $e_i^U(t)$, it replaces $e_i^U(t)$. We repeat Step 2 and Step 3 until set S is empty.

The main difficulty of this algorithm is how to determine the k^{th} minimum possible travel time path for each $(i,t) \in S$ in Step 2. We leave this as an open question.

Hall [12] proposes an algorithm, which combines a branch-and-bound technique and a k shortest paths problem, for finding the minimum expected travel time path. Although Algorithm METTP–B&B is inspired by Hall's algorithm, there are a couple of differences between the two algorithms.

Firstly, Algorithm METTP–B&B determines the minimum expected travel time paths from all nodes to a single destination node for all departure times, whereas Hall's algorithm determines the minimum expected travel time path from a single origin node to a single destination node for a given departure time. Secondly, compared to Hall's algorithm, Algorithm METTP–B&B uses a better lower bound ($\lambda_i^k(t)$) on the expected travel time of a path not yet examined. In Hall's algorithm, the minimum travel time on a path, which serves as a lower bound on the expected travel time of the path, is defined as

$$\sum_{(v,w)} \min_{t \in \mathcal{H}} \left\{ \min_{r \in \mathcal{R}_{vw}(t)} \{\tau_{vw}^r(t)\} \right\}, \tag{6.15}$$

where link (v,w) belongs to the path. This indicates that Hall's algorithm does not consider time-dependency of the link travel times when it computes the minimum travel time on a path. The value obtained by (6.15) is clearly smaller than or equal to the minimum travel time computed in Algorithm METTP–B&B. Since a higher value of the minimum travel time, i.e. a higer lower bound on the expected travel time on a path not yet considered, is used in Algorithm METTP–B&B, Algorithm METTP–B&B is able to prune "branches" earlier than Hall's algorithm. This would enable the practical running time of Algorithm METTP–B&B to be faster than that of Hall's algorithm.

6.3.3 Algorithm EV

Miller-Hooks [16] and Miller-Hooks and Mahmassani [18] propose a label-correcting algorithm called *Algorithm EV* for the all-to-one minimum expected travel time paths problem.

The worst-case running time complexity of Algorithm EV is known to be non-polynomial (Miller-Hooks and Mahmassani [18]). We refer the reader to Miller-Hooks and Mahmassani [18] for the details of the algorithm.

6.3.4 Algorithm METTP

Algorithm METTP described in Algorithm 15 is a decreasing order of departure time algorithm for the all-to-one minimum expected travel time paths problem. As mentioned before, decreasing order of departure time algorithms do not guarantee to find the minimum expected travel time paths correctly unless all paths are examined at each departure time. In Algorithm METTP, for departure times close to H, some paths may not be taken into consideration. Hence this algorithm is a heuristic.

$\omega_i^c(t)$ denotes the expected travel time on path c from node i to node d at departure time t. α_i^c denotes the successor of node i on path c from node i to node d. β_i^c indicates the index of the path from node α_i^c to node d, which is a subpath of path c from node i to node d. M is a number that is bigger than the number of paths between any node and the destination node. Set P_i contains the indexes of paths from node i to node d discovered during the execution of the algorithm.

In Step 2, we solve an all-to-one static shortest paths problem to compute $\overline{e}_i(H)$. By solving this problem, we have found a path from each node to node d. We keep the information of this path (Lines 8–10).

In Step 3, we compute $\overline{e}_i(t)$ in a decreasing order of departure time. In Line 16, we check whether path $i \rightarrow j \overset{c}{\rightsquigarrow} d$ is already discovered. If so, we compute the expected travel time on the path at the current departure time and store it in $\omega_i^q(t)$, where q is the index indicating the path. If not, path $i \rightarrow j \overset{c}{\rightsquigarrow} d$ is a newly constructed path. We check if the path contains a cycle. Note that this is the same as checking if path $j \overset{c}{\rightsquigarrow} d$ visits node i. If the path contains a cycle, we ignore the path. If it does not contain a cycle, we create an index for the path and store the index in P_i (Lines 20–21). We compute the expected travel time on the path at the current departure time (Line 22) and keep the information of the path (Line 23).

Once we consider all $j \in \mathcal{O}(i)$, we determine $\overline{e}_i(t)$, $s_i(t)$, and $\kappa_i(t)$ by Lines 27–28. Notice that at this point, there may exist some paths that are not examined from node i to node d. To elaborate on this, consider the network in Figure 6-2.

Algorithm 15 Algorithm METTP

Step 1: Initialization

$(\overline{e}_i(t), s_i(t), \kappa_i(t)) \leftarrow (\infty, \infty, \infty), \quad \forall i \neq d, \ \forall t \in \mathcal{H}$

$(\overline{e}_d(t), s_d(t), \kappa_d(t)) \leftarrow (0, d, 1), \quad \forall t \in \mathcal{H}$

$\omega_i^c(t) \leftarrow \infty, \quad \forall i \neq d, \ \forall t \in \mathcal{H}, \ \forall c = 1, 2, \ldots, M$

$(\alpha_i^c, \beta_i^c) \leftarrow (\infty, \infty), \quad \forall i \neq d, \ \forall c = 1, 2, \ldots, M$

Step 2: Minimum Expected Travel Time Paths in Static Domain

$(\overline{e}_i(H), s_i(H)) \leftarrow$ All-to-One SP$(\mathcal{N}, \mathcal{A}, \{E[T_{ij}(H)]\}, d)$

$\omega_i^1(H) \leftarrow \overline{e}_i(H), \quad \forall i \in \mathcal{N}$

$(\alpha_i^1, \beta_i^1) \leftarrow (s_i(H), 1), \quad \forall i \in \mathcal{N}$

$P_i \leftarrow \{1\}, \quad \forall i \in \mathcal{N}$

Step 3: Minimum Expected Travel Time Paths in Time-Dependent Domain

For $t \leftarrow H - 1$ down to 0 **do**

  For $i \in \mathcal{N} \setminus \{d\}$ **do**

    For $j \in \mathcal{O}(i)$ **do**

      For $c \in P_j$ **do**

        If P_i has the index indicating path $i \rightarrow j \overset{c}{\rightsquigarrow} d$ **then**

          let $q \in P_i$ be the index

          $\omega_i^q(t) \leftarrow \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + \omega_j^c(t + \tau_{ij}^r(t))\right) p_{ij}^r(t)$

        Else If path $i \rightarrow j \overset{c}{\rightsquigarrow} d$ does not contain a cycle **then**

          $q \leftarrow |P_i| + 1$

          insert q into P_i

          $\omega_i^q(t) \leftarrow \sum_{r \in \mathcal{R}_{ij}(t)} \left(\tau_{ij}^r(t) + \omega_j^c(t + \tau_{ij}^r(t))\right) p_{ij}^r(t)$

          $\alpha_i^q \leftarrow j, \quad \beta_i^q \leftarrow c$

        End (If)

      End (For)

    End (For)

    $z = \arg\min_{c \in P_i} \{\omega_i^c(t)\}$

    $\overline{e}_i(t) = \omega_i^z(t), \quad s_i(t) = \alpha_i^z, \quad \kappa_i(t) = \beta_i^z$

  End (For)

End (For)

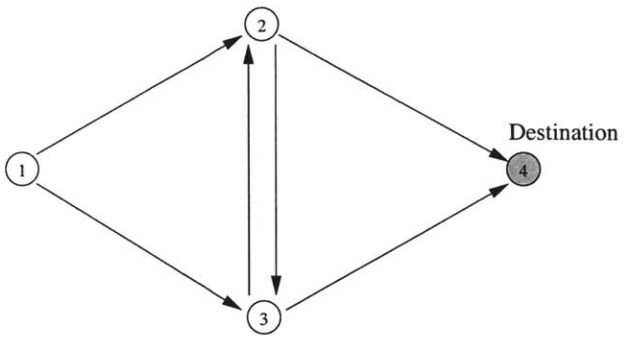

Figure 6-2: An Example Network

Suppose that paths $1 \rightarrow 2 \rightarrow 4$, $2 \rightarrow 4$, and $3 \rightarrow 4$ are identified as shortest paths in Step 2. Let node 1 be the first node we select at Line 13 when departure time $t = H - 1$. Then the algorithm finds a new path $1 \rightarrow 3 \rightarrow 4$, but does not construct paths $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ and $1 \rightarrow 3 \rightarrow 2 \rightarrow 4$ at departure time $t = H - 1$. Therefore $\overline{e}_1(H-1)$ determined at Line 27 in this case could not be the minimum expected travel time from node 1 to node 4 for departure time $H - 1$.

In fact, unexamined paths at a certain departure time depend on the order of selection of nodes (or the order of selection of links) at Lines 13–14 of the algorithm. If node 2 and node 3 are selected before node 1, all paths from node 1 to node 4 will be considered at departure time $t = H - 1$. Note that as departure time t decreases, the number of unexamined paths at each departure time gets smaller.

We may repeat the For loop of Lines 13–29 for each departure time to ensure that all paths are examined. However, this results in enumeration of all paths in the network.

Chapter 7

Computational Tests

In this chapter, we report on the results of computational tests on several algorithm presented in the preceding chapters. All computational tests were conducted on randomly generated stochastic time-dependent networks. We first describe the methodology for generating networks and then show the computational test results.

7.1 Generation of Stochastic Time-Dependent Networks

The random network generator we developed to create stochastic time-dependent networks requires the following eight parameters: *number of nodes* (n), *number of links* (m), *maximum indegree of a node* (MAX_IN), *maximum outdegree of a node* (MAX_OUT), *number of discrete departure times* (H), *number of link travel time realizations* (R), *minimum possible link travel time value*, and *maximum possible link travel time value*.

The first four parameters determine the topology of a network, so we may produce dense or sparse networks by choosing appropriate values of these parameters. The rest of the parameters are used for constructing time-dependent link travel time probability mass functions.

For various all-to-one routing problems addressed in this thesis, we need networks with a single destination node where connectivity from every node to the destination node is ensured. To do so, the random network generator first constructs a tree where a directed path from every node to the destination node exists. This tree consists of $n - 1$ links. Once the tree is constructed, the random network generator randomly adds $m - (n - 1)$ links to the tree to build a network of m links. During this procedure, the indegree and

the outdegree of a node are kept no greater than MAX_IN and MAX_OUT respectively. The random network generator also makes sure that no parallel links are introduced between any two nodes throughout the procedure. The following are the detailed steps by which the random network generator creates a network with a given destination node d.

1. Create two sets, S_1 and S_2.
2. $S_1 \leftarrow \mathcal{N} \setminus \{d\}, \quad S_2 \leftarrow \{d\}$.
3. Randomly select a node (called i) from S_1 and a node (called j) from S_2.
4. If the outdegree of node $i <$ MAX_OUT and the indegree of node $j <$ MAX_IN, then create link (i, j). Remove node i from S_1 and insert it into S_2.
5. Repeat Step 3 and Step 4 until S_1 becomes empty.
6. Randomly select two nodes (called i and j) from S_2.
7. If link (i, j) does not exist and the outdegree of node $i <$ MAX_OUT and the indegree of node $j <$ MAX_IN, then create link (i, j).
8. Repeat Step 6 and Step 7 until we obtain m link in total.

In order to create networks that are topologically similar to real transportation networks in urban areas, the number of links is assumed to be four times the number of nodes. We set the maximum indegree and the maximum outdegree of a node both to 5. By doing so, we allow nodes to have no more than five incoming or outgoing links, which is a topological property generally observed in real transportation networks.

After the random network generator creates a network, it establishes a probability mass function of the travel time on each link for each departure time. The random network generator first randomly selects R distinct integer travel time values, i.e. $\tau_{ij}^r(t),\ r = 1, \ldots, R$, from the range of a link travel time value which is given by the maximum possible link travel time value $-$ the minimum possible link travel time value $+$ 1. Then R fractional values are randomly chosen from the range of $(0, 1)$. In order to convert the R fractional values into valid probabilities associated with the R link travel time values already generated, we normalize the R fractional values so that their sum equals to 1. The normalized R fractional values serve as $p_{ij}^r(t),\ r = 1, \ldots, R$. Note that we let all link travel time random

variables have the same number of realizations in the computational tests. Also note that the range of a link travel time value should be greater than or equal to R. Otherwise, we cannot generate R distinct link travel time values.

7.2 Computational Test Results

Algorithms for the following problems were implemented and computationally tested:

- All-to-One Minimum Possible Travel Time Paths Problem
- All-to-One $k-$Minimum Path Travel Time Realizations Problem
- All-to-One $k-$Dynamic Shortest Paths Problem
- All-to-One Minimum Expected Travel Time Next-Arc Hyperpaths Problem

The algorithms were written in C++. The source codes of the algorithms are available for the interested readers. Running times were obtained using a Sun Blade 100 workstation with a 500MHz CPU and 512 MB RAM. All running times are in seconds and do not include input/output times.

7.2.1 All-to-One Minimum Possible Travel Time Paths Problem

We implemented Algorithm MPTTP2 and Algorithm LEAST to compare their practical running times. We considered three levels of the number of nodes (1000, 2000, 3000), three levels of the number of discrete departure times (30, 60, 90), and two levels of the number of link travel time realizations (5, 10). We set the minimum possible link travel time value and the maximum possible link travel time value to 1 and 15, respectively.

For each combination of n, H, and R, we randomly generated five different stochastic time-dependent networks. We obtained the running times of each algorithm for those five networks and computed the average of the running times. Tables 7.1–7.3 show the average running times for different combinations of n, H, and R.

The computational test results show that the running time of Algorithm MPTTP2 is 1.6 – 2.7 times faster than that of Algorithm LEAST. The ratio of the running times (Algorithm LEAST/Algorithm MPTTP2) becomes larger as the topological size of a network

increases, i.e. as n increases. This implies that Algorithm MPTTP2 will be much faster than Algorithm LEAST for larger networks.

It is interesting to observe that for fixed values of n and H, the running time of Algorithm LEAST may not monotonically increase as the number of link travel time realizations increases. Table 7.3 shows this case. The running time of Algorithm MPTTP2 is, however, always an increasing function of any of the parameters n, H, and R.

Table 7.1: Running Times of Algorithm LEAST and Algorithm MPTTP2 as a Function of H and R When $n = 1000$ and $m = 4000$

H	R	Algorithm LEAST	Algorithm MPTTP2	LEAST/MPTTP2
30	5	0.94	0.44	2.15
	10	1.07	0.65	1.66
60	5	1.86	0.83	2.25
	10	2.16	1.27	1.70
90	5	2.79	1.25	2.23
	10	3.21	1.92	1.67

Table 7.2: Running Times of Algorithm LEAST and Algorithm MPTTP2 as a Function of H and R When $n = 2000$ and $m = 8000$

H	R	Algorithm LEAST	Algorithm MPTTP2	LEAST/MPTTP2
30	5	2.42	0.99	2.44
	10	2.43	1.40	1.74
60	5	4.49	1.83	2.45
	10	4.75	2.68	1.77
90	5	6.49	2.71	2.39
	10	7.04	4.01	1.75

7.2.2 All-to-One $k-$Minimum Path Travel Time Realizations Problem

For the all-to-one $k-$minimum path travel time realizations problem, Algorithm $k-$MPTTR and the algorithm proposed in Miller-Hooks and Mahmassani [17] (we call it Algorithm $k-$LEAST hereafter) were implemented.

We obtained running times of the two algorithms for three levels of the number of nodes (200, 600, 1000), three levels of the number of discrete departure times (30, 60, 90), two levels of the number of link travel time realizations (5, 10), and two levels of the k value

Table 7.3: Running Times of Algorithm LEAST and Algorithm MPTTP2 as a Function of H and R When $n = 3000$ and $m = 12000$

H	R	Algorithm LEAST	Algorithm MPTTP2	LEAST/MPTTP2
30	5	4.63	1.69	2.75
	10	4.23	2.24	1.89
60	5	8.01	2.94	2.72
	10	7.76	4.19	1.85
90	5	11.13	4.28	2.60
	10	11.37	6.20	1.83

(5, 10). The minimum possible link travel time value and the maximum possible link travel time value are respectively set to 1 and 15 as before.

Since we should maintain k label couples, $(\lambda_i^n(t), \gamma_i^n(t))$, $n = 1, \ldots, k$, for each node-departure time pair, much more memory is needed to keep labels in this problem than in the all-to-one minimum possible travel time paths problem. This compels us to consider topologically smaller networks, i.e. networks with smaller number of nodes, than those used in the all-to-one minimum possible travel time paths problem in order to avoid running out of memory (RAM).

In the workstation we used, disk swapping resulting from running out of memory happened very often for networks with more than 1000 nodes, and it increased the running times of the algorithms enormously. This is the reason that we considered networks with no more than 1000 nodes.

For each combination of n, H, R, and k, we randomly generated five different stochastic time-dependent networks. The average running time of each algorithm obtained from these five different networks is shown in Tables 7.4–7.9.

It turns out that the running time of Algorithm $k-$MPTTR is 1.3 – 2.2 times faster than that of Algorithm $k-$LEAST. For a given H, R, and k, it is observed that the ratio of the running times is almost constant regardless of the number of nodes. We also observe that the running time ratio decreases as the k value increases for a fixed n, H, and R.

7.2.3 All-to-One $k-$Dynamic Shortest Paths Problem

We implemented Algorithm $k-$DSP discussed in Chapter 4. Since no solution algorithm for the all-to-one $k-$dynamic shortest paths problem has been published, we modified Al-

Table 7.4: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 200$, $m = 800$, and $k = 5$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	1.58	0.82	1.92
	10	2.25	1.60	1.40
60	5	3.32	1.62	2.05
	10	4.66	3.18	1.46
90	5	5.18	2.43	2.13
	10	7.18	4.78	1.50

Table 7.5: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 200$, $m = 800$, and $k = 10$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	5.02	2.79	1.80
	10	7.77	5.66	1.37
60	5	10.40	5.48	1.90
	10	15.96	11.25	1.42
90	5	15.77	8.15	1.94
	10	24.24	16.67	1.45

Table 7.6: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 600$, $m = 2400$, and $k = 5$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	4.73	2.55	1.85
	10	7.04	4.91	1.43
60	5	10.34	5.03	2.06
	10	14.13	9.73	1.45
90	5	18.04	8.21	2.20
	10	22.58	14.92	1.51

Table 7.7: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 600$, $m = 2400$, and $k = 10$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	14.87	8.49	1.75
	10	23.30	17.19	1.36
60	5	32.01	16.74	1.91
	10	49.74	36.72	1.35
90	5	49.22	25.00	1.97
	10	73.75	51.36	1.44

Table 7.8: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 1000$, $m = 4000$, and $k = 5$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	8.03	4.27	1.88
	10	11.59	8.25	1.40
60	5	17.71	8.44	2.10
	10	23.84	16.86	1.41
90	5	29.20	13.68	2.13
	10	36.75	24.36	1.51

Table 7.9: Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR as a Function of H and R When $n = 1000$, $m = 4000$, and $k = 10$

H	R	Algorithm $k-$LEAST	Algorithm $k-$MPTTR	$k-$LEAST/$k-$MPTTR
30	5	25.11	14.29	1.76
	10	41.04	31.00	1.32
60	5	54.02	28.00	1.93
	10	81.87	59.50	1.38
90	5	85.92	42.69	2.01
	10	122.79	86.86	1.41

gorithm k−LEAST so that it solves the all-to-one k−dynamic shortest paths problem. We refer to the modified algorithm as Algorithm k−D-LEAST. We compare the practical running time of Algorithm k−D-LEAST with that of Algorithm k−DSP.

The procedure to generate deterministic time-dependent networks with a single destination node is identical to the procedure described in Section 7.1 except that only one link travel time value is generated for each link and each departure time.

Three levels of the number of nodes (1000, 2000, 3000), three levels of the number of discrete departure times (30, 60, 90), and two levels of the k value (5, 10) were considered. For each combination of n, H, and k, we randomly generated five different deterministic time-dependent networks. The average running times are shown in Tables 7.10–7.15, indicating that Algorithm k−DSP is 2.7 - 3.9 times faster than Algorithm k−D-LEAST.

Table 7.10: Running Times of Algorithm k−D-LEAST and Algorithm k−DSP as a Function of H When $n = 1000$, $m = 4000$, and $k = 5$

H	Algorithm k−D-LEAST	Algorithm k−DSP	k−D-LEAST/k−DSP
30	2.14	0.76	2.83
60	5.07	1.50	3.39
90	8.29	2.22	3.73

Table 7.11: Running Times of Algorithm k−D-LEAST and Algorithm k−DSP as a Function of H When $n = 1000$, $m = 4000$, and $k = 10$

H	Algorithm k−D-LEAST	Algorithm k−DSP	k−D-LEAST/k−DSP
30	5.73	1.93	2.96
60	13.09	3.68	3.56
90	20.71	5.48	3.78

Table 7.12: Running Times of Algorithm k−D-LEAST and Algorithm k−DSP as a Function of H When $n = 2000$, $m = 8000$, and $k = 5$

H	Algorithm k−D-LEAST	Algorithm k−DSP	k−D-LEAST/k−DSP
30	4.49	1.61	2.79
60	10.53	3.10	3.39
90	16.94	4.65	3.65

Table 7.13: Running Times of Algorithm $k-$D-LEAST and Algorithm $k-$DSP as a Function of H When $n = 2000$, $m = 8000$, and $k = 10$

H	Algorithm $k-$D-LEAST	Algorithm $k-$DSP	$k-$D-LEAST/$k-$DSP
30	11.39	3.94	2.89
60	26.95	7.54	3.57
90	43.20	11.29	3.83

Table 7.14: Running Times of Algorithm $k-$D-LEAST and Algorithm $k-$DSP as a Function of H When $n = 3000$, $m = 12000$, and $k = 5$

H	Algorithm $k-$D-LEAST	Algorithm $k-$DSP	$k-$D-LEAST/$k-$DSP
30	6.73	2.46	2.73
60	16.45	4.77	3.45
90	35.93	9.91	3.63

Table 7.15: Running Times of Algorithm $k-$D-LEAST and Algorithm $k-$DSP as a Function of H When $n = 3000$, $m = 12000$, and $k = 10$

H	Algorithm $k-$D-LEAST	Algorithm $k-$DSP	$k-$D-LEAST/$k-$DSP
30	17.85	6.01	2.97
60	41.51	11.42	3.64
90	65.22	16.91	3.86

7.2.4 All-to-One Minimum Expected Travel Time Next-Arc Hyperpaths Problem

For this problem, Algorithm METTH and Algorithm ELB were computationally tested. The same levels of the number of nodes, of the number of discrete departure times, and of the number of link travel time realizations as in the all-to-one minimum possible travel time paths problem were considered. For each combination of n, H, and R, the average running times of the two algorithms were computed from five different randomly generated networks. The results are summarized in Tables 7.16–7.18.

The running time ratios for Algorithm ELB and Algorithm METTH range between 1.0 and 1.5, which is significantly smaller compared to the running time ratios for Algorithm LEAST and Algorithm MPTTP2 for the all-to-one minimum possible travel time paths problem. Considering that Algorithm METTH and Algorithm ELB have the same worst-case theoretical running time complexities as Algorithm MPTTP2 and Algorithm LEAST respectively, this appears to be a surprising result.

However there is a reason for this result. The running times of Algorithm LEAST and Algorithm ELB both mainly depend on the total number of times that nodes are inserted into the "scan eligible list" S (We used a queue data structure for S. Other data structures are also viable for implementing S). In Algorithm LEAST, a node is inserted into S whenever a better path travel time realization, which has a smaller travel time value or the same travel time value with a higher probability than the current path travel time realization from the node for any departure time, is found. In Algorithm ELB, a node is inserted into S whenever a better next-arc hyperpath, which has a lower expected travel time than the current next-arc hyperpath from the node for any departure time, is obtained.

This means that Algorithm LEAST examines all path travel time realizations in a stochastic time-dependent network under consideration, while Algorithm ELB examines all next-arc hyperpaths in the network. In general, the number of next-arc hyperpaths in a stochastic time-dependent network is considerably smaller than the number of path travel time realizations in the same network. Hence the total number of times that nodes are inserted into S in Algorithm ELB would be much smaller than that of Algorithm LEAST. This explains that the practical running time of Algorithm ELB is much better that that of Algorithm LEAST although their worst-case running time complexities are the same.

Algorithm METTH also results in slightly better running times compared to Algorithm MPTTP2 despite of the fact that they have the same optimal (lowest possible) running time complexity. We think that this is because the hidden constant associated with the running time complexity of Algorithm METTH is smaller than that of Algorithm MPTTP2.

Table 7.16: Running Times of Algorithm ELB and Algorithm METTH as a Function of H and R When $n = 1000$ and $m = 4000$

H	R	Algorithm ELB	Algorithm METTH	ELB/METTH
30	5	0.41	0.31	1.33
	10	0.52	0.48	1.09
60	5	0.79	0.64	1.25
	10	0.97	0.96	1.01
90	5	1.17	0.95	1.24
	10	1.48	1.45	1.02

Table 7.17: Running Times of Algorithm ELB and Algorithm METTH as a Function of H and R When $n = 2000$ and $m = 8000$

H	R	Algorithm ELB	Algorithm METTH	ELB/METTH
30	5	1.00	0.71	1.41
	10	1.17	1.05	1.11
60	5	1.82	1.34	1.35
	10	2.11	2.02	1.04
90	5	2.64	2.04	1.29
	10	3.01	3.00	1.00

7.3 Concluding Remarks

We have seen that several algorithms developed in this thesis are faster than the existing algorithms in the literature in practice through the computational tests. These computational test results are consistent with the theoretical running time analysis results presented in the previous chapters. Therefore we can conclude that our algorithms are more efficient than the existing ones both in theory and in practice.

In addition to having better running times, our algorithms are also more robust than the existing ones in the sense that the running times of our algorithms are affected by neither the network topology nor the link travel time data.

Table 7.18: Running Times of Algorithm ELB and Algorithm METTH as a Function of H and R When $n = 3000$ and $m = 12000$

H	R	Algorithm ELB	Algorithm METTH	ELB/METTH
30	5	1.83	1.19	1.54
	10	2.02	1.68	1.20
60	5	3.17	2.18	1.46
	10	3.44	3.18	1.08
90	5	4.33	3.20	1.35
	10	4.81	4.68	1.03

For given network parameters, each of our algorithms has a constant running time complexity that does not vary with the shape of a network or with the link travel time data. On the contrary, the running time of each of the existing algorithms (modified label-correcting algorithms) may change significantly depending on the topological shape of a network and/or on the link travel time data even with the same network parameters. This is because certain network topology and/or certain link travel time data may make label-correcting algorithms update labels more frequently.

To support this numerically, we compute the variance of each algorithm's running times obtained from the five different randomly generated networks for given network parameters. Tables 7.19–7.22 show those variances for selected network parameters. Our algorithms result in considerably small running time variances compared to the existing algorithms.

Finally it should be noted that the networks used in the computational tests are relatively sparse. If the networks become denser, our algorithms would outperform the existing algorithms more significantly in practice.

Table 7.19: Variances of Running Times of Algorithm LEAST and Algorithm MPTTP2 When $n = 3000$, $m = 12000$, $H = 60$, and $R = 5$

Network No.	Algorithm LEAST	Algorithm MPTTP2
1	8.75	2.93
2	7.90	2.97
3	7.78	2.98
4	7.74	2.89
5	7.88	2.95
Mean	8.01	2.94
Variance	0.140	0.0010

Table 7.20: Variances of Running Times of Algorithm $k-$LEAST and Algorithm $k-$MPTTR When $n = 1000$, $m = 4000$, $H = 60$, $R = 5$, and $k = 10$

Network No.	Algorithm $k-$LEAST	Algorithm $k-$MPTTR
1	54.25	27.90
2	53.88	28.05
3	52.97	28.01
4	54.65	28.04
5	54.35	28.02
Mean	54.02	28.00
Variance	0.336	0.0029

Table 7.21: Variances of Running Times of Algorithm $k-$D-LEAST and Algorithm $k-$DSP When $n = 3000$, $m = 12000$, $H = 60$, and $k = 10$

Network No.	Algorithm $k-$D-LEAST	Algorithm $k-$DSP
1	42.70	11.47
2	41.01	11.40
3	41.52	11.43
4	41.48	11.42
5	40.83	11.37
Mean	41.51	11.42
Variance	0.426	0.0011

Table 7.22: Variances of Running Times of Algorithm ELB and Algorithm METTH When $n = 3000$, $m = 12000$, $H = 60$, and $R = 5$

Network No.	Algorithm ELB	Algorithm METTH
1	3.14	2.18
2	3.12	2.18
3	3.04	2.17
4	3.22	2.19
5	3.35	2.18
Mean	3.17	2.18
Variance	0.011	4e-05

Chapter 8

Conclusions and Future Research Directions

8.1 Conclusions

In this thesis, we studied routing problems in stochastic time-dependent networks where link travel times are modeled as time-dependent discrete random variables. Although the consideration of both stochasticity and time-dependency of link travel times makes routing problems hard to solve, it gives better routing solutions in real transportation or communication networks.

Among various routing problems that arise in stochastic time-dependent networks, we focused on three classes of routing problems: the minimum possible travel time path problem, the minimum expected travel time next-arc hyperpath problem, and the minimum expected travel time path problem.

8.1.1 Minimum Possible Travel Time Path Problem

The objective of this class of routing problems is to find a path that has the minimum possible travel time. We studied the all-to-one minimum possible travel time paths problem in detail. This problem was extended to the all-to-one minimum possible travel cost paths problem, the all-to-one $k-$minimum path travel time realizations problem, and the all-to-one $k-$dynamic shortest paths problem.

We developed an efficient solution algorithm for each problem. Especially, the algorithm

for the all-to-one minimum possible travel time paths problem (Algorithm MPTTP) and the algorithm for the all-to-one minimum possible travel cost paths problem (Algorithm MPTCP) were proved to be optimal in the running time sense. We also observed that the algorithm for the all-to-one $k-$minimum path travel time realizations problem (Algorithm $k-$MPTTR) had a better worst-case running time complexity than the existing algorithm. It seems that Algorithm $k-$DSP is the first algorithm proposed for the all-to-one $k-$dynamic shortest paths problem, to the best of the author's knowledge.

8.1.2 Minimum Expected Travel Time Next-Arc Hyperpath Problem

In this class of routing problems, a next-arc hyperpath (a routing strategy) that results in the minimum expected travel time is sought when travelers are allowed to change their paths en route. This class of routing problems is important for route guidance systems within the context of ATIS.

We studied the all-to-one minimum expected travel time next-arc hyperpaths problem and developed an optimal running time solution algorithm (Algorithm METTH). By extending Algorithm METTH, we also developed an optimal running time solution algorithm for the all-to-one minimum expected travel cost next-arc hyperpaths problem (Algorithm METCH).

The all-to-one minimum expected travel time next-arc hyperpaths problems in signalized networks and in multimodal networks were investigated. We showed that the algorithms (Algorithm METTH–Signal and Algorithm METTH–Multimodal) developed for these two problems in this thesis had better worst-case running time complexities than the known algorithms in the literature.

8.1.3 Minimum Expected Travel Time Path Problem

The aim of this class of routing problems is to find a simple path that has the minimum expected travel time. We discussed the all-to-one minimum expected travel time paths problem. It turned out that this problem could not be solved efficiently. We provided a reason for that.

Single pass decreasing order of departure time-type solution algorithms, which worked for the routing problems mentioned in Sections 8.1.1 and 8.1.2, were not applicable to this

problem. Hence we presented ideas of alternative solution algorithms (Algorithm METTH–based–METTP, Algorithm METTP-B&B).

8.1.4 Results of Computational Tests

Computational tests were carried out to compare the practical running times of several algorithms developed in this thesis with those of the existing algorithms in the literature. The results showed that our algorithms outperformed the existing algorithms in practice as well. We, therefore, believe that our algorithms are useful especially for many real time routing applications in transportation or telecommunication area.

8.2 Future Research Directions

The following are the tasks left unanswered or unfinished in this thesis. They could be done in the short or the medium term.

1. For routing problems in the minimum possible travel time path problem class, we assumed that link travel time random variables were independent of each other. This assumption is hardly satisfied in congested transportation networks. Relaxation of this assumption should be explored.

2. Computational tests on Algorithm METTH–Signal and Algorithm METTH–Multimodal should be carried out to support the fact that they have theoretically better worst-case running time complexities than the existing algorithms.

3. Computational tests on Algorithm METTH–based–METTP should be performed.

4. In Algorithm METTH–B&B, we left how to determine the k^{th} minimum possible travel time path as an open question. This question should be answered, and computational tests on the algorithm should be done.

As further extensions of the work done in this thesis, we propose the following topics for the long term research.

1. We studied all-to-one versions of routing problems in the thesis. Other variants such as one-to-all routing problems, all-to-all routing problems, etc could be considered.

2. Other criteria for route selection, for instance, minimum variance, lowest probability of being longest, highest probability of being shortest, could be studied.

3. The formulation of various routing problems in stochastic time-dependent networks by using hyperpaths on suitably constructed hypergraphs could be an important research topic.

Appendix A

Notation

In this appendix, we summarize the notation used in the thesis. For each notation, we provide the chapter where it was first introduced in parentheses.

A.1 Network-related Notation

- $\mathcal{G}(\mathcal{N}, \mathcal{A})$: a directed graph (network) (Chapter 3)
- $\mathcal{N}$: set of n nodes (Chapter 3)
- $\mathcal{A}$: set of m directed links (Chapter 3)
- d: destination node (Chapter 3)
- $\mathcal{O}(i)$: set of end nodes of outgoing links from node i (Chapter 3)
- $\mathcal{I}(i)$: set of start nodes of incoming links to node i (Chapter 3)
- $\mathcal{H}$: set of discrete departure times (Chapter 3)
- $\widehat{\mathcal{N}}$: set of nodes in the time-space network of a deterministic time-dependent network (Chapter 3)
- $\widehat{\mathcal{A}}$: set of links in the time-space network of a deterministic time-dependent network (Chapter 3)
- $\widehat{\mathcal{O}}((i,t))$: set of end nodes of outgoing links from node (i,t) in the time-space network of a deterministic time-dependent network (Chapter 3)

- $\widetilde{\mathcal{N}}$: set of nodes in the time-space network of a stochastic time-dependent network (Chapter 3)
- $\widetilde{\mathcal{A}}$: set of links in the time-space network of a stochastic time-dependent network (Chapter 3)
- $\widetilde{\mathcal{O}}((i,t))$: set of end nodes of outgoing links from node (i,t) in the time-space network of a stochastic time-dependent network (Chapter 3)
- $\mathcal{M} = \{1, \cdots, M\}$: set of travel modes available in the network (Chapter 5)
- $\mathcal{M}_{ij} \subseteq \mathcal{M}$: set of modes available on link (i,j) (Chapter 5)

A.2 Random Variable-related Notation

- $T_{ij}(t)$: random variable denoting the travel time on link (i,j) at link entry time t (Chapter 3)
- $\mathcal{R}_{ij}(t) = \{1, \ldots, r_{ij}(t)\}$: set of indexes of realizations of $T_{ij}(t)$ (Chapter 3)
- $\tau_{ij}^r(t)$: travel time value of the r^{th} realization of $T_{ij}(t)$ (Chapter 3)
- $p_{ij}^r(t)$: probability of the r^{th} realization of $T_{ij}(t)$ (Chapter 3)
- $p_{T_{ij}(t)} = \left\{(\tau_{ij}^r(t),\, p_{ij}^r(t)) \mid r \in \mathcal{R}_{ij}(t)\right\}$: PMF of $T_{ij}(t)$ (Chapter 3)
- $L_i^c(t)$: random variable denoting the travel time on path c from node i to node d at departure time t (Chapter 3)
- $l_i^{c^k}(t)$: travel time value of the k^{th} realization of $L_i^c(t)$ (Chapter 3)
- $p_i^{c^k}(t)$: probability of the k^{th} realization of $L_i^c(t)$ (Chapter 3)
- $p_{L_i^c(t)} = \left\{(l_i^{c^k}(t),\, p_i^{c^k}(t)) \mid k = 1, 2, \ldots, k_i^c(t)\right\}$: PMF of $L_i^c(t)$ (Chapter 3)
- $\overline{T}_{ij}^c(t)$: random variable denoting the travel time from node i to node d when one leaves node i at time t via link (i,j) and then follows path c from node j to node d (Chapter 4)
- $C_{ij}(t)$: random variable denoting the travel cost on link (i,j) at link entry time t (Chapter 4)

- $\mathcal{X}_{ij}(t) = \{1, \ldots, x_{ij}(t)\}$: set of indexes of realizations of $C_{ij}(t)$ (Chapter 4)
- $\zeta_{ij}^{x}(t)$: travel cost value of the x^{th} realization of $C_{ij}(t)$ (Chapter 4)
- $g_{ij}^{x}(t)$: probability of the x^{th} realization of $C_{ij}(t)$ (Chapter 4)
- $p_{C_{ij}(t)} = \left\{(\zeta_{ij}^{x}(t),\, g_{ij}^{x}(t)) \mid x \in \mathcal{X}_{ij}(t)\right\}$: PMF of $C_{ij}(t)$ (Chapter 4)
- $\overline{C}_{ij}^{c}(t)$: random variable denoting the travel cost from node i to node d when one leaves node i at time t via link (i, j) and then follows path c from node j to node d (Chapter 4)
- $Y_j^c(t+\tau_{ij}^r(t))$: random variable denoting the travel cost on path c from node j to node d at departure time $t + \tau_{ij}^r(t)$ (Chapter 4)
- $y_j^{c^k}(t + \tau_{ij}^r(t))$: travel cost value of the k^{th} realization of $Y_j^c(t + \tau_{ij}^r(t))$ (Chapter 4)
- $g_j^{c^k}(t + \tau_{ij}^r(t))$: probability of the k^{th} realization of $Y_j^c(t + \tau_{ij}^r(t))$ (Chapter 4)
- $\overline{T}_{ij}(t)$: random variable denoting the travel time from node i to node d when one leaves node i at time t via link (i, j) and then takes an optimal next-arc hyperpath from node j to node d (Chapter 5)
- $L_j(t + \tau_{ij}^r(t))$: random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d at departure time $t + \tau_{ij}^r(t)$ (Chapter 5)
- $\overline{C}_{ij}(t)$: random variable denoting the travel cost from node i to node d when one leaves node i at time t via link (i, j) and then takes an optimal next-arc hyperpath from node j to node d (Chapter 5)
- $Y_j(t+\tau_{ij}^r(t))$: random variable denoting the travel cost on an optimal next-arc hyperpath from node j to node d at departure time $t + \tau_{ij}^r(t)$ (Chapter 5)
- $w_{hij}(t)$: penalty (the amount of waiting time) when one arrives at node i at time t from node h and then wants to take link (i, j) in a signalized network (Chapter 5)
- $\overline{T}_{hij}(t)$: random variable denoting the travel time from node i to node d when one arrives at node i at time t from node h and then takes link (i, j) and an optimal next-arc hyperpath from node j to node d in a signalized network (Chapter 5)

- $L_{ij}(t + w_{hij}(t) + \tau_{ij}^{r}(t + w_{hij}(t)))$: random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d when one arrives at node j at time $t + w_{hij}(t) + \tau_{ij}^{r}(t + w_{hij}(t))$ from node i in a signalized network (Chapter 5)

- $T_{ij}^{b}(t)$: random variable denoting the travel time on link (i, j) via mode b at link entry time t in a multimodal network (Chapter 5)

- $\mathcal{R}_{ij}^{b}(t) = \{1, \cdots, r_{ij}^{b}(t)\}$: set of indexes of realizations of $T_{ij}^{b}(t)$ (Chapter 5)

- $\tau_{ij}^{br}(t)$: travel time value of the r^{th} realization of $T_{ij}^{b}(t)$ (Chapter 5)

- $p_{ij}^{br}(t)$: probability of the r^{th} realization of $T_{ij}^{b}(t)$ (Chapter 5)

- $p_{T_{ij}^{b}(t)} = \left\{(\tau_{ij}^{br}(t),\, p_{ij}^{br}(t)) \mid r \in \mathcal{R}_{ij}^{b}(t)\right\}$: PMF of $T_{ij}^{b}(t)$ (Chapter 5)

- $D_{hij}^{ab}(t)$: random variable denoting the mode transfer delay when one arrives at node i at time t from node h via mode a and then takes mode b to go to node j in a multimodal network (Chapter 5)

- $\mathcal{R}_{hij}^{ab}(t) = \left\{1, \cdots, r_{hij}^{ab}(t)\right\}$: set of indexes of realizations of $D_{hij}^{ab}(t)$ (Chapter 5)

- $\delta_{hij}^{abs}(t)$: mode transfer delay value of the s^{th} realization of $D_{hij}^{ab}(t)$ (Chapter 5)

- $q_{hij}^{abs}(t)$: probability of the s^{th} realization of $D_{hij}^{ab}(t)$ (Chapter 5)

- $p_{D_{hij}^{ab}(t)} = \left\{(\delta_{hij}^{abs}(t),\, q_{hij}^{abs}(t)) \mid s \in \mathcal{R}_{hij}^{ab}(t)\right\}$: PMF of $D_{hij}^{ab}(t)$ (Chapter 5)

- $\overline{T}_{hij}^{ab}(t)$: random variable denoting the travel time from node i to node d when one arrives at node i at time t from node h via mode a and takes link (i, j) using mode b and then follows an optimal next-arc hyperpath from node j to node d in a multimodal network (Chapter 5)

- $L_{ij}^{b}(t + \delta_{hij}^{abs}(t) + \tau_{ij}^{br}(t + \delta_{hij}^{abs}(t)))$: random variable denoting the travel time on an optimal next-arc hyperpath from node j to node d when one arrives at node j at time $t + \delta_{hij}^{abs}(t) + \tau_{ij}^{br}(t + \delta_{hij}^{abs}(t))$ from node i via mode b in a multimodal network (Chapter 5)

A.3 Decision Variable-related Notation

- $\lambda_i(t)$: minimum possible travel time from node i to node d at departure time t (Chapter 4)

- $\gamma_i(t)$: highest probability that $\lambda_i(t)$ occurs among all minimum possible travel time realizations from node i to node d at departure time t (Chapter 4)

- $\xi_i(t)$: minimum possible travel cost from node i to node d at departure time t (Chapter 4)

- $\varphi_i(t)$: highest probability that $\xi_i(t)$ occurs among all minimum possible travel cost realizations from node i to node d at departure time t (Chapter 4)

- $\lambda_i^n(t)$: travel time value of the n^{th} minimum path travel time realization (or of the n^{th} shortest path) from node i to node d at departure time t (Chapter 4)

- $\gamma_i^n(t)$: probability of the n^{th} minimum path travel time realization from node i to node d at departure time t (Chapter 4)

- $e_i(t)$: minimum expected travel time from node i to node d at departure time t under the adaptive routing policy (Chapter 5)

- $c_i(t)$: minimum expected travel cost from node i to node d at departure time t under the adaptive routing policy (Chapter 5)

- $e_{hi}(t)$: minimum expected travel time from node i to node d under the adaptive routing policy when one arrives at node i at time t from node h in a signalized network (Chapter 5)

- $e_{hi}^a(t)$: minimum expected travel time from node i to node d under the adaptive routing policy when one arrives at node i at time t from node h via mode a in a multimodal network (Chapter 5)

- $\bar{e}_i(t)$: minimum expected travel time from node i to node d at departure time t under the non-adaptive routing policy (Chapter 6)

A.4 Miscellaneous Notation

- $s_i(t)$: successor of node i on the minimum possible travel time path (or on the minimum possible travel cost path, or on the minimum expected travel time next-arc hyperpath, or on the minimum expected travel cost next-arc hyperpath, or on the minimum expected travel time path) from node i to node d at departure time t (Chapter 4)

- $\pi_i(t)$: arrival time at node $s_i(t)$ on the minimum possible travel time path (or on the minimum possible travel cost path) from node i to node d at departure time t (Chapter 4)

- $s_i^n(t)$: successor of node i on the n^{th} minimum path travel time realization (or on the n^{th} shortest path) from node i to node d at departure time t (Chapter 4)

- $\pi_i^n(t)$: arrival time at node $s_i^n(t)$ on the n^{th} minimum path travel time realization (or on the n^{th} shortest path) from node i to node d at departure time t (Chapter 4)

- $\kappa_i^n(t)$: rank of the path travel time realization (or of the path) from node $s_i^n(t)$ to node d at departure time $\pi_i^n(t)$, which one should follow in order to achieve the n^{th} minimum path travel time realization (or the n^{th} shortest path) from node i to node d at departure time t (Chapter 4)

- $s_{hi}(t)$: successor of node i on the minimum expected travel time next-arc hyperpath from node i to node d when one arrives at node i at time t from node h in a signalized network (Chapter 5)

- $s_{hi}^a(t)$: successor of node i on the minimum expected travel time next-arc hyperpath from node i to node d when one arrives at node i at time t from node h via mode a in a multimodal network (Chapter 5)

- $m_{hi}^a(t)$: travel mode for link $(i, s_{hi}^a(t))$ on the minimum expected travel time next-arc hyperpath from node i to node d when one arrives at node i at time t from node h via mode a in a multimodal network (Chapter 5)

- $\kappa_i(t)$: index of the path from node $s_i(t)$ to node d that is a subpath of the minimum expected travel time path from node i at departure time t (Chapter 6)

- $\widetilde{m} = \sum_{(i,j)\in\mathcal{A}} \sum_{t\in\mathcal{H}} |\mathcal{R}_{ij}(t)|$: total number of link travel time realizations in the network during the peak period (Chapter 3)

- $\Theta(f(n,m))$: lowest possible running time to solve an all-to-one static shortest paths problem with n nodes and m links (Chapter 3)

- $\tau_{ij}(t)$: travel time on link (i,j) at link entry time t in deterministic time-dependent networks (Chapter 3)

- $\min^{n}_{i\in\mathcal{S}}\{x_i\}$: operator that returns the n^{th} smallest element from the set $\{x_i \mid i \in \mathcal{S}\}$ where $\mathcal{S}$ is some set (Chapter 4)

- $\underline{e}_i(t)$: wait-and-see expected minimum travel time from node i to node d at departure time t (Chapter 5)

Appendix B

Example of Minimum Possible Travel Time Paths Computation

This appendix demonstrates step-by-step execution of Algorithm MPTTP2 to find all-to-one minimum possible travel time paths in the stochastic time-dependent network depicted in Figure B-1 and Tables B.1–B.6.

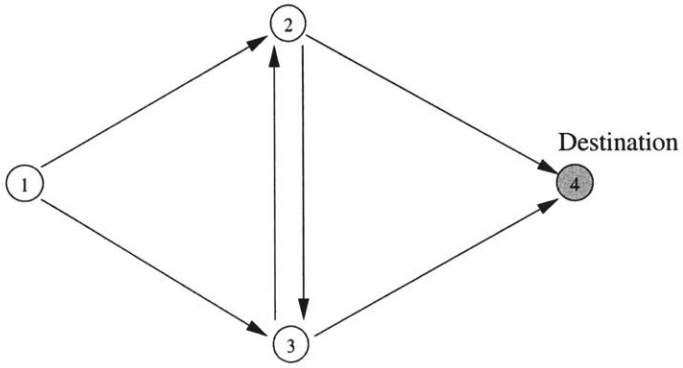

Figure B-1: An Example Network

Table B.1: Time-Dependent Link Travel Time PMFs

Link (i,j)	$(1,2)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (2, 0.5)	(2, 0.4) (3, 0.6)	(1, 0.6) (3, 0.4)	(3, 0.5) (4, 0.5)	(3, 0.3) (4, 0.7)	(2, 0.3) (4, 0.7)

Table B.2: Time-Dependent Link Travel Time PMFs (cont.)

Link (i,j)	$(1,3)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(2, 0.2) (3, 0.8)	(1, 0.7) (3, 0.3)	(1, 0.6) (3, 0.4)	(1, 0.6) (2, 0.4)	(2, 0.4) (5, 0.6)	(3, 0.4) (4, 0.6)

Table B.3: Time-Dependent Link Travel Time PMFs (cont.)

Link (i,j)	$(2,3)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(2, 0.7) (4, 0.3)	(1, 0.6) (2, 0.4)	(1, 0.8) (3, 0.2)	(2, 0.1) (3, 0.9)	(1, 0.9) (4, 0.1)	(1, 0.5) (4, 0.5)

Table B.4: Time-Dependent Link Travel Time PMFs (cont.)

Link (i,j)	$(3,2)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.1) (3, 0.9)	(2, 0.3) (3, 0.7)	(3, 0.1) (4, 0.9)	(1, 0.6) (3, 0.4)	(1, 0.8) (2, 0.2)	(1, 0.3) (3, 0.7)

Table B.5: Time-Dependent Link Travel Time PMFs (cont.)

Link (i,j)	$(2,4)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(2, 0.6) (3, 0.4)	(2, 0.4) (4, 0.6)	(1, 0.7) (3, 0.3)	(2, 0.5) (4, 0.5)	(2, 0.4) (4, 0.6)	(2, 0.5) (3, 0.5)

Table B.6: Time-Dependent Link Travel Time PMFs (cont.)

Link (i,j)	$(3,4)$					
Departure Time (t)	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
$(\tau_{ij}^r(t), p_{ij}^r(t))$	(1, 0.5) (3, 0.5)	(2, 0.2) (3, 0.8)	(1, 0.5) (2, 0.5)	(1, 0.8) (3, 0.2)	(3, 0.5) (5, 0.5)	(2, 0.8) (4, 0.2)

Step 1: Initialization

$(\lambda_i(t), \gamma_i(t), s_i(t), \pi_i(t)) = (\infty, 0, \infty, \infty), \quad \forall i \in \{1, 2, 3\}, \ \forall t \in \{0, 1, \ldots, 5\}$

$(\lambda_d(t), \gamma_d(t), s_d(t), \pi_d(t)) = (0, 1.0, d, t), \quad \forall t \in \{0, 1, \ldots, 5\}$

Step 2: Minimum Possible Travel Time Paths in Static Domain

$(\lambda_1(6), \gamma_1(6), s_1(6), \pi_1(6)) = (4, 0.15, 2, 8)$

$(\lambda_2(6), \gamma_2(6), s_2(6), \pi_2(6)) = (2, 0.5, 4, 8)$

$(\lambda_3(6), \gamma_3(6), s_3(6), \pi_3(6)) = (2, 0.8, 4, 8)$

$(\lambda_4(6), \gamma_4(6), s_4(6), \pi_4(6)) = (0, 1.0, 4, 6)$

Step 3: Minimum Possible Travel Time Paths in Time-Dependent Domain

At departure time $t = 5$

For link $(1, 2)$

$\mu = \tau_{12}^1(5) + \lambda_2(5 + \tau_{12}^1(5)) = 2 + \lambda_2(5 + 2) = 2 + 2 = 4$

$\nu = p_{12}^1(5) \times \gamma_2(5 + \tau_{12}^1(5)) = 0.3 \times \gamma_2(5 + 2) = 0.3 \times 0.5 = 0.15$

Since $\mu < \lambda_1(5) = \infty$, $\lambda_1(5) = 4$, $\gamma_1(5) = 0.15$, $s_1(5) = 2$, $\pi_1(5) = 7$

$\mu = \tau_{12}^2(5) + \lambda_2(5 + \tau_{12}^2(5)) = 4 + \lambda_2(5 + 4) = 4 + 2 = 6$

$\nu = p_{12}^2(5) \times \gamma_2(5 + \tau_{12}^2(5)) = 0.7 \times \gamma_2(5 + 4) = 0.7 \times 0.5 = 0.35$

Since $\mu > \lambda_1(5) = 4$, do not change $\lambda_1(5)$

For link $(1, 3)$

$\mu = \tau_{13}^1(5) + \lambda_3(5 + \tau_{13}^1(5)) = 3 + \lambda_3(5 + 3) = 3 + 2 = 5$

$\nu = p_{13}^1(5) \times \gamma_3(5 + \tau_{13}^1(5)) = 0.4 \times \gamma_3(5 + 3) = 0.4 \times 0.8 = 0.32$

Since $\mu > \lambda_1(5) = 4$, do not change $\lambda_1(5)$

$\mu = \tau_{13}^2(5) + \lambda_3(5 + \tau_{13}^2(5)) = 4 + \lambda_3(5 + 4) = 4 + 2 = 6$

$\nu = p_{13}^2(5) \times \gamma_3(5 + \tau_{13}^2(5)) = 0.6 \times \gamma_3(5 + 4) = 0.6 \times 0.8 = 0.48$

Since $\mu > \lambda_1(5) = 4$, do not change $\lambda_1(5)$

For link $(2, 3)$

$\mu = \tau_{23}^1(5) + \lambda_3(5 + \tau_{23}^1(5)) = 1 + \lambda_3(5 + 1) = 1 + 2 = 3$

$\nu = p_{23}^1(5) \times \gamma_3(5 + \tau_{23}^1(5)) = 0.5 \times \gamma_3(5 + 1) = 0.5 \times 0.8 = 0.4$

Since $\mu < \lambda_2(5) = \infty$, $\lambda_2(5) = 3$, $\gamma_2(5) = 0.4$, $s_2(5) = 3$, $\pi_2(5) = 6$

$\mu = \tau_{23}^2(5) + \lambda_3(5 + \tau_{23}^2(5)) = 4 + \lambda_3(5 + 4) = 4 + 2 = 6$

$\nu = p_{23}^2(5) \times \gamma_3(5 + \tau_{23}^2(5)) = 0.5 \times \gamma_3(5 + 4) = 0.5 \times 0.8 = 0.4$

Since $\mu > \lambda_2(5) = 3$, do not change $\lambda_2(5)$

For link $(2, 4)$

$\mu = \tau_{24}^1(5) + \lambda_4(5 + \tau_{24}^1(5)) = 2 + \lambda_4(5+2) = 2 + 0 = 2$

$\nu = p_{24}^1(5) \times \gamma_4(5 + \tau_{24}^1(5)) = 0.5 \times \gamma_4(5+2) = 0.5 \times 1.0 = 0.5$

Since $\mu < \lambda_2(5) = 3$, $\lambda_2(5) = 2$, $\gamma_2(5) = 0.5$, $s_2(5) = 4$, $\pi_2(5) = 7$

$\mu = \tau_{24}^2(5) + \lambda_4(5 + \tau_{24}^2(5)) = 3 + \lambda_4(5+3) = 3 + 0 = 3$

$\nu = p_{24}^2(5) \times \gamma_4(5 + \tau_{24}^2(5)) = 0.5 \times \gamma_4(5+3) = 0.5 \times 1.0 = 0.5$

Since $\mu > \lambda_2(5) = 2$, do not change $\lambda_2(5)$

For link $(3, 2)$

$\mu = \tau_{32}^1(5) + \lambda_2(5 + \tau_{32}^1(5)) = 1 + \lambda_2(5+1) = 1 + 2 = 3$

$\nu = p_{32}^1(5) \times \gamma_2(5 + \tau_{32}^1(5)) = 0.3 \times \gamma_2(5+1) = 0.3 \times 0.5 = 0.15$

Since $\mu < \lambda_3(5) = \infty$, $\lambda_3(5) = 3$, $\gamma_3(5) = 0.15$, $s_3(5) = 2$, $\pi_3(5) = 6$

$\mu = \tau_{32}^2(5) + \lambda_2(5 + \tau_{32}^2(5)) = 3 + \lambda_2(5+3) = 3 + 2 = 5$

$\nu = p_{32}^2(5) \times \gamma_2(5 + \tau_{32}^2(5)) = 0.7 \times \gamma_2(5+3) = 0.7 \times 0.5 = 0.35$

Since $\mu > \lambda_3(5) = 3$, do not change $\lambda_3(5)$

For link $(3, 4)$

$\mu = \tau_{34}^1(5) + \lambda_4(5 + \tau_{34}^1(5)) = 2 + \lambda_4(5+2) = 2 + 0 = 2$

$\nu = p_{34}^1(5) \times \gamma_4(5 + \tau_{34}^1(5)) = 0.8 \times \gamma_4(5+2) = 0.8 \times 1.0 = 0.8$

Since $\mu < \lambda_3(5) = 3$, $\lambda_3(5) = 2$, $\gamma_3(5) = 0.8$, $s_3(5) = 4$, $\pi_3(5) = 7$

$\mu = \tau_{34}^2(5) + \lambda_4(5 + \tau_{34}^2(5)) = 4 + \lambda_4(5+4) = 4 + 0 = 4$

$\nu = p_{34}^2(5) \times \gamma_4(5 + \tau_{34}^2(5)) = 0.2 \times \gamma_4(5+4) = 0.2 \times 1.0 = 0.2$

Since $\mu > \lambda_3(5) = 2$, do not change $\lambda_3(5)$

At departure time $t = 4$

For link $(1, 2)$

$\mu = \tau_{12}^1(4) + \lambda_2(4 + \tau_{12}^1(4)) = 3 + \lambda_2(4+3) = 3 + 2 = 5$

$\nu = p_{12}^1(4) \times \gamma_2(4 + \tau_{12}^1(4)) = 0.3 \times \gamma_2(4+3) = 0.3 \times 0.5 = 0.15$

Since $\mu < \lambda_1(4) = \infty$, $\lambda_1(4) = 5$, $\gamma_1(4) = 0.15$, $s_1(4) = 2$, $\pi_1(4) = 7$

$\mu = \tau_{12}^2(4) + \lambda_2(4 + \tau_{12}^2(4)) = 4 + \lambda_2(4+4) = 4 + 2 = 6$

$\nu = p_{12}^2(4) \times \gamma_2(4 + \tau_{12}^2(4)) = 0.7 \times \gamma_2(4+4) = 0.7 \times 0.5 = 0.35$

Since $\mu > \lambda_1(4) = 5$, do not change $\lambda_1(4)$

For link $(1, 3)$

$\mu = \tau_{13}^1(4) + \lambda_3(4 + \tau_{13}^1(4)) = 2 + \lambda_3(4+2) = 2 + 2 = 4$

$\nu = p_{13}^1(4) \times \gamma_3(4 + \tau_{13}^1(4)) = 0.4 \times \gamma_3(4+2) = 0.4 \times 0.8 = 0.32$

Since $\mu < \lambda_1(4) = 5$, $\lambda_1(4) = 4$, $\gamma_1(4) = 0.32$, $s_1(4) = 3$, $\pi_1(4) = 6$

$\mu = \tau_{13}^2(4) + \lambda_3(4 + \tau_{13}^2(4)) = 5 + \lambda_3(4+5) = 5 + 2 = 7$

$\nu = p_{13}^2(4) \times \gamma_3(4 + \tau_{13}^2(4)) = 0.6 \times \gamma_3(4+5) = 0.6 \times 0.8 = 0.48$

Since $\mu > \lambda_1(4) = 4$, do not change $\lambda_1(4)$

For link $(2,3)$

$\mu = \tau_{23}^1(4) + \lambda_3(4 + \tau_{23}^1(4)) = 1 + \lambda_3(4+1) = 1 + 2 = 3$

$\nu = p_{23}^1(4) \times \gamma_3(4 + \tau_{23}^1(4)) = 0.9 \times \gamma_3(4+1) = 0.9 \times 0.8 = 0.72$

Since $\mu < \lambda_2(4) = \infty$, $\lambda_2(4) = 3$, $\gamma_2(4) = 0.72$, $s_2(4) = 3$, $\pi_2(4) = 5$

$\mu = \tau_{23}^2(4) + \lambda_3(4 + \tau_{23}^2(4)) = 4 + \lambda_3(4+4) = 4 + 2 = 6$

$\nu = p_{23}^2(4) \times \gamma_3(4 + \tau_{23}^2(4)) = 0.1 \times \gamma_3(4+4) = 0.1 \times 0.8 = 0.08$

Since $\mu > \lambda_2(4) = 3$, do not change $\lambda_2(4)$

For link $(2,4)$

$\mu = \tau_{24}^1(4) + \lambda_4(4 + \tau_{24}^1(4)) = 2 + \lambda_4(4+2) = 2 + 0 = 2$

$\nu = p_{24}^1(4) \times \gamma_4(4 + \tau_{24}^1(4)) = 0.4 \times \gamma_4(4+2) = 0.4 \times 1.0 = 0.4$

Since $\mu < \lambda_2(4) = 3$, $\lambda_2(4) = 2$, $\gamma_2(4) = 0.4$, $s_2(4) = 4$, $\pi_2(4) = 6$

$\mu = \tau_{24}^2(4) + \lambda_4(4 + \tau_{24}^2(4)) = 4 + \lambda_4(4+4) = 4 + 0 = 4$

$\nu = p_{24}^2(4) \times \gamma_4(4 + \tau_{24}^2(4)) = 0.6 \times \gamma_4(4+4) = 0.6 \times 1.0 = 0.6$

Since $\mu > \lambda_2(4) = 2$, do not change $\lambda_2(4)$

For link $(3,2)$

$\mu = \tau_{32}^1(4) + \lambda_2(4 + \tau_{32}^1(4)) = 1 + \lambda_2(4+1) = 1 + 2 = 3$

$\nu = p_{32}^1(4) \times \gamma_2(4 + \tau_{32}^1(4)) = 0.8 \times \gamma_2(4+1) = 0.8 \times 0.5 = 0.4$

Since $\mu < \lambda_3(4) = \infty$, $\lambda_3(4) = 3$, $\gamma_3(4) = 0.4$, $s_3(4) = 2$, $\pi_3(4) = 5$

$\mu = \tau_{32}^2(4) + \lambda_2(4 + \tau_{32}^2(4)) = 2 + \lambda_2(4+2) = 2 + 2 = 4$

$\nu = p_{32}^2(4) \times \gamma_2(4 + \tau_{32}^2(4)) = 0.2 \times \gamma_2(4+2) = 0.2 \times 0.5 = 0.1$

Since $\mu > \lambda_3(4) = 3$, do not change $\lambda_3(4)$

For link $(3,4)$

$\mu = \tau_{34}^1(4) + \lambda_4(4 + \tau_{34}^1(4)) = 3 + \lambda_4(4+3) = 3 + 0 = 3$

$\nu = p_{34}^1(4) \times \gamma_4(4 + \tau_{34}^1(4)) = 0.5 \times \gamma_4(4+3) = 0.5 \times 1.0 = 0.5$

Since $\mu = \lambda_3(4) = 3$ and $\nu > \gamma_3(4) = 0.4$, $\lambda_3(4) = 3$, $\gamma_3(4) = 0.5$, $s_3(4) = 4$, $\pi_3(4) = 7$

$\mu = \tau_{34}^2(4) + \lambda_4(4 + \tau_{34}^2(4)) = 5 + \lambda_4(4+5) = 5 + 0 = 5$

$\nu = p_{34}^2(4) \times \gamma_4(4 + \tau_{34}^2(4)) = 0.5 \times \gamma_4(4+5) = 0.5 \times 1.0 = 0.5$

Since $\mu > \lambda_3(4) = 3$, do not change $\lambda_3(4)$

At departure time $t = 3$

For link $(1,2)$

$\mu = \tau_{12}^1(3) + \lambda_2(3 + \tau_{12}^1(3)) = 3 + \lambda_2(3+3) = 3 + 2 = 5$

$\nu = p_{12}^1(3) \times \gamma_2(3 + \tau_{12}^1(3)) = 0.5 \times \gamma_2(3+3) = 0.5 \times 0.5 = 0.25$

Since $\mu < \lambda_1(3) = \infty$, $\lambda_1(3) = 5$, $\gamma_1(3) = 0.25$, $s_1(3) = 2$, $\pi_1(3) = 6$

$\mu = \tau_{12}^2(3) + \lambda_2(3 + \tau_{12}^2(3)) = 4 + \lambda_2(3+4) = 4 + 2 = 6$

$\nu = p_{12}^2(3) \times \gamma_2(3 + \tau_{12}^2(3)) = 0.5 \times \gamma_2(3+4) = 0.5 \times 0.5 = 0.25$

Since $\mu > \lambda_1(3) = 5$, do not change $\lambda_1(3)$

For link $(1,3)$

$\mu = \tau_{13}^1(3) + \lambda_3(3 + \tau_{13}^1(3)) = 1 + \lambda_3(3+1) = 1 + 3 = 4$

$\nu = p_{13}^1(3) \times \gamma_3(3 + \tau_{13}^1(3)) = 0.6 \times \gamma_3(3+1) = 0.6 \times 0.5 = 0.3$

Since $\mu < \lambda_1(3) = 5$, $\lambda_1(3) = 4$, $\gamma_1(3) = 0.3$, $s_1(3) = 3$, $\pi_1(3) = 4$

$\mu = \tau_{13}^2(3) + \lambda_3(3 + \tau_{13}^2(3)) = 2 + \lambda_3(3+2) = 2 + 2 = 4$

$\nu = p_{13}^2(3) \times \gamma_3(3 + \tau_{13}^2(3)) = 0.4 \times \gamma_3(3+2) = 0.4 \times 0.8 = 0.32$

Since $\mu = \lambda_1(3) = 4$ and $\nu > \gamma_1(3) = 0.3$, $\lambda_1(3) = 4$, $\gamma_1(3) = 0.32$, $s_1(3) = 3$, $\pi_1(3) = 5$

For link $(2,3)$

$\mu = \tau_{23}^1(3) + \lambda_3(3 + \tau_{23}^1(3)) = 2 + \lambda_3(3+2) = 2 + 2 = 4$

$\nu = p_{23}^1(3) \times \gamma_3(3 + \tau_{23}^1(3)) = 0.1 \times \gamma_3(3+2) = 0.1 \times 0.8 = 0.08$

Since $\mu < \lambda_2(3) = \infty$, $\lambda_2(3) = 4$, $\gamma_2(3) = 0.08$, $s_2(3) = 3$, $\pi_2(3) = 5$

$\mu = \tau_{23}^2(3) + \lambda_3(3 + \tau_{23}^2(3)) = 3 + \lambda_3(3+3) = 3 + 2 = 5$

$\nu = p_{23}^2(3) \times \gamma_3(3 + \tau_{23}^2(3)) = 0.9 \times \gamma_3(3+3) = 0.9 \times 0.8 = 0.72$

Since $\mu > \lambda_2(3) = 4$, do not change $\lambda_2(3)$

For link $(2,4)$

$\mu = \tau_{24}^1(3) + \lambda_4(3 + \tau_{24}^1(3)) = 2 + \lambda_4(3+2) = 2 + 0 = 2$

$\nu = p_{24}^1(3) \times \gamma_4(3 + \tau_{24}^1(3)) = 0.5 \times \gamma_4(3+2) = 0.5 \times 1.0 = 0.5$

Since $\mu < \lambda_2(3) = 4$, $\lambda_2(3) = 2$, $\gamma_2(3) = 0.5$, $s_2(3) = 4$, $\pi_2(3) = 5$

$\mu = \tau_{24}^2(3) + \lambda_4(3 + \tau_{24}^2(3)) = 4 + \lambda_4(3+4) = 4 + 0 = 4$

$\nu = p_{24}^2(3) \times \gamma_4(3 + \tau_{24}^2(3)) = 0.5 \times \gamma_4(3+4) = 0.5 \times 1.0 = 0.5$

Since $\mu > \lambda_2(3) = 2$, do not change $\lambda_2(3)$

For link $(3,2)$

$\mu = \tau_{32}^1(3) + \lambda_2(3 + \tau_{32}^1(3)) = 1 + \lambda_2(3+1) = 1 + 2 = 3$

$\nu = p_{32}^1(3) \times \gamma_2(3 + \tau_{32}^1(3)) = 0.6 \times \gamma_2(3+1) = 0.6 \times 0.4 = 0.24$

Since $\mu < \lambda_3(3) = \infty$, $\lambda_3(3) = 3$, $\gamma_3(3) = 0.24$, $s_3(3) = 2$, $\pi_3(3) = 4$

$\mu = \tau_{32}^2(3) + \lambda_2(3 + \tau_{32}^2(3)) = 3 + \lambda_2(3+3) = 3 + 2 = 5$

$\nu = p_{32}^2(3) \times \gamma_2(3 + \tau_{32}^2(3)) = 0.4 \times \gamma_2(3+3) = 0.4 \times 0.5 = 0.2$

Since $\mu > \lambda_3(3) = 3$, do not change $\lambda_3(3)$

For link $(3,4)$

$\mu = \tau_{34}^1(3) + \lambda_4(3 + \tau_{34}^1(3)) = 1 + \lambda_4(3+1) = 1 + 0 = 1$

$\nu = p_{34}^1(3) \times \gamma_4(3 + \tau_{34}^1(3)) = 0.8 \times \gamma_4(3+1) = 0.8 \times 1.0 = 0.8$

Since $\mu < \lambda_3(3) = 3$, $\lambda_3(3) = 1$, $\gamma_3(3) = 0.8$, $s_3(3) = 4$, $\pi_3(3) = 4$

$\mu = \tau_{34}^2(3) + \lambda_4(3 + \tau_{34}^2(3)) = 3 + \lambda_4(3+3) = 3 + 0 = 3$

$\nu = p_{34}^2(3) \times \gamma_4(3 + \tau_{34}^2(3)) = 0.2 \times \gamma_4(3+3) = 0.2 \times 1.0 = 0.2$

Since $\mu > \lambda_3(3) = 1$, do not change $\lambda_3(3)$

At departure time $t = 2$

For link $(1,2)$

$\mu = \tau_{12}^1(2) + \lambda_2(2 + \tau_{12}^1(2)) = 1 + \lambda_2(2+1) = 1 + 2 = 3$

$\nu = p_{12}^1(2) \times \gamma_2(2 + \tau_{12}^1(2)) = 0.6 \times \gamma_2(2+1) = 0.6 \times 0.5 = 0.3$

Since $\mu < \lambda_1(2) = \infty$, $\lambda_1(2) = 3$, $\gamma_1(2) = 0.3$, $s_1(2) = 2$, $\pi_1(2) = 3$

$\mu = \tau_{12}^2(2) + \lambda_2(2 + \tau_{12}^2(2)) = 3 + \lambda_2(2+3) = 3 + 2 = 5$

$\nu = p_{12}^2(2) \times \gamma_2(2 + \tau_{12}^2(2)) = 0.4 \times \gamma_2(2+3) = 0.4 \times 0.5 = 0.2$

Since $\mu > \lambda_1(2) = 3$, do not change $\lambda_1(2)$

For link $(1,3)$

$\mu = \tau_{13}^1(2) + \lambda_3(2 + \tau_{13}^1(2)) = 1 + \lambda_3(2+1) = 1 + 1 = 2$

$\nu = p_{13}^1(2) \times \gamma_3(2 + \tau_{13}^1(2)) = 0.6 \times \gamma_3(2+1) = 0.6 \times 0.8 = 0.48$

Since $\mu < \lambda_1(2) = 3$, $\lambda_1(2) = 2$, $\gamma_1(2) = 0.48$, $s_1(2) = 3$, $\pi_1(2) = 3$

$\mu = \tau_{13}^2(2) + \lambda_3(2 + \tau_{13}^2(2)) = 3 + \lambda_3(2+3) = 3 + 2 = 5$

$\nu = p_{13}^2(2) \times \gamma_3(2 + \tau_{13}^2(2)) = 0.4 \times \gamma_3(2+3) = 0.4 \times 0.8 = 0.32$

Since $\mu > \lambda_1(2) = 2$, do not change $\lambda_1(2)$

For link $(2,3)$

$\mu = \tau_{23}^1(2) + \lambda_3(2 + \tau_{23}^1(2)) = 1 + \lambda_3(2+1) = 1 + 1 = 2$

$\nu = p_{23}^1(2) \times \gamma_3(2 + \tau_{23}^1(2)) = 0.8 \times \gamma_3(2+1) = 0.8 \times 0.8 = 0.64$

Since $\mu < \lambda_2(2) = \infty$, $\lambda_2(2) = 2$, $\gamma_2(2) = 0.64$, $s_2(2) = 3$, $\pi_2(2) = 3$

$\mu = \tau_{23}^2(2) + \lambda_3(2 + \tau_{23}^2(2)) = 3 + \lambda_3(2+3) = 3 + 2 = 5$

$\nu = p_{23}^2(2) \times \gamma_3(2 + \tau_{23}^2(2)) = 0.2 \times \gamma_3(2+3) = 0.2 \times 0.8 = 0.16$

Since $\mu > \lambda_2(2) = 2$, do not change $\lambda_2(2)$

For link $(2,4)$

$\mu = \tau^1_{24}(2) + \lambda_4(2 + \tau^1_{24}(2)) = 1 + \lambda_4(2+1) = 1 + 0 = 1$

$\nu = p^1_{24}(2) \times \gamma_4(2 + \tau^1_{24}(2)) = 0.7 \times \gamma_4(2+1) = 0.7 \times 1.0 = 0.7$

Since $\mu < \lambda_2(2) = 2$, $\lambda_2(2) = 1$, $\gamma_2(2) = 0.7$, $s_2(2) = 4$, $\pi_2(2) = 3$

$\mu = \tau^2_{24}(2) + \lambda_4(2 + \tau^2_{24}(2)) = 3 + \lambda_4(2+3) = 3 + 0 = 3$

$\nu = p^2_{24}(2) \times \gamma_4(2 + \tau^2_{24}(2)) = 0.3 \times \gamma_4(2+3) = 0.3 \times 1.0 = 0.3$

Since $\mu > \lambda_2(2) = 1$, do not change $\lambda_2(2)$

For link $(3,2)$

$\mu = \tau^1_{32}(2) + \lambda_2(2 + \tau^1_{32}(2)) = 3 + \lambda_2(2+3) = 3 + 2 = 5$

$\nu = p^1_{32}(2) \times \gamma_2(2 + \tau^1_{32}(2)) = 0.1 \times \gamma_2(2+3) = 0.1 \times 0.5 = 0.05$

Since $\mu < \lambda_3(2) = \infty$, $\lambda_3(2) = 5$, $\gamma_3(2) = 0.05$, $s_3(2) = 2$, $\pi_3(2) = 5$

$\mu = \tau^2_{32}(2) + \lambda_2(2 + \tau^2_{32}(2)) = 4 + \lambda_2(2+4) = 4 + 2 = 6$

$\nu = p^2_{32}(2) \times \gamma_2(2 + \tau^2_{32}(2)) = 0.9 \times \gamma_2(2+4) = 0.9 \times 0.5 = 0.45$

Since $\mu > \lambda_3(2) = 5$, do not change $\lambda_3(2)$

For link $(3,4)$

$\mu = \tau^1_{34}(2) + \lambda_4(2 + \tau^1_{34}(2)) = 1 + \lambda_4(2+1) = 1 + 0 = 1$

$\nu = p^1_{34}(2) \times \gamma_4(2 + \tau^1_{34}(2)) = 0.5 \times \gamma_4(2+1) = 0.5 \times 1.0 = 0.5$

Since $\mu < \lambda_3(2) = 5$, $\lambda_3(2) = 1$, $\gamma_3(2) = 0.5$, $s_3(2) = 4$, $\pi_3(2) = 3$

$\mu = \tau^2_{34}(2) + \lambda_4(2 + \tau^2_{34}(2)) = 2 + \lambda_4(2+2) = 2 + 0 = 2$

$\nu = p^2_{34}(2) \times \gamma_4(2 + \tau^2_{34}(2)) = 0.5 \times \gamma_4(2+2) = 0.5 \times 1.0 = 0.5$

Since $\mu > \lambda_3(2) = 1$, do not change $\lambda_3(2)$

At departure time $t = 1$

For link $(1,2)$

$\mu = \tau^1_{12}(1) + \lambda_2(1 + \tau^1_{12}(1)) = 2 + \lambda_2(1+2) = 2 + 2 = 4$

$\nu = p^1_{12}(1) \times \gamma_2(1 + \tau^1_{12}(1)) = 0.4 \times \gamma_2(1+2) = 0.4 \times 0.5 = 0.2$

Since $\mu < \lambda_1(1) = \infty$, $\lambda_1(1) = 4$, $\gamma_1(1) = 0.2$, $s_1(1) = 2$, $\pi_1(1) = 3$

$\mu = \tau^2_{12}(1) + \lambda_2(1 + \tau^2_{12}(1)) = 3 + \lambda_2(1+3) = 3 + 2 = 5$

$\nu = p^2_{12}(1) \times \gamma_2(1 + \tau^2_{12}(1)) = 0.6 \times \gamma_2(1+3) = 0.6 \times 0.4 = 0.24$

Since $\mu > \lambda_1(1) = 4$, do not change $\lambda_1(1)$

For link $(1,3)$

$\mu = \tau^1_{13}(1) + \lambda_3(1 + \tau^1_{13}(1)) = 1 + \lambda_3(1+1) = 1 + 1 = 2$

$\nu = p^1_{13}(1) \times \gamma_3(1 + \tau^1_{13}(1)) = 0.7 \times \gamma_3(1+1) = 0.7 \times 0.5 = 0.35$

Since $\mu < \lambda_1(1) = 4$, $\lambda_1(1) = 2$, $\gamma_1(1) = 0.35$, $s_1(1) = 3$, $\pi_1(1) = 2$

$\mu = \tau_{13}^2(1) + \lambda_3(1 + \tau_{13}^2(1)) = 3 + \lambda_3(1 + 3) = 3 + 3 = 6$

$\nu = p_{13}^2(1) \times \gamma_3(1 + \tau_{13}^2(1)) = 0.3 \times \gamma_3(1 + 3) = 0.3 \times 0.5 = 0.15$

Since $\mu > \lambda_1(1) = 2$, do not change $\lambda_1(1)$

For link $(2, 3)$

$\mu = \tau_{23}^1(1) + \lambda_3(1 + \tau_{23}^1(1)) = 1 + \lambda_3(1 + 1) = 1 + 1 = 2$

$\nu = p_{23}^1(1) \times \gamma_3(1 + \tau_{23}^1(1)) = 0.6 \times \gamma_3(1 + 1) = 0.6 \times 0.5 = 0.3$

Since $\mu < \lambda_2(1) = \infty$, $\lambda_2(1) = 2$, $\gamma_2(1) = 0.3$, $s_2(1) = 3$, $\pi_2(1) = 2$

$\mu = \tau_{23}^2(1) + \lambda_3(1 + \tau_{23}^2(1)) = 2 + \lambda_3(1 + 2) = 2 + 1 = 3$

$\nu = p_{23}^2(1) \times \gamma_3(1 + \tau_{23}^2(1)) = 0.4 \times \gamma_3(1 + 2) = 0.4 \times 0.8 = 0.32$

Since $\mu > \lambda_2(1) = 2$, do not change $\lambda_2(1)$

For link $(2, 4)$

$\mu = \tau_{24}^1(1) + \lambda_4(1 + \tau_{24}^1(1)) = 2 + \lambda_4(1 + 2) = 2 + 0 = 2$

$\nu = p_{24}^1(1) \times \gamma_4(1 + \tau_{24}^1(1)) = 0.4 \times \gamma_4(1 + 2) = 0.4 \times 1.0 = 0.4$

Since $\mu = \lambda_2(1) = 2$ and $\nu > \gamma_2(1) = 0.3$, $\lambda_2(1) = 2$, $\gamma_2(1) = 0.4$, $s_2(1) = 4$, $\pi_2(1) = 3$

$\mu = \tau_{24}^2(1) + \lambda_4(1 + \tau_{24}^2(1)) = 4 + \lambda_4(1 + 4) = 4 + 0 = 4$

$\nu = p_{24}^2(1) \times \gamma_4(1 + \tau_{24}^2(1)) = 0.6 \times \gamma_4(1 + 4) = 0.6 \times 1.0 = 0.6$

Since $\mu > \lambda_2(1) = 2$, do not change $\lambda_2(1)$

For link $(3, 2)$

$\mu = \tau_{32}^1(1) + \lambda_2(1 + \tau_{32}^1(1)) = 2 + \lambda_2(1 + 2) = 2 + 2 = 4$

$\nu = p_{32}^1(1) \times \gamma_2(1 + \tau_{32}^1(1)) = 0.3 \times \gamma_2(1 + 2) = 0.3 \times 0.5 = 0.15$

Since $\mu < \lambda_3(1) = \infty$, $\lambda_3(1) = 4$, $\gamma_3(1) = 0.15$, $s_3(1) = 2$, $\pi_3(1) = 3$

$\mu = \tau_{32}^2(1) + \lambda_2(1 + \tau_{32}^2(1)) = 3 + \lambda_2(1 + 3) = 3 + 2 = 5$

$\nu = p_{32}^2(1) \times \gamma_2(1 + \tau_{32}^2(1)) = 0.7 \times \gamma_2(1 + 3) = 0.7 \times 0.4 = 0.28$

Since $\mu > \lambda_3(1) = 4$, do not change $\lambda_3(1)$

For link $(3, 4)$

$\mu = \tau_{34}^1(1) + \lambda_4(1 + \tau_{34}^1(1)) = 2 + \lambda_4(1 + 2) = 2 + 0 = 2$

$\nu = p_{34}^1(1) \times \gamma_4(1 + \tau_{34}^1(1)) = 0.2 \times \gamma_4(1 + 2) = 0.2 \times 1.0 = 0.2$

Since $\mu < \lambda_3(1) = 4$, $\lambda_3(1) = 2$, $\gamma_3(1) = 0.2$, $s_3(1) = 4$, $\pi_3(1) = 3$

$\mu = \tau_{34}^2(1) + \lambda_4(1 + \tau_{34}^2(1)) = 3 + \lambda_4(1 + 3) = 3 + 0 = 3$

$\nu = p_{34}^2(1) \times \gamma_4(1 + \tau_{34}^2(1)) = 0.8 \times \gamma_4(1 + 3) = 0.8 \times 1.0 = 0.8$

Since $\mu > \lambda_3(1) = 2$, do not change $\lambda_3(1)$

At departure time $t = 0$

For link $(1, 2)$

$\mu = \tau_{12}^1(0) + \lambda_2(0 + \tau_{12}^1(0)) = 1 + \lambda_2(0 + 1) = 1 + 2 = 3$

$\nu = p_{12}^1(0) \times \gamma_2(0 + \tau_{12}^1(0)) = 0.5 \times \gamma_2(0 + 1) = 0.5 \times 0.4 = 0.2$

Since $\mu < \lambda_1(0) = \infty$, $\lambda_1(0) = 3$, $\gamma_1(0) = 0.2$, $s_1(0) = 2$, $\pi_1(0) = 1$

$\mu = \tau_{12}^2(0) + \lambda_2(0 + \tau_{12}^2(0)) = 2 + \lambda_2(0 + 2) = 2 + 1 = 3$

$\nu = p_{12}^2(0) \times \gamma_2(0 + \tau_{12}^2(0)) = 0.5 \times \gamma_2(0 + 2) = 0.5 \times 0.7 = 0.35$

Since $\mu = \lambda_1(0) = 3$ and $\nu > \gamma_1(0) = 0.2$, $\lambda_1(0) = 3$, $\gamma_1(0) = 0.35$, $s_1(0) = 2$, $\pi_1(0) = 2$

For link $(1, 3)$

$\mu = \tau_{13}^1(0) + \lambda_3(0 + \tau_{13}^1(0)) = 2 + \lambda_3(0 + 2) = 2 + 1 = 3$

$\nu = p_{13}^1(0) \times \gamma_3(0 + \tau_{13}^1(0)) = 0.2 \times \gamma_3(0 + 2) = 0.2 \times 0.5 = 0.1$

Since $\mu = \lambda_1(0) = 3$ but $\nu < \gamma_1(0)$, do not change $\lambda_1(0)$

$\mu = \tau_{13}^2(0) + \lambda_3(0 + \tau_{13}^2(0)) = 3 + \lambda_3(0 + 3) = 3 + 1 = 4$

$\nu = p_{13}^2(0) \times \gamma_3(0 + \tau_{13}^2(0)) = 0.8 \times \gamma_3(0 + 3) = 0.8 \times 0.8 = 0.64$

Since $\mu > \lambda_1(0) = 3$, do not change $\lambda_1(0)$

For link $(2, 3)$

$\mu = \tau_{23}^1(0) + \lambda_3(0 + \tau_{23}^1(0)) = 2 + \lambda_3(0 + 2) = 2 + 1 = 3$

$\nu = p_{23}^1(0) \times \gamma_3(0 + \tau_{23}^1(0)) = 0.7 \times \gamma_3(0 + 2) = 0.7 \times 0.5 = 0.35$

Since $\mu < \lambda_2(0) = \infty$, $\lambda_2(0) = 3$, $\gamma_2(0) = 0.35$, $s_2(0) = 3$, $\pi_2(0) = 2$

$\mu = \tau_{23}^2(0) + \lambda_3(0 + \tau_{23}^2(0)) = 4 + \lambda_3(0 + 4) = 4 + 3 = 7$

$\nu = p_{23}^2(0) \times \gamma_3(0 + \tau_{23}^2(0)) = 0.3 \times \gamma_3(0 + 4) = 0.3 \times 0.5 = 0.15$

Since $\mu > \lambda_2(0) = 3$, do not change $\lambda_2(0)$

For link $(2, 4)$

$\mu = \tau_{24}^1(0) + \lambda_4(0 + \tau_{24}^1(0)) = 2 + \lambda_4(0 + 2) = 2 + 0 = 2$

$\nu = p_{24}^1(0) \times \gamma_4(0 + \tau_{24}^1(0)) = 0.6 \times \gamma_4(0 + 2) = 0.6 \times 1.0 = 0.6$

Since $\mu < \lambda_2(0) = 3$, $\lambda_2(0) = 2$, $\gamma_2(0) = 0.6$, $s_2(0) = 4$, $\pi_2(0) = 2$

$\mu = \tau_{24}^2(0) + \lambda_4(0 + \tau_{24}^2(0)) = 3 + \lambda_4(0 + 3) = 3 + 0 = 3$

$\nu = p_{24}^2(0) \times \gamma_4(0 + \tau_{24}^2(0)) = 0.4 \times \gamma_4(0 + 3) = 0.4 \times 1.0 = 0.4$

Since $\mu > \lambda_2(0) = 2$, do not change $\lambda_2(0)$

For link $(3, 2)$

$\mu = \tau_{32}^1(0) + \lambda_2(0 + \tau_{32}^1(0)) = 1 + \lambda_2(0 + 1) = 1 + 2 = 3$

$\nu = p_{32}^1(0) \times \gamma_2(0 + \tau_{32}^1(0)) = 0.1 \times \gamma_2(0 + 1) = 0.1 \times 0.4 = 0.04$

Since $\mu < \lambda_3(0) = \infty$, $\lambda_3(0) = 3$, $\gamma_3(0) = 0.04$, $s_3(0) = 2$, $\pi_3(0) = 1$

$\mu = \tau^2_{32}(0) + \lambda_2(0 + \tau^2_{32}(0)) = 3 + \lambda_2(0 + 3) = 3 + 2 = 5$

$\nu = p^2_{32}(0) \times \gamma_2(0 + \tau^2_{32}(0)) = 0.9 \times \gamma_2(0 + 3) = 0.9 \times 0.5 = 0.45$

Since $\mu > \lambda_3(0) = 3$, do not change $\lambda_3(0)$

For link $(3, 4)$

$\mu = \tau^1_{34}(0) + \lambda_4(0 + \tau^1_{34}(0)) = 1 + \lambda_4(0 + 1) = 1 + 0 = 1$

$\nu = p^1_{34}(0) \times \gamma_4(0 + \tau^1_{34}(0)) = 0.5 \times \gamma_4(0 + 1) = 0.5 \times 1.0 = 0.5$

Since $\mu < \lambda_3(0) = 3$, $\lambda_3(0) = 1$, $\gamma_3(0) = 0.5$, $s_3(0) = 4$, $\pi_3(0) = 1$

$\mu = \tau^2_{34}(0) + \lambda_4(0 + \tau^2_{34}(0)) = 3 + \lambda_4(0 + 3) = 3 + 0 = 3$

$\nu = p^2_{34}(0) \times \gamma_4(0 + \tau^2_{34}(0)) = 0.5 \times \gamma_4(0 + 3) = 0.5 \times 1.0 = 0.5$

Since $\mu > \lambda_3(0) = 1$, do not change $\lambda_3(0)$

Table B.7: Results

Origin Node (i)	Departure Time (t)	$\lambda_i(t)$	$\gamma_i(t)$	Successor ($s_i(t)$)	Arrival Time at $s_i(t)$ ($\pi_i(t)$)
1	0	3	0.350	2	2
	1	2	0.350	3	2
	2	2	0.480	3	3
	3	4	0.320	3	5
	4	4	0.320	3	6
	5	4	0.150	2	7
	≥ 6	4	0.150	2	8
2	0	2	0.600	4	2
	1	2	0.400	4	3
	2	1	0.700	4	3
	3	2	0.500	4	5
	4	2	0.400	4	6
	5	2	0.500	4	7
	≥ 6	2	0.500	4	8
3	0	1	0.500	4	1
	1	2	0.200	4	3
	2	1	0.500	4	3
	3	1	0.800	4	4
	4	3	0.500	4	7
	5	2	0.800	4	7
	≥ 6	2	0.800	4	8

Appendix C

Example of Minimum Expected Travel Time Next-Arc Hyperpaths Computation

In this appendix, we demonstrate how Algorithm METTH solves the all-to-one minimum expected travel time next-arc hyperpaths problem using the stochastic time-dependent network in Appendix B.

Step 1: Initialization

$(e_i(t), s_i(t)) = (\infty, \infty), \quad \forall i \in \{1, 2, 3\}, \ \forall t \in \{0, 1, \ldots, 5\}$

$(e_d(t), s_d(t)) = (0, d), \quad \forall t \in \{0, 1, \ldots, 5\}$

Step 2: Minimum Expected Travel Time Next-Arc Hyperpaths in Static Domain

$(e_1(6), s_1(6)) = (5.9, 2)$

$(e_2(6), s_2(6)) = (2.5, 4)$

$(e_3(6), s_3(6)) = (2.4, 4)$

$(e_4(6), s_4(6)) = (0.0, 4)$

Step 3: Minimum Expected Travel Time Next-Arc Hyperpaths in Time-Dependent Domain

At departure time $t = 5$

For link $(1, 2)$

$$e = \sum_{r \in \mathcal{R}_{12}(5)} \left(\tau_{12}^r(5) + e_2(5 + \tau_{12}^r(5))\right) p_{12}^r(5)$$

$$= \left(2 + e_2(5 + 2)\right) 0.3 + \left(4 + e_2(5 + 4)\right) 0.7 = (2 + 2.5)\, 0.3 + (4 + 2.5)\, 0.7 = 5.9$$

Since $e < e_1(5) = \infty$, $e_1(5) = 5.9$, $s_1(5) = 2$

For link $(1,3)$

$e = \sum_{r \in \mathcal{R}_{13}(5)} \left(\tau_{13}^r(5) + e_3(5 + \tau_{13}^r(5))\right) p_{13}^r(5)$

$= \left(3 + e_2(5+3)\right) 0.4 + \left(4 + e_2(5+4)\right) 0.6 = (3 + 2.4)\,0.4 + (4 + 2.4)\,0.6 = 6.0$

Since $e > e_1(5) = 5.9$, do not change $e_1(5)$

For link $(2,3)$

$e = \sum_{r \in \mathcal{R}_{23}(5)} \left(\tau_{23}^r(5) + e_3(5 + \tau_{23}^r(5))\right) p_{23}^r(5)$

$= \left(1 + e_2(5+1)\right) 0.5 + \left(4 + e_2(5+4)\right) 0.5 = (1 + 2.4)\,0.5 + (4 + 2.4)\,0.5 = 4.9$

Since $e < e_2(5) = \infty$, $e_2(5) = 4.9$, $s_2(5) = 3$

For link $(2,4)$

$e = \sum_{r \in \mathcal{R}_{24}(5)} \left(\tau_{24}^r(5) + e_4(5 + \tau_{24}^r(5))\right) p_{24}^r(5)$

$= \left(2 + e_2(5+2)\right) 0.5 + \left(3 + e_2(5+3)\right) 0.5 = (2 + 0.0)\,0.5 + (3 + 0.0)\,0.5 = 2.5$

Since $e < e_2(5) = 4.9$, $e_2(5) = 2.5$, $s_2(5) = 4$

For link $(3,2)$

$e = \sum_{r \in \mathcal{R}_{32}(5)} \left(\tau_{32}^r(5) + e_2(5 + \tau_{32}^r(5))\right) p_{32}^r(5)$

$= \left(1 + e_2(5+1)\right) 0.3 + \left(3 + e_2(5+3)\right) 0.7 = (1 + 2.5)\,0.3 + (3 + 2.5)\,0.7 = 4.9$

Since $e < e_3(5) = \infty$, $e_3(5) = 4.9$, $s_3(5) = 2$

For link $(3,4)$

$e = \sum_{r \in \mathcal{R}_{34}(5)} \left(\tau_{34}^r(5) + e_4(5 + \tau_{34}^r(5))\right) p_{34}^r(5)$

$= \left(2 + e_2(5+2)\right) 0.8 + \left(4 + e_2(5+4)\right) 0.2 = (2 + 0.0)\,0.8 + (4 + 0.0)\,0.2 = 2.4$

Since $e < e_3(5) = 4.9$, $e_3(5) = 2.4$, $s_3(5) = 4$

At departure time $t = 4$

For link $(1,2)$

$e = \sum_{r \in \mathcal{R}_{12}(4)} \left(\tau_{12}^r(4) + e_2(4 + \tau_{12}^r(4))\right) p_{12}^r(4)$

$= \left(3 + e_2(4+3)\right) 0.3 + \left(4 + e_2(4+4)\right) 0.7 = (3 + 2.5)\,0.3 + (4 + 2.5)\,0.7 = 6.2$

Since $e < e_1(4) = \infty$, $e_1(4) = 6.2$, $s_1(4) = 2$

For link $(1,3)$

$e = \sum_{r \in \mathcal{R}_{13}(4)} \left(\tau_{13}^r(4) + e_3(4 + \tau_{13}^r(4))\right) p_{13}^r(4)$

$= \left(2 + e_3(4+2)\right) 0.4 + \left(5 + e_3(4+4)\right) 0.6 = (2 + 2.4)\,0.4 + (5 + 2.4)\,0.6 = 6.2$

Since $e = e_1(4) = 6.2$, do not change $e_1(4)$

For link $(2,3)$

$e = \sum_{r \in \mathcal{R}_{23}(4)} \left(\tau_{23}^r(4) + e_3(4 + \tau_{23}^r(4))\right) p_{23}^r(4)$

$= \big(1 + e_3(4+1)\big)\,0.9 + \big(4 + e_3(4+4)\big)\,0.1 = (1+2.4)\,0.9 + (4+2.4)\,0.1 = 3.7$

Since $e < e_2(4) = \infty$, $e_2(4) = 3.7$, $s_2(4) = 3$

For link $(2,4)$

$e = \sum_{r \in \mathcal{R}_{24}(4)} \big(\tau^r_{24}(4) + e_4(4 + \tau^r_{24}(4))\big)\, p^r_{24}(4)$

$= \big(2 + e_4(4+2)\big)\,0.4 + \big(4 + e_4(4+4)\big)\,0.6 = (2+0.0)\,0.4 + (4+0.0)\,0.6 = 3.2$

Since $e < e_2(4) = 3.7$, $e_2(4) = 3.2$, $s_2(4) = 4$

For link $(3,2)$

$e = \sum_{r \in \mathcal{R}_{32}(4)} \big(\tau^r_{32}(4) + e_2(4 + \tau^r_{32}(4))\big)\, p^r_{32}(4)$

$= \big(1 + e_2(4+1)\big)\,0.8 + \big(2 + e_2(4+2)\big)\,0.2 = (1+2.5)\,0.8 + (2+2.5)\,0.2 = 3.7$

Since $e < e_3(4) = \infty$, $e_3(4) = 3.7$, $s_3(4) = 2$

For link $(3,4)$

$e = \sum_{r \in \mathcal{R}_{34}(4)} \big(\tau^r_{34}(4) + e_4(4 + \tau^r_{34}(4))\big)\, p^r_{34}(4)$

$= \big(3 + e_4(4+3)\big)\,0.5 + \big(5 + e_4(4+5)\big)\,0.5 = (3+0.0)\,0.5 + (5+0.0)\,0.5 = 4.0$

Since $e > e_3(4) = 3.7$, do not change $e_3(4)$

At departure time $t = 3$

For link $(1,2)$

$e = \sum_{r \in \mathcal{R}_{12}(3)} \big(\tau^r_{12}(3) + e_2(3 + \tau^r_{12}(3))\big)\, p^r_{12}(3)$

$= \big(3 + e_2(3+3)\big)\,0.5 + \big(4 + e_2(3+4)\big)\,0.5 = (3+2.5)\,0.5 + (4+2.5)\,0.5 = 6.0$

Since $e < e_1(3) = \infty$, $e_1(3) = 6.0$, $s_1(3) = 2$

For link $(1,3)$

$e = \sum_{r \in \mathcal{R}_{13}(3)} \big(\tau^r_{13}(3) + e_3(3 + \tau^r_{13}(3))\big)\, p^r_{13}(3)$

$= \big(1 + e_3(3+1)\big)\,0.6 + \big(2 + e_3(3+2)\big)\,0.4 = (1+3.7)\,0.6 + (2+2.4)\,0.4 = 4.6$

Since $e < e_1(3) = 6.0$, $e_1(3) = 4.6$, $s_1(3) = 3$

For link $(2,3)$

¿$e = \sum_{r \in \mathcal{R}_{23}(3)} \big(\tau^r_{23}(3) + e_3(3 + \tau^r_{23}(3))\big)\, p^r_{23}(3)$

$= \big(2 + e_3(3+2)\big)\,0.1 + \big(3 + e_3(3+3)\big)\,0.9 = (2+2.4)\,0.1 + (3+2.4)\,0.9 = 5.3$

Since $e < e_2(3) = \infty$, $e_2(3) = 5.3$, $s_2(3) = 3$

For link $(2,4)$

$e = \sum_{r \in \mathcal{R}_{24}(3)} \big(\tau^r_{24}(3) + e_4(3 + \tau^r_{24}(3))\big)\, p^r_{24}(3)$

$= \big(2 + e_4(3+2)\big)\,0.5 + \big(4 + e_4(3+4)\big)\,0.5 = (2+0.0)\,0.5 + (4+0.0)\,0.5 = 3.0$

Since $e < e_2(3) = 5.3$, $e_2(3) = 3.0$, $s_2(3) = 4$

For link $(3,2)$

$e = \sum_{r\in\mathcal{R}_{32}(3)} \left(\tau^r_{32}(3) + e_2(3+\tau^r_{32}(3))\right) p^r_{32}(3)$

$= \left(1 + e_2(3+1)\right) 0.6 + \left(3 + e_2(3+3)\right) 0.4 = (1+3.2)\,0.6 + (3+2.5)\,0.4 = 4.7$

Since $e < e_3(3) = \infty$, $e_3(3) = 4.7$, $s_3(3) = 2$

For link $(3,4)$

$e = \sum_{r\in\mathcal{R}_{34}(3)} \left(\tau^r_{34}(3) + e_4(3+\tau^r_{34}(3))\right) p^r_{34}(3)$

$= \left(1 + e_4(3+1)\right) 0.8 + \left(3 + e_4(3+3)\right) 0.2 = (1+0.0)\,0.8 + (3+0.0)\,0.2 = 1.4$

Since $e < e_3(3) = 4.7$, $e_3(3) = 1.4$, $s_3(3) = 4$

At departure time $t = 2$

For link $(1,2)$

$e = \sum_{r\in\mathcal{R}_{12}(2)} \left(\tau^r_{12}(2) + e_2(2+\tau^r_{12}(2))\right) p^r_{12}(2)$

$= \left(1 + e_2(2+1)\right) 0.6 + \left(3 + e_2(2+3)\right) 0.4 = (1+3.0)\,0.6 + (3+2.5)\,0.4 = 4.6$

Since $e < e_1(2) = \infty$, $e_1(2) = 4.6$, $s_1(2) = 2$

For link $(1,3)$

$e = \sum_{r\in\mathcal{R}_{13}(2)} \left(\tau^r_{13}(2) + e_3(2+\tau^r_{13}(2))\right) p^r_{13}(2)$

$= \left(1 + e_3(2+1)\right) 0.6 + \left(3 + e_3(2+3)\right) 0.4 = (1+1.4)\,0.6 + (3+2.4)\,0.4 = 3.6$

Since $e < e_1(2) = 4.6$, $e_1(2) = 3.6$, $s_1(2) = 3$

For link $(2,3)$

$e = \sum_{r\in\mathcal{R}_{23}(2)} \left(\tau^r_{23}(2) + e_3(2+\tau^r_{23}(2))\right) p^r_{23}(2)$

$= \left(1 + e_3(2+1)\right) 0.8 + \left(3 + e_3(2+3)\right) 0.2 = (1+1.4)\,0.8 + (3+2.4)\,0.2 = 3.0$

Since $e < e_2(2) = \infty$, $e_2(2) = 3.0$, $s_2(2) = 3$

For link $(2,4)$

$e = \sum_{r\in\mathcal{R}_{24}(2)} \left(\tau^r_{24}(2) + e_4(2+\tau^r_{24}(2))\right) p^r_{24}(2)$

$= \left(1 + e_4(2+1)\right) 0.7 + \left(3 + e_4(2+3)\right) 0.3 = (1+0.0)\,0.7 + (3+0.0)\,0.3 = 1.6$

Since $e < e_2(2) = 3.0$, $e_2(2) = 1.6$, $s_2(2) = 4$

For link $(3,2)$

$e = \sum_{r\in\mathcal{R}_{32}(2)} \left(\tau^r_{32}(2) + e_2(2+\tau^r_{32}(2))\right) p^r_{32}(2)$

$= \left(3 + e_2(2+3)\right) 0.1 + \left(4 + e_2(2+4)\right) 0.9 = (3+2.5)\,0.1 + (4+2.5)\,0.9 = 6.4$

Since $e < e_3(2) = \infty$, $e_3(2) = 6.4$, $s_3(2) = 2$

For link $(3,4)$

$e = \sum_{r\in\mathcal{R}_{34}(2)} \left(\tau^r_{34}(2) + e_4(2+\tau^r_{34}(2))\right) p^r_{34}(2)$

$= \left(1 + e_4(2+1)\right) 0.5 + \left(2 + e_4(2+2)\right) 0.5 = (1+0.0)\,0.5 + (2+0.0)\,0.5 = 1.5$

Since $e < e_3(2) = 6.4$, $e_3(2) = 1.5$, $s_3(2) = 4$

At departure time $t = 1$

For link $(1, 2)$

$e = \sum_{r \in \mathcal{R}_{12}(1)} \left(\tau^r_{12}(1) + e_2(1 + \tau^r_{12}(1))\right) p^r_{12}(1)$

$= \left(2 + e_2(1 + 2)\right) 0.4 + \left(3 + e_2(1 + 3)\right) 0.6 = (2 + 3.0)\, 0.4 + (3 + 3.2)\, 0.6 = 5.7$

Since $e < e_1(1) = \infty$, $e_1(1) = 5.7$, $s_1(1) = 2$

For link $(1, 3)$

$e = \sum_{r \in \mathcal{R}_{13}(1)} \left(\tau^r_{13}(1) + e_3(1 + \tau^r_{13}(1))\right) p^r_{13}(1)$

$= \left(1 + e_3(1 + 1)\right) 0.7 + \left(3 + e_3(1 + 3)\right) 0.3 = (1 + 1.5)\, 0.7 + (3 + 3.7)\, 0.3 = 3.8$

Since $e < e_1(1) = 5.7$, $e_1(1) = 3.8$, $s_1(1) = 3$

For link $(2, 3)$

$e = \sum_{r \in \mathcal{R}_{23}(1)} \left(\tau^r_{23}(1) + e_3(1 + \tau^r_{23}(1))\right) p^r_{23}(1)$

$= \left(1 + e_3(1 + 1)\right) 0.6 + \left(2 + e_3(1 + 2)\right) 0.4 = (1 + 1.5)\, 0.6 + (2 + 1.4)\, 0.4 = 2.9$

Since $e < e_2(1) = \infty$, $e_2(1) = 2.9$, $s_2(1) = 3$

For link $(2, 4)$

$e = \sum_{r \in \mathcal{R}_{24}(1)} \left(\tau^r_{24}(1) + e_4(1 + \tau^r_{24}(1))\right) p^r_{24}(1)$

$= \left(2 + e_4(1 + 2)\right) 0.4 + \left(4 + e_4(1 + 4)\right) 0.6 = (2 + 0.0)\, 0.4 + (4 + 0.0)\, 0.6 = 3.2$

Since $e > e_2(1) = 2.9$, do not change $e_2(1)$

For link $(3, 2)$

$e = \sum_{r \in \mathcal{R}_{32}(1)} \left(\tau^r_{32}(1) + e_2(1 + \tau^r_{32}(1))\right) p^r_{32}(1)$

$= \left(2 + e_2(1 + 2)\right) 0.3 + \left(3 + e_2(1 + 3)\right) 0.7 = (2 + 3.0)\, 0.3 + (3 + 3.2)\, 0.7 = 5.8$

Since $e < e_3(1) = \infty$, $e_3(1) = 5.8$, $s_3(1) = 2$

For link $(3, 4)$

$e = \sum_{r \in \mathcal{R}_{34}(1)} \left(\tau^r_{34}(1) + e_4(1 + \tau^r_{34}(1))\right) p^r_{34}(1)$

$= \left(2 + e_4(1 + 2)\right) 0.2 + \left(3 + e_4(1 + 3)\right) 0.8 = (2 + 0.0)\, 0.2 + (3 + 0.0)\, 0.8 = 2.8$

Since $e < e_3(1) = 5.8$, $e_3(1) = 2.8$, $s_3(1) = 4$

At departure time $t = 0$

For link $(1, 2)$

$e = \sum_{r \in \mathcal{R}_{12}(0)} \left(\tau^r_{12}(0) + e_2(0 + \tau^r_{12}(0))\right) p^r_{12}(0)$

$= \left(1 + e_2(0 + 1)\right) 0.5 + \left(2 + e_2(0 + 2)\right) 0.5 = (1 + 2.9)\, 0.5 + (2 + 1.6)\, 0.5 = 3.7$

Since $e < e_1(0) = \infty$, $e_1(0) = 3.7$, $s_1(0) = 2$

For link $(1, 3)$

$e = \sum_{r \in \mathcal{R}_{13}(0)} \left(\tau^r_{13}(0) + e_3(0 + \tau^r_{13}(0))\right) p^r_{13}(0)$

$= \big(2 + e_3(0+2)\big)\,0.2 + \big(3 + e_3(0+3)\big)\,0.8 = (2+1.5)\,0.2 + (3+1.4)\,0.8 = 4.2$

Since $e > e_1(0) = 3.7$, do not change $e_1(0)$

For link $(2,3)$

$e = \sum_{r \in \mathcal{R}_{23}(0)} \big(\tau^r_{23}(0) + e_3(0 + \tau^r_{23}(0))\big)\, p^r_{23}(0)$

$= \big(2 + e_3(0+2)\big)\,0.7 + \big(4 + e_3(0+4)\big)\,0.3 = (2+1.5)\,0.7 + (4+3.7)\,0.3 = 4.8$

Since $e < e_2(0) = \infty$, $e_2(0) = 4.8$, $s_2(0) = 3$

For link $(2,4)$

$e = \sum_{r \in \mathcal{R}_{24}(0)} \big(\tau^r_{24}(0) + e_4(0 + \tau^r_{24}(0))\big)\, p^r_{24}(0)$

$= \big(2 + e_4(0+2)\big)\,0.6 + \big(3 + e_4(0+3)\big)\,0.4 = (2+0.0)\,0.6 + (3+0.0)\,0.4 = 2.4$

Since $e < e_2(0) = 4.8$, $e_2(0) = 2.4$, $s_2(0) = 4$

For link $(3,2)$

$e = \sum_{r \in \mathcal{R}_{32}(0)} \big(\tau^r_{32}(0) + e_2(0 + \tau^r_{32}(0))\big)\, p^r_{32}(0)$

$= \big(1 + e_2(0+1)\big)\,0.1 + \big(3 + e_2(0+3)\big)\,0.9 = (1+2.9)\,0.1 + (3+3.0)\,0.9 = 5.8$

Since $e < e_3(0) = \infty$, $e_3(0) = 5.8$, $s_3(0) = 2$

For link $(3,4)$

$e = \sum_{r \in \mathcal{R}_{34}(0)} \big(\tau^r_{34}(0) + e_4(0 + \tau^r_{34}(0))\big)\, p^r_{34}(0)$

$= \big(1 + e_4(0+1)\big)\,0.5 + \big(3 + e_4(0+3)\big)\,0.5 = (1+0.0)\,0.5 + (3+0.0)\,0.5 = 2.0$

Since $e < e_3(0) = 5.8$, $e_3(0) = 2.0$, $s_3(0) = 4$

The minimum expected travel times from all nodes for all departure times are shown in Table C.1. In this example, all minimum expected travel time next-arc hyperpaths except three turn out to be simple paths. The three minimum expected travel time next-arc hyperpaths that are not simple paths are depicted in Figures C-1–C-3.

Table C.1: Results

Origin Node (i)	Departure Time (t)	Min. Expected Time ($e_i(t)$)	Successor ($s_i(t)$)
1	0	3.7	2
	1	3.8	3
	2	3.6	3
	3	4.6	3
	4	6.2	2
	5	5.9	2
	≥ 6	5.9	2
2	0	2.4	4
	1	2.9	3
	2	1.6	4
	3	3.0	4
	4	3.2	4
	5	2.5	4
	≥ 6	2.5	4
3	0	2.0	4
	1	2.8	4
	2	1.5	4
	3	1.4	4
	4	3.7	2
	5	2.4	4
	≥ 6	2.4	4

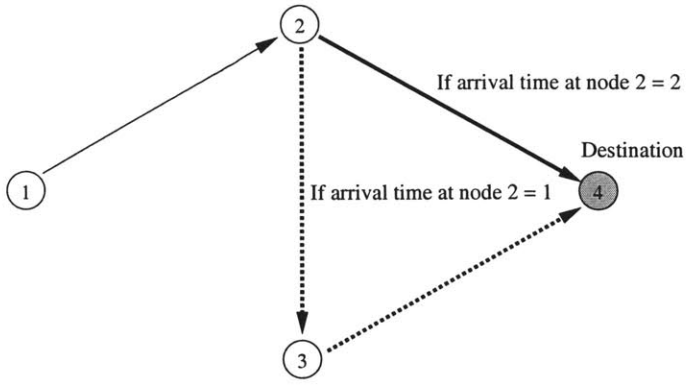

Figure C-1: Minimum Expected Travel Time Next-Arc Hyperpath from Node 1 to Node 4 at Departure Time 0

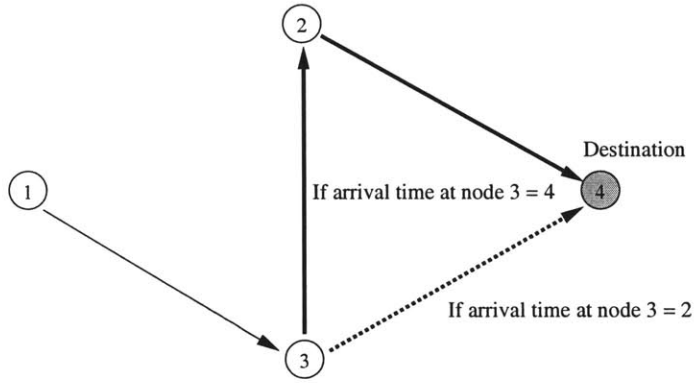

Figure C-2: Minimum Expected Travel Time Next-Arc Hyperpath from Node 1 to Node 4 at Departure Time 1

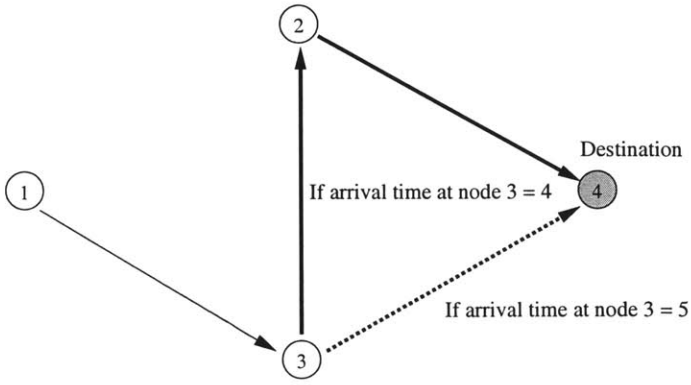

Figure C-3: Minimum Expected Travel Time Next-Arc Hyperpath from Node 1 to Node 4 at Departure Time 3

Bibliography

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows.* Prentice-Hall, 1989.

[2] D. Bertsekas. *Network Optimization: Continuous and Discrete Models.* Athena Scientific, 1998.

[3] I. Chabini. Discrete Dynamic Shortest Path Problems in Transportation Applications: Complexity and Algorithms with Optimal Run Time. *Transportation Research Record*, 1645:170–175, 1998.

[4] I. Chabini and B. Dean. Shortest Path Problems in Discrete-Time Dynamic Networks: Complexity, Algorithms and Implementations. Internal Report, Department of Civil and Environmental Engineering, MIT, 1999.

[5] K. Cooke and E. Halsey. The Shortest Route Through a Network with Time-Dependent Internodal Transit Times. *Journal of Mathematical Analysis and Applications*, 14:492–498, 1966.

[6] G. Corea and V. Kulkarni. Shortest Paths in Stochastic Networks with Arc Lengths Having Discrete Distributions. *Networks*, 23:175–183, 1993.

[7] N. Deo and C. Pang. Shortest-Path Algorithms: Taxonomy and Annotation. *Networks*, 14:275–323, 1984.

[8] S. Dreyfus. An Appraisal of Some Shortest-Path Algorithms. *Operations Research*, 17:395–412, 1969.

[9] A. Eiger, P. Mirchandani, and H. Soroush. Path Preferences and Optimal Paths in Probabilistic Networks. *Transportation Science*, 19(1):75–84, 1985.

[10] H. Frank. Shortest Paths in Probabilistic Graphs. *Operations Research*, 17:583–599, 1969.

[11] L. Fu and L. Rilett. Expected Shortest Paths in Dynamic and Stochastic Traffic Networks. *Transportation Research-B*, 32(7):499–516, 1999.

[12] R. Hall. The Fastest Path through a Network with Random Time-Dependent Travel Times. *Transportation Science*, 20(3):182–188, 1986.

[13] D. Kaufman and R. Smith. Fastest Paths in Time-Dependent Networks for Intelligent Vehicle-Highway Systems Application. *IVHS Journal*, 1(1):1–11, 1993.

[14] V. Kulkarni. Shortest Paths in Networks with Exponentially Distributed Arc Lengths. *Networks*, 16:255–274, 1986.

[15] R. Larson and A. Odoni. *Urban Operations Research.* McGraw-Hill, 1982.

[16] E. Miller-Hooks. *Optimal Routing in Time-Varying, Stochastic Networks: Algorithms and Implementation.* Ph.D. dissertation, Department of Civil Engineering, The University of Texas at Austin, 1997.

[17] E. Miller-Hooks and H. Mahmassani. Least Possible Time Paths in Stochastic, Time-Varying Networks. *Computers and Operations Research*, 25(12):1107–1125, 1998.

[18] E. Miller-Hooks and H. Mahmassani. Least Expected Time Paths in Stochastic, Time-Varying Transportation Networks. *Transportation Science*, 34(2):198–215, 2000.

[19] P. Mirchandani and H. Soroush. Optimal Paths in Probabilistic Networks: A Case with Temporary Preferences. *Computers and Operations Research*, 12(4):365–381, 1985.

[20] S. Nguyen and S. Pallottino. Hyperpaths and Shortest Hyperpaths. In B. Simeone, editor, *Combinatorial Optimization*, number 1403 in Lecture Notes in Mathematics, pages 258–271. Springer-Verlag, Berlin, 1986.

[21] S. Opasanon and E. Miller-Hooks. Least Expected Time Hyperpaths in Stochastic, Time-Varying Multimodal Networks. *Annual Meeting of the Transportation Research Board*, 2001.

[22] S. Pallottino and M. Scutellà. Shortest Path Algorithms in Transportation Models: Classical and Innovative Aspects. In P. Marcotte and S. Nguyen, editors, *Equilibrium and Advanced Transportation Modelling*, pages 245–281. Kluwer, 1998.

[23] D. Pretolani. A Directed Hypergraph Model for Random Time Dependent Shortest Paths. *European Journal of Operations Research*, 123:315–324, 2000.

[24] C. Sigal, A. Pritsker, and J. Solberg. The Stochastic Shortest Route Problem. *Operations Research*, 28(5):1123–1129, 1980.

[25] B. Yang and E. Miller-Hooks. The Adaptive Routing Problem in Signalized Networks. *Annual Meeting of the Transportation Research Board*, 2001.

[26] A. Ziliaskopoulos and H. Mahmassani. Time-Dependent, Shorest-Path Algorithm for Real-Time Intelligent Vehicle Highway System Applications. *Transportation Research Record*, 1408:94–100, 1993.