

OPTIMAL ROUTING THROUGH STOCHASTIC NETWORKS

by

Yueyue Fan

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(CIVIL ENGINEERING)

May 2003

Copyright 2003

Yueyue Fan

UMI Number: 3103886

UMI[®]

UMI Microform 3103886

Copyright 2003 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

UNIVERSITY OF SOUTHERN CALIFORNIA
THE GRADUATE SCHOOL
UNIVERSITY PARK
LOS ANGELES, CALIFORNIA 90089-1695

This dissertation, written by

Yueyue Fan

*under the direction of h_{er} dissertation committee, and
approved by all its members, has been presented to and
accepted by the Director of Graduate and Professional
Programs, in partial fulfillment of the requirements for the
degree of*

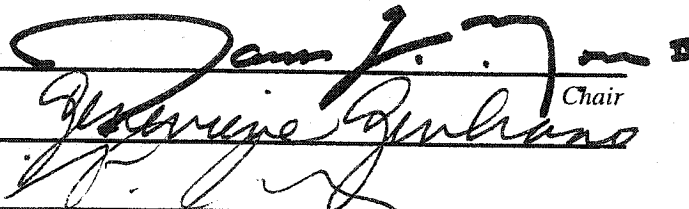
DOCTOR OF PHILOSOPHY



Director

Date May 16, 2003

Dissertation Committee



Chair

R. E. Kalaba

ACKNOWLEDGEMENTS

I have been very fortunate to have many excellent faculty mentors during the course of my education. They have made great contributions to my intellectual growth. In particular, I would like to thank my dissertation committee members Prof. James Moore, Prof. Genevieve Guiliano, Prof. Peter Gordon, and Prof. Robert Kalaba for their continuous support and very insightful comments. My dissertation could not have been begun or completed without their help. Prof. Kalaba served on my committee as an outside member. However, he is the one who is most involved in my dissertation research. The only way I can reward him for all of the effort he has made educating me is to keep myself productive and on the path to the academic life.

I do not think I have a very large number of friends, but I certainly have the sincerest friends one could ever hope for. These are the sort of friends who care for me no matter who I am and where I am. Their friendship has never been constrained by geography or time. They are Wang Yi in Sapporo, Xue Wei, Hu Yanli, and Li Chunhua in Beijing, Li Hailing and Zhou Zhenjun in Shanghai, Wang Yingjun in Dalian, and Wu Pei-Chen in Taiwan. I would also like to thank my friends in Los Angeles for their pleasant company, especially Yan Xu, Lei Huang, and Leland Farrar.

Finally, I want to give my deepest appreciation to my parents for every thing I have acquired from them, both from their nature and their nurture. "Keep ambitious goals in your mind, but try to reach them step by step," is one of the many

pieces of advice my father gave me when I was young. I have been following his advice, and continue to do so.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	ii
LIST OF TABLES	vi
LIST OF FIGURES	vii
ABSTRACT	viii
CHAPTER 1 INTRODUCTION.....	1
1.1 MOTIVATION.....	1
1.2 RESEARCH DESIGN	4
1.3 CONTRIBUTION	9
1.4 OUTLINE OF THE DISSERTATION	10
CHAPTER 2 LITERATURE REVIEW.....	12
2.1 PATH-FINDING PROBLEMS	12
2.2 BELLMAN'S PRINCIPLE OF OPTIMALITY AND DYNAMIC PROGRAMMING	17
2.3 PICARD'S METHOD OF SUCCESSIVE APPROXIMATION.....	24
2.4 EVALUATION OF CONVOLUTION INTEGRALS AND INTEGRAL EQUATIONS.....	26
CHAPTER 3 ARRIVING ON TIME.....	31
3.1 PROBLEM STATEMENT	31
3.2 MATHEMATIC MODEL.....	31
3.3 NUMERICAL SOLUTIONS	34
3.4 NUMERICAL EXAMPLES	43
3.5 CHANGING TIME SCALE.....	65
3.6 ALTERNATIVE METHOD OF EVALUATING CONVOLUTION INTEGRALS VIA DIFFERENTIAL APPROXIMATION	74
3.7 SUMMARY.....	86
CHAPTER 4 OPTIMAL ROUTING THROUGH NETWORKS IN THE CASE OF CORRELATED LINK TRAVEL TIMES	90
4.1 STATEMENT OF THE PROBLEM	91
4.2 FORMULATION OF THE PROBLEM	93
4.3 NUMERICAL SOLUTION – PICARD'S METHOD OF SUCCESSIVE APPROXIMATION.....	98
4.4 NUMERICAL EXAMPLES: APPLICATIONS TO SMALL NETWORKS	105
4.5 SUMMARY.....	121
CHAPTER 5 CONCLUSIONS AND DISCUSSION.....	122
BIBLIOGRAPHY	129

APPENDIX A	132
APPENDIX B.....	134
APPENDIX C	136
APPENDIX D	142
APPENDIX E.....	174
APPENDIX F.....	206
APPENDIX G	214
APPENDIX H	221
APPENDIX I.....	223
APPENDIX J	224

LIST OF TABLES

Table 3-1: True versus Estimated Values of Laplace Transforms and Their Inverses	45
Table 3-2: Probabilities of Arriving on Time at Destination Node $N = 5$	48
Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ within Time t or Less Given an Optimal Choice of Successor Nodes	53
Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$ within Time t or Less Given an Optimal Choice of Successor Nodes Using the Second Set of Gamma Distributions	58
Table 3-5: Numerical Comparison of $\sin(at)$ with Inverted Laplace Transforms of $\sin(at)$	69
Table 3-6: Numerical Comparison of MATLAB Quad Function Estimates of $y(t) = f(at)$ with Estimates Obtained via the Multiplicative Property of Laplace Transforms	71
Table 3-7: Probabilities of Arriving on Time at Destination Node $N = 9$ within Time t : Increasing the Number and Range of Time Points by Changing the Time Scale	73
Table 3-8: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation	81
Table 3-9: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation	85
Table 4-1: Minimum Expected Travel Time from Each Origin Node to Destination Node 4, and Optimal Choice of Successor Nodes	107
Table 4-2: Minimum Expected Travel Time from Each Origin Node to Destination Node 9, and Optimal Choice of Successor Nodes	108
Table 4-3: Average Uncongested Travel Times for Links in the 49-Node Network	112
Table 4-4: Minimum Expected Travel Time from Each Origin Node to Destination Node 49, and Optimal Choice of Successor Nodes	115

LIST OF FIGURES

Figure 2-1: Solving the Equation $f(x) = x$ Using Picard's Method of Successive Approximation	25
Figure 3-1: Values of t and ω for the Double Integral $L(h(t))$	37
Figure 3-2: A Small Network With Probabilistic Link Travel Costs.....	46
Figure 3-3: A 49-node Network.....	50
Figure 3-4: Changing the Time Scale Parameter a for $\sin(at)$	68
Figure 3-5: Changing Time Scale for Function of Convolution Integral.....	70
Figure 3-6: A 9-node Network.....	72
Figure 3-7: Maximum Probability of Arriving On Time from Each Node Using Changing Time Scale Procedure.....	74
Figure 3-8: Procedure of Solving Ordinary Differential Equations Using Modified Euler's Method.....	79
Figure 3-9: A 4-node Network with Same Link Travel Time Distributions	80
Figure 3-10: Maximum Probability of Arriving On Time from Node 2 to	83
Figure 3-11: A 4-node Network with Different Link Travel Time Distributions.....	84
Figure 3-12: The Maximum Probability of Arriving on Time using the Method of Differential Approximation.....	87
Figure 4-1: A General Sub-Network.....	93
Figure 4-2: A Four-Node Network	105
Figure 4-3: A Nine-node Network	107
Figure 4-4: A 49-node Network.....	109
Figure 4-5: Optimal Successor Nodes in the 49-node Network	119
Figure 4-6: A Cut Across the Links to the Left of Nodes 3, 9, 10, 15 and 16	120

ABSTRACT

Two stochastic routing problems are included in this document. The first addresses the maximum probability problem. The second addresses correlation problems. Both problems are significant, but have not been studied well in the literature. The maximum probability problem is constructed to capture the requirements of arriving at the destination on time or earlier. Consider a network with N nodes and various links connecting the nodes. Travel times over different links are treated as independent random variables. The probability density functions of link travel times are assumed to be known a priori. Given a traveler's current location at node i , we seek for the next successor node so that the probability of arriving at the destination node N on time or earlier is maximized. This is a stochastic on time arrival problem (SOTA).

The second problem addresses the shortest path problem in congestible networks with correlated link travel costs. The objective is to minimize expected travel time. In this problem, each link is assumed to be in one of two possible states, congested or un-congested. Conditional probability density functions for link travel times are assumed known for each state. The traveler takes into account his experience on the link leading to the current decision point (node) when determining the next node to go to.

These two problems are both formulated using Bellman's principle of optimality and solved through Picard's method of successive approximations. Some applied mathematics techniques are applied to improve the computational efficiency

of our numerical procedures. Our numerical solution schemes have been tested in various contexts, and proved to be reliable and efficient.

CHAPTER 1 INTRODUCTION

1.1 Motivation

Optimal routing problems are extensively studied in the fields of computer science, operations research, and transportation engineering. They are of importance in the routing of airplanes, trucks, and automobiles through transportation networks and of messages through communication networks. More generally, they are also of significance in the study of optimal system control, where we interpret nodes as the possible states of a system and the links as transformations from one state to another. Frequently, there is interest in transforming a system from a given initial state to a desired terminal state in an optimal manner.

Many previous studies have focused on shortest path problems, including deterministic and stochastic applications. The objectives are usually to minimize time, cost, or maximize utility. However, link travel times are random in most situations. Under uncertainty, there may be, for any origin and destination pair, more than one path that has some positive probability of being shortest. Some links may have a low expected travel time, yet have poor reliabilities. How might a model be constructed if punctuality of arrival or delivery is the primary goal? Interest in these problems has arisen from empirical experiences, as in transit and freight operations where missing a designated arrival time can be very expensive. However, these problems have not been studied well. One reason may be that this class of problems

usually requires probability density functions to be modeled explicitly, which makes it very computationally burdensome to obtain the solutions.

The first optimal routing problem in this document, the arriving on time problem, is constructed to capture the requirements of arriving at the destination on time or earlier. Consider a network with N nodes and various links connecting the nodes. Travel times over different links are treated as independent random variables. The probability density functions of link travel times are assumed to be known a priori. Given a traveler's current location at node i , what should be the next successor node so that the probability of arriving at the destination node N on time or earlier is maximized? This is a stochastic on time arrival problem (SOTA). This problem is formulated using Bellman's principle of optimality and solved through Picard's method of successive approximations. Some applied mathematics techniques are applied to improve the computational efficiency. Refer to Chapter 3 for a detailed description of the procedure of formulating and solving such problems. Although there are a couple of papers considering maximum likelihood in the literature of routing problems, no implementation has been given to our knowledge. The major contribution of this part of research is formulating such problems and providing reliable and efficient numerical implementations.

As in the SOTA problem, arc weights (link travel times, in the transportation context) are assumed to be independent in almost all of the literature describing network problems. However, this is not necessarily true in reality. In a transportation network, performance may be affected by non-recurrent events, such

as (in the extreme case) natural disasters or (more commonly) traffic incidents. These random reductions in supply make network performance uncertain. If the level of service of one transportation link is affected, it is very likely that the adjacent links are also affected. Burton (1993), proceeding from a mathematician's point of view, identified the necessity of handling correlations between arc weights in the applications of seismic wave propagation and traffic modeling. To our knowledge, there has been no other study addressing the correlation issue. How should the correlations between adjacent link conditions be accounted for? How should knowledge learned through the trip be used to support the remaining decisions?

The second optimal routing problem in this document addresses the shortest path problem in congestible networks with correlated link travel costs. The objective is to minimize expected travel time. In this problem, each link is assumed to be in one of two possible states, congested or un-congested. Conditional probability density functions for link travel times are assumed known for each state. The traveler takes into account his experience on the link leading to the current decision point (node) when determining which node to visit next, i.e., which link to traverse next. Only the next immediate link in the path is identified, and the knowledge gained during the course of the trip is used to optimally support the decisions defining the remaining journey. Bellman's principle of optimality is applied to formulate the basic mathematic model. Proof of existence and uniqueness of the solution and a procedure of obtaining the solution are provided. Refer to Chapter 4

for details. The major contribution of this part of the research is in identifying and formulating routing problems with consideration of correlation between links, and mathematically proving the some of the properties of the equations.

1.2 Research Design

Both routing problems involve multi-stage decisions. Dynamic programming offers a way to mathematically formulate this type of problem. Picard's method of successive approximation can be used to solve the resulting equations.

1.2.1 Arriving-on-time problem

Our primary goals in this research are to formulate the equations that capture the maximum probability requirement and to seek for a feasible solution scheme. Instead of minimizing expected total travel time as in the conventional shortest path problems, the objective is set to be maximizing the probability of arriving on time or earlier. Only the next immediate link needs to be identified at each decision point (node). As mentioned earlier, maximum probability problems have not been studied well. This class of problems usually requires probability density functions to be modeled explicitly. Even if a formulation of such problems is available, obtaining the solution to the equations can be difficult. Therefore, besides the two primary research goals, checking the accuracy of our procedure and studying the behavior of the unknown functions are also our major concerns. This is important especially in situations where no previous studies are available for comparison purposes.

The problem can be stated as the following. Label the nodes in a network $1, 2, \dots, N$. The travel times over any two links are assumed to be independent random variables. Probability density functions of link travel time on any link ij are known as $p_{ij}(\omega)$. These are assumed to be a priori knowledge. Given a traveler's current location at node i , what is the next node to visit to maximize the probability of arriving at the destination node N within time t or less? This is the stochastic on-time arrival problem (SOTA).

The original problem could be from one specific node to the destination. To handle this problem, we may imbed that problem within the class of problems in which we start at any node i and wish to arrive at node N in time t or less. Once the problem is imbedded, we can make use of the Bellman's principle of optimality. In general, the first step is to define an optimal return function. Then, Bellman's principle of optimality shall be used to formulate the relationship among the return functions.

Define $u_i(t)$ as the maximum probability of starting at node i and arriving at the destination node N with time t or less. In the manner describe above, the SOTA problem is formulated as

$$u_i(t) = \max_{j \neq i} \int_0^t p_{ij}(\omega) u_j(t - \omega) d\omega, \quad i = 1, 2, \dots, N-1, \quad (1.1a)$$

$$u_N(t) = 1; \quad (1.1b)$$

where

$p_{ij}(\omega)d\omega$ = the probability of traversing the direct link ij within time ω and $\omega + d\omega$; and

$u_i(t)$ = the probability that, starting from node i , the traveler arrives at node N within time t or less when an optimal sequence of choices is made, $i = 1, 2, \dots, N, 0 \leq t < \infty$

The functions $u_i(t)$, $i = 1, 2, \dots, N$ and $0 \leq t \leq \infty$, are the optimal return functions. The next node to visit starting from each node i with time t remaining is the decision variable to be determined. The solution gives $u_1(t), \dots, u_N(t)$. It also determines for each node i the value of j that does the maximizing. This is the correct next node to go to.

The detailed rational of the formulation is given in Chapter 3. Once the problem is formulated, the following questions arise. First, can the solution to the problem be uniquely defined? And second, how might the solution be obtained? These questions will be discussed in detail in Chapter 3.

1.2.2 Shortest path problem in networks with correlated link travel times

The second part of the research, described in Chapter 4, treats the shortest path problem in networks with correlated link service levels.

This problem can be modeled using a node-based definition of congestion or a link-based definition of congestion. Given the node-based definition of congestion, each node is assumed to have two possible states, congested or uncongested. In the case of a natural disaster, these states might be more generally labeled “affected” and

“unaffected.” The correlations between the states of adjacent nodes are taken into account by introducing two probabilities, α_{ij} and β_{ij} . The probability that, if node i is uncongested, then node j is uncongested is α_{ij} . The probability that if, node i is congested, then node j is congested is β_{ij} . For the sake of subsequent notational convenience, define λ_{ij} as the probability that, if node i is congested, then node j is uncongested, i.e., $1 - \beta_{ij}$. These probabilities are similar to but less restrictive than the *a priori* correlations assumed by Burton (1993). They are assumed to be *a priori* knowledge in this problem.

Similarly, in link-based congestion, each link is assumed to have two possible states, congested or uncongested. A link is considered uncongested if the time required to traverse this link ij is within an *a priori* bound t_{0ij} , and considered congested otherwise. The distributions of link travel times associated with each state are described by known probability density functions. The average link travel times between node i and node j is t_{ij} under uncongested conditions and τ_{ij} under congested conditions.

Experience from the previous link leading to the current decision node is taken into account when making the optimal decision for the remaining journey. That is, if a traveler experiences congestion at the current node or on the current link, he applies the corresponding conditional probability or probability density function to structure his decision about next successor node. The objective is to minimize the expected travel time. As in the SOTA problem, the goal is to identify the next immediate link to use, rather than identifying the entire path.

Similarly as in the SOTA problem, the original problem could be from one specific node to the destination. We may imbed that problem within the class of problems, in which we start at any node i and wish to arrive at the destination node N in minimum expected time. This imbedding makes it possible to formulate the problem using Bellman's principle of optimality. Optimal return variables have to be defined and the relationship between these return variables shall be described based on Bellman's principle of optimality. Define u_i as the minimum expected cost from node i to the destination node N if the link leading to node i is uncongested. Define v_i as the minimum expected cost from node i to the destination node N if the link leading to node i is congested. The variables u_i and v_i are the optimal return variables. The problem is formulated

$$u_i = \min_{j \neq i} \{t_{ij} + \alpha_{ij}u_j + (1 - \alpha_{ij})v_j\}, i = 1, 2, \dots, N-1, \quad (1.2a)$$

$$v_i = \min_{j \neq i} \{\tau_{ij} + \lambda_{ij}u_j + (1 - \lambda_{ij})v_j\}, i = 1, 2, \dots, N-1, \quad (1.2b)$$

and

$$u_N, v_N = 0, \quad (1.2c)$$

where

t_{ij} = the expected travel times on link ij if the link leading to node i is uncongested,

τ_{ij} = the expected travel times on link ij if the link leading to node i is congested,

α_{ij} = the transition probability that link ij is uncongested if the link leading to node i is uncongested, and

λ_{ij} = the transition probability that link ij is uncongested if the link leading to node i is congested.

The next node to visit starting from each node i given the previous link status is the decision variable to be determined. The solution gives u_1, \dots, u_N , and v_1, \dots, v_N . It also determines for each node i the value of j that does the maximizing of u_i and v_i . This is the correct node to visit from node i conditioned on the previous link status.

The detailed rational of the formulation is given in Chapter 4. Equations (1.2a-c) are a new set of equations. The following questions arise from the mathematic viewpoint. First, does a solution to these equations exist? Second, is the solution unique? And third, how should the solutions be pursued? These questions will be discussed in detail in Chapter 4.

1.3 Contribution

Two stochastic routing problems are included in this document. The first addresses the maximum probability problem. The second addresses correlation problems. Both problems are significant, but have not been studied well in the literature.

We have successfully formulated the problem considering maximum probability of on time arrival and provided an efficient numerical solution scheme. Our experience has indicated that even if a formulation of such problem is available,

obtaining the solution requires a great deal of mathematical effort. Our numerical solution scheme has been tested in various contexts. It is shown that the numerical scheme is reliable and efficient. To our knowledge, it is the first time that an applicable implementation of such problems is provided.

The formulation of the shortest path problem in networks with correlated travel costs is completely novel. The major contribution of this part of research is defining and formulating the problem, and proving some of the properties of the new set of equations.

1.4 Outline of the Dissertation

The remainder of this document consists of a report on the research summarized above. It includes, in addition, a summary of the literature that provides the perspective on which this research is based or related. The document is organized as follows:

Chapter 2

Chapter 2 consists of a literature review that has been classified into sections devoted to mathematic formulations and numerical computation of the path-finding problems. The applicability of some relevant applied mathematics techniques, such as solution of convolution integral equations, dynamic programming, and Picard's method of successive approximation, are also discussed.

Chapter 3

Chapter 3 is dedicated to the on-time-arrival problem. The various sections of Chapter 3 describe the formulation of the problem, the procedures of obtaining numerical solutions to the problem, and the validation of the presented numerical approaches.

Chapter 4

Chapter 4 is dedicated to the stochastic shortest path problem considering correlated link travel times. The various sections of Chapter 4 describe the formulation of the problem, the mathematic proof of the nature of the solutions to this problem, and the numerical procedures for obtaining such solutions.

Chapter 5

Chapter 5 summarizes what has been achieved in this research. The questions and problems that we have observed during our computational experiences are presented. Possible extensions for future research are provided at the end.

CHAPTER 2 LITERATURE REVIEW

The literature on the path-finding problems presented here is relevant. However, it does not directly lead to the research presented in this dissertation. The primary objective of this chapter is to provide a general view of what has been studied, and more importantly, what has not been addressed in the current literature of path-finding problems.

Besides the path-finding problems themselves, some applied mathematics techniques will also be described and further discussed together with the applicability and limitations. These include Bellman's principle of optimality and dynamic programming, Picard's method of successive approximation, evaluation of convolution integrals and solution of integral equations. These techniques are essential in this research in formulating the problems, solving the nonlinear equations, and improving the computational efficiency.

2.1 Path-Finding Problems

Various problems of finding optimal paths have been studied extensively in the fields of computer science, operations research, and transportation engineering. Depending on the application, the objective may be expressed in terms of cost, time, reliability, uncertainty, or a combination of multiple criteria (Loui, 1983; Eiger, Mirchandani and Soroush, 1985; Mruthy and Sarkar, 1996, 1998). The shortest path problem and k -shortest paths problems are the most intensively studied. Initial formulations involved only deterministic models, for which many efficient

algorithms have been developed (Bellman, 1958; Dijkstra, 1959 and Dreyfus, 1969). Excellent reviews of the earlier work have been done by Pollack and Dreyfus (Pollack, 1961; Dreyfus, 1969). The ordinary shortest path problem only involves looking for the best path, while the k -shortest path problems also look for the suboptimal paths. The ordinary shortest path algorithms can be categorized as tree-building algorithms and matrix algorithms. In tree-building algorithms, the shortest path between one node and many nodes is usually obtained, while in matrix algorithms, shortest paths between all nodes are found. Moore (1957), Ford (1956), and Bellman (1958) published similar tree-building algorithms, which are widely used in network studies. Those algorithms can be improved by applying the indexing system of d'Esopo (Pollack and Wiebenson, 1960). The first 'once through' algorithms were published by Dijkstra (1959). A sorting procedure was introduced by Dial (1969) to improve the efficiency of the 'once through' algorithm. Algorithms of Floyd (1962) and Dantzig (1966) are two excellent matrix algorithms for finding paths between all nodes. For large networks, a tree-building algorithm is much more efficient with respect to computer storage than a matrix algorithm (Steenbrink, 1973). There are also heuristic procedures for cases when a good solution is acceptable and the best solution is not required. Mill (1968) proposed a heuristic tree-building algorithm for finding the shortest path between two nodes. In this algorithm, less promising routes are not inspected further. Node aggregation (Jansen, 1971) and network partitioning (Hu, 1968) are used for simplifying the computations.

According to Dreyfus, the first efficient algorithm that calculates the k -shortest paths (allowing loops) was developed by Hoffman and Pavley (1959). This procedure is efficient only when k is small, especially so when k is 2. Bellman and Kalaba (1960) proposed an excellent and entirely different procedure, which outperforms the Hoffman and Pavley algorithm when more than two paths are sought. Dreyfus (1969) suggested a procedure based on both of the two previous procedures, which seems more computationally efficient. However, no detailed implementation was provided.

Many scholars contributed to implementation of k -shortest paths algorithms. Shier and Minieka (Minieka, 1974; Minieka and Shier, 1973; Shier, 1974 and 1979) have made significant contributions. A general bibliography with more than 300 references in k -shortest paths problem is available at the website <http://www.ics.uci.edu/~eppstein/bibs/kpath.bib>.

However, randomness caused by time-of-day variations in demand, accidents, construction, or disasters makes it difficult to predict network performance. Under such stochastic circumstances, random link travel times and their associated probability density functions should be considered explicitly. Expected shortest path problems in networks of a stochastic nature (Loui, 1983; Murthy, 1996), a time-dependent nature (Dreyfus, 1969; Kaufman, 1993), or both (Hall, 1988; Fu, 1998, 2001; Miller-Hooks and Mahmassani, 2000) have been also studied. Loui (1983) pointed out that the stochastic shortest path problem can be reduced to the deterministic shortest path problem by taking expectations of link

travel times and solving the resulting deterministic problem. In the time-dependent stochastic shortest path problems, the link travel times are usually given by time-dependent, random variables with probability distribution functions that are known a priori. The problems are to determine the least expected cost path from each node i to the given destination N for each departure time t . Almost all the time-dependent studies consider a set of discrete probabilities associated with a number of possible link travel times that may occur during a given time interval. Continuous distributions are rarely included to keep the formulation simple.

However, the identified expected shortest paths can be risky. An apparent optimal path may have low expected travel times, yet poor reliability. A path of slightly greater expected travel time, but with little probability of realizing very large travel times may be preferable in some situations. A few scholars have tried to construct different models to best account for risk. Frank (1969) proposed considering the path that maximizes the probability of realizing a weight less than k as the optimal path. This research provides an analytical formulation of calculating the probability distribution of the shortest path. The formulation is based on the assumption that the joint probability density functions of all arc weights are known. Even if these functions are available, integration over all the arc weights required by the formulation can be too complicated to proceed. It seems that Monte Carlo simulation is the only feasible way to evaluate the probability distribution of a shortest path proposed in this paper. Also, the algorithm only evaluates the probability distribution of a shortest path. It does not provide an optimal routing

strategy that leads to the shortest path. Sigal (1980) suggested considering the path that has the greatest probability of realizing the least weight. Again, no implementation has been provided.

How best to describe the random variables involved in shortest path problems is unclear. Most studies ignore the higher moments of random variables, and consider only their expectations. There are a few scholars that have considered both expectations and higher moments of travel times. The statistic measurements (means and variances) from historic traffic data are applied to describe the random nature of the network. Estimating the probability density functions for link variables may be burdensome. Even if the probability density functions for link variables are available, obtaining solution to the models involving the density functions may be difficult (Fu, 1998, 2001).

The large literature on path finding problems reveals two points. First, the objectives are usually minimum time/cost or maximum utilities. Maximum likelihood/reliability problems are seldom studied. This class of problems usually requires probability density functions to be modeled explicitly. This requirement makes it computationally burdensome to obtain the solutions. However, interest in these problems has arisen from empirical experience, especially in situations where punctuality of arrival or delivery is critically important. How do we construct models that capture such requirements?

Second, arc weights are assumed to be independent in almost all of the literature on path finding problems. In his Ph.D dissertation on the inverse shortest

path problem, Burton (1993), proceeding from a mathematician's point of view, identified the necessity of handling correlations between arc weights in the applications of seismic wave propagation and traffic modeling. Correlated arcs are aggregated into classes. The weights of the arcs in the same class are derived from a common value, which is referred to as the class's density. This new formulation reduces the number of variables, but involves more constraints. To our knowledge, there has been no other study addressing the correlation questions. In transportation networks, if the level of service of one transportation link is affected, it is very likely that the adjacent links are also affected. How should the correlations between adjacent link conditions be accounted for? How should knowledge learned through the trip be used to support remaining decisions?

2.2 Bellman's Principle of Optimality and Dynamic Programming

Bellman's principle of optimality is the most essential principle in this research. It is used to formulate the equations in both routing problems discussed here. The principle states that

"An optimal sequence of decisions has the property that whatever the initial state and decision are, the remaining decisions must be optimal with respect to the state resulting from the initial decision" (Bellman and Kalaba, 1965).

This principle is widely used to model systems involving multi-stage decisions.

The original problems could be from one specific stage to the final stage. To handle these problems, we may imbed that problems within the class of problems in

which we start at any stage i and wish to arrive at the final stage in an optimal manner. Once the problem is imbedded, we can make use of the Bellman's principle of optimality. In general, the first step is to define an optimal return function or variable. Then, Bellman's principle of optimality is used to formulate the relationship of the return functions or variables.

We shall give examples below of formulating some simple path-finding problems using Bellman's principle of optimality. The purpose is to help the reader to become familiar with dynamic thinking and therefore be prepared for the more complicated formulations occurring in the major part of this dissertation.

2.2.1 Formulating the deterministic shortest path problem using Bellman's principle of optimality

Consider a network with N nodes and various links connecting these nodes. In this problem, the shortest path with minimum time cost from any origin node i to the destination node N is sought. Link travel time on any direct link from node i to node j is known as a constant. Let the quantity

$$t_{ij} = \text{the link travel time over the direct link from node } i \text{ to node } j. \quad (2.1)$$

Introduce the unknown variable

$$T_i = \text{the minimum time cost required from node } i \text{ to the destination node } N. \quad (2.2)$$

These are the optimal return variables to be determined. For a traveler who is currently at node i , no matter which node he chooses to visit next (label as node j),

once he is at node j , the best return he can achieve is the minimum travel cost from that node j to the destination node N , which is T_j by definition. He may have several choices for node j . To optimize the entire journey, he should choose the one that leads to the minimum cost from node i to the destination node N . When the traveler is already at node N , the minimum total cost is always zero. The mathematical formulations are therefore

$$T_i = \min_{j \neq i} \{t_{ij} + T_j\}, \quad i = 1, 2, \dots, N-1, \quad (2.3a)$$

$$T_N = 0. \quad (2.3b)$$

The solution gives T_1, \dots, T_N . It also determines for each node i the value of j that does the minimizing. This is the correct next node to visit.

2.2.2 Formulating the stochastic shortest path problem using Bellman's principle of optimality

Consider a network with N nodes and various interconnecting links.

Introduce the function

$$p_{ij}(t) = \text{the probability density function of link travel time on the direct link from node } i \text{ to node } j. \quad (2.4)$$

These probability density functions are assumed to be a priori knowledge. Link travel times on any two different links are assumed to be independent random variables. The objective is to identify a path that leads to the minimum expected travel cost from node i to the destination node N . Let us introduce the unknown

T_i = the minimum expected time cost required from node i to the destination node N . (2.5)

Suppose a traveler is currently at node i , and chooses to visit next a node j (node j can be any node except for node i that has a direct connection to node i). The probability that it takes a time between ω and $\omega+d\omega$ to traverse link ij , by definition, is $p_{ij}(\omega)d\omega$, where $d\omega$ is sufficiently small. The quantity ω can be any number between 0 and infinity. There may be several options for the next node j . No matter which node j he chooses to visit next, once he is at that node j , the best return he can achieve over the remaining journey is the minimum expected travel cost from that node j to the destination node N , which is T_j by definition. He may have several choices for node j . To optimize the entire journey, he must choose the j that leads to the minimum cost from node i to the destination node N . When the traveler is already at node N , the minimum total cost is always zero. The mathematical formulations are therefore

$$T_i = \min_{j \neq i} \left\{ \int_0^\infty (\omega + T_j) p_{ij}(\omega) d\omega \right\}, \quad i = 1, 2, \dots, N-1, \quad (2.6a)$$

$$T_N = 0. \quad (2.6b)$$

The quantities T_1, \dots, T_N are the optimal return variables. The next successor node to visit from each node i is the decision variable. The solution to this problem gives T_1, \dots, T_N . It also determines for each node i the value of j that does the minimizing. This is the correct next node to go to.

Let us further study the relationship between the stochastic shortest path problem and the deterministic shortest path problem. Partitioning the integrand in Equation (2.6a), we have

$$T_i = \min_{j \neq i} \left\{ \int_0^\infty \omega p_{ij}(\omega) d\omega + \int_0^\infty T_j p_{ij}(\omega) d\omega \right\}, \quad i = 1, 2, \dots, N-1. \quad (2.7)$$

It is apparent that the first term in the parenthesis is actually the expected travel time over link ij . Introduce the quantity

$$\bar{t}_{ij} = \text{the expected travel time over the direct link } ij. \quad (2.8)$$

The second term is just T_j since T_j is independent of the variable ω . Recall that

$$\int_0^\infty p_{ij}(\omega) d\omega = 1. \quad (2.9)$$

Equation (2.7) then can be simplified to be

$$T_i = \min_{j \neq i} \{ \bar{t}_{ij} + T_j \}, \quad i = 1, 2, \dots, N-1. \quad (2.10)$$

Note that equations (2.10) and (2.3a) are equivalent. This agrees with what Loui (1983) has reported: a stochastic shortest path problem can be reduced to a deterministic problem by replacing the probability density functions describing link travel times with expectations.

So far, we have discussed the problems for shortest paths from any starting node i to the destination node N . If we wish to determine the shortest path from any starting node i to any destination node j , we could apply the procedure just discussed N times. However, there are more efficient ways to construct the model again using

Bellman's principle of optimality (Bellman and Kalaba, 1965). Let us introduce the unknown variable

$$T_{ij}^k = \text{the minimum time cost from node } i \text{ to node } j \text{ using a path with at most } k \text{ intermediate nodes.} \quad (2.11)$$

These unknowns are the optimal return variables to be determined. Based on the principle of optimality, the relationships between the optimal return variables are

$$T_{ij}^{k+1} = \min_{j \neq i} \{t_{ij} + T_{ij}^k\}, k = 0, 1, \dots, N-3. \quad (2.12)$$

For computational purpose, it may be reformulated as

$$T_{ij}^{2k+1} = \min_{j \neq i} \{T_{im}^k + T_{mj}^k\}, k = 0, 1, 3, \dots, N-3/2. \quad (2.13)$$

Equation (2.13) permits us to determine the sequence of matrices $T_{ij}^0, T_{ij}^1, T_{ij}^3, T_{ij}^7, T_{ij}^{15}, \dots$, which converges more rapidly.

The two examples of shortest path problems in deterministic and stochastic contexts helps us to better understand the shortest path problems in networks with correlated link travel times discussed in Chapter 4.

2.2.3 Formulating the maximum probability of connectivity problem using Bellman's principle of optimality

Consider a network with N nodes and various links connecting the nodes. Links in this network may be affected for some reason and become unavailable. Let us introduce the quatity

$$p_{ij} = \text{the probability that the direct link from node } i \text{ to node } j \text{ is available.} \quad (2.14)$$

These probabilities are assumed to be a priori knowledge. Our goal is to find a path with the maximum probability that there is a connection from any origin node i to the destination node N . Introduce the unknown

$$u_i = \text{the maximum probability that a connection is available from node } i \text{ to the destination node } N. \quad (2.15)$$

These unknowns are the optimal return variables to be determined in this problem.

For a traveler who is currently at node i , no matter which node he chooses to visit next (label as node j), once he is at node j , the best return he can achieve over the rest of the process is u_j . By definition, the quantity u_j is the maximum probability that there is a connection available from that node j to the destination node N . He may have several choices for node j . To optimize the entire journey, he should choose the one that leads to the maximum probability of connectivity from node i to the destination node N . Note that the link travel times on any two links are assumed to be independent and therefore multiplication of probabilities is applicable. When the traveler is already at node N , the maximum probability of connectivity is always one. The mathematical formulations are therefore

$$u_i = \max_{j \neq i} \{p_{ij} u_j\}, \quad i = 1, 2, \dots, N-1, \quad (2.16a)$$

$$u_N = 1. \quad (2.16b)$$

The quantities u_1, \dots, u_N are the optimal return variables. The next successor node to visit from each node i is the decision variable j . The solution to this problem gives u_1, \dots, u_N . It also determines for each node i the value of j that does the maximizing.

This is the correct node to go to. This example is relevant to the SOTA problem discussed in Chapter 3.

2.3 Picard's Method of Successive Approximation

Picard's method begins with initial approximations to the solution, and then refines these approximations by successive iterations. Let us illustrate how Picard's method works by solving a simple equation $f(x) = x$ for the unknown x , where $f(x)$ can be an arbitrary function. The analytical solution to this problem should be the point where the function $y = x$ and $y = f(x)$ intersect. Now let us use Picard's method to find the numerical solution. We start with the initial approximation as x_1 . The second approximation of x is chosen to be $x_2 = f(x_1)$. The third approximation of x then becomes $x_3 = f(x_2)$. Continuing in this manner, the approximations approach the limiting value, which is the true solution x^* .

Figure 2-1 shows how the sequence eventually converges to a limiting value, and this limiting value is the solution to the equation $f(x) = x$. There are two sufficient conditions to guarantee the convergence of the approximation scheme. First, the initial approximation has to be sufficient good. Second, the slope at x^* must satisfy the inequality $|f'(x)| < 1$.

Picard's method can often be used to solve a system of nonlinear equations. However, one has to first answer the following two questions before Picard's method can be applied. First, does the sequence converge to a limiting value? And second, does this limiting value satisfy the original equation? These questions arise while we

try to use Picard's method to obtain numerical solutions. We prove the convergence of the sequence by interpreting the physical meaning of the unknown variables and functions. The detailed proof is given in Chapters 3 and 4.

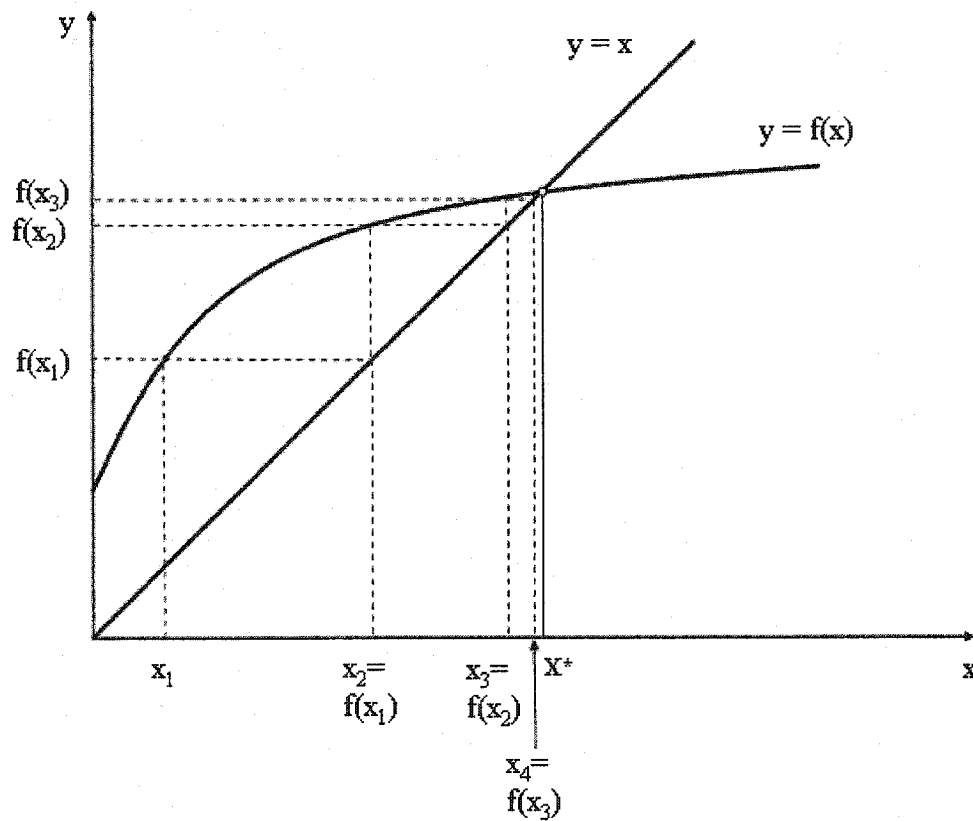


Figure 2-1: Solving the Equation $f(x) = x$ Using Picard's Method of Successive Approximation

2.4 Evaluation of Convolution Integrals and Integral Equations

Expressions in the form of $\int k(t - \tau)u(\tau)d\tau$ are called convolution integrals.

Equations with unknown functions occurring in the convolution integrand are called convolution integral equations.

The discussion about convolution integrals is relevant to the arriving-on-time problem presented in Chapter 3. Once the problem is formulated, the rest of the problem is to solve a system of nonlinear convolution integral equations.

Linear integral equations in the form

$$s(t) = \lambda u(t) + \int k(t, \tau)u(\tau)d\tau, \quad (2.17)$$

with unknown function $u(t)$ in the integrand, are called first kind if $\lambda = 0$ and second kind otherwise. Historically, integral equations are classified as:

- Fredholm, if the interval of integration in Equation (2.17) is finite and the kernel $k(t, \tau)$ is integrable;
- Volterra, if Equation (2.1) is Fredholm, and $k(t, \tau) = 0, \tau > t$;
- Cauchy Singular Equation, if $k(t, \tau) = g(t, \tau) / (t - \tau)$, where $g(t, \tau)$ is integrable.

Additional classification depends on the form of the kernel $k(t, \tau)$.

Mathematical and numerical properties of Equation (2.17) vary greatly as $k(t, \tau)$ changes. For example Equation (2.17) is called convolution if $k(t, \tau) = k(t - \tau)$, and weakly singular when $k(t, \tau)$ contains an integrable singularity. These classifications originated from analytical discussions on integral equations. From a numerical point of view, kernel's numerical behaviors, such as piecewise continuity, smoothness,

singularity, degeneracy, and linearity, are more essential (Anderson, R., Hoog, F., Lukas, M., 1980). Extensive studies on analytical and numerical solutions to integral equations exist in the current literature.

However, the equations involved in Chapter 3 differ from the general equations discussed right above in that first, there are a number of equations rather than a single equation to be solved simultaneously, and second, the system is nonlinear with respect to the unknown functions. These different natures make the well-studied approaches of solving convolution integral equations less applicable. The general approaches for solving convolution integrals are summarized below. It is our intension to encourage possible application of these approaches to the numerical procedure of solving the arriving-on-time problem.

Direct evaluation of convolution integrals in the time domain usually requires a large computational effort (proportional to the square of the number of the time stations) and storage (Paronesso and Wolf, 1995). This requirement makes it unrealistic to solve large practical problems that involve convolution integrals. Extensive research efforts have been given to reduce the computational burden of evaluating convolution integrals. Although most literature is about solving convolution integral equations, we extended our search to evaluation of convolution integrals as well, since integrals and integral equations often related to each other.

Srivastava and Buschman (1977) gave a review of analytical approaches for solving Volterra integral equation of the first kind with special function kernels. Based on the specific function kernels, equations can be solved via methods of

integral transforms, such as Mellin transformation, Laplace transformation, Hankel transformation, via successive integrations and differentiations, or via means of the resolvent kernel method and the method of fractional integration. Analytical solutions to integral equations usually depend heavily on the form of the kernel functions. Also most literature on analytical solutions emphasize the mathematical properties of such equations rather than implementation. We will pay more attention to numerical solutions in order to seek for possible implementation of a solution scheme from existing literature.

Bellman and Kalaba (1964) proposed to solve equations containing convolution integrals using the technique of differential approximation. This research suggests possible linkage between convolution integral and ordinary differential equation (ODE) problems. This method starts with a differential approximation of the kernel function. In case where the coefficients of the primary (ODE) of the kernel function are all constant, convolution integrals can be evaluated by solving a resultant differential equation with same coefficients but different constant term. With such an equation, the output could be computed directly as a linear process simulation without any further computation of integrals. If constant coefficients cannot be reached analytically, they may be sought in a least square sense. In later chapter, we will compare the estimated results of convolution integrals from method of using Laplace transforms and method of differential approximation. This exercise is for the purpose of computational validation.

Many efficient algorithms and approximation approaches have been developed for evaluating convolution integrals, given both functions in the integrand of the convolution integral are known. For example, recursive evaluation of convolution integrals in the time domain (Wolf, 1989) and the frequency domain (Wolf, 1989) were successfully implemented in problems treating unbounded medium in structure dynamics. Convolution integrals are approximated by a set of equations with unknown coefficient matrices. However, this method is only applicable for linear systems. Another remaining problem is that the matrices cannot be uniquely identified.

A quadrature rule for convolution integrals, the 'convolution quadrature method' was proposed by Lubich (1988). Approximations of $f * g(x)$ on the grid of evenly spaced time points are obtained from a discrete convolution with the value of $g(x)$ on the same grid. The quadrature weights are determined with the help of the Laplace transform of $f(x)$ and a linear multistep method. The convergence of the convolution quadrature method is of the order of the underlying multistep method. This method is suitable to the approximation of convolution integrals whose kernel $f(x)$ is singular or has components at different time-scale, and to the numerical evaluation of the expressions where the Laplace transform F of the convolution kernel is known a priori.

A thorough study on numerical evaluation of convolution integrals using Laplace transform and its inversion has been given by Bellman and Kalaba (1966). In this study, convolution integrals can be approximated by a finite sum of functions

evaluated at a number of discrete points. Problems of keeping the numerical inversion of Laplace transform stable have been observed and possible improvement has been suggested. This study is of significant relevance to our numerical procedure of evaluating convolution integrals.

CHAPTER 3 ARRIVING ON TIME

This chapter addresses the problem of maximizing the probability of arriving on time. Given a current location (node), the goal is to identify the next node to visit so that the probability of arriving at the destination node N within time t or less is maximized, given the probability density functions of link travel times. Bellman's principle of optimality is applied to formulate the mathematical model of this problem. Unknown functions describing the maximum probability of arriving on time are accurately estimated for a few sample networks by using Picard's method of successive approximations. The Laplace transform and its numerical inversion are introduced to reduce the computational cost of evaluating convolution integrals.

3.1 Problem Statement

Label the nodes in a network $1, 2, \dots, N$. The travel times over any two links are assumed to be independent random variables. Probability density functions of link travel time on any link ij are known as $p_{ij}(\omega)$. These are assumed to be a priori knowledge. Given a traveler's current location at node i , what is the next node to visit to maximize the probability of arriving at the destination node N within time t or less? This is the stochastic on-time arrival problem (SOTA).

3.2 Mathematic Model

Consider a network containing N nodes and various interconnecting links. Suppose a traveler is now at the decision node i . There may be several options for

the next possible node to visit from node i . The traveler's objective is to select the node that provides the maximum probability of arriving at the destination node N on time or earlier. The traveler must continue his choices in the optimal manner at each decision node until he reaches the destination. Bellman's principle of optimality states that an optimal sequence of decisions has the property that whatever the initial state and decision are, the remaining decisions must be optimal with respect to the state resulting from the initial decision (Bellman and Kalaba, 1965). Define $u_i(t)$ as the maximum probability of arriving at the destination N from node i within time t or less. It is the optimal return function. Consider a traveler at node i who wants to maximize the probability of arriving at the destination node N within time t or less. Suppose he chooses to visit node j next and spends time ω on link ij . The probability that he spends time between ω and $\omega+d\omega$ is $p_{ij}(\omega)d\omega$ by definition. The time left for the remaining journey is then $t-\omega$. Based on Bellman's principle, no matter which node j he chooses to go to, the traveler should at least achieve the best return from that node j to the destination within the remaining time $t-\omega$. By definition, the best return at node j is $u_j(t-\omega)$, the maximum probability of arriving at node N within time $t-\omega$ or less from the starting node j . There may be several options for the next node j at current node i , but only the one that provides the best return at the current node i should be chosen. The problem is therefore formulated as

$$u_i(t) = \max_{j \neq i} \int p_{ij}(\omega) u_j(t-\omega) d\omega, \quad i = 1, 2, \dots, N-1, \quad (3.1)$$

$$u_N(t) = 1; \quad (3.2)$$

where

$p_{ij}(\omega)d\omega$ = the probability of traversing the direct link ij within time ω and $\omega + d\omega$;

and

$u_i(t)$ = the probability that, starting from node i , the traveler arrives at node N within time t or less when an optimal sequence of choices is made, $i = 1, 2, \dots, N, 0 \leq t < \infty$

Recall that the travel times over any two links are assumed to be independent random variables. The functions $u_i(t)$, $i = 1, 2, \dots, N$ and $0 \leq t \leq \infty$, and the next nodes to visit starting from each node i are to be determined.

Suppose the traveler chooses to go from node i directly to node j . The real travel time spent on link ij will not be known until after the traveler has traversed this link. Once the traveler arrives at node j , the time left for completing the remaining journey is identified. The formulation for the problem to be solved at node j is the same as the formulation at node i , except for the change in starting node and the reduction in remaining allowable travel time. It is apparent that no fixed optimal path will be pre-determined at any node i . At each new decision node, the allowable remaining time must be updated and the resulting problem must be solved for an optimal choice of the next immediate node.

3.3 Numerical Solutions

3.3.1 Solving a system of nonlinear equations – Picard's method of successive approximations

Equations (3.1, 3.2) are a set of nonlinear convolution integral equations. How are these equations to be solved for the unknown probabilities $u_1(t)$, $u_2(t)$, ..., $u_N(t)$? How is the optimal sequence of nodes identified? Picard's method of successive approximation is one possible approach to solving this simultaneous system of nonlinear equations. Picard's method begins with initial approximations to the solution, and then refines these approximations by successive iterations. We begin with the initial, simple approximations

$$u_i^0(t) = \int_0^t p_{iN}(\omega) d\omega, \quad i = 1, 2, \dots, N-1, t \geq 0, \quad (3.3)$$

and

$$u_N^0(t) = 1, t \geq 0. \quad (3.4)$$

These approximations are based on the distributions of travel times over direct links between node i and node N . These first approximations may be poor, because most of the probabilities identified in Equation (3.3) are necessarily zero. Most nodes i have no direct link to node N .

The iterative relationships for successive approximations are

$$u_i^{k+1}(t) = \max_{j \neq i} \int_0^t p_{ij}(\omega) u_j^k(t - \omega) d\omega, \quad i = 1, 2, \dots, N-1, 0 \leq t \leq \infty, \quad (3.5)$$

and

$$u_N^{k+1}(t) = 1, \quad (3.6)$$

where the superscript k is the index of iteration.

The function $u_i^k(t)$ has a useful interpretation. It is the maximum probability of arriving on time if no more than k intermediate nodes are allowed between an origin and the destination. The results can always be improved, or at least remain unchanged, if the k -intermediate-nodes constraint is relaxed to some degree. Improvement may be possible if more intermediate nodes are allowed. Also, because the values $u_i^k(t)$ are probabilities, they are bounded below by 0 and above by 1. Therefore, $0 \leq u_i^k(t) \leq u_i^{k+1}(t) \leq 1$. Thus, this sequence of ever improving approximations is bounded, monotone, and converges to a limiting value. There are N nodes in the network, including the origin and destination nodes. Thus, an optimal path can have no more than $N-2$ intermediate nodes. This means the maximum number of iterations needed to compute exact probabilities is $N-2$.

3.3.2 Evaluating the convolution integrals – Laplace transforms

Equation (3.5) includes a convolution integral that must be evaluated at each successive approximation of $u_i^k(t)$. This requirement increases computational complexity dramatically. There are several methods available for dealing with convolution integrals (Srivastava and Buschman, 1992). Recall the convolution theorem of Laplace Transforms: The transform of a convolution is given by the product of the individual transforms. This theorem can be applied to greatly reduce the complexity of each iteration.

Let us show why this theorem holds. Given a function $f(t)$, the definition of its Laplace transform $F(s)$ is

$$F(s) = L(f(t)) = \int_0^{\infty} f(t)e^{-st} dt. \quad (3.7)$$

Suppose

$$h(t) = \int_0^t f(\omega)g(t-\omega)d\omega. \quad (3.8)$$

The Laplace transform of $h(t)$, denoted $L(h(t))$, is

$$L(h(t)) = \int_0^{\infty} e^{-st}h(t)dt = \int_0^{\infty} e^{-st} \left[\int_0^t f(\omega)g(t-\omega)d\omega \right] dt. \quad (3.9)$$

The right hand side of Equation (3.9) is a double integral over a defined area where $t \in [0, \infty]$, and $\omega \in [0, t]$. See Figure 3-1. The shaded area can also be represented as $\omega \in [0, \infty]$, and $t \in [\omega, \infty]$. Thus, Equation (3.9) can be re-written as

$$L(h(t)) = \int_0^{\infty} f(\omega) \left[\int_{\omega}^{\infty} e^{-st}g(t-\omega)dt \right] d\omega. \quad (3.10)$$

Let $x = t - \omega$. Since $t \in [\omega, \infty]$, $x \in [0, \infty]$. Thus, Equation (3.10) then can be further rewritten as

$$\begin{aligned} L(h(t)) &= \int_0^{\infty} f(\omega) \left[\int_0^{\infty} e^{-s(x+\omega)}g(x)dx \right] d\omega \\ &= \left[\int_0^{\infty} f(\omega)e^{-s\omega}d\omega \right] \left[\int_0^{\infty} e^{-sx}g(x)dx \right]. \end{aligned} \quad (3.11)$$

This shows that the Laplace transform of a convolution is given by the product of the individual Laplace transforms as asserted above. Thus, we have

$$L(h(t)) = L(f(t))L(g(t)). \quad (3.12)$$

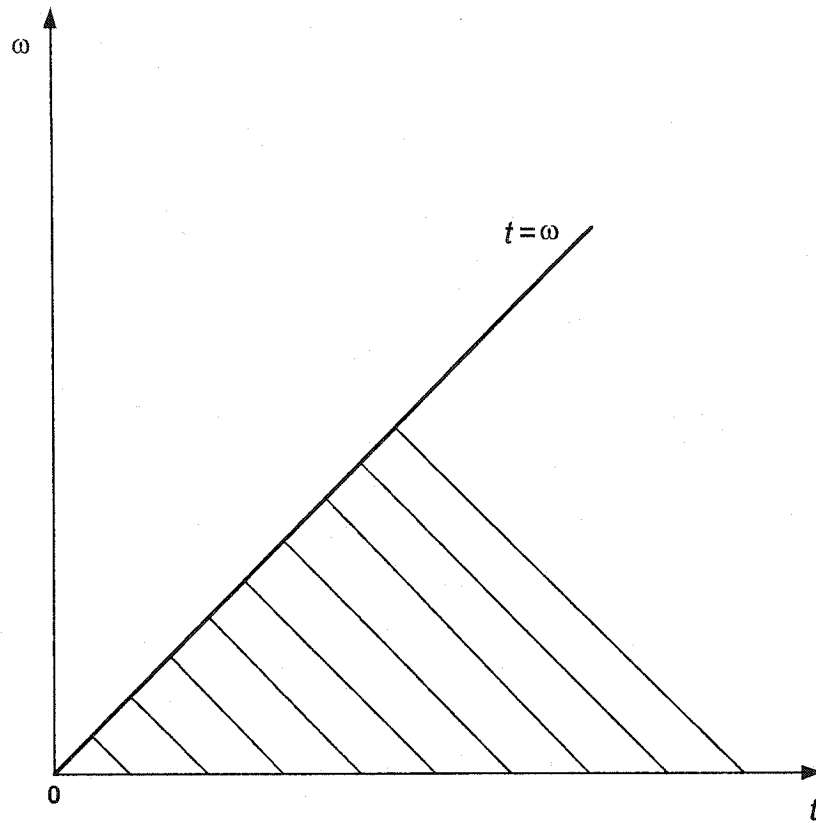


Figure 3-1: Values of t and ω for the Double Integral $L(h(t))$

If the Laplace transforms of the function $p_{ij}(t)$ and the function $u_i^k(t)$ are known, then the Laplace transform of the function $u_i^{k+1}(t)$ in Equation (3.5) can be obtained simply as the product of the two individual transforms. This leaves us with the following two questions. How do we obtain the Laplace transform of a given function? Given a function in the transform domain, how do we invert it into the time domain?

3.3.3 Numerical evaluation of a finite integral – Gaussian quadrature

To obtain a numerical evaluation of the Laplace transform of a given function $f(t)$, let us begin with the definition of the Laplace transform,

$$F(s) = L(f(t)) = \int_0^{\infty} f(t)e^{-st} dt. \quad (3.13)$$

The infinite interval in Equation (3.13) can be reduced to a finite interval between 0 and 1 by substituting τ for e^{-t} . Proceeding, we have

$$t = -\ln \tau, \quad (3.14)$$

and

$$dt = -\frac{d\tau}{\tau}. \quad (3.15)$$

Substituting, Equation (3.13) becomes

$$F(s) = \int_0^1 \tau^{s-1} f(-\ln \tau) d\tau. \quad (3.16)$$

The integral over this finite interval now can be approximated by a finite sum. This leads to the relationship

$$F(s) \cong \sum_{i=1}^n \tau_i^{s-1} f(-\ln \tau_i) w_i, \quad (3.17)$$

where $\tau_1, \tau_2, \dots, \tau_n$ are the points at which $f(-\ln \tau)$ is to be evaluated, and w_1, w_2, \dots, w_n are weights to be attached to these values.

All of the standard quadrature formulas have the form of Equation (3.17), including the rectangular rule, the trapezoid rule, and Simpson's rule. Gaussian quadrature is more sophisticated than these equal interval rules. It picks τ_i and w_i in

such a way that the value of the finite sum in equation (3.17) achieves the true value of the integral if the integrand is a polynomial of degree up to $2n-1$ or less (Bellman and Kalaba, 1966). Gaussian quadrature is employed here to evaluate integrals over a finite interval. This reduces computational costs and guarantees the quality of the approximation. The unknown probabilities in this research are expected to be smooth, monotone functions of t . As the time available to complete the journey decreases, the probability of arriving on time also decreases. Therefore, a polynomial of a low degree should be sufficient in estimating these unknown functions. This limits the number of quadrature points needed for the finite sum in Equation (3.17) to estimate the integral in Equation (3.16).

3.3.4 Numerical inversion of Laplace transforms – linear algebra and generalized inverses

Now let us determine how to obtain a function in the time domain, given the function's Laplace transform. Suppose the Laplace transform values $F(s)$ are obtained for a number of discrete values of s . The relationship between a single transform at a value s and the original function in the time t domain is given in Equation (3.17). The relationships between a set of values $F(s)$ and the values of the original function in the time t domain can be represented by a system of linear algebraic equations. These are

$$F(s) \cong \sum_{i=1}^n \tau_i^{s-1} x_i, \quad s = s_1, s_2, \dots, s_L, \quad (3.18)$$

where

$$x_i = f(-\ln \tau_i) w_i, \quad (3.19)$$

over L observations of s .

These equations can be presented in the matrix format

$$F = T \cdot X, \quad (3.20)$$

where

$$F = \begin{bmatrix} F(1) \\ F(2) \\ \vdots \\ F(L) \end{bmatrix}_{L \times 1}.$$

The quantity L is the total number of points to be evaluated for the function $F(s)$.

The matrix T is of the form

$$T = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \tau_1 & \tau_2 & \cdots & \tau_n \\ \vdots & \vdots & \vdots & \vdots \\ \tau_1^{L-1} & \tau_2^{L-1} & \cdots & \tau_n^{L-1} \end{bmatrix}_{L \times n}.$$

The number of quadrature points is n , and the number of observations of $F(s)$ is L .

The matrix T of the system is the well-known, ill-conditioned Vandermonde matrix.

The vector X is of the form

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n \times 1}.$$

The desired values $f(-\ln \tau_i)$ can be obtained by

$$f(-\ln \tau_i) = \frac{x_i}{w_i}, \quad i = 1, 2, \dots, n, \quad (3.21)$$

once the values of x_1, x_2, \dots, x_n have been determined.

If we have same number of independent conditions as we do unknown variables in the system of algebraic Equations (3.20), and if the matrix T is nonsingular, then the unknown vector X can be obtained by the formula

$$X = T^{-1} \cdot F, \quad (3.22)$$

where T^{-1} is the inverse of matrix T . However, it will not generally be true that the number of observations $F(s)$ and the number of quadrature points are equal. This is not trivial. Since the unknown probabilities, $u_1(t), u_2(t), \dots, u_N(t)$ are expected to be smooth monotone functions of t , they will be well represented by polynomials of a low degree. This means only a small number of quadrature points is needed. Obtaining good-quality results from the numerical inversion of the Laplace transform requires evaluating the Laplace Transform at more discrete values of s . Thus, finding X generally requires solution of a system of algebraic equations with more conditions than unknowns. Fortunately, non-singularity does not have to hold for this system of algebraic equations to have a solution in the least square sense. If the matrix T is singular, then the unknown vector X can be obtained by

$$X = T^+ \cdot F, \quad (3.23)$$

where T^+ is the Moore-Penrose inverse of matrix T . Methods to obtain the Moore-Penrose inverse are widely available in the applied mathematics literature. Fan and Kalaba (2001) give an efficient dynamic programming approach to computing the

Moore-Penrose inverse, including a procedure for treating matrices with nearly linearly dependent columns.

Based on our numerical experiments, inverse of Laplace transforms may be instable sometime. The mathematical reason for this problem is that the inverse Laplace transform is an unbounded operator. An arbitrary small change in $F(s)$ can produce arbitrary large changes in the value of $u(t)$. Consider a function $\sin(at)$. Its Laplace transform is $a / (a^2 + s^2)$. This term becomes very tiny when a reaches a very large number. If we add this small term to the Laplace transform of any function $f(t)$, the change may not even be noticeable. However, the original function now becomes $\{f(t) + \sin(at)\}$. The resultant change in the original function is significant, because $\sin(at)$ will cause high frequency oscillation when the value of a is large.

This instability can also be demonstrated in the behavior of the ill-conditioned matrix T . The ill-conditioning of T rapidly worsens as L increases. When T^+ contains elements of both signs of large magnitude, it becomes inevitable that a small change in the vector F produces a large change in X . Some applied mathematics techniques are available for improving such situation (Bellman and Kalaba, 1966).

In our research, the instability problem might occur as the number of quadrature points increases. However, we know how these probability functions should behave. They do not have high frequencies. As a matter of fact, these functions are smooth, monotone and bounded. Therefore, we will be able to detect

the instability problem if oscillation occurs in the estimated unknown functions $u_1(t), \dots, u_N(t)$. Examining the behavior of the estimated unknown functions can also be a means of checking the accuracy of our procedure of numerical inversion of Laplace transform.

3.4 Numerical Examples

Accuracy, reliability, and potential applicability are often the major concerns in situations where new formulations or algorithms are proposed. The numerical examples included below are designed to meet the following two principle purposes. The first is to validate the accuracy and reliability of our numerical schemes. Some of the numerical examples are small but not trivial, because accuracy is the most basic concern, especially when there is no other study to be relied on or compared with. With those simple examples, we have tested our numerical approaches in various contexts. The second purpose is to study the applicability of our approach to realistic network problems. The sizes of the numerical examples served for this purpose are not large enough for realistic network problems yet. However, these examples have shown that expansion of our approaches to more realistic problems is feasible.

3.4.1 Test 1: Numerical inversion of Laplace transforms

Examination of a few monotone, smooth functions validates the accuracy of our algorithms for obtaining a Laplace transform, and the transform's numerical inversion via solution of a system of linear equations. The functions selected are

$$f_1(t) = 1, \quad (3.24)$$

$$f_2(t) = e^{-t}, \quad (3.25)$$

$$f_3(t) = t^{0.5}, \quad (3.26)$$

and

$$f_4(t) = 1 - e^{-t}. \quad (3.27)$$

We proceed with Laplace transforms that can be analytically obtained so that the numerical results of our procedure can be compared with the corresponding exact solutions. For the sake of simplicity, the matrix T is defined to be square. This permits the inverse of T to be computed as a standard matrix inverse, rather than requiring the calculation of T^+ . In this case, the number of quadrature points and the number of observations on the Laplace transform, $F(s)$, are both taken to be eight. Calculations are executed with double precision. Agreement between numerical and analytical values is excellent, and we conclude that estimating the inverse of Laplace transforms via solution of a system of linear equations may be an effective procedure. Inputs and results are listed in Table 3-1. Estimated and analytical results are too similar to display graphically.

3.4.2 Test 2: Application to a small network

Testing our procedure on a small network validates the accuracy of this scheme. Consider the five-node network in Figure 3-2. Since link travel times are non-negative random variables, we assume they follow gamma distributions, a commonly made assumption in the theory of random duration.

**Table 3-1: True versus Estimated Values of Laplace Transforms and
Their Inverses**

	s	1	2	3	4	5	6	7	8
Function	Quadrature Points τ	0.0199	0.1017	0.2372	0.4083	0.5917	0.7628	0.8983	0.9801
	Gaussian Quadrature Weights	0.0506	0.1112	0.1569	0.1813	0.1813	0.1569	1.1119	0.0506
	$t = -\ln(\tau)$	3.9193	2.2861	1.4387	0.8958	0.5247	0.2708	0.1072	0.0201
$f_1(t) = 1$	True $F_1(s)$	1.0000	0.5000	0.3333	0.2500	0.2000	0.1667	0.1429	0.1250
	Estimated $F_1(s)$	1.0000	0.5000	0.3333	0.2500	0.2000	0.1667	0.1429	0.1250
	True $f_1(t)$	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Estimated $f_1(t)$	0.9987	0.9981	1.0020	0.9996	0.9979	0.9988	0.9997	0.9990
$f_2(t) = \exp(-t)$	True $F_2(s)$	0.5000	0.3333	0.2500	0.2000	0.1667	0.1429	0.1250	0.1111
	Estimated $F_2(s)$	0.5000	0.3333	0.2500	0.2000	0.1667	0.1429	0.1250	0.1111
	True $f_2(t)$	0.0199	0.1017	0.2372	0.4083	0.5917	0.7630	0.8983	0.9801
	Estimated $f_2(t)$	0.0188	0.1001	0.2389	0.4079	0.5900	0.7619	0.8981	0.9793
$f_3(t) = \sqrt{t}$	True $F_3(s)$	0.8862	0.3133	0.1706	0.1108	0.0793	0.0603	0.04785	0.03917
	Estimated $F_3(s)$	0.8847	0.3135	0.1707	0.1110	0.0794	0.0605	0.0480	0.0393
	True $f_3(t)$	1.9800	1.5120	1.1990	0.9465	0.7244	0.5204	0.3274	0.1418
	Estimated $f_3(t)$	1.9790	1.5110	1.2000	0.9463	0.7236	0.5198	0.3273	0.1413
$f_4(t) = 1 - \exp(-t)$	True $F_4(s)$	0.5000	0.1667	0.0833	0.0500	0.0333	0.0238	0.0179	0.0139
	Estimated $F_4(s)$	0.5000	0.1667	0.0833	0.0500	0.0333	0.0238	0.0179	0.0139
	True $f_4(t)$	0.9801	0.8983	0.7628	0.5917	0.4083	0.2372	0.1017	0.0199
	Estimated $f_4(t)$	0.9799	0.8980	0.7632	0.5916	0.4079	0.2370	0.1016	0.0197

A gamma distribution is smooth, which aids in computational treatments.

Gamma distributions have the form of

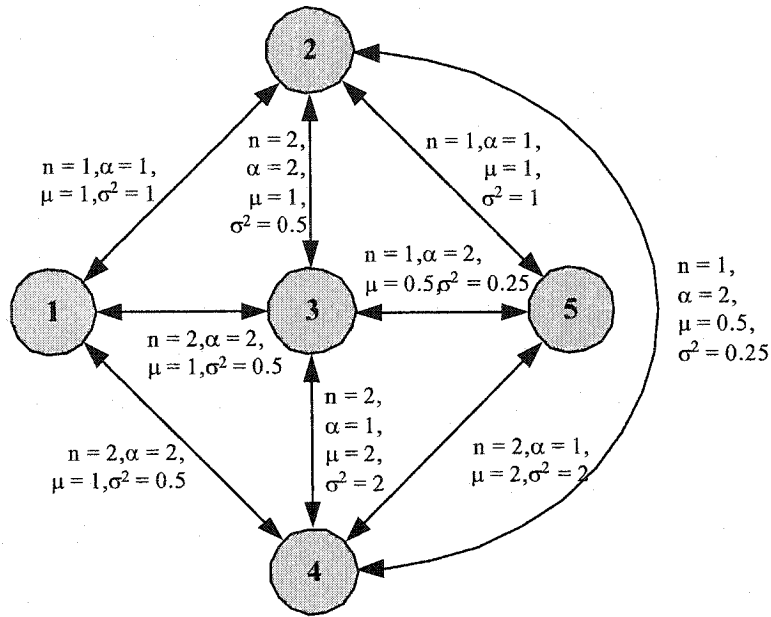


Figure 3-2: A Small Network With Probabilistic Link Travel Costs

$$p(t, n, \alpha) = \frac{\alpha^n e^{-\alpha t} t^{n-1}}{\Gamma(n)}, \quad (3.28)$$

in which $\Gamma(n)$ is the gamma function of n . The mean μ and variance σ^2 are

$$\mu = n/\alpha, \text{ and} \quad (3.29)$$

$$\sigma^2 = n/\alpha^2. \quad (3.30)$$

See Equations (A-8 and A-9) in Appendix A for detailed derivation of the mean and variance for gamma distribution. As before, the matrix T is defined square. In this case, the number of quadrature points and the number of observations on the Laplace transform, $F(s)$, are both taken to be eight.

Applying this procedure yields the probabilities defined in Equations (3.1, 3.2). These probabilities of arriving at the destination node $N = 5$ in time t or less given an optimal direction of departure from each potential origin are shown in Table 3-2. A MATLAB code for obtaining the maximum probability of arriving on time and the optimal successor node from each origin to the destination node 5 is given in Appendix B. Each value of t is associated by Equation (3.14) with one of the selected quadrature points. As expected, the probability of arriving at destination node N on time decreases monotonically for all origins as the time available for completing the trip is reduced.

The procedure identifies the successor node that should be visited from each potential origin. The entries in Table 3-2 demonstrate that the optimal strategies change as the amount of time available to complete the trip to node N decreases. In the case of trips originating from node 1, the optimal successor node changes from node 3 to node 2 as the time available for an on-time arrival diminishes. Even though the minimum expected travel time of link (2, 5) is bigger than that of link (3, 5), there is higher chance of on-time arrival by taking path (1, 2, 5) due to the higher variance of travel time over that path. In the case of trips originating from node 2, the SOTA strategy suggests node 3 as the best successor node when the time available is big, even though taking the direct link (2, 5) will result in a smaller average travel cost. This is again a result of considering both reliability and cost of

Table 3-2: Probabilities of Arriving on Time at Destination Node N = 5**Within Time t Given an Optimal Choice of Successor Nodes**

Origin		$t_1=$ 3.9193	$t_2=$ 2.2861	$t_3=$ 1.4387	$t_4=$ 0.8958	$t_5=$ 0.5247	$t_6=$ 0.2708	$t_7=$ 0.1072	$t_8=$ 0.0201
1	$u_1(t)$	0.9828	0.8351	0.5483	0.2672	0.0980	0.0303	0.0056	0.0001
	Successor Node	3	3	3	3	2	2	2	3
2	$u_2(t)$	0.9828	0.8980	0.7632	0.5916	0.4079	0.2370	0.1016	0.0197
	Successor Node	3	5	5	5	5	5	5	5
3	$u_3(t)$	0.9993	0.9891	0.9444	0.8332	0.6493	0.4178	0.1929	0.0391
	Successor Node	5	5	5	5	5	5	5	5
4	$u_4(t)$	0.9626	0.8066	0.5822	0.3498	0.1665	0.0561	0.0103	0.0003
	Successor Node	2	2	2	2	2	2	2	2
5	$u_5(t)$	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

the possible alternatives. Computational experience with double-precision calculations suggests that probabilities calculated in this fashion are accurate to only the second decimal. Thus, some of the results associated with $t_8 = 0.0201$ are suspect. As the time available to execute a complete trip becomes small, the probability of success also becomes small, regardless of the origin of the trip. Relative computation errors are largest under these circumstances. A computer output of the detail results containing the intermediate successive approximations is given in Appendix C to show how the results evolve.

The selection of quadrature points and associated weights is important because the quadrature points define the values of time for which probabilities will be evaluated. Additional probabilities can be computed if more quadrature points are

defined. However, the range of the time points does not increase much, while computational cost increases quickly with the number of quadrature points. Section 3.5 demonstrates a computationally efficient extension for evaluation of probabilities for more time values and a larger time range.

3.4.3 Test 3: Application to Larger Networks

3.4.3.1 *Application to a 49-node network*

Applying the procedure to a larger network tests the efficiency of the algorithm and the sensitivity of the procedure to cumulative computational error. Consider the 49-node network displayed in Figure 3-3. This is a fairly representative spatial network in which nodes are connected to all of their nearest neighbors. Link travel times are given in terms of gamma distributions. See Equation (3.28). The number and value of the Gaussian quadrature points and weights are unchanged relative to the previous examples. The parameters of the gamma distribution for each link are defined using the following rules,

$$n = 0.4 \log_{10}(10 + 0.8i + 0.7j), \text{ and} \quad (3.31)$$

$$\alpha = 2 \log_{10}(12 + 1.2i + 0.8j), \quad (3.32)$$

where i and j are the starting and ending nodes for link ij . This produces values of n on the approximate interval 0.4 to 0.75, and values of α on the approximate interval 2 to 4. Comparing to the 49-node network with another set of gamma distributed link travel times that will be shown later, this network has more reliable link travel times.

The definition of these two parameters is a matter of programming convenience.

Other parameter ranges are admissible.

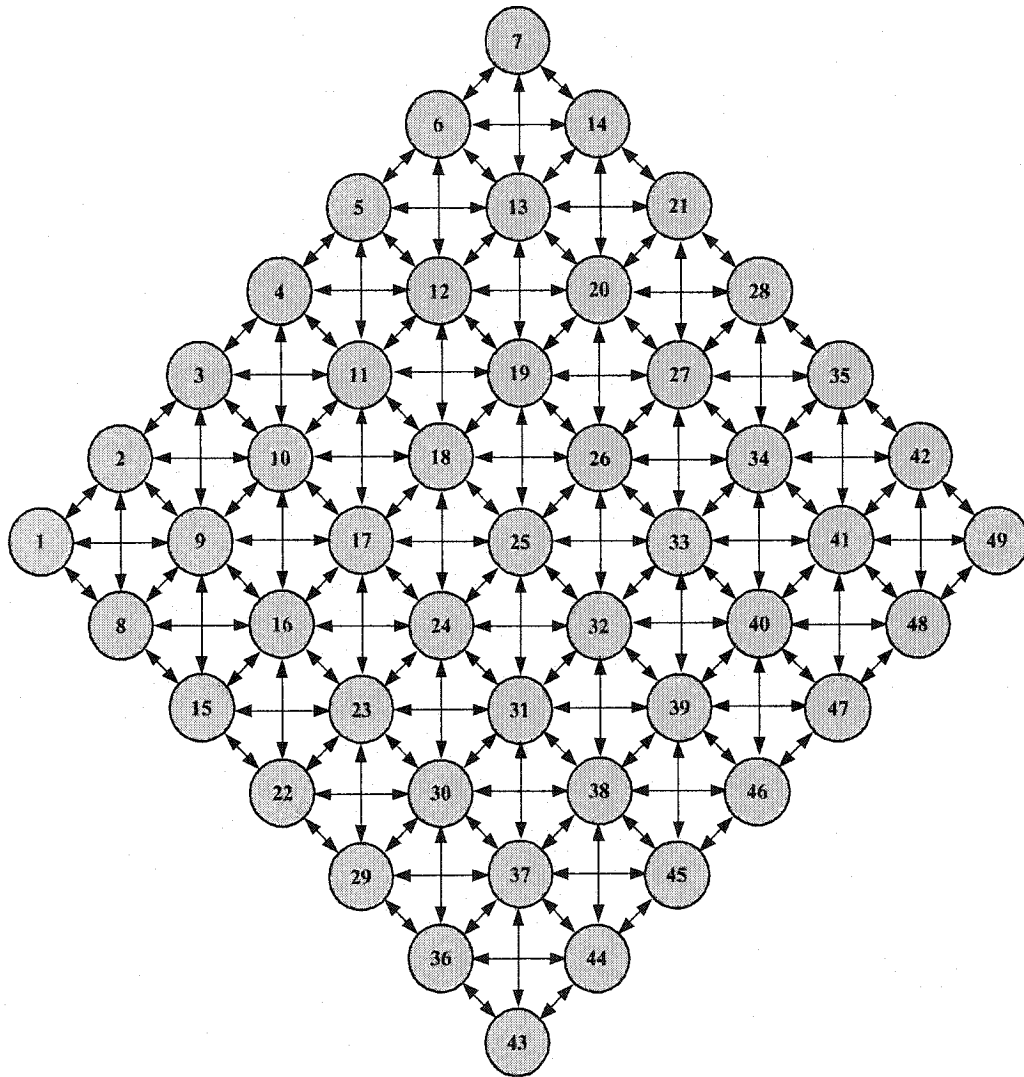


Figure 3-3: A 49-node Network

The procedure converges quickly. The intermediate results of successive approximation are available in Appendix D. Our experience indicates that the results remain unchanged after 5 times of successive iterations. The probabilities of arriving at the destination node $N = 49$ in time t or less given an optimal direction of departure from each potential origin are shown in Table 3-3. The next node that should be visited from each node i to maximize the probability of arriving at the destination within time t is also shown. Results for this larger network parallel the previous example. As the time available to execute a complete trip becomes small, the probability of success also becomes small regardless of the origin of the trip. The further the origin is from the destination node N , the smaller the probability of arriving on time for any given value of t . As before, for a given origin, the optimal choice of successor node changes as the amount of time available to complete the trip to node N decreases. See node 15 as an example.

It is useful to compare the results from this maximum probability problem with the results from the deterministic shortest path problem. The formulation of deterministic shortest path problems using dynamic programming is given as Equation (2.3a and 2.3b) in Chapter 2. Recall that the procedure for stochastic shortest path problems can be reduced to the standard shortest path procedure for deterministic networks by replacing the probability density function by the expectations. A 49-node network with the same configuration as the network in Figure 3-3 is used in the deterministic shortest path problem. The link travel times

are set to be the expectations of the link travel times in the SOTA problem. The equation for calculating the expectation of gamma distributed random variables is

$$mean = \frac{\Gamma(n+1)}{\Gamma(n)\alpha}. \quad (3.33)$$

Note that the values of the parameter n are not integers in this example. Equations (3.29 and 3.30) are no longer applicable. See Appendix A for the detail derivation of the mean and variance of gamma distributed random variables.

The minimum cost and the optimal successor node from each node i are shown in the last column of Table 3-3. In this particular network, the successor nodes suggested by the average shortest path algorithm and maximum probability algorithm generally agree with one another when the allowable travel time is limited. For some starting nodes, the maximum probability algorithm suggests different strategies when the time available for completing the journey is larger. Recall that link travel times in this example are quite reliable comparing to the other 49-node network shown later. It is more likely to find an agreement between stochastic model and deterministic model when variances of random variables are small. The results from the second 49-node network will show more impact of variance on the optimal decisions. However, it is difficult to quantitatively describe the pattern of the agreement between the two algorithms.

Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ within Time t or Less Given an Optimal Choice of Successor Nodes

		$t_1=$	$t_2=$	$t_3=$	$t_4=$	$t_5=$	$t_6=$	$t_7=$	$t_8=$	Average shortest path cost & next nodes
origin		3.9193	2.2861	1.4387	0.8958	0.5247	0.2708	0.1072	0.0201	
1	$u_1(t)$	0.7562	0.7205	0.5449	0.269	0.0732	0	0	0	1.1284
	Successor node	9	9	9	9	9	undefined			9
2	$u_2(t)$	0.7591	0.7236	0.5477	0.2703	0.0735	0	0	0	1.1279
	Successor node	9	9	9	9	9	undefined			9
3	$u_3(t)$	0.7618	0.7266	0.5502	0.2716	0.0738	0	0	0	1.1274
	Successor node	9	9	9	9	9	undefined			9
4	$u_4(t)$	0.7678	0.7328	0.5555	0.274	0.0742	0	0	0	1.1272
	Successor node	10	10	10	10	10	undefined			10
5	$u_5(t)$	0.7731	0.7384	0.5602	0.2761	0.0746	0	0	0	1.1269
	Successor node	11	11	11	11	11	undefined			11
6	$u_6(t)$	0.7796	0.7451	0.5658	0.2787	0.075	0	0	0	1.1268
	Successor node	12	12	12	12	12	undefined			12
7	$u_7(t)$	0.7855	0.7513	0.571	0.281	0.0754	0	0	0	1.1268
	Successor node	13	13	13	13	13	undefined			13
8	$u_8(t)$	0.8006	0.7674	0.5825	0.2766	0.0748	0	0	0	1.126
	Successor node	16	16	16	9	9	undefined			9
9	$u_9(t)$	0.8383	0.8222	0.6996	0.4272	0.1583	0.0254	0	0	0.9401
	Successor node	17	17	17	17	17	17	undefined		17
10	$u_{10}(t)$	0.8396	0.8235	0.7009	0.4281	0.1585	0.0254	0	0	0.9399
	Successor node	17	17	17	17	17	17	undefined		17
11	$u_{11}(t)$	0.8408	0.8249	0.7023	0.4289	0.1588	0.0254	0	0	0.9398
	Successor node	17	17	17	17	17	17	undefined		17

Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ **Within Time t or Less Given an Optimal Choice of Successor Nodes (Continue)**

12	$u_{12}(t)$	0.8438	0.828	0.7053	0.4307	0.1592	0.0254	0	0	0.9397
	Successor node	18	18	18	18	18	18	undefined		18
13	$u_{13}(t)$	0.8466	0.8309	0.7081	0.4324	0.1596	0.0253	0	0	0.9397
	Successor node	19	19	19	19	19	19	undefined		19
14	$u_{14}(t)$	0.8503	0.8348	0.7118	0.4345	0.16	0.0253	0	0	0.9397
	Successor node	20	20	20	20	20	20	undefined		20
15	$u_{15}(t)$	0.8372	0.8064	0.6102	0.2813	0.0757	0	0	0	1.125
	Successor node	23	23	23	9	9		undefined		9
16	$u_{16}(t)$	0.8567	0.8415	0.7183	0.4326	0.1597	0.0254	0	0	0.9391
	Successor node	24	24	24	17	17	17	undefined		17
17	$u_{17}(t)$	0.894	0.8884	0.8201	0.6032	0.2999	0.0804	0	0	0.7526
	Successor node	25	25	25	25	25	25	undefined		25
18	$u_{18}(t)$	0.8947	0.8892	0.821	0.6039	0.3002	0.0804	0	0	0.7525
	Successor node	25	25	25	25	25	25	undefined		25
19	$u_{19}(t)$	0.8955	0.89	0.8219	0.6046	0.3005	0.0804	0	0	0.7524
	Successor node	25	25	25	25	25	25	undefined		25
20	$u_{20}(t)$	0.8973	0.8919	0.8239	0.6062	0.301	0.0804	0	0	0.7524
	Successor node	26	26	26	26	26	26	undefined		26
21	$u_{21}(t)$	0.8991	0.8937	0.8258	0.6077	0.3015	0.0803	0	0	0.7523
	Successor node	27	27	27	27	27	27	undefined		27
22	$u_{22}(t)$	0.8631	0.8324	0.6387	0.3072	0.0771	0	0	0	1.125
	Successor node	30	30	30	30	16		undefined		16
23	$u_{23}(t)$	0.8802	0.867	0.7398	0.4365	0.1607	0.0253	0	0	0.9386
	Successor node	31	31	31	17	17	17	undefined		17

Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ **Within Time t or Less Given an Optimal Choice of Successor Nodes (Continue)**

24	$u_{24}(t)$	0.8988	0.8935	0.8257	0.6078	0.3017	0.0804	0	0	0.752
	Successor node	25	25	25	25	25	25	undefined		25
25	$u_{25}(t)$	0.9365	0.9351	0.9069	0.7712	0.5007	0.2046	0.0365	0	0.5649
	Successor node	33	33	33	33	33	33	33	undefined	33
26	$u_{26}(t)$	0.937	0.9356	0.9075	0.7718	0.501	0.2046	0.0365	0	0.5648
	Successor node	33	33	33	33	33	33	33	undefined	33
27	$u_{27}(t)$	0.9375	0.9361	0.908	0.7724	0.5014	0.2047	0.0365	0	0.5647
	Successor node	33	33	33	33	33	33	33	undefined	33
28	$u_{28}(t)$	0.9388	0.9374	0.9095	0.7737	0.5021	0.2047	0.0364	0	0.5647
	Successor node	34	34	34	34	34	34	34	undefined	34
29	$u_{29}(t)$	0.8721	0.8409	0.6438	0.3099	0.0781	0	0	0	1.1252
	Successor node	37	37	37	37	23		undefined		23
30	$u_{30}(t)$	0.8975	0.8842	0.7599	0.4605	0.1622	0.0251	0	0	0.9387
	Successor node	38	38	38	38	24	24	undefined		24
31	$u_{31}(t)$	0.9145	0.9098	0.8429	0.6114	0.303	0.0804	0	0	0.7516
	Successor node	39	39	39	25	25	25	undefined		25
32	$u_{32}(t)$	0.9398	0.9385	0.9107	0.7751	0.503	0.2049	0.0363	0	0.5645
	Successor node	33	33	33	33	33	33	33	undefined	33
33	$u_{33}(t)$	0.9707	0.9704	0.9636	0.903	0.7292	0.4391	0.1516	0.0124	0.3768
	Successor node	41	41	41	41	41	41	41	41	41
34	$u_{34}(t)$	0.9711	0.9708	0.964	0.9035	0.7296	0.4392	0.1515	0.0124	0.3768
	Successor node	41	41	41	41	41	41	41	41	41
35	$u_{35}(t)$	0.9714	0.9712	0.9644	0.904	0.7301	0.4393	0.1515	0.0123	0.3767
	Successor node	41	41	41	41	41	41	41	41	41

Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ **Within Time t or Less Given an Optimal Choice of Successor Nodes (Continue)**

36	$u_{36}(t)$	0.8697	0.8395	0.6451	0.31	0.0773	0	0	0	1.1259
	Successor node	30	30	30	30	30	undefined			30
37	$u_{37}(t)$	0.9001	0.8869	0.7583	0.4608	0.1636	0.0246	0	0	0.9388
	Successor node	38	38	45	45	31	31	undefined		31
38	$u_{38}(t)$	0.9256	0.9213	0.8553	0.63	0.3054	0.0801	0	0	0.7517
	Successor node	46	46	46	46	32	32	undefined		32
39	$u_{39}(t)$	0.9426	0.9414	0.914	0.7784	0.5049	0.205	0.0361	0	0.5642
	Successor node	33	33	33	33	33	33	33	undefined	33
40	$u_{40}(t)$	0.9731	0.9729	0.9663	0.9062	0.732	0.44	0.1511	0.0122	0.3765
	Successor node	41	41	41	41	41	41	41	41	41
41	$u_{41}(t)$	0.9995	0.9986	1	0.9834	0.9228	0.76	0.4832	0.1512	0.1885
	Successor node	49	49	49	49	49	49	49	49	49
42	$u_{42}(t)$	0.9995	0.9986	1	0.9835	0.9229	0.76	0.4828	0.1508	0.1885
	Successor node	49	49	49	49	49	49	49	49	49
43	$u_{43}(t)$	0.877	0.8461	0.6485	0.3119	0.0779	0	0	0	1.1264
	Successor node	37	37	37	37	37	undefined			37
44	$u_{44}(t)$	0.9023	0.8893	0.7652	0.4637	0.1636	0.0247	0	0	0.9393
	Successor node	38	38	38	38	38	38	undefined		38
45	$u_{45}(t)$	0.9192	0.9146	0.8484	0.6252	0.3071	0.0796	0	0	0.7518
	Successor node	39	39	39	39	39	39	undefined		39
46	$u_{46}(t)$	0.9494	0.9483	0.9217	0.7858	0.5087	0.2047	0.0354	0	0.5643
	Successor node	40	40	40	40	40	40	40	undefined	40
47	$u_{47}(t)$	0.9753	0.9751	0.9687	0.909	0.7345	0.4407	0.1505	0.0121	0.3763
	Successor node	41	41	41	41	41	41	41	41	41

Table 3-3: Probabilities of Arriving at the Destination Node $N = 49$ **Within Time t or Less Given an Optimal Choice of Successor Nodes (Continue)**

	$u_{48}(t)$	0.9995	0.9986	1.0001	0.984	0.9238	0.7603	0.4808	0.1482	0.1883
48	Successor node	49	49	49	49	49	49	49	49	49
49	$u_{49}(t)$	1	1	1	1	1	1	1	1	0

3.4.3.2 Application to another 49-node network with a second set of link travel time distributions

It is also useful to compare the results of the maximum probability problems with the same network configuration and average link travel times but different variance of link travel times. We set the parameters α and n in link travel time distribution as half of those in Equations (3.31 and 3.32). This change of parameters results in the same average travel times but doubled variances. Refer to Appendix A for the calculation of the mean and variance of the gamma distributed variables. The results of the maximum probability of arriving on time and the optimal successor nodes using the second set of gamma distribution parameters are reported in Table 3-4. Maximum probabilities of arriving on time in this example are overall smaller comparing to the data in Table 3-3. This is as expected because larger variances indicate greater uncertainty. We also have observed that for some starting nodes, the SOTA strategy and the shortest path strategy never agree with one another despite of the amount of the remaining time for the on-time arrival. See the cases of starting from nodes 23, 24, 31, 38, and 39 as an example.

Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$ within Time t or Less Given an Optimal Choice of Successor Nodes Using the Second Set of Gamma Distributions

		$t_1=$	$t_2=$	$t_3=$	$t_4=$	$t_5=$	$t_6=$	$t_7=$	$t_8=$	Average shortest path cost & next nodes
origin		3.9193	2.2861	1.4387	0.8958	0.5247	0.2708	0.1072	0.0201	
1	$u_1(t)$	0.2771	0.2417	0.1708	0.0923	0.0346	0.007	0.0003	0	1.1284
	Successor node	9	9	9	9	9	9	9	unidentified	9
2	$u_2(t)$	0.2848	0.2489	0.1762	0.0952	0.0356	0.0072	0.0003	0	1.1279
	Successor node	10	10	10	10	10	10	9	unidentified	9
3	$u_3(t)$	0.2918	0.2556	0.1811	0.0978	0.0365	0.0074	0.0003	0	1.1274
	Successor node	11	11	11	11	11	11	9	unidentified	9
4	$u_4(t)$	0.2978	0.2612	0.1853	0.1001	0.0373	0.0074	0.0003	0	1.1272
	Successor node	12	12	12	12	12	11	10	unidentified	10
5	$u_5(t)$	0.3026	0.2658	0.1887	0.1019	0.0379	0.0076	0.0003	0	1.1269
	Successor node	13	13	13	13	13	13	11	unidentified	11
6	$u_6(t)$	0.3059	0.269	0.1911	0.1032	0.0383	0.0076	0.0003	0	1.1268
	Successor node	14	14	14	14	14	12	12	unidentified	12
7	$u_7(t)$	0.3072	0.2702	0.192	0.1037	0.0385	0.0076	0.0003	0	1.1268
	Successor node	14	14	14	14	14	13	13	unidentified	13
8	$u_8(t)$	0.3232	0.2854	0.2033	0.1096	0.0405	0.008	0.0003	0	1.126
	Successor node	16	16	16	16	16	16	9	unidentified	9
9	$u_9(t)$	0.396	0.3664	0.2902	0.184	0.0863	0.0247	0.0024	0	0.9401
	Successor node	17	17	17	17	17	17	17	unidentified	17
10	$u_{10}(t)$	0.4016	0.372	0.2949	0.187	0.0876	0.0249	0.0024	0	0.9399
	Successor node	18	18	18	18	18	18	17	unidentified	17

Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$
Within Time t or Less Given an Optimal Choice of Successor Nodes Using the
Second Set of Gamma Distributions (Continue)

11	$u_{11}(t)$	0.4067	0.377	0.2992	0.1897	0.0888	0.0252	0.0024	0	0.9398
	Successor node	19	19	19	19	19	19	17	unidentified	17
12	$u_{12}(t)$	0.4107	0.381	0.3025	0.1919	0.0897	0.0254	0.0024	0	0.9397
	Successor node	20	20	20	20	20	20	18	unidentified	18
13	$u_{13}(t)$	0.4134	0.3837	0.3048	0.1933	0.0903	0.0255	0.0024	0	0.9397
	Successor node	21	21	21	21	21	20	19	unidentified	19
14	$u_{14}(t)$	0.4144	0.3847	0.3057	0.1939	0.0906	0.0256	0.0024	0	0.9397
	Successor node	21	21	21	21	21	20	20	unidentified	20
15	$u_{15}(t)$	0.3557	0.3164	0.2265	0.1219	0.0446	0.0087	0.0003	0	1.125
	Successor node	23	23	23	23	23	23	9	unidentified	9
16	$u_{16}(t)$	0.4313	0.4015	0.3197	0.2027	0.0942	0.0265	0.0024	0	0.9391
	Successor node	24	24	24	24	24	24	17	unidentified	17
17	$u_{17}(t)$	0.5234	0.503	0.4364	0.3196	0.1849	0.0732	0.013	0	0.7526
	Successor node	25	25	25	25	25	25	25	unidentified	25
18	$u_{18}(t)$	0.5277	0.5073	0.4404	0.3226	0.1865	0.0736	0.0131	0	0.7525
	Successor node	26	26	26	26	26	26	25	unidentified	25
19	$u_{19}(t)$	0.5313	0.511	0.4438	0.3252	0.1879	0.0741	0.0131	0	0.7524
	Successor node	27	27	27	27	27	27	25	unidentified	25
20	$u_{20}(t)$	0.5337	0.5134	0.4461	0.3269	0.1889	0.0744	0.0131	0	0.7524
	Successor node	28	28	28	28	28	28	26	unidentified	26
21	$u_{21}(t)$	0.5346	0.5143	0.447	0.3276	0.1893	0.0746	0.0132	0	0.7523
	Successor node	28	28	28	28	28	28	27	unidentified	27
22	$u_{22}(t)$	0.378	0.3377	0.2426	0.1304	0.0473	0.009	0.0003	0	1.125
	Successor node	30	30	30	30	30	30	16	unidentified	16

Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$
Within Time t or Less Given an Optimal Choice of Successor Nodes Using the
Second Set of Gamma Distributions (Continue)

23	$u_{23}(t)$	0.457	0.427	0.3414	0.2165	0.1001	0.0278	0.0026	0	0.9386
	Successor node	31	31	31	31	31	31	31	unidentified	17
24	$u_{24}(t)$	0.5504	0.5304	0.462	0.3386	0.195	0.0763	0.0134	0	0.752
	Successor node	32	32	32	32	32	32	32	unidentified	25
25	$u_{25}(t)$	0.6644	0.6528	0.6059	0.5015	0.3507	0.1876	0.058	0.0018	0.5649
	Successor node	33	33	33	33	33	33	33	33	33
26	$u_{26}(t)$	0.6673	0.6558	0.6089	0.5041	0.3525	0.1883	0.0581	0.0018	0.5648
	Successor node	34	34	34	34	34	34	33	33	33
27	$u_{27}(t)$	0.6696	0.6581	0.6112	0.5062	0.3538	0.1889	0.0583	0.0019	0.5647
	Successor node	35	35	35	35	35	35	35	33	33
28	$u_{28}(t)$	0.6704	0.659	0.6121	0.5069	0.3544	0.1892	0.0583	0.0019	0.5647
	Successor node	35	35	35	35	35	35	34	34	34
29	$u_{29}(t)$	0.3923	0.3515	0.2531	0.1359	0.0491	0.0093	0.0004	0	1.1252
	Successor node	37	37	37	37	37	37	23	unidentified	23
30	$u_{30}(t)$	0.4736	0.4436	0.3556	0.2255	0.1039	0.0286	0.0025	0	0.9387
	Successor node	38	38	38	38	38	38	24	unidentified	24
31	$u_{31}(t)$	0.5696	0.55	0.4803	0.3524	0.2025	0.0787	0.0136	0	0.7516
	Successor node	39	39	39	39	39	39	39	unidentified	25
32	$u_{32}(t)$	0.6832	0.6721	0.6253	0.5183	0.3618	0.1922	0.0589	0.0019	0.5645
	Successor node	40	40	40	40	40	40	40	33	33
33	$u_{33}(t)$	0.8222	0.8167	0.7928	0.7209	0.5938	0.4132	0.2106	0.0371	0.3768
	Successor node	41	41	41	41	41	41	41	41	41
34	$u_{34}(t)$	0.8237	0.8183	0.7945	0.7226	0.5951	0.4139	0.2108	0.0371	0.3768
	Successor node	42	42	42	42	42	42	41	41	41

Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$
Within Time t or Less Given an Optimal Choice of Successor Nodes Using the
Second Set of Gamma Distributions (Continue)

35	$u_{35}(t)$	0.8245	0.8191	0.7954	0.7234	0.5958	0.4144	0.211	0.0371	0.3767
	Successor node	42	42	42	42	42	42	41	41	41
36	$u_{36}(t)$	0.4001	0.3591	0.2589	0.1389	0.0501	0.0095	0.0004	0	1.1259
	Successor node	44	44	44	44	44	44	30	unidentified	30
37	$u_{37}(t)$	0.4827	0.4528	0.3634	0.2305	0.106	0.029	0.0026	0	0.9388
	Successor node	45	45	45	45	45	45	31	unidentified	31
38	$u_{38}(t)$	0.5801	0.5608	0.4905	0.3601	0.2067	0.08	0.0137	0	0.7517
	Successor node	46	46	46	46	46	46	46	unidentified	32
39	$u_{39}(t)$	0.6954	0.6847	0.638	0.5293	0.3693	0.1956	0.0595	0.0019	0.5642
	Successor node	47	47	47	47	47	47	47	33	33
40	$u_{40}(t)$	0.8321	0.827	0.8037	0.7316	0.6022	0.4177	0.2117	0.0372	0.3765
	Successor node	48	48	48	48	48	48	41	41	41
41	$u_{41}(t)$	0.9998	0.9963	0.9921	0.9588	0.8975	0.7741	0.6016	0.3129	0.1885
	Successor node	49	49	49	49	49	49	49	49	49
42	$u_{42}(t)$	0.9998	0.9964	0.9922	0.9589	0.8976	0.774	0.6012	0.3123	0.1885
	Successor node	49	49	49	49	49	49	49	49	49
43	$u_{43}(t)$	0.4026	0.3615	0.2607	0.1399	0.0504	0.0095	0.0004	0	1.1264
	Successor node	44	44	44	44	44	44	37	unidentified	37
44	$u_{44}(t)$	0.4855	0.4556	0.366	0.2321	0.1067	0.0292	0.0026	0	0.9393
	Successor node	45	45	45	45	45	45	38	unidentified	38
45	$u_{45}(t)$	0.5834	0.5642	0.4937	0.3626	0.2081	0.0804	0.0138	0	0.7518
	Successor node	46	46	46	46	46	46	39	unidentified	39
46	$u_{46}(t)$	0.6992	0.6887	0.642	0.5329	0.3718	0.1967	0.0598	0.0019	0.5643
	Successor node	47	47	47	47	47	47	47	40	40

**Table 3-4: Probabilities of Arriving at the Destination Node $N = 49$
Within Time t or Less Given an Optimal Choice of Successor Nodes Using the
Second Set of Gamma Distributions (Continue)**

	$u_{47}(t)$	0.8366	0.8315	0.8085	0.7364	0.6063	0.4203	0.2127	0.0372	0.3763
47	Successor node	48	48	48	48	48	48	41	41	41
	$u_{48}(t)$	0.9998	0.9965	0.9925	0.9595	0.8981	0.7737	0.5992	0.3089	0.1883
48	Successor node	49	49	49	49	49	49	49	49	49
49	$u_{49}(t)$	1	1	1	1	1	1	1	1	0

3.4.3.3 Application to a 100-node network

A network with 100 nodes is tested to further examine the applicability of the algorithm to larger network and the sensitivity of the procedure to cumulative computational error. The link travel time distributions are set to be the same as in the first 49-node network. Refer to Equations (3.28) for the gamma distribution and Equations (3.31 and 3.32) for the parameters α and n . The results of the maximum probability of arriving on time and the optimal successor nodes are given in Appendix F. These results are compared with those from the average shortest path problem. See Table F-1. The probability functions are plotted in Figure F-1 for better visual convenience. These functions are grouped in nine clusters, depending on the number of minimum intermediate links between an origin and the destination. Note that the difference of link travel time distributions is slight. The functions look smooth and monotone in all the cases.

The optimal successor nodes recommended by SOTA and shortest path procedures tend to agree with one another in most cases. In some cases, the SOTA procedure may recommend a different successor node if there is more time for on time arrival. These are the general pattern of the results observed in this example, which is similar as what has been observed in the 49-node network with small variance. However, there are exceptions. For a few starting nodes (nodes 71, 72, 74, 81, 82, and 83), the optimal successor nodes change frequently among several closest surrounding nodes as the remaining time for on-time arrival diminishes. This is unusual relative to other examples, and inconsistent with our anticipated results. These anomalous results create concern that cumulative computational errors and the numerical instability problem associated with inverting Laplace transforms might be degrading the quality of the outputs. Close examination of the maximum probability functions for the alternative successor nodes mitigates these concerns. Consider node 71. The recommended successor strategies for node 71 are to go to node 62 if the remaining time for on time arrival is 3.9193, to go to node 82 if the remaining time is 2.2861, to go to node 72 if the remaining time is 1.4387, to go to node 62 if the remaining time is 0.8958. Since travel time distributions on the links (71, 62), (71, 72) and (71, 82) are very similar, the optimal successor node depends mainly on the maximum probability of on time arrival given a travelers location at these successor nodes. The maximum probability functions of on time arrival from these alternative successor nodes are plotted in Figure F-2. These functions are all smooth and monotone, but increase at different rates. For some values of t , $u_{62}(t)$ is largest,

while for other values of t , $u_{72}(t)$ or $u_{82}(t)$ is largest. Therefore, the oscillation of optimal successor nodes seems reasonable. However, the exact impact of cumulative computational error on these results remains unclear.

Note that when the network becomes larger and link travel time distributions of adjacent links are very similar, the impact of the cumulative computational error may have relatively larger impact on the results. Another 100-node network, where travel time distributions of adjacent links are purposely designed to have greater difference, is tested to further study the possible reason of the frequent change of successor nodes observed in the last example. The parameters are set to be

$$n = 0.2 \log_{10}(10 + 4i + 6j), \quad (3.34)$$

and

$$\alpha = \log_{10}(2 + 6i + 10j), \quad (3.35)$$

which results in larger difference of link travel time distributions among the adjacent links. We hope that the relative impact of the cumulative computational error will be diminished to some degree as the difference of link travel times increases. The frequent change of the optimal successor nodes does not occur in this example. The results of maximum probability of arriving on time and the optimal successor nodes behave as the general pattern observed in other examples. These results are given in Appendix G. Similar as in other examples, these results are compared with those from the average shortest path problem. See Table G-1. The probability functions are plotted in Figure G-1 for better visual convenience.

3.5 Changing Time Scale

In previous sections, we used Gaussian quadrature to numerically evaluate the Laplace transforms and the inverse of Laplace transforms of convolution integrals. The unknown functions $u_i(t)$, $i = 1, 2, \dots, N$ and $0 \leq t \leq \infty$, are evaluated at a discrete set of time values $t_i = -\ln \tau_i$, where τ_i are the M Gaussian quadrature points. For example, $M = 8$ defines the quadrature points and times in Table 3-1. Note that the unknown function $u(t)$ is determined for only a small range of t -values. Increasing M increases the upper bound of t . However, if the objective is to increase the range of t , then increasing M is an inefficient approach. All of the quadrature points are defined on the interval 0 to 1, so as M becomes large the columns in the matrix T become nearly linearly dependant, and it becomes difficult to solve Equation (3.20) for X . Computational cost increases quickly with M , and the quality of results degrades.

3.5.1 Multiplicative property of the Laplace transform

Fortunately, the range of t can be efficiently extended by relying on the multiplicative property of the Laplace transform. If the Laplace transform of function $f(t)$ is $F(s)$, then the Laplace transform of $f(at)$ is $F(s/a)/a$. This is because if

$$L\{f(t)\} = \int_0^{\infty} e^{-st} f(t) dt = F(s), \quad (3.36)$$

then

$$L\{f(at)\} = \int_0^{\infty} e^{-st} f(at) dt$$

$$\begin{aligned}
&= \frac{1}{a} \int_0^{\infty} e^{-st/a} f(t) dt \\
&= \frac{F(s/a)}{a}.
\end{aligned} \tag{3.37}$$

Replacing the values $F(1), F(2), \dots, F(N)$ with the values $F(1/a), F(2/a), \dots, F(N/a)$ in Equation (3.17) makes it possible to approximate the unknown function $u(t)$ for the values $t = -a \ln \tau_i$ by solving the system of linear algebraic equations

$$\sum_{i=1}^n \tau_i^{s-1} u(-a \ln \tau_i) w_i \cong \frac{F(s/a)}{a}, \quad s = 1, 2, \dots, N. \tag{3.38}$$

Consider the convolution integral in Equation (3.1),

$$f(t) = \int_0^t p(\omega) u(t - \omega) d\omega. \tag{3.39}$$

Let $F(s)$, $P(s)$, and $U(s)$ be the respective Laplace transforms of functions $f(t)$, $p(t)$ and $u(t)$ in Equation (3.39), where $u(t)$ corresponds to any $u_j(t)$ and $p(\omega)$ corresponds to any associated $p_{ij}(\omega)$. According to the convolution theorem of Laplace transforms,

$$F(s) = P(s)U(s). \tag{3.40}$$

In our previous work, the unknown function $f(t)$ is to be evaluated at a discrete set of values, t_1, t_2, \dots, t_N . Defining

$$y(t) = f(at) \tag{3.41}$$

makes it possible to evaluate the function $f(t)$ over the sequence, at_1, at_2, \dots, at_N . By the multiplicative property of Laplace transforms,

$$Y(s) = \frac{1}{a} F\left(\frac{s}{a}\right). \quad (3.42)$$

Replacing the value s by s/a in Equation (3.40) and substituting into Equation (3.42) gives

$$Y(s) = \frac{1}{a} P(s/a) U(s/a), \quad (3.43)$$

where

$$P(s/a) \cong a \sum_{i=1}^n \tau_i^{s-1} p(-a \ln \tau_i) w_i, \quad s = 1, 2, \dots, N, \quad (3.44)$$

and

$$U(s/a) \cong a \sum_{i=1}^n \tau_i^{s-1} u(-a \ln \tau_i) w_i, \quad s = 1, 2, \dots, N. \quad (3.45)$$

The matrix T in equations (3.44) and (3.45) is same as the matrix T in equation (3.20). Once $Y(s)$, the Laplace transform of function $y(t)$, is known, the inverse of $Y(s)$ can be obtained by solving the set of linear algebraic Equations (3.23) for the vector X .

3.5.2 Validation: comparing numerical and analytical results

3.5.2.1 Test 1: Inverting the Laplace transform of $\sin(at)$

We test our reliance on the multiplicative property of the Laplace transform by estimating the Laplace transform for functions $\sin(at)$, $a = 1, 2, 3$, and then inverting the results to produce an estimate of the original function. These estimates reproduce the original function nicely. Further, both the range of t and the number of

t -values is increased. The numerical results are shown graphically in **Figure 3-4**, and more precisely in Table 3-5.

3.5.2.2 Test 2: Evaluating a convolution integral when $u(t) = 1$

Consider the simplest meaningful initial approximations for $u_i(t)$, the probability of arriving at N on or before time t given a current location at node i . The simplest available estimates are to set $u_i(t)$ to zero for all nodes other than node N , and to 1 for node N .

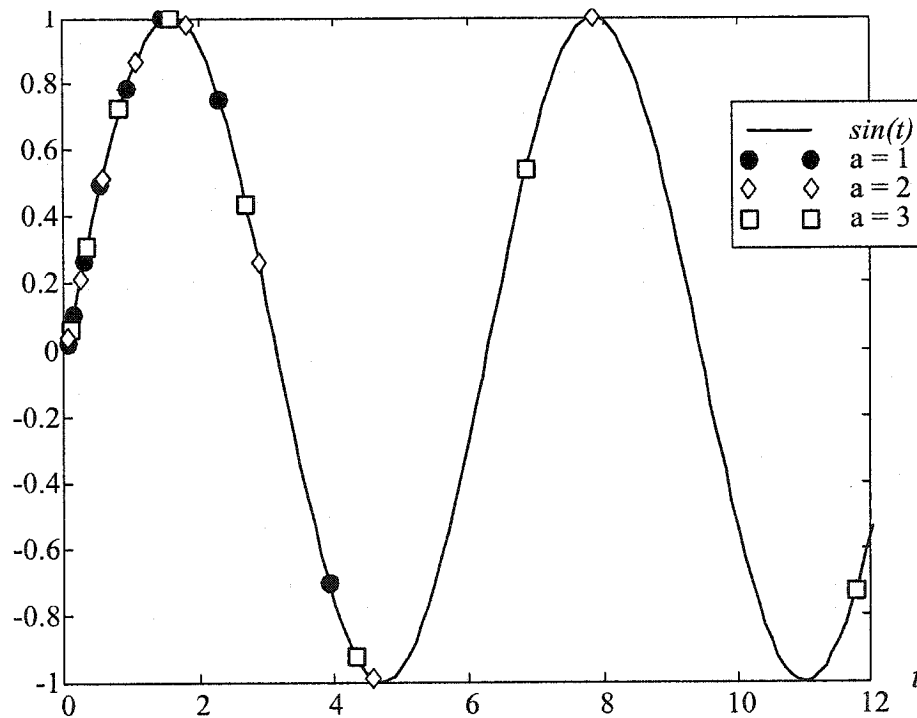


Figure 3-4: Changing the Time Scale Parameter a for $\sin(at)$

**Table 3-5: Numerical Comparison of $\sin(at)$ with Inverted Laplace Transforms
of $\sin(at)$**

a = 1			a = 2			a = 3		
at	sin(at) exact	sin(at) estimated	at	sin(at) exact	sin(at) estimated	at	sin(at) exact	sin(at) estimated
3.9193	-0.7016	-0.7019	7.8386	0.9999	0.9995	11.7579	-0.7232	-0.7238
2.2861	0.7549	0.7545	4.5721	-0.9902	-0.9908	6.8582	0.5438	0.5430
1.4387	0.9913	0.9917	2.8774	0.2611	0.2619	4.3161	-0.9225	-0.9215
0.8958	0.7807	0.7806	1.7916	0.9757	0.9756	2.6874	0.4387	0.4386
0.5247	0.5010	0.5006	1.0495	0.8672	0.8664	1.5742	1.0000	0.9991
0.2708	0.2675	0.2672	0.5416	0.5155	0.5150	0.8124	0.7259	0.7253
0.1072	0.1070	0.1070	0.2144	0.2128	0.2127	0.3216	0.3161	0.3160
0.0201	0.0201	0.0199	0.0401	0.0401	0.0398	0.0602	0.0602	0.0597

Referring again to Equation (3.39), define the functions $p(t)$ and $u(t)$ to be e^{-t} and 1, respectively, what is $y(t) = f(at)$, $a = 1, 2, 3$, corresponding to Equation (3.41)? Applying Equations (3.42) through (3.45) provides an estimate of the Laplace transform $Y(s)$. Applying Equation (3.23) produces an estimate of $y(t)$, which is also $f(at)$.

Figure 3-5 and Table 3-6 compare the results obtained from this procedure with the results provided by applying the MatLab Quad function to Equation (3.41) over the interval $t = 0$ to 15.

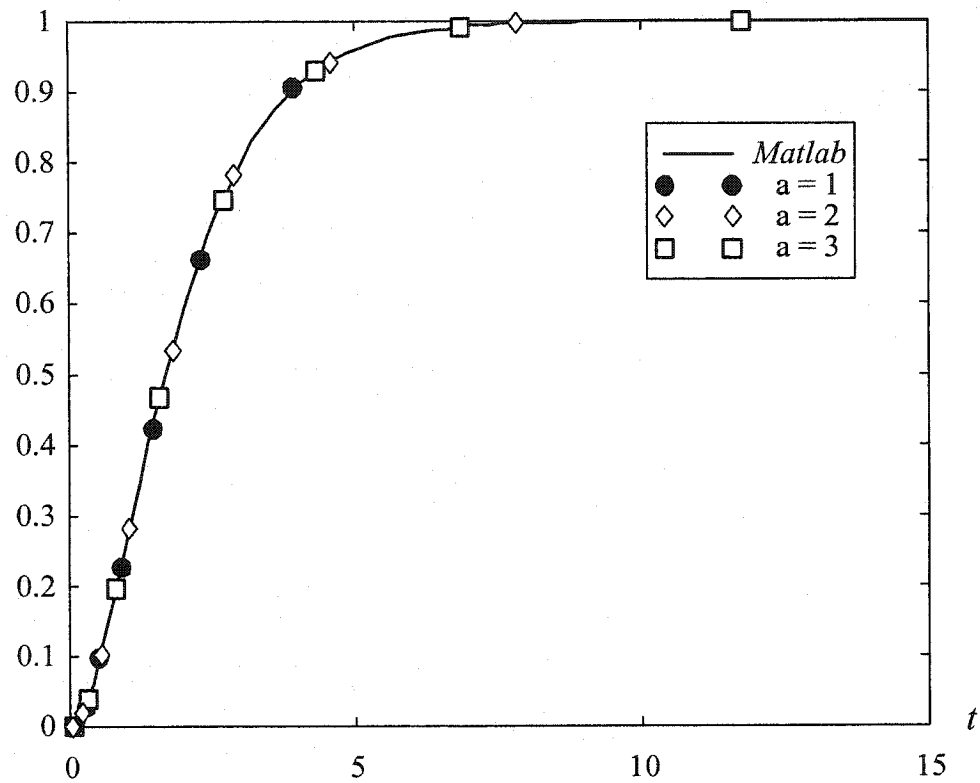


Figure 3-5: Changing Time Scale for Function of Convolution Integral

The Quad function numerically evaluates integrals over a finite interval, and thus provides a convenient benchmark when $p(t)$ and $u(t)$ are known. The points obtained by using different values of a all seem to lie on a smooth curve. They also agree with Matlab results. Unfortunately, in the network case, the functions $u(t)$ are not generally known.

Table 3-6: Numerical Comparison of MATLAB Quad Function Estimates of $y(t) = f(at)$ with Estimates Obtained via the Multiplicative Property of Laplace Transforms

a = 1			a = 2			a = 3		
at	quad	f(at)	at	quad	f(at)	at	quad	f(at)
3.9193	0.9023	0.9066	7.8386	0.9965	0.9961	11.7579	0.9999	0.9998
2.2861	0.6659	0.6638	4.5721	0.9424	0.9424	6.8582	0.9917	0.9914
1.4387	0.4215	0.4227	2.8774	0.7818	0.7818	4.3161	0.9290	0.9294
0.8958	0.2260	0.2252	1.7916	0.5346	0.5347	2.6874	0.7490	0.7490
0.5247	0.0978	0.0983	1.0495	0.2824	0.2822	1.5742	0.4667	0.4664
0.2708	0.0307	0.0302	0.5416	0.1031	0.1030	0.8124	0.1957	0.1954
0.1072	0.0054	0.0056	0.2144	0.0200	0.0199	0.3216	0.0419	0.0418
0.0201	0.0002	0.0000	0.0401	0.0008	0.0007	0.0602	0.0017	0.0016

3.5.2.3 Test3: Application on a 9-node Network

A 9-node network is used to test whether this procedure for changing the time scale is computable and efficient in our network context. See Figure 3-6. The parameters of the Gamma distribution for each link are arbitrarily defined using the following rules,

$$n = 1.4 \log_{10}(10 + 0.8i + 0.7j), \text{ and} \quad (3.46)$$

$$\alpha = 2 \log_{10}(12 + 1.2i + 0.8j), \quad (3.47)$$

where i and j are the starting and ending nodes for link ij . This ensures that the link travel time functions are similar, but slightly differ from one another. Under these circumstances, the optimal probability of arriving on time will be most strongly

influenced by the number of intermediate nodes between the origin and destination node 9.

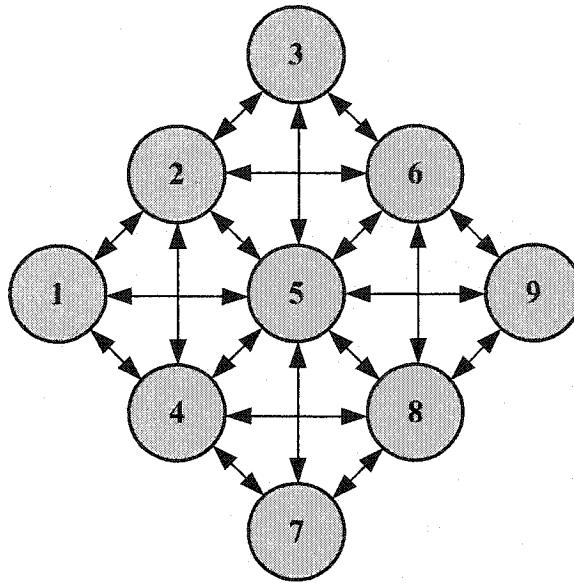


Figure 3-6: A 9-node Network

Three time scales are used, $a = 0.5$, $a = 1$, and $a = 2$. The results for these time scales are combined in Table 3-77. As expected, the probabilities $u_i(t)$, $i = 1, 2, \dots, n$, $0 \leq t \leq \infty$ can be clustered into three groups depending on the minimum number of intermediate nodes between starting node i and destination node $N = 9$. These groups are node 9; nodes 5, 6, and 8; and nodes 1, 2, 3, 4, and 7. Given the similarities in the link travel time distributions, the probability of arriving on time is necessarily higher when the origin node is closer to the destination. Node $N = 9$ is the destination, and thus a trivial case. The probabilities $u_i(t)$ in the other two groups

vary across the group members, but these difference are small enough that they cannot be distinguished graphically in this example. The averaged values of probabilities $u_i(t)$ for each value of t across nodes 1, 2, 3, 4, and 7 ($u_{12347}(t)$) and across nodes 5, 6, and 8 ($u_{568}(t)$) are plotted in Figure 3-7.

Table 3-7: Probabilities of Arriving on Time at Destination Node N = 9 within Time t: Increasing the Number and Range of Time Points by Changing the Time Scale

t	$u_1(t)$	$u_2(t)$	$u_3(t)$	$u_4(t)$	$u_5(t)$	$u_6(t)$	$u_7(t)$	$u_8(t)$	$u_9(t)$
0.01	0.000	0.000	0.000	0.000	0.001	0.001	0.000	0.001	1.000
0.0201	0.000	0.000	0.000	0.000	0.003	0.003	0.000	0.002	1.000
0.0401	0.000	0.000	0.000	0.000	0.010	0.010	0.000	0.009	1.000
0.0536	0.002	0.002	0.002	0.002	0.016	0.016	0.002	0.015	1.000
0.1072	0.000	0.000	0.000	0.000	0.053	0.052	0.000	0.049	1.000
0.1354	0.002	0.001	0.001	0.001	0.078	0.076	0.001	0.073	1.000
0.2144	0.009	0.008	0.008	0.008	0.157	0.155	0.007	0.151	1.000
0.2624	0.016	0.016	0.015	0.015	0.209	0.207	0.014	0.203	1.000
0.2708	0.017	0.016	0.016	0.016	0.218	0.216	0.015	0.213	1.000
0.4479	0.067	0.066	0.065	0.064	0.411	0.410	0.062	0.408	1.000
0.5247	0.101	0.099	0.098	0.097	0.487	0.486	0.094	0.485	1.000
0.5416	0.109	0.108	0.106	0.105	0.502	0.502	0.102	0.501	1.000
0.7194	0.208	0.207	0.205	0.204	0.648	0.648	0.201	0.649	1.000
0.8958	0.318	0.317	0.317	0.316	0.756	0.757	0.313	0.759	1.000
1.0495	0.418	0.417	0.417	0.417	0.825	0.826	0.415	0.828	1.000
1.143	0.474	0.474	0.474	0.474	0.858	0.860	0.474	0.862	1.000
1.4387	0.641	0.642	0.643	0.644	0.928	0.930	0.645	0.932	1.000
1.7916	0.787	0.788	0.789	0.792	0.969	0.970	0.795	0.971	1.000
1.9596	0.840	0.841	0.843	0.846	0.979	0.979	0.848	0.980	1.000
2.2861	0.906	0.908	0.909	0.912	0.990	0.991	0.915	0.992	1.000
2.8774	0.970	0.971	0.971	0.973	0.999	0.999	0.974	0.999	1.000
3.9193	0.996	0.996	0.996	0.997	1.000	1.000	0.997	1.000	1.000
4.5721	1.000	1.000	1.000	1.000	0.999	0.999	0.999	0.999	1.000
7.8386	1.007	1.007	1.006	1.007	1.007	1.006	1.005	1.005	1.000

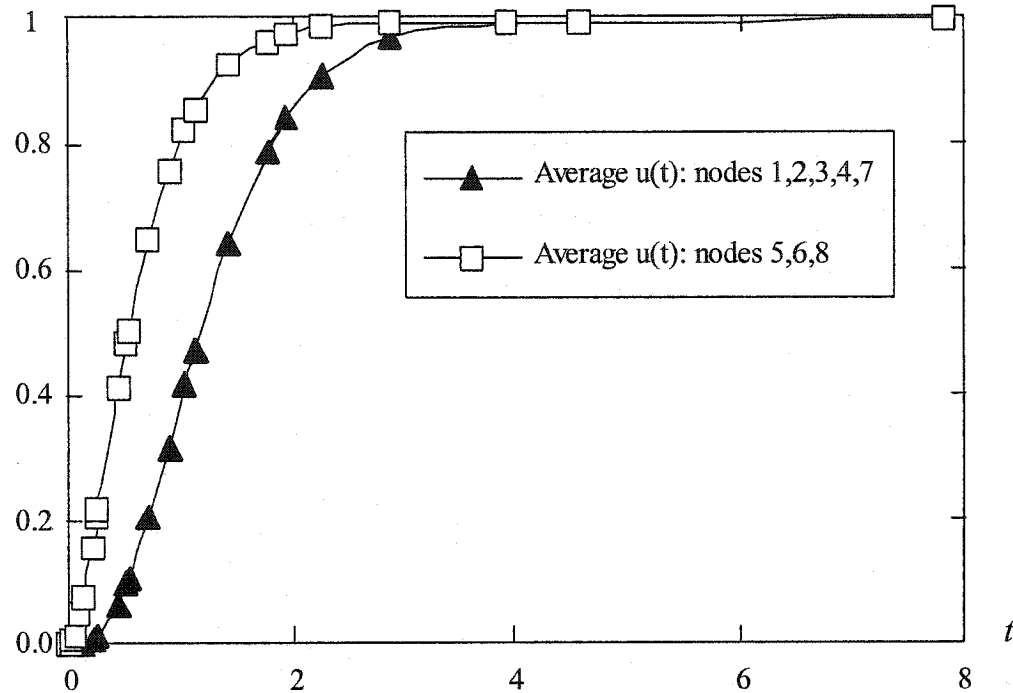


Figure 3-7: Maximum Probability of Arriving On Time from Each Node Using Changing Time Scale Procedure

3.6 Alternative Method of Evaluating Convolution Integrals via Differential Approximation

In the previous sections, we have shown how to evaluate a convolution integral,

$$z(t) = \int_0^t u(t-\tau)f(\tau)d\tau, \quad (3.48)$$

using the convolution theorem of Laplace transform. In this section, we will provide an alternative means of evaluating convolution integrals via solving ordinary

differential equations. Our intentions are, first, to seek for the possible existence of a more efficient way to evaluate convolution integrals, second, to validate our approach of evaluating convolution integrals via method of Laplace transforms presented in the previous sections.

3.6.1 Method of differential approximation

3.6.1.1 Seeking for differential relationships for the convolution integrals

If the kernel of the integrand satisfies a second order linear differential equation, then the convolution integral also satisfies a linear differential relationship. The proof is given as follows. If $z(t)$ is defined by Equation (3.48), then

$$z' = u(0)f(t) + \int_0^t u'(t-\tau)f(\tau)d\tau, \quad (3.49)$$

and

$$z'' = u(0)f'(t) + u(0)f(t) + \int_0^t u''(t-\tau)f(\tau)d\tau. \quad (3.50)$$

Based on Equations (3.48 – 3.50), we have

$$\begin{aligned} z'' + az' + bz &= [au(0) + u'(0)]f(t) + u(0)f'(t) + \int_0^t [bu(t-\tau) + au'(t-\tau) + u''(t-\tau)]f(\tau)d\tau, \end{aligned} \quad (3.51)$$

where a and b are coefficients to be determined from the linear differential equation of the kernel function.

If the kernel satisfies a linear differential equation; that is, if

$$u'' + au' + bu = 0, \quad (3.52)$$

then

$$z'' + az' + bz = [au(0) + u'(0)]f(t) + u(0)f'(t). \quad (3.53)$$

In the previous work, link travel times were assumed to follow a gamma distribution. That is

$$p(t) = \frac{\alpha^n e^{-\alpha t} t^{n-1}}{\Gamma(n)}. \quad (3.54)$$

The first order derivative and second order derivative are

$$p' = \frac{\alpha^n}{\Gamma(n)} [-\alpha e^{-\alpha t} t^{n-1} + (n-1)e^{-\alpha t} t^{n-2}], \quad (3.55)$$

and

$$p'' = \frac{\alpha^n}{\Gamma(n)} [\alpha^2 e^{-\alpha t} t^{n-1} - 2\alpha(n-1)e^{-\alpha t} t^{n-2} + (n-1)(n-2)e^{-\alpha t} t^{n-3}]. \quad (3.56)$$

Therefore, we have

$$p'' + ap' + bp = \frac{\alpha^n e^{-\alpha t} t^{n-3}}{\Gamma(n)} (\alpha^2 - a\alpha + b)t^2 + (a - 2\alpha)(n-1)t + (n-1)(n-2). \quad (3.57)$$

To let

$$p'' + ap' + bp = 0, \quad (3.58)$$

we must solve the following set of equations

$$\left\{ \begin{array}{l} \alpha^2 - a\alpha + b = 0, \end{array} \right. \quad (3.59)$$

$$\left\{ \begin{array}{l} (\alpha - 2\alpha)(n-1) = 0, \end{array} \right. \quad (3.60)$$

$$\left\{ \begin{array}{l} \text{and} \\ (n-1)(n-2) = 0. \end{array} \right. \quad (3.61)$$

Consequently, the value of n must be either 1 or 2. The values of a and b must be

$$a = 2\alpha, \quad (3.62)$$

and

$$b = \alpha^2. \quad (3.63)$$

Plug the values of a and b constrained by Equation (3.62, 3.63) to Equation (3.58), the function $p(t)$ has the following relationship

$$p'' + 2\alpha p' + \alpha^2 p = 0. \quad (3.64)$$

Equation (3.53) then becomes

$$z'' + 2\alpha z' + \alpha^2 z = [2\alpha p(0) + p'(0)]u(t) + p(0)u'(t). \quad (3.65)$$

The initial conditions of function $p(t)$ are

$$p(0) = 0, \quad (3.66)$$

and

$$p'(0) = \alpha^2 [e^{-\alpha t} + (-\alpha)e^{-\alpha t}t] = \alpha^2. \quad (3.67)$$

Therefore, Equation (3.65) can be written as

$$z'' + 2\alpha z' + \alpha^2 z = \alpha^2 u(t). \quad (3.68)$$

So far, we have shown that in case where the coefficients of the primary ODE of the kernel function are all constant, convolution integrals can be evaluated by

solving a resultant nonhomogenous differential equation with same coefficients. With such an equation, the output could be computed directly as a linear process simulation without any further computation of integrals. The detail procedure of solving differential equations via modified Euler's method is described in the next section right below.

If a function does not satisfy a second order linear differential equation analytically, there are ways to approximate a differential equation in a least square sense (Bellman and Kalaba, 1964)

3.6.1.2 Procedure of solving differential equations

Consider solving the specific differential equation defined by Equation (3.68). Let z' be y , Equation (3.68) can be written as two first order differential equations. They are

$$\left\{ \begin{array}{l} z' = y, \\ \text{and} \\ y' = -2\alpha y - \alpha^2 z + \alpha^2 u(t). \end{array} \right. \quad (3.69)$$

(3.70)

The unknown function $z(t)$ and $y(t)$ can be solved through modified Euler's method. The detailed procedure of this method is described in the flowchart shown in Figure 3-8.

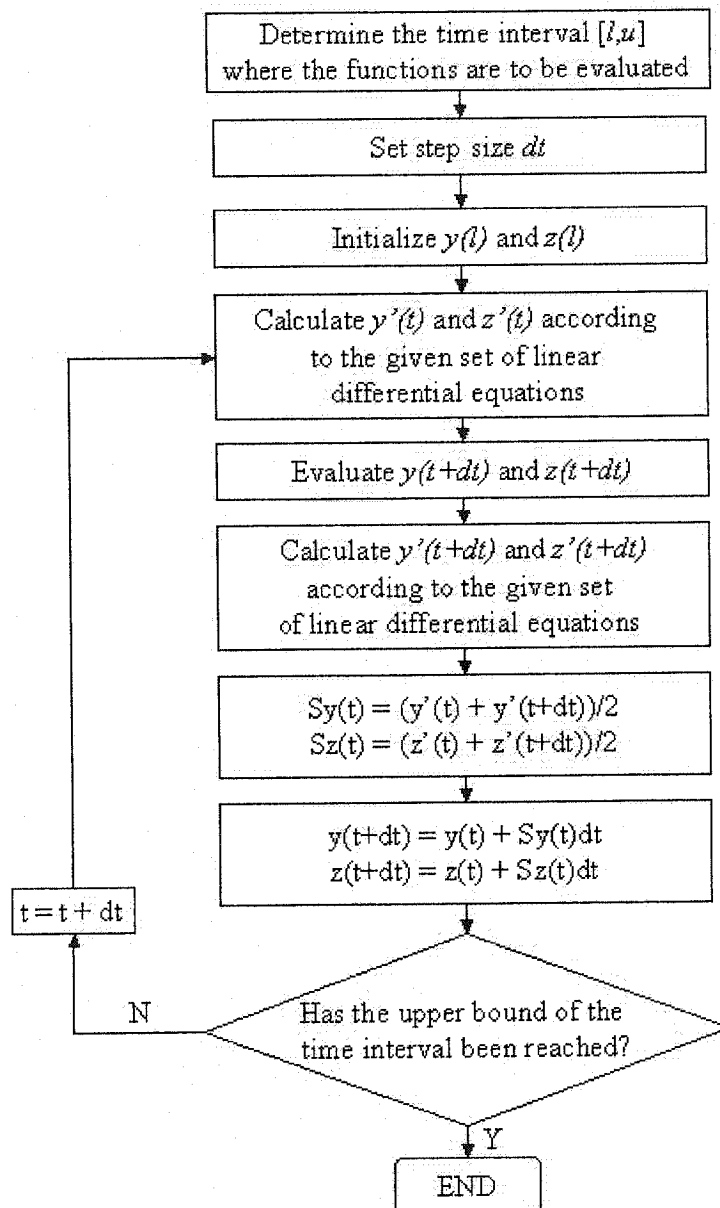


Figure 3-8: Procedure of Solving Ordinary Differential Equations Using Modified Euler's Method

3.6.2 Numerical examples

3.6.2.1 Test 1

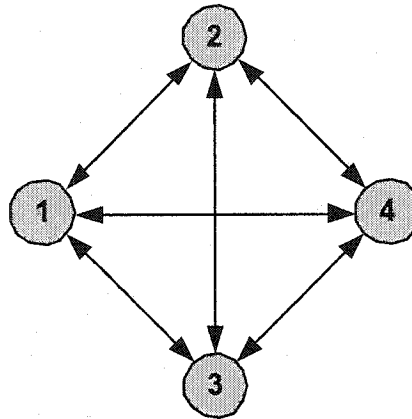


Figure 3-9: A 4-node Network with Same Link Travel Time Distributions

Consider a four-node network as shown in Figure 3-9. Assume the travel time distribution of each link to be

$$p(t) = te^{-t} \quad (3.71)$$

We pick this $p(t)$ on purpose so that it agrees with a homogenous linear differential equation. Therefore, the convolution integral can be evaluated by solving the resultant nonhomogenous linear differential equation. The time step in the procedure of differential approximation is set to be 0.01. A MATLAB program for this task is available in Appendix H.

The results from the method of differential approximation and the method of Laplace transform are compared in Table 3-8. Note that the results from the

Table 3-8: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation

t	u(t) from ODE	t	u(t) from Laplace transform
		0.0201	0.0001
0.09	0.0038		
		0.1072	0.0056
0.19	0.0159		
		0.2708	0.0304
0.29	0.0347		
0.39	0.0589		
0.49	0.0872		
		0.5247	0.0983
0.59	0.1186		
0.69	0.1523		
0.79	0.1876		
0.89	0.2239		
		0.8958	0.2255
0.99	0.2606		
1.09	0.2973		
1.19	0.3338		
1.29	0.3696		
1.39	0.4047		
		1.4387	0.4227
1.49	0.4388		
1.59	0.4718		
1.69	0.5036		
1.79	0.5342		
1.89	0.5634		
1.99	0.5913		
2.09	0.6178		
2.19	0.643		
		2.2861	0.6646
2.29	0.6668		
2.39	0.6894		
2.49	0.7106		
2.59	0.7307		
2.69	0.7495		
2.79	0.7672		
2.89	0.7838		
2.99	0.7994		

Table 3-8: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation (Continued)

3.09	0.8139
3.19	0.8275
3.29	0.8402
3.39	0.852
3.49	0.863
3.59	0.8733
3.69	0.8829
3.79	0.8918
3.89	0.9
3.9193 0.9066	
3.99	0.9077
4.09	0.9148
4.19	0.9214
4.29	0.9275
4.39	0.9332
4.49	0.9384
4.59	0.9432
4.69	0.9477
4.79	0.9519
4.89	0.9557
4.99	0.9592

method of differential approximation are reported using time step 0.1 to save the space, while the actual time step used by the model is 0.01. Same quadrature points are used here as in other examples. These results are also plotted in Figure 3-10 for better visual convenience. The results from the two different methods agree with each other very well.

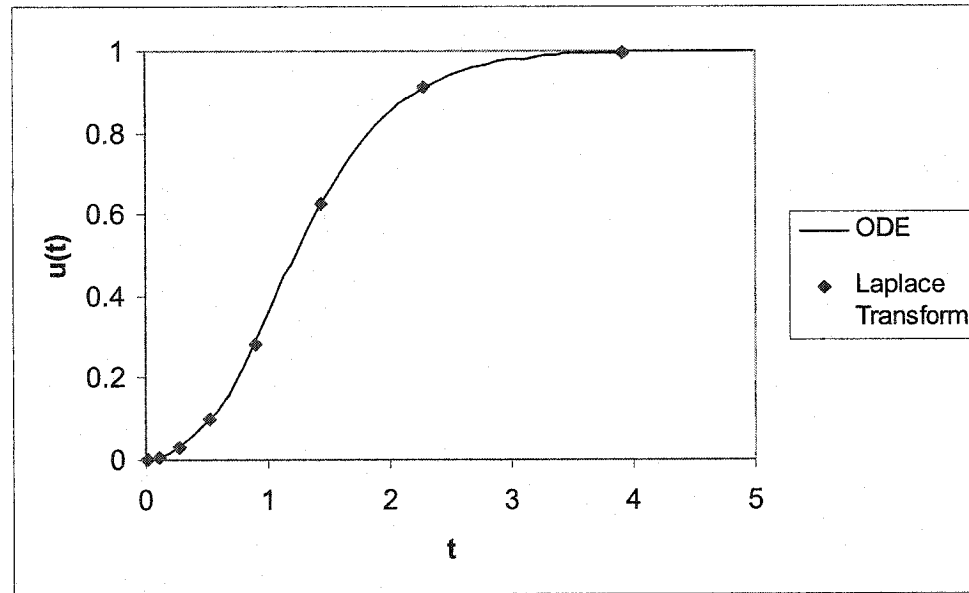


Figure 3-10: Maximum Probability of Arriving On Time from Node 2 to

Node 4

3.6.2.2 Test 2

Consider another four-node network as shown in Figure 3-11. Assume the travel time distribution of each link to be

$$p(t) = \alpha^2 t e^{-\alpha t} \quad (3.72)$$

The values of α for each link are shown in Figure 3-11. Similar to the previous example, the function $p(t)$ described by Equation (3.72) agrees with homogenous linear differential equations. Therefore, the convolution integral with such kernel can be evaluated by solving the resultant nonhomogenous linear differential

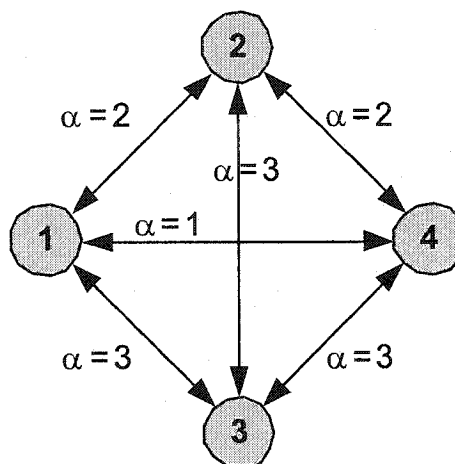


Figure 3-11: A 4-node Network with Different Link Travel Time Distributions

equation. The time step in the procedure of differential approximation is also set to be 0.01.

The results from the method of differential approximation and the method of Laplace transform are compared in Table 3-9. Note that the results from the method of differential approximation are reported using time step 0.1 to save the space, while the actual time step used by the model is 0.01. Same quadrature points are used here as in other examples. These results are also plotted in Figure 3-12 for better visual convenience. The results from the two different methods agree with each other very well.

Table 3-9: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation

t	u1		u2		u3	
	ODE	Laplace	ODE	Laplace	ODE	Laplace
0.0201		0.0002		0.0007		0.0016
0.09	0.0038		0.0144		0.0306	
0.1072		0.0056		0.0199		0.0418
0.19	0.0159		0.0563		0.1122	
0.2708		0.0304		0.103		0.1954
0.29	0.0347		0.1154		0.2166	
0.39	0.0589		0.1841		0.3266	
0.49	0.0872		0.2569		0.4321	
0.5247		0.0983		0.2822		0.4664
0.59	0.1186		0.3302		0.5282	
0.69	0.1558		0.4013		0.6126	
0.79	0.2151		0.4686		0.685	
0.89	0.2793		0.5312		0.7458	
0.8958		0.2828		0.5347		0.749
0.99	0.3461		0.5886		0.7963	
1.09	0.4131		0.6405		0.8377	
1.19	0.4784		0.6872		0.8713	
1.29	0.5407		0.7287		0.8984	
1.39	0.599		0.7655		0.9201	
1.4387		0.626		0.7818		0.9294
1.49	0.6526		0.7978		0.9374	
1.59	0.7012		0.8262		0.9511	
1.69	0.7446		0.8509		0.9619	
1.79	0.7831		0.8723		0.9703	
1.89	0.8168		0.8909		0.977	
1.99	0.8461		0.9069		0.9822	
2.09	0.8713		0.9207		0.9862	
2.19	0.8929		0.9326		0.9894	
2.2861		0.9103		0.9424		0.9914
2.29	0.9112		0.9428		0.9918	
2.39	0.9267		0.9515		0.9937	
2.49	0.9397		0.9589		0.9952	
2.59	0.9505		0.9652		0.9963	
2.69	0.9596		0.9706		0.9972	
2.79	0.9671		0.9752		0.9978	
2.89	0.9732		0.9791		0.9983	
2.99	0.9783		0.9823		0.9987	
3.09	0.9825		0.9851		0.999	

Table 3-9: Comparison between Results from the Method of Laplace Transform and the Method of Differential Approximation (Continued)

3.19	0.9859	0.9875	0.9993
3.29	0.9886	0.9895	0.9994
3.39	0.9909	0.9912	0.9996
3.49	0.9927	0.9926	0.9997
3.59	0.9942	0.9938	0.9998
3.69	0.9953	0.9948	0.9998
3.79	0.9963	0.9956	0.9999
3.89	0.997	0.9963	0.9999
3.9193	0.9973	0.9961	0.9998
3.99	0.9977	0.9969	0.9999
4.09	0.9981	0.9974	0.9999
4.19	0.9985	0.9978	1
4.29	0.9988	0.9982	1
4.39	0.9991	0.9985	1
4.49	0.9993	0.9987	1
4.59	0.9994	0.999	1
4.69	0.9996	0.9991	1
4.79	0.9996	0.9993	1
4.89	0.9997	0.9994	1
4.99	0.9998	0.9995	1

3.7 Summary

This research solves a stochastic network problem via applied mathematical techniques. We have shown that stochastic on-time arrival problems can be formulated using Bellman's Principle of Optimality and solved via Picard's method of successive approximation. Relying on the convolution theorem of Laplace transforms reduces the computational cost associated with evaluating convolution integrals. Linear algebraic equations are solved in a least square sense. The multiplicative property of Laplace transforms makes it possible to change the time

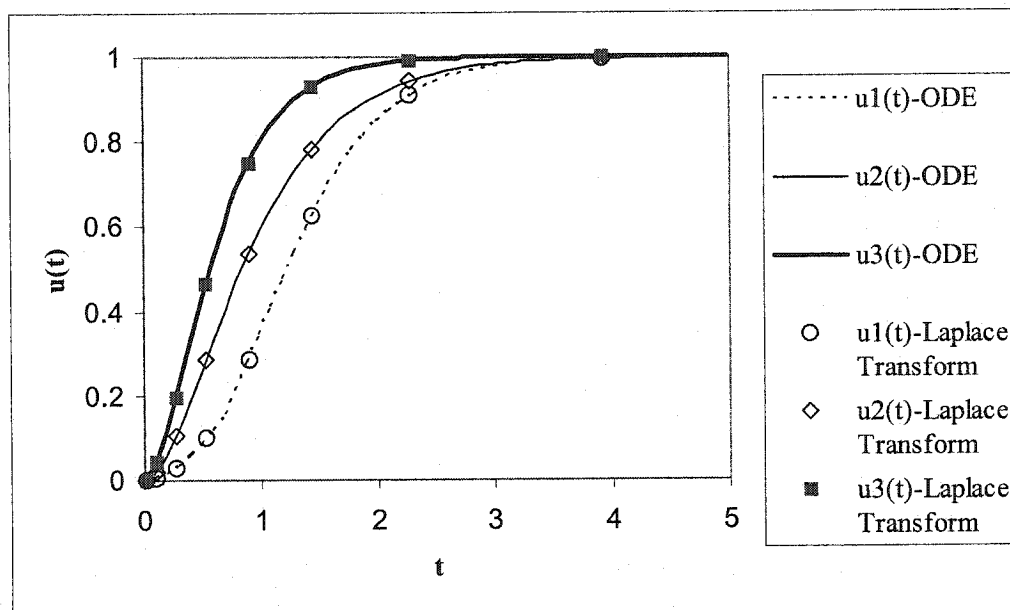


Figure 3-12: The Maximum Probability of Arriving on Time using the Method of Differential Approximation

scale of the problems to be solved and so to evaluate the unknown functions at more points in an efficient manner. Numerical examples and validation tests indicate that this numerical approach of solving the arriving-on-time problems is efficient and reliable.

Several methodological questions remain. First, the integer values s at which the Laplace transforms are evaluated were selected arbitrarily. It is possible that the quality of the results may be improved by choosing these values more systematically. Note that the value of s should not be too large. Refer to the definition of Laplace transform in Equation (3.7). If s becomes large, e^{-st} approaches zero when t is not close to 0. In this case, no matter what the value of $u(t)$ is, the product of e^{-st} and $u(t)$

will be close to 0. Therefore only $u(t)$ for t close to 0 contributes to the Laplace transform $U(s)$.

We know from computational experience that increasing the number of quadrature points does not necessarily improve the quality of the results. However, this relationship has not been studied closely and there may be circumstances in which increasing the number of quadrature points sufficiently improves the quality of results to justify the additional computational expense involved.

Note that other computational and analytical methods for dealing with convolution integrals are available. The method of Laplace transform demonstrates the feasibility of this approach and its performance with efficiency. This method is validated via comparison against the method of differential approximation. Based on our numerical experiments, evaluating a single convolution integral through differential approximation can be very accurate and reliable. However, implementation of this method to the arriving-on-time problem may cause huge computational cost because of the fine time step it requires. Evaluation of convolution integrals is in the most inner part of the iterations relative to the entire procedure of solving the arriving on time problem. Increasing number of iterations at inner level will increase the total number of operations dramatically. In our future research, we shall seek for other approaches that may offer advantages.

We evaluate the unknown functions $u_1(t)$, $u_2(t)$, ..., $u_N(t)$ describing the maximum probabilities of arriving on time and the optimal choice of the next node to visit for discrete points in time. The uniqueness and continuity of these functions

have not been proved mathematically here. Computing can be used as an experimental tool to explore the hidden nature of general solutions. Our extensive numerical experiments suggest that these functions should be unique and continuous, though perhaps not everywhere differentiable. The value of these functions for other times t , therefore, can likely be approximated via interpolation. However, it is not clear how to identify successor nodes for intermediate values of t for which $u(t)$ is not calculated.

Finally, the stochastic on-time arrival problem has been formulated based on the assumption that the travel times on any two links are independent. Further research is needed to accommodate the possibility of correlated travel times on adjacent links. Such correlation might be a prominent aspect of large scale applications, such as routing commercial airline to avoid meteorological threats. In contrast, correlation between link travel times may be less important in the context of urban road networks, because a large share of urban congestion is presumed to be nonrecurrent and the result of random accidents that reduce road capacity.

CHAPTER 4 OPTIMAL ROUTING THROUGH NETWORKS IN THE CASE OF CORRELATED LINK TRAVEL TIMES

Various problems of finding optimal paths have been studied extensively in the fields of computer science, operations research, and transportation engineering. Link costs are assumed to be independent in almost all of the path-finding literature. In transportation networks, knowledge of time costs on the link leading to the current decision point (node) may include information that should inform subsequent route decisions. How should these correlations between conditions in adjacent areas be accounted for by travelers? How should knowledge of travel costs acquired during the trip be used to inform subsequent choices of intermediate nodes?

This Chapter addresses the shortest path problem in congestible networks with correlated link travel costs. The motivation for this research is that natural disasters and accidents can be expected to affect a group of links or nodes in a specific region of the network. Node- or link-based service levels can be defined. A node is congested if the in-flow to that node exceeds the node's out-flow capacity. This definition is frequently applied in computer networks, airport operations, etc. A link is considered congested if the link travel time exceeds an acceptable value. Link-based congestion is frequently considered in roadway networks.

If the service level for one link or node is affected by some random incident, it is possible that adjacent links and nodes are also affected. If so, then experience on the link leading to the current decision node should be taken into account when making the optimal decision about the next intermediate node to be visited. The

objective is to minimize expected travel time to the destination. The problem will be formulated separately for each of the two different definitions of congestion. These two formulations are subsequently shown to be equivalent.

In this problem, each link is assumed to be in one of two possible states, congested or un-congested. Conditional probability density functions for link travel times are assumed known for each state. The traveler takes into account his experience on the link leading to the current decision point (node) when determining which node to visit next, i.e., which link to traverse next. Only the next immediate link in the path is identified, and the knowledge gained during the course of the trip is used to optimally support the decisions defining the remaining journey.

We have two principal goals in this chapter. The first is to formulate such problems. The second is to show that our formulation of the problem has a unique solution, and that a computable means for finding this solution is available.

4.1 Statement of the Problem

Consider a network with N nodes and various connections between them. In a transportation network, performance may be affected by non-recurrent events, such as (in the extreme case) natural disasters or (more commonly) traffic incidents. These random reductions in supply make network performance uncertain. Given an origin-destination pair, our objective is to define the sequence of nodes to be visited such that the lowest expected travel time is achieved.

Given the node-based definition of congestion, each node is assumed to have two possible states, congested or uncongested. In the case of a natural disaster, these states might be more generally labeled “affected” and “unaffected.” The correlations between the states of adjacent nodes are taken into account by introducing two probabilities, α_{ij} and β_{ij} . The probability that, if node i is uncongested, then node j is uncongested is α_{ij} . The probability that if, node i is congested, then node j is congested is β_{ij} . For the sake of subsequent notational convenience, define λ_{ij} as the probability that, if node i is congested, then node j is uncongested, i.e., $1 - \beta_{ij}$. These probabilities are assumed to be a priori knowledge. They are similar to but less restrictive than the *a priori* correlations assumed by Burton (1993).

Similarly, given the definition of link-based congestion, each link is assumed to have two possible states, congested or uncongested. A link is considered uncongested if the time required to traverse this link ij is within an a priori bound t_{0ij} , and considered congested otherwise. The distributions of link travel times associated with each state are described by known probability density functions. Introduce the function $p_{ij}(t)$ as the probability density function of link travel time on link ij given that the link traversed to arrive at node i was uncongested. Introduce the function $q_{ij}(t)$ as the probability density function of link travel time on ij given that the link traversed to arrive at node i was congested. The average link travel times between node i and node j is t_{ij} under uncongested conditions, and τ_{ij} under congested conditions.

If a traveler experiences congestion at the current node or on the current link, he assumes similar conditions exist on adjacent nodes or links, and applies the corresponding probability or probability density function to structure his decision about which node to visit next.

4.2 Formulation of the Problem

According to Bellman's Principle of Optimality, an optimal sequence of decisions has the property that whatever the initial state and decision are, the remaining decisions must be optimal with respect to the state resulting from the initial decision (Bellman and Kalaba, 1965).

Consider the sub-network in Figure 4-1. Suppose the traveler has traversed some sequence of links and is now at node i . He wants to arrive at the destination

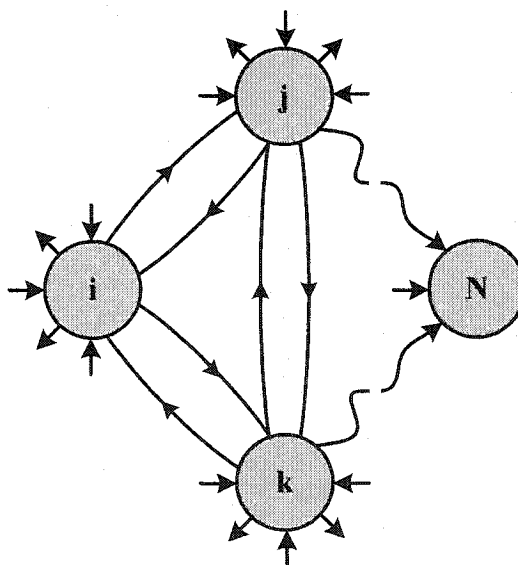


Figure 4-1: A General Sub-Network

node N as soon as possible. There are competing options for the next possible node to visit from node i , including nodes j and k . If he visits node j next, he then needs to make an optimal choice of the next node to visit from node j . If he visits node k the next, he then needs to make an optimal choice of the next node to visit from node k . In any case, his objective is to select the next node to visit such that the expected time until arriving at the destination node N is minimized.

4.2.1 Node-Based Congestion

In the case of node-based congestion, Bellman's principle of optimality may be applied to formulate this problem as

$$u_i = \min_{j \neq i} \{t_{ij} + \alpha_{ij}u_j + (1 - \alpha_{ij})v_j\}, \quad i = 1, 2, \dots, N-1, \quad (4.1)$$

$$v_i = \min \{\tau_{ij} + (1 - \beta_{ij})u_j + \beta_{ij}v_j\}, \quad i = 1, 2, \dots, N-1, \quad (4.2)$$

$$i = 1, 2, \dots, N-1$$

and

$$u_N, v_N = 0. \quad (4.3)$$

In these relations,

u_i = the lowest expected travel time from uncongested node i to the destination node N , and

v_i = the lowest expected travel time from congested node i to the destination node N .

Current conditions at each decision node must be considered, and an optimal choice of the next node to visit must be made. If the current node i is uncongested,

then the function u_i is evaluated to select the optimal next node j . Otherwise, the function v_i is evaluated. The condition of node j is ascertained once the traveler arrives there. The problem to be solved at node j then can be formulated in the same fashion as for node i . The traveler's decision at node j is conditioned on the new knowledge the traveler acquires about the state of node j once the traveler arrives there. Thus it is apparent that no optimal path can be entirely pre-determined.

4.2.2 Link-Based Congestion Case

In the case of link-based congestion, Bellman's principle of optimality may be applied to formulate this problem as

$$u_i = \min_{j \neq i} \left\{ \int_0^{t_{0ij}} (\tau + u_j) p_{ij}(\tau) d\tau + \int_{t_{0ij}}^{\infty} (\tau + v_j) p_{ij}(\tau) d\tau \right\}, i = 1, 2, N-1, \quad (4.4)$$

$$v_i = \min_{j \neq i} \left\{ \int_0^{t_{0ij}} (\tau + u_j) q_{ij}(\tau) d\tau + \int_{t_{0ij}}^{\infty} (\tau + v_j) q_{ij}(\tau) d\tau \right\}, i = 1, 2, N-1, \quad (4.5)$$

and

$$u_N, v_N = 0. \quad (4.6)$$

In these relations,

- t_{0ij} = a predefined acceptable link travel time. Link ij is considered unclogged if the travel time over link ij does not exceed t_{0ij} , and congested otherwise;
- u_i = the least expected time needed to travel from i to N if the link traversed to arrive at node i is unclogged;
- v_i = the least expected time to travel from i to N if the link traversed to arrive at node i congested;

$p_{ij}(\tau)d\tau$ = the probability traveling from i to j requires time between τ and $\tau+d\tau$

given that the link traversed to arrive at node i was uncongested; and

$q_{ij}(\tau)d\tau$ = the probability traveling from i to j requires time between τ and $\tau+d\tau$

given that the link traversed to arrive at node i was congested.

Partitioning and reorganizing the terms in the equations (4) and (5), we have

$$u_i = \min_{j \neq i} \left\{ \int_0^\infty \tau p_{ij}(\tau) d\tau + \int_0^{t_{0ij}} p_{ij}(\tau) d\tau \cdot u_j + \int_{t_{0ij}}^\infty p_{ij}(\tau) d\tau \cdot v_j \right\}, i = 1, 2, N-1, \quad (4.7)$$

and

$$v_i = \min_{j \neq i} \left\{ \int_0^\infty \tau q_{ij}(\tau) d\tau + \int_0^{t_{0ij}} q_{ij}(\tau) d\tau \cdot u_j + \int_{t_{0ij}}^\infty q_{ij}(\tau) d\tau \cdot v_j \right\}, i = 1, 2, N-1. \quad (4.8)$$

Note that the unknowns u_j and v_j are independent of τ , the variable of integration. Therefore, u_j and v_j can be taken out of the integral. Introduce the notation

$$\alpha_{ij} = \int_0^{t_{0ij}} p_{ij}(\tau) d\tau, \quad (4.9)$$

and

$$\lambda_{ij} = \int_0^{t_{0ij}} q_{ij}(\tau) d\tau. \quad (4.10)$$

Equation (4.10) gives the probability that the traveler traverses link ij within time t_{0ij} when the link is congested. This permits Equations (4.7) and (4.8) to be expressed as

$$u_i = \min_{j \neq i} \left\{ t_{ij} + \alpha_{ij} u_j + (1 - \alpha_{ij}) v_j \right\}, i = 1, 2, \dots, N-1, \quad (4.11)$$

and

$$v_i = \min_{j \neq i} \{ \tau_{ij} + \lambda_{ij} u_j + (1 - \lambda_{ij}) v_j \}, i = 1, 2, \dots, N-1, \quad (4.12)$$

where

$$t_{ij} = \int_0^\infty \tau p_{ij}(\tau) d\tau, \quad (4.13)$$

$$\tau_{ij} = \int_0^\infty \tau q_{ij}(\tau) d\tau, \quad (4.14)$$

and

$$u_N, v_N = 0. \quad (4.15)$$

It is apparent from Equations (4.13) and (4.14) that t_{ij} and τ_{ij} are the expected travel times on link ij corresponding to the uncongested and congested cases, respectively.

If the link traversed to arrive at current node i is uncongested, then the function u_i must be evaluated to select the optimal successor node j . Otherwise, if the link is congested, the function v_i will be evaluated. The condition of link ij is ascertained once the traveler arrives at the successor node j . The problem to be solved at node j then can be formulated in the same fashion as for node i . The traveler's decision at node j is conditioned on the new knowledge the traveler acquires about the state of link ij . Thus, it is apparent that no optimal path can be entirely pre-determined.

The simplified formulations given by Equations (4.11) and (4.12) for the case of link-based congestion have the same form as node-based Equations (4.1) and (4.2). Our analysis and discussion in the following sections are all based, without loss of generality, on Equations (4.11) and (4.12).

4.3 Numerical Solution – Picard's Method of Successive Approximation

From a mathematical point of view, Equations (4.11) and (4.12) invite the following questions. First, does the set of equations have a solution? Second, if a solution exists, is it unique? And third, how can the equations be solved to evaluate the unknown variables u_1, u_2, \dots, u_N and v_1, v_2, \dots, v_N ?

Let us first consider the question of existence. Picard's method of successive approximation is one possible approach to solve the system of nonlinear Equations (4.11) and (4.12). Picard's method begins with initial approximations to the solution, and then refines these approximations by successive iterations. Let k be an iteration counter. Set $k = 0$. We begin with the simple initial approximations

$$u_i^0 = t_{iN}, \quad i = 1, 2, \dots, N-1, \quad (4.16)$$

$$u_N^0 = 0, \quad (4.17)$$

$$v_i^0 = \tau_{iN}, \quad i = 1, 2, \dots, N-1, \quad (4.18)$$

and

$$v_N^0 = 0. \quad (4.19)$$

These approximations are based on the expected travel times over direct links between nodes i and N . This first approximation may be quite poor, because most of the travel times identified in Equations (4.16) and (4.18) are necessarily infinite: Most nodes i have no direct link to node N .

To obtain u_i and v_i , the solutions u_j and v_j are needed. At the conclusion of the k^{th} iteration, the best estimates of u_j and v_j available are u_j^k and v_j^k . The iterative relationships

$$u_i^{k+1} = \min_{j \neq i} \{t_{ij} + \alpha_{ij}u_j^k + (1 - \alpha_{ij})v_j^k\}, i = 1, 2, \dots, N-1, \quad (4.20)$$

$$u_N^{k+1} = 0, \quad (4.21)$$

$$v_i^{k+1} = \min_{j \neq i} \{\tau_{ij} + \lambda_{ij}u_j^k + (1 - \lambda_{ij})v_j^k\}, i = 1, 2, \dots, N-1, \quad (4.22)$$

and

$$v_N^{k+1} = 0 \quad (4.23)$$

are used to improve the solution from the k^{th} to the $(k+1)^{\text{st}}$ approximation.

4.3.1 Proof that u_i^k and v_i^k Are Bounded Below

When $k = 0$,

$$u_i^0 = t_{iN} \geq 0, \quad (4.24)$$

and

$$v_i^0 = \tau_{iN} \geq 0. \quad (4.25)$$

If $u_i^k \geq 0$ and $v_i^k \geq 0$,

then

$$u_i^{k+1} = \min_{j \neq i} \{t_{ij} + \alpha_{ij}u_j^k + (1 - \alpha_{ij})v_j^k\} \geq 0, \quad (4.26)$$

and

$$v_i^{k+1} = \min_{j \neq i} \{\tau_{ij} + \lambda_{ij}u_j^k + (1 - \lambda_{ij})v_j^k\} \geq 0. \quad (4.27)$$

Inequalities (4.26) and (4.27) hold because $0 \leq \alpha_{ij} \leq 1$, $0 \leq \lambda_{ij} \leq 1$, $t_{ij} \geq 0$, and $\tau_{ij} \geq 0$.

4.3.2 Proof that u_i^k and v_i^k Are Monotone Decreasing in k

When $k = 0$, $u_i^0 = t_{iN}$, and $v_i^0 = \tau_{iN}$.

When $k = 1$,

$$u_i^1 = \min \left\{ \begin{array}{l} t_{i1} + \alpha_{i1}u_1^0 + (1 - \alpha_{i1})v_1^0; \\ t_{i2} + \alpha_{i2}u_2^0 + (1 - \alpha_{i2})v_2^0; \\ \vdots \\ t_{iN} + \alpha_{iN}u_N^0 + (1 - \alpha_{iN})v_N^0 \end{array} \right\} = \min \left\{ \begin{array}{l} t_{i1} + \alpha_{i1}u_1^0 + (1 - \alpha_{i1})v_1^0; \\ t_{i2} + \alpha_{i2}u_2^0 + (1 - \alpha_{i2})v_2^0; \\ \vdots \\ t_{iN} + 0 \end{array} \right\} \leq t_{iN} = u_i^0. \quad (4.28)$$

Similarly we have $v_i^1 \leq v_i^0$.

Next, assume that $u_i^k \leq u_i^{k-1}$ and $v_i^k \leq v_i^{k-1}$. Then we have

$$u_i^{k+1} = \min \{ t_{ij} + \alpha_{ij}u_j^k + (1 - \alpha_{ij})v_j^k \} \leq \min \{ t_{ij} + \alpha_{ij}u_j^{k-1} + (1 - \alpha_{ij})v_j^{k-1} \} = u_i^k. \quad (4.29)$$

In the same way, we have that $v_i^{k+1} \leq v_i^k$. This is an inductive proof that u_i^k and v_i^k are monotone decreasing in k in addition to being bounded below by 0. Thus we have

$$u_i^k \geq u_i^{k+1} \geq 0, \quad (4.30)$$

and

$$v_i^k \geq v_i^{k+1} \geq 0. \quad (4.31)$$

4.3.3 Proof that u_i and v_i Are Determined in a Finite Number of Iterations

Because of Inequalities (4.30) and (4.31), and based on the principle of bounded monotone convergence, the sequence of iterations will necessarily converge to a single limit. Thus, we have

$$\lim_{k \rightarrow \infty} u_i^k = u_i, \quad (4.32)$$

and

$$\lim_{k \rightarrow \infty} v_i^k = v_i. \quad (4.33)$$

In the worst case, we might need to do an infinite number of iterations to converge to the true solution. However, in this problem we can see that the values of u_i^k and v_i^k have physical meanings. In fact, u_i^k is the minimum expected time needed to go from node i to node N over a path with at most k intermediate nodes, given that the link traversed to reach node i is uncongested. Similarly, v_i^k is the minimum expected time needed to go from node i to node N over a path with at most k intermediate nodes, given that the link traversed to reach node i is congested. There are N nodes in the network, two of which consist of an origin and the destination. Therefore, an optimal path can have no more than $N-2$ intermediate nodes. This means the actual number of iterations cannot exceed $N-2$.

Thus we have shown that a solution to Equations (4.11) and (4.12) can be obtained through a finite number of iterations. Therefore, this system of equations does have a solution. Furthermore, we have also shown how the equations can be

solved iteratively. Thus, the first (the existence of a solution) and third (a procedure for obtaining a solution) of our questions have been answered.

We now proceed to the question of the uniqueness of the solution to the system of Equations (4.11) and (4.12). Assume there are two alternative solutions. Call the first u_i and v_i and the second U_i and V_i , where $i = 1, 2, \dots, N$. If we show that it is necessarily true that $u_i = U_i$ and $v_i = V_i$, then the solution is unique. Consider the first solution. Assume the correct successor is r if the link traversed to reach node i is uncongested, and s if the link is congested. Thus

$$u_i = t_{ir} + \alpha_{ir}u_r + (1 - \alpha_{ir})v_r, \quad i = 1, 2, \dots, N-1, \quad (4.34)$$

and

$$v_i = \tau_{is} + \lambda_{is}u_s + (1 - \lambda_{is})v_s. \quad (4.35)$$

Consider the second solution. Assume the correct successor is l if the link traversed to reach node i is uncongested, and m if the link is congested. Thus

$$U_i = t_{il} + \alpha_{il}U_l + (1 - \alpha_{il})V_l, \quad (4.36)$$

and

$$V_i = \tau_{im} + \lambda_{im}U_m + (1 - \lambda_{im})V_m. \quad (4.37)$$

Because successor node r is optimal for u_i ,

$$u_i = t_{ir} + \alpha_{ir}u_r + (1 - \alpha_{ir})v_r \leq t_{il} + \alpha_{il}u_l + (1 - \alpha_{il})v_l. \quad (4.38)$$

Subtracting Equation (36) from Expression (38), and given that α_j is a probability,

$$u_i - U_i \leq \alpha_{il}(u_l - U_l) + (1 - \alpha_{il})(v_l - V_l) \leq \max(u_l - U_l, v_l - V_l), \quad (4.39)$$

where $l \neq i$. Recall that that node l is associated with the solution U_i .

Consider the two solutions at node l . Given the first solution, u_l and v_l , the correct successor node is node p if the link traversed to reach node l is uncongested, and q if the link is congested. Thus, we have

$$u_l = t_{lp} + \alpha_{lp} u_p + (1 - \alpha_{lp}) v_p, \quad (4.40)$$

and

$$v_l = \tau_{lq} + \lambda_{lq} u_q + (1 - \lambda_{lq}) v_q. \quad (4.41)$$

Given the second solution, U_l and V_l , the correct successor node to visit is node x if the link traversed to reach node l is uncongested, and node y if the link is congested. Thus, we have

$$U_l = t_{lx} + \alpha_{lx} U_x + (1 - \alpha_{lx}) V_x, \quad (4.42)$$

and

$$V_l = \tau_{ly} + \lambda_{ly} U_y + (1 - \lambda_{ly}) V_y. \quad (4.43)$$

As in the manner of Expressions (4.38) and (4.39), this gives

$$u_l - U_l \leq \max(u_x - U_x, v_x - V_x), \quad (4.44)$$

where $x \neq i$ and $x \neq l$, since nodes i and l already have been visited. Continuing in this manner, it would never be efficient to revisit a given node. Consequently, we must eventually reach the node N , where

$$u_N - U_N = 0, \quad (4.45)$$

and

$$v_N - V_N = 0 \quad (4.46)$$

Taken together, Inequality (4.44) and Equations (4.45) and (4.46) give the result

$$u_i - U_i \leq 0, i = 1, 2, \dots, N. \quad (4.47)$$

To complete the proof, we also want to show that

$$U_i - u_i \leq 0, i = 1, 2, \dots, N. \quad (4.48)$$

Refer to Equations (4.34) and (4.36). Since the optimal successor node for U_i is node l ,

$$U_i \leq t_{ir} + \alpha_{ir} U_r + (1 - \alpha_{ir}) V_r. \quad (4.49)$$

Given that α_{ir} is a probability, subtracting Equation (4.34) from Expression (4.49) gives

$$U_i - u_i \leq \alpha_{ir} (U_r - u_r) + (1 - \alpha_{ir}) (V_r - v_r) \leq \max(U_r - u_r, V_r - v_r), \quad (4.50)$$

where $r \neq i$. Recall that node r is associated with the solution u_i .

Applying the previous procedure, it follows that Expression (4.48) holds. Taken together, Expressions (4.47) and (4.48) require that

$$u_i - U_i = 0, i = 1, 2, \dots, N. \quad (4.51)$$

By the same argument,

$$v_i - V_i = 0, i = 1, 2, \dots, N. \quad (4.52)$$

The existence of two different solutions to the system of Equations (4.11) and (4.12) is a contradiction. Consequently, there is one and only one solution to this system of equations.

4.4 Numerical Examples: Applications to Small Networks

4.4.1 Test 1: Application to a Four-Node Network

A four-node network appears in Figure 4-2. The results for this very simple example can be checked manually. Set the probability that link ij is uncongested if the link traversed to arrive at node i is uncongested to be

$$\alpha_{ij} = 2/3, i, j = 1, 2, \dots, N; i \neq j. \quad (4.53)$$

Set the probability that link ij is uncongested if the link traversed to arrive at node i is congested to be

$$\lambda_{ij} = 1/3, i, j = 1, 2, \dots, N; i \neq j. \quad (4.54)$$

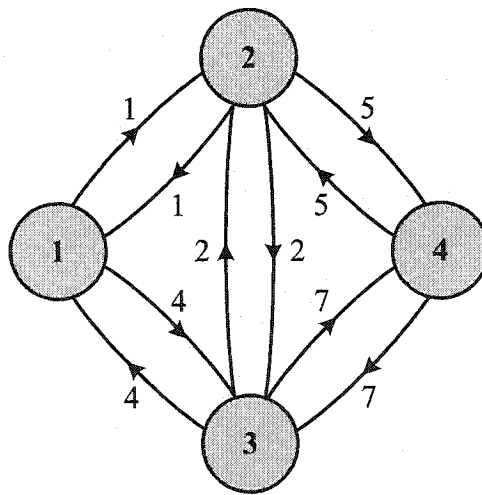


Figure 4-2: A Four-Node Network

Arc weights in Figure 4-2 consist of elements of the matrix $[t_{ij}]$, the set of expected travel times for each link under uncongested conditions. The weights are

symmetric in this example, but need not be. The matrix of expected link travel times under congested conditions is defined as follows:

$$[\tau_{ij}] = 2[t_{ij}]. \quad (4.55)$$

Set the destination node to node 4. A simple MATLAB program for obtaining these results is given in Appendix I. This program can easily be generalized to larger examples. The lowest expected travel times to node 4 and the optimal choice of successor node for origin nodes 1 – 3 are given in Table 4-1. We assume that a link is traversed to reach any origin node, and thus the conditional probabilities α_{ij} and λ_{ij} are always defined. If no link was traversed to arrive at the origin, then these conditional probabilities might practically be replaced with corresponding unconditional values.

The entries in Table 4-1 demonstrate that the optimal strategies can change based on link level of service information. In the case of trips originating from node 3, the optimal successor node changes from node 4 to node 2 if the level of service on the link traversed to reach node 3 is congested rather than not congested. Computational results are unchanged if the initial node labels are redefined, which serves as a check on the method and the MATLAB program.

4.4.2 Test 2: Application to a Nine-Node Network

A nine-node network appears in Figure 4-3. The probabilities α_{ij} and λ_{ij} are as before, as is the relationship between the matrices $[t_{ij}]$ and $[\tau_{ij}]$. In this case, the travel time matrix is not symmetric.

Table 4-1: Minimum Expected Travel Time from Each Origin Node to Destination Node 4, and Optimal Choice of Successor Nodes

Origin	Minimum Expected Travel Time to Destination Node 4	Optimal Successor Node
Node 1 previous link is uncongested	7.6667	2
Node 1 previous link is congested	10.3333	2
Node 2 previous link is uncongested	5.0000	4
Node 2 previous link is congested	10.0000	4
Node 3 previous link is uncongested	7.0000	4
Node 3 previous link is congested	12.3333	2

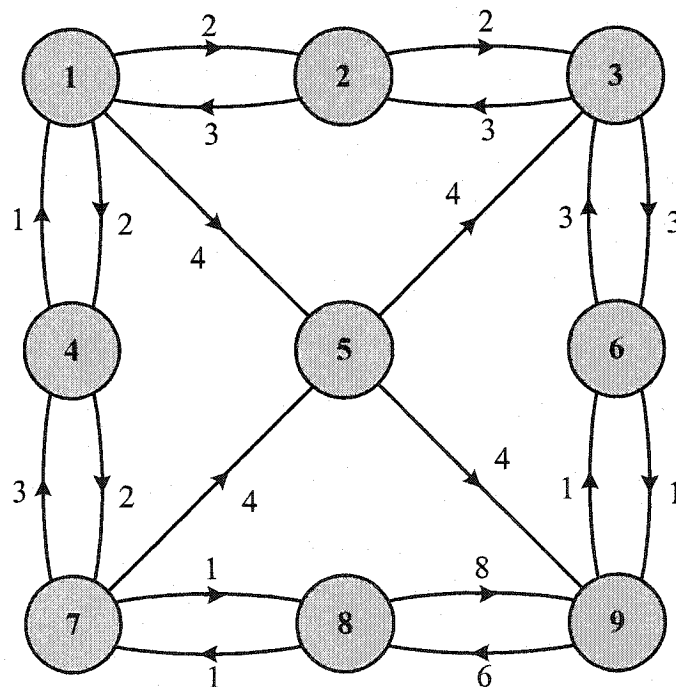


Figure 4-3: A Nine-node Network

Set the destination node to be node 9. The lowest expected travel times to node 9 and the optimal choice of successor node for origin nodes 1 – 8 are given in Table 4-2. As observed in the first example, the optimal strategy for proceeding

from each node changes with the level of service experienced on the link traversed to reach the node. See the results for nodes 1, 7, and 8. The network has at most seven intermediate nodes in a path.

Table 4-2: Minimum Expected Travel Time from Each Origin Node to Destination Node 9, and Optimal Choice of Successor Nodes

Origin	Minimum Expected Travel Time to Destination Node 9	Optimal Successor Node
Node 1 previous link is uncongested	9.3333	5
Node 1 previous link is congested	13.5185	2
Node 2 previous link is uncongested	7.4444	3
Node 2 previous link is congested	10.5556	3
Node 3 previous link is uncongested	4.3333	6
Node 3 previous link is congested	7.6667	6
Node 4 previous link is uncongested	11.7284	1
Node 4 previous link is congested	14.1235	1
Node 5 previous link is uncongested	4.0000	9
Node 5 previous link is congested	8.0000	9
Node 6 previous link is uncongested	1.0000	9
Node 6 previous link is congested	2.0000	9
Node 7 previous link is uncongested	9.3333	5
Node 7 previous link is congested	14.5450	8
Node 8 previous link is uncongested	8.0000	9
Node 8 previous link is congested	14.8078	7

The results remain unchanged after the fifth iteration of the successive approximation scheme. The intermediate results of the successive approximation are available in Appendix J. Thus, numerical experience is consistent with the proof of the convergence. As before, computational results are unchanged if the initial node labels are redefined.

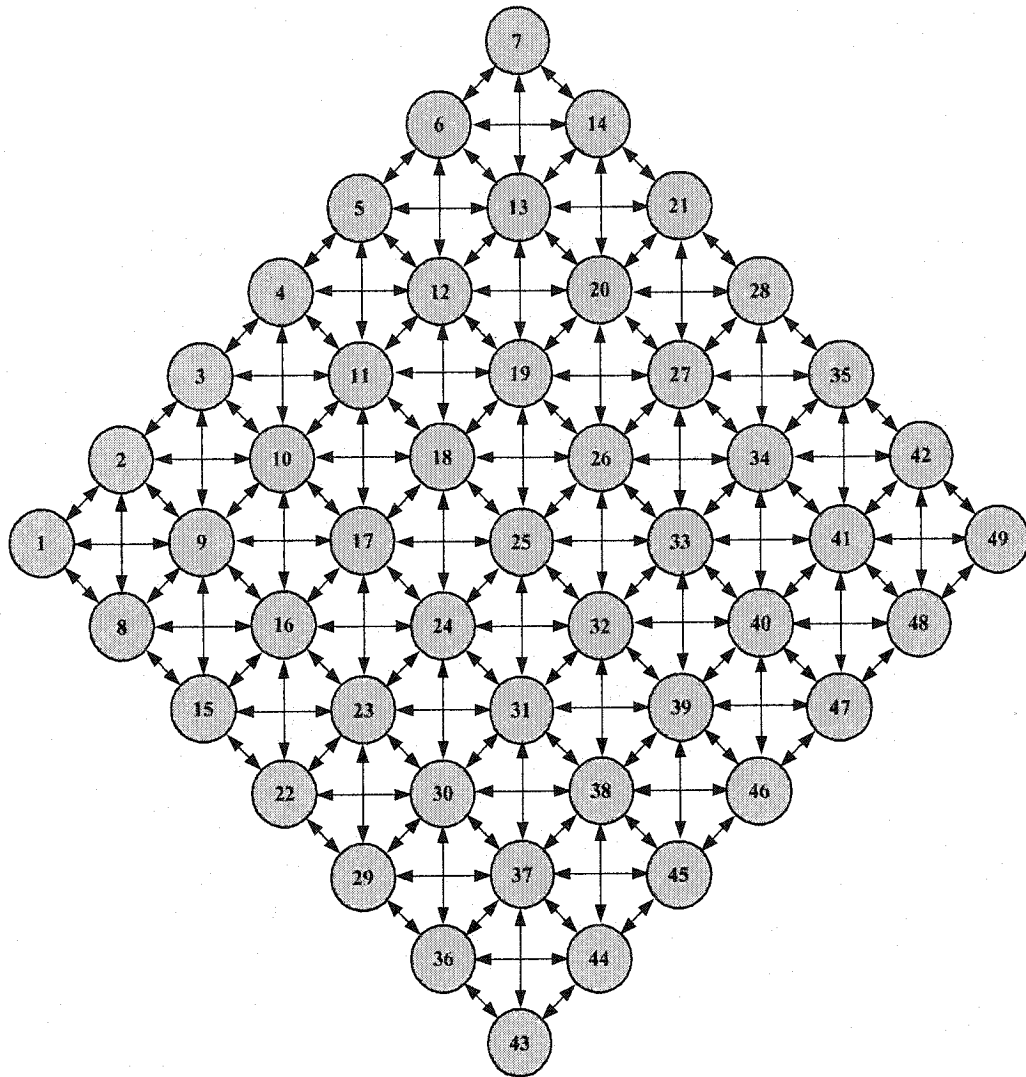


Figure 4-4: A 49-node Network

4.4.3 Test 3: Application to a 49-Node Network

The previous examples offer the advantage of results that can be replicated by hand, or are otherwise transparent enough to allow convenient

verification. A 49-node network appears in Figure 4-4. Average, travel times for uncongested links are given in Table 4-3. As in the previous examples, average congested travel times are taken to be twice the corresponding values for uncongested flows. The minimum expected travel times to node 49 and the optimal choice of successor nodes for origin nodes 1 – 48 are given in Table 4-4. Links to optimal successor nodes are provided in Figure 4-5. In some cases, congestion on the link traversed to reach the current node has no impact on the definition of the optimal successor node, though it always has an impact on the estimate of the minimum expected time to reach the destination node. For those cases in which traversing a congested link to arrive at the current node conditions a different choice for the successor node than would traversing an uncongested link, the link to the alternative successor node is shown in gray rather than black. The results summarized in Figure 4-5 are counter-intuitive from a deterministic perspective. Indeed, they are suboptimal from a deterministic perspective. However, this is not a deterministic problem. Any node in the network may be defined as an origin. As the minimum number of links that must be traversed to reach the destination node from an origin increases, the expected travel time to the destination generally (but not inevitably) increases. Consider the cut set defined in Figure 4-6. The minimum

five nodes is less than the minimum number of links that must be traversed on trips originating inside the cut set. This implies in neither the deterministic nor the stochastic case that the minimum expected travel times from nodes 3, 9, 10, 15, and 16 must all be less than the minimum expected travel time between any node in the cut set and node 49. Compare nodes 1 and 3, for example. However, optimality requires that the expected minimum travel time from at least one of these five nodes must be less than the corresponding minimum expected travel time for each node in the cut set. The values presented in Table 4-4 conform to these requirements for this cut and for all other cuts on the network.

Table 4-3: Average Uncongested Travel Times for Links in the 49-Node**Network**

Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
1	.	11	3	7
2	6	.	10	9	6	1
3	.	9	.	5	7	9	10
4	.	.	8	.	3	5	10	10
5	.	.	.	5	.	10	2	5	9
6	1	.	2	3	3	7
7	4	3	1
8	8	5	10	6	5
9	9	6	3	8	.	9	1	8	5
10	.	9	6	8	5	.	4	3	3	8
11	.	.	4	6	3	8	.	5	10	10	7
12	.	.	.	6	10	9	7	.	9	8	4	4
13	4	6	8	4	.	9	7	5	8
14	6	5	8	7	9
15	11	6	10	3	11	.	.
16	4	4	10	8	.	2	1	10	3	.
17	4	8	4	6	.	2	11	7	.
18	6	4	5	3	.	7	9
19	3	5	9	8	.	6
20	7	2	5	4	.	10
21	11	9	5
22	7	4	11	.	.
23	8	4	5	10	.	8	.
24	2	3	7	7	.	.
25	1	4	1	5
26	7	1	1
27	8	8	2
28	8	8
29	6	8	.	.
30	10	3	9	.
31	7	3	.
32	4	.
33
34
35
36
37
38
39
40
41

Table 4-3: Average Uncongested Travel Times for Links in the 49-Node**Network (Continued)**

34	.	10	1	2	9	.	5	10	8	8
35	.	.	4	3	3	3	5
36	10	6	9	6	6
37	6	5	10	1	.	4	1	8	8
38	11	7	5	3	.	7	.	.	.	8	5	1
39	5	9	9	10	.	9	.	.	.	5	7	8	.	.	.
40	3	10	7	7	.	3	6	10	4	.
41	8	5	7	8	.	5	5	8	9
42	5	5	7	7	8
43	6	9	6
44	3	8	11	.	.	.	9	.	8
45	6	2	8	.	.	.	5	.	2
46	7	9	8	11	.	11	.	.
47	2	5	6	4	.	7	.
48	1	7	8	11	.	9
49	8	5	7

Table 4-4: Minimum Expected Travel Time from Each Origin Node to Destination Node 49, and Optimal Choice of Successor Nodes

Origin	Minimal Expected Travel Time to Destination Node 49	Optimal Successor Node
Node 1 previous link is uncongested	43.1	8
Node 1 previous link is congested	48.2	8
Node 2 previous link is uncongested	38.4	10
Node 2 previous link is congested	40.9	10
Node 3 previous link is uncongested	46.4	10
Node 3 previous link is congested	56.7	4
Node 4 previous link is uncongested	42.4	10
Node 4 previous link is congested	48.9	10
Node 5 previous link is uncongested	39.8	12
Node 5 previous link is congested	45.5	11
Node 6 previous link is uncongested	37.8	12
Node 6 previous link is congested	42.5	12
Node 7 previous link is uncongested	38.3	14
Node 7 previous link is congested	42.7	14
Node 8 previous link is uncongested	38.0	16
Node 8 previous link is congested	44.3	16
Node 9 previous link is uncongested	40.7	17
Node 9 previous link is congested	47.0	15
Node 10 previous link is uncongested	36.0	16
Node 10 previous link is congested	40.3	16
Node 11 previous link is uncongested	36.0	19
Node 11 previous link is congested	44.3	19
Node 12 previous link is uncongested	33.0	19
Node 12 previous link is congested	38.3	19
Node 13 previous link is uncongested	32.8	21
Node 13 previous link is congested	42.1	20
Node 14 previous link is uncongested	33.8	21
Node 14 previous link is congested	44.2	21

Table 4-4: Minimum Expected Travel Time from Each Origin Node to Destination Node 49, and Optimal Choice of Successor Nodes (Continued)

Node 15 previous link is uncongested	41.8	22
Node 15 previous link is congested	46.6	22
Node 16 previous link is uncongested	31.7	24
Node 16 previous link is congested	35.6	24
Node 17 previous link is uncongested	33.7	25
Node 17 previous link is congested	39.6	25
Node 18 previous link is uncongested	34.7	25
Node 18 previous link is congested	41.6	25
Node 19 previous link is uncongested	27.8	27
Node 19 previous link is congested	31.6	27
Node 20 previous link is uncongested	30.4	26
Node 20 previous link is congested	32.9	26
Node 21 previous link is uncongested	23.3	28
Node 21 previous link is congested	27.7	28
Node 22 previous link is uncongested	37.0	16
Node 22 previous link is congested	42.3	16
Node 23 previous link is uncongested	36.7	24
Node 23 previous link is congested	42.3	16
Node 24 previous link is uncongested	27.8	32
Node 24 previous link is congested	30.5	32
Node 25 previous link is uncongested	27.8	32
Node 25 previous link is congested	30.5	32
Node 26 previous link is uncongested	28.0	34
Node 26 previous link is congested	32.3	19
Node 27 previous link is uncongested	24.0	34
Node 27 previous link is congested	29.3	34
Node 28 previous link is uncongested	19.0	35
Node 28 previous link is congested	23.0	35
Node 29 previous link is uncongested	40.6	37
Node 29 previous link is congested	46.4	37
Node 30 previous link is uncongested	32.8	38

Table 4-4: Minimum Expected Travel Time from Each Origin Node to Destination Node 49, and Optimal Choice of Successor Nodes (Continued)

Node 30 previous link is congested	36.2	38
Node 31 previous link is uncongested	29.8	32
Node 31 previous link is congested	34.5	32
Node 32 previous link is uncongested	25.1	33
Node 32 previous link is congested	30.2	33
Node 33 previous link is uncongested	20.0	41
Node 33 previous link is congested	29.3	34
Node 34 previous link is uncongested	18.7	42
Node 34 previous link is congested	25.7	28
Node 35 previous link is uncongested	15.0	41
Node 35 previous link is congested	21.0	41
Node 36 previous link is uncongested	39.9	30
Node 36 previous link is congested	47.1	30
Node 37 previous link is uncongested	34.8	38
Node 37 previous link is congested	40.2	38
Node 38 previous link is uncongested	29.3	46
Node 38 previous link is congested	33.7	46
Node 39 previous link is uncongested	26.0	40
Node 39 previous link is congested	37.0	40
Node 40 previous link is uncongested	15.0	41
Node 40 previous link is congested	21.0	41
Node 41 previous link is uncongested	9.0	49
Node 41 previous link is congested	18.0	49
Node 42 previous link is uncongested	8.0	49
Node 42 previous link is congested	16.0	49
Node 43 previous link is uncongested	45.6	37
Node 43 previous link is congested	56.4	37
Node 44 previous link is uncongested	40.1	45
Node 44 previous link is congested	49.9	45
Node 45 previous link is uncongested	30.3	46
Node 45 previous link is congested	35.7	46

**Table 4-4: Minimum Expected Travel Time from Each Origin Node to
Destination Node 49, and Optimal Choice of Successor Nodes (Continued)**

Node 46 previous link is uncongested	25.0	40
Node 46 previous link is congested	35.0	40
Node 47 previous link is uncongested	18.0	41
Node 47 previous link is congested	27.0	41
Node 48 previous link is uncongested	9.0	49
Node 48 previous link is congested	18.0	49

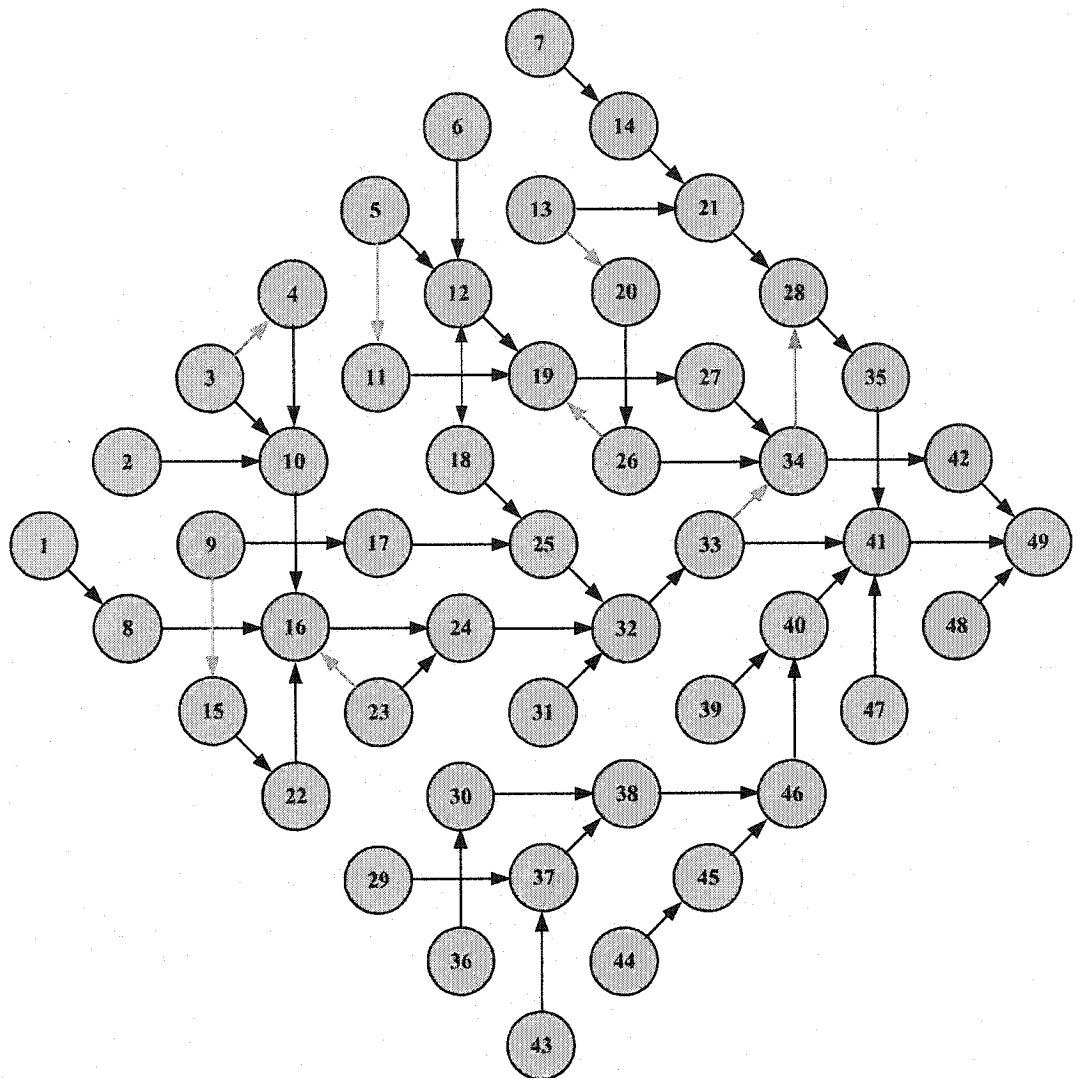


Figure 4-5: Optimal Successor Nodes in the 49-node Network

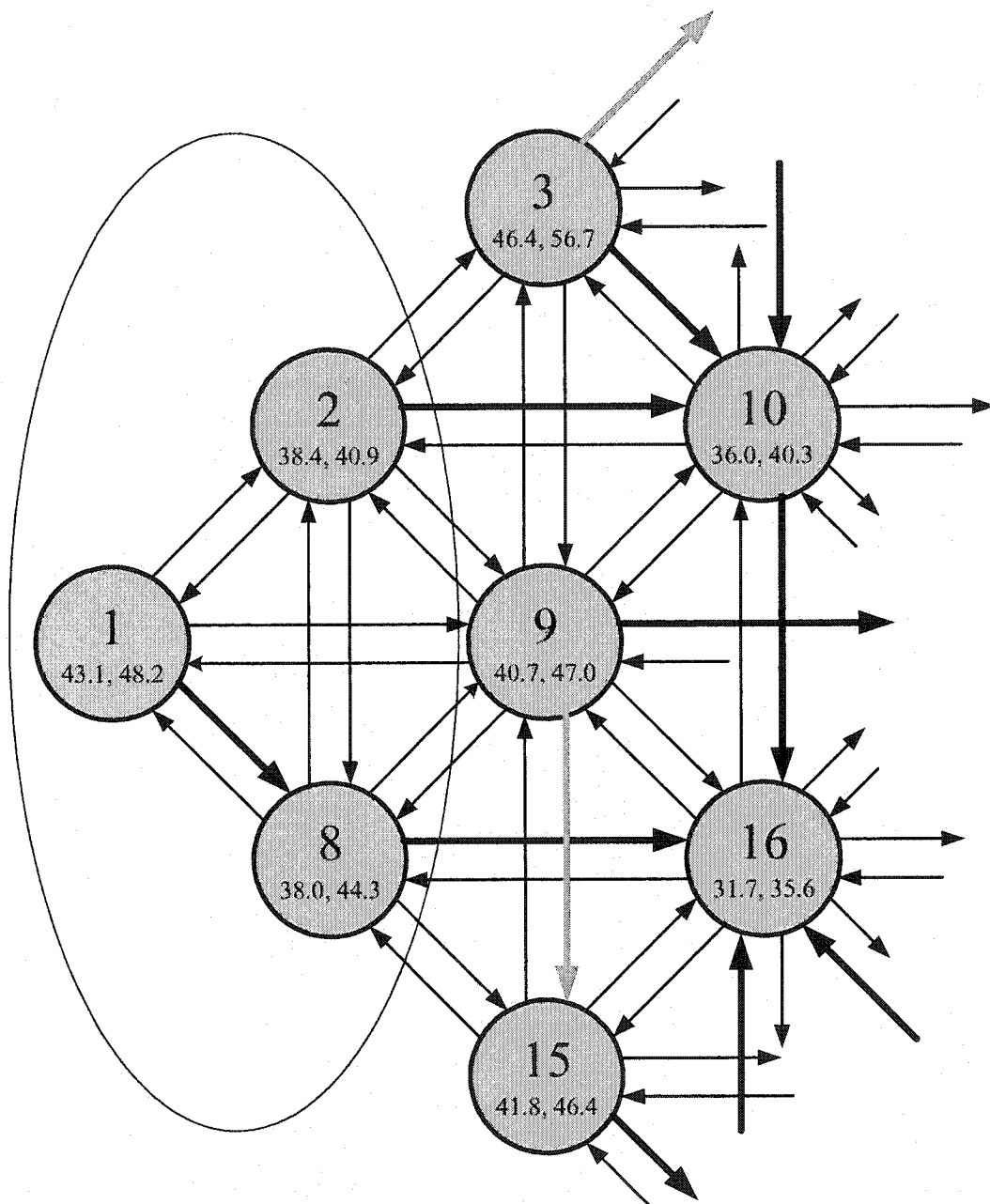


Figure 4-6: A Cut Across the Links to the Left of Nodes 3, 9, 10, 15 and 16

4.5 Summary

Network problems that include correlated link travel times have not been well studied. In this research, the case of a stochastic network routing problem with correlated link travel time is formulated using Bellman's principle of optimality, and solved via Picard's method of successive approximation. The formulation of this problem defines a new set of equations. The existence, uniqueness, and convergence of the approximations to the solution of these new equations are proved.

The major contribution of this research is defining and formulating the problem, and mathematically proving some of the properties of the approximation scheme. Our limited numerical experience demonstrates that the procedure can be implemented simply. Our dynamic programming approach to this problem ensures that the computation burden associated with solving this problem is low. However, more numerical experiments with larger networks and a less aggregated representation of level of service states are needed, as are field observations of the probabilities α and λ .

Only two possible level-of-service states are considered in these examples, but increasing the number of states changes neither the nature of the formulation nor the procedure for numerical solution. While the techniques sketched in this paper can be applied to other stochastic network problems, they lead to many questions for the future.

CHAPTER 5 CONCLUSIONS AND DISCUSSION

Routing problems have broad applications in transportation engineering, computer science, operations research, and neurophysiology. They are of importance for passenger and goods movement, message delivery, and more general system control. Our extensive literature review has brought two points to our attention. First, most studies in routing problems focus on minimum expected cost. Problems of maximum reliability or likelihood have not been well studied. Second, arc weights are assumed to be independent random variables in all most all the studies treating stochastic network problems. The two stochastic routing problems included in this document are designed to address these two missing points in the literature. We will summarize and discuss these two routing problems separately in this chapter.

The stochastic on time arrival problem (SOTA) is to determine the optimal strategy from each node i leads to the maximum probability of arriving at the destination node N within time t or earlier, in a network where the link travel times are given by independent random variables with probability density functions that are known a priori.

The problem is formulated using Bellman's principle of optimality. The formulation results in a set of nonlinear convolution integral equations. The solution to these equations is approximated through Picard's method of successive approximation. To our knowledge, this is the first time such problems are treated with provision of feasible numerical implementations in the transportation literature.

This class of problems considering probability or likelihood usually requires the probability density functions to be modeled explicitly. Our experience has shown that even if a formulation exists for such problem, obtaining numerical solutions can be difficult. Several applied mathematical techniques are applied to improve the computational efficiency of our numerical procedure. For example, relying on the convolution theorem of Laplace transforms reduces the computational cost associated with evaluating convolution integrals. Linear algebraic equations are solved in a least square sense by applying generalized inverses. The multiplicative property of Laplace transforms makes it possible to change the time scale of the problems to be solved and so to evaluate the unknown functions at more points.

Numerical examples and validation tests indicate that this numerical approach of solving the SOTA problems is efficient and reliable. The recommended routing strategies from the SOTA procedure, where the goal is to maximize the probability of arriving on time, is compared with that from the conventional shortest path procedure, where the goal is to arrive at the destination as soon as possible. In general, the strategies recommended by these two procedures tend to agree with one another if the variances of the link travel times are small. In some cases, the optimal successor nodes recommended by the SOTA procedure when the remaining time for on time arrival is relatively bigger differ from those recommended by the shortest path procedure. In most of the cases in our numerical examples, the optimal successor nodes from a given starting node change no more than once as the remaining time for on time arrival diminishes. There are a few exceptions observed

in the first 100-node network. The detailed discussion of the exceptional cases is given in Section 3.4. As observed in the numerical examples, the optimal strategies may change when the remaining time is relatively larger, and the SOTA strategy tends to agree with the shortest path strategy if the variances are small. However, how small variances are small enough for the two strategies to agree with each other completely? How large a remaining time is large relative to the minimum cost of the shortest path? These quantitative measures remain unclear.

Some remaining methodological questions of evaluating convolution integrals are discussed in detail in Chapter 3. They are briefly summarized as below. First, the integer values s at which the Laplace transforms are evaluated were selected arbitrarily. How to best select these s values to improve the quality of the numerical scheme of evaluating convolution integrals is unclear. Second, the relationship between the number of quadrature points and the quality of results of approximating convolution integrals via a finite sum needs to be studied further. Third, the instability problem seems inevitable during the procedure of numerical inversion of Laplace transform when the functions involved are not smooth. Based on our experience, a simple uniform distribution over a period of time may cause instability of inversion of Laplace transform due to the sudden change at the two extreme ends. This instability problem did not occur during the scope of this research. However, it can be the most vulnerable part of the procedure of evaluating convolution integrals via Laplace transform under some circumstances. Therefore, validation tests are necessary before this procedure can be applied to a new situation.

The applied techniques applied in this research, such as Picard's method of successive approximation and Laplace transform and its inversion, are proved to be feasible and efficient. However, there may be other methods available. In earlier chapter, we have shown that convolution integrals can be evaluated through the method of Laplace transform and the method of differential approximations. Based on our experiment, it seems that the method of Laplace transform requires less computing resources, while the method of differential approximation evaluates the unknown function at more flexible time points and provides more reliable solution. Implementation of our current version of differential approximation method to the SOTA problem seems inapplicable due to the huge computational resources it requires. However, there are ways to improve our procedure of solving differential equations. Besides, other methods of solving convolution integral equations may be available. In our future research, we shall seek for other approaches that may offer advantages.

We evaluate the unknown functions $u_1(t)$, $u_2(t)$, ..., $u_N(t)$ describing the maximum probabilities of arriving on time and the optimal choice of the next node to visit for discrete points in time. The uniqueness and continuity of these functions have not been proved mathematically here. Computing can be used as an experimental tool to explore the hidden nature of general solutions. Our extensive numerical experiments suggest that these functions should be unique and continuous, though perhaps not everywhere differentiable. The value of these functions for other

times t , therefore, can likely be approximated via interpolation. However, it is not clear how to identify successor nodes for intermediate values of t .

Finally, the stochastic on-time arrival problem has been formulated based on the assumption that the travel times on any two links are independent. Further research is needed to accommodate the possibility of correlated travel times on adjacent links.

Link costs are assumed to be independent random variables in most studies on network problems. Network problems that include correlated link travel times have not been well studied. The second optimal routing problem in this document addresses the shortest path problem in congestible networks with correlated link travel costs. The objective is to minimize expected travel time. In this problem, each link is assumed to be in one of two possible states, congested or un-congested. Conditional probability density functions for link travel times are assumed known for each state. The traveler takes into account his experience on the link leading to the current decision point (node) when determining which node to visit next, i.e., which link to traverse next. Only the next immediate link in the path is identified, and the knowledge gained during the course of the trip is used to optimally support the decisions defining the remaining journey.

This problem is formulated using Bellman's principle of optimality, and solved via Picard's method of successive approximation. The formulation of this problem defines a new set of equations. The existence, uniqueness, and convergence of the solution to these new equations are proved. The major contribution of this

research is defining and formulating the problem, and mathematically proving the nature of the solutions.

Our limited numerical experience demonstrates that the procedure can be implemented simply. Our dynamic programming approach to this problem ensures that the computation burden associated with solving this problem is low. However, more numerical experiments with larger networks and a less aggregated representation of level of service states are needed, as are field observations of the probabilities α and λ .

Only two possible level-of-service states are considered in these examples, but increasing the number of states changes neither the nature of the formulation nor the procedure for numerical solution.

Both routing problems involve multi-stage decisions. We have shown earlier how a specific problem can be imbedded to a class of such problems and therefore can be formulated using Bellman's principle of optimality. The two studies in this document, coupled with the formulations of deterministic and stochastic shortest path problems given in Chapter 2, demonstrate the significance of Bellman's principle of optimality in multi-decision process.

The two routing studies in this document provide optimal strategies for routing through stochastic networks. These strategies are dynamically determined, in the sense that only the next immediate successor node is sought for at a certain decision point. Therefore, the knowledge accumulated through the trip can be taken into account for the remaining journey. The decision support provided by this study

has a number of applications. Both centralized and vehicle-based route guidance applications in Advanced Traveler Management Information Systems (ATMIS) require the means to account for probabilistic link travel times, and to update results based on telemetric field data. Emergency responses to natural disasters must often be made under conditions of uncertainty. Vehicle routing in a transportation network damaged by an earthquake or a hurricane requires a mechanism for coping with random loss of network capacity. This is especially true if telemetric data systems are normally relied on to assess the state of the network. A natural disaster sufficient to reduce the traffic carrying capacity of the network is also likely to disrupt traffic data collection systems. We have focused our research in the context of transportation networks. However, this research can be extended to more general applications in system controls. We may interpret nodes as the possible states of a system and the links as transformations from one state to another. This research can therefore be used to determine a strategy for transforming a system from a given initial state to a desired terminal state in an optimal manner.

While the techniques adumbrated in this research can be applied to other stochastic network problems, they lead to many questions for the future.

BIBLIOGRAPHY

Anderson, Robert, Hoog, Frank, and Lukas, Mark, *The Application and Numerical Solution of Integral Equations*, SIJTHOFF & NOORDHOFF, 1980

Bellman, R. E., *Dynamic Programming*, Princeton University Press, Princeton, N. J., 1957.

Bellman, R. E., On a routing problem. *Quart. Appl. Math.* 16, 1958, 87-90.

Bellman, R. E. and Kalaba, R., Differential approximation applied to the solution of convolution equations, *Mathematics of Computation*, 1964, 487-491.

Bellman, R. and Kalaba, R., *Dynamic Programming and Modern Control Theory*, McGraw-Hill, N.Y., 1965.

Bellman, R. E. and Kalaba, R., *Numerical Inversion of the Laplace Transform*, American Elsevier Publishing Company, Inc., New York, 1966.

Burton, D., On the inverse shortest path problem, Ph.D dissertation, Facultes University of Notre-Dame De La Paix De Namur, 1993.

Burton, D., On the use of an inverse shortest paths problem for recovering linearly correlated cost, *Mathematical Programming*, 1994, 63, 1-22.

C. Lubich, Convolution quadrature and discretized operational calculus. I", *Numer. Math.*, 52, 129-145, 1988.

C. Lubich, Convolution quadrature and discretized operational calculus. II", *Numer. Math.*, 52, 413-425, 1988.

Dantzig, G. B., and Wolf, Decomposition principle for linear programs, *Operations Research*, 8, No. 1, 1960.

Dial, R. B., Algorithm 360: shortest path forest with topological ordering, *Comm. ACM*, 12, 1969, 632-633.

Dijkstra, E. W., A Note on Two Problems on Connexion with Graphs. *Numer. Math.* 1, 1959, 269-271.

Dreyfus, S. E., An Appraisal of Some Shortest-Path Algorithm. *Operations Research* 17, 1969, 395-412.

- Eiger, Mirchandani, and Soroush, Path Preferences and Optimal Paths in Probabilistic Networks. *Transportation Science*, 19 (1), 1985, 75-84
- Fan, Y., Kalaba R., Dynamic Programming and Pseudo-inverses, *Applied Mathematics and Computation*, in press.
- Frank, H., Shortest paths in probabilistic graphs, *Operations Research*, 17, 1969, 583-599.
- Ford, L. R. Jr. and Fulkerson, D. R., Maximal flow through a network, *Canadian Journal of Math.*, 8, 1956, 399-404.
- Fu, L. and Rilett, L. R., Expected shortest paths in dynamic and stochastic traffic networks, *Transportation Research, part B*, 32 (7), 1998, 499-516
- Fu, L., An adaptive routing algorithm for in-vehicle route guidance systems with real-time information, *Transportation Research, Part B*, 35, 2001, 749-765.
- Hall, R. W., The fastest path through a network with random time-dependent travel times. *Transportation Science*, 20 (3), 1986, 182-188.
- Howard, R. A., *Dynamic Probabilistic Systems, Vol. II*. John Wiley and Sons, New York, 1971.
- Kaufman, D. E., Fastest paths in time-dependent networks for intelligent vehicle highway systems application. *IVHS Journal*, 1 (1), 1993, 1-11.
- Kalaba, R., On some communication network problems, *Combinatorial Anal Proc. Symp appl. Math*, 10, 1960, 261-280.
- Loui, P., Optimal paths in graphs with stochastic or multidimensional weights. *Communications of the ACM* 26 (1983), 670-676.
- Loui, P., Optimal paths in graphs with stochastic or multidimensional weights. *Communications of the ACM* 26 (1983), 670-676.
- Miller-Hooks, E. D. and Mahmassani, H. S., "Least Expected Time Paths in Stochastic, Time-Varying Transportation Networks", *Transportation Science*, 34 (2), 2002, 198-215.
- Minieka, E., On computing sets of shortest paths in a graph, *Comm. ACM*, 17, 1974, 351-353.
- Minieka, E. and Shier, D. R., A note on an algebra for the k best routes in a network, *J. Inst. Math. Appl.* 11, 1973, 145-149.

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

Murthy, I. and Sarkar, S., Stochastic shortest path problems with piecewise linear concave utility functions, *Management Science*, 44(11), Part 2 of 2, 1998, 125-136.

Pallack, M. and Wiebenson, W., Solutions of the shortest-route problem – a review, *Operations Research*, 8, 1960, 224-230.

Pallack, M. and Wiebenson, W., Comments on “The shortest path problem” by Peart, Randolph, and Bartlett, *Operations Research*, 8, 1960, 224-230.

Shier, D.R., Computational experience with algorithm for finding the k shortest paths in a network, J. Res. Nat. Bur. Standards, Sect. B, 78B, 1974, 139-165.

Shier, D.R., On algorithms for finding the k shortest paths in a network, *Networks*, 9, 1979, 195-214.

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

Srivastava and Buschman, *Convolution Integral Equations*, Wiley Eastern Limited, New Delhi, 1977

Srivastava, H. M. and Buschman, R. G., *Theory and Applications of Convolution Integral Equations*, Kluwer Academic Publishers, Dordrecht, Netherlands, 1992.

Technical Report DTFH6 1-90-R-00074-FG, “Development and Testing of Dynamic Traffic Assignment and simulation Procedures for ATIS/ATMS Applications”, prepared by Center for Transportation Research, The University of Texas at Austin, Austin, Texas, June 1993.

Widder, D.V., *The Laplace Transform*, Princeton University Press, 1941

APPENDIX A: OBTAINING MEAN AND VARIANCE OF GAMMA DISTRIBUTED VARIABLES

Assume the variable t has Gamma probability distribution function, that is

$$p(t) = \frac{\alpha^n e^{-\alpha t} t^{n-1}}{\Gamma(n)}, \quad (A-1)$$

where n and α are associated parameters. The mean of t can be obtained by

$$mean = \int_0^{\infty} t p(t) dt \quad (A-2)$$

Define a new variable τ by

$$\tau = \alpha t \quad (A-3)$$

Equations (A-1) to (A-3) result in

$$\begin{aligned} mean &= \int_0^{\infty} t \frac{\alpha^n e^{-\alpha t} t^{n-1}}{\Gamma(n)} dt \\ &= \frac{\alpha^n}{\Gamma(n)} \int_0^{\infty} e^{-\tau} \left(\frac{\tau}{\alpha}\right)^n \frac{1}{\alpha} d\tau \\ &= \frac{1}{\Gamma(n)} \int_0^{\infty} e^{-\tau} \tau^n d\tau \\ &= \frac{\Gamma(n+1)}{\alpha \Gamma(n)} \end{aligned} \quad (A-4)$$

The variance can be obtained by

$$var = \int_0^{\infty} (t - mean)^2 p(t) dt \quad (A-5)$$

Substitute the distribution given by Equation (A-1) and the obtained mean given by Equation (A-4) into Equation (A-5). We have

$$\begin{aligned} var &= \int_0^{\infty} \left(t - \frac{\Gamma(n+1)}{\alpha \Gamma(n)}\right)^2 p(t) dt \\ &= \int_0^{\infty} t^2 p(t) dt - 2 \frac{\Gamma(n+1)}{\alpha \Gamma(n)} \int_0^{\infty} t p(t) dt + \left[\frac{\Gamma(n+1)}{\alpha \Gamma(n)}\right]^2 \int_0^{\infty} p(t) dt \\ &= \frac{1}{\alpha^2 \Gamma^2(n)} [\Gamma(n+2)\Gamma(n) - \Gamma^2(n+1)] \end{aligned} \quad (A-6)$$

When n is an integer,

$$\Gamma(n) = (n-1)! \quad (A-7)$$

Equations (A-4) and (A-6) then can be simplified as

$$mean = \frac{n}{\alpha},$$

(A-8)

and

$$var = \frac{n}{\alpha^2}.$$

(A-9)

APPENDIX B: MATLAB PROGRAMS FOR THE ARRIVING-ON-TIME PROBLEM IN THE 5-NODE NETWORK

```
%pathgamma8r5n.m
clear all
diary('Output8r5n.txt')

%input number of i in the quadrature approximation
M = 8
%input number of F(s) to be evaluated
L = 8
%roots are where the f(t) is to be evaluated
root = [1.98550718E-02  1.01666761E-01  2.37233795E-01
        4.08282679E-01  5.91717321E-01  7.62766205E-01
        8.98333239E-01  9.80144928E-01]
weight = [5.06142681E-02  1.11190517E-01  1.56853323E-01
          1.81341892E-01  1.81341892E-01  1.56853323E-01
          1.11190517E-01  5.06142681E-02]

%input the parameters of Gamma distribution for each link
alfa = [0 1 2 2 0; 1 0 2 2 1; 2 2 0 1 2; 2 2 1 0 1; 0 1 2 1 0]
n = [0 1 2 2 0; 1 0 2 1 1; 2 2 0 2 1; 2 1 2 0 2; 0 1 1 2 0]

connect = [0 1 1 1 0; 1 0 1 1 1; 1 1 0 1 1; 1 1 1 0 1; 0 1 1 1 0]
x=-log(root)
%input number of nodes N in this network
N=5

%the probability of starting at node N and
%arriving at node N in time t is always 1
for t=1:M
    u(N,t)=1;
end

%initial guess on probability based on direct link between nodes i
and N.
s = 1:L;
lapu(1,:)=zeros(1,L)
lapu(2,:)=(1 ./ s) .* ((1 ./ (1+s)))
lapu(3,:)=(1 ./ s) .* ((2 ./ (2+s)))
lapu(4,:)=(1 ./ s) .* ((1 ./ (1+s)).^2)

theu=lapinvmfunc18(s,lapu(1,:));
u(1,:) = theu';
theu=lapinvmfunc18(s,lapu(2,:));
u(2,:)= theu';
theu=lapinvmfunc18(s,lapu(3,:));
u(3,:)= theu';
theu=lapinvmfunc18(s,lapu(4,:));
u(4,:)= theu';
```

```

disp('The initial u(t) is:')
disp(u)

%Start Successive Approximation
for i=1:(N-1);
    next_node(i,:)=ones(1,M)*N;
end

for k=1:N
    k
    for i=1:(N-1)
        i
        max=u(i,:);
        for j=1:N %j could be a set of possible next nodes from i
            j
            if ((j ~= i) & (connect(i,j) ~= 0))
                for t=1:M
                    t;
                    ff(t)=gamma(x(t),alfa(i,j),n(i,j));
                end
                [sp,lp] = Lapfunc(root,ff,weight,L,M);
                thep=lapinvfunc18(sp,lp);
                [su,lu_current] = Lapfunc(root,u(j,:),weight,L,M);
                lu_new=lu_current.*lp;
                theu=lapinvfunc18(su,lu_new)

                for t=1:M
                    t;
                    if theu(t)>max(t)
                        max(t)=theu(t);
                        next_node(i,t)=j;
                    end
                end
            end
            max
            next_node(i,:)
        end
        u(i,:)=max
        next_node
    end
end
diary off

```

**APPENDIX C: MATLAB OUTPUT FILE CONTAINING DETAIL
INTERMEDIATE SUCCESSIVE APPROXIMATIONS FOR THE
ARRIVING-ON-TIME PROBLEM IN THE 5-NODE NETWORK**

M =

8

L =

8

root =

Columns 1 through 7

0.0199	0.1017	0.2372	0.4083	0.5917	0.7628
0.8983					

Column 8

0.9801

weight =

Columns 1 through 7

0.0506	0.1112	0.1569	0.1813	0.1813	0.1569
0.1112					

Column 8

0.0506

alfa =

0	1	2	2	0
1	0	2	2	1
2	2	0	1	2
2	2	1	0	1
0	1	2	1	0

n =

0	1	2	2	0
1	0	2	1	1

2	2	0	2	1
2	1	2	0	2
0	1	1	2	0

connect =

0	1	1	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1
0	1	1	1	0

x =

Columns 1 through 7

3.9193	2.2861	1.4387	0.8958	0.5247	0.2708
0.1072					

Column 8

0.0201

N =

5

The initial u(t) is:

Columns 1 through 7

0	0	0	0	0	0	
0	0.9799	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016	0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929	0.9047	0.6646	0.4223	0.2255	0.0981	0.0304
0.0055	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000						

Column 8

0
0.0197
0.0391
0.0001
1.0000

k =

1

u =

Columns 1 through 7

0.9828	0.8351	0.5483	0.2672	0.0980	0.0303
0.0055					
0.9828	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016					
0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929					
0.9626	0.8066	0.5822	0.3498	0.1665	0.0561
0.0103					
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000					

Column 8

0.0001
0.0197
0.0391
0.0003
1.0000

next_node =

3	3	3	3	2	2	2	3
3	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5
2	2	2	2	2	2	2	2

k =

2

u =

Columns 1 through 7

0.9828	0.8351	0.5483	0.2672	0.0980	0.0303
0.0056					
0.9828	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016					

0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929					
0.9626	0.8066	0.5822	0.3498	0.1665	0.0561
0.0103					
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000					

Column 8

0.0001
0.0197
0.0391
0.0003
1.0000

next_node =

3	3	3	3	2	2	2	3
3	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5
2	2	2	2	2	2	2	2

k =

3

u =

Columns 1 through 7

0.9828	0.8351	0.5483	0.2672	0.0980	0.0303
0.0056					
0.9828	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016					
0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929					
0.9626	0.8066	0.5822	0.3498	0.1665	0.0561
0.0103					
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000					

Column 8

0.0001
0.0197
0.0391
0.0003
1.0000

next_node =

3	3	3	3	2	2	2	3
3	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5
2	2	2	2	2	2	2	2

k =

4

u =

Columns 1 through 7

0.9828	0.8351	0.5483	0.2672	0.0980	0.0303
0.0056					
0.9828	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016					
0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929					
0.9626	0.8066	0.5822	0.3498	0.1665	0.0561
0.0103					
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000					

Column 8

0.0001
0.0197
0.0391
0.0003
1.0000

next_node =

3	3	3	3	2	2	2	3
3	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5
2	2	2	2	2	2	2	2

k =

5

u =

Columns 1 through 7

0.9828	0.8351	0.5483	0.2672	0.0980	0.0303
0.0056					
0.9828	0.8980	0.7632	0.5916	0.4079	0.2370
0.1016					
0.9993	0.9891	0.9444	0.8332	0.6493	0.4178
0.1929					
0.9626	0.8066	0.5822	0.3498	0.1665	0.0561
0.0103					
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000					

Column 8

0.0001
0.0197
0.0391
0.0003
1.0000

next_node =

3	3	3	3	2	2	2	3
3	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5
2	2	2	2	2	2	2	2


```

0.0123
    0
    0
    0
    0
0.0122
0.1512
0.1508
    0
    0
    0
    0
0.0121
0.1482
1.0000

```

```
next_node =
```

[illegible]

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365	0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365	0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365	0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.9145	0.9098	0.8429	0.6207	0.3056	0.0804
0.0069	0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363	0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516	0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515	0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515	0	0	0	0	0	0
0	0.8957	0.8826	0.7582	0.4611	0.1645	0.0269
0.0011	0.9275	0.9233	0.8576	0.6317	0.3084	0.0800
0.0068	0.9530	0.9520	0.9258	0.7896	0.5100	0.2050
0.0361	0.9751	0.9749	0.9686	0.9090	0.7339	0.4400
0.1511	0.9995	0.9986	1.0000	0.9834	0.9228	0.7600


```

0.0124
0.0123
    0
    0
    0
    0
0.0122
0.1512
0.1508
    0
    0
    0
    0
0.0121
0.1482
1.0000

```

```
next node =
```

[illegible]

0						
	0	0	0	0	0	0
0						
	0	0	0	0	0	0
0						
	0.8940	0.8884	0.8201	0.6032	0.2999	0.0804
0.0070						
	0.8970	0.8915	0.8235	0.6058	0.3007	0.0804
0.0070						
	0.8996	0.8942	0.8263	0.6079	0.3015	0.0804
0.0070						
	0.9013	0.8960	0.8282	0.6094	0.3019	0.0804
0.0070						
	0.9019	0.8967	0.8290	0.6100	0.3022	0.0803
0.0070						
	0	0	0	0	0	0
0						
	0.8805	0.8665	0.7423	0.4521	0.1635	0.0274
0.0012						
	0.9124	0.9076	0.8406	0.6187	0.3047	0.0804
0.0070						
	0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365						
	0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365						
	0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365						
	0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364						
	0.8677	0.8375	0.6433	0.3107	0.0790	0.0080
0.0002						
	0.8992	0.8863	0.7618	0.4626	0.1648	0.0271
0.0011						
	0.9246	0.9202	0.8542	0.6290	0.3078	0.0804
0.0069						
	0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363						
	0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516						
	0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515						
	0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515						
	0.8804	0.8511	0.6549	0.3150	0.0793	0.0079
0.0002						
	0.9057	0.8932	0.7686	0.4665	0.1653	0.0269
0.0011						
	0.9310	0.9269	0.8614	0.6345	0.3092	0.0800
0.0068						
	0.9530	0.9520	0.9258	0.7896	0.5100	0.2050
0.0361						
	0.9751	0.9749	0.9686	0.9090	0.7339	0.4400
0.1511						


```

0.0124
0.0124
0.0123
  0
    0
      0
        0
0.0122
0.1512
0.1508
  0
    0
      0
        0
0.0121
0.1482
1.0000

```

```
next_node =
```

49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
25	25	25	25	25	25	25	49
26	26	26	26	26	25	25	49
27	27	27	27	27	25	25	49
28	28	28	28	28	26	26	49
28	28	28	28	28	27	27	49
49	49	49	49	49	49	49	49
31	31	31	31	31	17	17	49
32	32	32	32	32	25	25	49
33	33	33	33	33	33	33	49
34	34	34	34	34	33	33	49
35	35	35	35	35	33	33	49
35	35	35	35	35	34	34	49
37	37	37	37	23	23	23	49
38	38	38	38	38	24	24	49
39	39	39	39	39	25	25	49
40	40	40	40	40	33	33	49

41	41	41	41	41	41	41	41
42	42	42	42	42	41	41	41
42	42	42	42	42	41	41	41
44	44	44	44	30	30	29	49
45	45	45	45	45	31	31	49
46	46	46	46	46	32	32	49
47	47	47	47	47	33	33	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	37	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	39	39	49
47	47	47	47	47	40	40	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49

k =

4

u =

Columns 1 through 7

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.8383	0.8222	0.6995	0.4275	0.1585	0.0273
0.0012	0.8435	0.8278	0.7048	0.4305	0.1592	0.0273
0.0012	0.8482	0.8327	0.7095	0.4333	0.1598	0.0274
0.0012	0.8520	0.8366	0.7133	0.4355	0.1602	0.0273
0.0012	0.8545	0.8393	0.7159	0.4370	0.1605	0.0273
0.0012						

0.8555	0.8403	0.7170	0.4376	0.1607	0.0273
0.0012					
0.8377	0.8058	0.6164	0.3001	0.0784	0.0082
0.0003					
0.8696	0.8551	0.7313	0.4456	0.1622	0.0274
0.0012					
0.8940	0.8884	0.8201	0.6032	0.2999	0.0804
0.0070					
0.8970	0.8915	0.8235	0.6058	0.3007	0.0804
0.0070					
0.8996	0.8942	0.8263	0.6079	0.3015	0.0804
0.0070					
0.9013	0.8960	0.8282	0.6094	0.3019	0.0804
0.0070					
0.9019	0.8967	0.8290	0.6100	0.3022	0.0803
0.0070					
0.8648	0.8344	0.6406	0.3094	0.0790	0.0081
0.0002					
0.8902	0.8768	0.7524	0.4574	0.1643	0.0274
0.0012					
0.9124	0.9076	0.8406	0.6187	0.3047	0.0804
0.0070					
0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365					
0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365					
0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365					
0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364					
0.8774	0.8478	0.6519	0.3138	0.0792	0.0080
0.0002					
0.9026	0.8899	0.7653	0.4645	0.1651	0.0271
0.0011					
0.9246	0.9202	0.8542	0.6290	0.3078	0.0804
0.0069					
0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363					
0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516					
0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515					
0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515					
0.8840	0.8549	0.6580	0.3162	0.0793	0.0079
0.0002					
0.9091	0.8968	0.7721	0.4683	0.1655	0.0269
0.0011					
0.9310	0.9269	0.8614	0.6345	0.3092	0.0800
0.0068					
0.9530	0.9520	0.9258	0.7896	0.5100	0.2050
0.0361					
0.9751	0.9749	0.9686	0.9090	0.7339	0.4400


```

0
0.0124
0.0124
0.0123
0
0
0
0
0
0.0122
0.1512
0.1508
0
0
0
0
0.0121
0.1482
1.0000

```

```
next_node =
```

```

49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
17  17  17  17  17  17  17  49
18  18  18  18  18  17  17  49
19  19  19  19  19  17  17  49
20  20  20  20  20  18  18  49
21  21  21  21  21  19  19  49
21  21  21  21  21  20  20  49
23  23  23  23  23  9   9   49
24  24  24  24  24  17  17  49
25  25  25  25  25  25  25  49
26  26  26  26  26  25  25  49
27  27  27  27  27  25  25  49
28  28  28  28  28  26  26  49
28  28  28  28  28  27  27  49
30  30  30  30  30  16  15  49
31  31  31  31  31  17  17  49
32  32  32  32  32  25  25  49
33  33  33  33  33  33  33  49
34  34  34  34  34  33  33  49
35  35  35  35  35  33  33  49
35  35  35  35  35  34  34  49
37  37  37  37  23  23  23  49
38  38  38  38  38  24  24  49
39  39  39  39  39  25  25  49

```

40	40	40	40	40	33	33	49
41	41	41	41	41	41	41	41
42	42	42	42	42	41	41	41
42	42	42	42	42	41	41	41
44	44	44	44	30	30	29	49
45	45	45	45	45	31	31	49
46	46	46	46	46	32	32	49
47	47	47	47	47	33	33	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	39	39	49
47	47	47	47	47	40	40	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49

k =

5

u =

Columns 1 through 7

0.7562	0.7205	0.5450	0.2692	0.0739	0.0081
0.0002					
0.7664	0.7312	0.5539	0.2732	0.0746	0.0081
0.0002					
0.7755	0.7407	0.5619	0.2768	0.0752	0.0081
0.0002					
0.7832	0.7488	0.5686	0.2797	0.0756	0.0081
0.0002					
0.7894	0.7552	0.5740	0.2822	0.0760	0.0081
0.0003					
0.7937	0.7598	0.5778	0.2839	0.0763	0.0081
0.0002					
0.7954	0.7615	0.5794	0.2846	0.0765	0.0081
0.0003					
0.8128	0.7798	0.5946	0.2908	0.0773	0.0082
0.0003					
0.8383	0.8222	0.6995	0.4275	0.1585	0.0273
0.0012					
0.8435	0.8278	0.7048	0.4305	0.1592	0.0273
0.0012					
0.8482	0.8327	0.7095	0.4333	0.1598	0.0274
0.0012					
0.8520	0.8366	0.7133	0.4355	0.1602	0.0273
0.0012					
0.8545	0.8393	0.7159	0.4370	0.1605	0.0273

0.0012					
0.8555	0.8403	0.7170	0.4376	0.1607	0.0273
0.0012					
0.8469	0.8156	0.6246	0.3032	0.0787	0.0082
0.0003					
0.8696	0.8551	0.7313	0.4456	0.1622	0.0274
0.0012					
0.8940	0.8884	0.8201	0.6032	0.2999	0.0804
0.0070					
0.8970	0.8915	0.8235	0.6058	0.3007	0.0804
0.0070					
0.8996	0.8942	0.8263	0.6079	0.3015	0.0804
0.0070					
0.9013	0.8960	0.8282	0.6094	0.3019	0.0804
0.0070					
0.9019	0.8967	0.8290	0.6100	0.3022	0.0803
0.0070					
0.8681	0.8379	0.6435	0.3105	0.0790	0.0081
0.0002					
0.8902	0.8768	0.7524	0.4574	0.1643	0.0274
0.0012					
0.9124	0.9076	0.8406	0.6187	0.3047	0.0804
0.0070					
0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365					
0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365					
0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365					
0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364					
0.8808	0.8513	0.6549	0.3148	0.0792	0.0080
0.0002					
0.9026	0.8899	0.7653	0.4645	0.1651	0.0271
0.0011					
0.9246	0.9202	0.8542	0.6290	0.3078	0.0804
0.0069					
0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363					
0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516					
0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515					
0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515					
0.8874	0.8584	0.6610	0.3172	0.0793	0.0079
0.0002					
0.9091	0.8968	0.7721	0.4683	0.1655	0.0269
0.0011					
0.9310	0.9269	0.8614	0.6345	0.3092	0.0800
0.0068					
0.9530	0.9520	0.9258	0.7896	0.5100	0.2050
0.0361					


```

0
0
0.0124
0.0124
0.0123
0
0
0
0
0
0.0122
0.1512
0.1508
0
0
0
0
0.0121
0.1482
1.0000

```

```
next_node =
```

9	9	9	9	9	9	9	49
10	10	10	10	10	9	9	49
11	11	11	11	11	9	9	49
12	12	12	12	12	10	10	49
13	13	13	13	13	11	11	49
14	14	14	14	14	12	12	49
14	14	14	14	14	13	13	49
16	16	16	16	16	9	9	49
17	17	17	17	17	17	17	49
18	18	18	18	18	17	17	49
19	19	19	19	19	17	17	49
20	20	20	20	20	18	18	49
21	21	21	21	21	19	19	49
21	21	21	21	21	20	20	49
23	23	23	23	23	9	9	49
24	24	24	24	24	17	17	49
25	25	25	25	25	25	25	49
26	26	26	26	26	25	25	49
27	27	27	27	27	25	25	49
28	28	28	28	28	26	26	49
28	28	28	28	28	27	27	49
30	30	30	30	30	16	15	49
31	31	31	31	31	17	17	49
32	32	32	32	32	25	25	49
33	33	33	33	33	33	33	49
34	34	34	34	34	33	33	49
35	35	35	35	35	33	33	49
35	35	35	35	35	34	34	49
37	37	37	37	23	23	23	49
38	38	38	38	38	24	24	49

39	39	39	39	39	25	25	49
40	40	40	40	40	33	33	49
41	41	41	41	41	41	41	41
42	42	42	42	42	41	41	41
42	42	42	42	42	41	41	41
44	44	44	44	30	30	29	49
45	45	45	45	45	31	31	49
46	46	46	46	46	32	32	49
47	47	47	47	47	33	33	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	39	39	49
47	47	47	47	47	40	40	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49

k =

6

u =

Columns 1 through 7

0.7562	0.7205	0.5450	0.2692	0.0739	0.0081
0.0002					
0.7664	0.7312	0.5539	0.2732	0.0746	0.0081
0.0002					
0.7755	0.7407	0.5619	0.2768	0.0752	0.0081
0.0002					
0.7832	0.7488	0.5686	0.2797	0.0756	0.0081
0.0002					
0.7894	0.7552	0.5740	0.2822	0.0760	0.0081
0.0003					
0.7937	0.7598	0.5778	0.2839	0.0763	0.0081
0.0002					
0.7954	0.7615	0.5794	0.2846	0.0765	0.0081
0.0003					
0.8128	0.7798	0.5946	0.2908	0.0773	0.0082
0.0003					
0.8383	0.8222	0.6995	0.4275	0.1585	0.0273
0.0012					
0.8435	0.8278	0.7048	0.4305	0.1592	0.0273
0.0012					
0.8482	0.8327	0.7095	0.4333	0.1598	0.0274
0.0012					
0.8520	0.8366	0.7133	0.4355	0.1602	0.0273
0.0012					

0.8545	0.8393	0.7159	0.4370	0.1605	0.0273
0.0012					
0.8555	0.8403	0.7170	0.4376	0.1607	0.0273
0.0012					
0.8469	0.8156	0.6246	0.3032	0.0787	0.0082
0.0003					
0.8696	0.8551	0.7313	0.4456	0.1622	0.0274
0.0012					
0.8940	0.8884	0.8201	0.6032	0.2999	0.0804
0.0070					
0.8970	0.8915	0.8235	0.6058	0.3007	0.0804
0.0070					
0.8996	0.8942	0.8263	0.6079	0.3015	0.0804
0.0070					
0.9013	0.8960	0.8282	0.6094	0.3019	0.0804
0.0070					
0.9019	0.8967	0.8290	0.6100	0.3022	0.0803
0.0070					
0.8681	0.8379	0.6435	0.3105	0.0790	0.0081
0.0002					
0.8902	0.8768	0.7524	0.4574	0.1643	0.0274
0.0012					
0.9124	0.9076	0.8406	0.6187	0.3047	0.0804
0.0070					
0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365					
0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365					
0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365					
0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364					
0.8808	0.8513	0.6549	0.3148	0.0792	0.0080
0.0002					
0.9026	0.8899	0.7653	0.4645	0.1651	0.0271
0.0011					
0.9246	0.9202	0.8542	0.6290	0.3078	0.0804
0.0069					
0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363					
0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516					
0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515					
0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515					
0.8874	0.8584	0.6610	0.3172	0.0793	0.0079
0.0002					
0.9091	0.8968	0.7721	0.4683	0.1655	0.0269
0.0011					
0.9310	0.9269	0.8614	0.6345	0.3092	0.0800
0.0068					
0.9530	0.9520	0.9258	0.7896	0.5100	0.2050


```

0
0
0
0.0124
0.0124
0.0123
0
0
0
0
0
0.0122
0.1512
0.1508
0
0
0
0
0.0121
0.1482
1.0000

```

```
next_node =
```

9	9	9	9	9	9	9	49
10	10	10	10	10	9	9	49
11	11	11	11	11	9	9	49
12	12	12	12	12	10	10	49
13	13	13	13	13	11	11	49
14	14	14	14	14	12	12	49
14	14	14	14	14	13	13	49
16	16	16	16	16	9	9	49
17	17	17	17	17	17	17	49
18	18	18	18	18	17	17	49
19	19	19	19	19	17	17	49
20	20	20	20	20	18	18	49
21	21	21	21	21	19	19	49
21	21	21	21	21	20	20	49
23	23	23	23	23	9	9	49
24	24	24	24	24	17	17	49
25	25	25	25	25	25	25	49
26	26	26	26	26	25	25	49
27	27	27	27	27	25	25	49
28	28	28	28	28	26	26	49
28	28	28	28	28	27	27	49
30	30	30	30	30	16	15	49
31	31	31	31	31	17	17	49
32	32	32	32	32	25	25	49
33	33	33	33	33	33	33	49
34	34	34	34	34	33	33	49
35	35	35	35	35	33	33	49
35	35	35	35	35	34	34	49
37	37	37	37	23	23	23	49

38	38	38	38	38	24	24	49
39	39	39	39	39	25	25	49
40	40	40	40	40	33	33	49
41	41	41	41	41	41	41	41
42	42	42	42	42	41	41	41
42	42	42	42	42	41	41	41
44	44	44	44	30	30	29	49
45	45	45	45	45	31	31	49
46	46	46	46	46	32	32	49
47	47	47	47	47	33	33	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	39	39	49
47	47	47	47	47	40	40	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49

k =

7

u =

Columns 1 through 7

0.7562	0.7205	0.5450	0.2692	0.0739	0.0081
0.0002					
0.7664	0.7312	0.5539	0.2732	0.0746	0.0081
0.0002					
0.7755	0.7407	0.5619	0.2768	0.0752	0.0081
0.0002					
0.7832	0.7488	0.5686	0.2797	0.0756	0.0081
0.0002					
0.7894	0.7552	0.5740	0.2822	0.0760	0.0081
0.0003					
0.7937	0.7598	0.5778	0.2839	0.0763	0.0081
0.0002					
0.7954	0.7615	0.5794	0.2846	0.0765	0.0081
0.0003					
0.8128	0.7798	0.5946	0.2908	0.0773	0.0082
0.0003					
0.8383	0.8222	0.6995	0.4275	0.1585	0.0273
0.0012					
0.8435	0.8278	0.7048	0.4305	0.1592	0.0273
0.0012					
0.8482	0.8327	0.7095	0.4333	0.1598	0.0274
0.0012					
0.8520	0.8366	0.7133	0.4355	0.1602	0.0273

0.0012					
0.8545	0.8393	0.7159	0.4370	0.1605	0.0273
0.0012					
0.8555	0.8403	0.7170	0.4376	0.1607	0.0273
0.0012					
0.8469	0.8156	0.6246	0.3032	0.0787	0.0082
0.0003					
0.8696	0.8551	0.7313	0.4456	0.1622	0.0274
0.0012					
0.8940	0.8884	0.8201	0.6032	0.2999	0.0804
0.0070					
0.8970	0.8915	0.8235	0.6058	0.3007	0.0804
0.0070					
0.8996	0.8942	0.8263	0.6079	0.3015	0.0804
0.0070					
0.9013	0.8960	0.8282	0.6094	0.3019	0.0804
0.0070					
0.9019	0.8967	0.8290	0.6100	0.3022	0.0803
0.0070					
0.8681	0.8379	0.6435	0.3105	0.0790	0.0081
0.0002					
0.8902	0.8768	0.7524	0.4574	0.1643	0.0274
0.0012					
0.9124	0.9076	0.8406	0.6187	0.3047	0.0804
0.0070					
0.9365	0.9351	0.9069	0.7712	0.5007	0.2046
0.0365					
0.9381	0.9368	0.9087	0.7730	0.5016	0.2046
0.0365					
0.9394	0.9381	0.9102	0.7744	0.5024	0.2047
0.0365					
0.9399	0.9385	0.9107	0.7749	0.5027	0.2047
0.0364					
0.8808	0.8513	0.6549	0.3148	0.0792	0.0080
0.0002					
0.9026	0.8899	0.7653	0.4645	0.1651	0.0271
0.0011					
0.9246	0.9202	0.8542	0.6290	0.3078	0.0804
0.0069					
0.9467	0.9456	0.9186	0.7826	0.5064	0.2049
0.0363					
0.9707	0.9704	0.9636	0.9030	0.7292	0.4391
0.1516					
0.9714	0.9711	0.9644	0.9039	0.7300	0.4392
0.1515					
0.9718	0.9715	0.9648	0.9044	0.7304	0.4393
0.1515					
0.8874	0.8584	0.6610	0.3172	0.0793	0.0079
0.0002					
0.9091	0.8968	0.7721	0.4683	0.1655	0.0269
0.0011					
0.9310	0.9269	0.8614	0.6345	0.3092	0.0800
0.0068					


```

0
0
0
0
0.0124
0.0124
0.0123
0
0
0
0
0
0.0122
0.1512
0.1508
0
0
0
0
0.0121
0.1482
1.0000

```

```
next_node =
```

```

9      9      9      9      9      9      9      49
10     10     10     10     10     9      9      49
11     11     11     11     11     9      9      49
12     12     12     12     12     10     10     49
13     13     13     13     13     11     11     49
14     14     14     14     14     12     12     49
14     14     14     14     14     13     13     49
16     16     16     16     16     9      9      49
17     17     17     17     17     17     17     49
18     18     18     18     18     17     17     49
19     19     19     19     19     17     17     49
20     20     20     20     20     18     18     49
21     21     21     21     21     19     19     49
21     21     21     21     21     20     20     49
23     23     23     23     23     9      9      49
24     24     24     24     24     17     17     49
25     25     25     25     25     25     25     49
26     26     26     26     26     25     25     49
27     27     27     27     27     25     25     49
28     28     28     28     28     26     26     49
28     28     28     28     28     27     27     49
30     30     30     30     30     16     15     49
31     31     31     31     31     17     17     49
32     32     32     32     32     25     25     49
33     33     33     33     33     33     33     49
34     34     34     34     34     33     33     49
35     35     35     35     35     33     33     49
35     35     35     35     35     34     34     49

```

37	37	37	37	23	23	23	49
38	38	38	38	38	24	24	49
39	39	39	39	39	25	25	49
40	40	40	40	40	33	33	49
41	41	41	41	41	41	41	41
42	42	42	42	42	41	41	41
42	42	42	42	42	41	41	41
44	44	44	44	30	30	29	49
45	45	45	45	45	31	31	49
46	46	46	46	46	32	32	49
47	47	47	47	47	33	33	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	39	39	49
47	47	47	47	47	40	40	49
48	48	48	48	48	41	41	41
49	49	49	49	49	49	49	49

**APPENDIX E: SOTA PROBLEM TESTED WITH A 49-NODE NETWORK
USING THE SECOND SET OF GAMMA DISTRIBUTIONS (DETAILED
COMPUTER OUTPUTS OF INTERMEDIATE SUCCESSIVE
APPROXIMATIONS)**

The parameters in gamma distributions are set to be $n = 0.2\log_{10}(10+0.8i+0.7j)$ and $\alpha = \log_{10}(12+1.2i+0.8j)$.

The intermediate results are shown below. The integer k is the index of the successive approximation. Matrix u contains the maximum probability of arriving at the destination within time t or less, starting from node i . Matrix next_node contains the optimal successor node from each starting node i with remaining time t . The columns of the two matrices correspond to the time t . The rows of the two matrices correspond to the starting node i . The results indicate that the convergence is reached after the 5th iteration of successive approximation. The results below are from row computer outputs. We apologize for the ugly format.

The initial $u(t)$ is:
Columns 1 through 7

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0


```

0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0.3129
0.3123
0
0
0
0
0
0
0.3089
1.0000

```

k = 1

u =

Columns 1 through 7

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0


```

0
0
0
0
0
0
0
0
0
0
0
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

```

49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49

```


	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580	0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581	0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583	0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.5531	0.5332	0.4646	0.3409	0.1966	0.0770
0.0135	0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589	0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106	0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108	0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110						


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

```

49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49

```

49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
49	49	49	49	49	49	49	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
37	37	37	37	37	37	37	49
38	38	38	38	38	38	38	49
46	46	46	46	39	39	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

$k = 3$

$u =$

Columns 1 through 7

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.5234	0.5030	0.4364	0.3196	0.1849	0.0732
0.0130	0.5277	0.5073	0.4404	0.3226	0.1865	0.0736
0.0131	0.5313	0.5110	0.4438	0.3252	0.1879	0.0741
0.0131	0.5337	0.5134	0.4461	0.3269	0.1889	0.0744
0.0131	0.5346	0.5143	0.4470	0.3276	0.1893	0.0746
0.0132	0	0	0	0	0	0
0	0.4437	0.4138	0.3303	0.2096	0.0974	0.0272
0.0026	0.5504	0.5304	0.4620	0.3386	0.1950	0.0763
0.0134	0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580	0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581	0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583	0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583	0.3767	0.3365	0.2419	0.1302	0.0474	0.0091
0.0004	0.4686	0.4387	0.3514	0.2228	0.1028	0.0284
0.0025	0.5696	0.5500	0.4803	0.3524	0.2025	0.0787
0.0136	0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589	0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106	0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108	0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110						


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

```

49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
49 49 49 49 49 49 49 49
25 25 25 25 25 25 25 49
26 26 26 26 26 26 25 49
27 27 27 27 27 27 25 49
28 28 28 28 28 28 26 49
28 28 28 28 28 28 27 49
49 49 49 49 49 49 49 49

```

31	31	31	31	31	31	31	49
32	32	32	32	32	32	32	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
37	37	37	37	37	37	23	49
38	38	38	38	38	38	24	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
44	44	44	44	44	44	30	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	37	37	37	37	37	49
45	45	45	45	38	38	38	49
46	46	46	46	46	46	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

k = 4

u =

Columns 1 through 7

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0.3960	0.3664	0.2902	0.1840	0.0863	0.0247
0.0024						

0.4016	0.3720	0.2949	0.1870	0.0876	0.0249
0.0024					
0.4067	0.3770	0.2992	0.1897	0.0888	0.0252
0.0024					
0.4107	0.3810	0.3025	0.1919	0.0897	0.0254
0.0024					
0.4134	0.3837	0.3048	0.1933	0.0903	0.0255
0.0024					
0.4144	0.3847	0.3057	0.1939	0.0906	0.0256
0.0024					
0.3453	0.3064	0.2191	0.1181	0.0434	0.0085
0.0003					
0.4313	0.4015	0.3197	0.2027	0.0942	0.0265
0.0024					
0.5234	0.5030	0.4364	0.3196	0.1849	0.0732
0.0130					
0.5277	0.5073	0.4404	0.3226	0.1865	0.0736
0.0131					
0.5313	0.5110	0.4438	0.3252	0.1879	0.0741
0.0131					
0.5337	0.5134	0.4461	0.3269	0.1889	0.0744
0.0131					
0.5346	0.5143	0.4470	0.3276	0.1893	0.0746
0.0132					
0.3740	0.3339	0.2398	0.1289	0.0469	0.0090
0.0003					
0.4570	0.4270	0.3414	0.2165	0.1001	0.0278
0.0026					
0.5504	0.5304	0.4620	0.3386	0.1950	0.0763
0.0134					
0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580					
0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581					
0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583					
0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583					
0.3882	0.3475	0.2501	0.1343	0.0486	0.0093
0.0004					
0.4736	0.4436	0.3556	0.2255	0.1039	0.0286
0.0025					
0.5696	0.5500	0.4803	0.3524	0.2025	0.0787
0.0136					
0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589					
0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106					
0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108					
0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110					


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

```

49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
49  49  49  49  49  49  49  49
17  17  17  17  17  17  17  49
18  18  18  18  18  18  17  49
19  19  19  19  19  19  17  49
20  20  20  20  20  20  18  49
21  21  21  21  21  20  19  49
21  21  21  21  21  20  20  49
23  23  23  23  23  23  9  49
24  24  24  24  24  24  17  49
25  25  25  25  25  25  25  49
26  26  26  26  26  26  25  49
27  27  27  27  27  27  25  49
28  28  28  28  28  28  26  49
28  28  28  28  28  28  27  49
30  30  30  30  30  30  16  49

```

31	31	31	31	31	31	31	49
32	32	32	32	32	32	32	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
37	37	37	37	37	37	23	49
38	38	38	38	38	38	24	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
44	44	44	44	44	44	30	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	37	37	37	49
45	45	45	45	45	45	38	49
46	46	46	46	46	46	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

k = 5

u =

Columns 1 through 7

0.2771	0.2417	0.1708	0.0923	0.0346	0.0070
0.0003					
0.2848	0.2489	0.1762	0.0952	0.0356	0.0072
0.0003					
0.2918	0.2556	0.1811	0.0978	0.0365	0.0074
0.0003					
0.2978	0.2612	0.1853	0.1001	0.0373	0.0074
0.0003					
0.3026	0.2658	0.1887	0.1019	0.0379	0.0076
0.0003					
0.3059	0.2690	0.1911	0.1032	0.0383	0.0076
0.0003					
0.3072	0.2702	0.1920	0.1037	0.0385	0.0076
0.0003					
0.3232	0.2854	0.2033	0.1096	0.0405	0.0080
0.0003					
0.3960	0.3664	0.2902	0.1840	0.0863	0.0247
0.0024					

0.4016	0.3720	0.2949	0.1870	0.0876	0.0249
0.0024					
0.4067	0.3770	0.2992	0.1897	0.0888	0.0252
0.0024					
0.4107	0.3810	0.3025	0.1919	0.0897	0.0254
0.0024					
0.4134	0.3837	0.3048	0.1933	0.0903	0.0255
0.0024					
0.4144	0.3847	0.3057	0.1939	0.0906	0.0256
0.0024					
0.3557	0.3164	0.2265	0.1219	0.0446	0.0087
0.0003					
0.4313	0.4015	0.3197	0.2027	0.0942	0.0265
0.0024					
0.5234	0.5030	0.4364	0.3196	0.1849	0.0732
0.0130					
0.5277	0.5073	0.4404	0.3226	0.1865	0.0736
0.0131					
0.5313	0.5110	0.4438	0.3252	0.1879	0.0741
0.0131					
0.5337	0.5134	0.4461	0.3269	0.1889	0.0744
0.0131					
0.5346	0.5143	0.4470	0.3276	0.1893	0.0746
0.0132					
0.3780	0.3377	0.2426	0.1304	0.0473	0.0090
0.0003					
0.4570	0.4270	0.3414	0.2165	0.1001	0.0278
0.0026					
0.5504	0.5304	0.4620	0.3386	0.1950	0.0763
0.0134					
0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580					
0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581					
0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583					
0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583					
0.3923	0.3515	0.2531	0.1359	0.0491	0.0093
0.0004					
0.4736	0.4436	0.3556	0.2255	0.1039	0.0286
0.0025					
0.5696	0.5500	0.4803	0.3524	0.2025	0.0787
0.0136					
0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589					
0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106					
0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108					
0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110					


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

9	9	9	9	9	9	9	49
10	10	10	10	10	10	9	49
11	11	11	11	11	11	9	49
12	12	12	12	12	11	10	49
13	13	13	13	13	13	11	49
14	14	14	14	14	12	12	49
14	14	14	14	14	13	13	49
16	16	16	16	16	16	9	49
17	17	17	17	17	17	17	49
18	18	18	18	18	18	17	49
19	19	19	19	19	19	17	49
20	20	20	20	20	20	18	49
21	21	21	21	21	20	19	49
21	21	21	21	21	20	20	49
23	23	23	23	23	23	9	49
24	24	24	24	24	24	17	49
25	25	25	25	25	25	25	49
26	26	26	26	26	26	25	49
27	27	27	27	27	27	25	49
28	28	28	28	28	28	26	49
28	28	28	28	28	28	27	49
30	30	30	30	30	30	16	49

31	31	31	31	31	31	31	49
32	32	32	32	32	32	32	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
37	37	37	37	37	37	23	49
38	38	38	38	38	38	24	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
44	44	44	44	44	44	30	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	44	44	37	49
45	45	45	45	45	45	38	49
46	46	46	46	46	46	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

k = 6

u =

Columns 1 through 7

0.2771	0.2417	0.1708	0.0923	0.0346	0.0070
0.0003					
0.2848	0.2489	0.1762	0.0952	0.0356	0.0072
0.0003					
0.2918	0.2556	0.1811	0.0978	0.0365	0.0074
0.0003					
0.2978	0.2612	0.1853	0.1001	0.0373	0.0074
0.0003					
0.3026	0.2658	0.1887	0.1019	0.0379	0.0076
0.0003					
0.3059	0.2690	0.1911	0.1032	0.0383	0.0076
0.0003					
0.3072	0.2702	0.1920	0.1037	0.0385	0.0076
0.0003					
0.3232	0.2854	0.2033	0.1096	0.0405	0.0080
0.0003					
0.3960	0.3664	0.2902	0.1840	0.0863	0.0247
0.0024					

0.4016	0.3720	0.2949	0.1870	0.0876	0.0249
0.0024					
0.4067	0.3770	0.2992	0.1897	0.0888	0.0252
0.0024					
0.4107	0.3810	0.3025	0.1919	0.0897	0.0254
0.0024					
0.4134	0.3837	0.3048	0.1933	0.0903	0.0255
0.0024					
0.4144	0.3847	0.3057	0.1939	0.0906	0.0256
0.0024					
0.3557	0.3164	0.2265	0.1219	0.0446	0.0087
0.0003					
0.4313	0.4015	0.3197	0.2027	0.0942	0.0265
0.0024					
0.5234	0.5030	0.4364	0.3196	0.1849	0.0732
0.0130					
0.5277	0.5073	0.4404	0.3226	0.1865	0.0736
0.0131					
0.5313	0.5110	0.4438	0.3252	0.1879	0.0741
0.0131					
0.5337	0.5134	0.4461	0.3269	0.1889	0.0744
0.0131					
0.5346	0.5143	0.4470	0.3276	0.1893	0.0746
0.0132					
0.3780	0.3377	0.2426	0.1304	0.0473	0.0090
0.0003					
0.4570	0.4270	0.3414	0.2165	0.1001	0.0278
0.0026					
0.5504	0.5304	0.4620	0.3386	0.1950	0.0763
0.0134					
0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580					
0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581					
0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583					
0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583					
0.3923	0.3515	0.2531	0.1359	0.0491	0.0093
0.0004					
0.4736	0.4436	0.3556	0.2255	0.1039	0.0286
0.0025					
0.5696	0.5500	0.4803	0.3524	0.2025	0.0787
0.0136					
0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589					
0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106					
0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108					
0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110					


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

```

9      9      9      9      9      9      9      49
10     10     10     10     10     10     9      49
11     11     11     11     11     11     9      49
12     12     12     12     12     11     10     49
13     13     13     13     13     13     11     49
14     14     14     14     14     12     12     49
14     14     14     14     14     13     13     49
16     16     16     16     16     16     9      49
17     17     17     17     17     17     17     49
18     18     18     18     18     18     17     49
19     19     19     19     19     19     17     49
20     20     20     20     20     20     18     49
21     21     21     21     21     20     19     49
21     21     21     21     21     20     20     49
23     23     23     23     23     23     9      49
24     24     24     24     24     24     17     49
25     25     25     25     25     25     25     49
26     26     26     26     26     26     25     49
27     27     27     27     27     27     25     49
28     28     28     28     28     28     26     49
28     28     28     28     28     28     27     49
30     30     30     30     30     30     16     49

```

31	31	31	31	31	31	31	49
32	32	32	32	32	32	32	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
37	37	37	37	37	37	23	49
38	38	38	38	38	38	24	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
44	44	44	44	44	44	30	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	44	44	37	49
45	45	45	45	45	45	38	49
46	46	46	46	46	46	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

k = 7

u =

Columns 1 through 7

0.2771	0.2417	0.1708	0.0923	0.0346	0.0070
0.0003					
0.2848	0.2489	0.1762	0.0952	0.0356	0.0072
0.0003					
0.2918	0.2556	0.1811	0.0978	0.0365	0.0074
0.0003					
0.2978	0.2612	0.1853	0.1001	0.0373	0.0074
0.0003					
0.3026	0.2658	0.1887	0.1019	0.0379	0.0076
0.0003					
0.3059	0.2690	0.1911	0.1032	0.0383	0.0076
0.0003					
0.3072	0.2702	0.1920	0.1037	0.0385	0.0076
0.0003					
0.3232	0.2854	0.2033	0.1096	0.0405	0.0080
0.0003					
0.3960	0.3664	0.2902	0.1840	0.0863	0.0247
0.0024					

0.4016	0.3720	0.2949	0.1870	0.0876	0.0249
0.0024					
0.4067	0.3770	0.2992	0.1897	0.0888	0.0252
0.0024					
0.4107	0.3810	0.3025	0.1919	0.0897	0.0254
0.0024					
0.4134	0.3837	0.3048	0.1933	0.0903	0.0255
0.0024					
0.4144	0.3847	0.3057	0.1939	0.0906	0.0256
0.0024					
0.3557	0.3164	0.2265	0.1219	0.0446	0.0087
0.0003					
0.4313	0.4015	0.3197	0.2027	0.0942	0.0265
0.0024					
0.5234	0.5030	0.4364	0.3196	0.1849	0.0732
0.0130					
0.5277	0.5073	0.4404	0.3226	0.1865	0.0736
0.0131					
0.5313	0.5110	0.4438	0.3252	0.1879	0.0741
0.0131					
0.5337	0.5134	0.4461	0.3269	0.1889	0.0744
0.0131					
0.5346	0.5143	0.4470	0.3276	0.1893	0.0746
0.0132					
0.3780	0.3377	0.2426	0.1304	0.0473	0.0090
0.0003					
0.4570	0.4270	0.3414	0.2165	0.1001	0.0278
0.0026					
0.5504	0.5304	0.4620	0.3386	0.1950	0.0763
0.0134					
0.6644	0.6528	0.6059	0.5015	0.3507	0.1876
0.0580					
0.6673	0.6558	0.6089	0.5041	0.3525	0.1883
0.0581					
0.6696	0.6581	0.6112	0.5062	0.3538	0.1889
0.0583					
0.6704	0.6590	0.6121	0.5069	0.3544	0.1892
0.0583					
0.3923	0.3515	0.2531	0.1359	0.0491	0.0093
0.0004					
0.4736	0.4436	0.3556	0.2255	0.1039	0.0286
0.0025					
0.5696	0.5500	0.4803	0.3524	0.2025	0.0787
0.0136					
0.6832	0.6721	0.6253	0.5183	0.3618	0.1922
0.0589					
0.8222	0.8167	0.7928	0.7209	0.5938	0.4132
0.2106					
0.8237	0.8183	0.7945	0.7226	0.5951	0.4139
0.2108					
0.8245	0.8191	0.7954	0.7234	0.5958	0.4144
0.2110					


```

0
0
0.0018
0.0018
0.0019
0.0019
0
0
0
0.0019
0.0371
0.0371
0.0371
0
0
0
0.0019
0.0372
0.3129
0.3123
0
0
0
0.0019
0.0372
0.3089
1.0000

```

```
next_node =
```

9	9	9	9	9	9	9	49
10	10	10	10	10	10	9	49
11	11	11	11	11	11	9	49
12	12	12	12	12	11	10	49
13	13	13	13	13	13	11	49
14	14	14	14	14	12	12	49
14	14	14	14	14	13	13	49
16	16	16	16	16	16	9	49
17	17	17	17	17	17	17	49
18	18	18	18	18	18	17	49
19	19	19	19	19	19	17	49
20	20	20	20	20	20	18	49
21	21	21	21	21	20	19	49
21	21	21	21	21	20	20	49
23	23	23	23	23	23	9	49
24	24	24	24	24	24	17	49
25	25	25	25	25	25	25	49
26	26	26	26	26	26	25	49
27	27	27	27	27	27	25	49
28	28	28	28	28	28	26	49
28	28	28	28	28	28	27	49
30	30	30	30	30	30	16	49

31	31	31	31	31	31	31	49
32	32	32	32	32	32	32	49
33	33	33	33	33	33	33	33
34	34	34	34	34	34	33	33
35	35	35	35	35	35	35	33
35	35	35	35	35	35	34	34
37	37	37	37	37	37	23	49
38	38	38	38	38	38	24	49
39	39	39	39	39	39	39	49
40	40	40	40	40	40	40	33
41	41	41	41	41	41	41	41
42	42	42	42	42	42	41	41
42	42	42	42	42	42	41	41
44	44	44	44	44	44	30	49
45	45	45	45	45	45	31	49
46	46	46	46	46	46	46	49
47	47	47	47	47	47	47	33
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49
49	49	49	49	49	49	49	49
44	44	44	44	44	44	37	49
45	45	45	45	45	45	38	49
46	46	46	46	46	46	39	49
47	47	47	47	47	47	47	40
48	48	48	48	48	48	41	41
49	49	49	49	49	49	49	49

APPENDIX F: SOTA PROBLEM TESTED WITH A 100-NODE NETWORK USING THE FIRST SET OF GAMMA DISTRIBUTIONS

The parameters in gamma distributions of link travel times over link ij are set to be

$$n = 0.4 \cdot \log_{10}(10 + 0.8 \cdot i + 0.7 \cdot j)$$

and

$$\text{alfa} = 2 \cdot \log_{10}(12 + 1.2 \cdot i + 0.8 \cdot j).$$

**Table F-1: The Maximum Probability of On Time Arrival and the Optimal
Successor Node from each Node, and the Shortest Paths in the First 100-Node
Network**

From Node	SOTA	t1= 3.9193	t2= 2.2861	t3= 1.4387	t4= 0.8958	t5= 0.5247	t6= 0.2708	t7= 0.1072	t8= 0.0201	average- shortest- path cost & next nodes
1	$u_1(t)$	0.7595	0.6132	0.2713	0.0535	0	0	0	0	1.7003
	Successor Node	12	12	12	12	100	100	100	100	12
2	$u_2(t)$	0.762	0.6156	0.2724	0.0536	0	0	0	0	1.6998
	Successor Node	12	12	12	12	100	100	100	100	12
3	$u_3(t)$	0.7643	0.6178	0.2735	0.0538	0	0	0	0	1.6994
	Successor Node	12	12	12	12	100	100	100	100	12
4	$u_4(t)$	0.7692	0.6224	0.2755	0.0541	0	0	0	0	1.6991
	Successor Node	13	13	13	13	100	100	100	100	13
5	$u_5(t)$	0.7736	0.6265	0.2774	0.0543	0	0	0	0	1.6989
	Successor Node	14	14	14	14	100	100	100	100	14
6	$u_6(t)$	0.7788	0.6313	0.2794	0.0546	0	0	0	0	1.6987
	Successor Node	15	15	15	15	100	100	100	100	15
7	$u_7(t)$	0.7837	0.6357	0.2814	0.0548	0	0	0	0	1.6987
	Successor Node	16	16	16	16	100	100	100	100	16
8	$u_8(t)$	0.7888	0.6404	0.2834	0.055	0	0	0	0	1.6986
	Successor Node	17	17	17	17	100	100	100	100	17
9	$u_9(t)$	0.7936	0.6448	0.2852	0.0552	0	0	0	0	1.6986
	Successor Node	18	18	18	18	100	100	100	100	18
10	$u_{10}(t)$	0.7985	0.6493	0.2871	0.0554	0	0	0	0	1.6987
	Successor Node	19	19	19	19	100	100	100	100	19
11	$u_{11}(t)$	0.7915	0.648	0.2799	0.0547	0	0	0	0	1.6974
	Successor Node	21	22	12	12	100	100	100	100	12
12	$u_{12}(t)$	0.8362	0.7378	0.4047	0.1068	0.0106	0	0	0	1.5112
	Successor Node	23	23	23	23	23	100	100	100	23

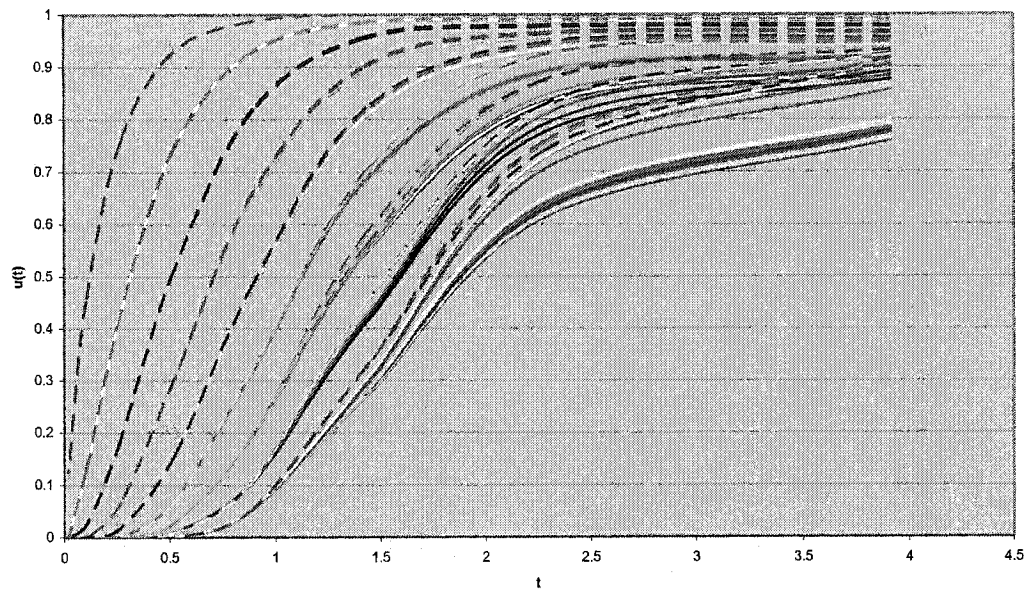
13	$u_{13}(t)$	0.8371	0.7387	0.4053	0.1069	0.0106	0	0	0	1.5111
	Successor Node	23	23	23	23	23	100	100	100	23
14	$u_{14}(t)$	0.838	0.7397	0.4058	0.107	0.0106	0	0	0	1.511
	Successor Node	23	23	23	23	23	100	100	100	23
15	$u_{15}(t)$	0.8401	0.7418	0.407	0.1072	0.0106	0	0	0	1.5109
	Successor Node	24	24	24	24	24	100	100	100	24
16	$u_{16}(t)$	0.8421	0.7438	0.4081	0.1074	0.0106	0	0	0	1.5109
	Successor Node	25	25	25	25	25	100	100	100	25
17	$u_{17}(t)$	0.8447	0.7464	0.4096	0.1076	0.0106	0	0	0	1.5109
	Successor Node	26	26	26	26	26	100	100	100	26
18	$u_{18}(t)$	0.8471	0.7488	0.4109	0.1078	0.0106	0	0	0	1.5109
	Successor Node	27	27	27	27	27	100	100	100	27
19	$u_{19}(t)$	0.8499	0.7516	0.4124	0.108	0.0106	0	0	0	1.511
	Successor Node	28	28	28	28	28	100	100	100	28
20	$u_{20}(t)$	0.8526	0.7543	0.4139	0.1082	0.0106	0	0	0	1.511
	Successor Node	29	29	29	29	29	100	100	100	29
21	$u_{21}(t)$	0.8418	0.684	0.2849	0.0554	0	0	0	0	1.6964
	Successor Node	32	32	12	12	100	100	100	100	12
22	$u_{22}(t)$	0.8441	0.746	0.4096	0.1078	0.0106	0	0	0	1.5102
	Successor Node	23	23	23	23	23	100	100	100	23
23	$u_{23}(t)$	0.8819	0.8259	0.5457	0.1936	0.0288	0	0	0	1.3232
	Successor Node	34	34	34	34	34	100	100	100	34
24	$u_{24}(t)$	0.8824	0.8264	0.5461	0.1937	0.0288	0	0	0	1.3231
	Successor Node	34	34	34	34	34	100	100	100	34
25	$u_{25}(t)$	0.8829	0.827	0.5465	0.1938	0.0288	0	0	0	1.323
	Successor Node	34	34	34	34	34	100	100	100	34
26	$u_{26}(t)$	0.8841	0.8283	0.5474	0.1941	0.0288	0	0	0	1.323
	Successor Node	35	35	35	35	35	100	100	100	35
27	$u_{27}(t)$	0.8852	0.8295	0.5483	0.1943	0.0288	0	0	0	1.323
	Successor Node	36	36	36	36	36	100	100	100	36
28	$u_{28}(t)$	0.8868	0.8312	0.5494	0.1945	0.0287	0	0	0	1.323
	Successor Node	37	37	37	37	37	100	100	100	37
29	$u_{29}(t)$	0.8884	0.8328	0.5506	0.1947	0.0287	0	0	0	1.3231
	Successor Node	38	38	38	38	38	100	100	100	38
30	$u_{30}(t)$	0.8902	0.8347	0.5519	0.195	0.0287	0	0	0	1.3231
	Successor Node	39	39	39	39	39	100	100	100	39
31	$u_{31}(t)$	0.8568	0.7045	0.2929	0.0562	0	0	0	0	1.6966

	Successor Node	41	42	22	22	100	100	100	100	22
32	$u_{32}(t)$	0.8757	0.7756	0.4132	0.1084	0.0106	0	0	0	1.5097
	Successor Node	43	43	23	23	23	100	100	100	23
33	$u_{33}(t)$	0.8864	0.8308	0.5494	0.1946	0.0288	0	0	0	1.3226
	Successor Node	34	34	34	34	34	100	100	100	34
34	$u_{34}(t)$	0.9143	0.8875	0.6838	0.3192	0.0729	0	0	0	1.1349
	Successor Node	45	45	45	45	45	100	100	100	45
35	$u_{35}(t)$	0.9146	0.8878	0.6841	0.3193	0.0729	0	0	0	1.1349
	Successor Node	45	45	45	45	45	100	100	100	45
36	$u_{36}(t)$	0.9149	0.8882	0.6844	0.3195	0.0729	0	0	0	1.1348
	Successor Node	45	45	45	45	45	100	100	100	45
37	$u_{37}(t)$	0.9157	0.889	0.6852	0.3197	0.0729	0	0	0	1.1348
	Successor Node	46	46	46	46	46	100	100	100	46
38	$u_{38}(t)$	0.9165	0.8899	0.6859	0.32	0.0729	0	0	0	1.1348
	Successor Node	47	47	47	47	47	100	100	100	47
39	$u_{39}(t)$	0.9176	0.891	0.6869	0.3203	0.0729	0	0	0	1.1349
	Successor Node	48	48	48	48	48	100	100	100	48
40	$u_{40}(t)$	0.9186	0.8922	0.6879	0.3206	0.0728	0	0	0	1.1349
	Successor Node	49	49	49	49	49	100	100	100	49
41	$u_{41}(t)$	0.885	0.7254	0.2967	0.0572	0	0	0	0	1.6969
	Successor Node	52	52	32	32	100	100	100	100	32
42	$u_{42}(t)$	0.8905	0.7902	0.4198	0.1092	0.0106	0	0	0	1.5098
	Successor Node	53	53	33	33	33	100	100	100	33
43	$u_{43}(t)$	0.9001	0.8453	0.5523	0.1954	0.0287	0	0	0	1.3222
	Successor Node	54	54	34	34	34	100	100	100	34
44	$u_{44}(t)$	0.9172	0.8907	0.6867	0.3204	0.0729	0	0	0	1.1345
	Successor Node	45	45	45	45	45	100	100	100	45
45	$u_{45}(t)$	0.9391	0.9288	0.8036	0.4796	0.1585	0.0202	0	0	0.9464
	Successor Node	56	56	56	56	56	56	100	100	56
46	$u_{46}(t)$	0.9393	0.929	0.8039	0.4798	0.1586	0.0202	0	0	0.9463
	Successor Node	56	56	56	56	56	56	100	100	56
47	$u_{47}(t)$	0.9395	0.9292	0.8042	0.4799	0.1586	0.0202	0	0	0.9463
	Successor Node	56	56	56	56	56	56	100	100	56
48	$u_{48}(t)$	0.9401	0.9299	0.8048	0.4803	0.1586	0.0201	0	0	0.9463
	Successor Node	57	57	57	57	57	57	100	100	57
49	$u_{49}(t)$	0.9407	0.9305	0.8054	0.4806	0.1586	0.0201	0	0	0.9463
	Successor	58	58	58	58	58	58	100	100	58

	Node									
50	$u_{50}(t)$	0.9415	0.9313	0.8063	0.481	0.1586	0.0201	0	0	0.9463
	Successor Node	59	59	59	59	59	59	100	100	59
51	$u_{51}(t)$	0.9056	0.7277	0.3028	0.0576	0	0	0	0	1.6976
	Successor Node	62	52	42	42	100	100	100	100	42
52	$u_{52}(t)$	0.9045	0.8089	0.4239	0.1101	0.0103	0	0	0	1.51
	Successor Node	63	63	43	43	43	100	100	100	43
53	$u_{53}(t)$	0.9096	0.8554	0.5582	0.1966	0.0286	0	0	0	1.3224
	Successor Node	64	64	44	44	44	100	100	100	44
54	$u_{54}(t)$	0.9196	0.8934	0.6892	0.3213	0.0729	0	0	0	1.1342
	Successor Node	45	45	45	45	45	100	100	100	45
55	$u_{55}(t)$	0.9412	0.931	0.8061	0.4811	0.1587	0.0201	0	0	0.9461
	Successor Node	56	56	56	56	56	56	100	100	56
56	$u_{56}(t)$	0.9588	0.956	0.8939	0.6556	0.3056	0.0694	0	0	0.7575
	Successor Node	67	67	67	67	67	67	100	100	67
57	$u_{57}(t)$	0.959	0.9562	0.8941	0.6558	0.3056	0.0693	0	0	0.7575
	Successor Node	67	67	67	67	67	67	100	100	67
58	$u_{58}(t)$	0.9592	0.9564	0.8943	0.6559	0.3057	0.0693	0	0	0.7575
	Successor Node	67	67	67	67	67	67	100	100	67
59	$u_{59}(t)$	0.9596	0.9568	0.8948	0.6563	0.3057	0.0693	0	0	0.7575
	Successor Node	68	68	68	68	68	68	100	100	68
60	$u_{60}(t)$	0.96	0.9573	0.8953	0.6567	0.3058	0.0692	0	0	0.7575
	Successor Node	69	69	69	69	69	69	100	100	69
61	$u_{61}(t)$	0.906	0.7404	0.3276	0.0582	0	0	0	0	1.6982
	Successor Node	72	72	72	52	100	100	100	100	52
62	$u_{62}(t)$	0.9233	0.8108	0.4299	0.1106	0.0101	0	0	0	1.5106
	Successor Node	73	63	53	53	53	100	100	100	53
63	$u_{63}(t)$	0.9186	0.8724	0.5627	0.1974	0.0284	0	0	0	1.3225
	Successor Node	74	74	54	54	54	100	100	100	54
64	$u_{64}(t)$	0.9254	0.8996	0.6946	0.3231	0.0728	0	0	0	1.1343
	Successor Node	55	55	55	55	55	100	100	100	55
65	$u_{65}(t)$	0.943	0.9329	0.8082	0.4823	0.1588	0.02	0	0	0.9458
	Successor Node	56	56	56	56	56	56	100	100	56
66	$u_{66}(t)$	0.9604	0.9577	0.8958	0.6572	0.306	0.0692	0	0	0.7573
	Successor Node	67	67	67	67	67	67	100	100	67
67	$u_{67}(t)$	0.9749	0.9745	0.952	0.8149	0.5147	0.1905	0.0263	0	0.5684
	Successor Node	78	78	78	78	78	78	78	100	78

68	$u_{68}(t)$	0.975	0.9746	0.9522	0.8151	0.5148	0.1905	0.0263	0	0.5684
	Successor Node	78	78	78	78	78	78	78	100	78
69	$u_{69}(t)$	0.9752	0.9748	0.9523	0.8153	0.5149	0.1905	0.0263	0	0.5684
	Successor Node	78	78	78	78	78	78	78	100	78
70	$u_{70}(t)$	0.9755	0.9751	0.9527	0.8157	0.5151	0.1904	0.0262	0	0.5684
	Successor Node	79	79	79	79	79	79	79	100	79
71	$u_{71}(t)$	0.9088	0.7566	0.3282	0.0577	0	0	0	0	1.6991
	Successor Node	62	82	72	62	100	100	100	100	62
72	$u_{72}(t)$	0.9217	0.8123	0.451	0.1113	0.0102	0	0	0	1.511
	Successor Node	83	63	83	63	83	100	100	100	63
73	$u_{73}(t)$	0.9367	0.8738	0.5689	0.1984	0.0282	0	0	0	1.3229
	Successor Node	84	74	64	64	64	100	100	100	64
74	$u_{74}(t)$	0.93	0.9153	0.6989	0.3244	0.0725	0	0	0	1.1344
	Successor Node	65	85	65	65	65	100	100	100	65
75	$u_{75}(t)$	0.9473	0.9376	0.8129	0.4847	0.1587	0.0198	0	0	0.9459
	Successor Node	66	66	66	66	66	66	100	100	66
76	$u_{76}(t)$	0.9618	0.9591	0.8975	0.6586	0.3063	0.0691	0	0	0.7571
	Successor Node	67	67	67	67	67	67	100	100	67
77	$u_{77}(t)$	0.9761	0.9758	0.9535	0.8166	0.5156	0.1903	0.0261	0	0.5682
	Successor Node	78	78	78	78	78	78	78	100	78
78	$u_{78}(t)$	0.9882	0.9881	0.984	0.9285	0.7469	0.4293	0.1309	0	0.3791
	Successor Node	89	89	89	89	89	89	89	100	89
79	$u_{79}(t)$	0.9883	0.9882	0.9841	0.9287	0.747	0.4293	0.1308	0	0.3791
	Successor Node	89	89	89	89	89	89	89	100	89
80	$u_{80}(t)$	0.9884	0.9883	0.9843	0.9288	0.7471	0.4293	0.1308	0	0.3791
	Successor Node	89	89	89	89	89	89	89	100	89
81	$u_{81}(t)$	0.9086	0.7537	0.3287	0.0561	0	0	0	0	1.6998
	Successor Node	72	92	72	72	100	100	100	100	72
82	$u_{82}(t)$	0.9258	0.8289	0.4517	0.111	0.0102	0	0	0	1.5117
	Successor Node	73	93	83	73	83	100	100	100	73
83	$u_{83}(t)$	0.933	0.8751	0.5833	0.1995	0.0276	0	0	0	1.3233
	Successor Node	94	74	94	74	74	100	100	100	74
84	$u_{84}(t)$	0.9469	0.9229	0.705	0.3261	0.0721	0	0	0	1.1348
	Successor Node	95	95	75	75	75	100	100	100	75
85	$u_{85}(t)$	0.9509	0.9414	0.8169	0.4866	0.1586	0.0195	0	0	0.946
	Successor Node	76	76	76	76	76	76	100	100	76
86	$u_{86}(t)$	0.9652	0.9627	0.9016	0.6616	0.3067	0.0685	0	0	0.7572

	Successor Node	77	77	77	77	77	77	100	100	77
87	$u_{87}(t)$	0.9772	0.9769	0.9548	0.818	0.5163	0.1902	0.026	0	0.5681
	Successor Node	78	78	78	78	78	78	78	100	78
88	$u_{88}(t)$	0.9892	0.9891	0.9851	0.93	0.7482	0.4295	0.1304	0	0.379
	Successor Node	89	89	89	89	89	89	89	100	89
89	$u_{89}(t)$	0.9994	0.9988	1.0004	0.9874	0.9289	0.7583	0.456	0.1233	0.1897
	Successor Node	100	100	100	100	100	100	100	100	100
90	$u_{90}(t)$	0.9994	0.9988	1.0004	0.9874	0.929	0.7584	0.4559	0.1231	0.1896
	Successor Node	100	100	100	100	100	100	100	100	100
91	$u_{91}(t)$	0.9165	0.7587	0.3284	0.0564	0	0	0	0	1.7007
	Successor Node	82	82	82	82	100	100	100	100	82
92	$u_{92}(t)$	0.9237	0.8229	0.4523	0.1102	0.0101	0	0	0	1.5123
	Successor Node	83	83	83	83	83	100	100	100	83
93	$u_{93}(t)$	0.9387	0.8841	0.5791	0.2001	0.0273	0	0	0	1.3238
	Successor Node	84	84	84	84	84	100	100	100	84
94	$u_{94}(t)$	0.9419	0.9175	0.71	0.3274	0.0717	0	0	0	1.1351
	Successor Node	85	85	85	85	85	100	100	100	85
95	$u_{95}(t)$	0.9561	0.947	0.8226	0.4893	0.1582	0.0192	0	0	0.9463
	Successor Node	86	86	86	86	86	86	100	100	86
96	$u_{96}(t)$	0.9681	0.9657	0.905	0.6642	0.3069	0.068	0	0	0.7572
	Successor Node	87	87	87	87	87	87	100	100	87
97	$u_{97}(t)$	0.98	0.9797	0.9581	0.8214	0.5176	0.1894	0.0255	0	0.5681
	Successor Node	88	88	88	88	88	88	88	100	88
98	$u_{98}(t)$	0.9908	0.99	0.9862	0.9313	0.7493	0.4296	0.1299	0	0.3788
	Successor Node	99	89	89	89	89	89	89	100	89
99	$u_{99}(t)$	0.9994	0.9988	1.0005	0.9877	0.9296	0.7586	0.4544	0.1216	0.1895
	Successor Node	100	100	100	100	100	100	100	100	100
100	$u_{100}(t)$	1	1	1	1	1	1	1	1	0



**Figure F-1: Nine Clusters of the Maximum Probability Functions of On Time
Arrival in the First 100-Node Network**

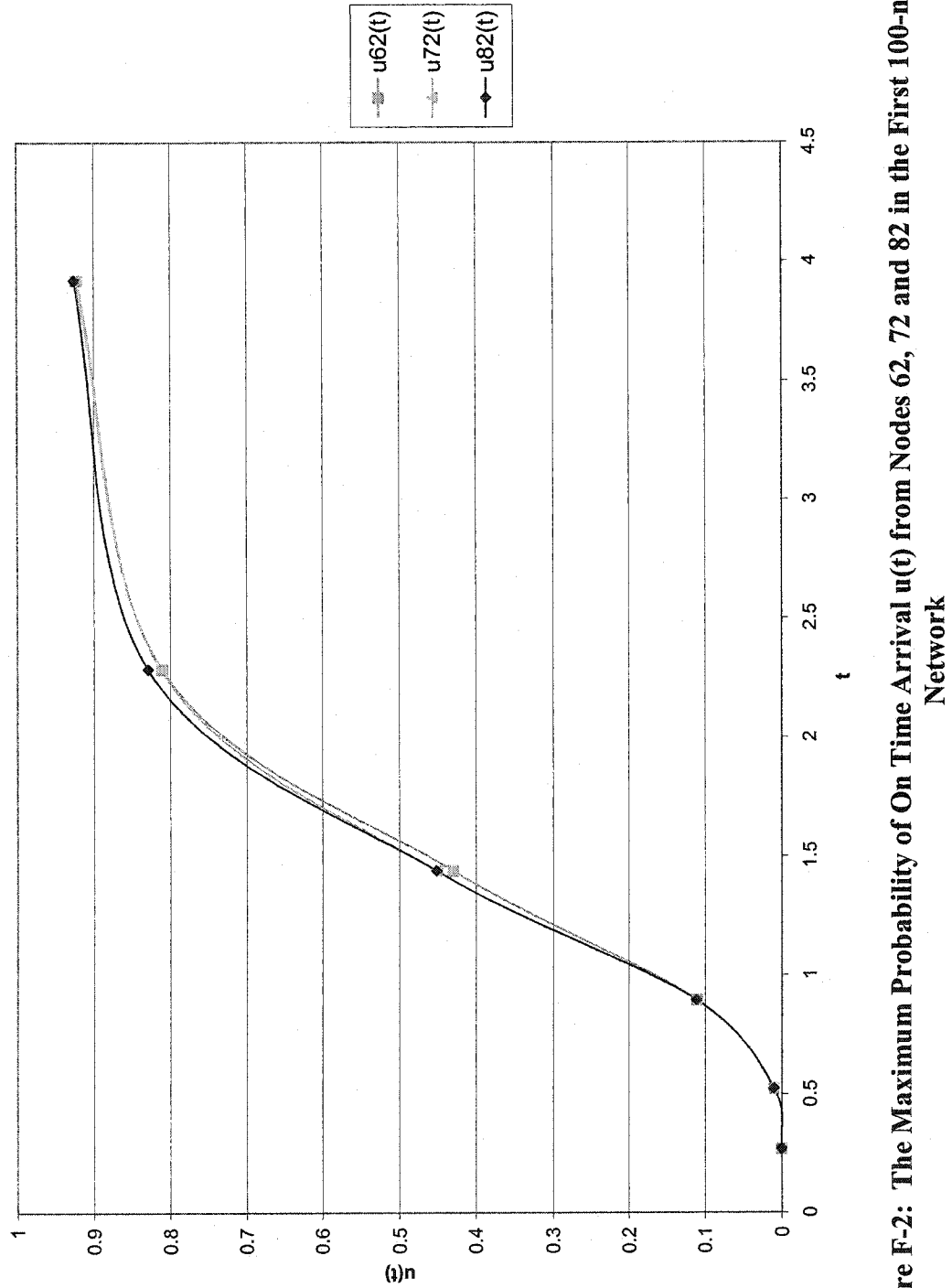


Figure F-2: The Maximum Probability of On Time Arrival $u(t)$ from Nodes 62, 72 and 82 in the First 100-node Network

APPENDIX G: SOTA PROBLEM TESTED WITH A 100-NODE NETWORK WITH THE SECOND SET OF GAMMA DISTRIBUTIONS

The parameters in gamma distribution of link travel times over link ij are

$$n = 0.2 \cdot \log_{10}(10 + 4 \cdot i + 6 \cdot j)$$

and

$$\text{alfa} = \log_{10}(2 + 6 \cdot i + 10 \cdot j).$$

From Node	SOTA	t1= 3.9193	t2= 2.2861	t3= 1.4387	t4= 0.8958	t5= 0.5247	t6= 0.2708	t7= 0.1072	t8= 0.0201	average- shortest- path cost & next nodes
1	$u_1(t)$	0.4528	0.3484	0.1621	0.0422	0.0048	0.0003	0	0	1.6741
	Successor Node	12	12	12	12	12	12	unidentified	unidentified	12
2	$u_2(t)$	0.4598	0.3542	0.1648	0.0428	0.0048	0.0004	0	0	1.6743
	Successor Node	13	13	13	13	12	unidentified	unidentified	unidentified	12
3	$u_3(t)$	0.4661	0.3595	0.1673	0.0433	0.0048	0.0003	0	0	1.6744
	Successor Node	14	14	14	14	12	unidentified	unidentified	unidentified	12
4	$u_4(t)$	0.4717	0.3642	0.1694	0.0437	0.0048	0.0003	0	0	1.6745
	Successor Node	15	15	15	15	13	unidentified	unidentified	unidentified	13
5	$u_5(t)$	0.4768	0.3685	0.1714	0.0442	0.0048	0.0003	0	0	1.6747
	Successor Node	16	16	16	16	14	unidentified	unidentified	unidentified	14
6	$u_6(t)$	0.4812	0.3722	0.1731	0.0445	0.0049	0.0003	0	0	1.6749
	Successor Node	17	17	17	17	15	unidentified	unidentified	unidentified	15
7	$u_7(t)$	0.485	0.3754	0.1746	0.0448	0.0049	0.0003	0	0	1.6751
	Successor Node	18	18	18	18	16	unidentified	unidentified	unidentified	16
8	$u_8(t)$	0.4881	0.3779	0.1757	0.0451	0.0049	0.0003	0	0	1.6754
	Successor Node	19	19	19	19	17	unidentified	unidentified	unidentified	17
9	$u_9(t)$	0.4902	0.3797	0.1766	0.0453	0.0049	0.0004	0	0	1.6756
	Successor Node	20	20	20	20	18	unidentified	unidentified	unidentified	18
10	$u_{10}(t)$	0.4909	0.3803	0.1768	0.0453	0.005	0.0004	0	0	1.6759
	Successor Node	20	20	20	20	19	unidentified	unidentified	unidentified	19
11	$u_{11}(t)$	0.5053	0.3924	0.1823	0.0464	0.0051	0.0004	0	0	1.6752
	Successor Node	22	22	22	22	22	unidentified	unidentified	unidentified	12
12	$u_{12}(t)$	0.5446	0.4591	0.2564	0.0838	0.0136	0.0007	0	0	1.4905
	Successor Node	23	23	23	23	23	unidentified	unidentified	unidentified	23
13	$u_{13}(t)$	0.5484	0.4626	0.2584	0.0843	0.0136	0.0007	0	0	1.4906
	Successor	24	24	24	24	23	unidentified	unidentified	unidentified	23

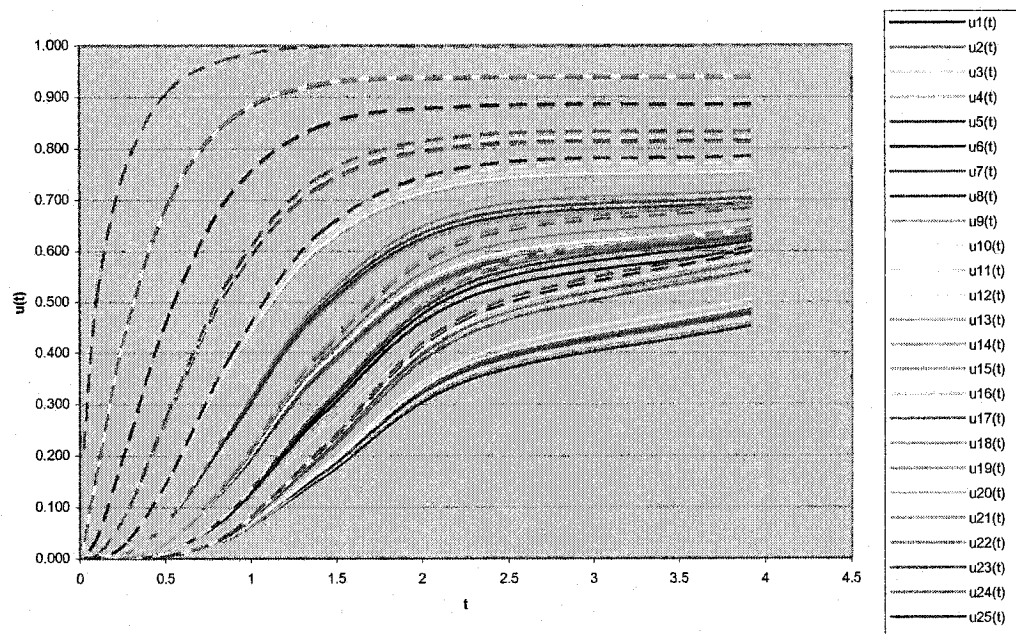
	Node									
	$u_{14}(t)$	0.552	0.4659	0.2602	0.0849	0.0136	0.0007	0	0	1.4907
14	Successor Node	25	25	25	25	23	unidentified	unidentified	unidentified	23
	$u_{15}(t)$	0.5552	0.4688	0.2619	0.0853	0.0136	0.0007	0	0	1.4908
15	Successor Node	26	26	26	26	24	unidentified	unidentified	unidentified	24
	$u_{16}(t)$	0.5581	0.4715	0.2634	0.0857	0.0137	0.0007	0	0	1.4909
16	Successor Node	27	27	27	27	25	unidentified	unidentified	unidentified	25
	$u_{17}(t)$	0.5605	0.4737	0.2646	0.086	0.0137	0.0007	0	0	1.4911
17	Successor Node	28	28	28	28	26	unidentified	unidentified	unidentified	26
	$u_{18}(t)$	0.5625	0.4755	0.2656	0.0863	0.0137	0.0007	0	0	1.4912
18	Successor Node	29	29	29	29	27	unidentified	unidentified	unidentified	27
	$u_{19}(t)$	0.5638	0.4766	0.2663	0.0865	0.0138	0.0007	0	0	1.4914
19	Successor Node	30	30	30	30	28	unidentified	unidentified	unidentified	28
	$u_{20}(t)$	0.5642	0.477	0.2665	0.0865	0.0138	0.0007	0	0	1.4916
20	Successor Node	30	30	30	30	29	unidentified	unidentified	unidentified	29
	$u_{21}(t)$	0.5381	0.4201	0.1948	0.0489	0.0052	0.0004	0	0	1.676
21	Successor Node	32	32	32	32	32	unidentified	unidentified	unidentified	12
	$u_{22}(t)$	0.5776	0.4894	0.2734	0.0883	0.014	0.0007	0	0	1.4912
22	Successor Node	33	33	33	33	33	unidentified	unidentified	unidentified	23
	$u_{23}(t)$	0.6186	0.5577	0.366	0.1505	0.0336	0.0028	0	0	1.3059
23	Successor Node	34	34	34	34	34	34	unidentified	unidentified	34
	$u_{24}(t)$	0.6213	0.5603	0.3677	0.1511	0.0336	0.0028	0	0	1.3059
24	Successor Node	35	35	35	35	34	34	unidentified	unidentified	34
	$u_{25}(t)$	0.6238	0.5627	0.3694	0.1517	0.0337	0.0028	0	0	1.306
25	Successor Node	36	36	36	36	36	34	unidentified	unidentified	34
	$u_{26}(t)$	0.626	0.5648	0.3708	0.1521	0.0337	0.0028	0	0	1.3061
26	Successor Node	37	37	37	37	35	35	unidentified	unidentified	35
	$u_{27}(t)$	0.6279	0.5666	0.3721	0.1526	0.0337	0.0028	0	0	1.3062
27	Successor Node	38	38	38	38	36	36	unidentified	unidentified	36
	$u_{28}(t)$	0.6293	0.568	0.373	0.1529	0.0338	0.0028	0	0	1.3063
28	Successor Node	39	39	39	39	37	37	unidentified	unidentified	37
	$u_{29}(t)$	0.6302	0.569	0.3736	0.1531	0.0338	0.0028	0	0	1.3064
29	Successor Node	40	40	40	40	38	38	unidentified	unidentified	38
	$u_{30}(t)$	0.6305	0.5692	0.3738	0.1532	0.0339	0.0028	0	0	1.3065
30	Successor Node	40	40	40	40	39	39	unidentified	unidentified	39

	$u_{31}(t)$	0.5613	0.4398	0.2036	0.0504	0.0053	0.0004	0	0	1.677
31	Successor Node	42	42	42	42	42	unidentified	unidentified	unidentified	22
	$u_{32}(t)$	0.6017	0.5115	0.2858	0.0915	0.0142	0.0007	0	0	1.4917
32	Successor Node	43	43	43	43	43	unidentified	unidentified	unidentified	23
	$u_{33}(t)$	0.6427	0.5812	0.3819	0.1559	0.0343	0.0028	0	0	1.3063
33	Successor Node	44	44	44	44	44	34	unidentified	unidentified	34
	$u_{34}(t)$	0.686	0.6478	0.4874	0.2491	0.0753	0.01	0.0003	0	1.1205
34	Successor Node	45	45	45	45	45	45	unidentified	unidentified	45
	$u_{35}(t)$	0.688	0.6498	0.489	0.2498	0.0754	0.01	0.0003	0	1.1205
35	Successor Node	46	46	46	46	46	45	unidentified	unidentified	45
	$u_{36}(t)$	0.6898	0.6517	0.4905	0.2505	0.0755	0.01	0.0003	0	1.1205
36	Successor Node	47	47	47	47	47	45	unidentified	unidentified	45
	$u_{37}(t)$	0.6913	0.6532	0.4917	0.251	0.0755	0.01	0.0003	0	1.1206
37	Successor Node	48	48	48	48	47	46	unidentified	unidentified	46
	$u_{38}(t)$	0.6924	0.6543	0.4926	0.2514	0.0757	0.01	0.0003	0	1.1207
38	Successor Node	49	49	49	49	49	47	unidentified	unidentified	47
	$u_{39}(t)$	0.6932	0.6551	0.4932	0.2517	0.0756	0.01	0.0003	0	1.1208
39	Successor Node	50	50	50	50	48	48	unidentified	unidentified	48
	$u_{40}(t)$	0.6934	0.6553	0.4934	0.2518	0.0757	0.01	0.0003	0	1.1208
40	Successor Node	50	50	50	50	49	49	unidentified	unidentified	49
	$u_{41}(t)$	0.5783	0.4542	0.21	0.0515	0.0053	0.0005	0	0	1.6778
41	Successor Node	52	52	52	52	52	52	unidentified	unidentified	32
	$u_{42}(t)$	0.6193	0.5278	0.2948	0.0937	0.0144	0.0007	0	0	1.4925
42	Successor Node	53	53	53	53	53	53	unidentified	unidentified	33
	$u_{43}(t)$	0.661	0.5991	0.394	0.16	0.0348	0.0028	0	0	1.3067
43	Successor Node	54	54	54	54	54	34	unidentified	unidentified	34
	$u_{44}(t)$	0.7041	0.6662	0.5021	0.2557	0.0765	0.01	0.0003	0	1.1208
44	Successor Node	55	55	55	55	55	45	unidentified	unidentified	45
	$u_{45}(t)$	0.7501	0.7295	0.6124	0.3812	0.1537	0.0316	0.0018	0	0.9345
45	Successor Node	56	56	56	56	56	56	56	unidentified	56
	$u_{46}(t)$	0.7516	0.731	0.6137	0.382	0.1539	0.0316	0.0018	0	0.9345
46	Successor Node	57	57	57	57	57	56	56	unidentified	56
	$u_{47}(t)$	0.7529	0.7324	0.6149	0.3827	0.1541	0.0316	0.0018	0	0.9346
47	Successor Node	58	58	58	58	58	56	56	unidentified	56
48	$u_{48}(t)$	0.7539	0.7334	0.6159	0.3832	0.1543	0.0316	0.0018	0	0.9346

	Successor Node	59	59	59	59	59	57	57 unidentified	57
	$u_{49}(t)$	0.7545	0.734	0.6164	0.3835	0.1543	0.0316	0.0018	0 0.9347
49	Successor Node	60	60	60	60	59	58	58 unidentified	58
	$u_{50}(t)$	0.7547	0.7342	0.6166	0.3836	0.1543	0.0316	0.0018	0 0.9347
50	Successor Node	60	60	60	60	59	59	59 unidentified	59
	$u_{51}(t)$	0.5906	0.4647	0.2146	0.0523	0.0053	0.0005	0	0 1.679
51	Successor Node	62	62	62	62	42	61 unidentified	unidentified	42
	$u_{52}(t)$	0.6321	0.5397	0.3014	0.0953	0.0145	0.0007	0	0 1.4932
52	Successor Node	63	63	63	63	63	43 unidentified	unidentified	43
	$u_{53}(t)$	0.6743	0.6122	0.4029	0.163	0.035	0.0028	0.0001	0.0001 1.3073
53	Successor Node	64	64	64	64	64	44 unidentified	unidentified	44
	$u_{54}(t)$	0.718	0.6803	0.5134	0.2607	0.0773	0.01	0.0003	0 1.1211
54	Successor Node	65	65	65	65	65	45 unidentified	unidentified	45
	$u_{55}(t)$	0.7637	0.7435	0.6251	0.3886	0.1557	0.0316	0.0018	0 0.9348
55	Successor Node	66	66	66	66	66	56	56 unidentified	56
	$u_{56}(t)$	0.8128	0.8035	0.7311	0.5391	0.2844	0.0882	0.0101	0 0.7482
56	Successor Node	67	67	67	67	67	67	67 unidentified	67
	$u_{57}(t)$	0.8138	0.8045	0.7322	0.5399	0.2847	0.0882	0.0101	0 0.7482
57	Successor Node	68	68	68	68	68	67	67 unidentified	67
	$u_{58}(t)$	0.8147	0.8054	0.733	0.5406	0.2849	0.0882	0.0101	0 0.7482
58	Successor Node	69	69	69	69	69	67	67 unidentified	67
	$u_{59}(t)$	0.8152	0.806	0.7336	0.541	0.2851	0.0882	0.0101	0 0.7482
59	Successor Node	70	70	70	70	70	68	68 unidentified	68
	$u_{60}(t)$	0.8154	0.8061	0.7338	0.5411	0.2851	0.0882	0.0101	0 0.7483
60	Successor Node	70	70	70	70	70	69	69 unidentified	69
	$u_{61}(t)$	0.5993	0.4721	0.2179	0.053	0.0053	0.0005	0	0 1.6799
61	Successor Node	72	72	72	72	52 unidentified	unidentified	unidentified	52
	$u_{62}(t)$	0.6411	0.5481	0.306	0.0963	0.0146	0.0007	0	0 1.4941
62	Successor Node	73	73	73	73	73 unidentified	unidentified	unidentified	53
	$u_{63}(t)$	0.6837	0.6214	0.4091	0.165	0.0352	0.0028	0.0001	0 1.3078
63	Successor Node	74	74	74	74	74	54 unidentified	unidentified	54
	$u_{64}(t)$	0.7277	0.6902	0.5213	0.2643	0.0779	0.0099	0.0003	0 1.1216
64	Successor Node	75	75	75	75	75	55 unidentified	unidentified	55
	$u_{65}(t)$	0.7738	0.7539	0.6346	0.3942	0.1572	0.0316	0.0018	0 0.935
65	Successor Node	76	76	76	76	76	56	56 unidentified	56

	$u_{66}(t)$	0.8224	0.8134	0.7411	0.5465	0.2872	0.0884	0.0101	0	0.7484
66	Successor Node	77	77	77	77	77	77	67 unidentified	67	
	$u_{67}(t)$	0.8748	0.8715	0.836	0.7054	0.4719	0.2144	0.0476	0	0.5615
67	Successor Node	78	78	78	78	78	78	78 unidentified	78	
	$u_{68}(t)$	0.8755	0.8721	0.8367	0.7061	0.4723	0.2144	0.0476	0	0.5615
68	Successor Node	79	79	79	79	79	78	78 unidentified	78	
	$u_{69}(t)$	0.876	0.8726	0.8372	0.7065	0.4726	0.2145	0.0476	0	0.5615
69	Successor Node	80	80	80	80	80	78	78 unidentified	78	
	$u_{70}(t)$	0.8761	0.8728	0.8373	0.7067	0.4726	0.2145	0.0476	0	0.5615
70	Successor Node	80	80	80	80	80	79	79 unidentified	79	
	$u_{71}(t)$	0.6049	0.4769	0.22	0.0533	0.0054	0.0006	0	0	1.681
71	Successor Node	82	82	82	82	62	81 unidentified	unidentified	62	
	$u_{72}(t)$	0.647	0.5535	0.309	0.097	0.0146	0.0007	0	0	1.4948
72	Successor Node	83	83	83	83	83	63 unidentified	unidentified	63	
	$u_{73}(t)$	0.6898	0.6274	0.4132	0.1663	0.0353	0.0027	0.0001	0	1.3086
73	Successor Node	84	84	84	84	84	64 unidentified	unidentified	64	
	$u_{74}(t)$	0.7341	0.6966	0.5264	0.2665	0.0783	0.0099	0.0003	0	1.122
74	Successor Node	85	85	85	85	85	65 unidentified	unidentified	65	
	$u_{75}(t)$	0.7804	0.7608	0.6408	0.3978	0.1581	0.0316	0.0018	0	0.9354
75	Successor Node	86	86	86	86	86	66	66 unidentified	66	
	$u_{76}(t)$	0.8293	0.8205	0.7482	0.5517	0.2893	0.0886	0.0101	0	0.7485
76	Successor Node	87	87	87	87	87	87	67 unidentified	67	
	$u_{77}(t)$	0.881	0.8778	0.8427	0.7115	0.4752	0.2149	0.0476	0	0.5616
77	Successor Node	88	88	88	88	88	88	78 unidentified	78	
	$u_{78}(t)$	0.9368	0.9358	0.924	0.8574	0.6969	0.4447	0.1794	0.0204	0.3745
78	Successor Node	89	89	89	89	89	89	89	89	89
	$u_{79}(t)$	0.9372	0.9362	0.9244	0.8577	0.6972	0.4447	0.1794	0.0204	0.3745
79	Successor Node	90	90	90	90	90	89	89	89	89
	$u_{80}(t)$	0.9373	0.9363	0.9245	0.8579	0.6973	0.4448	0.1794	0.0203	0.3746
80	Successor Node	90	90	90	90	90	89	89	89	89
	$u_{81}(t)$	0.6079	0.4794	0.2211	0.0535	0.0053	0.0005	0	0	1.682
81	Successor Node	92	92	92	92	72 unidentified	unidentified	unidentified	72	
	$u_{82}(t)$	0.6501	0.5564	0.3106	0.0974	0.0146	0.0007	0	0	1.4958
82	Successor Node	93	93	93	93	73 unidentified	unidentified	unidentified	73	
83	$u_{83}(t)$	0.6931	0.6306	0.4153	0.167	0.0354	0.0027	0.0001	0	1.3092

	Successor Node	94	94	94	94	94	74 unidentified	unidentified	74
	$u_{84}(t)$	0.7375	0.7001	0.5292	0.2677	0.0785	0.0099	0.0003	0 1.1226
84	Successor Node	95	95	95	95	95	75 unidentified	unidentified	75
	$u_{85}(t)$	0.784	0.7644	0.6442	0.3997	0.1586	0.0316	0.0018	0 0.9358
85	Successor Node	96	96	96	96	96	76	76 unidentified	76
	$u_{86}(t)$	0.8331	0.8244	0.752	0.5545	0.2904	0.0886	0.0101	0 0.7489
86	Successor Node	97	97	97	97	97	87	77 unidentified	77
	$u_{87}(t)$	0.8849	0.8818	0.8469	0.7152	0.4774	0.2153	0.0475	0.0004 0.5618
87	Successor Node	98	98	98	98	98	98	78 unidentified	78
	$u_{88}(t)$	0.9399	0.9389	0.9273	0.8608	0.6995	0.4453	0.1793	0.0203 0.3747
88	Successor Node	99	99	99	99	99	99	89	89 89
	$u_{89}(t)$	0.9997	0.9981	0.9986	0.9759	0.9133	0.7635	0.5257	0.2016 0.1874
89	Successor Node	100	100	100	100	100	100	100	100
	$u_{90}(t)$	0.9997	0.9981	0.9986	0.976	0.9134	0.7635	0.5256	0.2015 0.1874
90	Successor Node	100	100	100	100	100	100	100	100
	$u_{91}(t)$	0.6087	0.4801	0.2214	0.0535	0.0054	0.0005	0	0 1.6831
91	Successor Node	92	92	92	92	82 unidentified	unidentified	unidentified	82
	$u_{92}(t)$	0.651	0.5572	0.311	0.0975	0.0147	0.0007	0	0 1.4966
92	Successor Node	93	93	93	93	83 unidentified	unidentified	unidentified	83
	$u_{93}(t)$	0.694	0.6315	0.4159	0.1672	0.0354	0.0027	0.0001	0 1.31
93	Successor Node	94	94	94	94	94	84 unidentified	unidentified	84
	$u_{94}(t)$	0.7384	0.7011	0.5299	0.268	0.0785	0.0099	0.0003	0 1.1232
94	Successor Node	95	95	95	95	95	85 unidentified	unidentified	85
	$u_{95}(t)$	0.785	0.7654	0.6451	0.4002	0.1587	0.0316	0.0018	0 0.9363
95	Successor Node	96	96	96	96	96	86	86 unidentified	86
	$u_{96}(t)$	0.8341	0.8254	0.753	0.5553	0.2906	0.0886	0.01	0 0.7492
96	Successor Node	97	97	97	97	97	87	87 unidentified	87
	$u_{97}(t)$	0.886	0.8829	0.848	0.7162	0.4779	0.2154	0.0473	0.0004 0.5621
97	Successor Node	98	98	98	98	98	98	88 unidentified	88
	$u_{98}(t)$	0.941	0.94	0.9285	0.862	0.7004	0.4456	0.1791	0.0203 0.3748
98	Successor Node	99	99	99	99	99	99	89	89 89
	$u_{99}(t)$	0.9997	0.9981	0.9987	0.9761	0.9135	0.7633	0.5246	0.2002 0.1875
99	Successor Node	100	100	100	100	100	100	100	100
100	$u_{100}(t)$	1	1	1	1	1	1	1	1 0



**Figure G-1: Nine Clusters of the Maximum Probability Functions of On Time
Arrival in the Second 100-Node Network**

APPENDIX H: A MATLAB PROGRAM FOR SOLVING THE ARRIVING- ON-TIME PROBLEM WITH EVALUATING CONVOLUTION INTEGRAL VIA THE METHOD OF DIFFERENTIAL APPROXIMATION

```

function w = myode(alfa,init_t, dt, NumberOfPoints, u)
n = NumberOfPoints - 1;
w(1)=0;
v(1)=0;

%u = zeros(101);
%u = ones(101);
t(1) = init_t;
for i = 1:100
    t(i+1) = t(i) + dt;
end

for i = 1: n
    dw1 = v(i);
    dv1 = -2*alfa*v(i)-alfa^2*w(i)+alfa^2*u(i);
    w(i+1) = w(i) + dw1 * dt;
    v(i+1) = v(i) + dv1 * dt;
    dw2 = v(i+1);
    dv2 = -2*alfa*v(i+1)-alfa^2*w(i+1)+alfa^2*u(i+1);
    dw = (dw1+dw2)/2;
    dv = (dv1+dv2)/2;
    w(i+1) = w(i) + dw * dt;
    v(i+1) = v(i) + dv * dt;
end

%pathgamma8r4n.m
%clear all
%diary('Output8r4n.txt')

%input number of t at where the probabilities are to be evaluated
M = 500

%input the parameters of Gamma distribution for each link
alfa = [0 2 3 1; 2 0 1 2; 3 1 0 3; 1 2 3 0]
n = [0 2 2 2; 2 0 2 2; 2 2 0 2; 2 2 2 0]

%input number of nodes N in this network
N=4

%the probability of starting at node N and
%arriving at node N in time t is always 1
for t=1:M
    u(1,t)=0;
    u(2,t)=0;
    u(3,t)=0;

```

```

    u(N,t)=1;
end

%Start Successive Approximation
for i=1:(N-1);
    next_node(i,:)=ones(1,M)*N;
end

for k=1:N
    k
    for i=1:(N-1)
        i
        max=u(i,:);
        for j=1:N %j could be a set of possible next nodes from i
            j
            if (j ~= i)
                [theu]=myodealfa(alfa(i,j),0,0.01,M,u(j,:));
                for t=1:M
                    t;
                    if theu(t)>max(t)
                        max(t)=theu(t);
                        next_node(i,t)=j;
                    end
                end
            end
        end
        u(i,:)=max;
    end
end

t=0:0.01:4.99

disp('the output every 0.1 step')
for i=1:50
    theTT(i)=t(i*10);
    theUU(1,i)=u(1,i*10);
    theUU(2,i)=u(2,i*10);
    theUU(3,i)=u(3,i*10);
end
theTT
theUU

plot(t,u(1,:), 'r-',t,u(2,:), 'y-',t,u(3,:), 'b-')

%diary off

```

APPENDIX I: A MATLAB PROGRAM FOR FINDING THE OPTIMAL ROUTING THROUGH THE 4-NODE NETWORK IN THE CASE OF CORRELATED LINK TRAVEL TIMES

```

% expected shortest path with correlation in travel time for a 4-
% node network
% Yueyue Fan 10/9/01

clear all
% input
N=4
connect = [1 1 1 0; 1 1 1 1; 1 1 1 1; 0 1 1 1];
tp = [0 1 4 999999; 1 0 2 5; 4 2 0 7; 999999 5 7 0];
tq = 2*tp;
alfa = 2/3;
lamda = 1/3;
% initialize
for i = 1:N-1
    u(i) = 999999;
    v(i) = 999999;
end
u(N) = 0;
v(N) = 0;
% iteration
for k = 1:N
    for i = 1:N-1
        min_u(i) = 999999;
        min_v(i) = 999999;
        for j = 1:N
            if ((j ~= i) & (connect(i,j) ~= 0))
                the_u = tp(i,j) + alfa*u(j) + (1-alfa)*v(j);
                the_v = tq(i,j) + lamda*u(j) + (1-lamda)*v(j);
                if (the_u < min_u(i))
                    min_u(i) = the_u;
                    the_ju(i) = j;
                end
                if the_v < min_v(i)
                    min_v(i) = the_v;
                    the_jv(i) = j;
                end
            end
        end
    end
end
% output
u(1:N-1) = min_u
v(1:N-1) = min_v
the_ju
the_jv
end

```

**APPENDIX J: DETAILED COMPUTER OUTPUTS OF THE SHORTEST
PATH PROBLEM WITH CORRELATED LINK TRAVEL TIMES IN THE 9-
NODE NETWORK (INCLUDING INTERMEDIATE SUCCESSIVE
APPROXIMATIONS)**

The sequence converges after the 4th iteration.

k =

1

u =

Columns 1 through 6

9999999	9999999	9999999	9999999	4
1				

Columns 7 through 9

9999999	8	0
---------	---	---

v =

Columns 1 through 6

9999999	9999999	9999999	9999999	8
2				

Columns 7 through 9

9999999	16	0
---------	----	---

the_ju =

0	0	0	0	9	9	0	9
---	---	---	---	---	---	---	---

the_jv =

0	0	0	0	9	9	0	9
---	---	---	---	---	---	---	---

k =

2

u =

1.0e+006 *

Columns 1 through 7

0.0000	10.0000	0.0000	10.0000	0.0000	0.0000
0.0000					

Columns 8 through 9

0.0000	0
--------	---

v =

1.0e+006 *

Columns 1 through 7

0.0000	10.0000	0.0000	10.0000	0.0000	0.0000
0.0000					

Columns 8 through 9

0.0000	0
--------	---

the_ju =

5	0	6	0	9	9	5	9
---	---	---	---	---	---	---	---

the_jv =

5	0	6	0	9	9	5	9
---	---	---	---	---	---	---	---

k =

3

u =

Columns 1 through 7

9.3333	7.4444	4.3333	12.1111	4.0000	1.0000
9.3333					

Columns 8 through 9

8.0000	0
--------	---

v =

Columns 1 through 7

14.6667	10.5556	7.6667	14.8889	8.0000	2.0000
14.6667					

Columns 8 through 9

14.8889	0
---------	---

the_ju =

5	3	6	1	9	9	5	9
---	---	---	---	---	---	---	---

the_jv =

5	3	6	1	9	9	5	7
---	---	---	---	---	---	---	---

k =

4

u =

Columns 1 through 7

9.3333	7.4444	4.3333	12.1111	4.0000	1.0000
9.3333					

Columns 8 through 9

8.0000	0
--------	---

v =

Columns 1 through 7

13.5185	10.5556	7.6667	14.8889	8.0000	2.0000
14.5926					

Columns 8 through 9

14.8889	0
---------	---

the_ju =

5 3 6 1 9 9 5 9

the_jv =

2 3 6 1 9 9 8 7

k =

5

u =

Columns 1 through 7

9.3333 7.4444 4.3333 11.7284 4.0000 1.0000
9.3333

Columns 8 through 9

8.0000 0

v =

Columns 1 through 7

13.5185 10.5556 7.6667 14.1235 8.0000 2.0000
14.5926

Columns 8 through 9

14.8395 0

the_ju =

5 3 6 1 9 9 5 9

the_jv =

2 3 6 1 9 9 8 7

k =

6

u =

Columns 1 through 7

9.3333	7.4444	4.3333	11.7284	4.0000	1.0000
9.3333					

Columns 8 through 9

8.0000	0
--------	---

v =

Columns 1 through 7

13.5185	10.5556	7.6667	14.1235	8.0000	2.0000
14.5597					

Columns 8 through 9

14.8395	0
---------	---

the_ju =

5	3	6	1	9	9	5	9
---	---	---	---	---	---	---	---

the_jv =

2	3	6	1	9	9	8	7
---	---	---	---	---	---	---	---

k =

7

u =

Columns 1 through 7

9.3333	7.4444	4.3333	11.7284	4.0000	1.0000
9.3333					

Columns 8 through 9

8.0000	0
--------	---

v =

Columns 1 through 7

13.5185	10.5556	7.6667	14.1235	8.0000	2.0000
14.5597					

Columns 8 through 9

14.8176	0
---------	---

the_ju =

5	3	6	1	9	9	5	9
---	---	---	---	---	---	---	---

the_jv =

2	3	6	1	9	9	8	7
---	---	---	---	---	---	---	---

k =

8

u =

Columns 1 through 7

9.3333	7.4444	4.3333	11.7284	4.0000	1.0000
9.3333					

Columns 8 through 9

8.0000	0
--------	---

v =

Columns 1 through 7

13.5185	10.5556	7.6667	14.1235	8.0000	2.0000
14.5450					

Columns 8 through 9

14.8176	0
---------	---

the_ju =

5	3	6	1	9	9	5	9
---	---	---	---	---	---	---	---

the_jv =

2 3 6 1 9 9 8 7