A Routing Algorithm in Stochastic Networks

S. H. Hashemipour
Department of Electrical and Computer
Roudsar and Amlash Branch, Islamic Azad University,
Roudsar, Iran
c_e_hamid@yahoo.com

Abstract- This article investigates the routing problem in networks where edge lengths are independent stochastic variables. A standard routing method for such networks is to specify a path with the shortest travel time. Among the shortcomings of previous methods are disregarding factors like the travel time variance and also, considering routing objectives that are used by different users to be the same. In the present article, a new routing method in stochastic networks is introduced for solving the said problems. The fundamental concept here is that by taking his or her demands and routing objectives into account, a user positioned at a node is given an estimation of the travel time to reach the destination node from different paths. The advantage of this method over the previous works is that it is not limited to one routing objective and that different factors can be effective in computing the shortest travel time which include mean, variance, mathematical expectation, etc. The algorithm presented here has been written using MATLAB software, which presents different paths from the source node to the destination node in addition to the sum of means and variances of the whole path for series-parallel and non-series-parallel networks. The suggested algorithm has been tested on a sample network.

Keywords: Stochastic Networks, Normal Distribution, Shortest Travel Time, Routing.

I. INTRODUCTION

Obtaining the shortest expected travel time in stochastic networks is the objective of a route guidance system [1]. Although this seems to be quite normal, such a system does not consider that a user might accept a long path with a higher variance. Therefore, in stochastic networks, a system capable of adapting itself to the routing demands of different users would be of a higher quality than a common route guidance system. The shortest expected travel time is discussed in [2], where a regressive approximation of the travel time with a label-correcting algorithm has been applied and [3] uses a dynamic programming combined with a reference function for finding the shortest path. A thorough analysis of the application of dynamic programming with the shortest random path problem is introduced in [4]. Wardrop's user equilibrium principle for stochastic networks has been realized in [5]. This is done by specifying paths with the lowest disutility costs. A second-degree program is introduced in [6], whose objective is a linear combination of the mathematical expectation and variance. More complicated routing objectives in time-varying stochastic networks have been considered in [7] and [8]. The problem of the routing demands of different users can be solved by introducing a system that computes the travel time distribution from each node to the destination node. Here, routing demands of users do not have to be specified in advance; however, they can be comparatively used by a user who is travelling through networks. In fact, this method determines mean values and variances of different routes, so a user can choose his or her desired path based on his or her demands. Usually, a stochastic network is modeled by \( G = (N,A) \). The travel time along each \((i, g) \in A\) edge is an independent stochastic variable \( X_{ij} \). In this article, a network is divided into two categories: series-parallel and non-series-parallel. In series-parallel networks, the minimum travel time distribution is computed effectively; however, in non-series-parallel networks, there is no complete algorithm, yet effective approximations can be used.

In this article, at first, different paths from the source node to the destination node are calculated by a simple expansion of Bellman-Ford algorithm. Then, by determining his or her demands, a user makes a routing decision. The present article consists of the following sections: first, the expansion of Bellman-Ford algorithm will be discussed for an adaptation to the travel time distribution and different routing demands. Then, the calculation of the minimum travel time distribution will be discussed. Next, series-parallel and non-series-parallel networks would be proposed along with the introduction of a suggested algorithm and finally, the efficiency of the suggested algorithm will be tested using an example.

II. NOTES AND ASSUMPTIONS

A transportation network is modeled as \( G = (N,A) \). An independent finite and positive stochastic \( X_{ij} \) variable, which indicates the travel time along \((i, g) \) is assigned to each \((i, g) \) edge. Assume that \((i, g)^k \) is the \( k^{th} \) edge from \(i\).
to j and the travel time on \((i, g)\) is shown as \(X_{ij}^k\). Fig. 1 shows a stochastic network.

![Stochastic Network Diagram](image)

Fig 1. A non-series-parallel stochastic network

Suppose that \(N(\mu, \sigma^2)\) is a normal distribution with \(\mu\) and \(\sigma\) as its mean and variance, respectively. The momentary value of \(X_{ij}\) is shown by \(x_{ij}\). If \(Y\) is a stochastic variable, its probabilistic density and collective distribution functions would be shown as \(f_Y(y)\) and \(F_Y(y)\), respectively. If \(Y\) and \(Z\) are stochastic variables, their sum would be given as \(Y + Z\), while their minimum values will be shown as \(\text{MIN}(Y, Z)\). The sum of \(Y\) and \(Z\) is equal to the convolution of their density function, i.e. \(f_Y * f_Z\). The mathematical expectation of the stochastic variable \(Y\) and its variance are shown as \(E[Y]\) and \(\text{Var}(Y)\), respectively.

Also, assume that \(P_{jt}\) is the minimum travel time distribution from \(j\) to \(t\).

\(A^+(i)\) and \(A^-(i)\) show the set of edges that enter and exit the \(i\) node, respectively, while \(N^+(i)\) and \(N^-(i)\) indicate the set of nodes which enter and exit the said node. If there is a parallel edge in the network, \(|A^+(i)|\) would not be equal to \(|N^+(i)|\). Here, the \(i\) node is defined as a series if \(|A^+(i)| = 1\) and \(|A^-(i)| = 1\). Also, two \((i, g)\) nodes are parallel if the number of edges from \(i\) to \(j\) is larger than 1. Assume that \(A(i, g)\) is a list of all the edges from \(i\) to \(j\). In other words,

\[
|A(i, g)| = \sum_{j \in N(i)} |A(i, j)|
\]

If \(i\) is set with the \((k, i)\) input edge and the \((i, j)\) output edge, then a conversion would replace \((k, i)\) and \((i, j)\) series with a new \((k, j)\) edge. If \(i\) and \(j\) are parallel, then a parallel conversion would clear all \((i, j)\) edges and replace them with a new \((i, j)\) edge. Usually, a network is considered series-parallel if it is converted to a single edge by consecutive series and parallel conversions. Also, a \(G\) graph is homeomorphic with \(G'\) graph if \(G'\) is obtained by clearing the edges consecutively and by doing series conversions.

### III. Expanding Bellman-Ford Algorithm

Assume that a user reaches the \(i\) node and wants to get to the \(t\) node. One of the ways of choosing the best neighboring \(j \in N'(i)\) node would be Bellman-Ford algorithm. This algorithm is defined by the following equation:

\[
d_{jt}^* = \arg \min_{j \in \omega^*(i, j)} (d_{jt}^* + d_{ij}^*)
\]

Where \(d_{jt}^*\) is a given estimation of the travel time from \(i\) to \(t\), \(x_{ij}\) is the given travel time from the \((i, j)\) edge and \(d_{jt}^*\) is an estimation of the travel time from \(j\) to \(t\). Determining the neighboring output edge \(j\) at the \(i\) node is done by the following relation:

\[
j^* = \arg \min_{j \in \omega^*(i, j)} (x_{ij} + d_{jt}^*)
\]

Generally, Bellman-Ford algorithm is useful due to its intuition and being simple. The method that is used in this article is replacing \(d_{jt}^*\), \(x_{ij}\) and \(d_{jt}^*\) with their random equivalents, i.e. \(D_{jt}^*, X_{ij}\) and \(D_{jt}^*\), respectively in the said algorithm [15].

\[
D_{jt}^* = \min_{j \in \omega^*(i, j)} (X_{ij} + D_{jt}^*)
\]

In the above-mentioned relation, \(D_{jt}^*\) is a stochastic variable and a routing sub-problem per se because eventually, one has to decide from which node to pass. In order to do so, the following relation is considered.

\[
j^* = \arg \min_{j \in \omega^*(i, j)} (\Gamma(X_{ij} + D_{jt}^*))
\]

Here, \(\Gamma\) is an operator which takes \(X_{ij}\) and \(D_{jt}^*\) arguments, being stochastic variables, as inputs and gives a single actual value. This feature is expanded by the following equation:

\[
j^* \in \arg \min_{j \in \omega^*(i, j)} (\Phi(X_{ij}), \psi(D_{jt}^*))
\]

Where \(\Phi\) and \(\psi\) are operator functions. The purpose of \(\Phi\) and \(\psi\) is to decompose the routing method. \(\Phi\) is the quality of each neighboring \((i,j)\) edge and \(\psi\) indicates the network quality from \(j\) to \(t\). Table 1 gives different probabilities for \(\Phi\), \(\psi\) and \(\Gamma\). It should be noted that \(D_{jt}^*\) has been replaced with \(P_{jt}^*\). The said table provides several possible decompositions related to \(\Gamma\). For instance, in the first routing objective used for reaching a destination, the focus would be on the mathematical expectation.

\[
\Gamma(\Phi(X_{ij}), \psi(P_{jt}^*)) = \Phi(X_{ij}) + \psi(P_{jt}^*) = E[X_{ij}^\prime] + E[P_{jt}^*]
\]

In other words, \(\Gamma\) introduces the expected arrival time from the \(i\) node to the \(t\) node through the \(j\) node. To work on the above-mentioned relation, \(E[X_{ij}^\prime] + E[P_{jt}^*]\) should be computed for each \(j \in N'(i)\). Here, \(j^*\) is determined by minimizing \(E[X_{ij}^\prime] + E[P_{jt}^*]\) for all \(j \in N'(i)\). The second routing objective does not care about the original \((i,g)\) edge and only considers \(E[P_{jt}^*]\). This objective will be useful only when the realization of \(X_{ij}\) has a little effect on the total travel time. The third objective tries to create equilibrium between the mathematical expectation and variance. The special user parameter \(\theta \in R \setminus \{0\}\) determines the importance of the variance for a user. Thus, if a user prefers a low variance for the travel time, he or she should choose a large \(\theta\) value. In the fourth objective, \(\Phi(X_{ij})\) is a
set with the current realization \( x_{ij} \) at the travel time along the \((i,g)\) edge. Such a realization may be achieved by the real-time system information and updates the edge weights of the network in real-time. In the fifth objective, the mean-variance sum for each edge is computed. In this article, the fifth routing objective has been used for the sample network with \( \theta=1 \) [15].

IV. MINIMUM TRAVEL TIME DISTRIBUTION

Travel time distribution is in fact all the fastest possible routes which are shown as \( P^*_st \). Therefore, \( D^*_st \) is replaced with \( P^*_st \) in Bellman-Ford algorithm. One way for finding \( P^*_st \) is to set each \((i, g)\) edge with some of the \( x_{ij} \) realizations for each \( X_{ij} \). Then, the fastest route algorithm from \( s \) to \( t \) is implemented and its results will be recorded. In the next step, the edges are set with a new series of realization and the above-mentioned trend is repeated until the fastest route is obtained from all possible combinations. Though this method is not efficient, it is correct. Assume that \( \pi_{st}^k \) is the \( k^{th} \) route from \( s \) to \( t \). The travel time from \( s \) to \( t \) is shown as \( P_{st}^k \) along \( \pi_{st}^k \). In fact, \( P_{st}^k \) is calculated by adding up \( X_{ij} \) for all \((i, g)\) \( \in \pi_{st}^k \). It should be noted that this result is obtained from the convolution of density functions \( f_{xij} \) (\( X_{ij} \)). If \( \pi_{st}^k \) and \( \pi_{st}^l \) are the only two routes from \( s \) to \( t \) with their edges being independent, then \( P_{st}^* \) would be defined as the minimum \( P_{st}^k \) and \( P_{st}^l \).

\[
P_{st}^* = \min(P_{st}^k, P_{st}^l)
\]

(8)

Convolution and minimum are two fundamental operators for the calculation of \( P_{st}^* \). If \( \pi_{st}^k \) and \( \pi_{st}^l \) have similar edges, \( P_{st}^k \) and \( P_{st}^l \) won't be independent anymore. This dependence is the fundamental difficulty of computing \( P_{st}^* \). Consider the \( G= \langle N, A \rangle \) network, where \( N= \{s,i,j,t\} \) and \( A= \{(s,i),(i,j),(j,t)\} \). There are two ways for computing \( P_{st}^* \):

\[
P_{st}^* = \min\{X_{s} + X_{ij} + X_{ji} + X_{jt} + X_{t} \}
\]

(9)

\[
P_{st}^* = X_{s} + \min\{X_{ij} + X_{jt} \} + X_{t}
\]

(10)

Due to the fact that convolution and minimum operators include independent variables, the second equation is much easier. In fact, in the first equation, two minimum operator arguments are not independent, which is an indication of the importance of calculating convolution and minimum [15].

V. A PERSPECTIVE ON THE ROUTE GUIDANCE SYSTEM

In a route guidance system, whenever a user reaches node \( i \) and intends to go to node \( t \), the \( \Gamma, \Phi \) and \( \psi \) values are determined. Then, an algorithm corresponding to the calculation of the minimum travel time \( P^*_{st} \) for each \( j \in N \) (\( i \)) node is computed. The \( P^*_{st} \) computation algorithm on series-parallel and non-series-parallel networks will be discussed in the next section. It should be noted that in series-parallel and non-series-parallel networks, our network is assumed to be flat. However, in the \( G \) route guidance, the graph may not be flat so we should produce a flat sub-graph \( G' \) where \( G' \subseteq G \). For more discussion, please refer to [11].

<table>
<thead>
<tr>
<th>Table1. Samples of routing objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Gamma(\Phi(X_{ij}), \psi(P_{ij})) )</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

A. SERIES-PARALLEL NETWORK

Martin discussed the importance of series-parallel networks in the calculation of time distributions in a network [9]. He presented three ways for recognizing a stochastic series-parallel network with \( s \) source and \( t \) destination. At first, it is assumed that the \( i \) node is set with (\( k, i \) and (\( i, j \)). If a series conversion is done on node \( i \), then (\( k, i \) and (\( i, j \)) can be replaced with (\( k, j \)). Also, \( X_{ij} \) and \( X_{ki} \) are replaced with a travel time equivalent to \( X_{ij} = X_{ji} + X_{ij} \). In other words, a series conversion reduces a network to an equivalent network in a way that the number of edges is reduced to one. Moreover, assume that \( i \) and \( j \) are parallel with (\( i, j \)) \( \phi^1 \) and (\( i, j \)) \( \phi^2 \) edges. If a parallel conversion is performed on \( i, j \), then (\( i, j \)) \( \phi^1 \) and (\( i, j \)) \( \phi^2 \) can be replaced with a single (\( i, j \)) edge. Similarly, \( X_{ij} \) is replaced with \( \min\{X_{ij} + X_{ij} \} \) [15]. Finally, using these conversions leads the network to be converted to an edge. The travel time of this edge introduces the minimum travel time distribution from \( s \) to \( t \), i.e. \( P^*_{st} \). Martin expanded such a reduction algorithm in [9]. Figure 2 is a presentation of a series-parallel network.
B. NON-SERIES-PARALLEL NETWORK

The flat and directed graph $G$ is series-parallel, if and only if it is homeomorphic with $G'$ [12,13]. Non-parallel-series network related problems are given in [10] and [11]. Usually, the calculation of the minimum travel time distribution in non-series-parallel networks is difficult because an order of convolution or minimum operators, which maintains the independence in $X_{ij}$ is non-existent.

One method for non-series-parallel networks is to fix the travel time on the given edges. If these edges are fixed according to [14], then the network can be converted to a series-parallel one. If the order of convolution and minimum is recorded in this conversion, then $P^*_{st}$ could be approximated by the Monte Carlo simulation. Another method is faster than approximation, yet with less correctness potential and it is to fix edge weights by the mathematical expectation. Shown network in Figure 1 is a non-series-parallel network.

C. THE SUGGESTED ALGORITHM

In this article, an algorithm has been introduced, which specifies all routes from the source to the destination for series-parallel and non-series-parallel networks and gives the total sum of means and variances of the whole path. In table 1, each of the mentioned routing objectives can be easily defined for the suggested algorithm. In the present article, the fifth routing objective has been used for simulation. Initially in this algorithm, all network nodes, their connective edges and the mean-variance of the whole path are given as matrices. The algorithm is written using MATLAB software. The mathematical expectation of the edges can be easily defined using the functions of this software and the suggested algorithm informs the user of the mathematical expectation of different paths. In the present article, a parameter called 'teta' has been assigned to the demands of different users. Thus, by specifying the importance of the variance for every user, a number is assigned to 'teta'. This algorithm is presented in table 2 and its results are given in table 3. It has been tested for the sample non-series-parallel network shown in figure 1.
Table 3. Simulation results of the suggested algorithm

<table>
<thead>
<tr>
<th>Different Path</th>
<th>Average</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 3 6 9 8 10</td>
<td>30.2000</td>
</tr>
<tr>
<td>2</td>
<td>2 3 6 5 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>3</td>
<td>2 3 6 5 9 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>4</td>
<td>2 3 6 5 7 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>5</td>
<td>2 3 4 7 8 10</td>
<td>30.2000</td>
</tr>
<tr>
<td>6</td>
<td>2 3 4 5 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>7</td>
<td>2 3 4 5 9 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>8</td>
<td>2 3 4 5 7 8 10</td>
<td>40.2000</td>
</tr>
<tr>
<td>9</td>
<td>2 3 5 8 10</td>
<td>30.2000</td>
</tr>
<tr>
<td>10</td>
<td>2 3 5 9 8 10</td>
<td>30.2000</td>
</tr>
<tr>
<td>11</td>
<td>2 3 5 7 8 10</td>
<td>30.2000</td>
</tr>
</tbody>
</table>

After implementing this algorithm, total means and variances of different paths are given to the user as outputs. If the minimum travel time distribution is demanded, it can be easily computed by MATLAB software functions. For instance, the minimum travel time distribution path (1) is computed as follows. The convolution of two functions is calculated by the Conv order.

\[ X_{jt}^* = X_{23} + X_{36} + X_{69} + X_{98} + X_{810} \]  
\[ P_{jt}^* = f_{23}(x_{23}) \ast f_{36}(x_{36}) \ast f_{69}(x_{69}) \ast f_{98}(x_{98}) \ast f_{810}(x_{810}) \]

Where \( f_{ij}(x_{ij}) \) is the density function related to the normal distribution probability of i and j edges. The density function of the normal distribution probability and the mathematical expectation equation are given in the following equations:

\[ f_x(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-m)^2}{2\sigma^2}} \]  
\[ E(x) = \int_{-\infty}^{\infty} x f(x) \]  

VI. CONCLUSION

This article has investigated the routing problem in stochastic networks, where different routing demands of different users have been taken into account. Here, an algorithm is provided, which shows different paths from the source to the destination and gives the total mean and variance of the whole path. In the present article, different paths are given to a user, who is positioned at a node, so he or she can choose his or her desired path as wished. The suggested algorithm has been written by expanding Bellman-Ford algorithm and is tested for a non-series-parallel network and can be used for other networks as well.

REFERENCES


[10]- S. E. Elmaghraby, Activity Networks: Project Planning and Control by Network Models.


