A store-and-forward neural network to solve multicriteria optimal path problem in time-dependent networks

Jin Liu¹, Li Chen², Honghao Zhang¹*, Wei Huang¹, Kaiwen Jiang¹, Hongmin Zhang¹
¹ School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, 300380, China
² Force Early Warning Academy, Wuhan, Hubei, 430019, China
*Corresponding author’s e-mail: zhanghonghao@tjut.edu.cn

Abstract. This paper introduces the constrained multi-objective optimal path problem in time-dependent networks. In the existing literatures, the constraints are all imposed on the objective function while the problem constraints are related to the non-objective function. It is the difference that makes the traditional algorithm unable to get a better solution quality. In this light, we propose a store-and-forward neural network (SFNN) that finds the better result. In the design of SFNN, the topology of neural network is the same as that of time-varying network, and each node is designed as store-and-forward neuron. Each neuron transmits information to other neurons by sending signals. The experimental results show that compared with the traditional methods, the accuracy is significantly improved when the calculation time is acceptable.

1. Introduction

The constrained multi-objective optimal path problem has a wide range of applications in different engineering fields such as the Internet of Things [1] and transportation network [2]. In this regard, traditional network algorithms approximate the actual network to a static model, and the solution obtained is far from the actual optimal solution. Therefore, attention to the dynamic characteristics of arc attributes in time-dependent networks has attracted more and more attention. The constraints are all imposed on the objective function in the existing literatures, while our constraints of the proposed problem is imposed on the non-objective function. This difference directly results in the worse solution quality for traditional algorithms in our experiments. In order to compare with other algorithms, the common indicators of multi-objective optimization solution set are used to analyze the results.

In time-dependent networks, Orda and Rom [3] have found that computing multicriteria optimal paths violates the subpath optimality property due to time-varying arc weights. Kostreva and Wiecek [4] are the first to deal with the multicriteria optimal path problem in time-dependent networks. Their algorithm applies Bellman’s optimality principle in backward direction. It effectively identifies whether or not a subpath can belong to one of the optimal paths. Most of the research works [5, 6, 7] have been carried out under their idea. Hamacher et al. [6] have proposed backward label setting algorithm for solving time-dependent bicriterion shortest path problem. This algorithm reverses the path from the target node to the starting node, pruning part of the path, but the disadvantage is that it is difficult to determine the route time range of the target node. Androutsopoulos and Zografos [5] have
solved multi-objective routing problem where multiple objectives are equally important. However, these methods always delete part of the Pareto optimal path in the calculation process under resource constraints, so that the complete set of optimal paths cannot be obtained.

Neural network approaches are gaining considerable attention in recent years. Different neural network approaches to solve route-related problems have been reported in the literature. Dynamic neural networks [8], time-delay neural networks [9] and convolutional neural networks [10, 11] are examples of the reported methods. A time-delay neural networks [9] is proposed to solve time-dependent shortest path problem. Based on the ideas, a new neural network algorithm is designed to solve our problem.

This paper designs the store-and-forward neural network to solve the multicriteria shortest path problem in time-dependent network. Structure of the paper is organized as follows: In Section 2 the problem definition is stated. The proposed SFNN algorithm is described in Section 3. In Section 4 the numerical experiments are studied using multicriteria evaluation index. In Section 5 we summarize the advantages of SFNN.

2. Problem definition

This section first describes the relevant definitions of time-dependent networks, and then defines the multi-objective shortest path problem in a time-varying environment.

2.1. Time-dependent Network

Let \( G = (N, A, W) \) is a time-dependent network, where \( N \) is a set of \( n \) nodes, \( A \subseteq \{(i, j) \in N^2 | i \neq j\} \) is a set of arcs. \( W \) is a set of cost functions \( Y_{ij}(t_i) \) which stands for a cost function for a tourist from node \( v_i \) to node \( v_j \) at the depart time \( t_i \). Each arc \((i, j) \in A \) is associated with \( k \in N^+ \) weight variables \( c_{ij}^1(t_i), \cdots, c_{ij}^k(t_i) \). The values of variables are determined by \( Y_{ij}(t_i) \) which is defined as \( Y_{ij}: \mathbb{N} \rightarrow \mathbb{R}^+ \), that is, \( Y_{ij}(t_i) = (c_{ij}^1(t_i), \cdots, c_{ij}^k(t_i)) \).

A route with \( n \) nodes can be represented as a sequence of tuples \( r = ([r_1, r_2]; t_1), \cdots, ([r_i, r_{i+1}); t_i], \cdots, ([r_{n-1}, r_n); t_{n-1}] \), where \( r_i \in V \) with \( 1 \leq i \leq n \), \( (r_i, r_{i+1}) \in A \) is the \( i \)th edge on the path, and \( t_i \) is the departure time from node \( r_i \) for \( 1 \leq i \leq n - 1 \).

2.2. Multicriteria shortest path problem with resource constraints

Given a time-dependent network \( G = (V, A, Y) \), the constrained multicriteria shortest path problem(TCMPP) is to determine the optimal path from the source node \( s \in V \) to the destination node \( d \in V \) with limited battery power constraints to minimize the travel time and travel cost. The TCMPP problem can be formulate as (1)-(5).

\[
\min_{r \in R_{r,d}} Z(r) = (z_1(r), z_2(r))
\]  
\[
z_1(r) = t_0 + \sum_{(r_i,r_{i+1}) \in r} c_{ij}^1(t_i)
\]  
\[
z_2(r) = \sum_{(r_i,r_{i+1}) \in r} c_{ij}^2(t_i)
\]  
\[
\sum_{(r_i,r_{i+1}) \in r} c_{ij}^3(t_i) < R
\]  
\[
e_i \leq t_i \leq l_i
\]  

The objective function (2) sums the routing time of each arc of the path to calculate the total travel time of a path. Objective function (3) calculates the total cost by summing the cost consumed by each
arc of a path. Constraint (4) describes the resource consumption of any path $r \in R_{r,d}$ to tourists shall not exceed the limit value. However, the access time of each node of the constraint (5) path is within a certain time range, and it can arrive in advance and then wait until the earliest access time.

3. Design of store-and-forward neural network

A SFNN is a store-and-forward neural network without data training. Each neuron serves as input and output ports from precursor neurons and successor neurons respectively to communicate information through the mechanism of forwarding and storing signals.

3.1. A general neuron’s structure of SFNN

A neuron is composed of five parts: input, depart time selector, signal generator, signal encoder and output. Figure 1 depicts the overall structure of the general neuron of SFNN.

(1) Input: The input receives the signals from all associated neurons and transmits them to the signal decoder.

(2) Signal decoder: The signal decoder consists of five parts: $S_{x,t}$, $t_{x,t}$, $c_{x,t}$, $e_{x,t}$, and $r_{x,t}$. These variables can be regarded as parameters of a signal $S_{x,t}$ decoded by function $F$. 
(3) Depart time selector: The depart time selector part uses the function $H_{ij}(t)$ for each received and decoded signal to calculate the time to send it out again. Among them, let $DT_{ij}$ be the set of all possible departure times along the arc $(i,j)$, and $t$ represents the arrival time when the signal from the precursor neuron is received.

$$DT_{ij} = \{t_c | c = 1,2,\ldots,l_{ij}\}$$

where $c$ denotes the index of depart time, $l_{ij}$ represents the number of depart times on the arc $(i,j)$ and $(i,j) \in A$. Based on the formulation (13), we have

$$H_{ij}(t) = \arg \min_{t_c} \{O_{ij}(t,t_c) | c = 1,2,\ldots,l_{ij}\}$$

Where

$$O_{ij}(t,t_c) = \begin{cases} t_c, & t_c \geq t \\ Q, & t_c < t \end{cases}$$

From formula (7) and (8), we can see that the departure time of node $i$ along arc $(i,j)$ to node $j$ is calculated by the function $H_{ij}(t)$. In particular, the departure of the signal should occur after neuron $i$ receives the corresponding precursor neuron signals. Otherwise, it is set to a given infinite number $Q$.

(4) Signal generator: The signal generator part calculates the parameters for generating signals (viz. $t_{ij}$, $c_{ij}$, $e_{ij}$, and $p_{ij}$, $j \in \varphi^+_i$). Their expressions are given in the following:

$$t_{ij} = t_{x_i} + c_{ij}(t_{x_i})$$

$$w_{ij} = w_{x_i} + c_{ij}^2(t_{x_i})$$

$$u_{ij} = e_{x_i} + c_{ij}^3(t_{x_i})$$

$$r_{ij} = \langle r_{x_i}, i \rangle$$

Neuron $i$ will temporarily store the wave information in set $A_i$. When certain conditions are met, they will send out waves to successor neuron and delete them from $A_i$.

$$A_i[k_i] = \{t_{ij}, w_{ij}, u_{ij}, r_{ij}, j\}$$

(5) Signal Encoder: A signal $S_{ij}$ is encoded by function $E$, which the expression can be stated as follows:

$$L(A_i) = \begin{cases} A_i[b], & \text{if} \ t = t_{ij} \\ \emptyset, & \text{if} \ t \neq t_{ij}, b = 1,2,\ldots,k_i \end{cases}$$

(6) Output: The output will send signals to the corresponding subsequent nodes associated with the direction according to the wave information.

3.2. Overall Time Signal Neural Network Algorithm

**Algorithm 1. Store-and-forward neural network algorithm**

1. Set $t = t_0$, Paths = $\emptyset$.
2. Initialize SFNN using network initialization algorithm.
3. while $t \leq T$ do
4.   Update neurons based on network updating algorithm.
5.   $t = t + \Delta t$. 

Output: The output will send signals to the corresponding subsequent nodes associated with the direction according to the wave information.
6. until $t < T$.
7. Record optimal paths on Paths.

Algorithm 2. Network initialization algorithm
1. for $i \in N$ and $i \neq s$ do /* Initialization the $i$th neuron*/
2.      Set $t_{xi} = 0$, $w_{xi} = 0$, $r_{xi} = \langle \emptyset \rangle$, $u_{xi} = 0$, $A_i[|M|]$, $k_i = 0$.
3. End for /* $M$ is the predefined size of the array $A_i$ */
4. If $i = s$,
5.      Set $t_{xi} = 0$, $c_{xi} = 0$, $p_{xi} = \langle \emptyset \rangle$, $A_i[|M|] = \langle \emptyset \rangle$, $k_i = 0$.
6.      Calculate $t_{ij}$, $w_{ij}$, $u_{ij}$, $r_{ij}$ using Equations (9) - (12).
7.  End if
8. End for

Algorithm 3. Network updating algorithm
1. for $i \in N$ and $i \neq d$ do /* Update the non-destination neurons */
2.     Calculate $t_{ij}$, $w_{ij}$, $u_{ij}$, $r_{ij}$ using (9) - (12).
3.     Calculate $A_{i}[k_{ij}]$ based on (13).
4.     Calculate $S_{ij}$ based on (14).
5.     end for
6. if $i = d$ /* Update the destination neuron */
7.    if $t_{x_i}$, $c_{x_i}$ is dominated by signals in $A_i$, discard it.
8. else, add this signal to $A_i$.
9. End if

4. Experimental results
This section, through a group of experiments, verifies the high computational performance and accuracy of WFNN algorithm compared with traditional algorithms.

Table 1. Three quality indicators of the Pareto optimal set.

| Ins. | Hypervolume | Inverse generational distance | Diversity Comparison |
|------|-------------|------------------------------|----------------------|
|      | WD.         | And.                        | WD.                 | And.                | SFNN |
| 1    | 3190        | 3284                        | 0                   | 33%                 | 100% | 100% |
| 2    | 4310        | 4608                        | 4750                | 4.4                 | 3.7  | 0    |
| 3    | 4623        | 3617                        | 4087                | 8.3                 | 1.7  | 0    |
| 4    | 2880        | 2146                        | 3016                | 17.4                | 9.6  | 0    |
| 5    | 3534        | 3660                        | 4130                | 6.0                 | 4.0  | 0    |
| 6    | 2244        | 908                         | 2464                | 4.5                 | 5.1  | 0    |
| 7    | 4130        | 4018                        | 4150                | 2.6                 | 1.3  | 0    |
| 8    | 1208        | 1294                        | 1435                | 2.4                 | 1.8  | 0    |
| 9    | 3010        | 3010                        | 3010                | 11.5                | 3.6  | 0    |
| 10   | 2708        | 2862                        | 3021                | 12.4                | 3.6  | 0    |
| 11   | 1720        | 2106                        | 2612                | 17.1                | 8.4  | 0    |
| 12   | 1113        | 1101                        | 1155                | 26.9                | 5.0  | 0    |
| 13   | 4093        | 4540                        | 4550                | 5.8                 | 2.5  | 0    |
| 14   | 3795        | 3806                        | 3864                | 6.0                 | 4.6  | 0    |
| 15   | 2900        | 3360                        | 3353                | 3.7                 | 2.4  | 0    |
| 16   | 2132        | 3016                        | 3190                | 11.5                | 2.7  | 0    |
| 17   | 5538        | 5609                        | 5609                | 11.3                | 3.3  | 0    |
| 18   | 3745        | 4140                        | 4278                | 6.2                 | 2.5  | 0    |
| 19   | 4104        | 4536                        | 4680                | 8.1                 | 4.2  | 0    |
| 20   | 5830        | 5850                        | 5850                | 3.8                 | 2.1  | 0    |

The results concerning three metrics are shown in Table 1, the direct comparison of the different algorithms in terms of HV, IGD, and C-metric. WDIjksktra’s algorithm offers worse hypervolumes in which the few solutions found are not enough to obtain a high hypervolume. In instance 4 and 6,
Despite the number of solutions of the Androutsopoulos’ algorithm is not small, the HV metric is lower than the other two methods. The results suggest that some of the solutions of the Androutsopoulos’ algorithm are non-supported and their contribution to the hypervolume is lower than the SFNN.

5. Conclusion
This paper proposes a time-dependent multicriteria optimal path problem that constrains non-objective functions. We design a store-and-forward neural network (SFNN) that can work to find better results without any data training. Unlike traditional algorithms that need to be executed sequentially in the sequence of statements in the program, each neuron(node) works independently and at the same time, which can greatly increase the calculation speed. In addition, each neuron can faithfully record the state of the algorithm during execution, including sub-paths and corresponding cost functions, so that programmers can adjust and optimize the parameters of the algorithm.

Acknowledgments
This paper has been supported by the National Natural Science Foundation of China under grant 61673295, the Natural Science Foundation of Tianjin for Distinguished Young Scholars under grant 19JCQJC61500, Tianjin Science and Technology Planning Project under grant 20YDTPJC00520, 20KPZDRC00040.

References
[1] Xie, M., Bai, Y., Huang, M., Deng, Y., Hu, Z. (2020) Energy- and time-aware data acquisition for mobile robots using mixed cognition particle swarm optimization. IEEE Internet of Things Journal, 7:7734–7750.
[2] Veneti, A., Makrygiorgos, A., Konstantopoulos, C., Pantziou, G., Vetsikas, I.A. (2017) Minimizing the fuel consumption and the risk in maritime transportation: A bi-objective weather routing approach. Computers & Operations Research, 88:220–236.
[3] Orda, A., Rom, R. (1997) Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. Journal of the ACM, 37:607-625.
[4] Kostreva, M.M and Wiecek, M.M. (1993) Time dependency in multiple objective dynamic programming. Journal of Mathematical Analysis and Applications, 173:289–307.
[5] Androutsopoulos, K.N, Zografos, K.G. (2009) Solving the multi-criteria time-dependent routing and scheduling problem in a multimodal fixed scheduled network. European Journal of Operational Research, 192:18–28.
[6] Hamacher, H.W., Ruzika, S., Tjandra, S.A. (2006) Algorithms for Time-dependent Bicriteria Shortest Path Problems. Discrete Optimization, 3:238–254.
[7] Veneti, A., Makrygiorgos, A., Konstantopoulos, C., Pantziou, G., Vetsikas. I.A. (2017) Minimizing the fuel consumption and the risk in maritime transportation: A bi-objective weather routing approach. Computers & Operations Research, 88:220–236.
[8] Chen, M., Zhu, D. (2020) Optimal Time-Consuming Path Planning for Autonomous Underwater Vehicles Based on a Dynamic Neural Network Model in Ocean Current Environments. IEEE Transactions on Vehicular Technology, 69:14401–14412.
[9] Huang, W., Yan, C., Wang, J., Wang, W. (2017) A time-delay neural network for solving time-dependent shortest path problem. Neural Networks, 90:21–28.
[10] Balali, A., Asadpour, M., Campos, R., Jatowt, A. (2020) Joint event extraction along shortest dependency paths using graph convolutional networks. Knowledge-based Systems, 210:1-15.
[11] Wang, J., Chi, W., Li, C., Wang, C., and Meng, M.Q.H. (2020) Neural RRT*: Learning-Based Optimal Path Planning. IEEE Transactions on Automation Science and Engineering, 17(4):1748–1758.