Finding the Shortest Path in Dynamic Network using Labeling Algorithm

Sahar Abbasi
Department of Industrial Engineering
Najafabad Branch, Islamic Azad University
Esfahan, Iran

Sadoullah Ebrahimnejad
Department of Industrial Engineerin
Islamic Azad University, Karaj Branch
Rajaee Shahr, Karaj, Iran

Abstract

This study concerns the problem of finding shortest paths from one node to all other nodes in networks for which arc costs can vary with time, each arc has a transit time and parking with a corresponding time-varying cost is allowed at the nodes. it show that this problem is equivalent to a classical shortest path problem in a time-expanded network. The label correcting algorithm is used for finding shortest paths.

Keywords: Dynamic shortest paths, time-expanded network, label correcting algorithm.

1. Introduction

In congested transportation networks, arc travel times change over time due to time-of-day variations in traffic congestion. Even if one can account for these time-of-day variations, future travel times can at best be known a priori with uncertainty due to unforeseen events, such as poor roadway conditions, vehicle breakdowns, traffic accidents, and driver behavior. In this work, we develop path search techniques that explicitly consider the inherent time-varying nature of future travel times. Recent studies have focused on time-dependent graphs [1–6]. This type of generalization is useful for real word applications. A simple example is that of a computer communications network composed of dial up links each with its individual dialing schedules. Since delays depend on these predetermined schedules, finding the best route for a message from source to destination involves the computation of time-dependent functions [5]. Many types of networks exhibit this kind of dynamic behavior. in practical applications, weight of a link is often the time required to traverse that link. This weight may change as a function of time. Consider a network that represents a city with the usual rush hour traffic patterns. The dynamic shortest path problem is a generalization of the shortest path problem whose aim is to find a path of minimum cost (length) through a network for which

1. each arc has a *transit time* which specifies the amount of time to traverse through each arc,
2. parking (or waiting) is permitted at the nodes of the network for later departure,
   And
3. Network characteristics such as arc transit times and costs (or length) can change over time and are known for all values of time.

The aim of this paper is to study the dynamic shortest path problem in a discrete time setting with positive transit times. We show that the problem is reduced to a classical shortest path problem on a so-called *time-expanded network*. This allow us to apply algorithms that are available in the classical case to the dynamic case. Then we use label correcting algorithm for solving the above problem. The paper is organized as follows: After review of the shortest path problem in Section 2, we define necessary notation of the dynamic shortest path problem in Section 3, then we use *Label Correcting Algorithm* for solving this problem and summarize our conclusions the related problems in Sect. 4, 5. respectively.

2. Review of the shortest path problem

The shortest path problem has many useful applications. Hence, it has been a subject of extensive research. Dijkstra [7] provided a now well-known algorithm, which solves the problem in $O(n^2)$ time. Floyd [8] relaxed the constraint that all weights must be non-negative. Floyd provided an algorithm that solves this more general problem in $O(n^3)$ time. When the weights change as a function of time $w_{ij}(t)$ such networks are called dynamic networks. In a dynamic network, weights exhibit a dynamic pattern.

However, the weights are not deterministic at any time or over any time interval. These networks typify what is seen in the real world. Fu [4] and Dreyfus [2] show that previous algorithms, such as Dijkstra's [7], can be extended to provide polynomial time algorithms for this more general network type. However, they assume that the weight functions are known ahead of time, are monotonic and do not change. Such assumptions are not necessarily suited for many real life situations. Fu showed that finding the shortest path in these networks is an intractable problem. The intractability is a result of the violation of Bellman's principle. Bellman's principle of optimality states that ''any sub path of a shortest path must be a shortest path'' [4]. He has shown that when travel times do not change monotonically Bellman's principle may be violated. Fu developed several types of heuristic search algorithms. The general properties and algorithms have been discussed in both discrete time and continuous time settings by Ahuja et al. [9], Cai et al. [1], Chabini [10] Orda and Rom [6, 11], Pallottino and Scutella [12], and Philpott and Mees[13,14] among others. The problem considered in this paper is that of a dynamic network, where the weights (costs) $C_{ij}(t)$ change as a function of time.

3. Definitions and preliminaries

Given a dynamic network $G = (V, E, T)$ with discrete-time consists of a set of nodes V, ($|V| = n$), node set $V = \{1, 2, \ldots, n\}$,a set of arcs E,($|E| = m$), *arc set* $E \subseteq V \times V$ and a fixed time horizon $T \in R^{+}$. we assume that every pair of nodes is connected by at most one arc. Each arc $(i, j) \in E$ has an associated *transit time* $\lambda_{i,j}$, if a vehicle leaves node i at time t along the arc (i, j) then it arrives at node j at time $t + \lambda_{i,j}$. we define a node-time pair to be a member of $V \times \{0,1,\ldots,T-1\}$. A *discrete-time dynamic path* from node-time pair (i, α) to node-time pair (j, β) is a sequence of distinct node-time pairs as

$$P : (j, \alpha) = (i_1, t_1), (i_2, t_2), \ldots, (i_s, t_s) = (j, \beta),$$

in which either $(i_k , i_{k+1}) \in E$ and $t_{k+1} = t_k + \lambda_{ik\,,ik+1}$, in which case traffic leaves node i_k for node i_{k+1} at time t_k and arrives at t_{k+1}, or $i_k = i_{k+1}$, in which case parking occurs at node i_k at the time step t_{k+1}. Such a sequence is called a *discrete-time dynamic cycle* if $(i, \alpha) = (j, \beta)$ and the other node-time pairs are distinct. The *cost* of a dynamic path P is defined by

$$Cost\ P := \sum_{(i_k, i_{k+1})} c_{i_k, i_{k+1}}(t_k) + \sum_{i_k = i_{k+1}} \sum_{t_k}^{t_{k+1}} f_{i_k}(\delta)$$

where $c_{i,j}(t)$ is the traversal cost along arc (i, j) at time t, and $f_{i_k}(\delta)$ is the parking cost at node i at time t. A path P is said to be a *dynamic shortest path* from to node-time pair (i,α) to node-time pair (j, β), if $\text{Cost}[P] \leq \text{Cost}[P']$ for all dynamic paths P' from (i,α) to (j, β). We suppose that the dynamic network G contains a dynamic path from node-time pair (1,0) to every other node-time pair (i, t). by introducing artificial arcs $(1,i)$ joining node 1 to node i for each node $i \in V \setminus \{1\}$. Each artificial arc $(1, i)$ has a zero transit time and a large traversal cost. It is clear that no such arc would appear in a dynamic shortest path from (1, 0) to any node-time pair (i,t) unless network G contains no dynamic path from (1,0) to (i,t) without artificial arcs. We now use the Modified Label Correcting Algorithm for solving the discrete dynamic shortest path problem. The basic idea is to fan out from node-time pair (1, 0) and label other node-time pairs according to their distances from (1,0). Before preceding our discussion, by using [9] give necessary and sufficient conditions for a set of labels to represent the length of shortest dynamic paths.

Shortest Path Optimality Conditions: *For any node-time pair* (i,t), *let* $d_i(t)$ *denote the length of some dynamic path from node-time pair* $(1,0)$ *to node-time pair* (i,t). *Then the labels* $d_i(t)$ *represent the length of shortest dynamic paths if and only if they satisfy the following shortest path optimality conditions:*

1) $d_i(t) + \sum_t^T f_{i_k}(\delta)$ *is monotonic decreasing on* $\{0,1,\ldots,T-1\}$ *for every* $i \in V$;
2) $c_{i,j}(t) + d_i(t) - d_j(t + \lambda_{i,j}) \geq 0$, *for every* $(i, j) \in E$ *and* $t \in \{0,1,\ldots,T-1\}$.

Proof -It is obvious that if the labels $d_i(t)$ are the length of shortest augmenting paths, they satisfy the optimality conditions (1) and (2). So we assume that for any *node-time pair* (i,t), labels $d_i(t)$ is the length of some dynamic path from *node-time pair* (1,0) to *node-time pair* (i,t) satisfying conditions (1) and (2). Thus $d_i(t)$ is an upper bound on the length of the shortest dynamic path from *node-time pair* (1,0) to *node-time pair* (i,t). We show that $d_i(t)$ is also a lower bound on the length of the shortest dynamic path from *node-time pair* (1,0) to *node-time pair* (i,t), which implies the conclusion of the Optimality Conditions. Consider an arbitrary dynamic path $P : (1, 0) = (i_1, t_1), (i_2, t_2), \ldots, (i_q , t_q) = (i, t)$ from (1,0) to (i,t).

To simplify notation, we assume without loss of generality that $i_k = i_{k+1}$ for only one k, say τ $(1 \le \tau \le q-1)$ and $(i_k, i_{k+1}) \in E$ for all other $k = 1, \ldots, q$. This means that the parking only occurs at node i_τ from time t_τ to $t_{\tau+1}$. Hence we have $t_{k+1} = t_k + \lambda_{ik,ik+1}$ for $k = 1, \ldots, \tau-1, \tau+1, \ldots, q$. Conditions (1) and (2) imply that

$$d_i(t) = d_{i_q}(t_q) \le d_{i_{q-1}}(t_{q-1}) + c_{i_{q-1},i_q}(t_{q-1})$$

$$\le d_{i_{q-2}}(t_{q-2}) + c_{i_{q-2},i_{q-1}}(t_{q-2}) + c_{i_{q-1},i_q}(t_{q-1})$$

$$\vdots$$

$$\le d_{i_{\tau+1}}(t_{\tau+1}) + \sum_{k=\tau+1}^{q-1} c_{i_k,i_{k+1}}(t_k)$$

$$\le d_{i_\tau}(t_\tau) + \sum_{t_\tau}^{t_{\tau+1}} f_{i_\tau}(\delta) + \sum_{k=\tau+1}^{q-1} c_{i_k,i_{k+1}}(t_k)$$

$$\vdots$$

$$\le d_{i_1}(t_1) + c_{i_1,i_2}(t_1) + \sum_{k=2} \tau - c_{i_k,i_{k+1}}(t_k) + \sum_{t_\tau}^{t_{\tau+1}} f_{i_\tau}(\delta) + \sum_{k=\tau+1}^{q-1} c_{i_k,i_{k+1}}(t_k)$$

$$= Cost\,[P]$$

Therefore, $d_i(t)$ is a lower bound on the cost of any dynamic path from (1,0) to (i, t).■

Time-expanded Network: Ford and Fulkerson introduce the notion of time-expanded networks. A time-expanded network contains one copy of the node set of the underlying 'static' network for each discrete time step (building a time layer). For a dynamic network $G = (V,A,T)$ the time expanded network $G^T = (V^T, A^T)$ is defined as follows: A *time-expanded network* of G, denoted by $G(\varphi)$,where $\varphi = \{t_0, t_1, \ldots, t_p\}$ contains p+1 copies of V, denoted by V_0, $V_1, \ldots, V_p$,in which V_{q-1} corresponds to the time step t_{q-1} for $q = 1, \ldots, p-1$, and V_p to the time horizon T. Subsequently, index q varies from 1 to p. The copy of node $i \in V$ in V_{q-1} is denoted by i_{q-1}. For each arc $(i,j) \in E$ and each time $t_{q-1} \epsilon \varphi$ with $0 \le t_{q-1} + \lambda_{i,j} \le T$, Traversing through arc$(i_{q-1}, j_{q'})$ where, $t_{q'} = t_{q-1} + \lambda_{i,j}$ corresponds to leaving node i at time t_{q-1} and arriving at node j at time $t_{q'}$. Hence, arc(i_{q-1}, j_q) has an associated cost $c_{i,j}(t_{q-1})$. For each node i , there is a holdover arc from i_{q-1} to i_q . Traveling through arc (i_{q-1}, i_q) corresponds to the parking at node i from time t_{q-1} to t_q . So holdover arc (i_{q-1}, i_q) has an associated cost $f_i(t_{q-1})$. An illustration of a time-expanded network is given in Fig. 1.

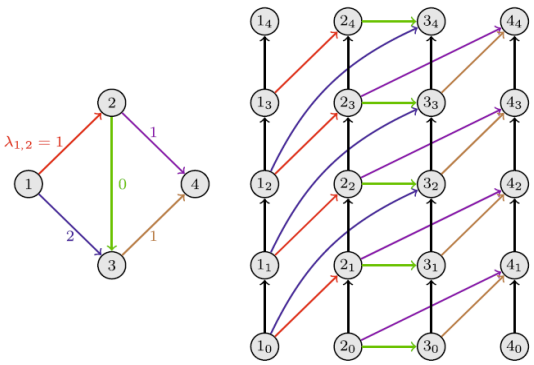

Fig. 1 on the *left hand side* a network G with transit times on the arcs is given. Let $T = 4$ be the time horizon and $\varphi = \{0, 1, 2, 3, 4\}$ be a valid partition. On the *right hand side* the corresponding time-expanded network $G(\varphi)$ with respect to the partition φ is depicted.[16]

Lemma 1 *If network G contains no dynamic cycle, then there are exists a set of labels d_i which satisfies the shortest path optimality conditions (1) and (2).*

The proof of this lemma relies on the concept of time-expanded networks which we introduce before.

Proof-We suppose that $\varphi = \{t_0, t_1, \ldots, t_p\}$ is a valid partition for the discrete dynamic shortest path problem. Consider the time-expanded network G(φ). Let $d_i(t_q)$ denote the cost of the shortest path from *node-time pair* (1,0) to *node-time pair* (i, t_q). in G(φ), for each $i \epsilon V$ and each $t_q \epsilon \varphi$.We note, from Sect. 5.2 in [17], that the labels $d_i(t_q)$ are well defined and satisfy the following conditions:

$$d_i(t_q) \le d_i(t_{q-1}) + \sum_{t_{q-1}}^{t_q} f_i(t) \quad , i\epsilon V \, , t_q \epsilon \varphi, \qquad (I)$$

$$d_i(t_q + \lambda_{i,j}) \le d_i(t_q) + c_{ij}(t_q) \qquad , (i,j)\epsilon E \, , t_q \epsilon \varphi, \qquad (II)$$

Now with respect to partition φ and conditions (I) and (II) hold for each time step $t_q \epsilon \varphi$, we can easily check that the labels d_i satisfy the shortest path optimality conditions(1) and (2). This completes the proof of the lemma.■
We wish to determine a shortest dynamic path from node-time pair (1, 0) to every other node-time pair (*i, t*).

4. Label Correcting Algorithm

A label correcting algorithm is iterative and assigns tentative distance labels to nodes at each step.The time complexity of the algorithm is *O* (|*V*||*E*|) [9]. Note that, for the network G^T, $|V^T| = nT$ and $|A^T| = mT$. According to the definition of time expanded network for a dynamic network, G = (V, A, T), the time complexity of the algorithm is *O* (|nT || mT |). The distance labels are estimates of (i.e. lower bounds on) the shortest path distances and are considered as temporary until the final step where $d(i,t_\alpha)$is the shortest path length from the source *node-time pair* (1,0) to *node-time pair* (i,t_α).The algorithm maintains a *LIST* of nodes with the property that if there is an *arc-time pair* $((i,t_\alpha),(j,t_\beta))$ for which $d(j,t_\beta) < d(i,t_\alpha) + C_{ij}(t_\alpha)$ then *LIST* must contain *node-time pair* (i, t_q) The algorithm runs as follows:

Step 1. Set the distance labels for the nodes as follows:

$$\begin{cases} d(1,0)=0 \\ d(1,1)=\ldots\ldots=d(1,T)=\infty \\ d(i,t_\alpha)=\infty \, , \quad 0 = t_0 \le \ldots\ldots \le t_{q-1} \le t_q = T \, , \; i=2,3,\ldots,n \, , \; 0 \le \alpha \le \beta \le q \end{cases}$$

Initialise the list of nodes *LIST*= [(1,0)].

Step 2. If *LIST* is empty, go to Step 5. Select the first node (i,t_α) from *LIST* .Delete (i,t_α) from *LIST*.

Step 3. If (i,t_α) has no uncorrected successors (successors for which the distance label has not been corrected), go to Step 2. Otherwise, select an arbitrary uncorrected successor of (i,t_α) . where the successor is $(i,t_{\alpha+1})$ or (j,t_β) .

Step 4. If uncorrected successor of (i,t_α) be (j,t_β) then $d(j,t_\beta) = \boldsymbol{min}\{ d(j,t_\beta), C_{ij}(t_\alpha) + d(i,t_\alpha)\}$and if the distance label $\Pi(j,t_\beta)$ has changed and $(j,t_\beta) \notin$ *LIST* add node (j,t_β) to the end of *LIST* and go to Step 3.

If uncorrected successor of (i,t_α) be $(i,t_{\alpha+1})$ then $d(i,t_{\alpha+1}) = \boldsymbol{min}\{ d(i,t_{\alpha+1}), f_{i_\alpha}(t_\alpha) + d(i, t_\alpha)\}$. If the distance label $d(i,t_{\alpha+1})$ has changed and $(i,t_{\alpha+1}) \notin$ *LIST* add node $(i,t_{\alpha+1})$ to the end of *LIST* and go to Step 3.

Step 5. Stop.

The algorithm assumes the presence of a single starting node to start the calculations in *Step* 1.

We can store all nodes whose distance labels change during a pass, and consider (or examine) only those nodes in the next pass. One plausible way to implement this approach is to store the nodes in a list whose distance labels change in a pass and examine this list in the first-in, first-out (FIFO) order in the next pass. If we follow this strategy in every pass, the resulting implementation is provided that we maintain LIST as a queue (i.e., select nodes from the front of LIST and add nodes to the rear of LIST).

5. Conclusion

In this paper we considered the dynamic shortest path problem, motivated by its applications in dynamic minimum cost flows. We showed that this problem is equivalent to a classical shortest path problem in a so-called time-expanded network. Although our approach allows us to apply any standard technique on the time-expanded network, the size of this network is typically very large for realistic problems and it may be beneficial to avoid such explicit expansion. We used the Label Correcting Algorithm for solving this problem that the time complexity of the algorithm is *O* (|nT || mT |).

References

[1] X. Cai, T. Kloks, C.K. Wong, Time-varying shortest path problems with constraints, Networks 29 (3) (1997) 141–149.
[2] E.S. Dreyfus, An appraisal of some shortest path algorithms, Operations Research 17 (1969) 395–412.
[3] H. Frank, Shortest paths in probabilistic graphs, Operations Research 17 (1968) 583–599.
[4] L. Fu, Real-time vehicle routing and scheduling in dynamic and stochastic networks, Ph.D. Thesis at the University of Alberta, 1996.
[5] A. Orda, R. Rom, Distributed shortest-path protocols for time-dependent networks, Distributed Computing 10 (1)(1996) 49–62.
[6] A. Orda, R. Rom, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, Journal of ACM 37 (3) (1990) 607–625.
[7] E.W. Dijkstra, A note on two papers in connection with graphs, Numeriske Mathematics 1 (1959) 269–271.
[8] R.W. Floyd, Algorithm 97: Shortest paths, Communications of the ACM 5 (1962) 345.
[9] Ahuja, R.K., Orlin, J.B., Pallottino, S., Scutella, M.G.: Dynamic Shortest Paths Minimizing Travel Times and Costs. Networks 41, 197–205 (2003) .
[10] Chabini, L.: Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. Transp. Res. Rec. 1645, 170–175 (1998).
[11] Orda, A., Rom, R.:Minimum weight paths in time-dependent networks. Networks 21, 295–320 (1991).
[12] Pallottino, S., Scutella, M.G.: Shortest path algorithms in transportation models: Classical and innovative aspects. In: Marcotte, P., Nguyen, S. (eds.) Equilibrium and advanced transportation modelling, pp. 245–281. Kluwer, Norwell (1998).
[13] Philpott, A.B., Mees, A.I.: Continuous-time shortest path problems with stopping and starting costs. Appl. Math. Lett. 5, 63–66 (1992).
[14] Philpot, A.B., Mees, A.I.: A finite-time algorithm for shortest path problems with time-varying costs. Appl. Math. Lett. 6, 91–94 (1993).
[15] S. Mehdi Hashemi , Shaghayegh Mokarami , Ebrahim Nasrabadi .:Dynamic shortest path problems with time-varying costs. Optim Lett 4,147–156(2010).
[16] Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Prentice-Hall, Inc., New Jersey (1993).