International Journal of Scientific and Innovative Mathematical Research (IJSIMR)
Volume 3, Issue 3, March 2015, PP 47-55
ISSN 2347-307X (Print) & ISSN 2347-3142 (Online)
www.arcjournals.org

The Dynamic Shortest Path Problems of Minimum Cost Length Time-Windows and Time-Varying Costs

Nasser A. El-Sherbeny

Mathematics Department, Faculty of Science, Al-Azhar University, (11884)-Nasr City, Cairo, Egypt
Mathematics Department, Faculty of Applied Medical Science, Taif University, Turabah, KSA
nasserelsherbeny@yahoo.com

Abstract: *This paper presents a new version of the dynamic shortest path problem. Given* $G=(V,A)$ *be directed a graph, where V is a set of vertices, A is a set of arcs, each arc* $e=(i,j)\in A$ *has a transit time* λ_{ij} *. The study concerns the problem of finding shortest paths from one vertex to all other vertices in networks for which costs can vary with time.Each vertex* $i\in V$ *has a time-window* $[a_i,b_i]$ *within which the vertex may be visited and waiting with a corresponding time-varying cost is allowed at the vertices. The transit times* λ_{ij} *can also take negative values. A general labeling method, as well as several implementations, are presented for finding theshortest path and detecting negative cycles under the assumption that arc traversal costs are piecewise linear and vertex waiting costs are piecewise constant.*

Keywords: *Shortest path problem, time-varying networks, time-windows, labeling algorithms, minimum cost length.*

1. INTRODUCTION

One of the most studied problems in graph algorithms is the shortest path problem. The shortest path problem is one among the most studied network optimization problems. This problem has applications in a large number of fields and arises as a stand-alone model in routing problem whenever we want to send some material between two specified points in a network as quickly or as cheaply as possible. Solution approaches for classical shortest path problem use some labeling procedure and are divided into two classes: label-setting and label-correcting. Label-setting algorithm can be applied only on acyclic networks and networks with nonnegative costs, whereas label-correcting algorithms are more general and applicable for all classes of problems. A complete discussion and comparison between these algorithms can be found in[1].

Let $G=(V,A)$ be a directed graph, where V is a set of vertices, $A\subseteq V\times V$ a set of arcs. Each arc $e=(i,j)\in A$ has an associated transit time λ_{ij}, which specifies the amount of time needed to travel through arc (i,j). 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:

- each vertex $i\in V$ has a time windows $[a_i,b_i]$ where, $a_i\le t_i\le b_i$, $t_i\in T\in\Re^+$, is a services vertex time see,[2,3, 4, 5, 6] and [7],
- each arc $e=(i,j)\in A$ has a transit time λ_{ij} which specifies the amount of time to traverse through each arc,
- waiting is permitted at the vertices of the network for later departure,
- Network characteristics such as arc transit and costs length can change over time and are known for all values of time.

The problem was first introduced in [8], who present a solution algorithm based on the Bellman's principle of optimality. In the model studied by [8], time is described into steps of unit length, leading to a discrete time model. This model suffers from a serious drawback: the times at which "decisions" have been made fixed in advance before the problem is solved. For many applications, this is a necessary feature of the problem. This is where the continuous time model comes into play, which allows decisions to be made at any arbitrary point in time. The general properties and algorithms have been discussed in both discrete time and continuous time settings in [9, 10, 11, 12, 13,14,15, 16, 17], and [18,19] among others.

In all of the work done so far on the subject in a continuous time model, it is assumed that the transit times are strictly positive. This assumption is not so restrictive in direct applications of the dynamic shortest path problem where arc transit times are positive. However, as for the classical shortest path problem, the dynamic shortest path problem arises as a sub-problem in developing algorithms, such as negative cycle canceling algorithm, for dynamic minimum cost flows in a continuous time framework see,[20]. As mentioned alreadyby[20], the most imported task in implementing a negative cycle canceling algorithm for dynamic minimum cost flows is how to check whether or not there exists a negative dynamic cycle in a residual network. Moreover, if such a dynamic cycle exists, then how to detect it. The residual network may contain arcs with associated negative transit times. Hence we cannot use the results in the literature for the dynamic shortest path problem since the assumption that all transit times are non-negative or strictly positive is central to all of them.

This paper studies a new version of the dynamic shortest path problem with time-windows andtime-varying costs in a continuous time setting, but in contrast to earlier work, now with possibly negative transit times. It this study shows that the problem is reduced to a classical shortest path problem on a so-called time-expanded network. This allows us to apply algorithm that are available in the classical case to the dynamic case.

The remainder of this paper is organized as follows: In section 2, we first give the problem formulation and basic concepts of the dynamic shortest path problem with time-windows and time-varying costs. In section 3, we then outline a generic labeling algorithm for solving this problem and present several special implementation of the generic algorithm under some assumption on the nature of the problem date. This work is terminated by a recommendation that the negative dynamic cycle detection problem needs further research. Finally, some concluding and discuss the related problems for future research in section 4.

2. PROBLEM FORMULATION AND BASIC CONCEPT

Given a directed network $G=(V,A)$, where $V=\{1,2,\ldots,n\}$ is a set of vertices and $A\subseteq V\times V$ is a set of arcs. Let m denote the number of arcs in network G, i.e., $m=|A|$. To simplify notation, we assume without loss of generality that every pair of vertices is connected by the most one arc. Each arc $e=(i,j)\in A$ has an associated transit time λ_{ij}, which specifies the amount of time needed to travel through arc (i,j). More precisely, if a traffic leaves vertex i at time t_i, along the arc (i,j), then it arrives at vertex j at time $t_i+\lambda_{ij}$, where the time-windows of two vertices i and j are $[a_i,b_i],[a_j,b_j]$ respectively, and $a_i\le t_i\le b_i, a_j\le t_j\le b_j$, $t_i,t_j\in T\in\Re^+$, see Figure 1.

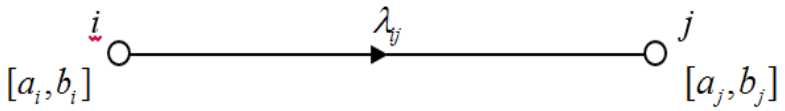

Figure 1. *Vertex service withtime-windows*

Definition 2.1. Given a fixed time horizon $T\in\Re^+$. A vertex-time pair to be a member of $V\times[0,T]$. A continuous-time dynamic path from a vertex-time pair (i,α) to a vertex-time pair (j,β) is a sequence of distinct vertex-time pair as:

$$p:(i,\alpha)=(i_1,t_1),(i_2,t_2),\ldots,(i_q,t_q)=(j,\beta) \quad (1)$$

in which either $(i_k, i_{k+1}) \in A$ where $a_{i_k} \le t_{i_k} \le b_{i_k}$ and $t_{i_k} + \lambda_{i_k, i_{k+1}}$, in which case traffic leaves vertex i_k for vertex i_{k+1} at time t_{i_k} and arrives at $t_{i_{k+1}}$, or $i_k = i_{k+1}$, in which case a waiting occurs at vertex i_k during the time interval $[t_{i_k}, t_{i_{k+1}}]$. Such a sequence is called a continuous-time dynamic cycle if $(i, \alpha) = (j, \beta)$ and the other a vertex-time pair are distinct.

The cost of a dynamic path *P* is defined by:

$$\min[\sum_{\substack{k:(i_k, i_{k+1}) \in A \\ a_{i_k} \le t_{i_k} \le b_{i_k}}} c_{i_k, i_{k+1}}(t_{i_k}) + \sum_{\substack{k:i_k = i_{k+1} \\ a_{i_k} \le t_{i_k} \le b_{i_k}}} \int_{t_{i_k}}^{t_{i_{k+1}}} w_{i_k}(t_{i_k}) dt] \tag{2}$$

where the traversal cost is $c_{ij}(t)$ along arcthe (i, j) at time t, and $w_i(t)$ is the waiting cost of the vertex i at time t. Without ambiguity, throughout the rest of this paper, we assume that the length of the path is equal to its cost and use interchangeably the terminologies cost and length. A path P issaidtobeadynamicshortestpathfrom (i, α) to (j, β) if $Cost(P) \le Cost(P')$ for all dynamic path P' from (i, α) to (j, β). A dynamic cycle is said to be negative if its cost is negative.

3. LABELING ALGORITHMS

We are now in a position to state the Continuous-Time Dynamic Shortest Path (CTDSP) problem: we wish to determine a shortest dynamic path time-windowsand time-varying costs from a vertex-time pair $(0,1)$ to every other a vertex-time pair (i, t). We suppose the following condition holds:

- **Condition:** The network $G = (V, A)$ contains a dynamic path time-windows and time-varying costs from a vertex-time pair $(0,1)$ to every other a vertex-time pair (i, t).

This condition imposes no loss of generality. In particular, it can be satisfied by introducing artificial arcs $(1, i)$ joining vertex 1 to vertex i for each vertex $i \in V \setminus \{1\}$. 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 shortest dynamic path time-windows and time-varying costs from $(1,0)$ to any a vertex-time pair (i, t) unless network G contains no dynamic path time-windows and time-varying costsfrom $(1,0)$ to (i, t) without artificial arcs.

We now present a generic algorithm based on a labeling method for solving the CTDSP problem. The basic idea is to find out from a vertex-time pair $(1,0)$ and label othera vertex-time pair according to their distances from $(1,0)$. The algorithm maintains a distance label $d_i(t)$ with each (i, t), which is an upper bound on the length of the shortest dynamic path time-windows and time-varying costs to (i, t). At any point of the algorithm, the label $d_i(t)$ is either ∞, indicating that we did not yet discover any dynamic path time-windows and time-varying costs from $(1,0)$ to a vertex-time pair (i, t), or it is the length of some dynamic path. Before proceeding our discussion, let us give the necessary and sufficient conditions for a set of labels to represent the length of shortest dynamic paths time-windows and time-varying costs.

Theorem 3.1. For any a vertex-time pair (i, t) and all vertices satisfy the time-windows and time varying-costs, let $d_i(t)$ denote the length of some dynamic path time-windows and time-varying costs from a vertex-time pair $(1,0)$ to a vertex-time pair (i, t). Then the labels $d_i(t)$ represent the length of shortest dynamic paths time-windows and time varying-costs if and only if they satisfy the following Shortest Path Optimality Conditions (SPOC):

$$d_i(t) + \int_t^T w_i(t_i) dt \text{ is monotonic decreasing on } [0, T], a_i \le t \le b_i, \text{ for every } i \in V \tag{3}$$

$$c_{ij}(t) + d_i(t) - d_j(t + \lambda_{ij}) \ge 0 \text{ for every } (i, j) \in A \text{ and } a_i \le t \le b_i,\ t \in [0, T] \tag{4}$$

Proof: It is clear that if the labels $d_i(t)$ are the length of shortest augmenting paths, they satisfy the optimality conditions (3) and (4). So we assume that for anya vertex-time pair (i,t), labels $d_i(t)$ is the length of some dynamic path time-windows and time-varying costs from $(1,0)$ to (i,t) satisfying conditions (3) and (4). Thus $d_i(t)$ is an upper bound on the length of the shortest dynamic path time-windows and time-varying costs from $(1,0)$ to (i,t). We show that $d_i(t)$ is also a lower bound on the length of the shortest dynamic pathtime-windows and time-varyingcosts from $(1,0)$ to (i,t), which implies the conclusion of the theorem. Consider an arbitrary the dynamic shortest path time-windows and time-varying costs $p:(1,0)=(i_1,t_1),(i_2,t_2),...,(i_q,t_q)=(i,t)$ from $(1,0)$ to (i,t). To simplify notation, we assume without loss of generality that $i_k=i_{k+1}$, $a_{i_k}\le t_{i_k}\le b_{i_k}$ for only one *k*, say *l*, $(1\le l\le q-1)$ and $(i_k,i_{k+1})\in A$ for all other $k=1,2,\ldots,q$. This means that the waiting only occurs at node i_l from time t_l to t_{l+1}. Hence we have $t_{i_{k+1}}=t_{i_k}+\lambda_{i_k,i_{k+1}}$ for $k=1,\ldots,l-1,l+1,\ldots,q$. Conditions (3) and (4) imply that, for $a_{i_k}\le t_{i_k}\le b_{i_k}, t_k\in[0,T]$

$$d_i(t)=d_{i_q}(t_{i_q})\le d_{i_{q-1}}(t_{i_{q-1}})+c_{i_{q-1},i_q}(t_{i_{q-1}})\le d_{i_{q-2}}(t_{i_{q-2}})+c_{i_{q-2},i_{q-1}}(t_{i_{q-2}})+c_{i_{q-1},i_q}(t_{i_{q-1}})$$
$$\vdots$$
$$\le d_{i_{l+1}}(t_{i_{l+1}})+\sum_{k=l+1}^{q-1}c_{i_k,i_{k+1}}(t_{i_k})$$
$$\le d_{i_l}(t_{i_l})+\int_l^{t_{l+1}}w_{i_l}(t_{i_l})d(t)+\sum_{k=l+1}^{q-1}c_{i_k,i_{k+1}}(t_{i_k})$$
$$\vdots$$
$$\le d_{i_1}(t_{i_1})+c_{i_1,i_2}(t_{i_1})+\sum_{k=2}l-c_{i_k,i_{k+1}}(t_{i_k})+\int_l^{t_{l+1}}w_{i_l}(t_{i_l})d(t)+\sum_{k=l+1}^{q-1}c_{i_k,i_{k+1}}(t_{i_k})=Cost(P) \quad (5)$$

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

Having proved Theorem 1, the algorithm starts by setting $d_1(0)=0$ and $d_i(t)=\infty$ for each other a vertex-time pair. It then proceeds by checking the shortest path optimality conditions (3) and (4) and updating the labels through a dynamic programming approach. In particular, at every iteration, the algorithm selects either:

- A vertex $i\in V$ and some time interval $[u,v)\subseteq[0,T]$ such that $d_i(t)+\int^T w_i(0)d0$ is strictly increasing on [*u*,*v*], and sets $d_i(t)=d_i(u)+\int_u w_i(t_i)dt$ for all $t\in[u,v)$, or
- An arc $(i,j)\in A$ and some time $t\in[0,T]$ such that $d_j(t+\lambda_{ij})>c_{ij}(t)+d_i(t)$,and sets $d_j(t+\lambda_{ij})=c_{ij}(t)+d_i(t)$, for i,j satisfies the time windows condition.

If no such vertices or arcs exist, then the algorithm terminates. Algorithm 1 specifies the generic version of this procedure.

Algorithm 1:

Initial distance labels *d* as
$d_i(t):=\infty\ \forall i\in V, t\in[0,T]$
$d_1(0):=0$
while labels *d* violate the shortest path optimality conditions **do**
if $d_i(t)+\int^T w_i(t)dt$ is strictly increasing on some interval $[u,v)\subseteq[0,T]$ for some $i\in V$

then

$d_i(t) = d_i(u) + \int_u^t w_i(t)dt$ for all $t \in [u,v)$, $a_i \le t \le b_i$,

end if

if $d_j(t+\lambda_{ij}) \succ c_{ij}(t) + d_i(t)$, for some $(i,j) \in A$ and $t \in [0,T]$ **then**

$d_j(t+\lambda_{ij}) = c_{ij}(t) + d_i(t), a_i \le t_i \le b_i, t \in [0,T]$

end if

end while

Theorem 3.1 implies that when the algorithm terminates, it has found optimal labels. More precisely, at termination, if the label $d_i(t)$ is finite, it represents the cost of the shortest dynamic path time-windows and time-varying costs from a vertex-time pair $(1,0)$ toa vertex-time pair (i,t); otherwise it indicates that the network G contains no dynamic path time-windows and time-varying costs from $(1,0)$ to (i,t). Hence termination of algorithm 1 after a finite number of iteration deserves attention from both theoretical and computational points of view. If network G has a dynamic cycle with negative cost, then Algorithm 1 may never terminate as a consequence of the following lemma.

Lemma 3.2. Suppose that network $G = (V, A)$ contains a dynamic cycle with negative cost. Then no set of labels d_i satisfies the shortest path time-windows and time-varying costs optimality conditions (3) and (4).

Proof: We suppose the opposite and derive a contradiction. Assume that there are some labels d_i satisfying the conditions (3) and (4). Then, we can show, by a similar argument as the proof of Theorem 3.1, that the conditions (3) and (4) imply that $Cost(w) \ge 0$ for each dynamic cycle *w*. This is a contradiction to the fact that network G has a dynamic cycle with negative cost due to the hypothesis of the lemma 3.2.

In what follows, we show that the opposite direction of Lemma 3.2 is also true for the special case that the traversal costs c_{ij} are piecewise linear and the waiting costs s_i are piecewise constant.

Definition 3.3. Let f be a real-valued function defined on the time interval $[0,T]$ and $\Omega = \{t_0,..., t_p\}$ be a partition of $[0,T]$ i.e., $0 = t_0 < t_1 < ... < t_p = T$. We say that f is piecewise constant with respect to the partition Ω, if it is constant on $[t_{q-1}, t_q)$ for $q = 1,2,\ldots,p$. We say that f is piecewise constant on $[0,T]$ if it is piecewise constant with respect to some partition of $[0,T]$. The breakpoints of a piecewise constant or piecewise linear function are the discontinuity points in the function or its derivatives. Thus a piecewise constant function clearly is discontinuous at the breakpoints, but a piecewise linear function itself may not be discontinuous at the breakpoints.

Definition 3.4. Let β denote the set of breakpoints of $c_{ij}, (i,j) \in A$. A partition $\Omega = \{t_0,.., t_p\}$ of $[0,T]$ is said to be valid for the CTDSP problem if it contains the set β, and for each arc (*i*,*j*) and any breakpoints $t_q \in \Omega$ with $t_q \pm \lambda_{ij} \in [0,T]$, we have $t_q \pm \lambda_{ij} \in \Omega$. A valid partition of minimum cardinality is denoted by Ω^*.

Definition 3.5. Given a valid partition $\Omega = \{t_0, t_1,..., t_p\}$, a time-expanded networkof G, denoted by $G(\Omega)$, is defined as follows: $G(\Omega)$ contains $p+1$ copies of N denoted by $N_0, N_1,..., N_p$, in which N_{q-1} corresponds to the time interval $[t_{q-1}, t_q)$ for $q = 1,\ldots,p-1$, and N_p to the time horizon T. Subsequently, index q varies from 1 to P. The copy of vertex $i \in N$ in N_{q-1} is

denoted by i_{q-1}. For each arc $(i,j) \in A$ and each time $t_{q-1} \in \Omega$ with $0 \le t_{q-1} + \lambda_{ij} \le T$, the network $G(\Omega)$ contains an arc $(i_{q-1}, j_{q'})$, where $t_{q'} = t_{q-1} + \lambda_{ij}, a_{i_{q-1}} \le t_{i_{q-1}} \le b_{i_{q-1}}$.Traversing through this arc corresponds to leaving vertex i at time t_{q-1} and arriving at vertex j at time $t_{q'}$. Hence, arc (i_{q-1}, j_q) has an associated cost $c_{ij}(t_{q-1})$. For each vertex*i*, there is a holdover arc from i_{q-1} to i_q. Travelling through arc (i_{q-1}, i_q) corresponds to the waiting at vertex*i* from time $t_{i_{q-1}}$ to t_{i_q}. So holdover arc (i_{q-1}, i_q) has an associated cost $s_i(t_{i_{q-1}})$ where, $a_{i_{q-1}} \le t_{i_{q-1}} \le b_{i_{q-1}}$.

Remark 3.6. The traversal costs c_{ij} are piecewise linear for all $(i,j) \in A$and the traversal costs s_i for all $i \in N$ are piecewise continuous from right.

Lemma 3.7. If network $G = (V, A)$ contains no negative dynamic cycle, then there exist a set of labels d_i which satisfy the shortest path time windows optimality conditions (3) and (4).

Proof: We suppose that $\Omega = \{t_0, t_1, ..., t_p\}$ is a valid partition for the CTDSP problem. Consider the time-expended network $G(\Omega)$.It is clear that $G(\Omega)$ has no negative cycle since network G contains no dynamic cycles with negative cost due to the hypothesis of the lemma. Let $d_i(t_q)$ denote the cost of the shortest path time windows from vertex $(1,0)$ to vertex (i,t_q) in $G(\Omega)$, for each $i \in V$ and each $t_q \in \Omega$. By [1],it is noted that the labels $d_i(t_q)$ are well defined and satisfy the following conditions:

$$d_i(t_q) \le d_i(t_{q-1}) + \int_{t_{q-1}}^{t_q} w_i(t)dt, i \in V, t_q \in \Omega \quad (6)$$

$$d_j(t_q + \lambda_{ij}) \le d_i(t_q) + c_{ij}(t_q), (i,j) \in A, t_q \in \Omega \quad (7)$$

We now define the labels $d_i(t)$ for all $t \in [0,T]$ by

$$d_i(t) = \gamma d_i(t_q) + (1-\gamma) d_i(t_{q-1}) \text{ for } t_{q-1} \le t \prec t_q \text{ where } \gamma = (t - t_{q-1})/(t_q - t_{q-1}) \quad (8)$$

Now by the fact that distance labels d_i are piecewise linear function with respect to partition Ω and conditions (6) and (7) hold for each time step $t_q \in \Omega$, $a_q \le t_q \le b_q$, we can easily check that the labels d_i, given by (8), satisfy the shortest path time windows optimality conditions (3) and (4).

Theorem 3.8. Let the network $G = (V, A)$ contains no negative dynamic cycle. Then shortest dynamic paths time-windows and time-varying costs from $(1,0)$ to all a vertex time pair (i,t), $a_i \le t \le b_i, t \in [0,T]$ can be found in $O(|\Omega^*|^2 nm)$ time.

Proof: Consider the time-expanded network $G(\Omega^*)$ with respect to the valid partition Ω^*. By the hypothesis of the theorem, we conclude that the network $G(\Omega^*)$ contains no negative cycles. Hence, by applying the First-In-First-Out(FIFO) label-correcting algorithm [1] in the time-expanded network $G(\Omega^*)$, we can find shortest paths time-windows and time-varying costs from vertex $(1,0)$ to all other vertices (i,t_q) with $a_q \le t_q \le b_q$ in $O(|\Omega^*|^2 nm)$ time. We now define the labels $d_i(t)$ for all $t \in [0,T]$ by

$$d_i(t) = \gamma d_i(t_q) + (1-\gamma) d_i(t_{q-1}) \text{ for } t_{q-1} \le t \prec t_q \text{ for}, \gamma = (t - t_{q-1})/(t_q - t_{q-1}) \quad (9)$$

As we have seen before in the proof of lemma 3.7, the labels d_i satisfy the optimality conditions (3) and (4). Moreover, for each $t \in [0,T]$, it can easily be shown that $d_i(t)$ represents the cost of some dynamic path time-windows in network G. Hence by Theorem 1, $d_i(t)$ is the length of the shortest dynamic pathtime-windows and time-varying costs from a vertex-time pair $(1,0)$ to a vertex-time pair (i,t).

The following two corollaries show that the running time $O(|\Omega^*|^2 nm)$ can be improved to $O(|\Omega^*|m)$ and $O(|\Omega^*|m + |\Omega^*| n \log(|\Omega^*|n))$ on networks with strictly positive transit times, for each vertex has the time-windows, and the network with the non-negative costs.

Corollary 3.9. Suppose that the transit times λ_{ij} are strictly positive. Thenthedynamic shortest paths time-windows and time-varying costs from $(1,0)$ to all a vertex time pair (i,t), $a_i \le t \le b_i, t \in [0,T]$ can be found in $O(|\Omega^*|m)$ time.

Proof: It is clear that network G contains no dynamic cycle since all transit time are strictly positive. Hence, we can apply the reaching algorithm [1]in the time-expanded network $G(\Omega^*)$. This leads to an algorithm of time $O(|\Omega^*|m)$ for the CTDSP problem.

Corollary 3.10. Suppose that the traversal costs c_{ij} and the waiting costs s_i are non-negative. Then shortest dynamic paths time-windows and time-varying costs from $(1,0)$ to all a vertex time pair (i,t), $a_i \le t \le b_i, t \in [0,T]$ can be found in $O(|\Omega^*|m + |\Omega^*| n \log(|\Omega^*|n))$ time.

Proof: We can apply the Fibonacci heap implementation of Dijkstra's Algorithm [1] in the time-expanded network $G(\Omega^*)$, leading to an algorithm with running time

$O(|\Omega^*|m + |\Omega^*| n \log(|\Omega^*|n))$ for the CTDSP problem.

So far we have considered that case where network G has no negative dynamic cycle. We now consider the more general case where network G may contain a negative dynamic cycle. The problem is to find a negative dynamic cycle in network G or to prove that there are none. This problem comes up as a sub-problem in algorithms for dynamic minimum cost flow problems with time windows andtime-varying parameters see, [5]. In the context of the classical shortest path problem, there are several algorithms for detecting the presence of a negative cycle if one exists. Most of them combine a shortest path algorithm and a negative cycle detection strategy. In [21] survey cycle detection strategies, study various combinations of shortest path algorithms and cycle detection strategies and find the best combinations. All of the results presented in [21] can be carried out on the time-expanded network $G(\Omega^*)$ to derive analogous results to the CTDSP problem. As a general result, we can state the following theorem.

Theorem 3.11. If network G contains a negative dynamic cycle reachable from a vertex time pair $(1,0)$, then the above algorithm finds such a dynamic cycle in $O(|\Omega|^2 nm)$ time.

Proof: Consider the time-expanded network $G(\Omega^*)$. We can show that $G(\Omega^*)$ contains a negative cycle by the fact that network G has a negative dynamic cycle. Then by the FIFO lable-correcting algorithm [1], we can find a negative cycle in $G(\Omega^*)$ in $O(|\Omega|^2 nm)$ time. This negative cycle corresponds a dynamic cycle in the original network of equal cost.

4. CONCLUSION

This paper presented a new version of the dynamic shortest path time-windows and time-varying costs with possible negative transit time on arcs, each vertex has a time-windows interval within

which the vertex may be visited, each vertex has visited exactly once, 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 restricted our attention to the cases that are traversal costs are piecewise linear and vertex waiting costs are piecewise constant. According to the more general setting of the cost functions should be piecewise analytic, this is important to the topic. This case requires a complicated argument and further work.

ACKNOWLEDGEMENTS

The author would like to thank an anonymous referee for his careful readings.

REFERENCES

[1]. Ahuja K., Magnanti L., Orlin B., Network Flows: Theory, Algorithms, and Applications, Prentice-hall, Inc. New Jersey, (1993).

[2]. El-Sherbeny N., Resolution of a vehicle routing problem with a multi-objective simulated annealing method, Ph.D. dissertation, Faculty of Science, Mons University, Mons, Belgium. pp. 205, (2002).

[3]. El-Sherbeny N., Tuyttens D., Optimization multicriteria of ruoting problem, Troisieme Journee de Travail sur la Programming Mathematique Multi-Objective, Faculte Polytechnique de Mons, Mons, Belgique, (2001).

[4]. El-Sherbeny N., The Algorithm of the Time-Dependent Shortest Path Problem with Time Windows, Applied Mathematics. 5(17), 2764-2770, (2014). http://dx.doi.org/10.4236/am.2014.517264

[5]. El-Sherbeny N., Imprecision and flexible constraints in fuzzy vehicle routing problem, American Journal of Mathematical and Management Sciences.(31) 55-71, (2011). http://dx.doi.org/10.1080/01966324.2011.10737800

[6]. El-Sherbeny N., Minimum Convex and Differentiable Cost Flow Problem and Time Windows, International Journal of Sciences: Basic and Applied Research. 20(1), 139-150, (2015). http://gssrr.org/index.php?journal=JournalOfBasicAndApplied

[7]. Tuyttens D., Teghem J., El-Sherbeny N., A particular multiobjective vehicle routing problem solved by simulated annealing, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, Berlin, Germany, 535: pp. 133-152, (2004). http://dx.doi.org/10.1007/978-3-642-17144-4

[8]. Cooke L., Halsey E., The shortest route through a network with time-depended inter-model transit time, J. Math. Anal. Appl. (14)492-498, (1966).

[9]. Ahuja K., Orlin B., Pallottino S., Scutella G., Dynamic Shortest Paths Minimum Travel Time and Costs, Networks.(41)197-205, (2003).

[10]. Cai X., Kloks T., Wong K., Time-varying shortest path problems with constraints, Networks.(29)141-149, (1997).

[11]. Chabini L., Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time, Transp. Res. 16(45)170-175,(1998).

[12]. Dreyfus E., An appraisal of some shortest-path algorithms, Oper. Res. (17) 395-412, (1969).

[13]. Kaufman E., Smith L., Fastest paths in time-dependent networks for intelligent vehicle-highway systems application, IVHSJ,(1)1-11, (1993).

[14]. Orda A., Rom R., Shortest-path and minimum-delay algorithms in networks with time-dependent edge length, J. Assoc. Comput. Mach. (37) 607-625, (1990).

[15]. Orda A., Rom R., Minimum weight paths in time-dependent networks, Networks. (21)295-320, (1991).

[16]. Pallottino S., ScutellaG., Shortest path algorithms in transportation models: Classical and innovative aspects, In: Marcotte P, Nguyen S. (eds.) Equilibrium and advanced transportation modeling. Kluwer, Norwell, 1998, pp.245-281.

[17]. Pilpoitt B., Continuous-time shortest path problems and linear programming, SIAM, J. Control Optim.(32) 538-552, (1994).

[18]. Philpott B., Mees I., Continuous-time shortest paths problems with stopping and starting costs, Appl. Math. Lett.(5)63-66, (1992).

[19]. Philpott B., Mees L., Afinite-time algorithm for shortest path problems with time varying costs, Appl. Math. Lett.(6) 91-94, (1993).

[20]. Nasrabadi E., Dynamic Flow in Time-varying Networks, Ph.D. thesis, Amirkabir University of Technology & Technische Universtat Berlin, Iran & Germany, (2009).

[21]. Cherkassky V., Goldberg V., Negative-cycle detection algorithms, Math. Program. (85) 277-311, (1999).

AUTHOR'S BIOGRAPHY

Dr. Nasser A. El-Sherbeny,is a Lecturer of Mathematics Department, Faculty of Science, Al-Azhar University, Nasr City, Cairo, Egypt. He had his Ph.D. in Mathematics from Mathematics Department, Faculty of Science, Mons University, Mons, Belgium. He is a member of the Egyptian Mathematical Society (EMS). His area of Specialization is in Network Optimization Flows.