Bulletin of the *Transilvania* University of Braşov • Vol 7(56), No. 1 - 2014
Series III: Mathematics, Informatics, Physics, 95-102

MAXIMUM FLOW OVER TIME PROBLEM IN PARAMETRIC NETWORKS

Nicoleta AVESALON (GRIGORAŞ)[1]

Abstract

The article presents the maximum parametric flow problem in discrete dynamic networks with linear capacities and zero lower bounds. Based on an approach of partitioning the values range of the parameter, the algorithm proposed for solving the problem finds the maximum flow for a sequence of the parameter values, in their increasing order. The flow is repeatedly augmented along the shortest paths from source to sink in the time-space network, avoiding the explicit time expansion of the network. In each of its iterations, besides the maximum flow, the algorithm also computes a new value of the parameter up to which the computed flow remains a maximum one. Finally, the complexity of the algorithm is computed.

2000 *Mathematics Subject Classification:* 90B10, 90C35, 90C47.
Key words: Dynamic networks, parametric flow, successive shortest paths.

1 Indroduction

The case of the maximum parametric dynamic flow in discrete dynamic networks is worth being considered not only because static (classic) network flow models (see [1]) fail to capture the dynamic property (see [2]) of many real-life problems (such as traffic planning, production and distribution systems, communication systems, and evacuation planning) but also because there are many important applied problems that can be formulated as parametric network flow problems. The partitioning approach (see [6]) proposed for solving *the maximum parametric flow over time problem* consists in iteratively finding the *quickest flow*, which minimises the transit time, and the corresponding *breakpoint* of the piecewise linear maximum dynamic flow value function.
Further on, the article is structured as follows: based on the basic dynamic networks terminology and notations which are presented in Section 2, Section 3 introduces the problem of finding the parametric maximum flow in discrete dynamic networks which is resolved via the algorithm presented in Section 4.

[1]Faculty of Mathematics and Informatics, *Transilvania* University of Braşov, Romania, e-mail: nicole_grigoras@yahoo.com

2 Dynamic networks terminology and notations

A *discrete-time dynamic network* $G = (N, A, T)$ is a directed graph where N is a set of nodes i with let $|N| = n$, A is a set of arcs a with $|A| = m$, and T is a finite time horizon discretized into the set $H = \{0, 1, \cdots, T\}$. An arc $a \in A$ from node i to node j is also denoted by (i, j). The following functions are associated with each arc $a = (i, j) \in A$: the time-dependent *upper bound (capacity)* function $u(i, j; \theta)$, $u : A \times H \to \Re^+$, which represents the maximum amount of flow that can enter the arc (i, j) at time θ, and the time-dependent *transit time* function $h(i, j; \theta)$, $h : A \times H \to \aleph$. Time is measured in discrete steps, so that if one unit of flow leaves from node i at time θ over arc $a = (i, j)$, one unit of flow arrives at node j at time $\theta + h(i, j; \theta)$, where $h(i, j; \theta)$ is the transit time over the arc (i, j). The time horizon T is the time limit up to which the flow can travel in the network. The *demand-supply* function $v(i; \theta)$, $v : N \times H \to \Re$, represents the demand of node $i \in N$ at the time-moment $\theta \in H$, if $v(i; \theta) < 0$, or the supply of node $i \in N$ at the time-moment $\theta \in H$, if $v(i; \theta) > 0$. The network has two special nodes: a source node s with $v(s; \theta) \geq 0$ for all $\theta \in H$ and $\exists \theta_0 \in H$ such that $v(s; \theta_0) > 0$; and a sink node t with $v(t; \theta) \leq 0$ for all $\theta \in H$ and $\exists \theta_1 \in H$ such that $v(t; \theta_1) < 0$. Under these circumstances, the required condition for the existence of a flow is that $\sum_{\theta \in H} \sum_{i \in N} v(i; \theta) = 0$.

The *time-space network* is a static network constructed by expanding the original network in the time dimension, by considering a separate copy of every node $i \in N$ at every discrete time step $\theta \in H$ within the time horizon T. A *node-time pair* (NTP) (i, θ) refers to a particular node $i \in N$ at a particular time step $\theta \in H$, i.e., $(i, \theta) \in N \times H$. The NTP (i, θ_1) is said to be *linked* to the NTP (j, θ_2) if either $(i, j) \in A$ and $\theta_2 = \theta_1 + h(i, j; \theta_1)$, or $(j, i) \in A$ and $\theta_1 = \theta_2 + h(j, i; \theta_2)$.

Definition 1. The *time-space network* G^T of the original dynamic network G is defined as follows:
(a) $N^T := \{(i, \theta) | i \in N, \theta \in H\}$;
(b) $A^T := \{a_\theta = ((i, \theta), (j, \theta + h(i, j; \theta))) | 0 \leq \theta \leq T - h(i, j; \theta), (i, j) \in A\}$;
(c) $u^T(a_\theta) := u(a; \theta) \qquad for \quad a_\theta \in A^T$.

Definition 2. A (discrete-time) *dynamic path* $P(i_1, i_2)$ is defined as a sequence of distinct, consecutively linked NTPs:
$P(i_1, i_2) : (i_1, \theta_1) = (i_{k_1}, \theta_{k_1}), (i_{k_2}, \theta_{k_2}), \cdots, (i_{k_q}, \theta_{k_q}) = (i_2, \theta_2)$.

3 Parametric maximum flow over time problem

Definition 3. A discrete-time dynamic network $G = (N, A, T)$ with the upper bounds $u(i, j; \theta)$ of some arcs $(i, j) \in A$ functions of a real parameter λ is referred to as a *parametric dynamic network* and is denoted by $\bar{G} = (N, A, \bar{u}, h, T)$.

For a parametric dynamic network $\bar{G}$, the *parametric upper bound* (*parametric capacity*) function $\bar{u} : A \times H \times [0, \Lambda] \to \Re^+$ associates to each arc $(i, j) \in A$ and for each of the parameter values λ in an interval $[0, \Lambda]$, the real number $\bar{u}(i, j; \theta; \lambda)$, referred to as the *parametric upper bound* of arc (i, j):

$$\bar{u}(i, j; \theta; \lambda) = u_0(i, j; \theta) + \lambda \cdot U(i, j; \theta), \quad \lambda \in [0, \Lambda], \quad \theta \in H, \tag{1}$$

where $U : A \times H \to \Re$ is a real valued function, which associates to each arc $(i,j) \in A$ and for each time-value $\theta \in H$ the real number $U(i,j;\theta)$, referred to as the *parametric part of the upper bound* of the arc (i,j). The nonnegative value $u_0(i,j;\theta)$ is the upper bound of the arc (i,j) for $\lambda = 0$, i.e., $\bar{u}(i,j;\theta;0) = u_0(i,j;\theta)$. The problem is correctly formulated if the upper bound function of every arc $(i,j) \in A$ respects the condition $\bar{u}(i,j;\theta;\lambda) \geq 0$ for the entire interval of the parameter values, i.e., $\forall (i,j) \in A$, $\forall \theta \in H$ and $\forall \lambda \in [0,\Lambda]$. It follows that $u_0(i,j;\theta) \geq 0$ and the parametric part of the upper bounds $U(i,j;\theta)$ meets the constraints: $U(i,j;\theta) \geq -u_0(i,j;\theta)/\Lambda$, $\forall (i,j) \in A$. The *parametric dynamic flow value function* $\bar{v} : N \times H \times [0,\Lambda] \to \Re$ associates to each of the nodes $i \in N$, at each time moment $\theta \in H$, a real number $\bar{v}(i;\theta;\lambda)$ referred to as the value of node i at time θ, for each of the parameter λ values.

Definition 4. A *feasible parametric flow over time* in network $\bar{G} = (N, A, \bar{u}, h, T)$ is a function $\bar{f} : N \times H \times [0,\Lambda] \to \Re^+$ that satisfies the following constraints for all $\lambda \in [0,\Lambda]$:

$$\sum_{j|(i,j)\in A} \bar{f}(i,j;\theta;\lambda) - \sum_{j|(j,i)\in A} \sum_{\vartheta|\vartheta+h(j,i;\vartheta)=\theta} \bar{f}(j,i;\vartheta;\lambda) = \bar{v}(i;\theta;\lambda), \quad \forall \theta \in H; \tag{2}$$

$$0 \leq \bar{f}(i,j;\theta;\lambda) \leq \bar{u}(i,j;\theta;\lambda), \quad \forall (i,j) \in A, \quad \forall \theta \in H; \tag{3}$$

$$\bar{f}(i,j;\theta;\lambda) = 0, \qquad \forall i \in N, \quad \forall \theta \in \overline{T - h(i,j;\theta) + 1, \quad T}; \tag{4}$$

$$\sum_{\theta\in\{0,1,\cdots,T\}} \bar{v}(s;\theta;\lambda) = -\sum_{\theta\in\{0,1,\cdots,T\}} \bar{v}(t;\theta;\lambda) = \bar{v}(\lambda); \tag{5}$$

where $\bar{v}(i;\theta;\lambda) = 0$, $\forall i \neq s,t$ and $\bar{f}(i,j;\theta;\lambda)$ determines the rate of flow (per time unit) entering arc (i,j) at time θ, for the parameter value λ, $\forall \theta \in H$ and $\forall \lambda \in [0,\Lambda]$.

Capacity constraints (3) mean that in a feasible parametric flow over time, at most $\bar{u}(i,j;\theta;\lambda)$ units of flow may enter the arc (i,j) at the time-moment θ. It is easy to observe that the flow does not enter arc (i,j) at time θ if it has to leave the arc after time T, which is ensured by condition (4).

Definition 5. The *parametric maximum flow over time*(PMFT) problem is to compute all maximum flows over time for every possible value of λ under flow constraints (2)-(5):

$$maximize \quad \bar{v}(\lambda) = -\sum_{\theta\in\{0,1,\cdots,T\}} \bar{v}(t;\theta;\lambda), \quad for \quad all \quad \lambda \in [0,\Lambda]. \tag{6}$$

For a feasible flow $\bar{f}(a;\theta;\lambda)$ in the parametric dynamic network $\bar{G}$, the function $\bar{f}^T(a_\theta;\lambda)$ represents the corresponding flow in the parametric time-space network $\bar{G}^T$ and is defined as: $\bar{f}^T(a_\theta;\lambda) = \bar{f}(a;\theta;\lambda)$, $\forall a_\theta \in A^T$.
The *parametric time-dependent residual network*, corresponding to a feasible parametric flow $\bar{f}$, represents the static parametric residual network built for the time-space network.

Definition 6. The *parametric time-dependent residual network*, defined for the maximum flow over time problem, with respect to a given feasible parametric flow over time $\bar{f}$ is defined as $\bar{G}(\bar{f}) := (N, A(\bar{f}), T)$, with $A(\bar{f}) := A^+(\bar{f}) \cup A^-(\bar{f})$, where
$A^+(\bar{f}) := \{(i,j)|(i,j) \in A, \exists \theta \leq T - h(i,j;\theta) \quad with \quad \bar{u}(i,j;\theta;\lambda) - \bar{f}(i,j;\theta;\lambda) > 0\}$;
$A^-(\bar{f}) := \{(i,j)|(j,i) \in A, \exists \theta \leq T - h(j,i;\theta) \quad with \quad \bar{f}(j,i;\theta;\lambda) > 0\}$.

While the direct arcs $(i,j) \in A^+(\bar{f})$ in the parametric time-dependent residual network $\bar{G}(\bar{f})$ have same transit times $h(i,j;\theta)$ with those in the original parametric dynamic network $\bar{G}$, the reverse, artificial arcs $(i,j) \in A^-(\bar{f})$ have negative transit times $h(i,j;\theta + h(j,i;\theta)) = -h(j,i;\theta)$, with $(j,i) \in A$ and $0 \leq \theta + h(j,i;\theta) \leq T$.

Definition 7. The *parametric residual capacities* of arcs (i,j) in the parametric time-dependent residual network $\bar{G}(\bar{f})$ defined for the maximum flow over time problem, are computed as follows:
$\tilde{\bar{r}}(i,j;\theta;\lambda) = \bar{u}(i,j;\theta;\lambda) - \bar{f}(i,j;\theta;\lambda), \quad (i,j) \in A, \quad 0 \leq \theta + h(i,j;\theta) \leq T;$
$\tilde{\bar{r}}(i,j;\theta + h(j,i;\theta);\lambda) = \bar{f}(j,i;\theta;\lambda), \quad (j,i) \in A, \quad 0 \leq \theta + h(j,i;\theta) \leq T.$

Definition 8. The subintervals $\tilde{I}(i,j;\theta) \subseteq [0,\Lambda]$ where an augmentation of the flow $\bar{f}(i,j;\theta;\lambda)$ is possible over the arc (i,j) at time θ are defined as follows:

$$\tilde{I}(i,j;\theta) = \{\lambda | (i,j) \in A, \quad \tilde{\bar{r}}(i,j;\theta;\lambda) > 0\}. \tag{7}$$

Definition 9. A *conditional augmenting dynamic path* $\tilde{P}$ is a dynamic path from the source node to the sink node $s = i_1, i_2, \cdots, i_q = t$ in the parametric time-dependent residual network $\bar{G}(\bar{f})$ with $\tilde{\bar{r}}(i_k, i_{k+1};\theta_k;\lambda) > 0$ for all $(i_k, i_{k+1}) \in \tilde{P}$, $k = 1, 2, \cdots, q-1$ and $\tilde{I}(\tilde{P};\theta) = \bigcap_{(i_k,i_{k+1}) \in \tilde{P}} \tilde{I}(i_k, i_{k+1};\theta_k) \neq \emptyset$.

Definition 10. Given a parametric flow over time $\bar{f}(i,j;\theta;\lambda)$, the *parametric residual capacity* $\tilde{\bar{r}}(\tilde{P};\theta;\lambda)$ of a conditional augmenting dynamic path $\tilde{P}$ is the inner envelope of the parametric residual capacity functions $\tilde{\bar{r}}(i,j;\theta;\lambda)$ for all arcs (i,j) composing the path and for all parameter λ values within the subinterval $\tilde{I}(\tilde{P};\theta)$:

$$\tilde{\bar{r}}(\tilde{P};\theta;\lambda) = \min_{\lambda \in \tilde{I}(\tilde{P};\theta)} \{\tilde{\bar{r}}(i,j;\theta;\lambda) | (i,j) \in \tilde{P}\}. \tag{8}$$

Definition 11. The *transit time* $\tau(\tilde{P};\theta)$ of a conditional augmenting dynamic path $\tilde{P}$ is defined by:

$$\tau(\tilde{P};\theta) = \sum_{(i,j) \in \tilde{P}} h(i,j;\theta). \tag{9}$$

4 Maximum parametric dynamic flow algorithm

The approach which will be presented below consists in successively applying a non-parametric maximum dynamic flow algorithm (see [5]) in a series of dynamic residual networks generated by a technique of partitioning the interval of the parameter values. Since the parametric residual capacities for all the arcs in $\bar{G}(\bar{f})$ are linear functions of λ, the parametric residual capacity $\tilde{\bar{r}}(\tilde{P};\theta;\lambda)$ of any conditional augmenting dynamic path $\tilde{P}$ (8) is a piecewise linear function of λ. Moreover, the subinterval $\tilde{I}(\tilde{P};\theta)$, where an augmentation of the flow is possible over $\tilde{P}$, equals the considered range of parameter values so that any dynamic path in the parametric time-dependent residual network $\bar{G}(\bar{f})$ is also a conditional augmenting dynamic path $\tilde{P}$.
In order to apply a non-parametric algorithm for a parametric flow problem it suffices that the parametric residual capacity functions of all arcs remains linear,

with no break points in the considered range of parameter values. The requirement is met at the beginning of the algorithm for the whole interval $[0, \Lambda]$ of the parameter λ values. During the running of the algorithm, the subinterval of the parameter values is to be continuously updated (narrowed) so that the above restriction remains valid. As soon as, for a subinterval of the parameter values, the parametric time-dependent residual network $\bar{G}(\bar{f})$ contains no conditional augmenting dynamic paths the algorithm computes the optimum flow for that subinterval and then reiterates on the next subinterval of the parameter values, until Λ value is reached. In fact, the algorithm repeatedly solves a series of maximum dynamic flow problems, via a *Successive Shortest Dynamic Paths* approach, for successive subintervals of the parameter values, where the residual capacities of all arcs remains linear, without crossing points (see [6]).

MPDF ALGORITHM;
(01) BEGIN
(02) $BP := \{0\}$; $k := 0$; $\lambda_k := 0$;
(03) REPEAT
(04) SDP(k, λ_k, BP);
(05) $k := k + 1$;
(06) UNTIL$(\lambda_k = \Lambda)$;
(07) END.

Algorithm 1: Maximum Parametric Dynamic Flow (MPDF) Algorithm

Theorem 1. (Theorem of correctness) *The maximum parametric dynamic flow (MPDF) algorithm computes correctly a parametric maximum flow over time for a given time horizon T and for $\lambda \in [0, \Lambda]$.*

Proof. The partitioning algorithm iterates on successive subintervals $[\lambda_k, \lambda_{k+1}]$, starting with $\lambda_0 = 0$ and ending with $\lambda_{k_{max}+1} = \Lambda$ and consequently, the correctness of the algorithm obviously follows from the correctness of the Shortest Dynamic Paths (SDP) procedure. Actually, the algorithm ends with a set of linear parametric maximum flows and with the partition BP of the interval of the parameter values in their corresponding subintervals. □

Theorem 2. (Theorem of complexity) *The maximum parametric dynamic flow (MPDF) algorithm runs in $O(Kn^2mT^3)$ time, where $K + 1$ is the number of λ values in the set BP at the end of the algorithm.*

Proof. For each of the K subintervals $[\lambda_k, \lambda_{k+1}]$, $k = 0, 1, \cdots, K-1$ in which the interval $[0, \Lambda]$ of the parameter λ values is partitioned, the algorithm makes one call to procedure SDP. Since the complexity of the SDP is $O(n^2mT^3)$ (see theorem 4), the total complexity of the maximum parametric flow over time (MPDF) algorithm is $O(Kn^2mT^3)$. □

Shortest dynamic paths (SDP) procedure repeatedly performs the following operations:
- Finds a shortest (quickest [4]) conditional augmenting dynamic path $\tilde{P}$ in $\bar{G}(\bar{f})$;
- Computes the parametric residual capacity $\tilde{\bar{r}}(\tilde{P}; \theta; \lambda)$ of path $\tilde{P}$;
- Computes the value of the parameter λ up to which the parametric residual capacity of the conditional augmenting dynamic path remains linear without breakpoints;

- Augments the flow along path $\tilde{P}$ and update $\bar{G}(\bar{f})$.
SDP procedure will terminate when none of the sink nodes, at any time moments $\theta \in \{0, 1, \cdots, T\}$, is reachable from any of the source node, which represents that there is no augmenting dynamic path from s to t. The *Shortest Dynamic Paths* (SDP) procedure is presented in Algorithm 2.

PROCEDURE SDP(k, λ_k, BP);
(01) BEGIN
(02) FOR all $\theta \in \{0, 1, \cdots, T\}$ DO
(03) BEGIN
(04) $p(s, \theta) := 0$; FOR all $i \in N - s$ DO $p(i, \theta) := -1$;
(05) FOR all $(i, j) \in A$ DO
$\bar{f}_k(i, j; \theta; \lambda) := 0$; $\alpha_k(i, j; \theta) := u_0(i, j; \theta) + \lambda_k \cdot U(i, j; \theta)$; $\beta_k(i, j; \theta) := U(i, j; \theta)$;
$\bar{f}_k(j, i; \theta; \lambda) := 0$; $\alpha_k(j, i; \theta) := 0$; $\beta_k(j, i; \theta) := 0$;
(06) END;
(07) $C := 1$; $\lambda_{k+1} = \Lambda$;
(08) DP(p, C);
(09) WHILE$(C = 1)$ DO
(10) BEGIN
(11) build $\tilde{P}$ based on p starting from $(t, \tilde{\theta})$;
(12) $\alpha := \min\{\alpha_k(i, j; \theta) | (i, j) \in \tilde{P}\}$; $\beta := \min\{\beta_k(i, j; \theta) | (i, j) \in \tilde{P},\ \alpha_k(i, j; \theta) = \alpha\}$;
(13) $(j, \vartheta) := (t, \tilde{\theta})$;
(14) WHILE$(j \neq s)$ DO
(15) BEGIN
(16) $(i, \theta) := p(j, \vartheta)$;
(17) IF$(\beta_k(i, j; \theta) < \beta)$ THEN $\lambda_{k+1} := \min\{\lambda_{k+1}, \lambda_k + (\alpha_k(i, j; \theta) - \alpha)/(\beta - \beta_k(i, j; \theta))\}$;
(18) $\alpha_k(i, j; \theta) := \alpha_k(i, j; \theta) - \alpha$; $\beta_k(i, j; \theta) := \beta_k(i, j; \theta) - \beta$;
(19) $\alpha_k(j, i; \theta) := \alpha_k(j, i; \theta) + \alpha$; $\beta_k(j, i; \theta) := \beta_k(j, i; \theta) + \beta$;
(20) IF$(\theta < \vartheta)$ THEN $\bar{f}_k(i, j; \theta; \lambda) := \bar{f}_k(i, j; \theta; \lambda) + \alpha + \beta \cdot (\lambda - \lambda_k)$
(21) ELSE $\bar{f}_k(j, i; \vartheta; \lambda) := \bar{f}_k(j, i; \vartheta; \lambda) - \alpha - \beta \cdot (\lambda - \lambda_k)$;
(22) $(j, \vartheta) := (i, \theta)$;
(23) END;
(24) FOR all $\theta \in \{0, 1, \cdots, T\}$ DO $p(s, \theta) := 0$; FOR all $i \in N - s$ DO $p(i, \theta) := -1$;
(25) DP(p, C);
(26) END;
(27) $BP := BP + \{\lambda_{k+1}\}$;
(28) END.

Algorithm 2: Shortest Dynamic Paths (SDP) Procedure

Theorem 3. (Theorem of correctness) *Procedure Shortest Dynamic Paths (SDP) computes correctly the maximum dynamic flow (of minimum transit time) for a given time horizon T.*

Proof. The procedure ends when none of the sink node-time pairs is reachable from any of the source node-time pairs, i.e. when there is no dynamic augmenting path from the source node to the sink node in the time-depending residual network. According to the classical flow augmenting path theorem this means that the obtained flow is a maximum

flow. Since every flow augmentation step is performed over a shortest (quickest) path, the obtained flow is the quickest maximum dynamic flow for the given time horizon. □

Theorem 4. (Theorem of complexity) *The complexity of Shortest Dynamic Paths (SDP) procedure is* $O(n^2mT^3)$.

Proof. The labelling operation and the updating of the time-dependent residual network requires an $O(mT)$ running time, since for each of the time values $0 \leq \theta \leq T$ all the m arcs must be examined. Considering that, in each of the iterations, for one of the time values, one arc is eliminated from the dynamic residual network; SDP procedure terminates in at most $O(mT)$ iterations. On each of the iterations the procedure finds a dynamic path (DP) with the complexity $O(n^2T^2)$ and updates the time-dependent residual network in $O(mT)$ time. Thus, the total complexity of the shortest dynamic paths (SDP) procedure is $O(n^2mT^3)$. □

PROCEDURE DP(p, C);
(02) BEGIN
(03) FOR all $\theta \in \{0, 1, \cdots, T\}$ DO
(04) BEGIN
(05) $\tau(s, \theta) := 0$;
(06) FOR all $i \in N - s$ DO $\tau(i, \theta) := \infty$;
(07) END;
(08) $\tilde{\tau}(t) := \infty$; $L := \{(s, \theta) | \theta = 0, 1, \cdots, T\}$;
(09) WHILE($L \neq \emptyset$) DO
(10) BEGIN
(11) select the first (i, θ_i) from L; $L := L - \{(i, \theta_i)\}$;
(12) FOR all $j \in N$ with $(i, j) \in A^+(\bar{f})$ DO
(13) IF($\theta_i + h(i, j; \theta_i) \leq T$ and $\tau(i, \theta_i) + h(i, j; \theta_i) < \tau(j, \theta_i + h(i, j; \theta_i))$) THEN
(14) $\tau(j, \theta_i + h(i, j; \theta_i)) := \tau(i, \theta_i) + h(i, j; \theta_i)$; $p(j, \theta_i + h(i, j; \theta_i)) := (i, \theta_i)$
(15) IF($(j, \theta_i + h(i, j; \theta_i)) \notin L$) THEN $L := L \cup \{(j, \theta_i + h(i, j; \theta_i))\}$;
(16) FOR all $j \in N$ with $(i, j) \in A^-(\bar{f})$ DO
(17) FOR all θ_j with $\theta_i = \theta_j + h(j, i; \theta_j)$ DO
(18) IF($\tau(i, \theta_i) - h(j, i; \theta_j) < \tau(j, \theta_j)$); THEN
(19) $\tau(j, \theta_j) := \tau(i, \theta_i) - h(j, i; \theta_j)$; $p(j, \theta_j) := (i, \theta_i)$;
(20) IF($(j, \theta_j) \notin L$) THEN $L := L \cup \{(j, \theta_j)\}$;
(21) END;
(22) $\tilde{\tau}(t) = \min_{\theta \in \{0,1,\cdots,T\}}\{\tau(t, \theta)\}$; $\tilde{\theta} = \min_{\theta \in \{0,1,\cdots,T\}}\{\theta | \tau(t, \theta) = \tilde{\tau}(t)\}$;
(23) IF($\tilde{\tau}(t) = \infty$) THEN $C := 0$;
(24) END.

Algorithm 3: Dynamic Path (DP) Procedure

The procedure for finding a shortest (quickest) **Dynamic Path (DP)** (presented in Algorithm 3) uses the label setting approach starting from the source node and labelling nodes which are reachable from s, according to their transit times. The procedure uses *transit time labels* $\tau(i, \theta)$ associated to all nodes at each discrete time values. Transit time labels $\tau(i, \theta)$ of all nodes are initialised to infinity with the exception of those of the source node which are initialised to

zero, $\tau(s,\theta)$, $\forall\theta \in \{0,1,\cdots,T\}$.
At any step of the DP procedure, the transit time labels are divided into two groups: *permanent* and *temporary* (see [3]). A label $\tau(s,\theta)$ is permanent once it denotes the length (transit time) of a shortest (quickest) augmenting path from s to (i,θ), otherwise it is temporary, indicating that it has not yet determined any augmenting path from s to (i,θ). Dynamic Path (DP) procedure maintains a set L of candidate nodes with temporary labels, initially including only source node-time pairs, which holds, in increasing order of their temporary labels, all nodes that have been reached so far by procedure and which are to be visited. For every node-time pair (i,θ) selected from the list, the arcs with positive residual capacity connecting (i,θ) to (j,ϑ) are explored, where $0 < \vartheta = \theta + h(i,j;\theta) \leq T$ if the arc connecting (i,θ) to (j,ϑ) is a forward arc and $0 \leq \vartheta = \theta - h(j,i;\vartheta) \leq T$ if it is a reverse arc. Then the minimum transit time labels are updated and the node-time pair (j,ϑ) is added to the candidate set if it is not already in it. At any of its steps, Dynamic Path procedure selects the node-time pair (i,θ) with the minimum temporary label, makes its transit time label permanent, checks optimality conditions and updates the labels accordingly. The process is repeated until there are no more candidate nodes in L. The transit time of the shortest (quickest) path, which is computed based on predecessor vector p, is given by $\tilde{\tau}(t) = \min_{\theta\in\{0,1,\cdots,T\}}\{\tau(t,\theta)\}$.

Theorem 5. (Theorem of complexity) *Procedure (DP) for finding a shortest dynamic path runs in $O(n^2T^2)$ time.*

Proof. DP procedure makes $O(nT)$ steps, investigating at each of them at most nT adjacent node-time pairs and removing one node-time pair from the list L. Thus, the complexity of finding a shortest dynamic path is $O(n^2T^2)$. □

References

[1] Ahuja, R., Magnanti, T. and Orlin, J., *Network Flows. Theory, algorithms and applications*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1993.

[2] Aronson, J.A., *A survey of dynamic network flows*, Annals of Operations Research **20** (1989), No. 5, 1-66.

[3] Cai, X., Sha, D. and Wong, C., *Time-varying Network Optimization*, Springer, 2007.

[4] Miller-Hooks, E. and Patterson, S.S., *On solving quickest time problems in time-dependent, dynamic networks*, Journal of Mathematical Modelling and Algorithms **3** (2004), 39-71.

[5] Parpalea, M. and Ciurea, E., *The quickest maximum dynamic flow of minimum cost*, International Journal of Applied Mathematics and Informatics **3** (2011), No.5, 266-274.

[6] Parpalea, M. and Ciurea, E., *Partitioning algorithm for the parametric maximum flow*, Applied Mathematics **4** (2013), Special Issue-Computer Mathematics, No.10A, 3-10.