Bulletin of the *Transilvania* University of Braşov • Vol 4(53), No. 1 - 2011
Series III: Mathematics, Informatics, Physics, 117–128

NETWORK SIMPLEX ALGORITHM FOR THE BI-CRITERIA MINIMUM COST FLOW OVER TIME PROBLEM

Mircea PARPALEA[1]

Abstract

The article formulates and studies the generalisation of the bi-criteria minimum cost flow problem for the case of dynamic flows. The approach is based on reducing the dynamic problem to a static bi-criteria problem on a time-expanded network with constant capacities, fixed transit times on arcs and for a flow leaving the source node only at time $\theta = 0$ and solving the problem via network-simplex based algorithm.

2000 *Mathematics Subject Classification:* 90B10, 90C35, 90C47, 05C35, 68R10.
Key words: Flows over time, dynamic network, minimum cost flows, network simplex algorithm, bi-criteria minimum cost flows.

1 Indroduction

Dynamic flows are widely used to model different network-structured, decision-making problems over time, but because of their complexity, dynamic flow models have not been investigated as much as classical flow models. The time is an essential component, either because the flows take time to pass from one location to another, or because the structure of the network changes over time. Dynamic networks were introduced by Ford and Fulkerson [6]. They introduced flows which take time, called travel time, to pass an arc of the network, called dynamic flows or flows over time. For the maximum flow problem in discrete time they developed a technique, based on reducing the dynamic problem to the classical static problem on a time-expanded network, which is still widely used.
On the other hand, in many combinatorial optimization problems, the selection of the optimum solution takes into account more than one criterion. Often, these criteria are in conflict and for this reason, a multi-objective network flow formulation of the problem is necessary. In this paper, the case of continuous flow values is considered where the flow variables take time to pass from one location to another, proposing a bi-criteria simplex-network based approach. The proposed method consists is a time-expanded network-based algorithm which finds the efficient boundary in the objective space. Further on, in Section 2 some basic dynamic network flow and bi-criteria minimum cost flow terminology are presented together with some results used in the rest of the paper. More specialized terminology is developed in the following sections. Section 3 presents the algorithm to

[1]National College, *Andrei Şaguna* of Braşov, Romania, e-mail: parpalea@gmail.com

find all efficient solutions for the bi-criteria minimum cost flow over time problem. Section 4 gives an example that helps to understand the steps performed by the former algorithm in a time-expanded network built out of a dynamic network.

2 Terminology and preliminaries

2.1 Discrete-time dynamic network flows

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

$$\sum_{j|(i,j)\in A} f(i, j; \theta) - \sum_{\substack{j|(j,i)\in A \\ \theta - h(j,i;\theta) \geq 0}} f(j, i; \theta - h(j, i; \theta)) = v(i; \theta),\ \forall i \in N; \tag{1}$$

where $f(i, j; \theta)$ determines the rate of flow (per time unit) entering arc (i, j) at time θ. Capacity constraints 2 mean that in a feasible dynamic flow, at most $u(i, j; \theta)$ units of flow can enter the arc (i, j) at the time-moment θ.

$$0 \leq f(i, j; \theta) \leq u(i, j; \theta),\ \forall \theta \in \{0, 1, \dots, T\},\ \forall (i, j) \in A; \tag{2}$$

$$f(i, j; \theta) = 0,\ \forall (i, j) \in A,\ \theta \in \overline{T - h(i, j; \theta) + 1, T}. \tag{3}$$

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; this is ensured by condition 3. The total cost of the dynamic flow $f(i, j; \theta)$ in a discrete-time dynamic network is defined as:

$$C(f) = \sum_{\theta \in \{0,1,\dots,T\}} \sum_{(i,j)\in A} f(i, j; \theta) \cdot c(i, j; \theta) \tag{4}$$

For the discrete-time stationary dynamic network the capacities, transit times and costs of all arcs are constants in time: $u(i,j;\theta) = u(i,j)$, $h(i,j;\theta) = h(i,j)$, $c(i,j;\theta) = c(i,j)$, $\forall (i,j) \in A$ and $\forall \theta \in \{0,1,\ldots,T\}$.

2.2 Time-expanded network

In the discrete time model, flows over time can be described and computed in time-expanded networks which were introduced by Ford and Fulkerson [6]. A time-expanded network contains a copy of the node set for each discrete time step in the time horizon $\theta \in \{0,1,\ldots,T\}$ and the arcs are redrawn between these copies to express their traversal times.

Definition 1. *The time-expanded version of network G is a digraph G^T defined as follows:*

$N^T := \{i_\theta | i \in N, \theta \in \{0,1,\ldots,T\}\}$;
$A^T := \{a_\theta = (i_\theta, j_{\theta+h(i,j)}) | a \in A, 0 \leq \theta \leq T - h(i,j)\}$;
$u^T(a_\theta) := u(a)$ *for* $a_\theta \in A^T$;
$c^T(a_\theta) := c(a)$ *for* $a_\theta \in A^T$.

For every arc $(i,j) \in A$ with traversal time $h(i,j)$, capacity $u(i,j)$ and cost $c(i,j)$, the graph G^T has arcs $(i_\theta, j_{\theta+h(i,j)})$ for $\theta = 0,1,\ldots,T-h(i,j)$ with capacities $u(i,j)$ and costs $c(i,j)$. For the flow $f(a;\theta)$ in the dynamic network G, the function $f^T(a_\theta)$ that represents the corresponding flow in the time-expanded network G^T is defined as: $f^T(a_\theta) = f(a;\theta)$, $\forall a_\theta \in A^T$. Since the dynamic network has $T+1$ copies of each source node and each sink node, the time-expanded network will have multiple sources and multiple sinks. Therefore in order to handle many sources and sinks, a super source s^* and a super sink t^* are introduced to create a single source/single sink network. The super source is connected to all time-copies of the source node through arcs having zero travel time. Similarly all copies of the sink node are connected to the super sink node. All connections to the super sink have zero travel time, zero cost and infinite upper bounds: $h(t_\theta, t^*) = 0$, $u(t_\theta, t^*) = \infty$ and $c(t_\theta, t^*) = 0$ for all $\theta \in T(t)$.

Theorem 1. *If f is a flow in the dynamic network G and f^T is a corresponding flow in the time-expanded network G^T, then $C(f) = C^T(f^T)$. Moreover, for each minimum cost flow f^* in the dynamic network G there is a corresponding minimum cost flow f^{*T} in the static network G^T such that $C(f^*) = C^T(f^{*T})$ and vice-versa.* (see [4])

Definition 2. *The length of a path P in G with respect to the travel times is given by $h(P) := \sum_{(i,j)\in P} h(i,j)$.*

The length of the shortest path from the source node s to node i is denoted by $h(i)$ and the length of the shortest path from node i to the sink node t is denoted by $\bar{h}(i)$. For every node $i \in N$, a set of times $T(i)$ is defined so that node i at time $\theta \in T(i)$ is reachable from the source and also can reach the sink within the time horizon T, as follows:

$$T(i) := \{\theta + h(i) | \theta + h(i) + \bar{h}(i) \leq T,\ \theta \in \{0,1,\ldots,T\}\}. \tag{5}$$

For every arc $(i,j) \in A$, a set of times $T(i,j)$ for which node i of arc (i,j) at time $\theta \in T(i,j)$ is reachable from the source and the sink is also reachable within T from node j at time $\theta + h(i,j)$ is defined as follows:

$$T(i,j) := \{\theta + h(i) | \theta + h(i) + h(i,j) + \bar{h}(j) \leq T,\ \theta \in \{0,1,\ldots,T\}\}. \tag{6}$$

By definition, $h(s) = 0$ and $T(s) = \{0,1,\ldots,T - \bar{h}(s)\}$. The set $T(i)$ determines a set of times where copying the node i guarantees that there exists a path from the source to the sink that passes through node i at time $\theta \in T(i)$ within the time horizon T. Whereas the set $T(i,j)$ determines a set of times where copying the arc (i,j) guarantees that the time-copy of this arc belongs to at least one path from the source to the sink within T.

Proposition 1. *If $n = |N|$ and $m = |A|$ then $n \cdot (T+1)$ and $(n+m) \cdot T + m - \sum_{(i,j) \in A} h(i,j)$ are the upper bound for the number of nodes and arcs in G^T without considering super source and super sink, respectively. Hence, G^T has $O(nT)$ many nodes and $O((n+m)T)$ many arcs.* (see [7])

The main complexity of building the expanded network consists in computing the lengths of the shortest paths from the source node to the nodes i, the lengths of the shortest paths from nodes i to the sink node and consequently in computing the sets $T(i)$ and $T(i,j)$. While the values of $h(i)$ for all $i \in N$ can be calculated by using the forward version of Dijkstras label setting algorithm, the values of $\bar{h}(i)$ for all $i \in N$ can be obtained by applying the backward version of this algorithm.

Theorem 2. *The expanded network is built in $O(n^2)$ time.* (see [4], [7])

A discrete time dynamic network flow problem is a discrete time expansion of a static network flow problem. In this case the flow is distributed over a set of predetermined time periods $\theta = 0,1,\ldots,T$. Unfortunately, due to the time expansion, the size of the resulting time-expanded network is not polynomial in the size of the input. (see [6]) On the other hand, the advantage of this approach is that it turns the problem of determining an optimal flow over time into a classical network flow problem on the time-expanded network.

2.3 Network Simplex Method

The Network Simplex Method (NSM) maintains a feasible spanning tree (basis) and successfully goes toward the optimality conditions until it becomes optimal. The method iterates towards an optimal solution by exchanging basic and non-basic arcs. At each of its iterations, the arcs in the graph are divided into three sets: the arcs belong to the spanning tree (B); the arcs with flow at their lower bound (L); the arcs with flow at their upper bound (U). A spanning tree structure (B,L,U) is optimal if the reduced cost for every arc $(i,j) \in L$ is higher than zero and at the same time the reduced cost for every arc $(i,j) \in U$ is less than zero [10]. Under these circumstances, the current solution is optimal. Otherwise, there are arcs in the graph that violate the optimal conditions. An arc is a violated arc if it belongs to L (U) with negative (positive) reduced cost. To create the

initial or Basic Feasible Solution (BFS), an artificial node 0 and artificial arcs are appended to the graph. The node 0 will be the root of spanning tree (B) and the artificial arcs, with sufficiently large costs and capacities, connect the nodes to the root. The set (L) consists of the main arcs in the graph, and the set (U) is empty. Appending the entering arc (k, l), which is a violated arc, to the spanning tree forms a unique cycle, W with the arcs of the basis. In order to eliminate this cycle, one of its arcs must leave the basis. The cycle is eliminated when we have augmented flow by a sufficient amount to force the flow in one or more arcs of the cycle to their upper or lower bounds. By augmenting flow in a negative cost augmenting cycle, the objective value of the solution is improved. The first task in determining the leaving arc is the identification of all arcs of the cycle. The flow change is determined by the equation $\delta = min\{f(i,j)|(i,j) \in W\}$. The leaving arc is selected based on cycle W. The substitution of entering for the leaving arc and the reconstruction of new tree is called a pivot. After pivoting to change the basis, the reduced costs for each arc $(i,j) \in B$ are calculated. If the reduced costs for all $(i,j) \in L \cup U$ satisfy the optimality condition then the current basic feasible solution is optimal. Otherwise, an arc (i,j) where there is a violation should be chosen and operations of the algorithm should be repeated. There are many strategies for selecting the entering arc, and these determine the speed of solution. In order to reduce the number of degenerate pivots, i.e. pivots in which a flow change of zero occurs, and to prevent cycling, the *strongly feasible basis technique* proposed by Cunningham [3] is used. The basis structure (B, L, U) is *strongly feasible* if a positive amount of flow can be sent from any node to root along arcs in the spanning tree without violating any of the flow bounds. The technique specifies that when there is more than one blocking arc, the last blocking arc encountered in traversing the cycle, in the direction of its orientation, starting at the joint predecessor in the basis tree, should be selected as the leaving arc. By consistently applying this technique, a strongly feasible basis is maintained throughout the operation of the algorithm, and the number of iterations required to obtain the optimal solution is guaranteed to be finite.

Theorem 3. *A strongly feasible basis is preserved by the Cunningham technique.* (see [3])

Theorem 4. *The strongly feasible basis technique guarantees that Network Simplex Algorithm will obtain the optimal solution in a finite number of pivots.* (see [3])

Theorem 5. *The complexity of a pivot operation is* $O(n+m)$ *with* $n = |N|$, $m = |A|$. (see [10])

Theorem 6. *The complexity of the Network Simplex minimum cost flow algorithm in a network having* n *nodes with* $\bar{u}$ *being the maximum upper bound and* $\bar{c}$ *being the maximum cost among all its* m *arcs is* $O((n+m) \cdot m \cdot n^2 \cdot \bar{c}^2 \cdot \bar{u})$. (see [10])

2.4 Bi-criteria minimum cost flow problem

Given a directed network $G = (N, A)$ with N being the set of nodes and A being the set of arcs, let $u(i,j)$ be the upper bound (capacity) of the arc (i,j), $v(i)$ the supply/demand of the node i and $c_k(i,j)$ the cost per unit of flow on arc (i,j) in the $k-th$ objective function, $k = 1, 2$. If $f(i,j)$ denotes the amount of flow on an arc (i,j), the bi-criteria

minimum cost flow problem is to determine a total flow of v units which simultaneously minimizes two cost functions. Any vector of flow f that satisfies the flow conservation and the capacity constraints is called a feasible solution of the bi-criteria minimum cost flow (BMCF) problem. The set of feasible solutions, or *decision space*, is denoted by F and its image through $Y(F) := \{y_1(f), y_2(f) | f \in F\}$, where $y_1(f)$ and $y_2(f)$ are the two objective functions, is called *objective space*. In general, there is no feasible solution of the (BMCF) problem that simultaneously minimizes both objectives. In other words, an optimum global solution does not exist. For this reason, the solutions of these problems are searched for among the set of efficient points.

Definition 3. *A feasible solution $f \in F$ of the bi-criteria minimum cost flow problem is called efficient if and only if there does not exist another feasible solution $f' \in F$ so that $Y(f') \leq Y(f)$ with $Y(f') \neq Y(f)$ (i.e. $y_k(f') \leq y_k(f)$, $y_k(f') \neq y_k(f)$, $k = 1, 2$).*

Definition 4. *$Y(f)$ is a non-dominated criterion vector if f is an efficient solution. Otherwise $Y(f)$ is a dominated criterion vector.*

The set of efficient solutions of F will be denoted by $E[F]$ while, by extension, $E[Y(F)]$ is called the set of non-dominated solutions of $Y(F)$. It is well known that to characterize $E[Y(F)]$ for the bi-criteria continuous minimum cost flow problem, it is only necessary to identify the extreme efficient points of $Y(F)$. The set of efficient extreme points of F will be denoted by $E_{ex}[F]$ and their corresponding points of $Y(F)$ will be denoted by $E_{ex}[Y(F)]$. For the continuous case of the (BMCF) problem all the efficient solutions lie on the efficient boundary of $Y(F)$. The algorithm proposed in this paper determines the entire efficient boundary of objective space for the bi-criteria continuous minimum cost flow over time problem.

3 Bi-criteria minimum cost flow over time problem

The bi-criteria continuous minimum cost flow over time problem is to determine a flow over time transporting v units of flow which simultaneously minimizes two cost functions, where the continuous flow values are permissible. Assuming that all the v units of flow leave the source node only at time $\theta = 0$, then the problem can be formalized as follows:

$$\textit{minimize} \quad y_k(f) = \sum_{\theta \in \{1,2,\ldots,T\}} \sum_{(i,j) \in A} c_k(i,j) \cdot f(i,j;\theta), \quad k = 1,2 \quad \textit{subject to} \tag{7}$$

$$\sum_{j|(i,j)\in A} f(i,j;\theta) - \sum_{\substack{j|(j,i)\in A \\ \theta - h(j,i) \geq 0}} f(j,i;\theta - h(j,i)) = \begin{cases} v(i;\theta), & i = s \\ 0, & \forall i \in N - \{s,t\} \\ -v(i;\theta), & i = t \end{cases} \tag{8}$$

$$0 \leq f(i,j;\theta) \leq u(i,j) \qquad \forall \theta \in \{1,2,\ldots,T\}, \forall (i,j) \in A, \tag{9}$$

with $v(s;0) = v$, $v(s;\theta) = 0$, $\forall \theta \in \{1,2,\ldots,T\}$ and $\sum_{\theta \in \{1,2,\ldots,T\}} v(t;\theta) = -v$.

3.1 Time-expanded network

Since for the studied problem $v(s;\theta) = 0$, $\forall\theta \in \{1, 2, \ldots, T\}$ results that $T(s) = \{0\}$ and from the expanded network all the nodes which are not reachable from the source s_0 and all arcs connecting these nodes are deleted. The procedure which calculates the expanded network, EN is summarized in the algorithm in Table 1.

```
1  procedure EN(G = (N, A, T));
2  begin
3    calculate h(i) and h̄(i) for every i ∈ N;
4    determine T(i) for every i ∈ N;
5    determine T(i, j) for every (i, j) ∈ A;
6    N^T := {i_θ | i ∈ N; θ ∈ T(i)};
7    delete all nodes i_θ ∈ N^T, θ ∈ T(i) not reachable from s_0 and the arcs incident to nodes i_θ;
8    A^T := {(i_θ, j_{θ+h(i,j)}) | (i, j) ∈ A; θ ∈ T(i, j)};
9    add a super sink t* and the arcs (t_θ, t*);
10 end;
```

Table 1: The algorithm for building a compact G^T from G with constant travel times.

3.2 Bi-criteria network-simplex algorithm

The method used to obtain the efficient boundary of the objective space starts with the efficient extreme point in the objective space $Y(f^0)$ which results when only the first objective is considered and finds the remaining efficient points by a finite sequence of pivots, always choosing a basis entering arc with the least ratio of improvement of y_2 and worsening of y_1. The procedure generates extreme non-dominated solutions moving in a left-to-right fashion. The strongly feasible spanning tree (B^0, L^0, U^0) for the flow (f^0 and the node potentials p_1 with respect to the first objective are obtained using the network simplex method presented in Table 2. (see [1], [2])

```
1  procedure Network Simplex(G, c);
2  begin
3    generate initial BFS, (B, L, U);
4    (k, l) ← entering arc ∈ L ∪ U;
5    while (exists an entering arc (k, l)) do
6      begin
7        find cycle W ∈ B ∪ (k, l);
8        δ ← flow change;
9        (p, q) ← leaving arc ∈ W;
10       update flow in W by δ;
11       update BFS, tree B;
12       update node potentials;
13       (k, l) ← entering arc ∈ L ∪ U;
14     end;
15 end;
```

Table 2: The procedure Network Simplex.

Let $c_1^p(i,j) = c_1(i,j) - p_1(i) + p_1(j)$ be the reduced costs for all arcs $(i,j) \in A^T$. Then

(B^0, L^0, U^0) is optimum and satisfies the following optimality conditions with respect to the first objective: Once (B^0, L^0, U^0) and p_1 are obtained, the node potentials p_2

$$\begin{array}{rll} 0 \leq f^0(i,j) \leq u(i,j) & \text{and} & c_1^p(i,j) = 0,\ \forall (i,j) \in B^0, \\ f^0(i,j) = 0 & \text{and} & c_1^p(i,j) \geq 0,\ \forall (i,j) \in L^0, \\ f^0(i,j) = u(i,j) & \text{and} & c_1^p(i,j) \leq 0,\ \forall (i,j) \in U^0. \end{array}$$

with respect to the second objective can be obtained by solving the system of equations: $c_2^p(i,j) = c_2(i,j) - p_2(i) + p_2(j) = 0,\ \forall (i,j) \in B^0$.
As f^0 is an efficient extreme point in the decision space and $Y(f^0)$ is a non-dominated extreme point in objective space, the next step consists in finding the non-dominated extreme point in the objective space that is adjacent to $Y(f^0)$. Every efficient solution corresponds to a basic spanning tree and two solutions are called adjacent if the two corresponding spanning trees differ in only two arcs. A candidate arc, i.e. $(i,j) \in L^0 \cup U^0$ is called *eligible* if $(i,j) \in U^0$ and $c_2^p(i,j) > 0$ or $(i,j) \in L^0$ and $c_2^p(i,j) < 0$. For $\lambda_1 = min\{\frac{c_2^p(i,j)}{c_1^p(i,j)} | c_2^p(i,j) < 0, \forall (i,j) \in L^0\}$, $\lambda_2 = min\{\frac{c_2^p(i,j)}{c_1^p(i,j)} | c_2^p(i,j) > 0, \forall (i,j) \in U^0\}$ and $\lambda = min\{\lambda_1, \lambda_2\}$, let S be the set containing all non-basic variables that reach the value λ, i.e. the non-basic variables associated with an efficient edge in the objective space. An efficient edge is referred to as the edge of the polyhedral $Y(F)$ that connects two adjacent non-dominated extreme points. (see [8], [12])
By entering the basis, an arc $(k,l) \in S$ generates a single unique cycle W called *pivot cycle* oriented in the same way as the arc (k,l) if $(k,l) \in L^0$ and in the opposite direction if $(k,l) \in U^0$. Let W^+ be the set of forward arcs and W^- the set of backward arcs in W. As the residual capacities of the arcs composing the cycle W are defined as:

$$r(i,j) = \begin{cases} u(i,j) - f(i,j), & if \quad (i,j) \in W^+; \\ f(i,j), & if \quad (i,j) \in W^-, \end{cases} \tag{10}$$

the residual capacity of the cycle W is $r(W) = min\{r(i,j) | (i,j) \in W\}$. Therefore, the flow will be increased along the arcs in the cycle by $r(W)$ if $(i,j) \in W^+$, respectively decreased by $r(W)$ if $(i,j) \in W^-$, when one non-basic variable of S, $(k,l) \in S$ enters the basis. The adjacent non-dominated extreme point is obtained by adding this variable to the basis and correspondingly changing the flow $f(i,j)$ with $r(W)$ units of flow. If $r(W) = 0$, the basis and the node potentials are updated without changing the flow (degenerate basis). The algorithm always maintain the last investigated efficient solution in the decision space stored in the list R. The network simplex method uses the tree vectors: π, η and ω to improve the operations in the pivot process. The *predecessor* vector π is defined as $\pi(j) = k$ where k is one before the last node in the single path from the root node to the node j in the minimum spanning tree; by convention, $\pi(root) = 0$. The *depth* vector η is defined as $\eta(j) = \ell$ where ℓ is the number or arcs in the single path from the root node to the node j in the minimum spanning tree; by convention, $\eta(root) = 0$. The *thread* vector ω is defined as $\omega(j) = q$ where q is the node after j in preordering (the order that the nodes were first visited by the depth-first search) of the minimum spanning tree; by convention, $q = (root)$ if j is the last node in the sequence. (see [1], [3])
The Bi-criteria Network Simplex (BiNS) algorithm is presented in Table 3.

```
1  procedure BiNS(G^T);
2  begin
3   Network Simplex(G^T, c_1);     // gets f^0, p_1, p_2, (B^0, L^0, U^0), π, η, ω
4   c_k^p(i,j) := c_k(i,j) − p_k(i) + p_k(j), ∀(i,j) ∈ A^T, k = 1,2;
5   add {f^0, p_1, p_2, (B^0, L^0, U^0)} to R;
6   while (R ≠ ∅) do
7    begin
8     R := R − {f, p_1, p_2, (B, L, U)} ;
9     y_k(f) := Σ_{(i,j)∈A^T} c_k(i,j) · f(i,j), k = 1,2;
10    Y := (y_1(f), y_2(f));   Q := 0;
11    S_L := {(i,j) ∈ L | c_2^p(i,j) < 0};   S_U := {(i,j) ∈ U | c_2^p(i,j) > 0};
12    λ' := {λ(i,j) = c_2^p(i,j) / c_1^p(i,j) | (i,j) ∈ S_L ∪ S_U};
13    S := {(i,j) ∈ S_L ∪ S_U | λ(i,j) = λ'};
14    if (S ≠ ∅) then Q := 1;
15    while (S ≠ ∅) do
16     begin
17      remove the first arc (i,j) from S;
18      Pivot(f, (i,j), (B, L, U), p_1, p_2, π, η, ω, R);
19     end;
20    if (Q = 1) then add {f, p_1, p_2, (B, L, U)} to R;
21   end;
22 end;
```

Table 3: The bi-criteria Network Simplex (BiNS) algorithm for the minimum two cost functions flow problem.

The Pivot procedure presented in Table 4 identifies all efficient points that are reached from the current checked extreme efficient point. For the new basis the potentials and the three vectors π, η and ω are updated and the next candidate arc is selected until all the efficient solution that lie on an efficient edge are found. The procedure ends when the set S is empty and the structure (B, L, U) identifies an efficient extreme point in the objective space which is added to the set R. Then the BiNS procedure repeats the process with the following point in R. At the end of the algorithm, the set of all points on the efficient boundary is obtained.

```
1  procedure Pivot(f, (i,j), (B, L, U), p_1, p_2, π, η, ω, R);
2  begin
3   if (f(i,j) = u(i,j)) then remove (i,j) from U;
4   else   if (f(i,j) = 0) then remove (i,j) from L;
5   add (i,j) to B;     find the pivot cycle W in B ∪ (i,j);
6   compute r(W) := min{r(i,j) | (i,j) ∈ W};
7   if (r(W) > 0) then update f(i,j) for all arcs (i,j) ∈ W;
8   remove the leaving arc (p,q) ∈ {(i,j) ∈ W | r(i,j) = 0} from B;
9   if (f(p,q) = u(p,q)) then add (p,q) to U;
10  else   if (f(p,q) = 0) then add (p,q) to L;
11  update p_1, p_2, π, η, ω;
12 end;
```

Table 4: The procedure Pivot which computes the adjacent efficient solution for any given extreme efficient solution.

Putting all together, the algorithm solving the Bi-criteria minimum cost flow over time problem (BCMCFT) is presented in Table 5.

```
1  BCMCFT(f, (i, j), (B, L, U), p1, p2, π, η, ω, R);
2  Begin
3    EN(G = (N, A, T));
4    BiNS(G^T));
5  End.
```

Table 5: The procedure PIVOT which computes the adjacent efficient solution for any given extreme efficient solution.

Theorem 7. *The complexity of the Bi-criteria minimum cost flow over time algorithm is* $O((n_T + m_T) \cdot m_T \cdot n_T^2 \cdot \bar{c}^2 \cdot \bar{u})$ *with* $|N^T| = n_T$ *being the number of nodes and* $|A^T| = m_T$ *being the number of arcs in the expanded network.*

Proof. Results directly from theorems 2 and 6. □

4 Example

For the network presented in Figure 1 (left) let the value of the flow sent from source node 1 at time $\theta = 0$ be $v(s;0) = 8$ while $v(s;\theta) = 0$, $\forall\theta \in \{1, 2, \ldots, T\}$ with $T = 4$. The pair of values above every arc denotes the two cost functions while the pair of values below the arc denotes the transit time and the upper bound of every arc in the network, in that order. Procedure EN computes the expanded network and eliminates the nodes which are not reachable from the source at time $\theta = 0$ and the arcs incident to these nodes. Then all the time-copies of the sink node 5 are connected to the super sink node 6 through arcs having zero travel time, zero costs and infinite upper bounds.

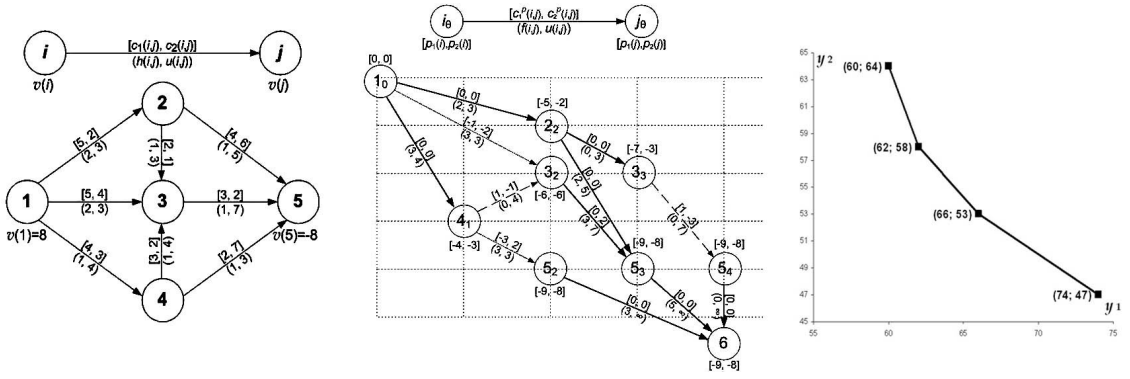

Figure 1: (letf:) An example of dynamic network G; (centre:) The optimal solution with respect to the first objective. The arcs in the minimum spanning tree B^0 are represented by solid lines, the non basic arcs in L^0 are represented by dashed lines while the non basic arcs in U^0 are represented by dotted lines; (right:) The set of all non-dominated points which lie on the efficient boundary in the objective space.

In the expanded network G^T, the Network Simplex algorithm finds the minimum cost

flow considering only the first cost function $c_1(i,j)$. The strongly feasible spanning tree (B^0, L^0, U^0) and the node potentials $p_1(i)$ with respect to the first objective and the reduced costs for all arcs $(i,j) \in A^T$ are presented in Figure 1 (centre). The values of the three vectors,π, η and ω are presented in Table 6. Since all arcs in B^0 have zero reduced costs and all arcs in $L^0 = \{(4_1, 3_2), (3_3, 5_4)\}$ and $U^0 = \{(1_0, 3_2), (4_1, 5_2)\}$ satisfy the optimality conditions with respect to the first objective, the solution is optimal and the node potentials $p_2(i)$ with respect to the second objective are computed.

i	1_0	2_2	3_2	3_3	4_1	5_2	5_3	5_4	6
π	0	1_0	5_3	2_2	1_0	6	2_2	6	5_3
η	0	1	3	2	1	4	2	4	3
ω	4_1	5_3	6	1_0	2_2	5_4	3_2	3_3	5_2

Table 6: The values of the vectors, π, η and ω for the bases B^0.

***Iteration* 1**: The first efficient extreme point in the decision space $f^0, p_1, p_2, (B^0, L^0, U^0)$ is added to the set R and the while loop runs for the first time since $R \neq 0$. The efficient extreme point in the decision space is removed from R and the first non-dominated extreme point in objective space $Y(f)$ is computed. According to $y_k(f) = \sum_{(i,j)\in A^T} c_k(i,j) \cdot f^0(i,j)$ with $k = 1, 2$ the following values are obtained: $y_1(f^0) = 60$, $y_2(f^0) = 60$ and $Y(f^0) = (60, 64)$.
Then the flag Q is set to 0 and the two sets $S_U = \{(4_1, 5_2)\}$ and $S_L = \{(4_1, 3_2), (3_3, 5_4)\}$ are built with $\lambda(4_1, 5_2) = -2/3$, $\lambda(4_1, 3_2) = -1$ and $\lambda(3_3, 5_4) = -3$. As can easily be noticed, the non-basic arc $(1_0, 3_2)$ is not selected since it belongs to U^0 but it has a negative reduced cost with respect to the second objective. The minimum λ value is $\lambda' = -3$ and the corresponding arc $(3_3, 5_4)$ is added to the set S. Since now $S = \{(3_3, 5_4)\}$ is not an empty set the flag Q is set to 1 and the first (and single) arc is removed from S and the algorithm draws on the PIVOT procedure having as its argument arc $(i,j) = (3_3, 5_4)$. As $f^0(3_3, 5_4) = 0$, the arc is removed from the set L^0 and added to the bases B^0. Based on predecessor and depth vectors the pivot cycle $W = (2_2, 3_3, 5_4, 6, 5_3, 2_2)$ is obtained which is oriented in the same way as the entering arc since $(3_3, 5_4) \in L^0$. The residual capacity of the pivot cycle, computed for all arcs in the cycle is $min\{3, 7, \infty, 5, 2\} = 2$. Consequently, the flow on arcs $(2_2, 3_3)$, $(3_3, 5_4)$ and $(5_4, 6)$ will be increased by two units while the flow on arcs $(5_3, 6)$ and $(2_2, 5_3)$ will be decreased by the same value. The arc which leaves the bases after updating the flow is the arc $(2_2, 5_3)$ for which the flow decreases to zero so that it is added to the set L and the potentials and the three vectors π, η and ω are updated. Since $S = \emptyset$ and $Q = 1$ the updated set $\{f, p_1, p_2, (B, L, U)\}$ is added to the set R and the algorithm reiterates with the new efficient extreme point in the decision space for which the second non-dominated extreme point in objective space is computed: $y_1(f) = 62$, $y_2(f) = 58$ and $Y(f) = (62, 58)$.
In two more iterations, the algorithm finds the consecutive non-dominated extreme points $y_1(f) = 66$, $y_2(f) = 53$ with $Y(f) = (66, 53)$ and $y_1(f) = 74$, $y_2(f) = 47$ with $Y(f) = (74, 47)$.
In the updated network all arcs in B have zero reduced costs and all arcs in $L =$

$\{(2_2, 5_3), (4_1, 5_2)\}$ and $U = \{(1_0, 3_2), (2_2, 3_3)\}$ satisfy the optimality conditions with respect to the second objective, the solution is optimal. Consequently, $S_L = \emptyset$, $S_U = \emptyset$ and since $S = \emptyset$ the value of flag $Q = 0$ is maintained and the updated configuration is no longer saved in R which remains an empty set and the algorithm stops.
The set of all non-dominated extreme points in objective space $E_{ex}[Y(F)]$ together with the non-dominated non-extreme points $E_{nex}[Y(F)]$ that lie on the efficient boundary connecting two consecutive non-dominated extreme points are presented in Figure 1 (right).

References

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

[2] Ciurea, E., Ciupala, L.: *Algoritmi. Introducere in algoritmica fluxurilor in retele*, Ed. MATRIX ROM. Bucureşti, 2006.

[3] Cunningham, W.H.: *A network simplex method*, Mathematical Programming **11** (1976), 105-106.

[4] Fonoberova, M.: *Optimal flows in dynamic networks and algorithms for their finding*, Institute of Mathematics and Computer Science, Academy of Sciences of Moldova, 2007.

[5] Fleischer, L., Skutella, M.: *Minimum cost flows over time without intermediate storage*, Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (2003), 6675.

[6] Ford, L. R., Fulkerson, D. R.: *Constructing maximal dynamic flows from static flows*, Operations Research **6** (1958), 419433.

[7] Hamacher, H.W., Tjandra, S.A.: *Mathematical modelling of evacuation problems: a state of art*, Berichte des Fraunhofer ITWM **24** (2001), 1-38.

[8] Lee, H., Pulat, S.: *Bicriteria network flow problems: continuous case*, European Journal of Operational Research **51** (1991), 119-126.

[9] Pulat, P., Huarng, F., Lee, H.: *Efficient solutions for the bicriteria network flow problem*, Computers and Operations Research (1992) **19(7)**, 649-655.

[10] Rashidi, H., Tsang, E.: *An efficient extension of network simplex algorithm*, Journal of Industrial Engineering, **2** (2009), 1-9.

[11] Ruhe, G.: *Algorithmic aspects of flows in networks*, Klüwer Academic Publishers, Dordrecht, The Netherlands, 1991.

[12] Sedeo-Noda, A., Gonzlez-Martn, C.: *An algorithm for the biobjective integer minimum cost flow problem*, Computers and Operations Research **28** (2001), 139-156.