The quickest maximum dynamic flow of minimum cost

Mircea Parpalea and Eleonor Ciurea

***Abstract*—** This article states and solves the multi-criteria maximum flow problem in discrete dynamic networks for the case of two objective functions. This represents a generalisation of the maximum flow of minimum cost problem for the case of minimizing the travelling cost (minimum cost flow) and travelling time (quickest flow). The approach is actually based on generating efficient extreme points in the objective space by iteratively solving a series of maximum flow problems with different single objective functions. Each time, the dynamic flow is augmented along a cheapest (minimum cost) path from the source node to the sink node in the time-space network avoiding the explicit time expansion of the network.

***Keywords*—** Bi-criteria flow, Discrete dynamic network, Dynamic maximum flow, Successive shortest path.

I. Introduction

NETWORK flow problems form a large area of optimization and are central problems in operations research, computer science, applied mathematics, and many fields of engineering. Static network flow problems have been in the focus of interest for many years and they represent a very successful area of combinatorial optimization. Classical (static) network flow models have been well known as valuable tools for many applications [1] and therefore efficient algorithms have been developed. However, they fail to capture the dynamic property of many real-life problems, such as traffic planning, production and distribution systems, communication systems, and evacuation planning. 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 well as classical flow models. The time is an essential component, either because the flows take time to pass from one location to another, to rest at certain nodes [6] or because the structure of the network changes over time. Not surprisingly, dynamic flow problems are significantly more difficult to tackle, yet much closer to reality than the static ones. Dynamic networks were introduced by Ford and Fulkerson [12]. 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. For example, in transportation problems or in network flows problems, the criteria that can be considered are the minimization of the cost for selected routes, the minimization of arrival time at the destinations, the minimization of the deterioration of goods, the minimization of the load capacity that would not be used in the selected vehicles, the maximization of safety, reliability, etc. Often, these criteria are in conflict and for this reason, a multi-objective network flow formulation of the problem is necessary, as for example in [22]. In this paper, the case of discrete dynamic maximum flow of bi-criteria cost problem is considered where the two criteria consists in minimizing the travelling time and the travelling cost for a maximum possible flow which can be sent from a source to a sink within a time horizon *T*. The proposed method consists in iteratively generating efficient extreme points in the objective space by solving a series of single objective maximum flow problems with different objective functions. On each of the iterations, the flow is augmented along a cheapest path from the source node to the sink node in the time-space network, avoiding the explicit time expansion of the network. Another similar but yet different approach is that of solving a maximum flow problem with a parametric objective function. The case of the parametric flows is more detailed described in [8], [9] and [23].

Further on, in Section II some basic dynamic network flow terminology is presented together with some results used in the rest of the paper. More specialized terminology is developed in later sections. Section III deals with the successive shortest path algorithm for solving the maximum flow of minimum cost problem in discrete dynamic networks while Section IV presents the formulation and a possible solution for the discrete dynamic maximum flow of bi-criteria minimum cost problem. In Section V is given an example that helps understanding the steps performed by the former algorithm in a discrete dynamic network. In the presentation to follow, some familiarity with flow algorithms is assumed and many details are omitted, since they are straightforward modifications of known results. The notions and results presented in Section II and Section III are taken from [1], [3], [5], [7], [19] and [27].

Manuscript received June 26, 2011.

M. Parpalea, National College *Andrei Saguna* 1, Saguna street, 500123, Brasov (e-mail: parpalea@gmail.com).

E. Ciurea, Department of Computer Science, *Transilvania* University of Brasov, 25, Eroilor, blvd, 500030, Brasov (e-mail: e.ciurea@ unitbv.ro).

II. Terminology and Notations

Many dynamic network flow problems are considered as extensions of static network flow problems. These include maximum dynamic flow and minimum cost dynamic flow problems. The maximum dynamic flow problem seeks a dynamic flow which sends as many as possible a commodity from a single source to a single sink of the network within the time horizon T. The minimum cost dynamic flow problem seeks a dynamic flow that minimizes the total shipment cost of a commodity in order to satisfy demands at certain nodes within T.

A. 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. 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 $i\in N$ at the time-moment $\theta\in\{0,1,\dots,T\}$, if $v(i;\theta)<0$ or the supply of node i at the time-moment $\theta\in\{0,1,\dots,T\}$, if $v(i;\theta)>0$. 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 $G=(N,A,u,h,c,T)$ with time horizon T is a function $f: A\times\{0,1,\dots,T\}\to\Re^+$ that satisfies the following flow conservation constraints $\forall\theta\in\{0,1,\dots,T\}$:

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

where $f(i,j;\theta)$ determines the rate of flow (per time unit) entering arc (i,j) at time θ.

Capacity constraints (1.b) 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.

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

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

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 (1.c). The total cost of the dynamic flow $f(i,j;\theta)$ in a dynamic network is defined as:

$$C(f)=\sum_{\theta\in\{0,1,\dots,T\}}\sum_{(i,j)\in A} f(i,j;\theta)\cdot c(i,j;\theta). \quad (2)$$

B. Time-space network

In the discrete time model, a useful tool for studying the minimum cost flow over time problem is the *time-space network*. 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 time step in the time horizon T, $\theta\in\{0,1,\dots,T\}$.

A *node-time pair* (NTP) (i,θ) refers to a particular node $i\in N$ at a particular time step $\theta\in\{0,1,\dots,T\}$, i.e., $(i,\theta)\in N\times\{0,1,\dots,T\}$.

The NTP (i,θ_1) is *linked* to the NTP (j,θ_2) if either

(i) $(i,j)\in A$ and $\theta_2=\theta_1+h(i,j;\theta_1)$, or

(ii) $(j,i)\in A$ 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:*

$$N^T:=\{(i,\theta)\,|\,i\in N,\theta\in\{0,1,\dots,T\}\}; \quad (3.a)$$

$$A^T:=\{a_\theta=((i,\theta),(j,\theta+h(i,j)))\,|\,(i,j)\in A, 0\leq\theta\leq T-h(i,j)\}; \quad (3.b)$$

$$u^T(a_\theta):=u(a)\quad for\quad a_\theta\in A^T; \quad (3.c)$$

$$c^T(a_\theta):=c(a)\quad for\quad a_\theta\in A^T. \quad (3.d)$$

For every arc $(i,j)\in A$ with traversal time $h(i,j)$, capacity $u(i,j)$ and cost $c(i,j)$, the time-space network G^T contains 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-space network G^T is defined as:

$$f^T(a_\theta)=f(a;\theta),\quad \forall a_\theta\in A^T. \tag{4}$$

A *dynamic path* is defined as a sequence of distinct, consecutively linked NTPs:

$$P(i_1,i_2):(i_1,\theta_1)=(i_{k_1},\theta_{k_1}),(i_{k_2},\theta_{k_2}),\ldots,(i_{k_q},\theta_{k_q})=(i_2,\theta_2). \tag{5}$$

C. *Time-dependent residual network*

The *time-dependent residual network* corresponding to a feasible flow f can be viewed as the static residual network of the time-space network corresponding to the dynamic network.

For $f(i,j;\theta)$ being the flow entering arc (i,j) at time θ, an additional flow $u(i,j;\theta)-f(i,j;\theta)$ departing from node i at time θ to node j along the arc (i,j) can be sent. Also, $f(i,j;\theta)$ units of flow can be sent from node j departing at time $\theta+h(i,j;\theta)$ and consequently arriving at node i at time θ over the arc (i,j), which amounts to cancelling the existing flow on the arc. Here, an arc with negative travel time (i.e. departing at $\theta+h(i,j;\theta)$ and arriving at θ) is considered. Whereas sending a unit of flow from i at time θ to j along (i,j) increases the flow cost by $c(i,j;\theta)$ units, sending a unit of flow in reverse direction from j departing at time $\theta+h(i,j;\theta)$ to i on the same arc decreases the flow cost by $c(i,j;\theta)$ units. Considering the above mentioned ideas, the residual network with respect to a current dynamic flow f is defined as follows.

***Definition* 2**: *The residual dynamic network with respect to a given feasible dynamic flow* f *is defined as* $G(f):=(N,A(f),T)$ *with* $A(f):=A^+(f)\cup A^-(f)$, *where*

$$A^+(f):=\{(i,j)\,|\,(i,j)\in A,\ \ \exists\,\theta\le T-h(i,j;\theta)\}$$
$$\textit{with } u(i,j;\theta)-f(i,j;\theta)>0 \tag{6.a}$$

and

$$A^-(f):=\{(i,j)\,|\,(j,i)\in A,\ \ \exists\,\theta\le T-h(j,i;\theta)\}$$
$$\textit{with } f(j,i;\theta)>0. \tag{6.b}$$

While the direct arcs $(i,j)\in A^+(f)$ have the same transit times $h(i,j;\theta)$ and costs $c(i,j;\theta)$ as in the original dynamic network G, the artificial reverse arcs $(i,j)\in A^-(f)$ in the residual dynamic network $G(f)$ are provided with the following attributes:

$$h(i,j;\theta+h(j,i;\theta)):=-h(j,i;\theta), \tag{7}$$

$$c(i,j;\theta+h(j,i;\theta)):=-c(j,i;\theta), \tag{8}$$

for $(j,i)\in A,\ 0\le\theta+h(j,i;\theta)\le T,\ f(j,i;\theta)>0$

The residual capacities of the arcs (i,j) in the residual dynamic network $G(f)$ are defined as follows:

$$r(i,j;\theta):=u(i,j;\theta)-f(i,j;\theta),$$
$$(i,j)\in A,\ 0\le\theta+h(i,j;\theta)\le T \tag{9.a}$$

$$r(i,j;\theta+h(j,i;\theta)):=f(j,i;\theta),$$
$$(j,i)\in A,\ 0\le\theta+h(j,i;\theta)\le T \tag{9.b}$$

***Definition* 3**: *A dynamic path* $P(s=i_1,i_2,\ldots,i_q=i)$ *from node* s *to node* i *is said to be a* dynamic augmenting path *if* $r(i_k,i_{k+1};\theta_k)>0$ *for* $(i_k,i_{k+1})\in A(f)$ *and* $k=1,\ldots,q-1$.

***Definition* 4**: *Given a dynamic flow* f, *the* residual capacity *of a dynamic augmenting path* $P(s=i_1,i_2,\ldots,i_q=i)$ *is defined by*:

$$r(P):=\min_{1\le k\le q-1} r(i_k,i_{k+1};\theta_k),\ \textit{for}\ (i_k,i_{k+1})\in A(f)\ \textit{and}\ k=1,\ldots,q-1. \tag{10}$$

***Definition* 5**: *The* cost *of a dynamic augmenting path* $P(s=i_1,i_2,\ldots,i_q=i)$ *is defined by*:

$$C(P):=\sum_{(i_k,i_{k+1})\in A(f)} c(i_k,i_{k+1};\theta_k)\quad\text{for } k=1,\ldots,q-1.$$

A dynamic augmenting path $P(s=i_1,i_2,\ldots,i_q=i)$ is referred to as a *dynamic shortest augmenting path* (DSAP) from node $s=i_1$ to node $i_q=i$ if $C(P)\le C(P')$ for all dynamic augmenting paths P' from node s to node i.

A dynamic path $P(i_1,i_q)$: $(i_1,\theta_1),(i_2,\theta_2),\ldots,(i_q,\theta_q)$ is called a *dynamic cycle* if $i_q=i_1$ and $\theta_q=\theta_1$. A *negative cycle* is defined as a dynamic cycle whose total cost is negative and whose capacity is greater than zero.

D. *The Maximum dynamic flow–minimum dynamic cut theorem*

A dynamic cut is considered as a dynamic extension of the static cut.

***Definition* 6**: *A node* j *is said to be* reachable *from another node* i, *if there exists a dynamic augmenting path from* i *to* j.

Considering two set-valued functions $S(\theta)$ and $\overline{S}(\theta)=N-S(\theta)$ for all $\theta\in\{0,1,\ldots,T\}$, the collection of all

$S(\theta)$ is referred to as a generalized cut of separating node s and t and is defined as follows:

***Definition* 7**: *The generalized* $s-t$ *dynamic cut* S *is a set of valued function of time defined as*:

$$S := \{S(\theta) \mid S(\theta) \subset N, s \in S(\theta), t \notin S(\theta), \theta = 0,1,\ldots,T\}.$$

***Definition* 8**: *The* capacity *of the generalized* $s-t$ *dynamic cut* S *is defined as*:

$$Cap(S) := \sum_{\theta=0}^{T} \sum_{\substack{i \in K(\theta), \\ j \in \bar{K}(\theta + h(i,j;\theta))}} u(i,j;\theta).$$

The *minimum* $s-t$ *dynamic cut* is the $s-t$ dynamic cut having the minimum value of the capacity among all $s-t$ dynamic cuts.

***Theorem* 1**: *Let* v *be the value of any feasible dynamic flow* f *in* $G=(N,A,T)$ *and* $Cap(S)$ *be the value of any generalized cut* S. *Then,* $v \le Cap(S)$. (*see.* [3])

***Theorem* 2** (*Maximum dynamic flow - minimum dynamic cut*): *The value of the maximum dynamic flow from a source node* s *to the sink node* t *equals the value of minimum* $s-t$ *dynamic cut.* (*see.* [27]).

III. Successive Shortest Path Algorithm for Dynamic Maximum Flow of Minimum Cost

The dynamic minimum cost flow problem is to determine how a given amount of flow that minimizes the total shipment cost should be sent from a source node to a sink node within the time horizon *T*, subject to the capacity limits on the arcs of the network.

The successive shortest path approach adapted to the dynamic residual network is based on solving a series of successive shortest path problems, where each is solved in a residual time-space network. An amount of flow equal to the capacity of each minimum cost path obtained is augmented, until the entire flow has been sent from the source to the sink. The main difference among the algorithms consists in solving the shortest path problem in the dynamic residual network.

A. Dynamic shortest paths

Solution approaches for classical shortest path problems are divided into two classes: label-setting and label-correcting [25]. Label setting algorithms can be applied only on acyclic networks whereas label-correcting algorithms are more general and applicable for all classes of problems. The residual time-space network is composed of two sub-networks: a forward network consisting of the set of forward arcs, denoted by $A^{+}(f)$, having positive travel times and travel costs; and a reverse network consisting of the set of reverse arcs, denoted by $A^{-}(f)$ and having negative travel times and travel costs. Each of the two sub-networks, alone, is acyclic.

```
procedure DSP(p,B);
begin
  for all $\theta \in \{0,1,\ldots,T\}$ do
    begin
      $\pi(s,\theta) := 0$;
      for all $i \in N-\{s\}$ do $\pi(i,\theta) := \infty$;
    end;
  $\bar{\pi}(t) := \infty$; $L := \{(s,\theta) \mid \theta = 0,1,\ldots,T\}$;
  while ($L \neq \phi$) do
    begin
      select$(i,\theta_i)$ from $L$; $L = L - \{(i,\theta_i)\}$;
      for all $j \in A^{+}(i)$ with $r(i,j;\theta_i) > 0$ do
        begin
          $\theta_j := \theta_i + h(i,j;\theta_i)$;
          if (($\theta_j \le T$) and ($\pi(i,\theta_i) + c(i,j;\theta_i) < \pi(j,\theta_j)$)) then
            begin
              $\pi(j,\theta_j) := \pi(i,\theta_i) + c(i,j;\theta_i)$;
              $p(j,\theta_j) := (i,\theta_i)$;
              if ($(j,\theta_j) \notin L$) then $L := L \cup \{(j,\theta_j)\}$;
            end;
        end;
      for all $j \in A^{-}(i)$ do
      for all $\theta_j$ with ($\theta_i = \theta_j + h(j,i;\theta_j)$ and $r(j,i;\theta_j) < u(j,i;\theta_j)$) do
        begin
          if ($\pi(i,\theta_i) - c(j,i;\theta_j) < \pi(j,\theta_j)$) then
            begin
              $\pi(j,\theta_j) := \pi(i,\theta_i) - c(j,i;\theta_j)$;
              $p(j,\theta_j) := (i,\theta_i)$;
              if ($(j,\theta_j) \notin L$) then $L := L \cup \{(j,\theta_j)\}$;
            end;
        end;
    end;
  $\bar{\pi}(t) = \min_{\theta \in \{0,1,\ldots,T\}} \{\pi(t,\theta)\}$;
  if ($\bar{\pi}(t) = \infty$) then $B := 0$;
end;
```

Table 1. The Dynamic Shortest Path (DSP) procedure

There are two approaches in exploring the residual time-space network to compute minimum cost paths. The first approach is to explore the two sub-networks successively, making use of the acyclicity property. In this approach, the forward sub-network is explored first, and minimum cost labels at nodes are computed. Next the reverse network is explored to update the minimum cost labels, using as initial values the labels computed from the forward sub-network. The forward and reverse sub-networks are explored successively until the minimum cost labels at all nodes, as obtained from both sub-networks, are equal.

The second approach to compute minimum cost paths is to explore the forward and reverse arcs simultaneously. For every augmentation, a set L of candidate nodes is maintained, which initially includes only the source node. The set L holds all node-time pairs which have been reached so far by the algorithm and which are to be visited. The minimum cost labels $\pi(i,\theta)$ of all node-time pairs are initialised to infinity with the exception of the minimum cost labels of the source node which are initialised to zero, $\pi(s,\theta):=0,\ \forall\theta\in\{0,1,..,T\}$.

For every node-time pair (i,θ) selected from L, 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 cost labels are updated and the node-time pair (j,ϑ) is added to the candidate set if it is not already in L. The process is repeated until there are no more candidate nodes in L. The travel cost of the minimum cost path computed based on predecessor vector p is given by $\bar{\pi}(t)=\min_{\theta\in\{0,1,\dots,T\}}\{\pi(t,\theta)\}$.

The Dynamic Shortest Path (DSP) procedure is presented in Table 1.

Cai, Sha, and Wong [3] proved that the complexity of finding a shortest dynamic flow-augmenting path, by exploring the two sub-networks successively, is $O(mnT^2)$. For algorithms which explores the forward and reverse arcs simultaneously, Miller-Hooks and Patterson [18] reported a complexity of $O(n^2T^2)$.

By using special node addition and selection procedures, Nasrabadi and Hashemi [19] succeeded to reduce significantly the number of node time pair that needs to be visited. The worst-case complexity of their algorithm is $O(nT(n+T))$, and the Fibonacci heap implementation runs in $O(Tm+Tn\log(Tn))$.

B. Successive shortest path algorithm

The successive shortest path algorithm for finding a maximum flow of minimum cost will repeatedly perform the following operations:

(i) Compute a minimum cost path P from the source node to the sink node ;

(ii) Find the residual capacity $r(P)$ of the minimum cost path;

(iii) Augment the flow along the minimum cost path and update the residual network.

The algorithm will terminate when none of the sink node-time pairs (t,θ), $\forall\,\theta\in\{0,1,\dots,T\}$ is reachable from any of the source node-time pairs (s,θ), $\forall\,\theta\in\{0,1,\dots,T\}$ which represent that there is no feasible dynamic flow augmenting path from s to t. The Successive Shortest Path (SSP) procedure is presented in Table 2.

```
procedure SSP($c$);
begin
  for all $\theta\in\{0,1,\dots,T\}$ do
    begin
      $p(s,\theta):=0$;
      for all $(i,j)\in A$ do $f(i,j;\theta):=0$;
      for all $i\in N-\{s\}$ do $p(i,\theta):=-1$;
    end;
  $B:=1$;
  DSP($p,B$);
  while ($B=1$) do
    begin
      build path $P$ based on predecessor vector $p$;
      $r(P):=\min_{(i,j)\in P}\{r(i,j;\theta_i)\}$;
      for all arcs $(i,j)\in P$ do
        if ($\theta_j>\theta_i$) then $r(i,j;\theta_i):=r(i,j;\theta_i)-r(P)$
                      else $r(j,i;\theta_j):=r(j,i;\theta_j)+r(P)$;
      for all $\theta\in\{0,1,\dots,T\}$ do
        begin
          $p(s,\theta):=0$;
          for all $i\in N-\{s\}$ do $p(i,\theta):=-1$;
        end;
      DSP($p,B$);
    end;
  return $f$;
end;
```

Table 2. The Successive Shortest Path (SSP) procedure

***Theorem* 3**: *Procedure Successive Shortest Path (SSP) computes correctly the maximum dynamic flow of minimum cost for a given time horizon T.*

Proof: The procedure terminates when the sink node is not reachable from the source node, i.e. there does not exist a dynamic augmenting path from the source node to the sink node in the time-depending residual network, meaning that a maximum flow is obtained. Since in every step the augmentation is performed over the current minimum cost path, the obtained flow is also a minimum cost flow. ■

Denoting by $\bar{u}$ the maximum value for the upper bounds of all arcs, the following theorem can be formulated:

***Theorem* 4**: *Procedure Successive Shortest Path (SSP) can be implemented in $O(\bar{u}mnT^2)$ time.*

Proof: For the labelling operation, all arcs at all times may be examined, so the running time is $O(mT)$. Updating the residual networks also requires a running time of $O(mT)$, hence the complexity of one iteration is bounded by $O(mT)$. Since at each time $0 \le \theta \le T$ there may be no more than n paths sending flow to the sink node t and the maximum flow on any possible path is at most $\overline{u}$, the maximum flow value is bounded by $O(nT\overline{u})$. Considering that each iteration at least augments one unit of flow, i.e. the algorithm terminates in $nT\overline{u}$ iterations, the total running time is bounded by $O(\overline{u}mnT^2)$. ■

IV. Bi-criteria Minimum Cost Maximum Dynamic Flow

A. Problem formulation

Let us consider a dynamic network flow problem which searches for the optimum solution by taking into account more than one criterion. By setting the two objective functions to minimizing the travelling time and the travelling cost, the bi-criteria problem of finding the maximum dynamic flow of minimum travel time and travel cost can be formulated as follows:

$$maximize \quad v = \sum_{\theta=0}^{T} \sum_{(i,t)\in A} \sum_{\vartheta \mid \vartheta + h(i,t;\vartheta)=\theta} f(i,t;\vartheta) \qquad (11.a)$$

$$minimize \quad y_c(f) = \sum_{\theta=0}^{T} \sum_{(i,j)\in A} c(i,j;\theta)\cdot f(i,j;\theta) \qquad (11.b)$$

$$minimize \quad y_h(f) = \sum_{\theta=0}^{T} \sum_{(i,j)\in A} h(i,j;\theta)\cdot f(i,j;\theta) \qquad (11.c)$$

subject to:

$$\sum_{j\mid(i,j)\in A} f(i,j;\theta) - \sum_{j\mid(j,i)\in A} \sum_{\vartheta \mid \vartheta + h(j,i;\vartheta)=\theta} f(j,i;\vartheta) = 0 \ \forall i \in N-\{s,t\} \qquad (11.d)$$

$$0 \le f(i,j;\theta) \le u(i,j;\theta), \forall \theta \in \{0,1,\ldots,T\}, \forall (i,j)\in A. \qquad (11.e)$$

Here, the value of the maximum dynamic flow for a time horizon T is denoted by v where f is the vector of flow on arcs. Any vector f that satisfies the flow conservation constraint (11.d) at the different node-time pairs and the bound constraint (11.e) is called a *feasible solution* of the dynamic maximum flow of bi-criteria minimum cost (DMFBiMC) problem.

The set of feasible solutions or *decision space* is denoted by F and its image through $Y(F) = \{(y_c(f), y_h(f)) \mid f \in F\}$ is called *objective space*.

In general, there is no feasible solution of the DMFBiMC 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* 9**: *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* $Y(f') \le Y(f)$ *with* $Y(f') \ne Y(f)$ (*i.e.* $y_k(f') \le y_k(f)$, *with at least one strict inequality* , $k \in \{1,2\}$).

***Definition* 10**: $Y(f)$ *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 the $Y(F)$. The set of efficient extreme points of F will be denoted by and by $F_{ex}[F]$ and the corresponding points of $Y(F)$ will be denoted by $Y_{ex}[Y(F)]$. The set of non-extreme efficient points on the efficient boundary of F will be denoted by $F_{nex}[F]$ and the corresponding set in the objective space by $Y_{nex}[Y(F)]$. In the DMFBiMC problem all the efficient solutions lie on the efficient boundary of $Y(F)$.

B. The algorithm

Aneja and Nair [2] developed a simple algorithm for bi-criteria transportation problems. Their procedure generates efficient extreme points on the objective space $Y(F)$ rather than on the decision space F. A series of single objective problems are solved with different objective functions and each problems leads to either a new efficient extreme point or changes the direction of search in the objective space. The algorithm terminates when no extreme point or no improving direction is available.

Let Y_1 and Y_2 be two efficient extreme points in the objective space $Y(F)$ which correspond to the efficient solutions f_1 and f_2 obtained by solving the two single objective problems $min\, y_c(f)$ and $min\, y_h(f)$ respectively. Setting $\alpha := y_c(f_2) - y_c(f_1)$ and $\beta := y_h(f_1) - y_h(f_2)$, the slope between the two efficient extreme points in the objective space Y_1 and Y_2 is $\lambda = -\beta/\alpha$. New artificial costs $c'(i,j;\theta) := \beta \cdot c(i,j;\theta) + \alpha \cdot h(i,j;\theta))$ are computed for all the arcs in the network and another single objective problem is solved with the objective function:

$$y(f) = \sum_{\theta=0}^{T} \sum_{(i,j)\in A} c'(i,j;\theta)\cdot f(i,j;\theta).$$

The algorithm maintains two sets Q and R, containing pairs of indices of the extreme points in the objective space which may or may not have additional extreme points between them. Without loosing the generality, it is assumed that $y_c(f_\ell) < y_c(f_r)$ for any pairs of indices $(\ell, r) \in Q$. Then the algorithm makes a call to the successive shortest path algorithm SSP (c') with the cost function defined as $c'(i,j;\theta)$. The efficient extreme points in the objective space Y' is computed for the obtained solution f' and, whether Y' equals Y_ℓ or Y' equals Y_r, no other efficient extreme point is needed to be searched for between Y_ℓ and Y_r. The algorithm terminates when the set Q becomes empty, meaning that there is no need to search for other efficient extreme points between any of the adjacent efficient extreme points which are already found. The bi-criteria minimum cost maximum dynamic flow is presented in Table 3.

```
Algorithm BiMCMDF(G);
begin
  $f_1$:=SSP($c$);  $Y_1$:={ ($y_c(f_1)$, $y_h(f_1)$) };
  $f_2$:=SSP($h$);  $Y_2$:={ ($y_c(f_2)$, $y_h(f_2)$) };
  $Y_{ex}$:={$Y_1$,$Y_2$};  $F_{ex}$:={$f_1$,$f_2$} $k$:=2;
  $R$:=$\phi$;  $Q$:=$\phi$;
  if ($Y_1 \neq Y_2$) then $Q$:={(1,2)};
  while ($Q \neq \phi$) do
    begin
      select ($\ell$,$r$)$\in Q$;
      $\alpha$:=$y_c(f_r)-y_c(f_\ell)$);  $\beta$:=($y_h(f_\ell)-y_h(f_r)$);
      $c'(i,j;\theta)=\beta \cdot c(i,j;\theta)+\alpha \cdot h(i,j;\theta)$
      $f'$:=SSP($c'$);
      $Y'$:={ ($y_c(f')$, $y_h(f')$) };
      if (($Y_\ell$=$Y'$)or($Y_r$=$Y'$))
        then
          begin
            $Q$:=$Q$-{($\ell$,$r$)};
            $R$:=$R$+{($\ell$,$r$)};
          end
        else
          begin
            $k$:=$k$+1;
            $f_k$:=$f'$;    $F_{ex}$:=$F_{ex}$+{$f_k$};
            $Y_k$:=$Y'$;  $Y_{ex}$:=$Y_{ex}$+{$Y_k$};
            $Q$:=$Q$+{($\ell$,$k$),($k$,$r$)}-{($\ell$,$r$)};
          end;
    end;
end;
```

Table 3. Bi-criteria Minimum Cost Maximum Dynamic Flow (BiMCMDF) algorithm

Denoting by K the number of extreme non-dominated points in the objective space, the following theorem can be formulated:

***Theorem* 5**: *The Bi-criteria Minimum Cost Maximum Dynamic Flow* (BiMCMDF) *algorithm computes the set of extreme non-dominated points in the objective space in* $O(K \cdot \bar{u} m n T^2)$ *time.*

Proof: The proof results directly from Theorem 4. ■

V. Example

In the discrete-time dynamic network presented in Fig. 1, node 1 is the source node s and node 5 is the sink node t; the value indicated on every arc denotes the upper bound (capacity) of the arc. The time horizon is set to $T = 4$ and the transit rimes and costs of the arcs are the followings:

$$h(1,2) = \begin{cases} 2, & 0 \le \theta < 1 \\ 3, & \theta \ge 1 \end{cases}, \quad h(1,3) = \begin{cases} 1, & 0 \le \theta < 2 \\ 2, & \theta \ge 2 \end{cases},$$

$$h(2,4) = \begin{cases} 3, & 0 \le \theta < 2 \\ 1, & \theta \ge 2 \end{cases}, \quad h(3,4) = \begin{cases} 2, & 0 \le \theta < 2 \\ 1, & \theta \ge 2 \end{cases},$$

$$h(2,5) = 1,\ 0 \le \theta, \quad h(3,5) = 1,\ 0 \le \theta, \quad h(4,5) = 1,\ 0 \le \theta;$$

$$c(1,2) = 2,\ 0 \le \theta, \quad c(1,3) = 2,\ 0 \le \theta, \quad c(2,4) = 7,\ 0 \le \theta,$$

$$c(3,4) = \begin{cases} 4, & 0 \le \theta < 2 \\ 5, & \theta \ge 2 \end{cases}, \quad c(3,5) = \begin{cases} 7, & 0 \le \theta < 2 \\ 12, & \theta \ge 2 \end{cases},$$

$$c(2,5) = 9,\ 0 \le \theta, \quad c(4,5) = 1,\ 0 \le \theta.$$

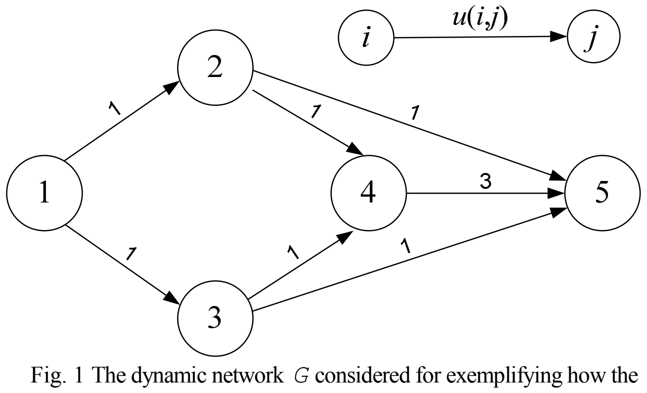

Fig. 1 The dynamic network G considered for exemplifying how the BiMCMDF algorithm works

In the initialisation step, the discrete dynamic maximum flow of minimum cost f_1 is computed by procedure SSP(c) and the maximum flow of minimum transit time f_2 is computed by procedure SSP(h). The two efficient extreme points in the objective space, $Y_1 = (y_c(f_1), y_h(f_1))$ and $Y_2 = (y_c(f_2), y_h(f_2))$, corresponding to the computed efficient flows f_1 and f_2, will take the following values: $Y_1 = (25, 11)$ and $Y_2 = (34, 7)$.

The flows f_1 and f_2 are added to the set of efficient extreme flows in the decision space F_{ex} and the efficient extreme points in the objective space Y_1 and Y_2 are added to the set Y_{ex}. Since $Y_1 \neq Y_2$, the set Q is initialised as $Q = \{(1,2)\}$ and the slope between the two efficient extreme points Y_1 and Y_2 is investigated. For $\alpha := y_c(f_2) - y_c(f_1) = 9$ and $\beta := y_h(f_1) - y_h(f_2) = 4$, the following artificial costs are computed:

$$c'(1,2) == \begin{cases} 26, & 0 \le \theta < 1 \\ 35, & \theta \ge 1 \end{cases}, \quad c'(1,3) == \begin{cases} 17, & 0 \le \theta < 2 \\ 26, & \theta \ge 2 \end{cases},$$

$$c'(2,4) == \begin{cases} 55, & 0 \le \theta < 2 \\ 37, & \theta \ge 2 \end{cases}, \quad c'(3,4) = \begin{cases} 34, & 0 \le \theta < 2 \\ 29, & \theta \ge 2 \end{cases},$$

$$c'(3,5) = \begin{cases} 37, & 0 \le \theta < 2 \\ 57, & \theta \ge 2 \end{cases}, \quad c'(2,5) = 45,\ 0 \le \theta,\ c'(4,5) = 13,\ 0 \le \theta.$$

The algorithm makes a call to procedure $\mathrm{SSP}(c')$ which will initialize to 0 the flow values over all the arcs and the predecessor vector to $p(1,\theta) := 0$ for all $\theta \in \{0,1,\ldots,T\}$. Then, while the sink node 5 is reachable from the source node 1, at any moment $\theta \in \{0,1,\ldots,T\}$, the algorithm finds a shortest augmenting path from the source to the sink relative to the artificial costs and augments the flow along this path with the corresponding residual capacity. The shortest augmenting paths, in increasing order of their artificial cost values are: P_1=((1,0),(3,1),(5,2)), P_2=((1,1),(3,2),(4,3),(5,4)) and P_3=((1,0),(2,2),(5,3)) with $r(P_1) = r(P_2) = r(P_3) = 1$, $c'(P_1) = 54$, $c'(P_2) = 59$ and $c'(P_3) = 71$.

The new efficient point, corresponding to the computed flow f', is $Y' = (y_c(f'), y_h(f')) = (28,8)$. Since both $Y' \neq Y_1$ and $Y' \neq Y_2$, the value of the index k is incremented to 3, indicating that there have been found 3 efficient extreme points in the objective space so far. The flow $f_3 := f'$ is added to the set of efficient extreme flows in the decision space F_{ex}, the corresponding value $Y_3 := Y'$ is added to the set of efficient extreme point in the objective space and the set Q is updated by removing the pair $(1,2)$ from it and adding the two new pairs of indices $(1,3)$ and $(3,2)$.

Then the algorithm selects the pair of indices $(1,3)$, computes the slope between Y_1 and Y_3 for which $\alpha := y_c(f_3) - y_c(f_1) = 3$ and $\beta := y_h(f_1) - y_h(f_3) = 3$. For the new artificial costs $c'(i,j;\theta) = 3 \cdot c(i,j;\theta) + 3 \cdot h(i,j;\theta)$, the shortest augmenting paths, in increasing order of their cost values are: P_1=((1,0),(3,1),(5,2)), P_2=((1,1),(3,2),(4,3),(5,4)) and P_3=((1,0),(2,2),(5,3)), leading to the efficient point $Y' = (28,8)$. Since $Y' = Y_3$, the pair $(1,3)$ is simply removed from the set Q. Finally, the algorithm selects the last pair of indices from Q and, based on the slope between the two extreme points Y_3 and Y_2 computes the efficient point $Y' = (28,8)$. Since $Y' = Y_3$ again, the pair of indices $(3,2)$ is removed from the set Q which becomes empty and the algorithm terminates. The objective space with the three efficient extreme points is presented in Fig. 2.

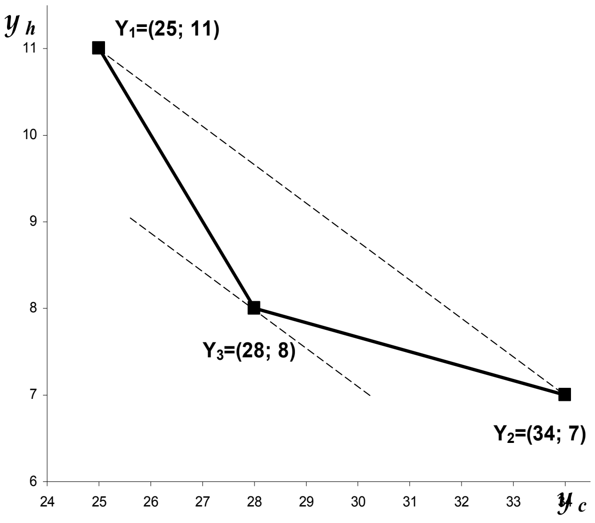

Fig. 2 The set of all non-dominated points which lie on the efficient boundary in the objective space for the discrete dynamic maximum flow of bi-criteria minimum cost and minimum travel time problem in network G

References

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

[2] Aneja, Y.P., Nair, K.P. "Bicriteria transportation problem" *Management Science*, 25(1), 1979.

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

[4] Calvete, H.I., Mateo, P.M. "An approach for the network flow problem with multiple objectives" *Computers and Operations Research*, 22(9), 1995, pp. 971-983.

[5] Chabini, I., Abou-Zeid, M. "The Minimum Cost Flow Problem in Capacitated Dynamic Networks" *Annual Meeting of the Transportation Research Board*, Washington, D.C., 2003.

[6] Ciupală, L. "About flow problems in networks with node capacities", *WSEAS Transactions on Computer Research*, 8(8), 2009, pp. 1266-1275.

[7] Ciurea, E., Ciupală, L. *ALGORITMI Introducere în algoritmica fluxurilor în rețele*. Ed. MATRIX ROM. București, 2006.

[8] Ciurea, E., Parpalea, M. “Minimum Flow in Monotone Parametric Bipartite Networks” *NAUN International Journal of Computers*, 4(4), 2010, pp. 124-135.

[9] Ciurea, E., Parpalea, M. “Balancing Algorithm for the Minimum Flow Problem in Parametric Bipartite Networks” *in Latest Trends on Computers*, *Proceedings of the 14th WSEAS International Conference on Computers*, 2010, pp. 226-231.

[10] Fonoberova, M. *Optimal Flows in Dynamic Networks and Algorithms for their Finding*. Institute of Mathematics and Computer Science, Academy of Sciences of Moldova, 2007.

[11] Fleischer, L., Skutella, M. “Minimum Cost Flows Over Time Without Intermediate Storage” *in Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, 2003, pp. 66–75.

[12] Ford, L. R., Fulkerson, D. R. “Constructing maximal dynamic flows from static flows” *Operations Research*, vol.6, 1958, pp. 419–433.

[13] Hamacher, H.W., Pedersen, C.R., Ruzika, S. “Multiple Objective Minimum Cost Flow Problems: A Review” *Department of Operations Research, University of Aarhus, Denmark,* 2005.

[14] Klinz, B., Woeginger, G. “Minimum cost dynamic flows: The series-parallel case” *in Integer Programming and Combinatorial Optimization*, vol. 920, *Lecture Notes in Computer Science*, Springer, 1995, pp. 329–343.

[15] Kostreva, M., Wiecek, M. “Time Dependency in Multiple Objective Dynamic programming” *Mathematical Analysis and Applications*, 173(1), 1993, pp. 289-307.

[16] Lee, H., Pulat, S. “Bicriteria network flow problems: Continuous case” *European Journal of Operational Research*, 51, 1991, pp. 119-126.

[17] Lee, H., Pulat, S. “Bicriteria network flow problems: Integer case” *European Journal of Operational Research*, 66, 1993, pp. 148-157.

[18] Miller-Hooks, E., Patterson, S.S. “On solving quickest time problems in time-dependent, dynamic networks” *Journal of Mathematical Modelling and Algorithms*, 3, 2004, pp. 39–71.

[19] Nasrabadi, E., Hashemi, S.M. “Minimum cost time-varying network flow problems” *Optimization Methods and Software*, 25(3), 2010, pp. 429-447.

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

[21] Ruhe, G. *Algorithmic Aspects of Flows in Networks*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.

[22] Sângeorzan, L., Parpalea, M., et al. “Multi-objective Optimization Technique for Simulated Active Body Control with Frictional Contacts” *in Mathematical Methods and Applied Computing, Proceedings of the 11th WSEAS International Conference on Mathematical Methods & Computational Techniques in Electrical Engineering*, 2009, pp. 594-598.

[23] Sângeorzan, L., Parpalea, M., et al. “Partitioning preflow-pull algorithm for the parametric network flow problem – a Linguistic rule-based constraints optimisation approach” *in Recent Advances in Neural Networks, Fuzzy Systems & Evolutionary Computing, Proceedings of the 11th WSEAS International Conference on Neural Networks*, 2010, pp. 111-116.

[24] Sedeño-Noda, A., González-Martín, C. “An algorithm for the biobjective integer minimum cost flow problem” *Computers and Operations Research*, 28, 2001, pp. 139-156.

[25] Skriver, A. “A Classification of Bicriterion Shortest Path (BSP) Algorithms” *Asia-Pacific Journal of Operational Research*, 17, 2000, pp. 199-212.

[26] Skutella, M. “An Introduction to Network Flows Over Time” *Research Trends in Combinatorial Optimization*, Springer, 2008, pp. 451-482.

[27] Tjandra, S.A. *Dynamic Network Optimization with Application to the Evacuation Problem*. Shaker, 1. edition, 2003.