Optimization Letters (2021) 15:1009–1026
https://doi.org/10.1007/s11590-019-01519-5

ORIGINAL PAPER

Minimum cost multicommodity network flow problem in time-varying networks: by decomposition principle

Salman Khodayifar[1]

Received: 16 December 2018 / Accepted: 4 December 2019 / Published online: 10 December 2019

Abstract
Time-varying network flows (also called dynamic network flows) generalize standard network flows by introducing an element of time. In this paper, we consider the dynamic version of the minimum cost multicommodity flow problem with storage at intermediate nodes. This problem is known to be NP-hard. By using of the flow decomposition theorem in network flows, we propose an efficient model based on dynamic path flows for this problem. For some special structures of the path-flow formulation, we provide an algorithm based on decomposition principle, for solving the proposed model. In the end, the efficiency of the proposed approach is evaluated through a number of experimental tests.

Keywords Minimum cost flow problem · (Dynamic) multicommodity flows · Combinatorial optimization · Decomposition principle

1 Introduction

Classical (static) network flow models have been well known as valuable tools for many applications (see, e.g., [1,11]). However, they fail to capture a crucial element of many routing problems in real-world applications: routing occurs over time such as routing in logistics and transportation, wireless sensor networks, airline, location and layout of facilities, traffic planning, telecommunications, social and biological networks, and other fields. A static flow can not properly consider the evolution of a system over time. The time here is an essential component, either because the flows of some commodity take time to pass from one location to another, or because the topology of the network changes over time. Various interesting examples can be found in the survey articles of [2,15,20,21]. This class of network flows was first introduced

✉ Salman Khodayifar
s.khodayifar@iasbs.ac.ir

[1] Department of Mathematics, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran

by Ford and Fulkerson [12]. Ford and Fulkerson introduced flows over time (also called dynamic flows) and considered the problem of sending the maximal possible amount of flow from a source node s to a sink node t on the networks with only transit times within a given time horizon T. They shown that this problem can be solved in polynomial time by using one min-cost flow computation on the given network [12]. Since then, a large number of authors have studied different features of time varying networks (see, e.g., [6,7,10,14,16–18]).

Flows over time are meaningfully harder than their static counterparts. Whenever some network parameters such as cost, capacity, and supply/demand vary with time, Klinz and Woeginger shown that both minimum cost $s-t$-flows over time and multicommodity flows over time are NP-hard, even for very simple series-parallel network flows or to the case of only two commodity [17]. Traditionally, flows over time are computed in time-expanded networks that contains one copy of the original network for each discrete time point (building a time layer). A discrete flow over time in the given network can be interpreted as a static flow in the corresponding time-expanded network. While this method makes available the whole algorithmic toolbox developed for static flows, its main and often serious drawback is the enormous size of the time-expanded network. However, the time complexity of such algorithms are pseudo-polynomial in time-expanded networks and thus dose not directly lead to efficient algorithms for computing flows over time.

Fleischer and Skutella introduced a condensed variant of time-expanded network that leads to network whose size is polynomially bounded in the input size. They gave approximation algorithms for various variants of the network flows problem over time [8–10]. Also, Hall et al. [13] presented efficient algorithms under certain assumptions on the transit times of arcs and the network topology.

In the network flows, the multicommodity flow problem consists of shipping several different commodities from their respective sources to their sinks through a given network so that the total flow going through each arc does not exceed its bundle capacity. The multicommodity network flow problem requires to find the minimum cost flow of a set of commodities through a network, where the arc flows satisfied the network requirements.

The multicommodity network flow problems are significantly harder than their single-commodity counterparts. For example, the only known polynomial-time algorithms for static multicommodity flow computations are general linear programming (for short, LP) techniques (e.g., the ellipsoid or interior point methods) [1]. The dynamic multicommodity flow problem is the generalization of the static multicommodity flow problem in which cost and bundle capacity on arcs or supply/demand values are time-varying in a time horizon T.

While dynamic minimum cost flow problem has been known to be NP-hard, the complexity of the dynamic multicommodity flow problem has been open for many years until Hall and et al. [13] have shown that this problem is NP-hard, even for series-parallel networks or to the case of only two commodity.

Fleischer an Skutella shown that for single commodity problems, storage at intermediate nodes is unnecessary [9]. Hall et al. proved that storage of flow at intermediate nodes can be useful in the multicommodity flow problems. They shown that the multiple commodity flow over time problem with simple paths solutions and forbidding

flow at intermediate nodes is strongly NP-hard. Of course, this problem can be solved in pseudo-polynomial time in the time-expanded network. The best result for the multicommodity flow problem is 2-approximation algorithm for the quickest multicommodity flow problem (provided by Fleischer and Skutella [10]).

Also, Hall et al. presented efficient algorithms under certain assumptions on the transit times of arcs and the network topology. In fact, they assumed that all paths between every fixed pair of nodes have the same transit time. They shown that under this assumption the size of the time-expanded network corresponding to many flow over time problems is polynomial. Also, Lozovanu, and Fonoberova solved this problem by time-expanded network that contains one copy of the original network for each discrete time step. Its main drawback is the enormous size of the time-expanded network [19].

Contribution of this paper Motivated by the above results and hardness of multicommodity flows variation over time, we are interested in the dynamic version of the minimum cost multicommodity flow problem on networks with storage at intermediate nodes and fixed transit time on arcs. We assume that cost and bundle capacity defined on arcs and capacity storage on nodes are time-varying. This problem can be solved in pseudo-polynomial time in the time-expanded network. This method is never effective for large-scale networks. By using of the flow decomposition theorem in network flows [1], we propose an efficient model based on dynamic path flows and show that the size of this model is smaller than the arc-flow based model. Since, the number of dynamic paths, is usually very large, attempting to explicitly finding all the dynamic paths, and explicitly solving the proposed model is a very difficult task. We want to find an optimal solution of this problem without explicitly enumerating all the dynamic paths. According to the special structure of the proposed model, we solve it based on a technique of decomposition principle (inspired by revised simplex method) that find an optimal solution of this problem without explicitly enumerating all the dynamic paths.

The rest of this paper is organized as follows. In Sect. 2, model of static multicommodity flow problem based on arc-flow formulation and its path-flow formulation counterpart are presented. The dynamic multicommodity flow problem with storage at intermediate nodes is proposed in Sect. 3. In Sect. 4, a solution methodology based on decomposition principle is presented. In Sect. 5, the efficiency of the proposed approach is evaluated through a number of experimental tests. Finally, the paper ends with some conclusions in Sect. 6.

2 Multicommodity flows and decomposition principle

Let $G = (N, A)$ be a directed network with node set N ($|N| = n$), arc set A ($|A| = m$), and set of commodities $K = \{1, 2, \ldots, h\}$ that must be routed through the same network. Every commodity $k \in K$ has only one source $s_k^+ \in N$ and one sink $s_k^- \in N$ and R_k is the amount of supply or demand of commodity $k \in K$. Each arc $(i, j) \in A$ has a capacity u_{ij} that restricts the total flow of all commodities on that arc. For commodity k, let $\mathbf{x}^k = (x_{ij}^k)_{(i,j)\in A}$, $\mathbf{d}^k = (d_i^k)_{i\in N}$, and $\mathbf{c}^k = (c_{ij}^k)_{(i,j)\in A}$ present the flow vector, supply/demand vector, and per unit cost vector. Using defined notations, the multicommodity flow (for short, MCF) problem can be formulated as follows:

$$\text{MCF}: \min \sum_{k=1}^{h} \sum_{(i,j)\in A} c_{ij}^k x_{ij}^k \tag{2.1}$$

s.t.

$$\sum_{k=1}^{h} x_{ij}^k \le u_{ij}, \qquad \forall (i,j) \in A, \tag{2.2}$$

$$\sum_{\{j:(i,j)\in A\}} x_{ij}^k - \sum_{\{j:(j,i)\in A\}} x_{ji}^k = d_i^k, \qquad \forall i \in N,\ k \in K, \tag{2.3}$$

$$x_{ij}^k \ge 0, \qquad \forall (i,j) \in A,\ k \in K, \tag{2.4}$$

where

$$d_i^k = \begin{cases} R_k & \text{if } i = s_k^+ \\ -R_k & \text{if } i = s_k^- \\ o & o.w. \end{cases}$$

The objective function (2.1) minimizes the total cost of the multicommodity flow. Constraints (2.2) implement the bundle constraint on each arc $(i, j) \in A$. Constraints (2.3) are separate flow conservation constraints for each commodity $k \in K$. Constraints (2.4) are the continuous restrictions on the decision variables. This formulation is an LP model with $|A||K|$ continuous variables and $|K||N| + |A|$ constraints. Since the MCF model is an LP, it can be solved in polynomial time by using of general LP techniques for example the ellipsoid method or interior point methods [1]. In practice, the MCF problem is too large. It usually contains many thousands of rows and a seemingly unlimited number of columns and some method must be applied to convert the problem into one or more smaller problems of desirable size. The decomposition principle does exactly this. The decomposition principle is a procedure that separate the original problem into one with general structure and one with special structure where an efficient method can be applied.

Due to the fact that the constraints (2.2) have a special structure (block diagonal), it can be solved by decomposition principle. The block diagonal form of the MCF model is as follows:

$$\begin{array}{lllll} \min & c^1x^1 + c^2x^2 + \cdots + c^hx^h & & & \\ \text{s.t.} & & & & \\ & A^1x^1 + A^2x^2 + \cdots + A^hx^h & \le u, & & \\ & D^1x^1 & = d^1, & & \\ & & D^2x^2 & = d^2, & \\ & & \ddots & \vdots & \\ & & & D^hx^h & = d^h, \\ & x^1, & x^2, \quad \ldots, & x^h & \ge 0, \end{array}$$

where $A_1 = \cdots = A^h = I_{m\times m}$, $D^1 = \cdots = D^h = D_{n\times m}$.

Matrix $I_{m\times m}$ is the identity matrix of size $m\times m$ and $D_{n\times m}$ is the node-arc incidence matrix of a network that has exactly one $+1$ and one -1 in each column, and a zero elsewhere. Note that $D\in\mathbb{R}^{n\times m}$ and $rank(D)=n-1$. This model can be solved by the classical Wolfe-Dantzig decomposition method [1].

According to the flow decomposition theorem, we can reformulate the MCF problem using path-cycle flows instead of arc flows. The path-cycle flow formulation of the MCF problem has a simple constraint structure than the block diagonal form. In our study of the MCF problem, we will impose below assumption.

Assumption 2.1 (Non-negative cost condition) For every commodity, the underlying network does not contain a negative cycle (i.e., a directed cycle with negative length).

Assumption 2.1 implies that in some optimal solution, the flow on each cycle is zero. Therefore, with Assumption 2.1, we can delete the cycle flow variables.

For each commodity k, let P^k denote the collection of all paths from the source node s_k^+ to the sink node s_k^- in the network $G=(N,A)$. In the path-flow formulation, each decision variable $f(p)$ is the flow on some path p and for the kth commodity, we define this variable for every path $p\in P^k$.

For each commodity k and every path $p\in P^k$, arc-path indicator variable $\delta_{ij}(p)$ is defined as follows:

$$\delta_{ij}(p)=\begin{cases}1, & \text{if } (i,j)\in p\\ 0, & \text{o.w.}\end{cases}$$

By using of the flow decomposition theorem in the network flows (with Assumption 2.1), we can obtain the following equivalent path-flow (for short, PF) formulation of the MCF problem:

$$PF:\min\sum_{k=1}^{h}\sum_{p\in P^k}c^k(p)f(p) \tag{2.5}$$

s.t.

$$\sum_{k=1}^{h}\sum_{p\in P^k}\delta_{ij}(p)f(p)\le u_{ij},\qquad \forall(i,j)\in A, \tag{2.6}$$

$$\sum_{p\in P^k}f(p)=R_k,\qquad \forall k\in K, \tag{2.7}$$

$$f(p)\ge 0,\qquad \forall k\in K,\ p\in P^k. \tag{2.8}$$

The PF formulation of the MCF problem has a single constraint (2.6) for each arc $(i,j)\in A$ which indicates that the sum of the path flows passing through the arc (i,j) is at most u_{ij}. Also, for each commodity k, the model has a single constraint (2.7) which indicates that the total flow on all the paths from the source node s_k^+ to the sink node s_k^- must be equal the demand of the commodity k. The PF formulation of the MCF problem can be solved in polynomial time [22].

It is worth noting that the PF formulation contains $|A|+|K|$ constraints and therefore any LP basis for the PF formulation will contains $|A|+|K|$ basic variables (because at least one basic variable must be in each of the $|K|$ constraints (2.7), that is, in any basis at least one path must carry a positive flow for every commodity $k = 1, 2, \ldots, h$). Also, the arc-flow formulation contains $|K||N| + |A|$ constraints and therefore any LP basis for the arc flow formulation or block diagonal structure formulation of the MCF will contains $|K||N| + |A|$ basic variables. So, the PF formulation of the MCF problem has a simple constraint and it can efficiently be solved by decomposition principle (inspired by the revised simplex method of LP [3]).

In the next section, we describe the MCF problem on dynamic network flows.

3 Dynamic multicommodity flow problem with storage at intermediate nodes

We consider a directed network $G = (N, A, K, c, u, \tau, \mathcal{T})$ with set of nodes $|N| = n$, set of arcs $|A| = m$, and set of commodities $K = \{1, 2, \ldots, h\}$ that must be routed through the same network. We consider the discrete time model, in which all times are integral and bounded by time horizon T, which defines the set $\mathcal{T} = \{0, 1, \ldots, T\}$ of time moments. We define $c^k_{ij}(t)$ and $c^k_i(t)$ as the cost for sending one unit of flow kth on arc (i, j) in time t and the cost for storing one unit of flow kth at node i from time $t-1$ to t, respectively. Moreover, $u_{ij}(t)$, $u^k_i(t)$, and $\tau_{ij}(t)$ are an upper bound on the amount of flow that can enter to arc (i, j) at time $t \in \mathcal{T}$, an upper bound on the amount of flow that can be stored in node i from time $t-1$ to t, and positive transmit time on arc (i, j) at time $t \in \mathcal{T}$, respectively.

Time is measured in discrete steps, so that if one unit of flow of commodity k leaves node i at time t, one unit of flow arrives at node j at time $t+\tau_{ij}(t)$. For commodity k, let $\mathbf{x}^k = (x^k_{ij}(t))_{(i,j)\in A}$ denote the dynamic flow vector and flow variables $x^k_{ij}(t)$ present the flow of the commodity k on arc (i, j) at time t. We assume that the flow variables $x^k_{ij}(t)$ have no individual flow bounds; that is, each $u^k_{ij}(t) = +\infty$. For commodity k, let $\mathbf{y}^k = (y^k_i(t))_{i\in N}$ denote the storage vector and $y^k_i(t)$ gives the amount of commodity flow k stored at node i from time $t-1$ to t.

Using this notations, the discrete dynamic multicommodity flow (for short, DDMF) problem with storage at intermediate nodes can be modeled as follows:

$$\text{DDMF}: \min \sum_{t=0}^{T} \left(\sum_{k\in K} \sum_{(i,j)\in A} c^k_{ij}(t) x^k_{ij}(t) + \sum_{k\in K} \sum_{i\in N} c^k_i(t) y^k_i(t) \right) \tag{3.1}$$

s.t.

$$\sum_{k\in K} x^k_{ij}(t) \le u_{ij}(t), \quad \forall (i, j) \in A,\ t \in \mathcal{T} \tag{3.2}$$

$$\sum_{\{j:(i,j)\in A\}} x^k_{ij}(t) - \sum_{\{j:(j,i)\in A\}} \sum_{\{t':t'+\tau_{ji}(t')=t\}} x^k_{ji}(t') + y^k_i(t) - y^k_i(t-1) = 0, \quad \forall k \in K,\ i \in N - \{s^+_k, s^-_k\},\ t \in \mathcal{T} \tag{3.3}$$

$$\sum_{t=0}^{T}\left(\sum_{\{j:(i,j)\in A\}} x_{ij}^{k}(t)-\sum_{\{j:(j,i)\in A\}}\sum_{\{t':t'+\tau_{ji}(t')=t\}} x_{ji}^{k}(t')\right)=d_i^k,\quad \forall i\in N,\ k\in K, \tag{3.4}$$

$$x_{ij}^{k}(t)\geq 0,\ 0\leq y_i^k(t)\leq u_i^k(t),\quad \forall (i,j)\in A,\ i\in N,\ k\in K,\ t\in\mathcal{T}, \tag{3.5}$$

where

$$d_i^k=\begin{cases} R_k & if\ i=s_k^+\\ -R_k & if\ i=s_k^-\\ o & o.w.\end{cases}$$

The objective function (3.1) minimizes the total cost of the dynamic flow vector $\mathbf{x}^k$ and storage flow vector $\mathbf{y}^k$ in time horizon T. Constraints (3.2) implement the bundle constraint on each arc $(i,j)\in A$ at any time step t. For commodity k, constraints (3.3) indicate the flow conservation constraints and amount of stored flow at intermediate node i in each time step t. For commodity k, constraints (3.4) indicate the flow conservation constraints in time horizon T. That is, after all the time steps have passed, the dynamic flow vectors $\mathbf{x}^k$, $\forall k\in K$ should be satisfy the supply/demand of every commodity and there are no additional flows in the intermediate nodes. Constraints (3.5) indicate capacity constraint and storage capacity constraint for dynamic flow vectors $\mathbf{x}^k$, $\forall k\in K$ and storage vectors $\mathbf{y}^k$, $\forall k\in K$, respectively. However, in the DDMF problem we want to find a dynamic flow vector $\mathbf{x}^k$, $\forall k\in K$ and a storage vector $\mathbf{y}^k$, $\forall k\in K$ such that satisfy the constraints (3.2)–(3.5) and minimize the total cost (3.1).

The DDMF model can be solved by the typical approach called time-expanded network technique. As mentioned earlier, the size of the time-expanded network is linear in time horizon T. Thus, any polynomial time algorithm for solving minimum cost flow problem on the time-expanded network will yield a pseudo-polynomial time algorithm for the DDMF problem. In the next section, we reformulate the DDMF problem (with storage at intermediate nodes) with dynamic path flows and solve it by a decomposition approach inspired by revised simplex method.

4 Solution method

4.1 Reformulation with dynamic path flows

Let $G=(N,A,K,c,u,\tau,\mathcal{T})$ be a directed dynamic network with set of commodities $K=\{1,2,\ldots,h\}$ that must be routed through the same network, time varying cost vector $\mathbf{c}^k=(c_{ij}^k(t))_{(i,j)\in A}$, fixed transit time vector $\boldsymbol{\emptyset}=(\tau_{ij})_{(i,j)\in A}$, time varying capacity vector $\mathbf{u}=(u_{ij}(t))_{(i,j)\in A}$, time varying capacity storage vector $\mathbf{u^k}=(u_i^k(t))_{(i)\in N}$, and discrete time steps $\mathcal{T}=\{0,1,\ldots,T\}$. We suppose that the storage of flow at intermediate nodes is allowed and transit time on arcs fixed assumed.

Below, we present the path-flow formulation corresponding to the DDMF problem with storage at intermediate nodes. Before stating the path-flow formulation, we indicate some definitions.

In the following, we refer to (i, α) of $N \times \mathcal{T}$ as a node-time pair (for short, NTP) (i.e., a node in a arbitrary time step). For an arc $(i, j) \in A$, we say that the NTP (i, α) is arc-inked to the NTP (j, β) if $\beta = \alpha + \tau_{ij}$. Also, we say that the NTP (i, α) is node-linked to the NTP (j, β) if $i = j$ and $\alpha \neq \beta$.

Definition 4.1 In the dynamic network G, for a commodity k, a dynamic path from node s_k^+ with departure time α to node s_k^- is a sequence of distinct NTPs (arc-inked or node-linked) as follows:

$$p^\alpha : (s_k^+, \alpha) = (i_1, t_1), (i_2, t_2), \ldots, (i_r, t_r) = (s_k^-, \beta).$$

For each commodity k, let P^k denote the collection of dynamic paths from the source node s_k^+ to the sink node s_k^- in the dynamic network G on time horizon T with a fixed departure time. It is worth noting that for any dynamic path $p \in P^k$, we have $T - \tau_p$ dynamic path (paths) p^α with departure times $\alpha = 0, 1, \ldots, T - \tau_p$ where we let $\tau_p = \sum_{(i,j)\in p} \tau_{ij}$ be the total transit time of p. In the path-flow formulation, each decision variable $f(p^\alpha)$ is the flow on some dynamic path p with departure time α from node s_k^+. For the kth commodity, we define this variable for every dynamic path $p \in P^k$ with departure times $\alpha = 0, 1, \ldots, T - \tau_p$. In fact, decision variables $f(p^\alpha)$ are not defined for $\alpha > T - \tau_p$. For the convenience of work, we assume $f(p^\alpha) = 0$ for $\alpha > T - \tau_p$.

For a given dynamic path $p \in P^k$ with departure time α in the form of

$$p^\alpha : (s_k^+, \alpha) = (i_1, t_1), (i_2, t_2), \ldots, (i_r, t_r) = (s_k^-, \beta),$$

arc-path indicator variable $\delta_{ij}(p^\alpha, t)$ for $t \geq \alpha$ and $(i, j) \in A$ is defined as follows:

$$\delta_{ij}(p^\alpha, t) = \begin{cases} 1 & \text{if } (i, t) \text{ is arc-linked to } (j, t + \tau_{ij}) \text{ on dynamic path } p^\alpha \\ 0 & \text{o.w.} \end{cases}$$

and node-path indicator variable $\gamma_i(p^\alpha, t)$ for $t \geq \alpha$ and $i \in N$ is defined as follows:

$$\gamma_i(p^\alpha, t) = \begin{cases} 1 & \text{if } (i, t) \text{ is node-linked to } (i, t + 1) \text{ on dynamic path } p^\alpha \\ 0 & \text{o.w.} \end{cases}$$

Without loss of generality, we assume $\delta_{ij}(p^\alpha, t) = \gamma_i(p^\alpha, t) = 0$ for $t < \alpha$.

Definition 4.2 For commodity k, the cost of a dynamic path

$$p^\alpha : (s_k^+, \alpha) = (i_1, t_1), (i_2, t_2), \ldots, (i_r, t_r) = (s_k^-, \beta),$$

is defined as

$$c^k(p^\alpha) = \sum_{(i_r,i_{r+1})\in p^\alpha, i_r\neq i_{r+1}} c^k_{i_r,i_{r+1}}(t_r) + \sum_{(i_r,i_{r+1})\in p^\alpha, i_r=i_{r+1}} c^k_r(t_r)$$
$$= \sum_{t=0}^{T}\sum_{(i,j)\in A} \delta_{ij}(p^\alpha,t)c^k_{ij}(t) + \sum_{t=0}^{T}\sum_{i\in N} \gamma_i(p^\alpha,t)c^k_i(t).$$

Definition 4.3 For commodity k, given a dynamic flow vector $\mathbf{x}^k$ and flow storage vector $\mathbf{y}^k$, a path p^α is said to be dynamic shortest path from NTP (s_k^+,α) to NTP (s_k^-,β), if $c^k(p^\alpha) \le c^k(p'^\alpha)$ for all dynamic paths $p' \in P^k$ from NTP (s_k^+,α) to NTP (s_k^-,β).

Definition 4.4 According to the flow decomposition theorem of network flows (with considering the Assumption 2.1), for commodity k in time step t, arc flows $x^k_{ij}(t)$ and storage flows $y^k_i(t)$ can be calculated as follows:

$$x^k_{ij}(t) = \sum_{p\in P^k}\sum_{\alpha=0}^{T-\tau_p} \delta_{ij}(p^\alpha,t)f(p^\alpha),$$
$$y^k_i(t) = \sum_{p\in P^k}\sum_{\alpha=0}^{T-\tau_p} \gamma_i(p^\alpha,t)f(p^\alpha).$$

By considering the above definitions, the objective function (3.1) can be rewritten as follows:

$$\sum_{t=0}^{T}\sum_{k\in K}\left(\sum_{(i,j)\in A} c^k_{ij}(t)x^k_{ij}(t) + \sum_{i\in N} c^k_i(t)y^k_i(t)\right)$$
$$= \sum_{t=0}^{T}\sum_{k\in K}\left(\sum_{(i,j)\in A} c^k_{ij}(t)\left(\sum_{p\in P^k}\sum_{\alpha=0}^{T-\tau_p}\delta_{ij}(p^\alpha,t)f(p^\alpha)\right)\right.$$
$$\left. + \sum_{i\in N} c^k_i(t)\left(\sum_{p\in P^k}\sum_{\alpha=0}^{T-\tau_p}\gamma_i(p^\alpha,t)f(p^\alpha)\right)\right)$$
$$= \sum_{k\in K}\sum_{p\in P^k} f(p^\alpha)\sum_{\alpha=0}^{T-\tau_p}\left(\sum_{t=0}^{T}\sum_{(i,j)\in A}\delta_{ij}(p^\alpha,t)c^k_{ij}(t) + \sum_{t=0}^{T}\sum_{i\in N}\gamma_i(p^\alpha,t)c^k_i(t)\right)$$
$$= \sum_{k\in K}\sum_{p\in P^k}\sum_{\alpha=0}^{T-\tau_p} c^k(p^\alpha)f(p^\alpha).$$

This observation shows that we can express the cost of any dynamic solution as either the cost of dynamic arc-flows or the cost of dynamic path flows.

By substituting the path-flow variables in the DDMF formulation, we obtain the following equivalent dynamic path flow formulation (hereafter, we call DPF problem) of the DDMF problem:

$$DPF: \min \sum_{k\in K}\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p} c^k(p^\alpha) f(p^\alpha) \tag{4.1}$$

s.t.

$$\sum_{k\in K}\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p} \delta_{ij}(p^\alpha, t) f(p^\alpha) \le u_{ij}(t), \qquad \forall (i,j)\in A,\ t\in\mathcal{T}, \tag{4.2}$$

$$\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p} f(p^\alpha) = R_k, \qquad \forall k\in K, \tag{4.3}$$

$$\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p} \gamma_i(p^\alpha, t) f(p^\alpha) \le u_i^k(t), \quad \forall i\in N-\{s_k^+, s_k^-\},\ k\in K,\ t\in\mathcal{T}, \tag{4.4}$$

$$f(p^\alpha)\ge 0, \qquad \forall\, p\in P^k,\ k\in K, \alpha = 0,1,\ldots,T-t_p. \tag{4.5}$$

Note that the DPF formulation of the DDMF problem has a very simple constraint structure. The problem has Tm constraints (4.2) for each arc (i, j) which state that the sum of dynamic path flows passing through the arc (i, j) at time t is at most $u_{ij}(t)$. Moreover, the problem has a single constraint (4.3) for each commodity k which states that the sum of dynamic path flows connecting the source node s_k^+ and sink node s_k^- of commodity k must be equal the demand R_k for this commodity on time horizon T. For every NTP (i, t) and commodity k, the problem has a constraint (4.4) which states that the sum of dynamic path flows passing through the node i at time t is at most $u_i^k(t)$. In other words, this constraint shows that the capacity storage for the commodity k in time t at intermediate node i is at most $u_i^k(t)$. The constraints (4.5) guarantee that $f(p^\alpha) = 0$ for all $t\notin\{0, 1, \ldots, T - t_p\}$. In fact, in order to one unit of flow reach to its destination after time horizon T, it must leave its source before a certain time.

For a network with n nodes, m arcs, and h commodities, the DPF problem contains $mT + nhT + h$ constraints. In contrast, the arc formulation DDMF contains $mT + nhT + nh$ constraints since it contains one mass balance constraint for every node and commodity combination.

Since, the number of dynamic paths, is usually very large, attempting to explicitly finding all the dynamic path, and explicitly solving the DPF problem very difficult task. However, we expect that only very few of the dynamic paths will carry flow in the optimal solution to the DPF problem. In fact, from LP theory, at most $mT + nhT + h$ dynamic paths carry positive flow in some optimal solution. We want to find an optimal solution of this problem without explicitly enumerating all the dynamic

paths. Therefore, it is useful to design an algorithm based on decomposition principle for the DPF problem (inspired by revised simplex method).

4.1.1 Optimality conditions

The revised simplex method determines a basis at every step, and using this basis finds a vector of dual variables for the constraints. Suppose that the dual variables corresponding to constraints (4.2)–(4.4) are $w_{ij}(t)$, σ^k, and $v_i^k(t)$, respectively. With respect to these dual variables, the reduced cost $c_{p^\alpha}^{w,v,\sigma}$ for commodity k and dynamic path p^α from NTP (s_k^+, α) to NTP (s_k^-, β) (corresponding to the dynamic path flow variable $f(p^\alpha)$) is defined as follows:

$$c_{p^\alpha}^{w,v,\sigma} = -\sigma^k + \sum_{(i_r,i_{r+1})\in p^\alpha, i_r\neq i_{r+1}} \Big(w_{i_r,i_{r+1}}(t_r) + c^k_{i_r,i_{r+1}}(t_r)\Big) + \sum_{(i_r,i_{r+1})\in p^\alpha, i_r=i_{r+1}} \Big(v^k_{i_r}(t_r) + c^k_{i_r}(t_r)\Big).$$

Dynamic path flow complementary slackness conditions The dynamic path flows $f(p^\alpha)$ are optimal in the DDMF formulation of the dynamic multicommodity flow problem if and only if for some dual variables $w_{ij}(t)$, σ^k, and $v_i^k(t)$, the following complementary slackness conditions are established

$$(1)\ w_{ij}(t)\Big(\sum_{k\in K}\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p}\delta_{ij}(p^\alpha,t)f(p^\alpha) - u_{ij}(t)\Big) = 0, \quad \forall (i,j)\in A,\ t\in\mathcal{T},$$

$$(2)\ v_i^k(t)\Big(\sum_{p\in P^k}\sum_{\alpha=0}^{T-t_p}\gamma_i(p^\alpha,t)f(p^\alpha) - u_i^k(t)\Big) = 0,$$

$$\forall k\in K, i\in N-\{s_k^+, s_k^-\},\ t\in\mathcal{T},$$

$$(3)\ c_{p^\alpha}^{w,v,\sigma}\geq 0, \quad \forall k\in K,\ p\in P^k,\ \alpha = 0,1,\ldots,T-t_p,$$

$$(4)\ f(p^\alpha)c_{p^\alpha}^{w,v,\sigma} = 0, \quad \forall k\in K,\ p\in P^k,\ \alpha = 0,1,\ldots,T-t_p.$$

These optimality conditions are useful in several respects. Condition (1) states that the dual variable $w_{ij}(t)$ of arc (i, j) on time period t is zero if the optimal dynamic solution $f(p^\alpha)$ does not use all of the capacity $u_{ij}(t)$ of the arc in time point t. Also, condition (2) states that the dual variable $v_i^k(t)$ on time period t is zero if the optimal dynamic solution $f(p^\alpha)$ for commodity $k\in K$ does not use all of the capacity $u_i^k(t)$. For a given dynamic path $p\in P^k$ with departure time α in the form of

$$p^\alpha : (s_k^+,\alpha) = (i_1,t_1),(i_2,t_2),\ldots,(i_r,t_r) = (s_k^-,\beta),$$

the reduced cost $c_{p^\alpha}^{w,v,\sigma}$ is just the cost of that dynamic path with respect to the modified costs $w_{i_r,i_{r+1}}(t_r)+c^k_{i_r,i_{r+1}}(t_r)+v^k_{i_r}(t_r)+c^k_{i_r}(t_r)$ minus the dual value σ^k. So, condition

(3) states that the modified dynamic path cost

$$\sum_{(i_r,i_{r+1})\in p^\alpha, i_r\neq i_{r+1}} \Big(w_{i_r,i_{r+1}}(t_r)+c^k_{i_r,i_{r+1}}(t_r)\Big)+\sum_{(i_r,i_{r+1})\in p^\alpha, i_r= i_{r+1}} \Big(v^k_{i_r}(t_r)+c^k_{i_r}(t_r)\Big),$$

for each dynamic path p^α of commodity k must be at least as large as the dual value σ^k. The condition (4) implies that reduced cost $c^{w,v,\sigma}_{p^\alpha}$ must be zero for any dynamic path p^α with $f(p^\alpha)>0$. Therefore, conditions (3) and (4) imply the following lemma.

Lemma 4.1 (Dynamic shortest path optimality conditions) *In the optimal solution, every dynamic path p^α from NTP (s^+_k,α) to NTP (s^-_k,β) with $f(p^\alpha)>0$ must be a shortest dynamic path with respect to the modified costs and the dual value σ^k is the shortest dynamic path distance with respect to the modified costs.*

Proof Suppose that an optimal solution of the DPF problem is given. This optimal solution must satisfy the complementary slackness conditions (1–4). From condition (3) and by using of the definition of the reduced cost $c^{w,v,\sigma}_{p^\alpha}$,

$$\sum_{(i_r,i_{r+1})\in p^\alpha, i_r\neq i_{r+1}} \Big(w_{i_r,i_{r+1}}(t_r)+c^k_{i_r,i_{r+1}}(t_r)\Big)+\sum_{(i_r,i_{r+1})\in p^\alpha, i_r= i_{r+1}} \Big(v^k_{i_r}(t_r)+c^k_{i_r}(t_r)\Big)\geq \sigma^k.$$

Thus, σ^k is a lower bound on the length of any dynamic path p^α from NTP (s^+_k,α) to NTP (s^-_k,β) with respect to the modified costs and therefore is a lower bound on the dynamic shortest distance between these NTPs. On the other hand, from condition (4), every dynamic path p^α from NTP (s^+_k,α) to NTP (s^-_k,β) in the form of

$$p^\alpha:(s^+_k,\alpha)=(i_1,t_1),(i_2,t_2),\ldots,(i_r,t_r)=(s^-_k,\beta),$$

with $f(p^\alpha)>0$ implies that $c^{w,v,\sigma}_{p^\alpha}=0$, i.e.,

$$\sum_{(i_r,i_{r+1})\in p^\alpha, i_r\neq i_{r+1}} \Big(w_{i_r,i_{r+1}}(t_r)+c^k_{i_r,i_{r+1}}(t_r)\Big)+\sum_{(i_r,i_{r+1})\in p^\alpha, i_r= i_{r+1}} \Big(v^k_{i_r}(t_r)+c^k_{i_r}(t_r)\Big)=\sigma^k.$$

Therefore, σ^k is the dynamic shortest path length p^α from NTP (s^+_k,α) to NTP (s^-_k,β) with respect to the modified costs

$$w_{i_r,i_{r+1}}(t_r)+c^k_{i_r,i_{r+1}}(t_r)+v^k_{i_r}(t_r)+c^k_{i_r}(t_r).$$

□

In the next section, we propose a method based on the decomposition principle for solving the DPF problem by using of Lemma 4.1 and the revised simplex method.

4.1.2 Decomposition principle procedure

The revised simplex method, at every step, maintains a basic matrix. Then, using the fact that the reduced cost of every variable in the basis is zero, we can compute the dual variables $w_{ij}(t)$, σ^k and $v_i^k(t)$. In other words, if the dynamic path p^α from NTP (s_k^+, α) to NTP (s_k^-, β) for commodity k to form

$$p^\alpha : (s_k^+, \alpha) = (i_1, t_1), (i_2, t_2), \ldots, (i_r, t_r) = (s_k^-, \beta),$$

is one of the basic variables, then $c_{p^\alpha}^{w,v,\sigma} = o$. Thus, by using of matrix computations, the dual variables $w_{ij}(t)$, σ^k and $v_i^k(t)$ can be calculated via the following equations:

$$\begin{cases} \displaystyle\sum_{(i_r, i_{r+1}) \in p^\alpha, i_r \neq i_{r+1}} \left(w_{i_r, i_{r+1}}(t_r) + c_{i_r, i_{r+1}}^k(t_r) \right) \\ + \displaystyle\sum_{(i_r, i_{r+1}) \in p^\alpha, i_r + i_{r+1}} \left(c_{i_r}^k(t_r) + v_{i_r}^k(t_r) \right) = \sigma^k, \\ \text{for every dynamic path } p^\alpha \text{ in the basis.} \end{cases}$$

From revised simplex method, for a some basis, it always satisfies the complementary slackness conditions (1), (2), and (4). Therefore, it is optimal if satisfies condition (3). The question that arises is how this condition can be checked? In other words, how can we check the following condition:

$$\begin{cases} c_{p^\alpha}^{w,v,\sigma} = \displaystyle\sum_{(i_r, i_{r+1}) \in p^\alpha, i_r \neq i_{r+1}} \left(w_{i_r, i_{r+1}}(t_r) + c_{i_r, i_{r+1}}^k(t_r) \right) \\ + \displaystyle\sum_{(i_r, i_{r+1}) \in p^\alpha, i_r + i_{r+1}} \left(c_{i_r}^k(t_r) + v_{i_r}^k(t_r) \right) - \sigma^k \geq 0, \\ \forall k \in K,\ p \in P^k \text{ and departure times } \alpha = 0, 1, \ldots, T - t_p. \end{cases}$$

To check this condition, we can solve the following subproblems (i.e., a dynamic shortest path problem for each commodity k):

$$\begin{aligned} Z_k = \min & \sum_{(i_r, i_{r+1}) \in p^\alpha, i_r \neq i_{r+1}} \left(w_{i_r, i_{r+1}}(t_r) + c_{i_r, i_{r+1}}^k(t_r) \right) \\ & + \sum_{(i_r, i_{r+1}) \in p^\alpha, i_r + i_{r+1}} \left(c_{i_r}^k(t_r) + v_{i_r}^k(t_r) \right) \\ s.t. & \\ & k \in K,\ p \in P^k \quad \text{and departure times } \alpha = 0, 1, \ldots, T - t_p. \end{aligned} \tag{4.6}$$

In other words, to check the complementary slackness condition (3), we solve a dynamic shortest path problem for each commodity k. In fact, subproblems (4.6) indicate that if for all commodity k, the length of the dynamic shortest path is at least as large as dual value σ^k (i.e., $Z_k \geq \sigma^k$, $\forall k \in K$), the complementary slackness condition (3) is satisfied. In this case, the current basis is optimal. Otherwise, if for

some commodity k and dynamic shortest path F with departure time α', $Z_k < \sigma^k$, i.e.,

$$\sum_{(i_r,i_{r+1})\in F^{\alpha'},i_r\neq i_{r+1}} \left(w_{i_r,i_{r+1}}(t_r) + c^k_{i_r,i_{r+1}}(t_r)\right) + \sum_{(i_r,i_{r+1})\in F^{\alpha'},i_r+i_{r+1}} \left(c^k_{i_r}(t_r) + v^k_{i_r}(t_r)\right) - \sigma^k < 0,$$

then, $c^{w,v,\sigma}_{F^{\alpha'}} < 0$. Hence, the dynamic path F with departure time α' is eligible to enter the basis. The simplex method introduces dynamic path F with departure time α' into basis and determines one dynamic path to remove from the basis (by using of the minimum ratio test). After pivoting, a new set of dual variables $w_{ij}(t)$, $v^k_i(t)$, and σ^k (a new modified dynamic shortest path distance for each commodity k) are determined. Then, we transfer these dual variables into subproblems (4.6). We solve the subproblem corresponding to each commodity (by solving a dynamic shortest path problem) and see whether any dynamic path has a shorter length than σ^k. If is not such dynamic path, the optimal solution is reached. Otherwise, we introduce this path into basis and continue the revised simplex method.

5 Computational experiments

In this section, in order to demonstrate the effectiveness of the DPF model and the solution technique, computational results are presented on different classes of randomly generated instances. To achieve these objectives, we considered 12 classes of instances. For each class, 100 instances were generated (i.e., a total of 1200 instances) according to the methodology proposed by Erdös and Rényi [4], where it was assumed that there was a link from node i to node j with probability $p = 0.5$. Therefore, the number of arcs in a graph with n nodes is a random variable with expected value $\binom{n}{2} \times p$. Depending on the class of generated instances, the time horizon and the number of commodities are chosen from sets $\{20, 60, 180, 520\}$ and $\{4, 8, 16, 32\}$, respectively. The classes of generated instances are summarized in Table 1.

For each class instance, we simply decided to test our solution process on nominal data. Nominal data are generated randomly using the random distributions specified in Table 2. In the set of experiments, the amount of demand of each commodity $k \in K$ is randomly generated between 50 and 150. For each arc $(i, j) \in A$, the linear cost and bundle capacity (increasing or decreasing) functions with random coefficients are defined. Similarly, for each node $i \in N$, the linear cost and storage capacity (increasing or decreasing) functions with random coefficients are considered.

All the test instances are carried out on a Core i5-4670 and 3.40 GHz computer with 8.00 GB RAM. The proposed model and algorithm are coded in GAMS 24.9.1 and CPLEX (version 12.3) is used as an optimization solver for solving the LP model [5]. It is worth noting that the default setting is used in our runs.

Table 1 Class of instances

Class	n (number of nodes)	$\|K\|$	T
1	200	4	20
2	200	8	60
3	200	16	180
4	200	32	540
5	500	4	20
6	500	8	60
7	500	16	180
8	500	32	540
9	1000	4	20
10	1000	8	60
11	1000	16	180
12	1000	32	540

Table 2 The random parameters used in this work

Parameters	Corresponding random values
R_k	Unif(50, 150)
$c_{ij}^k(t)$	$H_{ij}^k + Z_{ij}^k t$
H_{ij}^k	Unif(0, 50)
Z_{ij}^k	Unif(0, 50)
$c_i^k(t)$	$h_i^k + z_i^k t$
h_i^k	Unif(0, 20)
z_i^k	Unif(0, 20)
$u_{ij}(t)$	$W_{ij} + S_{ij} t$
W_{ij} and S_{ij}	Unif(0.1, 0.5) $\times \max_{k \in K}(R_k)$
$u_i^k(t)$	$w_i^k + s_i^k t$
w_i^k and s_i^k	Unif(0, 0.5) $\times R_k$

In order to show the importance of the DPF model and the solution technique, two scenarios are considered in our computational results. In the first scenario, the arc-flow based formulation DDMF is directly implemented on time-expanded network with LP solver CPLEX ("TEN" in Table 3). In the second scenario, the path-flow formulation DPF is implemented by proposed algorithm ("Proposed algorithm" in Table 3). In Table 3, the average computational times in seconds (on 1200 instances) are reported. It is worth noting that we imposed a time limit of 200 s on the computation times. In this table, columns 2 and 3 refer to the average computational times for proposed algorithm and the time expanded network with LP solver CPLEX, respectively. By imposing the time limit of 200 s, there are some instances in some classes had not been solved up to optimality after 200 s of the CPU time. For these classes, we report the average

Table 3 Computational results

Class	CPU times (in secs.)		Timeout (%)	
	Proposed algorithm	TEN	Proposed algorithm	TEN
1	2.0325	0.4018	0	0
2	2.4009	1.0009	0	0
3	3.0931	15.1610	4	28
4	26.2060	86.3118	21	55
5	3.4596	63.4306	0	18
6	12.5208	98.0026	8	23
7	96.6182	186.0018	12	78
8	100.4226	NI*	39	100
9	64.9602	113.1402	10	46
10	79.4205	194.0022	26	91
11	158.7401	NI	49	100
12	NI	NI	100	100

**NI* None of the instances of this class was solved within the time limit by the corresponding method

computational times just for the instances that were solved up to optimality within the time limit of 200 s. Also, columns 4 and 5 refer to the percentage of instances for each class that passed the time limit of 200 s ("Timeout (%)" in Table 3) for the proposed algorithm and the time-expanded network with LP solver CPLEX, respectively. In some rows of Table 3, phrase "NI" illustrates that none of the instances of this class were solved within the time limit of 200 seconds.

From Table 3, we observe that almost in all instances except for small-scale instances in classes 1 and 2, the computational times of our proposed algorithm are smaller than that of the LP solver CPLEX for the time-expanded network. On the other hand, performance of the DDMF problem on the time-expanded network with LP solver CPLEX is more effected by the size of instances and especially by the value of the time horizon T (time horizon equal to 540). According to columns 3 and 5 of Table 3, none of the instances in classes 8, 11, and 12 were solved within the time limit by TEN technique. From columns 2 and 4 of Table 3, only the instances in Class 12 was not solved within the time limit by proposed algorithm (dense networks with 1000 nodes and time horizon equal to 540). In other words, instances with $T = 540$ and $n = 1000$ are the hardest instances for proposed algorithm. In particular, from two last columns of Table 3, the average of percentages of unsolved instances within the time limit by proposed algorithm and TEN technique are roughly 22.42 and 53.25%, respectively.

6 Conclusions

In this research, we have provided an LP formulation based on dynamic paths for the dynamic minimum cost multicommodity network flow problem. In this problem,

we assumed that the storage at intermediate nodes was allowed and cost and bundle capacity defined on arcs and capacity storage on nodes were time-varying. According to the special structure of the proposed model, we solved it based on technique of decomposition principle. To assess the performance of proposed model and the solution technique, computational results have presented on different classes of randomly generated instances. The computational times of the proposed approach and the time-expanded network with LP solver CPLEX have been compared. We observe that the proposed method, in general, presents good results compared with the time expanded network with LP solver CPLEX. Also, we observe that the performance of the DDMF problem on the time expanded-network with LP solver CPLEX compared with our proposed method is more depended to the size of the time horizon T.

Acknowledgements This research is jointly supported by grants from the Institute for Advanced Studies in Basic Sciences (IASBS) and the Ministry of Science, Research and Technology of the Islamic Republic of Iran.

References

1. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows. Prentice Hall, Englewood Cliffs (1993)
2. Aronson, J.E.: A survey of dynamic network flows. Ann. Oper. Res. **20**(1), 1–66 (1989)
3. Bazaraa, Mokhtar S., Jarvis, John J., Sherali, Hanif D.: Linear Programming and Network Flows. Wiley, New York (2011)
4. Bollobás, B.: Random Graphs (No. 73). Cambridge University Press, Cambridge (2001)
5. Brooke, A., Kendric, D., Meeraus, A., Raman, R.: Release 2.50 GAMS A User Guide. GAMS Development Corp., Washington, DC (1998)
6. Calvete, H.I., del-Pozo, L., Iranzo, J.A.: The energy-constrained quickest path problem. Optim. Lett. **11**(7), 1319–1339 (2017)
7. Fathabadi, H.S., Khodayifar, S., Raayatpanah, M.A.: Minimum flow problem on network flows with time-varying bounds. Appl. Math. Model. **36**(9), 4414–4421 (2012)
8. Fleischer, L., Skutella, M.: Quickest flows over time. SIAM J. Comput. **36**(6), 1600–1630 (2007)
9. Fleischer, L., Skutella, M.: Minimum cost flows over time without intermediate storage. In: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 66–75). Society for Industrial and Applied Mathematics (2003)
10. Fleischer, L., Skutella, M.: The quickest multicommodity flow problem. In: International Conference on Integer Programming and Combinatorial Optimization (pp. 36–53). Springer, Berlin, Heidelberg (2002)
11. Floudas, C.A., Pardalos, P.M. (eds.): Encyclopedia of Optimization, vol. 1. Springer, Berlin (2001)
12. Ford Jr., L.R., Fulkerson, D.R.: Constructing maximal dynamic flows from static flows. Oper. Res. **6**(3), 419–433 (1958)
13. Hall, A., Hippler, S., Skutella, M.: Multicommodity flows over time: efficient algorithms and complexity. Theor. Comput. Sci. **379**(3), 387–404 (2007)
14. Hashemi, S.M., Mokarami, S., Nasrabadi, E.: Dynamic shortest path problems with time-varying costs. Optim. Lett. **4**(1), 147–156 (2010)
15. Hoppe, B.: Efficient Dynamic Network Flow Algorithms. Cornell University, New York (1995)
16. Khodayifar, S., Raayatpanah, M.A., Pardalos, P.M.: A polynomial time algorithm for the minimum flow problem in time-varying networks. Ann. Oper. Res **272**(1–2), 29–39 (2019)
17. Klinz, B., Woeginger, G.J.: Minimum-cost dynamic flows: the series-parallel case. Netw. Int. J. **43**(3), 153–162 (2004)
18. Koch, R., Nasrabadi, E.: Flows over time in time-varying networks: optimality conditions and strong duality. Eur. J. Oper. Res. **237**(2), 580–589 (2014)
19. Lozovanu, D., Fonoberova, M.: Optimal dynamic multicommodity flows in networks. Electron. Notes Discrete Math. **25**, 93–100 (2006)

20. Powell, W.B., Jaillet, P., Odoni, A.: Stochastic and dynamic networks and routing. Handb. Oper. Res. Manag. Sci. **8**, 141–295 (1995)
21. Skutella, M.: An introduction to network flows over time. In: Research Trends in Combinatorial Optimization (pp. 451–482). Springer, Berlin, Heidelberg (2009)
22. Tardos, E.: A strongly polynomial algorithm to solve combinatorial linear programs. Oper. Res. **34**(2), 250–256 (1986)