The Nepali Mathematical Sciences Report, Vol. 39, No. 1, 2022: 1-13
DOI: https://doi.org/10.3126/nmsr.v39i1.46912

LEXICOGRAPHIC MAXIMUM FLOW ALLOWING INTERMEDIATE STORAGE

MOHAN CHANDRA ADHIKARI[1], URMILA PYAKUREL[2]

[1,2]*Central Department of Mathematics, Tribhuvan University, Kathmandu, Nepal*
Emails: [1]*mohan80700@gmail.com and* [2]*urmilapyakurel@gmail.com*

Abstract: In a capacitated network, an optimum solution of the maximum flow problem is to send as much flow as possible from the source node to the sink node as efficiently as possible by satisfying the capacity and conservation constraints. But, because of the limited capacity on the arcs, total amount of flow outgoing from the source may not reach to the sink. If the excess amount of flow can be stored at the intermediate nodes, total amount of flow outgoing from the source can be increased significantly. Similarly, different destinations have their own importance with respect to some circumstances. Motivated with these scenarios, we introduce the lexicographic maximum flow problems with intermediate storage in static and dynamic networks by assigning the priority order to the nodes. We extend this notion to arc reversals approach, a flow maximization technique, which is widely accepted in evacuation planning as it increases the outbound arc capacities by using the arc capacities on the opposite direction as well. Travel times along the anti-parallel arcs is considered to be unequal and we take into account the travel time of the reversed arcs to be equal to the travel time of the non-reversed arc towards which the arc is reversed. We present polynomial time algorithms for the solution of these problems.

Keywords: Flows in network, prioritization of nodes, lexicographic maximum flow, intermediate storage, asymmetric travel time.

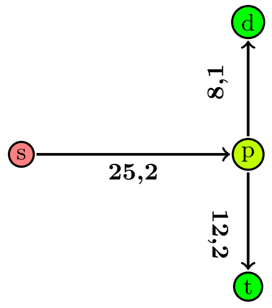
AMS (MOS) Subject Classification. Primary: 90B10, 90C27, 68Q25; Secondary: 90B06, 90B20.

1. Introduction

Motivation. Network flow theory is applicable for the solution of multiple real life problems. Among the different applications of network flow theory, evacuation planning is one. For the proper management of chaotic movement of people in the unsafe location, an efficient evacuation planning is required. For this purpose, the evacuation scenario is modeled mathematically as a dynamic transportation network, where unsafe location is a source and the safe locations are sinks. The different intermediate locations are the intermediate nodes and the links connecting to two distinct locations are arcs. The group of evacuees is considered as flow, that requires a certain cost (time) to travel along the arcs. Due to network topology, all evacuees can not be taken to the sinks. It is wise to place them at the intermediate locations, where basic requirements are available. The survey article of Dhamala et al. [4] and the citations therein provide an overview of different approaches for the solution of evacuation planning problem. For the applications of network flow theory, we refer the readers to Ahuja et al. [2].

Received: April 20, 2022 Accepted: June 20, 2022 Published Online: June 30, 2022
Corresponding author: Urmila Pyakurel .

. Let us consider an example with initial location s, intermediate location p and final locations d and t as in figure alongside. The numbers along the arcs represent capacity and travel time (we can send 25 units flow along the arc (s,p) in 2 units time. Suppose that we are given total time period 5 units. Along the path $s-p-t$ we can send 12 units flow twice, along the path $s-p-d$ we can send 8 units flow thrice. Hence, we can send 48 units flow to the sinks. However, if we are allowed to store flow at the intermediate location p, we can store 5 units twice, 17 units once and 25 units once from s to p. Here, we can extract 100 units flow from s among which 48 units reaches to the sinks and 52 units is stored at the intermediate node p.



Literature Review. Ford and Fulkerson [7, 8] developed mathematical model and algorithm for the solution of maximum network flow problem. The algorithm determines the s-d paths and augments flow along such path by satisfying the feasibility and conservation constraints. According to the model, total flow extracted from the source reaches into the sink and this value is equal to the minimum cut capacity. When there are multiple sinks in a network, flow can be sent with respect to some order. Maximization of flow by specifying the order of sinks, is lexicographic maximum flow. With the given priority ordering to the sources and sinks, Minieka [16] introduced the concept of lexicographic maximum flow. Later on, Hoppe and Tardos [11, 12] extended the notion of lexicographic flows in a netwrok with temporal dimension. Similarly, Pyakurel and Dhamala [22] introduced lexicographic maximum static and dynamic flow problems with arc reversals technique. They have presented polynomial time algorithms for the solution of these problems. The lexicographic maximum flow problem at every point of time is studied in [13]. However, these flow models do not take into account the storage capacity of the intermediate nodes.

When incoming flow into the intermediate node exceeds the outgoing flow, the difference is excess flow. The excess amount of flow can be stored at the intermediate nodes. Pyakurel and Dempe [20] introduced mathematical models and algorithms for the solution of the maximum static and dynamic flow problems with intermediate storage in two terminal general network. Their models assume that intermediate nodes have storage capacity. Furthermore, they have investigated the earliest arrival flow problem with intermediate storage, as an application to evacuation planning [21]. For the solution of these problems, they have presented the polynomial time algorithms. A major contribution of their work is the utilization of the outgoing full arcs capacity from the source. As an incremental approach to the maximum flow problems, intermediate storage is studied in [1]. Khanal et al. [14] introduced multicommodity flow problems with intermediate storage in static and dynamic networks. For the solution of these problems they have contributed polynomial and pseudo-polynomial algorithms, respectively.

Arc reversals introduced by Kim and Shekhar [15], is a flow maximization technique, widely accepted in evacuation planning. According to the different evacuation models[3, 22, 23, 24], movement towards the danger zone is not allowed. As a result, road (arcs) towards the source remain unused. The empty arc capacities are used to increase the flow value. The solution for the evacuation planning problem with arc reversals in continuous time settings is the research contribution of Pyakurel et al. [23]. All these models for evacuation planning have considered the symmetric travel time along the arcs in opposite direction. But, the travel times may not be symmetric always. Recently, Nath et al. [17] introduced arc reversal approach with asymmetric travel time along the anti-parallel arcs and considered the time along the direction of the arc towards which it is reversed. With this consideration they have introduced the maximum dynamic and quickest flow problems and presented polynomial time algorithms for the solution. Similarly, dynamic multicommodity contraflow problem with asymmetric travel times on the arcs is studied in Gupta et al. [10]. Their major contribution is development of an algorithm that is based on time expanded network.

Research Gap. During the transshipment of flow, different locations have their own relative importance with respect to some circumstances. In a capacitated network, the solution of the lexicographic maximum flow problem is to push as much flow as possible from source to the sinks in priority order with respect to some circumstance and maximum flow value is determined by max-flow min-cut theorem [8]. But, when sum of the arcs capacity outgoing from the source is greater than the minimum cut capacity, existing flow models do not use the full arcs capacity. In this case, using full arcs capacity outgoing from the source results in storage of flow at the intermediate nodes provided they have storage capacity. There exists a number of models and algorithms for the solution of the maximum static and dynamic flow problems without intermediate storage. Solution of the lexicographic maximum flow problems allowing storage of flow at the intermediate nodes with respect to their relative importance is one of the major issue. Similarly, in the solution procedure of lexicographic maximum flow problem with arc reversals technique, existing flow models and algorithms are based on the assumption that the travel time along the anti-parallel arcs is symmetric. As this assumption may not remain always true, appropriate solution procedure to address the asymmetric travel time is required.

Our Contribution. This paper considers a single-source multiple-sink network in which sum of the arcs capacity outgoing from the source exceeds the minimum cut capacity. Furthermore, it is assumed that intermediate nodes have capacity similar to the arcs. During the transshipment of flow from one location to another, multiple locations have their own importance. Based on their relative importance, for the extraction of the maximum amount of flow from the source, we introduce the lexicographic maximum static flow (LMSF) and lexicographic maximum dynamic flow (LMDF) problems allowing storage of flow at the intermediate nodes. Initial introduction of the LMSF and LMDF is presented in [19]. The discrete time LMDF solution is extended to the continuous time settings. We extend the notion of LMDF problem with intermediate storage in arc reversals approach. Arc reversals is a flow maximization technique widely used in evacuation. Here, the unused arc capacities are used to increase the outbound arc capacities by reversing the direction of arcs towards sinks. We assume that the travel time along the anti parallel arcs is asymmetric and take into account the travel time of reversed arc to be equal to the travel time along the arc towards which it is reversed (orientation dependent travel time). For the solution of these problems we present polynomial time algorithms.

Structure of Paper. Rest of the paper is organized as follows. In Section 2, we develop some mathematical notations and flow models with intermediate storage. The LMSF and LMDF problems with intermediate storage and their solution procedure are presented in Section 3 and Section 4, respectively. Similarly, orientation dependent LMDF problem with intermediate storage allowing arc reversals and its solution procedure are presented in Section 5. The paper concludes in Section 6.

2. Preliminaries

Let $A \subseteq V \times V$ be the set of m arcs, where V is the set of n nodes, s be the source, D be the set of multiple sinks and $I = V \setminus \{s, D\}$ be the set of intermediate nodes. The functions $u : A \to [0, \infty)$ and $c : A \to [0, \infty)$ determine the maximum amount flow that can be send from node i to j along the arc $e = (i, j)$ and associated cost, respectively. For an arc $e = (i, j) \in A$, its reverse arc is $(j, i) \in A$. We assume that s and D have infinite capacity and node $i \in I$ has finite capacity bounded by the function $v : I \to [0, \infty)$. With these parameters we have a single-source multiple-sinks static network $N = (V, A, s, I, D, u, v, c)$. If we have to send total flow within a given time horizon T, then the network is dynamic with parameters $N = (V, A, s, I, D, u, v, \tau, T)$ where $\tau_e : A \to [0, \infty)$ is the travel time required along the arc. Here, the travel time means if a unit flow is sent from node i at time θ, then it reaches to the node j at time $\theta + \tau_e$ where $\theta \in \mathbf{T} = \{0, 1, 2, \ldots, T\}$ is the discrete time settings for the total time horizon T. In continuous time settings, $\mathbf{T} = [0, T]$. The set of arcs incoming to and outgoing from the node $i \in V$ are denoted by A_i^{in} and A_i^{out}, respectively. In general, no arcs enter to the source node and exit from the sinks. Hence, we consider $A_i^{\text{in}} = A_i^{\text{out}} = \emptyset$. However, $A_i^{\text{in}} \neq \emptyset$ and $A_i^{\text{out}} \neq \emptyset$ in arc reversals approach.

2.1. **Maximum Static Flow Model.** On the given single-source multiple-sinks static network $N = (V, A, s, I, D, u, v, c)$, the function $f : A \to [0, \infty)$ is a static flow function such that it satisfies the feasibility condition on arcs $f(e) \in [0, u(e)]$ and the conservation constraints at nodes

$$\sum_{e \in A_i^{\text{in}}} f(e) \geq \sum_{e \in A_i^{\text{out}}} f(e), \quad \forall i \in I \tag{2.1}$$

for all $e \in A$. The value of maximum static flow is the total amount of flow out of the source given by

$$V(f) = \sum_{e \in A_s^{out}} f(e) = \sum_{e \in A_D^{\text{in}}} f(e) + \sum_{i \in I,\ v(i)>0} f(i), \quad \forall i \in I \tag{2.2}$$

where $f(i) : I \to [0, \infty)$, $f(i) \in [0, v(i)]$ is the amount of flow stored at the intermediate nodes $i \in I$, within node capacity. The feasibility condition on the arcs shows that flow does not exceed the arc capacity. Similarly, according to the conservation constraints, inflow into an intermediate node may exceed its outflow.

2.2. **Residual network.** The residual network of the given network N, with respect to the static flow f is a network N_f that shares the same vertices. The set of arc A_f contains the forward arcs $(i, j) \in A$ and the backward arcs $(j, i) \in A$, for all $i, j \in V$. The unused arc capacity is the residual capacity which is defined as $u_f(i, j) = u(i, j) - f(i, j)$ for forward arcs and $u_f(j, i) = f(i, j)$ for backward arcs. Similarly, $c_f(i, j) = c(i, j)$ and $c_f(j, i) = -c(i, j)$.

2.3. **Path, cycle and flow decomposition.** A path P in a network is a sequence of arcs $e_1, e_2, \ldots, e_k \in E$ for $k \in \mathbb{N}$ with $e_1 = (i_1, i_2), e_2 = (i_2, i_3), \ldots, e_k = (i_k, i_{k+1})$ where $e_k \in E$ and $i_1, i_2, \ldots, i_{k+1} \in V$ and $i_x \neq i_y$ unless $x = y$. Similarly, the sequence is called a cycle C if $e_k = (i_k, i_1)$ for $i_1, i_2, \ldots i_k$ and $i_x \neq i_y$ unless $x = y$. Travel time along the path and cycle is defined as $\tau_P = \sum_{e \in P} \tau_e$ and $\tau_C = \sum_{e \in C} \tau_e$. Let $\mathcal{P}$ and $\mathcal{C}$ be the set of simple paths and cycles and $P \in \mathcal{P} \cup \mathcal{C}$. A static flow f along the arc $e \in E$ can be decomposed into a paths and cycles $f_p \geq 0$ such that $f_e = \sum_{p \in P \cup C : e \in P} f_p$ for each $e \in E$.

2.4. **Maximum Dynamic Flow Model.** On the given single-source multiple-sink dynamic network $N = (V, A, s, I, D, u, v, \tau, T)$, the function $g : A \times \mathbf{T} \to [0, \infty)$ is a dynamic flow function, such that it satisfies the feasibility condition $g(e, \theta) \in [0, u(e, \theta)]$, $\forall \theta \in \mathbf{T}$ and the conservation constraints

$$\sum_{\sigma=\tau_e}^{\theta} \sum_{e \in A_i^{\text{in}}} g(e, \sigma - \tau_e) \geq \sum_{\sigma=0}^{\theta} \sum_{e \in A_i^{\text{out}}} g(e, \sigma), \quad \forall i \in I, \theta \in \mathbf{T} \tag{2.3}$$

for all $e \in A$. The value of maximum dynamic flow is the total amount of flow out of the source given by

$$V(g) = \sum_{\sigma=0}^{T} \sum_{e \in A_s^{\text{out}}} g(e, \sigma) = \sum_{\sigma=\tau_e}^{T} \sum_{e \in A_D^{\text{in}}} g(e, \sigma - \tau_e) + \sum_{\sigma=0}^{T} \sum_{i \in I, v(i)>0} g(i, \sigma) \tag{2.4}$$

where $g(i) : I \times \mathbf{T} \to [0, \infty)$ is the amount of flow stored at the intermediate nodes $i \in I$ at each time $\theta \in \mathbf{T}$ and $g(i, \theta) \in [0, v_i(\theta)]$.

2.5. **Temporally repeated flow.** For a given time period T, a static flow f can be decomposed along a set of paths $\mathcal{P}$ such that flow along the path P is f_P. Flow is sent along the path P at a constant rate f_P for $T + 1 - \tau_P$ times from time 0 to max $\{T - \tau_P, 0\}$ such that for each $e \in A$ and $\theta \in \mathbf{T}$, dynamic flow g defined by $g_e(\theta) = \sum_{p \in P_e(\theta)} f_p$

3. Lexicographic Maximum Static Flow

Priority is an act of providing the relative importance with respect to some circumstances. There are different approaches for setting the priority order of the nodes: reliability, storage capacity, distance. Authors in [9] used hesitant fuzzy hybrid average aggregation operator to set the priority order of nodes. In this work, our aim is to maximize the flow out of the source by pushing away towards the sinks in the specific order as far as practicable. Hence, we fix the priority ordering for all $i \in V \backslash \{s\}$ in given network $N = (V, A, u, v, c, s, I, D)$ as in [20] on the basis of distance from the source. At first, we set the priority order for the sinks and then for the intermediate nodes. For this task, we determine the minimum distance $d(s, i), i \in D$ for each sink from the source by considering cost as distance and assign first priority to the sink which is farthest. Similarly, second priority is assigned to the sink which is in the second farthest distance and so on. After all the sinks are prioritized, we assign priority order to the intermediate nodes $i \in I, v(i) > 0$ similar to the sinks.

Algorithm 1: Three Steps Priority Ordering [19]

```
Input  : Network $N = (V, A, u, v, c, s, I, D)$
Output: Nodes priority oreder
    (1) Order for sinks
        • For each sink $d \in D$, determine the minimum distance $d(s, d)$.
        • If $d(s, d_1) < d(s, d_2) < \cdots < d(s, d_r)$, set priority ordering as $(d_r, d_{r-1}, \ldots, d_2, d_1)$, $r \in \mathbb{Z}^+$.
        • If $d(s, i_p) = d(s, i_q)$ for some $p \neq q,$, $p, q \in \mathbb{Z}^+$ set priority order arbitrarily.
    (2) Order for intermediate nodes
        • For each intermediate node $i \in I$ with $v(i) > 0$, determine the minimum distance $d(s, i)$.
        • If $d(s, i_1) < d(s, i_2) < \cdots < d(s, i_l)$, set priority ordering as $(i_l, i_{l-1}, \ldots, i_2, i_1)$, $l \in \mathbb{Z}^+$.
        • If $d(s, i_p) = d(s, i_q)$ for some $p \neq q$, set priority order arbitrarily.
    (3) Final order
        • First, follow the order of Step (1) and then Step (2).
```

Lemma 3.1. *Algorithm 1 computes priority ordering for the nodes in polynomial time complexity.*

Proof. Finding the minimum distance for each node is feasible since the cost associated along the arcs $c(i, j) \geq 0$ and can be obtained in $O(n^2)$ time Using Dijkstra algorithm [5]. There are n nodes for priority ordering which requires linear time. Thus, time complexity required for the priority ordering of the nodes is polynomial. □

Definition 3.2. If we have $D_1 \subseteq D_2 \subseteq D_3 \subseteq \cdots \subseteq D_r \subseteq D$, then the lexicographic maximum flow on the sinks with intermediate storage is the maximal flow that delivers $V(D_x)$ units into each of the subsets D_x, for $x = 1, 2, \ldots, r$ and $V(I_y)$ units flow that does not reach to the sinks will be stored in the intermediate nodes I_y for $y = 1, 2, 3, \ldots, l$ where $r + l < n$.

Problem 3.3. *Let $N = (V, A, u, v, s, c, I, D)$ be a static network. The solution of LMSF problem with intermediate storage on N is to find the maximum amount of feasible flow, with fixed priority ordering to the sinks and the excess amount of flow that does not reach to the sinks is to push as far as possible from the source and store at the intermediate nodes in priority order, within their capacities.*

This problem is solved in two steps. In the first step, flow is maximized at each sink lexicographically and in the second step, the excess amount of flow that does not reach to the sinks is stored at the intermediate nodes within their capacity. The flow maximization procedure is as follows: First, we assign the priority order of sinks and then intermediate nodes using Algorithm 1. Second, for each $i \in I$, with $v(i) > 0$, we create dummy node $i' \in I'$ with same priority order and capacity. The node capacity is defined as $v(i') = v(i) \geq \sum_{e \in A_i^{\text{in}}} u_e > 0$. The dummy nodes are not the sinks but they are considered as sinks.

With this consideration, conservation constraints at the intermediate nodes are satisfied. The cost and capacity along the dummy arcs are $c(i,i')=0$ and $u(i,i')=v(i)=v(i')$. With the new parameters, the network is transformed as $N'=(V',A',u',v',c',s,I,D')$ where $V'=V\cup I'$, $A'=A\cup\{(i,i')\}$, $v'(i)=v(i)$, $u'(i,j)=u(i,j)\cup\{u(i,i')\}$ and $D'=D\cup I'$. On the transformed network N', the lexicographic order for the sinks is $\{d_1\}\subset\{d_1,d_2,\}\subset\cdots\subset D\subset D\cup\{i'_1\}\subset D\cup\{i'_1,i'_2\}\subset\cdots\subset D\cup I'=D'$.

On N', we determine the LMSF solution with the algorithm in [16]. Let $V_{\max}(D')$ be the total amount of flow into the sinks in D'. Now, $V_{\max}(D')$ is equal to the sum of $V_{\max}(D)$ and $V_{\max}(I')$, and it is the LMSF solution without intermediate sttorage on the transformed network N'. For the LMSF solution with intermediate storage on the given network N, the dummy nodes $i'\in I'$ and arcs $(i,i')\in A'$ are removed from N' and $V_{\max}(I')$ is returned back to respective intermediate nodes $i\in I$, that gives $V_{\max}(I)$ which is the flow value at the intermediate nodes. Now, we present an algorithm for the solution of Problem 3.3.

```
Algorithm 2: LMSF with intermediate storage [19]
Input  : Network $N=(V,A,u,v,c,s,I,D)$
Output: LMSF with intermediate storage on $N$.
    (1) Assign priority orderings to the nodes using Algorithm 1.
    (2) Transform the network $N=(V,A,u,v,c,s,I,D)$ into $N'=(V',A',u',v',s,I,D')$.
    (3) Determine the LMSF at each sink of $D'$ with priority ordering of Step (1).
    (4) Transform the flow from $N'$ to $N$ by deleting dummy arcs and nodes.
```

Theorem 3.4. *An optimum solution to the lexicographic maximum static flow problem with intermediate storage is obtained in polynomial time.*

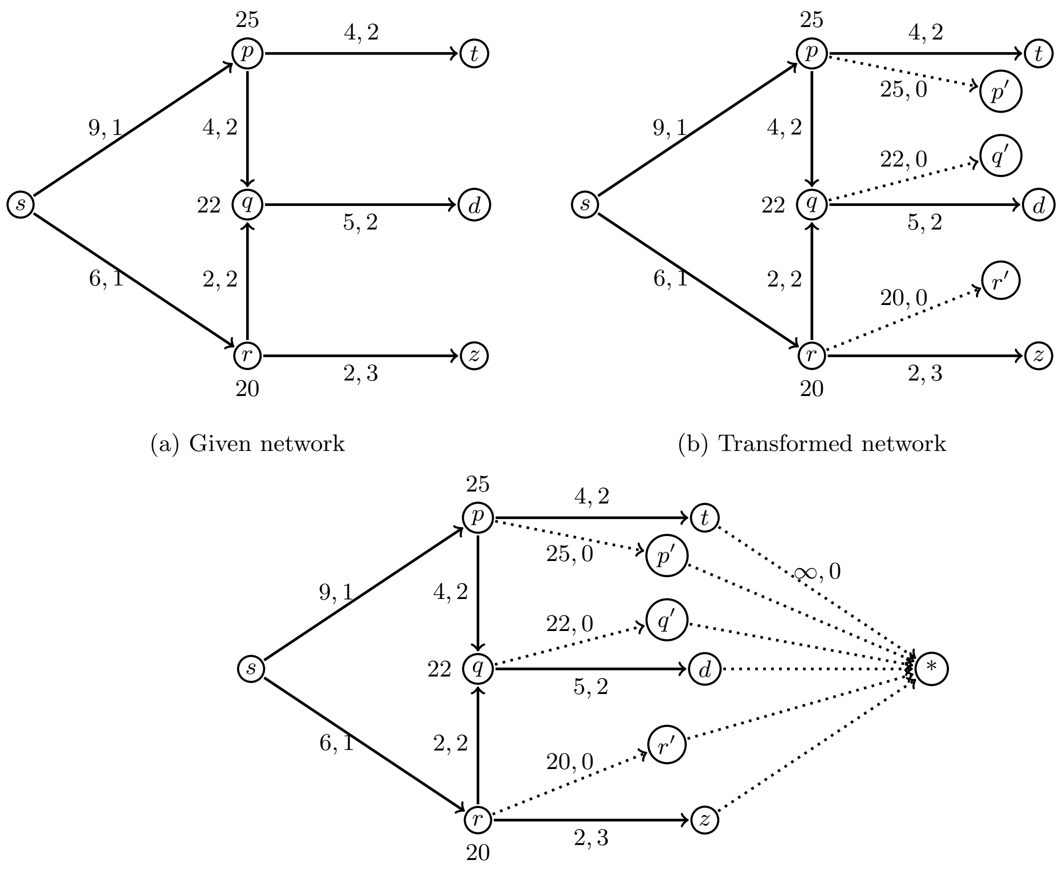
Proof. Feasibility of priority ordering follows from Algorithm 1. Storage capacity and priority orderings for the dummy sinks is similar to the intermediate nodes and capacities along dummy arcs are bounded by node capacities. Hence, transformation of network is feasible in Step (2). According to [16], Step (3) gives feasible maximum flow at each sink in D'. Finally, transformation of the LMSF solution from the transformed network $N'=(V',A',u',v',c,s,I',D')$ to the original network $N=(V,A,u,v,c,s,I,D)$ does not violate the feasibility constraints. Thus, Algorithm 2 is feasible. Now, we observe the optimality. Step (1) fixes the priority orderings of sinks as well as intermediate nodes in polynomial time complexity on the basis of minimum cost (distance). The transformed network in Step (2) is obtained in linear time and it is unique, since dummy nodes follow the existing priority order with same capacity. On the transformed network N', determine the LMSF at each sink in D'. At first, begin with the sink in the first priority and send as much flow as possible by satisfying flow conservation constraints. Now, consider this flow value as an initial flow, and obtain the corresponding residual network N_f. Augment flow along the $s-D'$ paths in N_f and gradually increase in each iteration. If there exist no flow augmenting paths for the current sink, the obtained flow is maximum [8]. Continue this procedure for each sink. Thus, we solve the maximum flow problem lexicographically and obtain LMSF on N' without intermediate storage. Finally, flow accumulated in the dummy sinks is returned back to the respective intermediate nodes as in [20] that gives the LMSF with intermediate storage.

The complexity of the algorithm is determined by Step (3) as Steps (1), (2) and (4) are obtained in polynomial time. Step (3) determines the time complexity of algorithm and with the solution procedure of [16] it is obtained in polynomial time. Thus, the time complexity of Algorithm 2 is polynomial. □

Example 3.5. We consider a s-D network in Figure 1 (a) where s is source, $D=\{t,d,z\}$ and $I=\{p,q,r\}$ are the set sinks and intermediate nodes. The set of arcs is $A=\{(s,p),(s,r),(p,q),(r,q),(p,t),(q,d),(r,z)\}$. We assume that s and D have sufficient capacities and nodes p, q and r have finite capacities 25 units, 22 units and 20 units, respectively. The numbers along the arcs denote capacity and cost (9 units flow can be sent along the arc (s,p) at one unit cost). Similarly, the number along the nodes denote their holding capacity (node p can hold at most 25 units flow). Capacity of the minimum cut is 11 and sum of the arcs capacity outgoing from the source is 15. Hence, there exists lexicographic solution with intermediate storage.

The minimum distance (cost) for the sinks $\{t, d, z\}$ from the source $\{s\}$ are 3, 5 and 4 units, respectively. Since $d(s,d) > d(s,z) > d(s,t)$, their priority order is assigned as (d, z, t). Similarly, intermediate nodes have priority order (q, r, p) or (q, p, r). Corresponding dummy sinks and arcs are shown in Figure 1 (b). The transformed network has the set of nodes $V' = V \cup I'$ where $I' = \{p', q', r'\}$, $A' = A \cup \{(p'p'), (q, q'), (r, r')\}$ and $D' = D \cup I'$ and s is the source. Dummy arcs have capacities as $u(p, p') = 25$, $u(q, q') = 22$ and $u(r, r') = 20$. The sinks in Figure 1 (b) are prioritized as (d, z, t, q', p', r'). We consider a super sink $*$ and connect it to each sink by an arc with infinite capacity and zero cost (time) as shown in Figure 1 (c) where PMSF is calculated with the solution procedure of [16].

The sinks d, z and t receive 5 units, 2 units and 4 units flow. Similarly, the dummy sinks r', p' and q' receive 2 units, 1 unit and 1 unit flow, respectively which is returned back to the intermediate nodes r, p and q for the LMSF solution with intermediate storage . Hence, maximum flow out of the source is 15 units out of which 11 units reaches to the sinks and 4 units is stored at intermediate nodes.



(a) Given network

(b) Transformed network

(c) Extended network

FIGURE 1. Lexicographic maximum static flow (Arc: capacity, cost; Node: capacity)

4. Lexicographic Maximum Dynamic Flow

Flows that require time to travel over the arc and arc capacities change with respect to time are dynamic flows. Solution of the maximum dynamic flow problem is to determine the maximum amount of flow into the sink for a given time period T. But, the solution of the LMDF problem with intermediate storage is to determine the maximum amount of feasible flow at each prioritized sink and to store the excess amount of flow at the prioritized intermediate nodes.

Problem 4.1. *Let $N = (V, A, u, v, s, I, D, \tau, T)$ be a given dynamic network. The solution of LMDF problem with intermediate storage on N is to find the maximum amount of feasible flow, with priority ordering to the sinks. The excess amount of flow that does not reach to the sinks is sent as far as possible from the source and stored at the intermediate nodes in priority order respecting their capacities within the given time horizon T.*

The LMDF problem without intermediate storage was introduced by Hoppe and Tardos [11, 12], where flow is conserved at each node. Pyakurel and Dempe [20] introduced the maximum dynamic flow problem with intermediate storage on two terminal general network. For the solution of these problems they used lexicographic properties. Here, we introduce the LMDF problem with intermediate storage in s-D network $N = (V, A, u, v, s, I, D, \tau, T)$. For the solution, we present a polynomial time algorithm based on [11, 12]. Initially, the solution is obtained on the transformed network $N' = (V', A', u', v', s, I, D', \tau, T)$ similar to Section 3, by considering the travel time $\tau(i, i') = 0$. The solution obtained on the transformed network N' is the LMDF solution without intermediate storage. Finally, for the LMDF solution with intermediate storage, solution from the transformed network N' is transformed into the original network N by removing the dummy arcs and nodes similar to [20]. Now, we present an algorithm for the solution of Problem 4.1.

Algorithm 3: LMDF with intermediate storage [19]

```
Input  : Network $N = (V, A, u, v, s, I, D, \tau, T)$
Output: LMDF with intermediate storage on $N$.
    (1) Assign priority orderings to the nodes using Algorithm 1.
    (2) Transform the network $N = (V, A, u, v, s, I, D, \tau, T)$ into $N' = (V', A', u', v', s, D', \tau, T)$.
    (3) Determine the LMDF at each sink of $D'$ with priority ordering of Step 1.
    (4) Transform the flow from $N'$ to $N$ by deleting dummy arcs and nodes..
```

Theorem 4.2. *An optimal solution to the lexicographic maximum dynamic flow problem with intermediate storage is obtained in polynomial time.*

Proof. Feasibility of the solution follows from Theorem 3.4 as $\tau(i, j) > 0$ and $\tau(i, i') = 0$. Now, we prove its optimality. Each $i \in I$, on N' contains dummy sinks so, flow conservation constraints at the intermediate nodes are satisfied, where N' is the transformed single-source multi-sink network with priority ordering on sinks. At each prioritized sink on N', we determine the maximum flow. First, take a sink that is in the first priority and determine the maximum amount of flow that can be sent at this sink. By taking this flow value as an initial flow determine the residual network N_f. On the residual network N_f, determine the maximum amount of flow that can be send to the sink of second priority and continue this procedure for other sinks. Let us assume that there are r many sinks. Now, repeated use of minimum cost circulation for r times gives the optimum LMDF solution without intermediate storage N' [11, 12]. Finally, flow collected to the dummy sinks is returned back to the corresponding intermediate nodes by removing the dummy nodes and arcs from the transformed network. According to [20], the obtained solution is equal to the LMDF solution with intermediate storage.

The time complexity required for Priority ordering of sinks and intermediate nodes in Step (1) is polynomial. Since transformation of network N into N' in Step (2) and transformation of solution from N' to

N (4) require linear time, time complexity of our algorithm depends upon Step (3). The time complexity required for the minimum cost circulation is $O((m \log n(m + n \log n)))$ [18] and the LMDF problem without intermediate storage is solved in $O(r \times (m \log n(m + n \log n)))$ time [11, 12]. Hence, the LMDF problem with intermediate storage is solved within the time complexity of $O(r \times (m \log n(m + n \log n)))$. □

Example 4.3. We consider Figure 1 (a), with cost replaced by time and total time horizon $T = 5$. The minimum distance to each node from s with respect to the travel time is $d(s,t) = 3$, $d(s,d) = 5$, $d(s,4) = 3$, $d(s,p) = 1$, $d(s,q) = 3$ $d(s,r) = 1$. Hence, the Priority order for the nodes is (d, z, t, q, p, r). By assumption, p', q' r' are dummy sinks and have priority order (q', p', r'). The sinks in D' are prioritized as (d, z, t, q', p', r'). First, determine the LMDF without intermediate storage as follows: Introduce a super source s^* and connect it to the source s with $u(s^*, s) = \infty$ and $\tau(s^*, s) = 0$. Similarly, for each $x \in D'$, join x to s^* with $u(x, s^*) = \infty$ and $\tau(x, s^*) = -(T + 1)$. The minimum cost cycles for d are $C_1 : s - p - q - d - s^* - s$ and $C_2 : s - r - q - d - s^* - s$. Feasible flow along C_1 is $f_1 = 4$. After sending 4 units flow along the path $s - p - q - d$ residual network is determined. In the residual network, the feasible flow along C_2 is $f_2 = 1$. Now, in the residual network, there does not exist minimum cost circulations for d so we delete this sink. This process is carried out for all sinks in their respective priority order. Here, sinks d, z and t receive 5 units, 4 units and 12 units flow and dummy sinks q', p', r' receive 13 units, 21 units and 20 units flow. Transformation of flow from the dummy sinks p', q', r' to the respective intermediate nodes p, q, r gives the LMDF solution with intermediate storage. The LMDF solution without and with intermediate storage for different time periods are tabulated in Table 1.

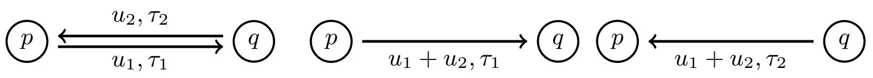
Table 1. LMDF without and with intermediate storage

	LMDF without intermediate storage						LMDF with intermediate storage					
Nodes	t=1	t=2	t=3	t=4	t=5	Total	t=1	t=2	t=3	t=4	t=5	Total
d	-	-	-	-	5	5	-	-	-	-	5	5
z	-	-	-	2	2	4	-	-	-	2	2	4
t	-	-	4	4	4	12	-	-	4	4	4	12
q	-	-	-	-	-	-	-	-	1	6	6	13
p	-	-	-	-	-	-	1	1	1	9	9	21
r	-	-	-	-	-	-	2	2	4	6	6	20
	Total flow out of the source					21	Total flow out of the source					75

Solution in continuous time settings. The solution of lexicographic maximum dynamic flow problem with intermediate storage obtained in the discrete time settings can be extended into the continuous time settings. A continuous dynamic flow is a function $\Psi : A \times [0, T] \to [0, \infty)$ that defines the flow rate continuously over the time. Flow values in continuous time settings are more accurate than the flow values obtained in discrete time settings, but are relatively harder. Let $g_e^r(\theta)$ be the rate of flow passing through an arc $e \in A$ at discrete time settings $\theta \in \{0, 1, 2, \ldots, T\}$. Now, one can set the continuous flow rate $\Psi_e^r(t)$ for $\theta \leq t < \theta + 1$ to reflect the equivalent discrete flow rate $g_e^r(\theta)$. According to Fleischer and Tardos [6], this transformation preserves optimality if we consider the constant arc capacities. In particular, at any interval $[\theta, \theta + k), k \in Z^+$ these two flow values are equivalent.

5. Orientation Dependent Lexicographic Maximum Dynamic Flow

Arc reversal is a flow maximization technique, widely used in emergency as it increases the flow value and reduces the time period. Authors in [3, 24] introduced maximum dynamic flow problem without intermediate storage and the solution with intermediate storage is presented in [20]. Theses problems are solved within the capacity of polynomial time complexity. The unused arc capacities, towards the source are used by reversing the direction of the arcs towards the sink as in [22, 20, 3, 24], where authors have considered the symmetric travel time along the arcs. Here, we assume that, the anti-parallel arcs have unequal travel time and in this work we consider the travel time along the direction of the arcs towards which it is reversed, similar to [17], as shown in Figure 2. The components along the arcs represent their capacity and required travel time. Figure 2 (a) is the given network. Figure 2 (b) arc (q, p) is reversed with same travel time. Figure 2 (c) and (d) represent the orientation dependent travel time when arc (q, p) and (p, q) are reversed respectively.



(a) Anti parallel arcs (b) Arc (q, p) is reversed (c) Arc (p, q) is reversed

FIGURE 2. Arc reversals with orientation based travel time

Problem 5.1. *Let $N = (V, A, u, v, s, I, D, \tau, T)$ be a given dynamic s-D network. For a prespecified time horizon T, the solution of orientation dependent LMDF problem with intermediate storage, is to determine the maximum amount of feasible flow that can be sent from source to the sinks and intermediate nodes in the priority ordering. The excess amount of flow that does not reach to the sinks is stored at the intermediate nodes within their capacities, if direction of the arcs can be reversed at zero time.*

For the solution of Problem 5.1, an auxiliary network $N_a = (V, A_a, u_a, v, s, I, D, \tau_a, T)$ is constructed corresponding to the given network $N = (V, A, u, v, s, I, D, \tau, T)$. On auxiliary network N_a, set of nodes and their capacity remain same but this case does not remain same for the set of arcs. Arc capacities are increased by using the capacities of the arcs in the both directions. Travel time along the arcs on N_a depends upon the direction of the arcs towards which it is reversed. The auxiliary network $N_a = (V, A_a, u_a, v, s, I, D, \tau_a, T)$ is transformed to $N'_a = (V', A'_a, u'_a, v, s, I, D', \tau'_a, T)$ similar to Section 3 and 4, where $\tau'_a(i, i') = 0$. On N'_a, determine the LMDF solution, which is independent of intermediate storage. The solution from N'_a is transformed to N_a by removing the dummy arcs and sinks, which is the LMDF solution with intermediate storage on N_a [20]. The LMDF solution on the auxiliary network N_a is equiivalent to the LMDF solution with arc reversals in the given network N [22]. We present an algorithm for the solution of Problem 5.1.

Algorithm 4: Orientation dependent LMDF with intermediate storage

Input : A dynamic network $N = (V, A, u, v, s, I, D, \tau, T)$

Output: Orientation dependent LMDF with intermediate storage

(1) Construct the auxiliary network $N_a = (V, A_a, u_a, v, s, D, \tau_a T)$, with redefined arcs capacity $u_a(i, j) = u(i, j) + u(j, i)$ where $u(i, j) = 0$ if $(i, j) \in A$ and travel times

$$\tau_a(i,j) = \begin{cases} \tau(i,j) & \text{if} \quad (j,i) \text{ is reversed along } (i,j) \text{ or } (j,i) \notin A. \\ \tau(j,i) & \text{if} \quad (i,j) \text{ is reversed along } (j,i) \text{ or } (i,j) \notin A. \end{cases}$$

(2) Set priority orderings on N_a using Algorithm 1.

(3) Transform the network N_a into $N'_a = (V', A'_a, u'_a, v', s, I, D', \tau, T)$ with dummy sinks.

(4) Determine the LMDF solution on N'_a without intermediate storage.

(5) Transform the solution from N'_a to N_a by removing the dummy nodes and arcs to obtain the LMDF solution with intermediate storage.

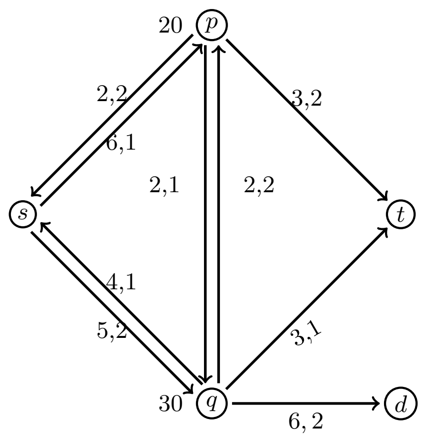
(6) Decompose the flow obtained in Step 4 into paths and cycles and remove the cycle flows.

(7) Reverse the arc $(j, i) \in A$ iff $g(i, j) > u(i, j)$ up to $g(i, j) - u(i, j)$ and $u(i, j) = 0$ for $(i, j) \notin A$

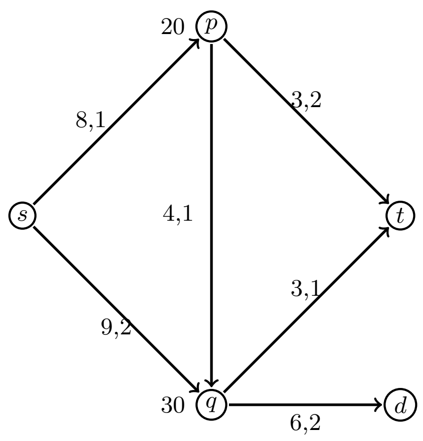
Theorem 5.2. *Algorithm 4 solves the orientation dependent lexicographic maximum dynamic flow problem with intermediate storage optimally in polynomial time.*

Proof. For the proof of the theorem, we observe the feasibility, optimality and complexity of the Algorithm 4. For the feasibility of Algorithm 4, it is enough to observe the feasibility of Step (7). Decomposition of flow along the cycles and paths in Step (6) ensures that flow is sent along one direction but not on both directions. Moreover, such flow does not exceed sum of the arc capacities. Hence, Step (7) is also feasible. For the optimal solution, we construct an auxiliary network N_a by reversing the arcs at zero time and once the arcs are reversed they remain reversed throughout the execution of the algorithm. Construction of auxiliary network requires linear time. The auxiliary network N_a is transformed to N_a', where we determine the LMDF solution similar to Section 4 using the procedure in [11, 12] which gives an optimal solution within the capacity of polynomial time complexity. Decomposition of flow in Step (6) requires $O(mn)$ time [2]. Flow from dummy nodes is returned back to the respective intermediate nodes at zero time. The LMDF solution on auxiliary network N_a is equivalent to the LMDF solution with arc reversals in the given network N [22]. Finally, the amount of flow stored at the intermediate nodes in auxiliary network N_a is equal to the amount of flow stored at the intermediate nodes in the original network N [20]. □

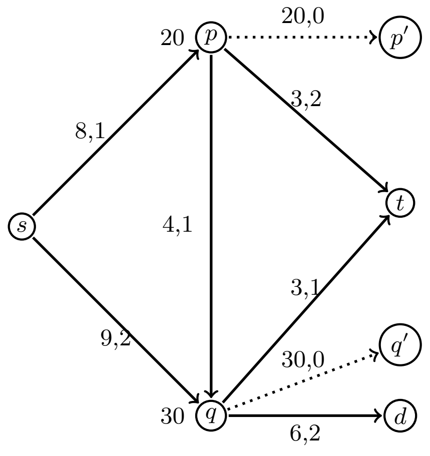
Example 5.3. Consider a flow network with source s and sinks $\{t, d\}$ in Figure 3 (a), where flow is allowed to move in either direction. The numbers along the arcs represent capacity and travel time and the number along the nodes represent their storage capacity. Figure 3 (b) is an auxiliary network N_a constructed according to Step 1 of Algorithm 4. The reversals of arc (p, q) does not increase flow value due to the smaller capacity on the arc (p, t) so arc (q, p) is reversed. Minimum time required to reach the nodes p, q, t, d is 1, 2, 3 and 4 units, respectively. Based on travel time, the nodes are prioritized as d, t, q, p. The LMDF solution with intermediate storage at different time period, without arc reversals and with orientation dependent arc reversals is shown in Table 2.



(a) Given network



(b) Auxiliary network



(c) Transformed network with dummy sinks

FIGURE 3. LMDF with travel time depending upon the orientation of non-reversed arc (Arc: capacity, travel time and node: capacity).

TABLE 2. LMDF without and with arc reversals

	LMDF without arc reversals					LMDF with arc reversals				
Nodes	$t = 1$	$t = 2$	$t = 3$	$t = 4$	Total	$t = 1$	$t = 2$	$t = 3$	$t = 4$	Total
t	-	-	5	5	10	-	-	6	6	12
d	-	-	-	-	5	-	-	-	6	6
q	-	-	5	7	12	-	4	10	13	27
p	1	1	4	6	12	1	1	4	8	14
	Total flow out of the source				39	Total flow out of the source				59

6. Conclusions

Based on some circumstances, different locations have relative importance and based on some circumstances they should be prioritized. Network flow models without intermediate storage are studied literature, where total amount of flow that can be sent into the sinks from the source is equal to the minimum cut capacity. But, if sum of the arcs capacity outgoing from the source is more than the minimum cut capacity, total amount of flow extracted from the source may not reach to the sinks and there remains excess amount flow at the intermediate nodes. If flow is allowed to store at the intermediate nodes within their capacity, flow value outgoing from the source can be increased significantly and this flow value is more than the minimum cut capacity. To hold the excess amount of flow at the intermediate nodes, we have introduced lexicographic maximum flow problems with intermediate storage in static and dynamic networks. This approach maximizes the flow value outgoing from the source significantly by holding the excess amount of flow at the intermediate nodes in priority order without violating the nodes capacity under the assumption that sum of the arcs capacity outgoing from the source exceeds the minimum cut capacity. The notion of lexicographic maximum flow is extended to the arc reversals technique where travel time along the anti-parallel arcs is assumed to be asymmetric. For the solution of these problems, polynomial time algorithms are presented.

Acknowledgment

Second author acknowledges to Alexander von Humboldt Foundation for *Digital cooperation Fellowship* (August 1, 2021 to January 31 2022) and *Remote Cooperation Abroad Fellowship* (March 1, 2022 to August 31, 2022) with host Prof. Dr. Stephan Dempe, TU Bergakademie Freiberg, Germany.

References

[1] M. C. Adhikari, *Incremental approach for maximum network flow solution*, M.Phil Thesis, Tribhuvan University Katathmandu, Nepal (2020).

[2] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, *Network Flows: Theory, Algorithm and Applications*, Prentice Hall, Upper Saddle River, New Jersey, USA, 1993.

[3] A. Arulselvan, *Network model for disaster management*, PhD Thesis, University of Florida, USA, 2009.

[4] T. N. Dhamala, U. Pyakurel, S. Dempe, A critical survey on the network optimization algorithms for evacuation planning problems, *International Journal of Operations Research*, **15**(3), 101-133, 2018.

[5] E. W. Dijkstra, A note on two problems in connection with graphs, *Numerische mathematik*, 1(3), 269-271, 1959.

[6] L. Fleischer, E. Tardos, Efficient continuous-time dynamic network flow algorithms, *Operations Research Letters*, **23** 71-80, 1998.

[7] F. R. Ford, D. R. Fulkerson, *Flows in Networks*, Princeton University Press, Princeton, New Jersey, USA, 1956.

[8] F. R. Ford, D. R. Fulkerson, Maximal flows through a network, *Canadian Journal of Mathematics,* 8, 399-404, 1956.

[9] E. Gerasimenko, V. Kureichi, E. Kuliev, Maximum dynamic flow model for hesitant fuzzy evacuation with intermediate storage at nodes, *In: Kahraman C., Cebi S., Cevik Onar S., Oztaysi B., Tolga A. C., Sari I. U. (eds) Intelligent and Fuzzy Techniques for Emerging Conditions and Digital Transformation*, INFUS 2021. Lecture Notes in Networks and Systems, vol 307, 2022, Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-85626-7_81

[10] S. P. Gupta, U. Pyakurel, T. N. Dhamala, Dynamic multicommodity contraflow problem with asymmetric transit times, *Journal of Applied Mathematics, Hindawi*, Volume 2022, Article ID 3697141, 8 pages, 2021. https://doi.org/10.1155/2022/3697141

[11] B. Hoppe, E. Tardos, Polynomial time algorithms for some evacuation problems, *Proc. of 5th Ann ACM-SIAM Symp on discrete algorithms,* Pages 433-441, 1994.

[12] B. Hoppe, E. Tardos, The quickest transshipment problem, *Mathematics of Operations Research,* 25, 36-62, 2000.

[13] N. Kamiyama, Lexicographically optimal earliest arrival flows, *Networks*, 75, 18-33, 2019.

[14] D. P. Khanal, U. Pyakurel, T. N. Dhamala, Maximum multicommodity flow with intermediate storage, *Mathematical Problems in Engineering, Hindawi*, Volume 2021, Article ID 5063207, 11 pages, 2021. https://doi.org/10.1155/2021/5063207

[15] S. Kim, S. Shekhar, Contraflow network configuration for evacuation planning: a summary of results, *In: proceedings of 13th Annual ACM International Workshop on Geographic Information System*, GIS 05, ACM, New York, NY, USA, pp 250-259, 2005.

[16] E. Minieka, Maximal, lexicographic, and dynamic network flows, *Operations Research*, 21, 517-527, 1973.

[17] H. N. Nath, U. Pyakurel, T. N. Dhamala, Network reconfiguration with orientation dependent travel times, *International Journal of Mathematics and Mathematical Sciences, Hondawi*, Article ID 6613622, 11 pages, 2021. https://doi.org/10.1155/2021/6613622

[18] J. B. Orlin, A faster strongly polynomial minimum cost flow algorithm, *Operations Research*, 21, 517-527, 1993.

[19] U. Pyakurel, M. C. Adhikari, Priority based flow improvement with intermediate storage, 2020. (cf. Optimization Online $http://www.optimization-online.org/DB_HTML/$2020/07/7891$.html$).

[20] U. Pyakurel, S. Dempe, Network flow with intermediate storage: models and algorithms. *SN Operations Research Forum*, 1 (4), 37, 2020. https://doi.org/10.1007/s43069-020-00033-0

[21] U. Pyakurel, S. Dempe, Universal maximum flow with intermediate storage for evacuation planning, *In: Kotsireas I.S., Nagurney A., Pardalos P. M., Tsokas A. (eds) Dynamics of Disasters. Springer Optimization and Its Applications*, vol 259. Springer, Cham, 2021. https://doi.org/10.1007/978-3-030-64973-9_14

[22] U. Pyakurel, T. N. Dhamala, Models and algorithms on contraflow evacuation planning network problems, *International Journal of Operations Research*, 12(2), 36-46, 2015.

[23] U. Pyakurel, T. N. Dhamala, S. Dempe, Efficient continuous contraflow algorithms for evacuation planning problems, *Annals of Operations Research (ANOR)*, 254, 335-364, 2017.

[24] S. Rebennack, A. Arulselvan, L. Elefteriadou, P. M. Pardalos, Complexity analysis for maximum flow problems with arc reversals, *Journal of Combinatorial Optimization*, 19, 200-225, 2010.