SN Operations Research Forum (2020) 1: 37
https://doi.org/10.1007/s43069-020-00033-0

ORIGINAL RESEARCH

Network Flow with Intermediate Storage: Models and Algorithms

Urmila Pyakurel[1] · **Stephan Dempe**[2]

Received: 31 March 2020 / Accepted: 2 October 2020 / Published online: 17 November 2020

Abstract

Various network flow models, such as a flow maximization, a time minimization, a cost minimization, or a combination of them, have already been investigated. In most of the cases, they are considered subject to the flow conservation constraints. Here, we investigate the network flow models with intermediate storage, i.e., the inflow may be greater than the outflow at intermediate nodes. We introduce a maximum static and a maximum dynamic flow problem where an intermediate storage is allowed. Then, polynomial time algorithms are presented to solve these problems in two terminal general networks. We also study the earliest arrival property of the maximum dynamic flow in two terminal series-parallel networks and present its efficient solution procedure with intermediate storage. Moreover, we introduce a dynamic contraflow model with intermediate storage and present a polynomial time algorithm to solve the maximum dynamic contraflow problem in two terminal networks. We also solve an earliest arrival contraflow problem with intermediate storage. Our investigation is focused to solve the evacuation planning problem where the intermediate storage is permitted.

Keywords Network flow · Intermediate storage · Algorithms · Contraflow · Evacuation planning

1 Introduction

The contributions in network flow models for last six decades can be found very interesting and important to solve many real-life problems. Ford and Fulkerson [12] introduced a network flow model to solve the *maximum dynamic flow problem*

✉ Urmila Pyakurel
urmilapyakurel@gmail.com

[1] Central Department of Mathematics, Tribhuvan University, P.O.Box 13143, Kathmandu, Nepal

[2] Fakultät für Mathematik und Informatik, TU Bergakademie Freiberg, Freiberg, Germany

(MDFP), where the flow value leaving a source and entering a sink is maximized within a given time horizon without intermediate storage. If the flow is maximized at every time point, then a solution to the *earliest arrival flow problem* (EAFP) is obtained [13]. Wilkinson [33] solved the EAFP in a two terminal network by computing a maximum flow–minimum cut on its time-expanded network. Minieka [20] also solved the EAFP presenting a pseudo-polynomial time algorithm that computes the successive shortest paths as in Ford and Fulkerson [12] on the residual network and shifts flow along the paths in temporally repeated way.

If the flow is maximized according to a given priority ordering of sources and sinks, then a *lexicographically maximum flow problem* arises. Minieka [20] introduced the *lex-max static flow problem* (LMSFP) and presented a polynomial time algorithm to solve it. Hoppe and Tardos [15] modeled the evacuation problems as flow problems in dynamic networks. They presented a polynomial time algorithm to solve the *lexicographic maximum dynamic flow problem* (LMDFP) in a network with any number of sources, see also in [16]. In a two terminal general network, they presented a polynomial time algorithm that approximates an earliest arrival flow within a factor of $(1+\epsilon)$ for any $\epsilon > 0$. The management of intersections of the streets is a serious concern while dealing with evacuation planning problems. Cova and Johnson [6] investigated this issue while the power is not available and hence the traffic lights are not functional. By presenting an optimal lane-based evacuation routing plan in a complex road network, they minimized the conflicts that take place at intersections.

The network flow models are applicable to solve many real-life problems, for example, evacuation planning, transportation planning, production management, and logistics [2]. We can find a number of new algorithmic approaches and applications to different kinds of disasters in the book series *Dynamic of Disasters*, edited by Kotsireas et al. [19]. To investigate the network flow models in evacuation planning problems, one considers a directed graph in which different numbers are assigned to the arcs as non-negative capacity, non-negative cost, or integer transit time. Also a capacity is assigned to each node. We search for an optimal network flow from the sources (risk nodes) to the sinks (safe nodes) which have infinite capacities. We assume that each intermediate node has a finite capacity. In any final solution of a network flow model, the total outflow from the sources should be equal to the total inflow into the sinks; therefore, storage at the intermediate nodes is not allowed. These models also have the drawbacks that they do not care the capacity of intermediate nodes and possible stay of a flow amount there.

The maximum flow sent to the sink is equal to the minimum cut capacity of a network [11]. If the leaving arcs from the sources have larger capacities than the minimum cut capacity of the network, the excess capacity is not used to save people. For example, we consider Fig. 1 to explain this situation more clearly. Suppose a state L has 20,000 people suffering from the terrorist attack and it needs to send them in the safe areas within a day. Suppose state Q offers the permanent place for these people but there is no direct connection from L to Q. Suppose P and R be transit places in which P has capacity 16,000 to hold people. The transportation from L to P is available for 15,000 people per day but from P to Q, the capacity allows only for 10,000 people per day.

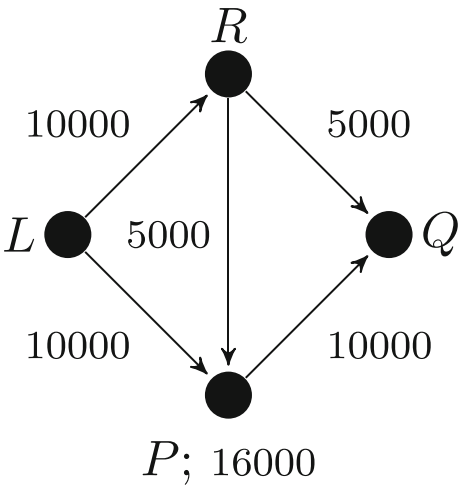

Fig. 1 Evacuation network

In this situation, if we apply the existing network flow models without intermediate storage, we are able to send only 15,000 people from L per day via paths $L-R-Q$ and $L-P-Q$ with flows 5000 and 10,000, respectively. Capacities of intermediate nodes are not used since flow into such nodes is assumed to be equal to flow out of them.

In this paper, we are going to address the capacities at the intermediate nodes of a network and investigate the models where the intermediate storage is allowed, i.e., the inflow at an intermediate node is allowed to be greater than the outflow, and the excess flow can be stored at that node provided it does not exceed the node capacity. As the total flow value is bounded by arc capacities, we assume that a node capacity should be equal or greater than the sum of the capacities of all the incoming arcs. Hence, the total outflow from the sources can be larger than the total inflow into the sinks. We propose a modification of the existing maximum flow models from this perspective. Our main aim is to use the maximum arc capacity to push as much flow as possible out of the sources. Our newly proposed model can only be used if the total capacity of arcs leaving a source is not smaller than the minimum cut capacity of the network. However, we need to clarify the situation at intermediate nodes. This is not the same as in sink nodes but has limiting capacity constraint and the farthest nodes from the source are assumed to be more safer. On the other hand, sink node is well defined with respect to capacity, i.e., it has unbounded capacity, has enough facility and safety. There may be a number of questions that arise related to the intermediate storage. Here, we are not giving all answers to these questions; however, we just assume that the intermediate nodes have some capacities in which we can store the excess flow and utilize them as much as possible. Furthermore, we also assume that it is beneficial to push the flow as far from the sources as possible towards the sinks for more safety. We claim that this is a novel idea which can be used to solve many real-life problems. In Fig. 1, by allowing intermediate storage at the transit state P

respecting its capacity, our model uses the excess capacity 5000 from L to P via arcs (L, R) and (R, P) or (L, P), and shifts 20,000 people per day in which 5000 people are allowed to stay at P. Then, transferring people from intermediate states to the final destination in suitable time is another problem that we are not dealing with in this work.

By allowing the excess flow storage at the intermediate nodes, we introduce maximum static and maximum dynamic flow models and solve these problems in polynomial time in two terminal networks. If the two terminal network is series-parallel, then we solve the EAFP with intermediate storage in polynomial time complexity. In the solution procedure, we adopt the known solution techniques of LMSFP and LMDFP. For this, we fix the priority ordering of the nodes except the source node. We always give first priority to the sink. We find the minimum distances from the source to each intermediate nodes. We give the second priority for the intermediate node with the longest distance from the source and so on. Then, we maximize the flow according to the priority orderings. A compact version of this work is found in [22]. Recently, a similar idea has been used in [5] for the lexicographically maximum flow in the set of terminals with given priority ordering. However, the idea to send flow to intermediate nodes has first been suggested in [8].

If we assume that some emergency facilities are located at intermediate nodes with given capacities and nodes with longer distance from the source are safer than nodes with shorter one, then our model is applicable for the evacuation planning. The evacuation planning problem models the process of shifting residents from risk areas to safe places as quickly and efficiently as possible. After disasters, the movement of flow towards the sources is not allowed and the corresponding lanes remain empty. On the other hand, due to heavy traffic and large number of evacuees on the street, the lanes outwards from the sources are congested. The contraflow configuration is a widely accepted technique to utilize the empty lane by reversing its direction towards safe areas that helps to remove the congestion and flow can be maximized effectively. There are number of heuristics developed to solve the contraflow problem. Kim and Shekhar [18] defined the contraflow problem using graph and flow theory, and developed two heuristics, i.e., flip high flow edge and simulated annealing to minimize the evacuation time. Kim et al., [17] studied a macroscopic model integrating road capacity constraints, multiple sources, congestion, and scalability; presented its computational difficulties; and proved that it is NP-hard in general network. They also presented two heuristics, i.e., greedy and bottleneck, and reduced at least 40% evacuation time with at most 30% of the total arc reversals.

In the cases, when each node has an associated danger factor, Vogiatzis [34] presented a heuristic algorithm, reversing at most a given number of arcs, that can efficiently solve large size evacuation problems. Within the heuristic, they provided a clustering technique, using modified Dijkstra algorithm introducing the danger factor to each node in which a vehicle from lower danger factor node does not move to the node with higher danger factor, to divide the problem into smaller and easier subproblems. Their main objective is to minimize the number of vehicles that have to spend time on the most endangered nodes. The management of lanes plays vital role for the efficient evacuation planning. Zheng and Arulselvan [35] investigated in the large size lane management problem and presented a discrete time traffic

assignment model that minimizes the overall delay. They introduced a time-expanded network in discrete time without intermediate storage to reduce the problem in small size and presented a mixed integer program on it for which several efficient heuristic algorithms exist. In their time-expanded network, they used a scaling factor in order to condense the large network. By introducing a penalty function on each arc with capacity and transit time, Andreas and Smith [3] presented an optimal priority evacuation tree in which their main objective is to minimize the expected evacuation penalty. With lane reversals and convergence policies, Achrekar and Vogiatzis [1] investigated the evacuation planning problem on tree network.

On the other hand, there are a number of mathematical models and efficient algorithms dealing with evacuation planning problems using the technique of contraflow configuration in literature, [4, 7, 24, 25, 27–31] from the analytical point of views. If the minimum cut capacity in a two terminal network is symmetric, then the maximum dynamic flow can be increased up to double with contraflow configurations [27]. In multi-terminal networks, the maximum dynamic contraflow problem is NP-hard [31]. However, all these models concern the case without intermediate storage.

In this paper, we introduce a contraflow model with intermediate storage and present a polynomial time algorithm to solve the maximum dynamic contraflow problem (MDCFP) in a two terminal network. By reversing the direction of arcs at time zero and fixing it for all latter time, we maximize the flow from a single source to a single sink. Additionally, we use the remaining arc capacity of the network to push the flow from the source to the intermediate nodes. In the procedure, for the intermediate storage, we adopt the solution technique of LMDFP but, as we have given first priority to the sink, the flow entered in the sink gives the MDCF solution for the two terminal network and it is equal to the MDCF solution without intermediate storage as in [4, 31]. The excess arc capacity of the network is used to transship the flow from the source to the intermediate nodes in their priority ordering. Thus, the MDCF with intermediate storage is equal to the MDCF with flow conservation plus the amount of flow storage at intermediate nodes. Moreover, if the network is series-parallel, this solution has the earliest arrival property and thus, we get the earliest arrival contraflow solution in polynomial time complexity.

The organization of the paper is as follows. In Section 2, we discuss the basic terminology and mathematical models that are necessary throughout the paper. Section 3 investigates the static flow with intermediate storage. We present an efficient algorithm for the dynamic flow problem with intermediate storage in Section 4. The corresponding problem with lane reversal strategy is solved in Section 5. The paper concludes with Section 6.

2 Mathematical Model

Let $\mathcal{N} = (V, A, u, b, \tau, s, d, T)$ be a dynamic network, where V is a finite set of nodes with $|V| = n$, A is a finite set of directed arcs with $|A| = m$, $s \in V$ is a single source, and $d \in V$ is a single sink. Each node $v \in V$ has a node capacity $u : V \to \mathcal{Z}^+$ which bounds the amount of flow allowed to remain at v. Each arc $e = (v, w) \in A$ has a non-negative capacity $b : A \to \mathcal{Z}^+$, a transit time $\tau : A \to$

$\mathcal{Z}^+$, or a cost $c : A \to \mathcal{Z}^+$. The arc capacity b_e for all $e = (v, w) \in A$ represents the maximum units of flow that may enter the arc e from its tail v per time period. The time needed to travel the arc $e = (v, w)$ from node v to node w is the transit time τ_e. If a flow starts from v at time θ to move along arc $e = (v, w)$, it reaches to the node w at time $\theta + \tau_e$. The cost needed for sending one unit of flow through the arc e is c_e. The predefined parameter T denotes the permissible time window within which the whole evacuation process has to be completed. It may be discretized into discrete time steps $\mathbf{T} = \{0, 1, \ldots, T\}$. Let $I \subset V \setminus \{s, d\}$ be the set of intermediate nodes. Discarding the time function, the network $\mathcal{N} = (V, A, u, b, S, D)$ is the static network. Generally, we assume that no arcs enter the source node and no arcs exit the sink node so that $B_s = \emptyset$ and $A_d = \emptyset$ hold, where $B_v = \{e \mid e = (u, v) \in A\}$ and $A_v = \{e \mid e = (v, u) \in A\}$. However, in the case of contraflow configuration below, B_s and A_d are not empty in general.

2.1 Flow Models Without Intermediate Storage

Neglecting the cost of the arcs, a static $s - d$ flow without intermediate storage $\phi : A \to R^+$ of value $\text{val}(\phi)$ in objective function (1) satisfies the flow conservation and capacity constraints (2) and (3), respectively, [12]:

$$\text{val}(\phi) = \sum_{e \in B_d} \phi_e = \sum_{e \in A_s} \phi_e \tag{1}$$

$$\sum_{e \in B_v} \phi_e - \sum_{e \in A_v} \phi_e = 0, \quad \forall\, v \in V \setminus \{s, d\} \tag{2}$$

$$b_e \geq \phi_e \geq 0, \quad \forall\, e \in A \tag{3}$$

The maximum static flow (MSF) maximizes the objective functions (1) subject to constraints (2) and (3). Constraint (2), called the flow conservation constraint, indicates that total inflow into a node v must be equal to total outflow from it. These constraints imply that the total inflow into sink d must be equal to the total outflow from source s. Feasibility of the flow is bounded by the capacity constraint (3), i.e., flow ϕ_e cannot be greater than the capacity of an arc e for all $e \in A$. We denote the MSF value by $\text{val}_{\max}$. If the flow conservation constraint is also posed at s and d, then a circulation is obtained.

If the unit costs along the arcs are also considered, then we have to minimize the total cost. If we are given a flow value $\text{val}(\phi)$, the *minimum cost flow* (MCF) problem seeks to shift the flow with minimum cost $\sum_{e \in A} c_e \phi_e$ subject to (2) and (3) and $\sum_{e \in B_d} \phi_e = \text{val}(\phi)$. The MCF problem with zero circulation turns into the *minimum cost circulation flow* (MCCF) problem.

Let $\psi : A \times \mathbf{T} \to R^+$ be a dynamic flow in discrete time $\mathbf{T}$. The maximum dynamic flow problem maximizes the objective function $val(\psi, T)$ (4) subject to the constraints (5–7) without intermediate storage in time T, although, according to constraint (6), some flow may wait in intermediate nodes at time $\theta < T$, [12].

$$\text{val}(\psi, T) = \sum_{e \in As} \sum_{\sigma=0}^{T} \psi_e(\sigma) = \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{T} \psi_e(\sigma - \tau_e) \tag{4}$$

$$\sum_{e \in B_v} \sum_{\sigma=\tau_e}^{T} \psi_e(\sigma - \tau_e) = \sum_{e \in A_v} \sum_{\sigma=0}^{T} \psi_e(\sigma), \ \forall\, v \notin \{s, d\} \tag{5}$$

$$\sum_{e \in B_v} \sum_{\sigma=\tau_e}^{\theta} \psi_e(\sigma - \tau_e) \geq \sum_{e \in A_v} \sum_{\sigma=0}^{\theta} \psi_e(\sigma), \ \forall\, v \notin \{s, d\}, \ \theta \in \mathbf{T} \tag{6}$$

$$b_e(\theta) \geq \psi_e(\theta) \geq 0, \ \forall\, e \in A, \ \theta \in \mathbf{T} \tag{7}$$

If the objective is to maximize the flow at every time from the beginning, we call it the EAFP. The objective function of EAFP is (8), which has to be maximized at every time θ subject to the constraints (5–7).

$$\text{val}(\psi, \theta) = \sum_{e \in As} \sum_{\sigma=0}^{\theta} \psi_e(\sigma) = \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{\theta} \psi_e(\sigma - \tau_e). \tag{8}$$

2.2 Flow Models with Intermediate Storage

Let $C = (X, X')$ be an $s - d$ cut such that $s \in X$ and $d \in X'$ for $X,\ X' \subset V,\ X \cap X' = \emptyset$. Let X_C be the cut set of any given cut C such that $X_C = \left\{(v, w) \in A \mid v \in X, w \in X'\right\}$. We denote the cut capacity of an $s - d$ cut as

$$\lambda(X, X') = \sum_{(v,w) \in X_C} b_{(v,w)} = \sum_{(v,w) \in A} b_{(v,w)} a_{(v,w)}$$

where $a_{(v,w)} = 1$ if $v \in X$ and $w \in X'$, 0 otherwise. Then, the minimum cut, i.e., determine X and X' such that the capacity of the $X - X'$ cut is minimal, is obtained as

$$\lambda_{\min} = \min\left\{\lambda(X, X')\right\}.$$

Similarly, the maximum cut, i.e., determine X and X' such that the capacity of the $X - X'$ cut is maximum, is

$$\lambda_{\max} = \max\left\{\lambda(X, X')\right\}.$$

According to the max-flow min-cut theorem of [11], $\text{val}_{\max}(\phi) = \lambda_{\min}$. Due to the flow conservation on each node, to push flow outward from the source by exploiting the full arc capacity greater than $\lambda_{\min}$ is impossible. This is modified if we allow the intermediate storage, i.e., if the flow may not be conserved at each node. Then, we can push maximum flow outward from source using full arc capacities. Of course, only the flow equal to the minimum cut capacity can reach the sink. The remaining flow can be stored at intermediate nodes respecting their node capacities. Our aim in this article is to push the flow as far as possible from the source. Thus, the static flow (cf. model (1–3)) is modified to model (9–12) with permissible intermediate storage.

$$\text{val}(\phi) = \sum_{e \in A_s} \phi_e = \sum_{e \in B_d} \phi_e + \sum_{v \in I : u_v \geq 0} \phi_v \tag{9}$$

$$\phi_v = \sum_{e \in B_v} \phi_e - \sum_{e \in A_v} \phi_e \geq 0, \quad \forall\, v \in I \tag{10}$$

$$b_e \geq \phi_e \geq 0, \quad \forall\, e \in A \tag{11}$$

$$u_v \geq \phi_v \geq 0, \quad \forall\, v \in I \tag{12}$$

Here, the function $\phi_v : I \to \mathcal{Z}^+$ be the amount of flow storage at the intermediate nodes.

Similarly, if we maximize the flow leaving from the source and allow storage at intermediate nodes in time $\theta \in \mathbf{T}$, then the maximum dynamic flow problem maximizes the objective function $\text{val}(\psi, T)$ in (13) with respect to $\psi_e(T)$ and $\psi_v(T)$ by satisfying the constraints (14–16).

$$\text{val}(\psi, T) = \sum_{e \in As} \sum_{\sigma=0}^{T} \psi_e(\sigma) = \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{T} \psi_e(\sigma - \tau_e) + \sum_{v \in I,\ u_v > 0} \psi_v(T) \tag{13}$$

$$\psi_v(\theta) = \sum_{e \in B_v} \sum_{\sigma=\tau_e}^{\theta} \psi_e(\sigma - \tau_e) - \sum_{e \in A_v} \sum_{\sigma=0}^{\theta} \psi_e(\sigma) \geq 0, \quad \forall\, v \in I,\ \theta \in \mathbf{T} \tag{14}$$

$$b_e(\theta) \geq \psi_e(\theta) \geq 0, \quad \forall\, e \in A,\ \theta \in \mathbf{T} \tag{15}$$

$$u_v \geq \psi_v(\theta) \geq 0, \quad \forall\, v \in I\ \theta \in \mathbf{T} \tag{16}$$

Here, the function $\psi_v(\theta) : I \times \mathbf{T} \to \mathcal{Z}^+$ be the amount of flow storage in the intermediate nodes within time θ.

Unlike to the objective function (4), we maximize the flow value leaving from the source in objective function (13). All flow value sent from the source may not reach the sink in time T. Certain flow value is allowed to be stored at intermediate nodes in every time θ as well as in time period T as shown by constraint (14). However, we respect the capacity of each arc with constraint (15) and of each node in (16). The *earliest arrival flow* (EAF) with intermediate storage is given by maximization of (17) for each $\theta \in \mathbf{T}$ subject to the constraints (14–16).

$$\text{val}(\psi, \theta) = \sum_{e \in As} \sum_{\sigma=0}^{\theta} \psi_e(\sigma) \geq \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{\theta} \psi_e(\sigma - \tau_e) \tag{17}$$

3 Static Flow

In this section, we investigate the maximum static flow problem with intermediate storage and present a polynomial time algorithm to solve it.

Definition 1 For the static network $\mathcal{N} = (V, A, b, u, s, d)$, the maximum static flow problem (MSFP) with intermediate storage maximizes the flow leaving the source and pushes flow as far as possible towards the sink allowing excess flow storage at intermediate nodes.

To solve the MSFP with intermediate storage, we convert our network into a single source and multi-sink network and also fix their priority ordering as follows. As we have a single source, the priority of sources is unique. For the maximum flow problem, as much flow as possible should be shifted to the sink and the sink's priority should be first. Then, we fix the priority of the intermediate nodes having positive capacity to store excess flow, based on their distances from the source. Let, for $v \in I$, $\mathrm{dis}(s, v)$ be the shortest distance of node v from the source s with respect to the cost (time). To find $\mathrm{dis}(s, v)$ from s to each $v \in I$ with $u_v > 0$, we use Dijkstra's algorithm, [9], because our network has non-negative weights $\tau_e \geq 0$ or $c_e \geq 0$ for each arc. Here, we assume that the lower bound of u_v is $u_v \geq \sum_{e \in B_v} u_e > 0$ and upper bound is sufficient because total flow value is bounded by only arc capacities. As we are pushing flow as far as possible from the source, the node v_1 having the longest distance, i.e., the maximum distance among the shortest distances, of an intermediate node with $u_v > 0$ gets the second priority, the node v_2 with the second longest distance gets the third priority and so on.

Then, a dummy shelter v' with dummy arc (v, v') having distance $c_{(v,v')} = 0$ and capacity $b_{(v,v')} = u_{v'} = u_v$ is added to each intermediate node $v \in I$ with capacity $u_v > 0$. Let the set of dummy shelters be denoted by I'. The dummy shelter v' is also a sink and gets the same priority as the node v. Thus, the given s-d network is modified into one with a single source and multiple sinks with updated parameters as $\mathcal{N}' = (V, A, b, u, c, s, D)$, where $V = V \cup I'$, $A = A \cup \{(v, v')\}$, $b = b \cup \{b_{(v,v')}\}$, $u = u \cup \{u'_v\}$, $c = c \cup \{c_{(v,v')}\}$ and multiple sinks $D := \{d\} \cup I'$. Now, we have an order $\{d\} \subset D_1 = \{d, v_1\} \subset D_2 = \{d, v_1, v_2\} \subset \cdots \subset \{d, v_1, v_2, \ldots, v_r\}$ of subsets of D and, if the maximum flow value that can enter the sinks in D is $\mathrm{val}_{\max}(D)$, then a maximal flow that delivers $\mathrm{val}_{\max}(D_k)$ units into each $D_k, k = 1, \ldots, r$ is a lexicographically maximum static flow on the sinks, [20]. Thus, we solve the LMSFP without intermediate storage.

We maximize the flow entering to the sink set D with fixed priority ordering. After getting this solution, the dummy shelters are removed and the flows are shifted to the corresponding intermediate nodes. This gives the MSF with intermediate storage. It should be noted that the MSF obtained at sink d is equal to the MSF without intermediate storage. It can be obtained by any maximum flow algorithm. Moreover,

for intermediate storage, we use only excess capacities outward from source and send the excess flow as far as possible at nodes $v \in I$ with capacities at least $\sum_{e \in B_v} u_e$ so that it does not violate the optimality of MSF solution. We present the resulting Algorithm 1 as follows:

Algorithm 1 Maximum static flow with intermediate storage.

Input : Given network $\mathcal{N} = (V, A, b, u, s, d)$.
Output: The MSF with intermediate storage.

1. Use shortest path algorithm to find the minimum distance $dis(s, v)$ for all $v \in I$ with $u_v \geq \sum_{e \in B_v} u_e$.
2. Give first priority to the sink d, second priority of the intermediate node v at longest distance from the source with $u_v \geq \sum_{e \in B_v} u_e$ and so on.
3. Modify the network into a single source and multi-sink by creating dummy shelters I' with updated parameter as $\mathcal{N}' = (V, A, b, u, c, s, D)$.
4. In modified network $\mathcal{N}'$, compute a maximum flow in the given priority ordering.
5. Transform the solution to the original network $\mathcal{N}$ by removing dummy shelters and dummy arcs and obtain
   - Maximum flow at sink d.
   - Intermediate storage at $v \in I$.

Theorem 1 *An optimal solution for the MSFP with intermediate storage can be computed with Algorithm 1.*

Proof First, we show that Algorithm 1 is feasible. Steps 1 and 2 are feasible. As creating dummy shelters v' with zero cost and capacity $u_{v'} = u_v$ for all $v \in I$ with $u_v > 0$ is feasible, Step 3 is also feasible. A maximum flow computation without intermediate storage in Step 4 gives the feasible solution in the modified network $\mathcal{N}'$ as the creation of dummy shelters do not violate the capacity constraints. The transformation to a solution in the original network does not violate the feasibility. Thus, our algorithm is feasible.

Now, we prove the optimality of the solution computed by our algorithm. As, with the dummy shelters, the flow conservation constraint at the intermediate nodes is satisfied, the maximum flow is computed iteratively by satisfying the given priority ordering. In the procedure, we choose the sink with first priority and compute a MSF using any maximum flow algorithm. Considering the obtained flow as initial flow ϕ, the residual network $\mathcal{N}'_r(\phi) = (V, A(\phi))$, where $A(\phi) = A_F(\phi) \cup A_B(\phi)$ with $A_F(\phi) = \{e = (v, w) \in A \mid \phi_e < b_e\}$ and $A_B(\phi) = \{e' = (w, v) \in A \mid \phi_e > 0\}$ with arc length τ_e, i.e., cost for $e \in A_F(\phi)$ and $-\tau_e$ for $e' \in A_B(\phi)$ is constructed. The residual capacity $b(\phi) : A(\phi) \rightarrow \mathcal{R}^+$ is defined as $b_e(\phi) = b_e - \phi_e$ for $e \in A_F(\phi)$ and $b_e(\phi) = \phi_{e'}$ for $e' \in A_B(\phi)$. This network is used to increase the source-sink flow through augmenting paths. If there does not exist an augmenting

path to enlarge the flow from the source to the chosen sink, then we obtain the maximum flow for the sink. In next iteration, we choose the sink with second priority and find the maximum flow as in first sink and so on. According to [20], the priority-based maximum flow is the LMSF in the modified network $\mathcal{N}'$ without intermediate storage. As we have dummy shelters in modified network, flow reaching these shelters is shifted back to the corresponding intermediate nodes that gives the solution to the MSFP problem and it satisfies

$$\text{val}(\phi) = \sum_{e \in A_s} \phi_e = \sum_{e \in B_d} \phi_e + \sum_{v \in I,\, u_v > 0} \phi_v.$$

This completes the proof. □

Corollary 1 *The MSFP with intermediate storage can be computed in polynomial time complexity.*

Proof The measurement of distances in Step 1 can be done in $O(n^2)$ time. The priority ordering can be fixed in linear time. The construction of dummy shelters is also done in linear time. As the maximum flow computation with priority ordering can be completed in polynomial time as in [20], our algorithm has also polynomial time complexity. □

Example 1 Consider a network with a single source s and a single sink d in which $V = \{s, x, y, d\}$, $A = \{(s, x), (s, y), (x, y), (x, d), (y, d)\}$ and $I = \{x, y\}$ as given in Fig. 2(i). Source s and sink d have infinite capacities. Intermediate nodes x and y have capacities 17 and 15, i.e., 17 and 15 units of flow can be stored in these nodes, respectively. Each arc has a capacity and a cost function value. For example, arc (s, x) has capacity 6 units and cost 1 unit. That means, 6 units of flow can together travel along arc (s, x) with 1 unit cost for each unit of flow. The minimum cut $\lambda_{\min}(X, X')$ of this network is 10 and the maximum capacity outgoing from source is 14. As the capacity of outgoing arcs from the source is greater than the $\lambda_{\min}(X, X') = 10$, the maximum flow with intermediate storage is possible.

First, we calculate the shortest distances of nodes in I from s. Using the shortest path algorithm, we get $\text{dis}(s, x) = 1$ and $\text{dis}(s, y) = 2$ in Step 1 of Algorithm 1. In Step 2, we fix the priority ordering as d, y, x, i.e., we maximize the flow to node d first, then to node y and last to node x. We use an existing maximum flow algorithm designed for the case without intermediate storage to maximize the flow in this priority ordering. However, we cannot apply it directly in this example because nodes x and y are not sink nodes. So, we modify the network by creating dummy shelters x' and y' of x and y with their capacities and priorities, respectively, as given in Fig. 2(ii). The set of dummy shelters is $I' = \{x', y'\}$, $V = V \cup I'$, $A = A \cup \{(x, x'), (y, y')\}$ and $D = \{d\} \cup I'$. Thus, the resulting network has a single source s and multiple sinks D as in Step 3.

Based on the solution procedure of the LMSF problem in [20], we maximize the flow in the given priority ordering. Then, the obtained solution is transformed into a

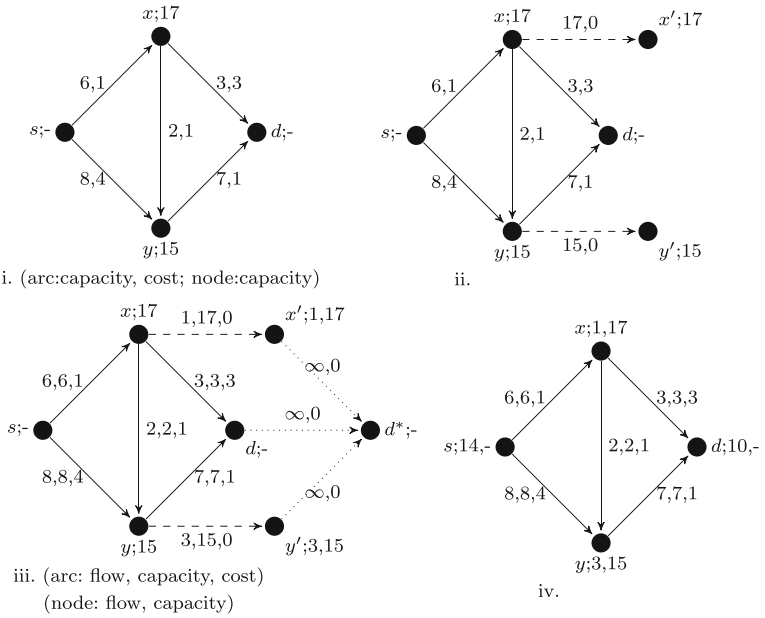

Fig. 2 (i) Given network. (ii) Network with dummy shelters. (iii) Extended network. (iv) MSF with intermediate storage

solution in the original network as in Fig. 2(iv) which gives the MSF with intermediate storage. Here, paths $\gamma_1 = s - x - y - d$, $\gamma_2 = s - x - d$, and $\gamma_3 = s - y - d$ with respective flows 2, 3, and 5 give the maximum flows to the sink d. Path $\gamma_4 = s - y - y'$ carries maximum 3 units of flow to store at node y and arc (s, y) becomes saturated. As arc (x, y) is already saturated, path $\gamma_5 = s - x - x'$ stores 1 unit of flow at node x and arc (s, x) becomes saturated. Thus, we get that the maximum flow leaving from the source is 14 which is the maximum flow with intermediate storage.

Notice that we can use any maximum flow algorithm in the modified network without intermediate storage to find the MSF with the intermediate storage in the original network. There is a long history of the polynomial time maximum flow algorithms whose optimal solutions are guaranteed by the fundamental max-flow min-cut theorem, i.e., the maximum flow value equals to the minimum cut capacity. The maximum flow algorithm of [12] ensures a maximum static flow ϕ if and only if the corresponding residual network does not contain an augmenting path. For integer arc capacity, the augmenting path algorithm always terminates, since each augmentation increases the flow value by at least one. By scaling the capacities, the running time can be improved to $O(m^2 \log b_{\max})$, where $b_{\max}$ represents the maximum integer valued arc capacity. If we use the maximum flow algorithm of [14], to compute the maximum static flow in the modified network with given priority orderings, the

complexity of the algorithm is still improved. The shortest augmenting path algorithm that uses a unit path length function, the blocking flow algorithm that augments along a maximal set of shortest paths with respect to a blocking flow and the push-relabel algorithm that works with non-conservation of flows except at the source and sink nodes turn into a strongly polynomial time algorithm with complexity $O(nm^2)$, $O(n^2m)$, and $O(nm\log(n^2/m))$, respectively. We refer to [14] for a brief review of maximum flow algorithms. Thus, we solve the MSFP with intermediate storage which takes n times the complexity of existing algorithms for the problem without intermediate storage in the worst case.

4 Dynamic Flow

In this section, we investigate the dynamic flow problems with intermediate storage. We introduce the MDFP with intermediate storage and present an efficient algorithm to solve it on a two terminal network. If the two terminal network is series-parallel, then we can compute the EAF with intermediate storage in polynomial time.

Definition 2 Given a two terminal dynamic network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$, the MDFP with intermediate storage maximizes the flow leaving the source and pushes the flow as far as possible towards the sink within the given time horizon T allowing excess flow storage at intermediate nodes.

To solve this problem, we modify the given network. We fix the priority ordering as in the static case in Section 3. Then, dummy shelters are created for the intermediate nodes with their respective capacities and priorities. We assume that each dummy arc has zero transit time $\tau_{(v,v')}$. Now, the network has a single source s and multi-sink D with priority ordering. Let $\mathcal{S} = \{s\} \cup D$ be the set of terminals with $|\mathcal{S}| = \delta$, where δ is the number of terminals. Notice that, in worst case, $\mathcal{S}$ is the same as V and $\delta = n$ but in general $\mathcal{S} \subseteq V$ and $\delta \leq n$. The updated parameters V, A, b, u are the same as in static network with additional updated transit time $\tau = \tau \cup \left\{\tau_{(v,v')}\right\}$ that gives the modified network $\mathcal{N}' = (V, A, b, u, \tau, s, D)$ in which we solve the lexicographically maximum dynamic flow problem (LMDFP). For the given time T and an ordered set of terminals, the LMDFP finds a feasible flow that lexicographically maximizes the flow entering each terminal with the given priority.

We adopt the lexicographically maximum dynamic flow (LMDF) algorithm of [16]. An overview of LMDF algorithm is as follows. We connect all the terminals $s_i \in \mathcal{S}, i = 1, \ldots, k$ with a super terminal d^* with $b_{(d^*,s_i)} = \infty$ and $\tau_{(d^*,s_i)} = 0$ so that $V := V \cup \{d^*\}$, $A_{k+1} = A \cup \{(d^*, s_i) : s_i \in \mathcal{S}\}$ and the resulting network is $\mathcal{N}'_{k+1}$. We start with $\psi_{k+1} = 0$ and the set of chains Γ_{k+1} is empty in $\mathcal{N}'_{k+1}$. The algorithm has k iterations in decreasing order. For $i = k, \ldots, 1$, set the arc set $A_i = A_{i+1}$ and, if $s_i \in \mathcal{S}$ is a sink, then the arc (s_i, d^*) is added to A_i with $b_{(s_i,d^*)} = \infty$ and $\tau_{(s_i,d^*)} = -(T+1)$ so that ϕ_i is a min-cost circulation in $\mathcal{N}'_i(\phi_{i+1})$ using τ as cost. On the other hand, if $s_i \in \mathcal{S}$ is a source, then arc (d^*, s_i) is deleted from A_i and ϕ_i

is a min-cost maximum flow from d^* to s_i in the residual network $\mathcal{N}_i'(\phi_{i+1})$ using τ as arc costs. Then, the dynamic flow is updated as $\psi_i = \psi_{i+1} + \phi_i$. Let Δ_i be the standard chain decomposition of ϕ_i. Then, set the chain $\Gamma_i = \Gamma_{i+1} + \Delta_i$. Finally, the output is obtained as $\Gamma = \Gamma_1$.

Lemma 1 [16] *For given priority ordering, the LMDF problem can be solved in polynomial time on a multi-terminal network.*

As our modified network has single source and multi-terminal, the adoption of LMDF algorithm to compute the priority-based maximum dynamic flow leads to Algorithm 2.

Algorithm 2 Maximum dynamic flow with intermediate storage.

Input : Given network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$.
Output: The MDF with intermediate storage.

1. Find the minimum distances $d(s, v)$ for all $v \in I$ with $u_v \geq \sum_{e \in B_v} u_e$ using a shortest path algorithm.
2. Give first priority to the sink d, second priority of the intermediate node v with longest distance from the source with $u_v \geq \sum_{e \in B_v} u_e$ and so on.
3. Modify the network into single source and multi-sink by creating dummy shelters. The resulting network is $\mathcal{N}' = (V, A, b, u, \tau, s, D)$.
4. Compute priority based maximum dynamic flow, i.e., LMDF on $\mathcal{N}'$.
5. Transform the solution to the original network by removing dummy shelters and dummy arcs and obtain
   - Maximum dynamic flow at sink d.
   - Intermediate storage at $v \in I$.

Theorem 2 *Algorithm 2 computes an optimal solution for the MDFP with intermediate storage on two terminal network.*

Proof As in Theorem 1, Algorithm 2 is feasible. Due to the dummy shelters, the flow conservation at each node is satisfied. With the priority ordering of nodes in D, Step 4 computes the dynamic flow via δ MCF computations which is feasible and is lexicographically maximal for a given time horizon T. Then, the priority-based maximum dynamic flow is shifted from the dummy shelters to the corresponding intermediate nodes keeping the sink flow unchanged. That gives a maximum dynamic flow with intermediate storage. Thus, we have

$$\text{val}(\psi, T) = \sum_{e \in As} \sum_{\sigma=0}^{T} \psi_e(\sigma) = \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{T} \psi_e(\sigma - \tau_e) + \sum_{v \in I,\, u_v > 0} \psi_v(T).$$

This completes the proof. □

Theorem 3 *The maximum flow entering the sink gives the maximum dynamic flow without intermediate storage.*

Proof As the sink is ordered with first priority, the flow entering the sink is maximized as long as the capacity permits at each time point. Thus, the flow at sink is equal to $\lambda_{\min}$, where $s \in X$ and $d \in X'$ at each time point from $0, 1, \ldots, T$. If there is no intermediate storage, i.e., flow is conserved at the nodes, $\lambda_{\min}$ is equal to a maximum flow leaving from the source. Thus, we have

$$\mathrm{val}_{\max}(\psi, T) = \sum_{e \in As} \sum_{\sigma=0}^{T} \psi_e(\sigma) = \sum_{e \in Bd} \sum_{\sigma=\tau_e}^{T} \psi_e(\sigma - \tau_e).$$

□

Corollary 2 *The MDFP with intermediate storage capacity can be solved in $O(\delta \times MCF(n, m))$ time, where $\delta = |\mathcal{S}|$ is the number of terminals and $MCF(n, m)$ is the complexity of the single minimum cost flow computation.*

Proof The complexity of Algorithm 2 is dominated by the complexity of Step 4. The priority-based maximum dynamic flow is computed in $O(\delta \times MCF(n, m))$ time, [16], where δ gives the number of terminals and $MCF(n, m)$ represents the time complexity of a single minimum cost flow computation MCF using the original network which is bounded within the polynomial time $O(m \log n(m + n \log n))$, [21]. □

Example 2 If we consider the cost (time) function of the arcs in Fig. 2(i), then the network is a two terminal dynamic network in which we compute the MDF with intermediate storage using Algorithm 2 as follows. By creating dummy shelters, the modified network is considered as in Fig. 2(ii). The single source and multi-sink modified network is reduced to the single source and single sink network $\mathcal{N}^*$, i.e., an extended network is constructed by connecting all the sinks with a super sink d^* using arcs with zero costs and infinite capacities as in Fig. 2(iii). By solving the minimum cost flow problem at each sink in the modified network, we obtained LMDF without intermediate storage. Shifting the flow into original network, if the predetermined time horizon is $T = 5$, we can compute the MDF with intermediate storage and the solution is visualized in the following time-expanded network.

For the extended dynamic network (cf. Fig. 2(iii)), the time-expanded network is defined as an expansion of the dynamic network where each node v is copied T times to obtain a node $v(\theta)$ for each $v \in V$ and each $\theta \in \{0, 1, \ldots, T\}$. For each arc $e = (v, w) \in A$, the arc from $v(\theta)$ to $w(\theta + \tau_{(v,w)})$ has capacity $b_{(v,w)}$, called movement arc. For each node $v \in V$, the arc from $v(\theta)$ to $v(\theta + 1)$ has capacity u_v, and it is called a holdover arc and allows storage of flow in the node. Moreover, the MSF solution on the time-expanded network is equivalent to the MDF solution on original network, [12]. Thus, at time $T = 5$, sink d receives 17 units flow via the paths $\gamma_1 = s - x - y - d$, $\gamma_2 = s - x - d$ and $\gamma_3 = s - y - d$. Node y collects

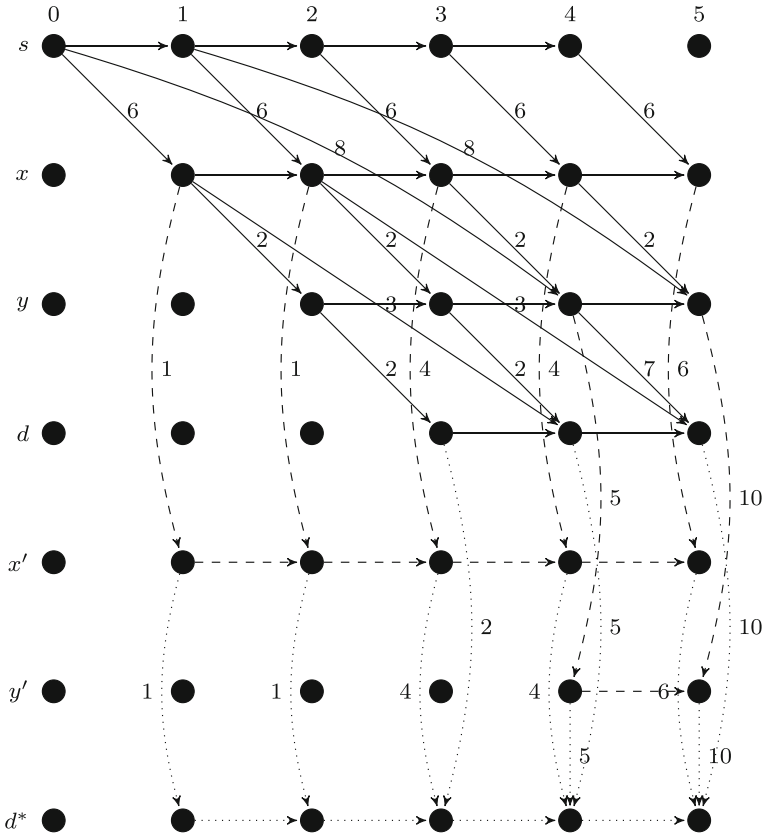

Fig. 3 Maximum dynamic flow without intermediate storage in the time-expanded network

15 units flow via the paths $\gamma_4 = s - x - y$ and $\gamma_5 = s - y$. Similarly 16 units flow reaches to node x through the path $\gamma_6 = s - x$ (Fig. 3).

If we have a single arc $e = (s, d)$, it is a series-parallel network with source s and sink d. Let G_1 and G_2 be two series-parallel graphs with sources s_1 and s_2, and sinks d_1 and d_2, respectively. Then, the combination $S(G_1, G_2)$ obtained by identifying d_1 with s_2 in the series is a series-parallel graph with s_1 and d_2 as its terminals. The graph $P(G_1, G_2)$ obtained by identifying s_1 with s_2 and also d_1 with d_2 in the parallel combination is a series-parallel graph with $s_1 (= s_2)$ and $d_1 (= d_2)$ as its terminals.

We can solve the earliest arrival flow problem with intermediate storage in a two terminal series-parallel network with $n > 2$ and the boundary condition of [32] using Algorithm 2. The minimum cost flow is computed by successive computation of shortest paths with modified boundary condition where the cost of these paths is less than $T + 1$ instead being less than T.

Definition 3 Given a dynamic network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$ with $n > 2$, the earliest arrival flow problem (EAFP) with intermediate storage maximizes the flow leaving the source and pushes flow as far as possible towards the sink allowing excess flow storage at intermediate nodes at each time $0 \leq \theta \leq T$.

Theorem 4 *For a two terminal series-parallel network, the EAFP with intermediate storage can be solved in polynomial time.*

Proof The series-parallel graph does not contain any cycles with positive flow. Once we find the maximum dynamic flow with intermediate storage using Algorithm 2, the solution has the earliest arrival property, i.e., a cumulative amount of flows reaching the sink in every considered time period and all preceding time periods of the considered one has to be maximal. On the two terminal series-parallel network, the earliest arrival flow can be computed in $O(nm + m \log m)$ time. As we have $|\mathcal{S}| = \delta$, the earliest arrival flow with intermediate storage can be computed in $O(\delta \times (nm + m \log m))$ time complexity. □

5 Contraflow with Intermediate Storage

In this section, we investigate the dynamic flow problems with intermediate storage having the capability of arc reversals. Specially, the contraflow configuration of the network has been investigated from the emergency point of view. If the intermediate nodes have been designed to address the requirements of emergency situations (caused by different large scale natural and man-made disasters), then the contraflow configuration of our network flow model with intermediate storage can be applied. Using various related models and algorithms [7, 24, 25, 27–29, 31], the evacuation planer discourages people to move towards risk areas from safer places. Thus, the corresponding road arcs are unoccupied. However, the arcs outwards from sources become more congested due to a large number of evacuees and vehicles on the streets. The optimal arc reversal strategy plans make the traffic systematic and smooth by removing traffic jams.

Definition 4 Given a network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$ with symmetric transit times, the maximum dynamic contraflow problem (MDCFP) with intermediate storage maximizes the flow leaving the source and pushes flow as far as possible towards the sink allowing excess flow storage at intermediate nodes in the given time horizon T by reversing the direction of arcs at time zero.

The maximum dynamic contraflow (MDCF) model without intermediate storage has been introduced in [4, 31]. They presented a polynomial time algorithm to solve the problem in two terminal network. Here, we adopt their technique to solve the MDCFP with intermediate storage, i.e., we solve the single source and a single sink contraflow problem and pushes maximum flow value using maximum capacity away from the source. At zero time, we construct an auxiliary network

$\overline{\mathcal{N}} = (V, E, \bar{b}, u, \bar{\tau}, s, d, T)$ of the given network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$ in which the capacity and transit times are modified. Let us assume that arc capacity $\bar{b} = b_{\bar{e}}$ and arc transit time $\bar{\tau} = \tau_{\bar{e}}$ for all $\bar{e} \in E$ are given. The modified arc capacity $\bar{b}$ is constructed by adding the arc capacities of oppositely directed arcs; the transit time is the same as for the original arc. For each node $v \in I$, we compute the shortest distance $d(s, v)$ from source s and fix the priority ordering as in Section 4. For each node v with $u_v \geq \sum_{e \in B_v} u_e$, we construct dummy shelter v' at zero distance with capacity $b_{(v,v')} = u_v = u_{v'}$ having the same priority ordering as in v so that the network becomes the modified auxiliary network. In this network, we compute the MCCF with single sink d having first priority and obtains MDF without intermediate storage. Then, we update the remaining capacities of the network and compute the MCCF for the dummy shelter v' with second priority respecting the node capacity $u_{v'}$. Again we update the arc capacities of the network and solve another MCCF problem for the dummy sink with third priority and so on as in [16].

Then, the solution is transformed into auxiliary network by removing the dummy shelters that gives the MDF at sink d and storage at intermediate nodes $v \in I$, i.e., we get the MDF with intermediate storage. As we are doing the contraflow configuration for the single sink network, the MDF at d on the auxiliary network is equivalent to the MDCF for original network $\mathcal{N}$ as in [4, 31]. The intermediate storage does not change the MDCF solution for the original network with single sink d. We use Algorithm 3 to find the MDCF with intermediate storage on a two terminal network.

Algorithm 3 Maximum dynamic contraflow with intermediate storage.

Input : Given network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$

Output: The MDCF with intermediate storage.

1. Construct the corresponding auxiliary network $\overline{\mathcal{N}} = (V, E, \bar{b}, u, \bar{\tau}, s, D, T)$, where capacities and transit times of each arcs are defined as

$$b_{\bar{e}} = b_e + b_{e'}$$
$$\tau_{\bar{e}} = \begin{cases} \tau_e & \text{if } e \in A \\ \tau_{e'} & \text{otherwise} \end{cases}$$

   with the reversal of an arc $e = (v, w)$ as denoted by $e' = (w, v)$.
2. Solve Steps (1-3) of Algorithm 2 and obtain the modify auxiliary network $\overline{\mathcal{N}}' = (V, A, b, u, \tau, s, D, T)$
3. Find the priority based maximum dynamic flow, without intermediate storage on modified auxiliary network $\overline{\mathcal{N}}'$.
4. Decompose the flow in Step 3 into paths and removable cycles.
5. Remove the dummy shelters and transform the solution on auxiliary network that gives the priority based MDF with intermediate storage.
6. Reverse arc $e' \in A$ upto the capacity $\phi_e - b_e$ if and only if $b_e < \phi_e$, b_e replaced by 0 whenever $e \notin A$.
7. The obtained priority based MDF without intermediate storage is equivalent to the MDCF with intermediate storage for single sink d for original network.

Theorem 5 *Algorithm 3 computes an optimal solution for the MDCFP with intermediate storage on a two terminal network.*

Proof We prove this theorem in two steps. First, we prove the feasibility of Algorithm 3. Steps 1, 2, 3, 4, 5 and 7 of the algorithm are feasible. So, it is enough to show that only Step 6 is well defined. As Step 4 does not contain any cycle, there is either a flow along arc $e = (v, w)$ or $e' = (w, v)$ but never in both arcs in Step 6. This proves that the flow is not greater than the reversed capacities on all arcs at all time units. Thus, Step 6 is also feasible.

Now, we prove that Algorithm 3 yields an optimal solution as follows. For the given s-d network, the contraflow configuration is performed as in Step 1 of Algorithm 3 at zero time and it is fixed for all time periods. In the auxiliary network $\overline{\mathcal{N}}$, shortest distance $\text{dis}(s, v)$ for all $v \in I$ is computed, the priority ordering is fixed, and the modified auxiliary network $\overline{\mathcal{N}}' = (V, A, b, u, \tau, s, D, T)$ is constructed by creating dummy shelters as in Algorithm 2. With dummy shelters, the modified auxiliary network is multi-sink network with fixed priority ordering. We use the algorithm of [16] and compute the priority-based MDF without intermediate storage on the multi-sink network. The dummy shelters are removed and solution is transformed into the s-d auxiliary network that gives the MDF for single sink d with intermediate storage. According to [4, 31], every maximum dynamic flow without intermediate storage on s-d auxiliary network is equivalent to the maximum dynamic contraflow without intermediate storage on given s-d network. As we use the excess arc capacity to transship the flow from s to intermediate nodes, the s-d MDCF does not change by allowing intermediate storage. This completes the proof. □

Corollary 3 *The MDCFP with intermediate storage can be solved in polynomial time complexity in two terminal network.*

Proof Steps 1 and 6 can be solved in linear time. As the complexity of finding MDF with intermediate storage in Step 3 is of polynomial time (cf. Corollary 2), the overall complexity of Algorithm 3 is polynomial time. □

Example 3 Consider a two terminal network as given in Fig. 4(i). Here, flow is allowed to move in both directions. From the evacuation planning point of view, flow along arcs towards sources is discouraged and thus these arcs remain empty. By reversing these empty arcs using Step 1 of Algorithm 3, we construct an auxiliary network as in Fig. 4(ii). In the auxiliary network, we compute the maximum static contraflow (MSCF) using Algorithm 1 as in Example 1. Figure 4(iv) gives the MSCF with intermediate storage which is obtained as a fundamental solution for the MDCF problem with intermediate storage. In this static solution, we can see that due to priority ordering there is no flow storage in intermediate node x although it has enough capacity.

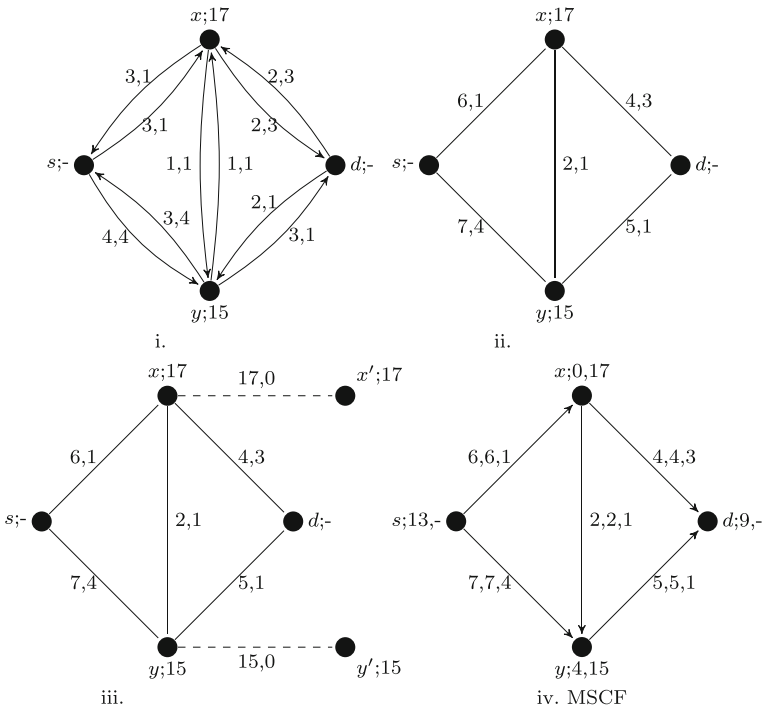

Fig. 4 (i) Given two-way network. (ii) Auxiliary network. (iii) Auxiliary network with dummy shelters. (iv) MSCF with intermediate storage

For the given time period $T = 5$, we compute the MDF with intermediate storage using Algorithm 2 as in Example 2 in the auxiliary network which is equal to the MDCF with intermediate storage for the original network. Table 1 compares the MDF and MDCF with intermediate storage in the original network. This shows that contraflow configuration of the given network increases the flow significantly.

If flow is maximized at every time point from the beginning in MDCFP (cf. Definition 4), then this results in the earliest arrival contraflow problem (EACFP) with

Table 1 MDF and MDCF with intermediate storage from Fig. 3

	MDFwIS			MDCFwIS		
Time T	Node x	Node y	Node d	Node x	Node y	Node d
1	-	-	-	-	-	-
2	-	-	-	-	-	-
3	2	-	1	4	-	2
4	2	2	3	4	4	6
5	3	5	5	6	9	9

intermediate storage [23]. The EACFP without intermediate storage has been solved in [24].

Definition 5 Given a $s-d$ series-parallel network $\mathcal{N} = (V, A, b, u, \tau, s, d, T)$ with $n > 2$, the EACFP with intermediate storage maximizes the flow leaving the source at each point of time from the beginning and pushes flow as far as possible towards the sink allowing excess flow storage at intermediate nodes.

The MDCF computed by Algorithm 3 in *s-d* series-parallel network has the earliest arrival property [23]. Thus, we can solve the EACFP with intermediate storage with polynomial time complexity. Based on this fact, we state the following theorem.

Theorem 6 *In the $s - d$ series-parallel networks, the MDCF and the EACF with intermediate storage are equivalent.*

Authors in [28, 30] presented the partial contraflow model. The main strategy of this work is to save the unused capacities of arcs using partial contraflow techniques. The saved arcs can be used for logistics and emergency supports. We can also adopt their models and algorithms to solve the dynamic contraflow problems with intermediate storage.

Moreover, the MDCF and the EACF solutions with intermediate storage are found in discrete time setting. If we assume that the time interval is continuous, then these problems can be solved in continuous time setting as well with the same time complexity using the natural transformation of [10]. The MDCF and EACF without intermediate storage in continuous time setting have been solved in [25, 26].

6 Conclusions

The dynamic flow models without intermediate storage are well investigated. Based on the max-flow min-cut relation, the full arc capacities may not be used to push remaining flow from the sources to the sinks. If we push flow out of the sources using full arc capacities, then intermediate storage should be possible because, in this case, flow may not be conserved on each intermediate node. Therefore, we have introduced the network flow models where intermediate storage is allowed. Polynomial time algorithms have been presented to solve the maximum flow problem in both static and dynamic two terminal networks. Moreover, we have solved the earliest arrival flow problem in a two terminal series-parallel network with polynomial time complexity.

Furthermore, we have introduced the dynamic contraflow approach with intermediate storage. A polynomial time algorithm has been presented to solve the MDCFP with intermediate storage on a two terminal network. If the network is two terminal series-parallel, then the MDCF has the earliest arrival property and thus, the EACFP with intermediate storage is also solved in polynomial time. The unused arc capacities can be saved with partial contraflow configuration.

To the best of our knowledge, the network flow models with intermediate storage are introduced and polynomial time algorithms are designed to solve the maximum static, maximum dynamic, and earliest arrival flow problems for the first time. These solutions with intermediate storage have more advantages in an evacuation planning in such a way that additional evacuees can be shifted onwards the sink node than the solutions without intermediate storage. Moreover, we are interested to extend this model to more general cases in future.

Acknowledgments The first author acknowledges the support of Alexander von Humboldt Foundation to her post doctoral research stay (November 2017–October 2019) at TU Bergakademie, Freiberg, Germany and her return fellowship (November 2019–October 2020) at Central Department of Mathematics, TU, Nepal. The authors would also like to thank the anonymous reviewers for their constructive suggestions.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Achrekar O, Vogiatzis C (2018) Evacuation trees with contraflow and divergence considerations. In: Kotsireas I, Nagurney A, Pardalos P (eds) Dynamics of Disasters. DOD 2017. Springer Optimization and Its Applications, vol 140. Springer, Cham. https://doi.org/10.1007/978-3-319-97442-2_1
2. Ahuja RK, Magnati TL, Orlin JB (1993) Network flows: theory, algorithms and applications. Prentice Hall, Englewood Cliffs, New Jersey
3. Andreas AK, Smith JC (2009) Decomposition algorithms for the design of a nonsimultaneous capacitated evacuation tree network. Networks 53(2):91–103
4. Arulselvan A (2009) Network model for disaster management. PhD Thesis, University of Florida, USA
5. Bhandari PP, Khadka SR, Ruzika S, Schaefer LE (2020) Lexicographically maximum dynamic flow with vertex capacities. Journal of Mathematics and Statistics 16:142–147
6. Cova T, Johnson JP (2003) A network flow model for lane-based evacuation routing. Transportation Research Part A:, Policy and Practice 37(7):579–604
7. Dhamala TN, Pyakurel U, Dempe S (2018) A critical survey on the network optimization algorithms for evacuation planning problems. International Journal of Operations Research (TW) 15(3):101–133
8. Dhamala TN, Pyakurel U (2012) Contraflow emergency evacuation by earliest arrival flow, PhD progress report presentation at University Grant Commission Nepal. www.researchgate.net/profile/Urmila_Pyakurel3/publications
9. Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer. Math. 1(1):269–271
10. Fleischer LK, Tardos E (1998) Efficient continuous-time dynamic network flow algorithms. Oper. Res. Lett. 23:71–80
11. Ford LR, Fulkerson DR (1956) Maximal flow through a network. Can. J. Math. 8:399–404
12. Ford LR, Fulkerson DR (1962) Flows in networks Princeton University Press , Princeton, New Jersey
13. Gale D (1959) Transient flows in networks. Michigan Math. J. 6:59–63
14. Goldberg AV, Tarjan RE (2014) Though maximum flow algorithms have a long history, revolutionary progress is still being made. Commun. ACM 57(8):82–89
15. Hoppe B, Tardos E (1994) Polynomial time algorithms for some evacuation problems. In: Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms, pp 433–441
16. Hoppe B, Tardos E (2000) The quickest transshipment problem. Math. Oper. Res. 25:36–62
17. Kim S, Shekhar S, Min M (2008) Contraflow transportation network reconfiguration for evacuation route planning. IEEE Trans. Knowl. Data Eng. 20(8):1115–1129

18. Kim S, Shekhar S (2005) Contraflow network reconfiguration for evacuation planning: a summary of results. In: Proceedings of 13th ACM Symposium on Advances in Geographic Information Systems GIS, vol 05, pp 250–259
19. Kotsireas IS, Nagurney A, Pardalos PM (2018) Dynamics of disasters-algorithmic approaches and applications. Springer Optimization and Its Applications
20. Minieka E (1973) Maximal, lexicographic, and dynamic network flows. Oper. Res. 21:517–527
21. Orlin JB (1993) A faster strongly polynomial minimum cost flow algorithm. Oper. Res. 41(2):338–350
22. Pyakurel U, Dempe S (2019) Network flow with intermediate storage: models and algorithms. preprint, 01/2009 tu bergakakademie freiberg
23. Pyakurel U, Dempe S (2019) Universal maximum flow with intermediate storage for evacuation planning. In: 4th International conference on Dynamics of Disasters July 1-4, Kalamata, Greece, appearing in the book Dynamics of Disasters, 2020
24. Pyakurel U, Dhamala TN (2015) Models and algorithms on contraflow evacuation planning network problems. Int. J. Oper. Res. (Taichung) 12(2):36–46
25. Pyakurel U, Dhamala TN (2016) Continuous time dynamic contraflow models and algorithms. Advances in Operations Research - Hindawi; Article ID 368587:1–7
26. Pyakurel U, Dhamala TN (2017) Continuous dynamic contraflow approach for evacuation planning. Ann. Oper. Res. 253(1):573–598
27. Pyakurel U, Dhamala TN, Dempe S (2017) Efficient continuous contraflow algorithms for evacuation planning problems. Ann. Oper. Res. 254(1–2):335–364
28. Pyakurel U, Nath HN, Dhamala TN (2018) Efficient contraflow algorithms for quickest evacuation planning. Sci. China Math. 61(11):2079–2100
29. Pyakurel U, Nath HN, Dhamala TN (2019) Partial contraflow with path reversals for evacuation planning. Ann. Oper. Res. 283:591–612
30. Pyakurel U, Nath HN, Dempe S, Dhamala TN (2019) Efficient algorithms for flow over time evacuation planning problems with lane reversal strategy. Mathematics 7(10):993. https://doi.org/10.3390/math7100993
31. Rebennack S, Arulselvan A, Elefteriadou L Pardalos PM (2010) Complexity analysis for maximum flow problems with arc reversals. J. Comb. Optim. 19:200–216
32. Ruzika S, Sperber H, Steiner M (2011) Earliest arrival flows on series-parallel graphs. Networks 10:169–173
33. Wilkinson WL (1971) An algorithm for universal maximal dynamic flows in a network. Oper. Res. 19(7):1602–1612
34. Vogiatzis C, Walteros JL, Pardalos PM (2013) Evacuation through clustering techniques. In: Goldengorin B., Kalyagin V., Pardalos P (eds) Models, algorithms, and technologies for network analysis. Springer Proceedings in Mathematics and Statistics, vol 32. Springer, New York, NY
35. Zheng QP, Arulselvan A (2011) Discrete time dynamic traffic assignment models and solution algorithm for managed lanes. J. Global Optim. 51(1):47–68