Journal of Institute of Science and Technology, 26(1), 63-73 (2021)
ISSN: 2469-9062 (print), 2467-9240 (e)
https://doi.org/10.3126/jist.v26i1.37826

Research Article

QUICKEST FLOW ALGORITHMS WITH TIME-VARYING ATTRIBUTES

Dipak Babu Amgain[1*], **Tanka Nath Dhamala**[1]

[1]*Central Department of Mathematics, Tribhuvan University, Kathmandu, Nepal*
**Correspondence author: diapkamgain@gmail.com*

*(**Received**: February 25, 2021; **Revised**: May 21, 2021; **Accepted**: May 30, 2021)*

ABSTRACT

In many real-world situations, there are numerous network optimization problems where the network attributes depend on time. In this paper, we consider single-source single-sink discrete-time dynamic network flow problems. We review some algorithms for the quickest flow problems in two environments (to the network attributes): time-invariant and time-variant. This paper mainly focuses on the existing algorithms for a later one. In literature, most of the authors have made their objectives to determine the earliest arrival time paths along which a given amount of flow can be sent in the minimum time. Evacuation is the most recent research area of network optimization, where quickest flow models allow the estimation of the minimum time required to bring a given number of evacuees to safety.

Keywords: Dynamic network, Optimization, Time-dependent, Quickest flow, Pseudo-polynomial algorithm.

INTRODUCTION

For the first time, Ford and Fulkerson (1962) considered the maximum dynamic flow problem in a discrete-time dynamic network with constant attributes and solved it in polynomial time. They considered a dynamic network into a time-expanded network as a static version of a dynamic network. A time-expanded network is a network constructed with copies of each node of an original network at specified periods over some time horizon, where arcs connect nodes in different periods according to the traversal times in the original network. Halpern (1979) and Anderson *et al.* (1989) considered the network with time-varying capacities and constant arc travel time functions. Cai *et al.* (2001) solved the minimum cost flow problem as well as the universal maximum flow problem by considering the network with time-dependent attributes and gave pseudo-polynomial time algorithms. They discussed both problems with no-waiting, finite waiting, and infinite waiting capacities at all nodes in a non-first-in-first-out (FIFO) network with discrete periods. The static shortest path problem is polynomially solvable (Ahuja *et al.*, 1993) but the time-dependent shortest path problem is NP-hard (Cai *et al.*, 2001).

The quickest flow problem (QFP) has historically been defined on a discrete-time dynamic network as a network clearing problem (Burkard *et al.*, 1993). They showed that the QFP can be reduced to maximum dynamic flow problems via search techniques such as binary search and Newton’s search. They proposed several polynomial time and strongly polynomial-time algorithms for the quickest flow problem constant attributes (QFP-CA). Lin and Jaillet (2015) solved the QFP-CA with integer arc costs (travel time) based on the cost scaling approach of Goldberg and Tarjan (1990) in the same time bound as one of their cost scaling minimum cost flow algorithms. Saho and Shigino (2017) also proved that a strongly polynomial bound is possible by making use of a different cost scaling minimum cost flow algorithms: Cancel-and-Tighten of Goldberg and Tarjan (1989). These works can be useful in evacuation planning (Nath *et al.*, 2020; Pyakurel *et al.*, 2018; Adhikari & Dhamala, 2020). A contra-flow approach can be used to solve evacuation planning problem in a continuous time dynamic network (Pyakurel & Dhamala, 2016).

Tjandra (2003) studied QFP with time-dependent attributes (QFP-TDA) in two environments: supply does not change over time and supply changes over time. To adjust the initial supply contents, he suggested a modification of the original network by introducing a dummy node (super source) and a dummy arc with zero travel time and arc capacity same as the supply value according to the supply time. He also solved the problem for finite and infinite waiting at nodes in pseudo-polynomial time. Similarly, Miller-Hooks & Patterson (2004) also proposed a pseudo-polynomial time algorithm called TDQFP algorithm to solve the problem QFP-TDA when infinite waiting is permitted at all nodes in a non-FIFO network with discrete time intervals while solving QFP-TDA, Miller-Hooks and Patterson (2004), and Tjandra (2003) aimed to determine the earliest arrival time paths along which a given supply can be sent such that the last unit of flow arrives at the sink in a minimum time. The authors also suggested considering QFP-TDA as a time-dependent minimum time flow problem to solve in a time-expanded network and suggested applying the successive shortest path procedure for the solution. Dhundia (2005) modified the TDQFP algorithm for zero waiting at intermediate nodes by using a re-optimization technique (a heuristic approach). Opasanon (2005) extended the TDQFP algorithm for solving QFP-TDA in deterministic, time-varying networks for use in stochastic environments. That is, it iteratively determines the maximum probability paths from source to sink in a residual network and incrementally pushes flow along the paths until all demand is fulfilled. The network can also be studied with inflow-dependent transit time (Dhamala *et al.*, 2018; Khanal *et al.*, 2018). Pundir *et al.* (2015) presented an efficient algorithm for solving the linear bottleneck assignment problem. Several researchers have been carried out in production and logistic network independently (Yu & Thapa, 2014).

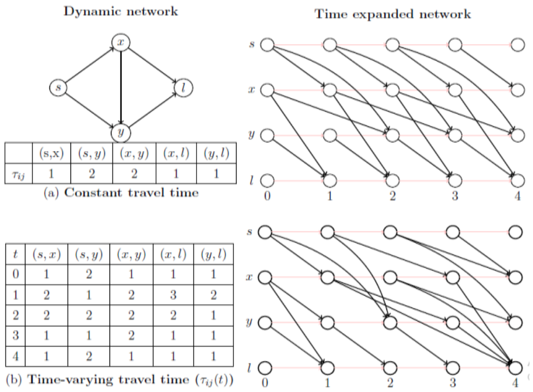
In this paper, we consider the attributes and parameters as integral valued functions and follow the model of Tjandra (2003) herein.

Preliminaries

A discrete-time dynamic network is a dynamic network, where time is divided into discrete units, flow moves through the network over time, capacity restricts the flow rate on an arc, and travel time determines how long each unit of flow can spend traversing an arc. Let $N = (V, A, b, \tau, u, s, l, T)$ be a discrete-time dynamic network with time horizon T, where V is the set of vertices (nodes), A the set of arcs, τ the arc travel time function, u the arc capacity function, s and l be a source and a sink. Let $n =$ number of nodes and $m =$ number of arcs. For each node $i \in V$ and time $t \in T = \{0, 1, \dots, T\}$, we denote supply contents, waiting times, and waiting capacities by $b_i(t)$, $\tau_{ii}(t)$ and $u_{ii}(t)$ respectively. Let $A_i^- = \{j: (j, i) \in A\}$ and $A_i^+ = \{j: (i, j) \in A\}$ be the sets of predecessors and successors of node i.

Motivation

In Fig. 1(a), we can see that there is a regular pattern of the arcs because of constant arc travel time. Many polynomial-time algorithms have been developed for dynamic network flow problems of this model. But in Fig. 1(b), a regular pattern of the arcs was not observed because of the time-varying nature of the arc travel time function.



	(s,x)	(s,y)	(x,y)	(x,l)	(y,l)
τ_{ij}	1	2	2	1	1

(a) Constant travel time

t	(s,x)	(s,y)	(x,y)	(x,l)	(y,l)
0	1	2	1	1	1
1	2	1	2	3	2
2	2	2	2	2	1
3	1	1	2	1	1
4	1	2	1	1	1

(b) Time-varying travel time ($\tau_{ij}(t)$)

Fig. 1. Dynamic network in different environments

There is no efficient algorithm for this model, although there have been a lot of research works in this area. According to Tjandra (2003), discrete-time dynamic flow on N can be considered as a function $x: A \times T \to \mathbb{R}^+ \cup \{0\}$ given by $x\big((i, j), t\big) = x_{ij}(t)\ \forall\ (i, j) \in A\ \forall\ t \in T$ satisfying the following conditions (1) - (4).

$$\sum_{t=0}^{T} \sum_{j\in A_s^+} x_{sj}(t) = \sum_{t=0}^{T} \sum_{j\in A_l^-} \sum_{\{t^*|t^*+\tau_{ij}(t^*)=t\}} x_{jl}(t^*) \qquad (1)$$

$$\sum_{j\in A_i^-} \sum_{\{t^*|t^*+\tau_{ij}(t^*)=t\}} x_{ji}(t^*) - \sum_{\{j|j\in A_i^+, t+\tau_{ij}(t)\leq T\}} x_{ij}(t)$$

$$= x_{ii}(t) - x_{ii}(t-1)\ \forall\, i \in V \setminus \{s, l\}\ \forall\, t \in T \qquad (2)$$

$$0 \leq x_{ij}(t) \leq u_{ij}(t)\ \forall\ (i, j) \in A, t \in$$

$$\{t' \mid t' \in T, t' + \tau_{ij}(t') \leq T\} \qquad (3)$$

$$0 \leq x_i(t) \leq u_i(t)\ \forall\, i \in V, t \in T \qquad (4)$$

Without loss of generality, we may suppose that $x_{ij}(t) = 0 = x_i(t)$ (waiting flow) for $t < 0$ and $t + \tau_{ij}(t) > T$ for convenience. Here, (1) represents that outflow at the source node should be equal to inflow at sink node, (2) is flow conservation law and (3) and (4) are capacity constraints for arcs and nodes respectively. Here, we suppose that lower bounds for capacity functions (arc capacity and waiting capacity) are zero. If waiting is not allowed at its intermediate nodes, the value on the right-hand side of (2) should be zero.

If a network was taken without a time parameter and travel time function as a distance function, the network becomes a static network.

Let $t \in T$, the flow that can be sent to the sink node l from its predecessors at time t is called terminal flow. For $t \in T$, define a sum function of terminal flows,

$$F_t = \sum_{j\in A_l^-} \sum_{\{t^*|t^*+\tau_{ij}(t^*)=t\}} x_{jl}(t^*)$$

a) **Maximum dynamic flow problem:** For a given time horizon T, a discrete-time maximum dynamic flow problem is to maximize the total flow value $\sum_{t=0}^{T} F_t$ for the constraints (1) - (4).

b) **Earliest arrival flow problem:** For a given time horizon T, a discrete-time earliest arrival flow problem is to find the total maximum flow value $\sum_{t=0}^{\theta} F_t$ for each period $\theta \leq T$ concerning (1) - (4).

c) **Minimum weighted sum flow problem:** For a given time horizon T, let c_t be the weight (or cost) increasing function over the time t associated with F_t for each $t \in T$. Then a discrete-time minimum weighted sum flow problem is to minimize the total weighted sum flow $\sum_{t=0}^{T} c_t F_t$ with respect to (1) - (4).

d) **Quickest flow problem:** For a given supply value $v = \sum_{t=0}^{T} b_s(t)$, the quickest flow problem determines a $T_v = \min\{T^* \mid T^* \leq T, \sum_{t=0}^{T^*} F_t \geq v\}$ with respect to the constraints (1) - (4).

Consider the value of earliest arrival flow at each period $\theta \in [0, T]$ as

$$V_\theta = \max \sum_{t=0}^{\theta} F_t$$

Then V_θ is the value of maximum dynamic flow in the period $[0, \theta]$ for $\theta \in T$.

FIFO Property

An arc $(i, j) \in A$ is said to have a FIFO (first-in-first-out) property if

$$t_1 \leq t_2 \Longrightarrow a_{ij}(t_1) \leq a_{ij}(t_2) \qquad (5)$$

where $a_{ij}(t) = t + \tau_{ij}(t)$ is the arc arrival time function for node j at time t. If every arc in a network has a FIFO property, the network is known as a FIFO network. In the FIFO network, the arc arrival time function is non-decreasing over the period. In a time-dependent network, there may not be a FIFO property because of time-varying attributes. So, waiting at a node may give an earliest arrival time path even for the late starting time (Fig. 2).

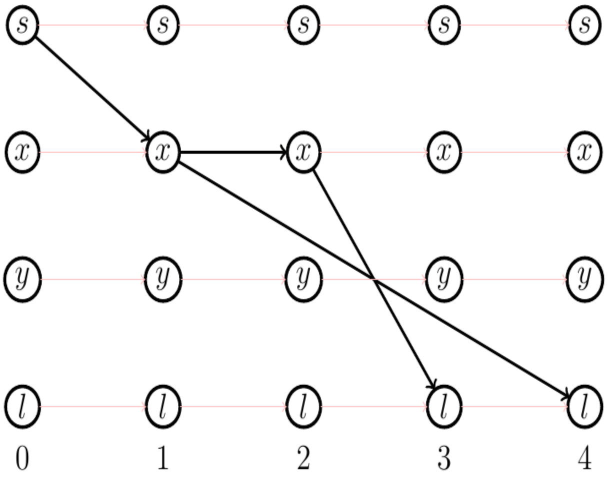


Fig. 2. Earliest arrival time path with waiting at x

Dynamic augmenting path

Let G(x) be a residual network for an augmenting flow x. A dynamic augmenting path is a dynamic s-l path $P(t_0)$ in G(x) composed of a sequence of time-arc pairs $t_k(j_k, j_{k+1})$ with departure time t_k for arc (j_k, j_{k+1}) from node s to node l that ready at node s at the time $t_0 \in T$, as given by

$$P(t_0) = \{t_0(j_0, j_1), t_1(j_1, j_2), \ldots, t_{r-1}(j_{r-1}, j_r)\},$$

where $j_0 = s, j_r = l, t_{k+1} = t_k + \tau_{j_k j_{k+1}} \in T$ for $k = 1, 2, \ldots, r-1$.

MATERIALS AND METHODS

It is an algorithmic study of a graph-theoretic model, where the correctness of the algorithms is proven through lemmas, theorems, and propositions and verified with examples by using computational programming with computer implementation.

Generally, we have three approaches to solve time-dependent network flow problems.

- To create a so-called time expanded network as a static version of a dynamic network and use algorithms for static network,
- To reduce it into a static one and use existing algorithms,
- To explore only the time-dependent property of network attributes without doing time expansion.

The main contribution of this paper is based on a review and discussion of the previously published documents.

RESULTS

The QFP is closely related to the maximum dynamic flow problem in the sense that it is to send a given flow v from a given source to a given sink in the shortest possible time T^*, whereas the maximum dynamic flow problem sends maximum units of flow $\sum_{t=0}^{T} F_t$ from a given source s to a given sink l within the given time horizon T.

Ford and Fulkerson (1962) introduced temporally repeated flow as a special type of dynamic flow that turns any feasible static flow into a feasible dynamic flow. If x_{stat} be a feasible static flow and T a given time horizon, x_{stat} can be decomposed into the set P of path flows (paths P_k's with their flows $val(P_k)$) which is called flow decomposition. A temporally repeated flow is a dynamic flow obtained by temporarily repeating all paths flows of some path decomposition. Let $\tau(P_k)$ be the length of P_k with respect to the travel time. Mathematically, the value of temporally repeated flow, denoted by val(P) can be defined as

$$val(P) = \sum_{P_k \in \mathbf{P}} val(P_k)\big(T + 1 - \tau(P_k)\big)$$

Lemma 1 (Ford & Fulkerson, 1962)

For a maximum dynamic flow problem, there exists an optimal solution in the temporally repeated form.

Lemma 1 is the direct consequence of the definition of temporally repeated flow because of flow decomposition.

Theorem 1 [Triple Optimization Theorem] (Jarvis & Ratliff, 1982).

Consider the above three problems b), c) and d) under the assumption that there exists a feasible flow of v units (i.e.,

$V_T \geq v$). Then the solution for either problem b) or c) is also the solution of the other two problems.

Proof: [b) $\Leftrightarrow$ c)] If it is possible to get a feasible flow of v units from s to l, then there exists an earliest arrival flow which equals v (Minieka, 1973). Suppose that there exists a set of terminal flows F_t^* for $t \in T$ satisfying b) and hence $\sum_{t=0}^{T} F_t^* = v$ and $\sum_{t=0}^{\theta} F_t^* \geq \sum_{t=0}^{\theta} F_t$ for $\theta \in T$. Since, $\sum_{t=0}^{T} F_t^* = \sum_{t=0}^{T} F_t$, if there exist $t_k \in T$ such that $F_{t_k}^* \neq F_{t_k}$then $\sum_{t=0}^{T} c_t F_t^* < \sum_{t=0}^{T} c_t F_t$ (Jarvis and Ratlift, 1982). This shows that F_t does not correspond to a solution of c). Hence F_t and F_t^* must be equal for all $t \in T$. This implies that b) and c) are equivalent.

[b) or c) $\Rightarrow$ d)] As b) and c) are equivalent, we show that a solution of b) can be used d). Let r be the largest index such that $F_r^* > 0$. On the contrary, if the solution of b) does not solve d) then there exist some $p < r$ such that $\sum_{t=0}^{p} F_t \geq v$. This contradicts that $\sum_{t=0}^{r} F_t^*$ is the maximum flow in the first r period.

Quickest flow problem with constant attributes

Here, the attributes are considered as constant functions over time t (e.g.,$u_{ij}(t) = u_{ij}, \tau_{ij}(t) = \tau_{ij}, x_{ij}(t) = x_{ij}$, $b_s(t) = b_s$) Ford and Fulkerson (1962). In our consideration, a network with constant attributes is a FIFO network and in a FIFO network, waiting at nodes is never beneficial.

Based on the search techniques and maximum dynamic flow algorithms, the authors developed several polynomial-time algorithms by considering a discrete-time dynamic network with nonnegative integer-valued arc capacity and arc travel time functions. Burkard *et al.* (1993) made a close relationship between the maximum flow problem and the quickest flow problem in the following lemma.

Lemma 2

Let x be a dynamic flow of value v in the period $[0, \theta], \theta \in T$. If $V_{\theta-1} < v$ then x is the quickest flow of value v. Moreover, $T_v = \theta$.

Lemma 1 implies that finding an optimal solution to the quickest flow problem with a given amount of flow v is equivalent to finding the minimum time T^* such that $V_{T^*} \geq v$. Let $\delta_\theta = V_\theta - V_{\theta-1}$ and $\lambda = \min\{\theta | \theta \in T,\ V_\theta > 0\}$ (length of the shortest s-l path with respect to the travel times). Suppose that $|x_{stat}|$ is the value of an arbitrary maximum static flow x_{stat} in the network N.

Theorem 2 (Burkard *et al.*, 1993)

(i) V_θ is nondecreasing as time θ increases and for $\theta \geq \lambda$ it is strictly increasing.

(ii) δ_θ is nondecreasing. That is, $\forall\ \theta > 0\ \delta_\theta \leq \delta_{\theta+1}$

(iii) δ_θ can attain only values from the set $\{0, 1, \ldots, |x_{stat}|\}$.

From Theorem 2, V_θ is non-decreasing and θ attains only integer values. One can use a binary search method over $\theta \in [\theta_1, \theta_2]$ such that $T_v \in [\theta_1, \theta_2]$ and solve maximum dynamic flow problem in each iteration, until T_v is found. While applying a pure binary search technique, the efficiency can be improved by applying the Newton's like method and regula falsi to obtain the bounds. A new example of this approach can be found in (Bhandari & Dhamala, 2020). Using an improved binary search technique the QFP can be solved in $O(\min(\log v, |x_{stat}|) \cdot MCF(m, n))$ (Burkard *et al.*, 1993), where $MCF(m, n)$ is the time complexity for solving one minimum cost flow problem with n nodes and m arcs.

Newton's method can also be used to solve the QFP with constant attributes with an initial guess θ_1 for T_v. Newton's method needs at most $O(\log v + \log|x_{stat}|)$ time (Burkard *et al.*, 1993).

In conclusion, we can state a theorem as below.

Theorem 3 (Burkard *et al.*, 1993)

The QFP-CA can be solved polynomially.

Lemma 3 (Lin & Jaillet, 2015)

The QFP-CA can be formulated as the fractional programming problem as below.

$$T_v = \min \frac{v + \sum_{(i,j)\in A} \tau_{ij} x_{ij}}{|x_{stat}|}$$

Subject to

$$\sum_{j\in A_i^+} x_{ij} - \sum_{k\in A_i^-} x_{ki} = \begin{cases} |x_{stat}| & i = s \\ -|x_{stat}| & i = l \\ 0 & \text{otherwise} \end{cases}$$

$$0 \leq x_{ij} \leq u_{ij}\ \forall\ (i, j) \in A$$

Keeping the idea of Lemma 2 in mind, the formulation in Lemma 3 can be developed by using the temporally repeated form of the maximum dynamic flow model. For detailed proof, we refer to (Lin & Jaillet, 2015).

Lemma 4 (Lin & Jaillet, 2015)

The QFP with constant attributes can be solved in $O\left(nm \log\left(\frac{n^2}{m}\right) \log(nC)\right)$ time, where C is the maximum arc cost (travel time).

Proof: The QFP with constant attributes can be solved by applying the cost scaling algorithm of Goldberg and Tarjan (1990) for minimum cost circulations. The time complexity of the algorithm is $O\left(nm \log\left(\frac{n^2}{m}\right) \log(nC)\right)$. For more detailed proof see (Lin & Jaillet, 2015).

Lemma 5 (Saho & Shigeno, 2017)

The QFP with constant attributes and integer arc costs can be solved in strongly polynomial time.

Proof: The weakly polynomial-time algorithm of Lin and Jaillet (2014) can be improved with strongly polynomial time (i.e. $O(nm^2(\log n)^2)$) bound for the solution for the QFP by Cancel-and-Tighten algorithm of Goldberg and Tarjan (1990) for min-cost flow problem.

Quickest flow problem with time-dependent attributes

Here, we consider the QFP-TDA, where capacity, travel time, and supply value may change over time. Before discussing the algorithms for this problem, certain assumptions have been made as follows.

- Each arc has a non-negative time-varying capacity.
- Residual network permits negative travel time.
- Demand will equal the total supply.
- When flow arrives before time T, it simply waits without penalty until time T in order to satisfy the demand.
- At each node $\in V$, and for every time $t \in T$, there exists a holdover arc (i, i) with waiting capacity $u_i(t)$ and unit waiting time $\tau_i(t)$.
- When flow arrives at the sink node before the time T, it simply waits without penalty until time T in order to satisfy the demand.
- Once the flow has been sent, the time-dependent residual network is updated (i.e. capacities and travel times of all affected arcs in the network at the appropriate departure periods should be updated).
- Backward arcs are introduced to permit the return of capacity to an arc, corresponding to reversing decisions from previous iterations as a means for improving the objective.

Time-dependent residual network: As given in (Tjandra, 2003), the time-dependent residual network to a given feasible dynamic flow x is defined as $G(x) = (V, A_x, T)$ with $A_x = A_x^+ \cup$Where, $A_x^+ = \{(i,j) | (i,j) \in A, \exists\, t \leq T - \tau_{ij}(t) \text{ s.t. } u_{ij}(t) > x_{ij}(t)$ and $A_x^+ = \{(i,j) | (j,i) \in A, \exists\, t \leq T - \tau_{ji}(t) \text{ s.t. } x_{ji}(t) > 0$.

The attributes of the time-dependent residual network are as follows.

- Residual arc travel time

$$\tau_{ij}^x(t) = \begin{cases} \tau_{ij}(t), & (i,j) \in A, t + \tau_{ij}(t) \leq T \\ -\tau_{ji}(t), & (j,i) \in A, t' + \tau_{ij}(t') = t \leq T, x_{ji}(t') > 0 \end{cases}$$

- Residual arc capacity

$$u_{ij}^x(t) = \begin{cases} u_{ij}(t) - x_{ij}(t), & (i,j) \in A, t + \tau_{ij}(t) \leq T \\ x_{ji}(t), & (j,i) \in A, t' + \tau_{ij}(t') = t \leq T \end{cases}$$

- Residual waiting capacity

$$u_i^{x+}(t) = u_i(t) - x_i(t), i \in V, t \leq T$$
$$u_i^{x-}(t) = x_i(t), i \in V, t < T$$

Tjandra (2003) developed a successive earliest arrival augmenting path (EAAP) algorithm to solve an earliest arrival flow problem with time-varying attributes. The author showed that by allowing infinite waiting at nodes, the EAAP algorithm is more efficient by factor T than implementing the successive shortest path algorithm on the time-expanded network. He also applied the EAAP algorithm to solve QFP-TDA.

EAAP Algorithm (Tjandra, 2003)

```
Input:   Residual dynamic network $G(x) = (V, A_x, T)$ of a
         flow x
Output:  An EAAP path P with $val(P)$
Step 0:  Initialization $SE = \{s\}$ and set the labels for each
$i \in V$
             $\pi_i = \begin{cases} 0, & i = s \\ \infty, & \text{otherwise} \end{cases}$ , $\forall\, t \in \{0,1,\dots,T\}$
             $pred_i(t) = \begin{cases} 0, & i = s \\ \infty, & \text{otherwise} \end{cases}$
             $dep_i(t) = \begin{cases} t, & i = s \\ \infty, & \text{otherwise} \end{cases}$
Step 1:  Select current node
         If $SE = \emptyset$, go to step 3
         Otherwise, select $i \in SE$ and set $SE = SE - \{i\}$
Step 2:  Scan the current node and update the labels
         For all $(i,j) \in A_x$ do
         {For all
$t \in \{t | \pi_i \leq t \leq T, u_{ij}(t) > 0, t + \tau_{ij}(t) \leq T\}$ do
             { if ($pred_i(t + \tau_{ij}(t)) = \infty$) then
             { $\pi_j = t + \tau_{ij}(t), t' := \pi_j + 1$
              $pred_j(\pi_j) := i$ ; $dep_j(\pi_j) := t$
              $SE: = SE + \{j\}$
              While ($t' \leq T$) and ($pred_j(t') := \infty$) do
                 { $pred_j(t') = j$ ; $dep_j(t') := t' - 1; t' + +$}
                Define $t' := t + \tau_{ij}(t) - 1$
              While ($t' \geq 0, u_i^{x-}(t' + 1) > 0, pred_j(t') = \infty$)
```

```
                do { if $(\pi_j > t')$ then $\pi_j := t'$
                 $\mathrm{pred}_j(t') = j\,;\mathrm{dep}_j(t') := t' + 1\,;t' - -;$
            End while;
        End while
    End for;
Return to step 1
```

Step 3: Constructing an s-l earliest arrival augmenting path

```
    If $\pi_d > T$ then $P := \emptyset$ and $\mathrm{val}(P) = 0;$
    Else { $j := d;\ t := \pi_j; i := \mathrm{pred}_j(t);$
    Define $P := \{l(t, t)\}$ and $\mathrm{val}(P) := \infty$
    While $(j \neq 0)\{t' := \mathrm{dep}_j(t)$ do
        If $(i \neq j)$ then { $cap := u_{ij}^{x}(t')\,; t'' := t'\}$
        Else if $(t > t')$, then $cap := u_i^{x+}(t');$
        Else $cap := u_i^{x+}(t');$
      If $\left(cap < val(P)\right)$ then $\mathrm{val}(P) := cap\ i' := i;$
      $j := i\,; i := \mathrm{pred}_j(t'); t := t'$
      If $(i' \neq i)$ then $:= P + \{i'(t', t'')\};$
    End while;
```

Proposition 1 (Tjandra, 2003)

The EAAP algorithm finds the earliest arrival augmenting path in $O(mnT)$ for infinite waiting.

To apply the EAAP algorithm for QFP-TDA, Tjandra (2003) suggested adjusting the supply value by adding a super source s^* and a dummy arc (s^*, s) with the properties.

$\tau_{s^*s}(t) = 0, u_{s^*s}(t) = b_s(t), u_{s^*s^*}(t) = 0\ \forall\ t \in T.$

Procedure overview of Tjandra's algorithm

1. Modify the original network by adding a dummy source and dummy arc to adjust the supply contents.
2. Set the time horizon T large enough so that the problem is feasible.
3. Find the earliest arrival flow by the EAAP algorithm.
4. Update the residual network of the current flow and augment the flow.
5. Repeat the process from step 3 until $\sum_{t=0}^{\theta} F_t \geq v$ for $\theta \leq T$

Theorem 4 (Tjandra, 2003)

If x^* is the earliest arrival flow for a time horizon T with a value V_T and $T^* = \min\{T' | V_{T'} \geq b_s(0)\}$ exists, then T^* solves the discrete-time QFP with initial supply content $b_s(0)$.

Proof of Theorem 4 is directly followed from the Triple Optimization Theorem.

Proposition 2 (Tjandra, 2003)

The time complexity of Tjandra's algorithm is $O(vmnT^2)$ for finite waiting and $O(vmnT)$ for infinite waiting at every node of the network.

Proof: From the proposition 1 we have the worst case time complexity for finding an EAAP as $O(mnT)$. In the worst case, only one unit of flow might be sent at each timing path. So, the total time complexity of Tjandra's algorithm is $O(vmnT)$

Miller-Hooks and Patterson (2004) solved the QFP-TDA with the objective

$$\min \left[\max_{\substack{i \in A_l^- \\ t \in \mathbf{T}}} \left(t + \tau_{il}(t)\right) I_{il}(t) \right],$$

Where $I_{il}(t) = \begin{cases} 1, & x_{il}(t) > 0 \\ 0, & \text{otherwise} \end{cases}$ and capacity constraints be restated as $x_{il}(t) \leq u_{il}(t) I_{il}(t)\ \forall\ i \in A_l^-\ , t \in T$

While solving the QFP-TDA, we may set the time horizon T large enough so that the problem is feasible. However, Miller-Hooks and Patterson (2004) proposed an idea to set a time horizon for QFP-TDA, which is given as follows.

$$T = t^{stat} + \pi_l^{stat} + \left\lceil \frac{v}{\pi_l^{stat}} \right\rceil \tag{6}$$

where t^{stat} returns the time at which travel conditions become stationary, π_l^{stat} returns the shortest path time between the origin and destination using the arc travel times at t^{stat}, and κ_l^{stat} returns the corresponding capacity of this shortest path.

Let $e(t)$ be the excess (supply value at s) at time t. Suppose that $\hat{t} = \min\{t \mid e(t) > 0\}$.

TDEAT Algorithm (Miller-Hooks & Patterson, 2004)

Input: $(G(x), s, \hat{t}, T)$

Output: $\pi_l,\ \kappa_l,\ P_l$

Step 0: Initialization $SE = \{s\}$ and set the labels for each $i \in V$

$$\pi_i = \begin{cases} \hat{t}, & i = s \\ T, & \text{otherwise} \end{cases},$$

For all $t \in \{0,1, \dots, T\}$

$$P_i(t) = \begin{cases} 0, & i = s, t = \hat{t} \\ s, & i = s, t \neq \hat{t} \\ \infty, & \text{otherwise} \end{cases}, \ U_j(t) = \infty$$

$$D_i(t) = \begin{cases} \hat{t}, & i = s, t = \hat{t} \\ t-1, & i = s, \hat{t} < t \leq T \\ \infty, & \text{otherwise} \end{cases}$$

```
Step 1:  Select current node
        If $SE = \emptyset$, go to step 3;
        Otherwise, select $i \in SE$ and set $SE = SE - \{i\}$
Step 2:  Scan the current node and update the labels
For all $j \in A_x^+(i)$ $I_{ij} = \begin{cases} 0, & (i,j) \in A \\ 1, & \text{otherwise} \end{cases}$
    $T_{ij} = \{t \mid u_{ij}^x(t) > 0, t \geq \pi_i, t + \tau_{ij}(t) \leq T\}$;
        For all $t \in T_{ij} \cap \{t \mid t + \tau_{ij}(t) I_{ij} \leq \pi_j\}$;
            If ($P_i(t + \tau_{ij}(t)) = \infty$) then
            $\pi_j = t + \tau_{ij}(t); t' = \pi_j + 1$;
            $P_j(t + \tau_{ij}(t)) := i; D_j(\pi_j) := t$
            $U_j(\pi_j) = \min\{U_i(t), u_{ij}^x(t)\}$;
            While ($t' \leq T$) and ($P_j(t') := \infty$);
                $P_j(t') = j; D_j(t') := t' - 1$;
                $U_j(t') = U_j(t' - 1); t' + +$;
            End while;
            End if;
        End for;
    End for;
    Return to step 1
Step 3:  Constructing an earliest arrival time path
        For all $i \in V$;
        $\kappa_i = U_i(\pi_i); P_i := \emptyset; j = i; d = \pi_i$
        While ($j \neq s$)
            $d' = D_j(d)$;
            $P_i = \{(P_j(d), j), d'\} \cup P_i$
            $j = P_j(d)$;
            $d = d'$;
        End while;
```

Proposition 3 (Miller-Hooks & Patterson, 2004)

The TDEAT algorithm terminates with the earliest arrival time paths from a given origin at a selected departure time to all other nodes in V, where waiting is permitted.

Procedure overview of TDQFP algorithm

1. Initialization.
2. A time-dependent earliest arrival time (TDEAT) algorithm is implemented to find the TDEAT path.
3. Find the augmenting flow and update its residual network.
4. If all the supply has reached the sink node, stop. Otherwise, go to step 2.

Proposition 4 (Miller-Hooks & Patterson, 2004)

The TDQFP algorithm can solve the QFP-TDA in $O(vn^2T)$.

Proof: Step 2 of the TDQFP algorithm can take $O(n^2T)$ time to obtain the TDEAT path in the worst-case. If 1 unit of flow can be sent at a time in step 3, we have to repeat v times. So, step 2 can be repeated at most v times. Hence, the total time complexity is $O(vn^2T)$.

Theorem 5 (Miller-Hooks & Patterson, 2004)

The solution of time-varying QFP for a single-source single-sink network can be extended to solve the time-varying QFP in a multi-source multi-sink network.

Proof: According to Miller-Hooks and Patterson (2004), we can divide its proof into two parts.

Converting multiple sources to a single super source: Let S the set of sources. We add a super source node s^* to the original network with dummy arcs $(s^*, i), i \in S$ such that $\tau_{s^*i}(t) = 0,\ u_{s^*i}(t) = b_i(t)\ \forall\, t \in \forall\, t \in T$. We also add a self-loop at s^* with infinite capacity and unit travel time for every period $t \in T$.

Converting multiple sinks to a single super sink: Let L be the set of given sinks. We add a super sink $l^*$and a copy j' (dummy sink) for each sink node $j \in L$ to the network. We also add new dummy arcs $(j, j')\ \forall\, j \in L$ to the network with infinite capacities and zero travel times for all $t \in T$. For every j', add a self-loop with infinite capacity and unit travel time for each $t \in T$. Now for each dummy sink j' of $j \in L$, add dummy arc (j', l^*) to the network with $u_{j'l^*}(T) = |b_j(T)|$ (absolute value of demand at $j \in L$) and zero travel time for all $t \in T$.

DISCUSSION

Based on network attributes, one may classify the quickest flow problem as the quickest flow problem with constant attributes (Burkard *et al.*, 1993; Lin & Jaillet, 2015; Saho & Shigeno, 2017) and quickest flow problem with time-dependent attributes (Tjandra, 2003; Miller-Hooks & Patterson, 2004).

The main contributions of Tjandra (2003) were pseudo-polynomial time algorithms for finite waiting capacities as well as for infinite waiting capacities at nodes in a time-dependent network. Similarly, Miller-Hooks and Patterson (2004) also developed a pseudo-polynomial time algorithm (the TDQFP algorithm) for solving the time-dependent quickest flow problem for infinite waiting capacities and that does not require an explicit space-time expansion. They showed how the algorithm can be used to solve both the time-dependent evacuation and the time-dependent quickest transshipment problems by adding a small number of nodes and arcs to the existing network. In both algorithms, they have maintained FIFO property using scan eligible (SE) list to determine the earliest arrival time at every node for departure at a given time from a single source node in the time-dependent network.

Example 1: Consider an example of a time-dependent network Fig. 3 for infinite waiting at every node and attributes in Table 1.

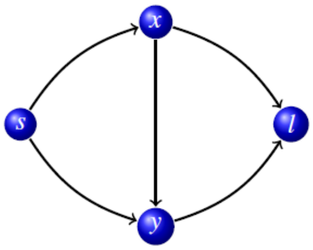


Fig. 3. A dynamic network

Table 1: $\tau_{ij}(t), u_{ij}(t)$

	(s, x)	(s, y)	(x, y)	(x, l)	(y, l)
0	1, 9	1, 8	1, 5	2, 8	1, 7
1	2, 9	1, 9	2, 11	1, 7	3, 11
2	2, 5	2, 3	1, 6	2, 5	2, 4
3	1, 5	2, 5	1, 8	2, 5	2, 5
4	1, 9	1, 8	2, 9	1, 7	2, 12
5	2, 2	1, 4	1, 5	1, 4	1, 1

Supply contents, $b_s(t) = \begin{cases} 21 & \text{if } t = 0 \\ 4 & \text{if } t = 1 \\ 0 & \text{otherwise} \end{cases}$

It introduces super source to adjust the supply contents and update the table of the attributes, as given in Table 2 and Fig. 4.

After implementing Tjandra's algorithm, we have found the quickest time $T^* = 4$ as a minimum time to clear 25 units of supply contents. The final solution can be shown in time-expanded network as given in Fig. 5.

From Fig. 6, $t^{stat} = 6, \pi_l^{stat} = 5, v = 7,$ and $\kappa_l = 3$ were for the path $P: s - y - l$. Using the formula given in equation (6) it gets $T = 14$. Implementing the TDQFP algorithm with $T = 14$, it gets the following results.

Table 2. $\tau_{ij}(t), u_{ij}(t)$

	(s*, s)	(s, x)	(s, y)	(x, y)	(x, l)	(y, l)
0	0, 21	1, 9	1, 8	1, 5	2, 8	1, 7
1	1, 4	2, 9	1, 9	2, 11	1, 7	3, 11
2	0, 0	2, 5	2, 3	1, 6	2, 5	2, 4
3	0, 0	1, 5	2, 5	1, 8	2, 5	2, 5
4	0, 0	1, 9	1, 8	2, 9	1, 7	2, 12
5	0, 0	2, 2	1, 4	1, 5	1, 4	1, 1

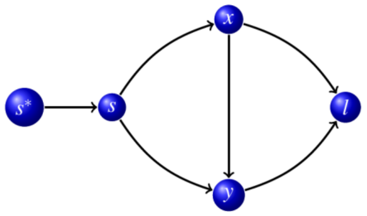


Fig. 4. A dynamic network

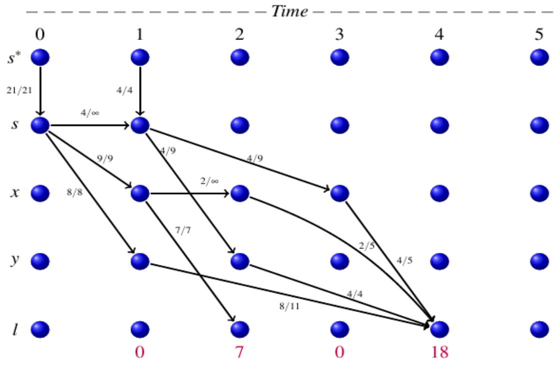


Fig. 5. The final representation of the solution

Example 2: Consider a dynamic network with time-varying attributes, as given in Table 3 for TDQFP algorithm.

Table 3. Piecewise defined constant attributes

(i,j)	(s,x)	(s,y)	(x,y)	(x,l)	(y,l)
$u_{ij}(t)$	2, t ≤ 2 3, t ≥ 3	3	1, t ≤ 3 2, t ≥ 4	2, t ≤ 2 3, t = 3 2, t ≥ 4	4, t ≤ 1 3, t ≥ 2
$\tau_{ij}(t)$	3, t ≤ 1 2, t ≥ 2	2, t = 0 3, t = 1 3, t ≥ 2	1	5, t ≤ 2 3, t = 3 4, t ≥ 4	3, t ≤ 3 4, t = 4 2, t ≥ 5

Let excess supply at the source s at time t be $e(t)$. Since $e(0) = 5 > 0$, starting time $\hat{t} = 0$. The TDEAT algorithm gives an earliest arrival time path $P_1: s - y - l$ with path capacity $\kappa_l = 3$, augmenting flow $\epsilon = min\{e(\hat{t}), \kappa_l\} = 3$. It takes 5 units of time to reach the sink node l from the source node s at time 0.

$$b_s(t) = \begin{cases} 5, & t = 0 \\ 2, & t = 1 \\ 0, & otherwise \end{cases}$$

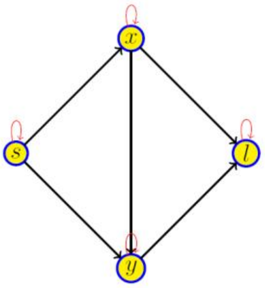


Fig. 6. A dynamic network with time-varying supply content

Update excess $e(\hat{t}) = e(\hat{t}) - \epsilon = 2$. Update the residual capacities $x(t)$ and travel times of the arcs of the path P_1 at the appropriate departure periods in the residual network as in Fig. 7.

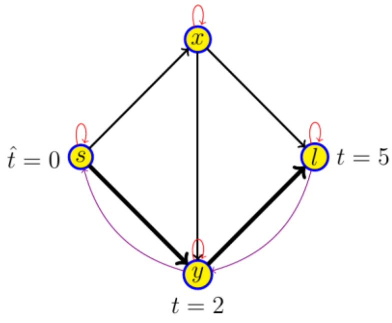


Fig. 7. The first iteration of TDQFP algorithm

Again at time $\hat{t} = 0$, excess $e(\hat{t}) = 2$. So, applying the TDEAT algorithm we get another earliest arrival time path $P_2: s - z - y - x - l$ with path capacity $\kappa_l = 2$, augmenting flow $\epsilon = min\{e(\hat{t}), \kappa_l\} = 2$. And it takes 6 units of time to clear 2 units of supply content from s to l along the path P_2.

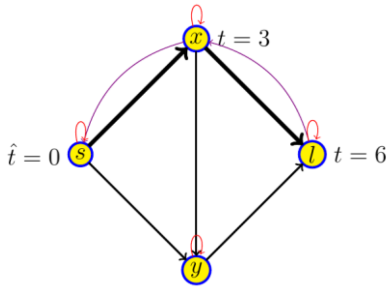


Fig. 8. A second iteration of TDQFP algorithm

Now, update excess $e(\hat{t}) = e(\hat{t}) - \epsilon = 0$. Update the capacities and travel times of the arcs of the path P_2 at the respective departure periods in the residual network as in Fig. 8. But at time $\hat{t} = 0$, excess $e(\hat{t}) = 0$. So, we have to check excess value for $\hat{t} = 1$ excess $e(\hat{t}) = 0$. Again, for $\hat{t} = 1$ excess $e(\hat{t}) = 2$. Now, applying the TDEAT algorithm we have another earliest arrival time path $P_3: s - y - l$ with waiting at y path capacity $\kappa_l = 3$, augmenting flow $\epsilon = min\{e(\hat{t}), \kappa_l\} = 2$. We can send this flow to the sink node l along the path P_3 at time 7. The updated residual network is given in Fig. 9.

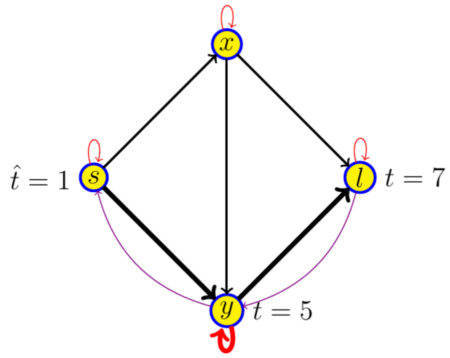


Fig. 9. The third iteration of TDQFP algorithm

Now, the demand has been fulfilled. After updating the final residual arc capacities and travel times, we may have the residual arc capacities and arc travel time as in Table 4.

Table 4. Final table of example 2

(i,j)	(s,x)	(s,y)	(x,y)	(x,l)	(y,l)
$u_{ij}(t)$	0, t=0 2, 1≤ t≤ 2 3, t ≥ 3	0, t = 0 1, t = 1 3, t ≥ 2	1, t ≤ 3 2, t ≥ 4	2, t ≤ 2 3, t = 3 0, t = 4 2, t ≥ 5	4, t ≤ 1 0, t =2 3, 3 ≤ t ≤ 4 1, t = 5 3, t ≥ 6
$\tau_{ij}(t)$	3, t ≤ 1 2, t ≥ 2	2, t = 0 3, t = 1 3, t ≥ 2	1	5, t ≤ 2 3, t = 3 4, t ≥ 4	3, t ≤ 3 4, t=4 2, t≥5

(i,j)	(x,s)	(y,s)	(l,x)	(l,y)
$u_{ij}(t)$	0, t ≤ 2 2, t = 3 0, t ≥ 4	0, t ≤ 1 3, t = 2 2, t = 4 0, t ≥ 5	0, t ≤ 5 2, t = 6 0, t ≥ 7	0, t ≤ 4 3, t = 5 0, t = 6 2, t = 7 0, t ≥ 8
$\tau_{ij}(t)$	14, t ≤ 2 -2, t = 3 14, t ≥ 4	14, t ≤ 1 -3, t = 2 -3, t = 4 14, t ≥ 5	14, t ≤ 5 -4, t = 6 14, t ≥ 7	14, t ≤ 4 -2, t = 5 14, t = 6 -2, t = 7 14, t ≥ 8

CONCLUSION

In this paper, variant models and different algorithmic approaches were investigated to solve the time-varying quickest flow problem in a single source and single sink discrete-time dynamic network. The studied algorithms are pseudo-polynomial time algorithms. The algorithms used the SE list to maintain the FIFO nature while finding the earliest arrival paths in the time-varying network. The main reviewed papers were dedicated to determining the earliest arrival time paths along which a given amount of flow can be sent in the minimum time. In this paper, the performance of the algorithms was checked with different examples.

In the real world situation, the problems are time-varying. Most of the researchers in network optimization have chosen evacuation as an applied field of a time-dependent quickest flow problem. The existing algorithms for solving the time-varying quickest flow problem are not efficient. Therefore, if the QFP-TDA is solved efficiently then the evacuation network flow problem and quickest transshipment problem will be solved more accurately and efficiently in a time-dependent environment.

ACKNOWLEDGEMENT

The University Grant Commission, Nepal is highly acknowledged for granting the M. Phil. Fellowship (MPhil-76/77-S&T-01) to one of the authors (DBA) for conducting this work.

REFERENCES

Adhikari, I. M., & Dhamala, T. N. (2020). Minimum clearance time on the prioritized integrated evacuation network. *American Journal of Applied Mathematics, 8*(4), 207-215.

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. Princeton, New Jersey, USA: Prentice-Hall, Inc.

Anderson, E. J., Nash, P., & Philpott, A. B. (1989). A class of continuous network flow problems. *Mathematics of Operations Research, 7*, 479-635.

Bhandari, A. D., & Dhamala, T. N. (2020). On quickest flow problem using improved binary search algorithm. *International Journal of Innovative Science, Engineering & Technology*, *7*(12), 136-145.

Burkard, R., Dlaska, K., & Klinz, B. (1993). The quickest flow problem. *ZOR-Methods and Models of Operations Research, 37*, 31-58.

Cai, X., Sha, D., & Wong, C. K. (2001). Time-varying universal maximum flow problems. *Mathematical and Computer Modelling, 33*(4-5), 407-430.

Dhamala, T. N., Pyakurel, U., & Dempe, S. (2018). A critical survey on the efficient algorithms for evacuation planning problems. *International Journal of Operation Research, 3*, 101-133.

Dhundia, H. (2005). The quick time-dependent quickest flow problem: A lesson in zero-sum cycles (Master's Thesis), Department of Civil and Environmental Engineering, University of Maryland, College Park, USA.

Ford, L., & Fulkerson, D. R. (1962). Flows in network. Princeton, New Jersey, USA: Princeton University Press.

Goldberg, A. V., & Tarjan, R. E. (1989). Finding minimum-cost circulations by canceling negative cycles. *Journal of ACM*, *36*(4), 873-886.

Goldberg, A. V., & Tarjan, R. E. (1990). Finding minimum-cost circulations by successive approximation. *Mathematical Operation Research, 15*(3), 430-466.

Halpern, J. (1979). A generalized dynamic flows problem. *Networks*, 9, 133-167.

Jarvis, J. R., & Ratliff, D. H. (1982). Note-Some equivalent objectives for dynamic network flow problems. *Management Science*, *28*(1), 106-109.

Khanal, D., Pyakurel, U., & Dhamala, T. N. (2018). Flow overtime problem with inflow-dependent transit times. *Journal of Institute of Science and Technology*, *23*(1), 49-56.

Lin, M., & Jaillet, P. (2015). On the quickest flow problem in dynamic networks: a parametric min-cost flow approach. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 1343-1356).

Miller-Hooks, E., & Patterson, S. (2004). On solving quickest time problems in time-dependent, dynamic networks. *Journal of Mathematical Modelling and Algorithms*, *3*, 39-71.

Minieka, E. (1973). Maximal, lexicographic, and dynamic network flows. *Operation Research*, *21*(2), 517-527.

Nath, H. N., Pyakurel, U., Dhamala, T. N., & Dempe, S. (2020). Dynamic network flow location models and algorithms for quickest evacuation planning. *Journal of Industrial and Management Optimization*. https://doi.org/10.3934/jimo.2020102

Opasanon, S. (2005). *On finding paths and flows in multicriteria, stochastic, and time-varying networks* (Ph.D. Thesis). Department of Civil and Environmental Engineering, University of Maryland, College Park, USA.

Pundir, P. S., Porwal, S. K., & Singh, B. P. (2015). A new algorithm for solving linear bottleneck assignment problem. *Journal of Institute of Science and Technology*, *20*(2), 101-102.

Pyakurel, U., & Dhamala, T. N. (2016). Continuous-time dynamic contraflow models and algorithms. *Advanced Operation Research*, Hindawi. https://doi.org/10.1155/2016/7902460

Pyakurel, U., Nath, H. N., & Dhamala, T. N. (2018). Efficient contraflow algorithms for quickest evacuation planning problems. *Science China Mathematics*, 2079-2100.

Saho, M., & Shigeno, M. (2017). Cancel-and-tighten algorithm for quickest flow problems. *Networks*, *69*(2), 179-188.

Tjandra, S. A. (2003). *Dynamic network optimization with application to the evacuation problem* (Ph.D. Thesis). Department of Mathematics, Universitat Kaiserlautern, Germany.

Yu, H., & Thapa, G. B. (2014). A cross-docking framework for supply chain logistics in production scheduling. *Journal of Institute of Science and Technology*, *19*(1), 39-44.