Journal of Institute of Science and Technology, 27(2), 77-83 (2022)
ISSN: 2467-9062 (print), e-ISSN: 2467-9240
https://doi.org/10.3126/jist.v27i2.47223

Research Article

MAXIMUM CAPACITY PATH PROBLEM WITH ARC REVERSALS

Hari Nandan Nath
Tribhuvan University, Bhaktapur Multiple Campus, Bhaktapur, Nepal
Correspondence: hari672@gmail.com

(Received:** August 04, 2022; **Final Revision:** December 30, 2022; **Accepted:** December 31, 2022**)

ABSTRACT

In a network with capacitated arcs, the maximum capacity path problem is to find the bottleneck path between two nodes that has the maximum capacity among all the paths between the nodes. In this work, a problem of identification of such a path is considered allowing arc reversals in a directed graph where the capacity of the reversed arc can be added to the capacity of its counterpart. We propose an $O(m)$ algorithm to solve the problem in a directed graph with m arcs.

Keywords: bottleneck path, maxmin path, maxmin path, shortest path, complexity
Mathematics Subject Classification: Primary: 90B10, 90C27, 68Q25 Secondary: 90B06, 90B20.

INTRODUCTION

Network flow problems constitute an important class of combinatorial optimization problems. They arise in the various areas of study including applied mathematics, engineering, operations research, management, and computer science with numerous practical applications (Ahuja *et al.*, 1993). The problems are formulated in a graph (or network) consisting of nodes and arcs connecting the nodes. The basic problems are to find the shortest distance between the nodes, the maximum amount of flow from some nodes to the other nodes, and the flow of the minimum cost. The cost related to an arc can be its length, time for the flow to traverse the arc, or any other weight. The flow in an arc is also limited by its capacity.

In the applications related to the optimization of traffic flow, the streets are taken as the arcs and their intersections, the nodes of a directed graph. The direction of an arc is taken as the direction of the the traffic flow in it. Various applications related to the traffic optimization in emergency evacuation planning are reviewed in Hamacher and Tjandra (2001); Dhamala (2015); Dhamala *et al.* (2018). One of the research interests in such problems is to optimize the traffic flow by finding the ideal direction of the traffic flow, reversing the usual direction of traffic flow in appropriate road segments (see Kim *et al.* (2008), Rebennack *et al.* (2010), Pyakurel (2016), Pyakurel and Dhamala (2017); Pyakurel *et al.* (2018), Pyakurel *et al.* (2019), and Gupta *et al.* (2021)).

Among all the network flow problems, identification of a path between two nodes, especially the shortest path is the simplest and a basis to many other network flow problems. There are numerous variants of the problem, e.g. finding the shortest path between two specific nodes, to all the nodes from a specific node, between all pairs of nodes (Ahuja *et al.*, 1993), and shortest paths with multiple criteria (Hansen, 1980; Tarapata, 2007; Nath *et al.*, 2021). Apart from shortest paths, the research interests also include finding a path with other objectives, e.g. the quickest path problem (Chen & Chin, 1990), maximum capacity path problem (Punnen, 1991; Kaibel & Peinhardt, 2006).

The present work extends the maximum capacity path problem to the one with arc reversals in a directed graph. Section 2 sets basic graph theoretic ideas required for the development of the paper. Section 3 describes the maximum capacity path problem, Section 4 presents the main results, and Section 5 concludes the paper.

BASIC NOTIONS

For the formal treatment, we consider a graph or network $G = (N, A)$ consisting of a set of nodes N and a set of arcs A. The elements of A are pairs of distinct nodes in N. Each arc $(i, j) \in A$ has an associated non-negative capacity u_{ij}. We call an arc (i, j) directed from i to j, if it is ordered, i.e. $(i, j) \neq (j, i)$, and undirected if $(i, j) = (j, i)$. An undirected arc between two nodes i and j can be represented by (i, j) or (j, i). We call G directed if all the arcs in A are directed (Figure 1(i)), undirected if the arcs in A are undirected (Figure 1(ii)), and mixed if some arcs are directed and some are not. We denote the number of nodes $|N|$ by n and the number of arcs $|A|$ by m.

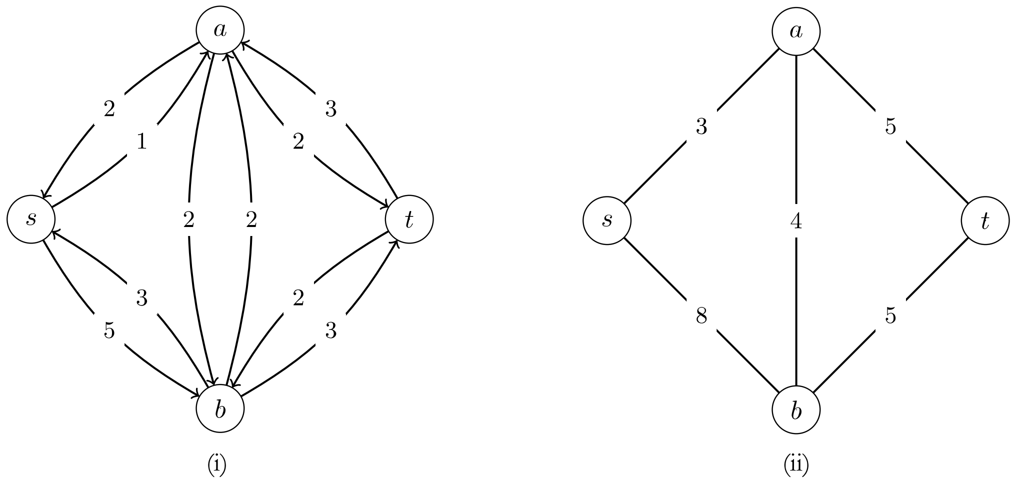


Figure 1: (i) A directed network, (ii) an undirected network (each arc label represents its capacity)

For a sequence of nodes $i_1, \cdots, i_k \in N$, a path P denoted by $i_1 - \cdots - i_k$ is a sequence of arcs $\{(i_1, i_2), \cdots, (i_{k-1}, i_k)\}$. A path with a starting node i_1 and end node i_k is also known as i_1-i_k path. If all the arcs in P are directed, then P is called a directed path or a chain. We define the capacity of a path P as $u_P = \min\{u_{ij}: (i,j) \in P\}$.

Example 1. In Figure 1(i), the capacity of the path s–a–b–t is

$$\min\{u_{sa}, u_{ab}, u_{bt}\} = \min\{1, 2, 3\} = 1$$

and that of s–b–t is

$$\min\{u_{sb}, u_{bt}\} = \min\{5, 3\} = 3.$$

MAXIMUM CAPACITY PATH PROBLEM

Among all the paths between two distinct nodes in a network, a path with the maximum capacity is called the maximum capacity path. Given a network with arc capacities, the problem of identification of such a path is called the maximum capacity path problem or the bottleneck path problem (Punnen, 1991; Kaibel & Peinhardt, 2006).

Example 2. Consider the network given in Figure 1(i). The following table gives all the s-t paths and their respective capacities.

Path	Capacity
$s-a-t$	1
$s-a-b-t$	1
$s-b-t$	3
$s-b-a-t$	2

The maximum capacity $s-t$ path is $s-b-t$ with capacity 3.

Definition 1. Let $G = (N, A)$ be a network and $s, t \in N$ be two distinct nodes. Suppose $\mathcal{P}(s,t)$ be the collection of all $s-t$ paths in G. If u_P is the capacity of path P, then a maximum capacity path is the path with the capacity $\max\{u_P : P \in P(s,t)\}$.

According to Punnen (1991), then the maximum capacity path problem can be stated as $\max_{P \in \mathcal{P}(s,t)} \min_{(i,j) \in P} u_{ij}$.

In fact, the above problem finds the maximum capacity of an arc through which there exists at least one $s-t$ path. Any $s-t$ path considering only the arcs of capacity not less than this capacity is a maximum capacity path.

Let $u_{ij} = U_0\ \forall (i,j) \in A$, and suppose that there exists an $s-t$ path. Then it is clear that $\max_{P \in \mathcal{P}(s,t)} \min_{(i,j) \in P} u_{ij} = U_0$.

This gives the following result.

Proposition 1. In a network $G = (N, A)$ with the capacity $u_{ij} = U_0\ \forall (i,j) \in A$ with at least one $s-t$ path, each $s-t$ path is a maximum capacity $s-t$ path with capacity U_0.

Definition 2 (Critical arc). If P is a maximum capacity $s-t$ path, then $(i^*, j^*) \in P$ is called a critical arc if $u_{i^*j^*} = \min\{u_{ij} : (i,j) \in P\}$.

Punnen (1991) gave an $O(m)$ recursive procedure to find the bottleneck capacity, hence a critical arc, in an undirected network with distinct arc capacities. The procedure is given in Algorithm 1. Although the arc capacities are assumed to be distinct, the algorithm can be used for the general case by numbering the arcs arbitrarily and using lexicographic order in arc capacity and arc number for the purpose of finding median in Line 5 and construction of A^k in Line 6.

To use the algorithm to directed graphs, the connected components in Line 11 have to be strongly connected because in this way such an algorithm does not have a linear running time bound. To solve the maximum capacity path problem in directed graphs, Kaibel and Peinhardt (2006) devise an algorithm that solves the problem in $O(m \log \log m)$ time.

Algorithm 1 Procedure BottleneckCapacity(G, s, t) (Punnen, 1991)

```
1: Input: An undirected network G = $(N, A)$ with arc capacity $u_{ij} \geq 0$ distinct for each $(i,j) \in A$, source
          $s \in V$ and sink $t \in N$
2: if G contains a single edge (s, t) only then
3:        return $u_{st}$
4: else
5:        $k :=$ median of $\{u_{ij}: (i,j) \in A\}$
6:        $A^k := \{(i,j) \in A: u_{ij} \geq k\}$
7:        $G^k := (V, A^k)$
8:        if $s$ and $t$ are in the same connected component $\tilde{G}$ of $G^k$ then
9:        return BottleneckCapacity$(\tilde{G}, s, t)$
10:       else
11:               Let $G_1^k := (N_1^k, A_1^k), \cdots, G_q^k := (N_q^k, A_q^k)$ be the connected components of $G^k$
12:               for $1 \leq v \leq q$ and $1 \leq w \leq q$ do
13:                       $S^k(v,w) := \{(i,j) \in A: i \in G_v^k, j \in G_w^k\}$
14:               end for
15:               $\bar{N}^k := \{1, \cdots, q\}$
16:               $\bar{A}^k := \{(v,w): S^k(v,w) \neq \emptyset\}$ with $u_{vw} := \max_{(i,j) \in S^k(v,w)} u_{ij}$
17:               $\bar{G}^k := (\bar{N}^k, \bar{A}^k)$
18:               Let $x, y$ be such that $s \in N_x^k, t \in N_y^k$
19:               return BottleneckCapacity$(\bar{G}^k, x, y)$
20:       end if
21: end if
```

MAXIMUM CAPACITY PATH PROBLEM ALLOWING ARC REVERSALS IN A DIRECTED GRAPH

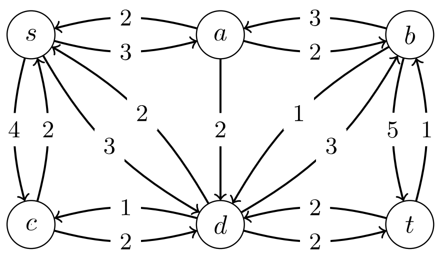
In a directed graph $G = (N, A)$, let us assume that the direction of each arc can be reversed, and that the capacity of an arc $(i,j) \in A$ can be increased to $u_{ij} + u_{ji}$ by reversing the direction of (j,i), where u_{xy} represents the capacity of the arc $(x,y) \in A$. Consequently, the capacity of the reversed arc (j,i) becomes zero or (j,i) is removed.

To solve various network flow problems with arc reversals, an idea is to construct an undirected network by adding the capacities of the opposite arcs and solve the corresponding problem in the resulting network (Pyakurel & Dhamala, 2016). Such a network is known as the auxiliary network of the original network.

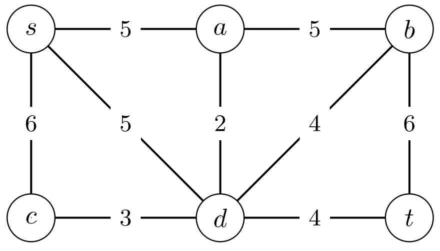
Definition 3 (Auxiliary network). Given a directed network $G = (N, A)$, with capacity u_{ij} for each $(i,j) \in A$, the auxiliary network is defined as $\bar{G} = (N, \bar{A})$, where $\bar{A} = \{(i,j): (i,j) \in A \text{ or } (j,i) \in A\}$ and for each $(i,j) \in \bar{A}$, the capacity $\bar{u}_{ij}$ is given by

$$\bar{u}_{ij} = \begin{cases} u_{ij} + u_{ji}, & \text{if } (i,j),(j,i) \in A \\ u_{ij}, & \text{if } (i,j) \in A, (j,i) \notin A. \end{cases}$$

Example 3. Consider the directed network G as shown in Figure 2(i). The arc labels show the corresponding arc capacity. Its auxiliary network $\bar{G}$ is shown in Figure 2(ii). In $\bar{G}$, the vertices remain the same and the arc pair $(i,j),(j,i)$ becomes an undirected arc (i,j) in $\bar{G}$. The capacity of the edge (i,j) is defined as $\bar{u}_{ij} = u_{ij} + u_{ji}$, e.g., $\bar{u}_{ab} = u_{ab} + u_{ba} = 2 + 3 = 5$.



(i)



(ii)

Figure 2: (i) A directed network $G = (N, A)$, (ii) Auxiliary network $\bar{G} = (N, \bar{A})$

Example 4. In the network given in Figure 2(i), a maximum capacity s–t path is a path of capacity 3 ($s-d-b-t$). If we allow arc reversals, we get a maximum capacity $s-t$ path of capacity 5 ($s-a-b-t$), the arcs $(a,s),(b,a)$ being reversed to enhance the capacities of $(s,a),(a,b)$ respectively. In the auxiliary network (Figure 2(ii)) also the maximum path is is also $s-a-b-t$ with capacity 5.

We design Algorithm 2, that uses the auxiliary network construction of the original network to find the maximum capacity path with arc reversals and the set of paths to be reversed.

Algorithm 2: Maximum capacity path with arc reversals

```
1: Input: Directed graph $G = (N, A)$ with capacity $u_{ij}$ for each $(i,j) \in A$, two specific nodes $s$ (the source
       node), $t$ (the sink node) such that there is at least one $s-t$ path
2: Construct the undirected auxiliary network $\bar{G} = (N, \bar{A})$ with capacity $\bar{u}_{ij}$ for each $(i,j) \in \bar{A}$.
3: Compute the bottleneck $s-t$ path capacity $U$.
4: Construct $G^* = (N, A^*)$ with $A^* = \{(i,j) \in \bar{A}: \bar{u}_{ij} \geq U\}$.
5: Perform breadth first search in $G^*$ to find an $s-t$ path $P := s-x_1-\cdots-x_k-t$ with ordered edges
     $(s,x_1),(x_1,x_2),\cdots,(x_k,t)$.
6: Set of reversed arcs $R = \{(j,i) \in A : (i,j) \in P \text{ and } u_{ij} < U\}$.
7: return $P$ (the maximum capacity path with arc reversals), and $R$ (the set of reversed arcs)
```

As the capacity of an arc after reversing the opposite arc is defined to be $u_{ij} + u_{ji}$, the maximum capacity path in the auxiliary network is the maximum capacity path with arc reversals in the original network. If U is the bottleneck capacity of $s-t$ paths in the auxiliary network, and if (i,j) is in the bottleneck path with $u_{ij} < U$ in the original network, then (j,i) has to be reversed so that the capacity of (i,j) becomes at least U. In this way, we can realize the correctness of the algorithm.

Theorem 1. *Algorithm 2 computes the maximum capacity path with arc reversals correctly.*

Complexity Analysis of Algorithm 2. The auxiliary network construction in Line 2 can be done in $O(m)$ time. The crucial Step in Algorithm is the computation of the bottleneck $s-t$ path capacity in Line 3, which can be computed using Algorithm 1 if the arc capacities in the auxiliary network are distinct. If the arc capacities are not all distinct, then we can number the arcs arbitrarily and consider the lexicographical order in (arc capacity, arc number) to find the median and use the same algorithm. In either case, the running time is $O(m)$. Construction of G^* in Line 4 requires deletion of arcs of $\bar{G}$ with capacity less than U. This also takes $O(m)$ time in the worst case. The breadth-first search can be done, in general, in $O(m + n)$ time. In Line 5, however, it has to be done in a connected graph. For a connected graph, $n \leq m + 1$. So, the step can also be performed in $O(m)$ time. Line 6 requires identification of reversed arcs in the arcs of the chosen maximum capacity path only, the running time of the step is also dominated by $O(m)$. Thus we can conclude the following.

Theorem 2. *The maximum capacity path problem with arc reversals can be solved in $O(m)$ time.*

We illustrate the working of the algorithm 2 in Example 6 below. As algorithm 2 uses Algorithm 1 as a subroutine, we first illustrate its working in Example 5.

Example 5. Consider an undirected network given in Figure 3(i). The arc labels represent capacities of the corresponding arcs. For simplicity, we take the capacities distinct.

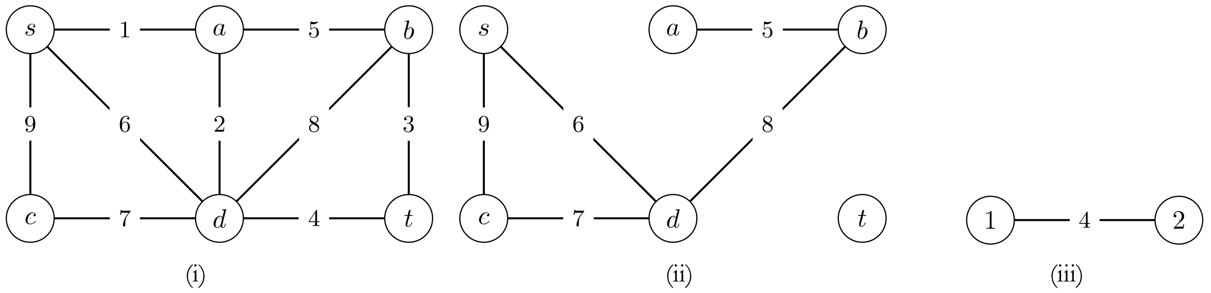


Figure 3: Working of Algorithm 1

To identify the critical arc in a maximum capacity path, we apply Algorithm 1, that works as follows.

Iteration 1:

$V = \{s, a, b, c, d, t\}$. k = median of $\{1, 2, 3, 4, 5, 6, 7, 8, 9\} = 5$,

$A^k = \{(s, c), (s, d), (a, b), (b, d), (c, d)\}$. $G^k = (V, A^k)$

s, t are not in the same connected compont of G^k (Figure 3(ii)). There are two connected components.

$N_1 = \{s, a, b, c, d\}, N_2 = \{t\}, A_1^k = \{(s, c), (s, d), (a, b), (b, d)\}, A_2^k = \{\}$.

$G_1^k = (N_1^k, A_1^k), G_2^k = (N_2^k, A_2^k)$. $v = 1, 2, w = 1, 2$.

$S^k(1,1) = S^k(2,2) = \{\}, S^k(1, 2) = S^k(2,1) = \{(b, t), (d, t)\}$.

$\bar{N}^k = \{1, 2\}, \bar{A}^k = \{(1,2)\}, u_{12} = \max\{3, 4\} = 4, \bar{G}^k = (\bar{N}^k, \bar{A}^k), s \in N_1^k, t \in N_2^k$.

Iteration 2:

$G = (N, A)$ with $N = \bar{N}^k = \{1, 2\}, A = \{(1, 2)\}$. Since G contains only a single edge $(1, 2)$, the maximum capacity is 4. The critical arc is (d, t).

Example 6. To illustrate Algorithm 2, let us consider a directed network G given in Figure 4(i). Its auxiliary network $\bar{G}$ constructed in Step 2 is shown in Figure 4(ii). Step 3 computes the bottleneck maximum path capactiy $U = 4$ (See Example 6).

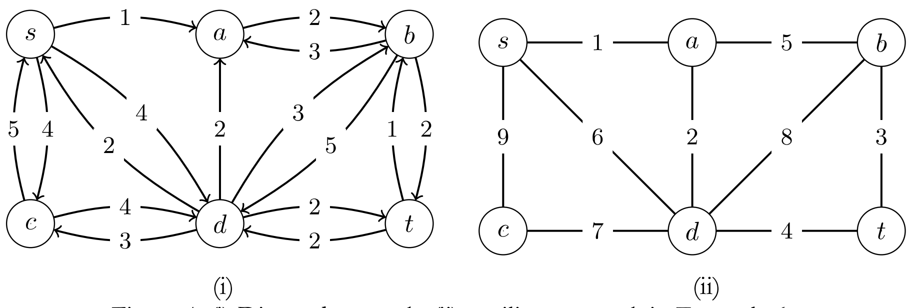


Figure 4: (i) Directed network, (ii) auxiliary network in Example 6.

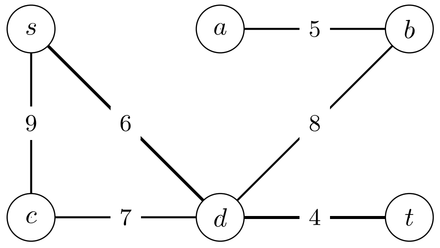


Figure 5: Newotk G^* constructed by Step 4 of Algorithm 2 in Example 6.

The network G^* constructed in Step 4 of the algorithm is depctied in Figure 5. Performing breadth first search in G^*, we obtain the path $P = s - d - t$, with ordered edges $(s,d), (d,t)$, as the maximum capacity path after arc reversals. As $u_{dt} = 2 < U$, (t,d) is reversed, and thus $R = \{(t,d)\}$.

CONCLUSIONS

In this work, we have considered a problem of finding the path of maximum capacity by allowing arc reversals so that the capacities of the arcs can be added towards the preferred direction. We have designed an algorithm that solves the problem in a graph with m arcs in $O(m)$ time. The problem is particularly useful in finding the unsplittable maximum flow between two nodes of a network with capacitated arcs. For example, the problem can be used in transportation planning in urban road networks in cases where a single path has to be chosen to send a maximum amount of flow allowing reversal of the usual direction of the traffic flow in appropriate road segments. There are several possible extensions of the problem, e.g. maximum capacity paths between a node and every other node, between every pair of nodes, maximum capacity path with the consideration of costs. Further, we are also interested in finding such a path with multiple criteria including minimization of the time of the flow and the number of reversed arcs.

ACKNOWLEDGEMENTS

The author is thankful to anonymous reviewers for their careful reading of the manuscript and valuable suggestions to improve the quality of the paper.

CONFLICT OF INTEREST

The authors do not have any conflict of interest pertinent to this work.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author, upon reasonable request.

REFERENCES

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

Chen, Y.L., & Chin, Y.H. (1990). The quickest path problem. *Computers & Operations Research*, *17*(2), 153–161.

Dhamala, T.N. (2015). A survey on models and algorithms for discrete evacuation planning network problems. *Journal of Industrial and Management Optimization*, *11*(1), 265–289.

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

Gupta, S.P., Pyakurel, U., & Dhamala, T.N. (2021). Multi-commodity contraflow problem on lossy network with asymmetric transit times. In *Computer Sciences & Mathematics Forum,* 2(21). https://doi.org/10.3390/IOCA2021-10878.

Hamacher, H.W., & Tjandra, S.A. (2001). *Mathematical modelling of evacuation problems*: *A state of art*. Fraunhofer-Institut für Techno-und Wirtschaftsmathematik, Fraunhofer (ITWM).

Hansen, P. (1980). Bicriterion path problems. In Fandel, G., & Gal, T. (Eds.), *Multiple criteria decision making theory and application. Lecture notes in economics and mathematical systems* (pp. 109–127). Springer, Berlin. https://doi.org/10.1007/978-3-642-48782-8_9

Kaibel, V., & Peinhardt, M. (2006). *On the bottleneck shortest path problem*. ZIB-Report # 06-22. Otto-von-Guericke-Universit¨ at Magdeburg. Retrieved January 27, 2022 from http://www.math.uni-magdeburg.de/~kaibel/Downloads/BSP.pdf

Kim, S., Shekhar, S., & Min, M. (2008). Contraflow transportation network reconfiguration for evacuation route planning. *IEEE Transactions on Knowledge and Data Engineering*, *20*(8), 1115–1129.

Nath, H.N., Dempe, S., & Dhamala, T.N. (2021). A bicriteria approach for saving a path maximizing dynamic contraflow. *Asia-Pacific Journal of Operational Research*, *39*(03), 2150027. doi: https://doi.org/10.1142/S0217595921500275

Punnen, A.P. (1991). A linear time algorithm for the maximum capacity path problem. *European Journal of Operational Research*, *53*(3), 402–404.

Pyakurel, U. (2016). *Evacuation planning problem with contraflow approach*. PhD thesis. Central Department of

Mathematics, Institute of Science and Technology, Tribhuvan Univeristy, Nepal.

Pyakurel, U., & Dhamala, T.N. (2016). Continuous time dynamic contraflow models and algorithms. *Advances in Operations Research*, 7902460. doi: http://dx.doi.org/10.1155/2016/7902460

Pyakurel, U., & Dhamala, T.N. (2017). Continuous dynamic contraflow approach for evacuation planning. *Annals of Operations Research, 253*(1), 573–598.

Pyakurel, U., Nath, H.N., Dempe, S., & Dhamala, T.N. (2019). Efficient dynamic flow algorithms for evacuation planning problems with partial lane reversal. *Mathematics*, *7*(10), 933. doi: https://doi.org/10.3390/math7100993

Pyakurel, U., Nath, H.N., & Dhamala, T.N. (2018). Efficient contraflow algorithms for quickest evacuation planning. *Science China Mathematics*, *61*(11), 2079–2100.

Rebennack, S., Arulselvan, A., Elefteriadou, L., & Pardalos, P.M. (2010). Complexity analysis for maximum flow problems with arc reversals. *Journal of Combinatorial Optimization*, *19*(2), 200–216.

Tarapata, Z.M. (2007). Selected multicriteria shortest path problems: An analysis of complexity, models and adaptation of standard algorithms. *International Journal of Applied Mathematics & Computer Science, 17*(2), 269-287.