Flows on Network with Intermediate Storage Capability: Evacuation Planning Perspective

Phanindra Prasad Bhandari[1] and Shree Ram Khadka[2]

[1]Science and Humanities Department
Khwopa Engineering College, Nepal

[2]Central Department of Mathematics
Tribhuvan University, Nepal

September 28, 2021

Outline

- Background
- Problem Formulation
- Literature
- Result
- References

Evacuation Planning via Network Flow Model

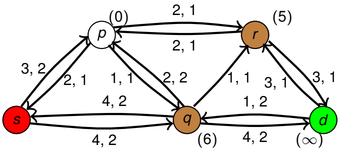

Figure 1: An evacuation scenario represented by a network N.

- Evacuation region and its scenario is modeled as network N.
- Risk zone is source s and safe the sink d.
- Intersection of routes are vertices v and links between them are arcs a.
- Capacities on vertices $k(v)$ and arcs $u(a)$, travel time on arcs $\tau(a)$ are associated.
- Temporary shelters (prioritized!) at intermediate places with capacities could be introduced.
- Evacuees on road is the flow $f : A \times \mathcal{T} \to \mathbb{N} \cup \{0\}$ on arc a at time t.

Definition (Evacuation Planning Problem)

Shift as many evacuees as possible in given time period, or given number of evacuees in minimum possible time period from risk zone to safer through existing road network.

Non- conservation Flow Model: Features and Importance

- Amount of evacuees into an intermediate vertex may not be equal to the amount out of it.
- Extra evacuees may leave the source who cannot reach the sink.
- Allows to hold evacuees at temporary shelters. Applicable in modeling EPP.

Lexicographic Maximum Flow Problem (Bhandari et al. (2020))

Given network $N = (V, A, l(a), u(a), k(v), \tau(a), s, d, T)$, set of terminals $\mathcal{S} = \{v_1, \dots, v_k\}$ with $d = v_1 \succ \cdots \succ v_k$.

Then

$$\text{lex max}(ex_f(v_1, T), \dots, ex_f(v_k, T))^\top$$

such that

$$0 \le f(a,t) \le u(a) \;\; \forall \;\; a \in A \;\text{ and }\; \forall \;\; t \in \mathcal{T}. \tag{1}$$

$$ex_f(v_i, T) \le k(v_i) \text{ for } i = 1, \dots, k \tag{2}$$

where

$$0 \le ex_f(v,t) := \sum_{a \in \delta^-(v)} \sum_{\xi=0}^{t-\tau(a)} f(a,\xi) - \sum_{a \in \delta^+(v)} \sum_{\xi=0}^{t} f(a,\xi). \tag{3}$$

Consequently,

$$\sum_{a \in \delta^+(s)} \sum_{t=0}^{T} f(a,t) - \sum_{a \in \delta^-(s)} \sum_{t=0}^{T} f(a,t) = \sum_{v \in \mathcal{S}} ex_f(v,T). \tag{4}$$

Literature

- Maximum Flow Problem: Sufficient $k(v_i)$
  (Khadka and Bhandari (2019)), (Bhandari and Khadka (2020b)
- Fixed $k(v_i)$: Static Version (Bhandari et al. (2020))
- Dynamic Version (Time-expanded Network) (Bhandari et al. (2020))
- Dynamic Version: Sufficient $k(v_i)$ (Bhandari et al. (2020))

Limitation!

LexMDF for Uniform Path Length (UPL) Network with Capacitated Vertices (Bhandari and Khadka (2020a)

A directed network $\mathcal{N}$ is UPL network for which the sum of transit time on arcs on any possible path from source s to any vertex $v \in \mathcal{N}$ are equal.

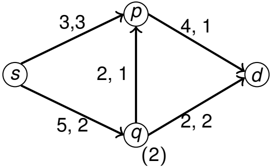

Figure 2: A uniform path length (UPL) network $\mathcal{N}$ with source vertex s.

Solution: Main Idea (Bhandari (2021))

- Lexicographically Minimum Cost Circulation (LexMCC) Problem $\Rightarrow \Gamma_{v_i}$, set of paths $\gamma_{v_i}, \forall v_i \in \mathcal{S}$
  - $f(\gamma_{v_i})$ – flow value that can be sent along γ_{v_i} at once
  - $\tau(\gamma_{v_i})$ – time required to travel γ_{v_i} by a flow unit
  - $I_t(\gamma_{v_i})$ – time step at which the flow along γ_{v_i} starts to get repeated
  - $F_t(\gamma_{v_i})$ – time step after which the flow along γ_{v_i} stops to get repeated
- Extended set $\Gamma^E_{v_i}$ given by

$$\Gamma^E_{v_i} := \begin{cases} \Gamma_{v_i} & \text{for } i = 1 \\ \Gamma_{v_i} \cup \Gamma'_{v_i} & \text{for } i > 1 \end{cases}$$

where Γ'_{v_i} is set of all $s - v_i$ paths free to carry flow units at v_i at time intervals $I_1(\gamma_{v_{i-1}}) = [I_t(\gamma_{v_{i-1}}), F_t(\gamma_{v_{i-1}}) - N(\gamma_{v_{i-1}})]$ and $I_2(\gamma_{v_{i-1}}) = [F_t(\gamma_{v_{i-1}}) + 1, T]$ w.r.t. each path $\gamma_{v_{i-1}} \in \Gamma^E_{v_{i-1}}$.

- Path Flows Repetition (PFR) Technique on $\Gamma^E_{v_i}$ (TRFs !)

Solution

Algorithm (DT-LexMDF Algorithm for UPL Network)

1. *Given a dynamic UPL network* $\mathcal{N} = (G, u(a), \tau(a), k(v), s, d, T)$, $\mathcal{S} = \{v_1, \ldots, v_k\}$ *with* $d = v_1 \succ \cdots \succ v_k$.
2. *Find* $\Gamma_{v_i}\ \forall\ i = 1, 2, \ldots, k$ *by solving the LexMCC problem on* $\mathcal{N}$ *with additional arcs* (v_i, s) *with capacity* $k(v_i)$ *and transit time* $-(T+1)$.
3. *For* $i = 1$, *set* $\Gamma^E_{v_i} = \Gamma_{v_i}$ *and apply PFR technique on* $\Gamma^E_{v_i}$. *For* $i > 1$, *go to step 4.*
4. *For each path* $\gamma_{v_{i-1}} \in \Gamma^E_{v_{i-1}}$, *find the interval* $[F_t(\gamma_{v_{i-1}}) + 1 - N(\gamma_{v_{i-1}}), F_t(\gamma_{v_{i-1}})]$ *and intervals* $I_1 = [I_t(\gamma_{v_{i-1}}), F_t(\gamma_{v_{i-1}}) - N(\gamma_{v_{i-1}})]$ *and* $I_2 = [F_t(\gamma_{v_{i-1}}) + 1, T]$.
5. *Renovate the network* $\mathcal{N}_{\Gamma_{v_k}}$ *with respect to path* $\gamma_{v_{i-1}}$ *for intervals* I_1 *and* I_2.
6. *Find* $\Gamma'_{v_i}\ \forall\ i = 2, \ldots, k$ *solving LexMCC problem on renovated* $N_{\Gamma_{v_k}}$ *as initial time* $I_t(\gamma_{v_i})$ *with additional arcs* (v_i, s) *with capacity* $k(v_i)$ *and transit time* $-(T+1)$.
7. *Set* $\Gamma^E_{v_i} = \Gamma_{v_i}$ *and update* $\Gamma^E_{v_i} := \Gamma^E_{v_i} \cup \Gamma'_{v_i}\ \forall\ i = 2, \ldots, k$.
8. *Apply PFR technique on* $\Gamma^E_{v_i}$.
9. *Obtain dynamic* $s - v_i$ *flow on* $\mathcal{N}$.

Optimality and Complexity

Lemma

Given a UPL network $\mathcal{N}$ with prioritized set of vertices $\mathcal{S} \subset V$. Then LexMCC problem can be solved in $O(n \times MCF(n, m))$ time on $\mathcal{N}$ where $MCF(n, m)$ is the time complexity for single MCF problem.

Lemma

In the Algorithm, LexMCC problem is solved at most $2n$ time for each $v_i \in \mathcal{S}$.

Lemma

Renovation of the residual network $\mathcal{N}_{\Gamma_{v_k}}$ is well defined for each iteration.

Lemma

For any $v_i \in \mathcal{S}$, the number of paths in the extended set $\Gamma_{v_i}^{E}$ is bounded above by $2nm$.

Optimality and Complexity, contd.

Lemma

The PFR technique executes in time $O(nm + logT)$.

Lemma

The residual network $\mathcal{N}_{\Gamma_{v_k}}$ is renovated in time $O(nm)$ for each $v_i \in \mathcal{S}$.

Theorem

Given a UPL network $\mathcal{N} = (G, u(a), k(a), \tau(a), T)$, source s and terminal set $\mathcal{S} = \{v_1, \ldots, v_k\} \subset V$ with $d = v_1 \succ \cdots \succ v_k$. Then Algorithm yields an optimal solution to the LexMDF problem on $\mathcal{N}$.

Theorem

"DT-LexMDF Algorithm for UPL Network" runs in polynomial time with order $O(n^5\, m^3\, logn + n^2\, m + n\, logT)$.

Lexicographic Earliest Arrival Flow (LexEAF) Problem

(Bhandari (2021))

Algorithm (DT-LexEAF Algorithm for UPL-TTSP Network)

1. *Given a UPL-TTSP network* $\mathcal{N} = (G, u(a), \tau(a), k(v), s, d, T)$, $\mathcal{S} = \{v_1, \ldots, v_k\}$ *with* $d = v_1 \succ \cdots \succ v_k$.
2. *Solve LexMCC problem on* $\mathcal{N}$ *with additional arcs* (v_i, s) *with capacity* $k(v_i)$ *and transit time* $-(T+1)$*, Ruzika et al. (2011).*
3. *Construct extended set* $\Gamma^E_{v_i}$.
4. *Push as much flow as possible along each path in* $\Gamma^E_{v_i}$ *as long as possible within* T.
5. *Obtain dynamic* $s - v_i$ *flow on* $\mathcal{N}$.

Theorem

The Algorithm yields an optimal solution to the DT-LexEAF problem on UPL-TTSP network $\mathcal{N}$ *in polynomial time.*

Lexicographic Quickest Flow Problem
(Bhandari (2021))

Given network $N = (V, A, u(a), \mu(v), \tau(a))$, $\mathcal{S} = \{v_1, \ldots, v_k\}$ with $d = v_1 \succ \cdots \succ v_k$ such that $\sum_i \mu(v_i) = -\mu(s)$.

Then

$$\text{lex min}\,(T(\mu(v_1)), T(\mu(v_2)), \ldots, T(\mu(v_k))^\top$$

such that

Capacity constraints (1) and

$$ex_f(v_i, T(\mu(v_i))) = \mu(v_i) \ \forall \ v_i \in \mathcal{S}. \tag{5}$$

UPL network case:

Existence:

1. $\Gamma^E_{v_i} \neq \phi$ due to PFR technique and s being *mother vertex* for all $v_i \in \mathcal{S}$

Solution:

1. Apply procedure of Burkard et al. (1993), repeatedly
2. Apply "LexMDF Algorithm for UPL network" to compute maximum dynamic flows as subroutine

Future Works

- Search efficient solutions of dynamic flow problems for general network with prioritized capacitated vertices.
- Study multicommodity network flow problems with prioritized capacitated vertices that models the crowded traffic situation like Kathmandu

References I

[1] Bein WW, Brucker P and Tamir A (1985) Minimum cost flow algorithms for series-parallel networks, *Discrete Applied Mathematics* **10**(2) 117–124.

[2] P.P. Bhandari (2021) *Dynamic Network Contraflow Evacuation Planning Problem*, doctoral dissertation, Central Department of Mathematics, Tribhuvan University, Nepal.

[3] Bhandari PP and Khadka SR (2020a) Evacuation planning problems with intermediate storage, In *International Conference on Applied Mathematics & Computational Sciences (ICAMCS-2019), Dehradun*, 90–95.

[4] Bhandari PP and Khadka SR (2020b) Maximum flow evacuation planning problem with non-conservation flow constraint, *International Annals of Science* **10**(1) 25–32.

[5] Bhandari PP, Khadka SR, Ruzika S and Schäfer LE (2020) Lexicographically maximum dynamic flow with vertex capacities, *Journal of Mathematics and Statistics* **16**(1) 142–147.

[6] Burkard RE, DlaskaK and Klinz B (1993) The quickest flow problem, ZOR-Methods and Models of Operations Research **37**(1) 31–58.

References II

[7] Ford LR and Fulkerson DR (1958) Constructing maximal dynamic flows from static flows, *Operations research* **6**(3) 419–33.

[8] Goldberg AV and Tarjan RE (1988) A new approach to the maximum-flow problem, *Journal of the ACM (JACM)* **35**(4) 921–940.

[9] Goldberg AV and Tarjan RE (1989) Finding minimum-cost circulations by canceling negative cycles, *Journal of the ACM (JACM)* **36**(4) 873–886.

[10] Khadka SR and Bhandari PP (2019) Model and solution for non-conservation flow evacuation planning problem, *The Nepali Mathematical Sciences Report* **36**(1-2) 11–16.

[11] Ruzika S, Sperber H and Steiner M (2011) Earliest arrival flows on series-parallel networks, *Networks* **57** 169–173.

Suggestions are welcome!

Thank You