A novel polynomial algorithm for the lexicographic maximum dynamic flow problem with several sources applied to evacuation planning

Houssein Alaeddine[a,b], Emmanuel Néron[a], Kamal Serrhini[b], Mindjid Maizia[b]

[a] *Laboratoire Informatique de l'université de Tours*
[b] *Laboratoire Citères de l'université de Tours*
Tours, France

Abstract

Evacuation problems can be modeled in flow dynamic networks. Several variants of these networks are explored in the literature to address several issues, among them maximum dynamic flow, quickest dynamic flow, universal maximum flow, lexicographic maximum dynamic flow. Hoppe and Tardos [22] implemented a polynomial time algorithms for solving some problems of dynamic flows including the lexicographic maximum dynamic flow. They developed an algorithm based on the minimum cost flow computations with a complexity equals to $O[k(m' log n')(m' + n' log m')]$, where k is the number of sources, n' and m' are successively the number of nodes and the number of arcs in the time-expanded graph. In this latter n' is equal to $T \times n$ (n is the number of nodes in the static graph), while m' cannot be computed since it depends on constant travel time on arcs. We propose here a polynomial time algorithm based on push-relabel algorithm [18] for solving the lexicographic maximum flow problem at each discrete time step (time slot). The minimum number of time slots required for the evacuation process is provided. The discrete time step can be large here unlike the one of the time-expanded network since the time on roads depends on flow. The complexity of our algorithm can be estimated as between $O[\sum_{i=1}^{k} \frac{i^2}{k^2} nm \log(\frac{i}{k} \frac{n^2}{m})]$ and $O[k(nm \log(\frac{n^2}{m})]$, where k is the number of sources, n and m are successively the number of nodes and the number of arcs in the static graph. This complexity is performed using a dynamic tree data structure [28]. Moreover we show in this paper the importance of the lexicographic evacuation in order to ensure a non-preemption building evacuation and to minimize the time intervals during which a building is evacuated.

Keywords:
scheduling, optimization, lexicographic maximum dynamic flow, evacuation, risk

1. Introduction

Dynamic networks can model evacuation of buildings exposed to risk i.e. fire, flood, hurricane, etc. A dynamic network also called flows over time is defined by a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ that contains a set of nodes source $\mathcal{N}_b$ that holds a supplies, a set of nodes destination $\mathcal{N}_s$ that holds a demands and a set of nodes network $\mathcal{N}_t$ that holds nothing. An evacuation horizon time T is defined i.e. no flow can leave its source if its last particle does not reach its destination before T. Minieka [25] introduced the problem of the lexicographic dynamic flow in the network and proved that this flow exists in a static network. Given a set of sources $\{s_1, \ldots, s_k\}$, he showed that the amount of flow incoming from the set of sources $S_i = \{s_1, \ldots, s_i\}$ is simultaneously maximum for every i. Hamacher and Tufekci [21] addressed the evacuation of a building as a lexicographic min cost flow problem. Their objective is to minimize the total evacuation time, i.e. the time the last person leaves the building in danger. They proposed an algorithm based on the time-expanded network introduced by Ford-Fulkerson [14] with flow-independent travel time and constant capacity of network roads. The strategy followed in their algorithm is to evacuate the sources (that are a parts of the building) which have a high priorities to another sources having a low priorities i.e. an evacuation of a source s_i means that their occupants leave and go into a source s_j, $j > i$, or leave the building. They prove the utility of this strategy in the event of a fire inside a building where the different parts of the building will be burned successively starting from the venue of the fire (see also [19]). Hoppe and Tardos [22] implemented a polynomial time algorithms based on the time-expanded graph for solving some problems of dynamic flows among them the lexicographic maximum dynamic flow. They proposed an algorithm based on the minimum cost flow computations using flow-independent transit time as cost. A supersource s and a supersink p are created and an arc ps with transit time $-(T+1)$ is also created. A zero transit time is assigned to the arc ss_i for every $1 \leq i \leq k$. The algorithm proposed starts by computing a minimum cost circulation then the rest of this algorithm consists of k iterations indexed in descending fashion by deleting at each iteration $0 \leq i \leq k-1$ the arc (ss_{i+1}). More details about this algorithm in [22]. Kamiyama and Katoh [23] introduced the lexicographically maximum flow problem with hierarchies, a generalization of classical lexicographically maximum flow, and developed a polynomial-time algorithm for solving this problem.

Lim et al. [24] developed a flow scheduling/optimization approach using the time-expanded network. An evacuation priorities list is established by assigning a vulnerability degree

Email addresses: houssein.alaeddine@hotmail.fr (Houssein Alaeddine), emmanuel.neron@univ-tours.fr (Emmanuel Néron), kamal.serrhini@univ-tours.fr (Kamal Serrhini), mindjid.maizia@univ-tours.fr (Mindjid Maizia)

for each region exposed to risk. In other words, all buildings in the same area or region have the same vulnerability level. To overcome the huge size of the time-expanded network and especially in case of a mass evacuation, the authors developed a greedy heuristic approach for maximizing and scheduling flow at each time slot. The Dijkstra's algorithm [9] is used for finding the evacuation routes. The travel times on road links are independent of the flows on these arcs and therefore are estimated based on the peak rush hour travel time on a major highway [24]. Saadatsresht et al. [27] developed a multiobjective algorithm based on the Pareto front for constructing an efficient evacuation network. Their approach is based on three step : the selection of safe areas, the assignment of optimal safe area to each building block and finally the computation of optimal path between each building block and candidate safe area. That approach, as the authors mentioned, requires further development in order to take into consideration the traffic situation in the network over time. In other words it requires an evacuation scheduling model for the evacuee groups based on a prioritization system. Bretschneider and Kimms [6] developed a mixed-integer heuristic for reorganizing the traffic routing in the case of an evacuation, such that the people leaves the dangerous area as early as possible. The time-expanded network is used in order to describe the network at different points of time. That approach includes two stages, the first one treats the traffic in the network without taking into consideration the detailed modeling of the intersections. And the second stage, which is based on the first one, takes into account every entrance/exit of the intersections and all possible turnings within the intersections [6]. The evacuation process is not subjected to a prioritization system. The developed approach is based on the dynamic minimum cost flow problem which aims to minimize the total clearance time. Stepanov and Smith [29] considered an integer traffic assignment model for meeting several objectives : minimizing the total clearance time, total distance and traffic-jams. From several candidate paths, the linear program aims, under the announced objectives, to find one evacuation egress route for each origin. To manage network traffic, a road link is modeled as a queuing service facility with several physical characteristics such that the vehicles enter the evacuation network according to Markovian Process, there is no waiting space i.e. as soon as a vehicle enters the arc it starts obtaining the service, and others.

After synthesizing the literature on the modeling of evacuation problem, we present in this paper a new dynamic lexicographic maximum flow algorithm based on push-relabel algorithm. Before presenting the reformulation of problem we discuss briefly continuous and discrete time, also the aspect of flow-dependent transit time. Then we propose an efficient evacuation algorithm which enables to maximize lexicographically the flow in the network, to ensure a non-preemption building evacuation and to minimize the time intervals during which a building is evacuated. The transit times on roads are dependent on flows that cross these roads. An experimentation section is provided at the end of this paper.

2. Dynamic flows

Two aspects distinguish the dynamic flow from static one. Firstly, the dynamic flow varies on arcs over time ; secondly, it does not spread instantly across the network, but requires some time to travel through each arc.

Flows over time traditionally is solved by a time-expanded network, introduced by Ford and Fulkerson (1958), that contains a copy of the original network (static network) for each discrete time step $t \in \{0, 1, \ldots, T-1\}$. Given a static graph $G = (N, A)$, a time-expanded graph $G_{T-1} = (N_{T-1}, A_{T-1})$ can be constructed and the dynamic maximum flow problem during the period T is a classical problem of the maximum flow from source $s(0)$ to destination $p(T-1)$ [12].

Another variant of dynamic flow network, introduced also by Ford and Fulkerson, is the flow temporally repeated that is based on maximum flow minimum cost in a static network. A decomposition paths algorithm is used [8] and the idea is to send flow via each path p_i at every $t \in \{0, 1, \ldots, T-1-\tau(p^i)\}$, where $\tau(p^i)$ corresponds to the total travel time on the path p_i. Many authors addressed the flows over time problem offering different efficient approaches for different problems (dynamic maximum flow [16, 15, 12], universal dynamic maximum flow [17, 22, 25, 30, 22, 10, 26, 13], quickest dynamic flows [11, 7, 5, 22], lexicographic dynamic maximum flow [25, 20, 21, 22]). It's well known that any multi-sources problem can be transformed into a single source problem by creating a super-source but this transformation does not hold valid here in the dynamic flow model i.e. the flow temporally repeated algorithm computes for every source an amount of flow that can be circulated in the network at each time step but it's not at all sure that all sources will have a positive amount, this is first, and secondly a small source in terms of number of vehicles to be evacuated will success to evacuate all its vehicles in a shorter time. In other words a place in evacuation network is reserved for this source during the horizon time while this source needs a little time only.

Here in this paper we will study the flows over time from scheduling side where at each time slot θ we evacuate from sources (buildings) by priority order the lexicographic maximum number of vehicles. The minimum number of time slots required for evacuation is provided and we show the importance of the lexicographic evacuation in order to firstly, ensure a non-preemption building evacuation, and secondly, minimize the time intervals during which a building is evacuated. We assume the existence of a traffic model (Greenshield's model, polynomial model, etc.) for computing the flow-dependent travel time on arcs. Moreover, being the sources from which the flows incoming change over time, as we will show, a vehicles pursuit model is required in order to avoid any overlap in the network. In other words that model which traces the paths of vehicles aims to compute the earliest departure times of each building while preventing any traffic-jams to occur in the network. This model is also assumed existing in order to simplify this paper (see [1]). Our model [1] computes and provides the total evacuation time T_e.

1. This paper is a part of STOM (A Spatio-Temporel Optimization Model for the evacuation of people exposed to natural disasters) model developed in the project ACCELL funded by Région Centre and FEDER, France

3. Graph construction

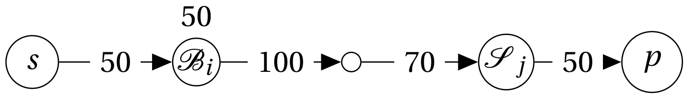
Let us consider a set **B** of n buildings to be evacuated, each building $\mathscr{B}_i, i \in \{1,\ldots,n\}$ is defined by its number of residents p_i and a number of vehicles v_i to be evacuated computed from p_i. A set **S** of m shelters is defined. Each shelter $\mathscr{S}_j, j \in \{1,\ldots,m\}$ is defined by its capacity of vehicles c_j. Each building $\mathscr{B}_i$ is associated to one or more shelters and we denote by $\mathscr{H}_i$ the set of shelters assigned to $\mathscr{B}_i$ and by $v_{i,j}$ and $\eta_{i,j}$ successively, the number of vehicles to be evacuated and the dimension of the set of paths $\mathscr{K}_{i,j}$ from $\mathscr{B}_i$ to $\mathscr{S}_j \in \mathscr{H}_i$. Finally a transportation network among buildings and shelters is considered. Let $\mathscr{G} = (\mathscr{N},\mathscr{A})$ be the graph representing the network. As previously mentioned, network nodes set $\mathscr{N}$ is decomposed into three different sets, set of building nodes $\mathscr{N}_b$, shelter nodes $\mathscr{N}_s$ and crossing nodes only $\mathscr{N}_c$. We define $\hat{\phi}_l, l \in$ A, the capacity (maximal flow rate) of the road link l expressed in vehicles per unit of time (veh/h, etc.). Moreover the problem of multiple arcs between two nodes (u,v) is solved by creating for each multiple arc a virtual new node w and two virtual new arcs (u,w) and (w,v). Also the problem of reverse arcs is solved by the same principle. Capacity or maximal flow rate of each network arc is computed by a polynomial traffic model such as no congestion is allowed. We represent the k-th-path of dimension q connecting $\mathscr{B}_i$ with $\mathscr{S}_j$ by the vector of arcs $\mathrm{A}rc^k_{i,j}$=$(e,\ldots l,\ldots y) \in \mathscr{A}^q$. As the graph $\mathscr{G}$ corresponds to the specific paths hat have been determined among buildings and shelters, then the next equality can be proved evidently :

$$\cup_{i=1}^{n} \cup_{\mathscr{S}_j \in \mathscr{H}_i} \cup_{k=1}^{\eta_{i,j}} \mathrm{A}rc^k_{i,j} = \mathscr{A} \quad (1)$$

We define a new graph $\mathscr{G}^* = (\mathscr{N}^*,\mathscr{A}^*)$ extended from $\mathscr{G}$ and which includes in addition to $\mathscr{N}$ and $\mathscr{A}$, two virtual nodes $\{s,p\}$, a sub-set of virtual nodes **M** of dimension $\sum_{i=1}^{n} |\mathscr{H}_i|$ including one node $\mathscr{M}_{j,i}$ by each safety point $\mathscr{S}_j$ associated to each building $\mathscr{B}_i$, a sub-set of virtual arcs between s and **M** which represent the maximum numbers of vehicles that may be evacuated from buildings to safety points, a sub-set of arcs of dimension $\eta_{i,j}$ between each node $\mathscr{M}_{j,i} \in$ **M** and $\mathscr{B}_i$, and finally a sub-set of virtual arcs among shelters and p which represent the capacities of shelters in terms of vehicles.

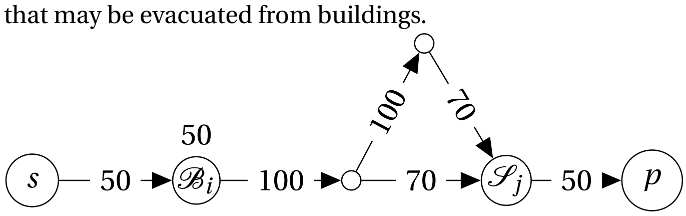
The sub-set of arcs between s and **M** is built in a manner that each shelter $\mathscr{S}_j \in \mathscr{H}_i$ associated to $\mathscr{B}_i$, $i \in \{1,\ldots,n\}$, has a virtual arc, this is the arc between s and $\mathscr{M}_{j,i}$. The weight (capacity) given for the arc $(s,\mathscr{M}_{j,i})$ is the maximum number that may be evacuated from building $\mathscr{B}_i$ to shelter $\mathscr{S}_j$ associated. This weight is the minimum value between $r_{i,j}$ (remaining vehicles to be evacuated from $\mathscr{B}_i$ to $\mathscr{S}_j$) and $\hat{\mathscr{K}}_{i,j}$ (the capacity of paths connecting $\mathscr{B}_i$ with $\mathscr{S}_j$) (see the first procedure in algorithm 2, page 7).

In the new graph $\mathscr{G}^*$, **M** is necessary firstly to define the maximum number of vehicles that may be evacuated from buildings at each time slot and secondly to know the number of vehicles evacuated on paths connecting buildings to shelters associated. The figure 1 (cases 1, 2 and 3) shows the need of virtual nodes (s and p) and the set of nodes **M** while the figure 2 illustrates the new graph $\mathscr{G}^*$. It should be noted that in the figure 2 the weights on the arcs $(s,\mathscr{M}_{j,i})$ correspond to capacity, while those on the edges $(\mathscr{M}_{j,i},\mathscr{B}_i)$ correspond to the flow.



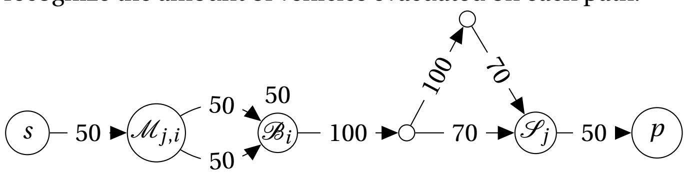
a: Case 1

Arc $(s,\ \mathscr{B}_i)$ is necessary in order to restrict the maximum number of vehicles that may be evacuated from $\mathscr{B}_i$ to $\mathscr{S}_j$. Absence of this arc leads to evacuate 70 vehicles (capacity of path) from $\mathscr{B}_i$. For $\mathscr{S}_j$, an arc $(\mathscr{S}_j,\ p)$ is necessary for respecting the capacity of $\mathscr{S}_j$. It should be noted that the capacity of each shelter is respected during the assignment phase of shelters to buildings and therefore the capacities of shelters used here are in order to limit the appearance of a problem of non-compliance to initial destinations (recurring problem in some algorithms). These capacities are varying with time - depending on the maximum number of vehicles that may be evacuated from buildings.



b: Case 2

The arc $(s,\ \mathscr{B}_i)$ is not sufficient in case of presence of several paths between a building and a shelter and especially when the first arcs of these paths are identical. In this case it is difficult to recognize the amount of vehicles evacuated on each path.



c: Case 3

FIGURE 1: Graph construction

4. Problem formulation

An evacuation plan is intended to assign for each building the departure times and the evacuation routes to be taken. It is based on the definition of two pairs of concepts (time slot, flow) and (time interval, flow rate). The first pair is built chronologically from a priorities list by evacuating at each time slot the maximal number of vehicles from each eligible building. A building is called eligible if it has the highest priority and is not completely evacuated. In a complementary way, the second pair includes the departure and arrival times computed based on the determined flow in the network.

In this paper, we focus on the first pair of concept (time slot, flow) by addressing the following objectives :

- Maximization of flow at each time slot
- Compliance with evacuation priorities list established
- Observance to evacuation paths predetermined
- Compliance to evacuation destinations
- Minimization of number of evacuation time slots for each building
- Sequence of evacuation time slots of each building (non-preemption)

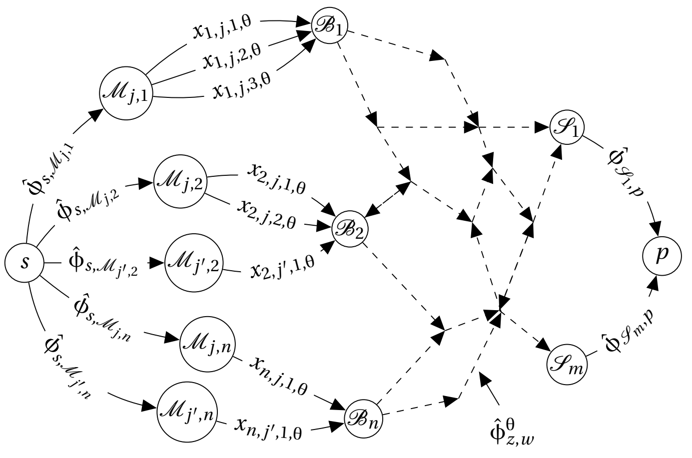


FIGURE 2: Graph $\mathcal{G}^* = (\mathcal{N}^*, \mathcal{A}^*)$

Notation :

- i : an index for buildings (population sources) $\mathcal{B}_i$
- v_i : number of vehicles of $\mathcal{B}_i$
- j : an index for shelters (destinations) $\mathcal{S}_j$
- c_j : capacity of $\mathcal{S}_j$
- k : an index for k-th best path
- θ : an index for time slot
- $\hat{\phi}^{\theta}_l$: constant capacity of arc l at time slot θ
- $\bar{\phi}_l$: variable capacity of arc l at time slot θ
- l : road link (arc)
- ϕ^{θ}_l : flow on arc l coming from all buildings sharing this arc at time slot θ
- $\overrightarrow{\phi}^{\theta}_l$: flow on arc l coming from some buildings ($\subseteq$ all buildings) sharing this arc at time slot θ
- $\mathcal{M}_{j,i} : \mathcal{B}_i \to \mathcal{S}_j$ ($\mathcal{S}_j$ is assigned to $\mathcal{B}_i$)

Decision variables :

- $x_{i,j,k,\theta}$: number of vehicles evacuated from building $\mathcal{B}_i$ to safety point $\mathcal{S}_j$ using the k-th path connecting $\mathcal{B}_i$ and $\mathcal{S}_j$ at time slot θ
- $\phi^{\theta}_{v,z}$: number of vehicles crossing the road (v,z) at time slot θ

Auxiliary variables :

- $e_{i,\theta} = \begin{cases} 1 & \text{if } \mathcal{B}_i \text{ is effectively evacuated, completely} \\ & \text{or partially, at time slot } \theta \\ 0 & \text{otherwise} \end{cases}$

Parameters :

- $\beta_{i,j} = \begin{cases} 1 & \text{if } \mathcal{B}_i \text{ is connected to } \mathcal{S}_j \\ 0 & \text{otherwise} \end{cases}$
- $\alpha_{i,j,k,l} = \begin{cases} 1 & \text{if } l \in \mathrm{A}rc^k_{i,j} \\ 0 & \text{otherwise} \end{cases}$

Constraints :

- Conservation of flows :

$$\forall \theta \in \{1,\ldots,\Theta\}, \forall z \in \mathcal{N}^* \setminus \{s,p\}, \sum_{v \in pred(z)} \Phi^{\theta}_{v,z} = \sum_{w \in succ(z)} \Phi^{\theta}_{z,w} \quad (2)$$

- Capacity on the arcs of network :

$$\forall \theta \in \{1,\ldots,\Theta\}, \forall (v,z) \in \mathcal{A}^*, \Phi^{\theta}_{v,z} \leq \hat{\Phi}_{v,z} \quad (3)$$

- Shelters capacity :

$$\forall j \in \{1,\ldots,m\}, \sum_{\theta=1}^{\Theta} \sum_{i=1}^{n} \sum_{k=1}^{\eta_{i,j}} x_{i,j,k,\theta} = c_j \quad (4)$$

- Compliance to predetermined routes :

$$\forall \theta \in \{1,\ldots,\Theta\}, \sum_{i=1}^{n} \sum_{\mathcal{S}_j \in \mathcal{H}_i} \sum_{k=1}^{\eta_{i,j}} x_{i,j,k,\theta} \cdot \alpha_{i,j,k,l} = \Phi^{\theta}_l \quad , \quad \forall l \in \mathcal{A} \quad (5)$$

The first objective function in this model is to minimize the number of time slots Θ by maximizing the number of vehicles evacuated at each time slot $\theta \in \{1,\ldots,\Theta\}$. This function can be represented by :

$$\mathcal{F}_v = \sum_{i=1}^{n} \sum_{\mathcal{S}_j \in \mathcal{H}_i} \sum_{k=1}^{\eta_{i,j}} x_{i,j,k,\theta} \quad (6)$$

5. Flow routing optimization

The first method developed here to meet the objectives mentioned in the previous section involves a greedy heuristic.

5.0.1. *Greedy algorithm*

Given a priority list of buildings $\{\mathcal{B}_1,\ldots,\mathcal{B}_n\}$ to be evacuated, this algorithm used to assign at each time slot θ, by order of priority, the maximum number of vehicles from each building $\mathcal{B}_i \in \{\mathcal{B}_1,\ldots,\mathcal{B}_n\}$ not yet totally evacuated. The assignment of flow from each building $\mathcal{B}_i$ is performed by order of paths $\mathcal{K}_{i,j}$ linking $\mathcal{B}_i$ to $\mathcal{S}_j$. This flow is the minimum value between the residual number of vehicles $r_{i,j}$ to be evacuated from $\mathcal{B}_i$ to $\mathcal{S}_j$ and the residual capacity of these paths (lines 11-24, algorithm 2). At each time slot, the greedy algorithm starts again from the first building $\mathcal{B}_{i^*}$ (lines 25-27) not yet totally evacuated at previous time slots. This algorithm stops (lines 33-35) when all vehicles from all buildings are already assigned i.e. the sum of residual vehicles of buildings is equal zero. This algorithm derives two main results : firstly the number of time slots Θ and secondly the vector $\mathbf{x} = \{x_{i,j,k,\theta} / \forall 1 \leq i \leq n, \forall \mathcal{R}_j \in \mathcal{H}_i, \forall 1 \leq k \leq \eta_{i,j}, \forall 1 \leq \theta \leq \Theta\}$, where $x_{i,j,k,\theta}$ contains the number of vehicles evacuated from $\mathcal{B}_i$ to $\mathcal{S}_j$ through $\mathrm{A}rc^k_{i,j}$ at time slot θ.

A sample evacuation network is given by the figure 3. In this network we consider 4 buildings to be evacuated, the priority list established is $\{\mathcal{B}_2, \mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3\}$. Being $\mathcal{B}_4$ must be evacuated to two shelters so an other level of priority is established and the list can be seen as $\{\mathcal{M}_{1,2}, \mathcal{M}_{1,1}, \mathcal{M}_{1,4}, \mathcal{M}_{2,4}, \mathcal{M}_{2,3}\}$. The input of this network is given in figure 4.

The figure 5 shows the evacuation scheduling of the set $\{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$ using the greedy heuristic. The number of time slots Θ for evacuating these buildings is equal to 3, and for each time slot $\theta \in \{1,\ldots,\Theta\}$ we know $x_{i,j,k,\theta}$ the number of vehicles evacuated from $\mathcal{B}_i$ to $\mathcal{S}_j$ through the route $\mathrm{A}rc^k_{i,j}, \forall 1 \leq k \leq \eta_{i,j}$ and a priori $\sum_{k=1}^{\eta_{i,j}} x_{i,j,k,\theta}$ the number of vehicles evacuated from $\mathcal{B}_i$ to $\mathcal{S}_j$ at that time slot, $\forall 1 \leq i \leq 4, \forall \mathcal{S}_j \in \mathcal{H}_i$.

Quite fast in its execution, the greedy algorithm does not ensure the maximization of flow in the network. Its limit is

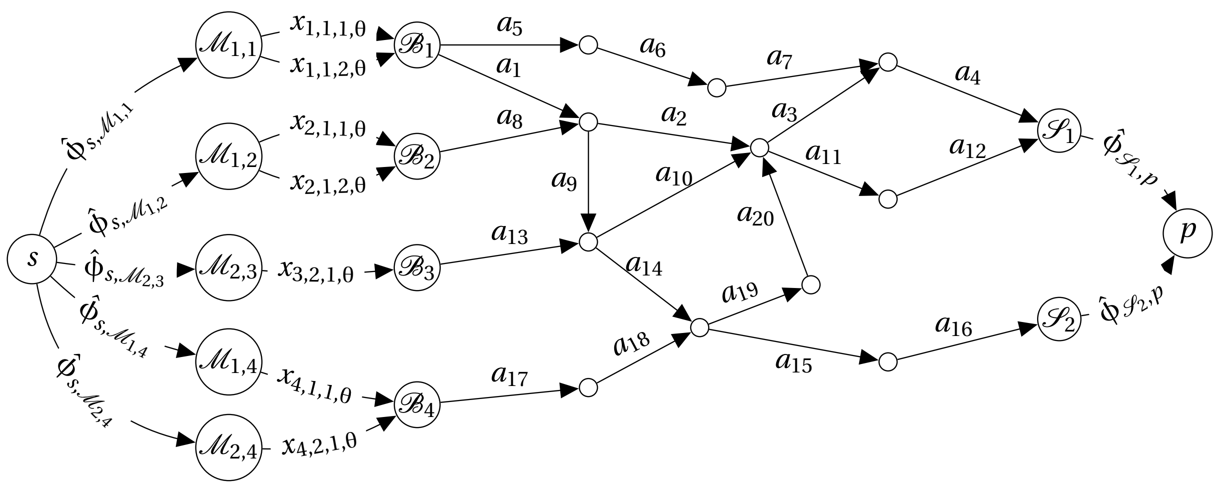


FIGURE 3: Sample evacuation network $\mathcal{G}^* = (\mathcal{N}^*, \mathcal{A}^*)$

$Arc^1_{1,1}$	$Arc^2_{1,1}$	$Arc^1_{2,1}$	$Arc^2_{2,1}$	$Arc^1_{3,2}$	$Arc^1_{4,1}$	$Arc^1_{4,2}$
a_1	a_5	a_8	a_8	a_{13}	a_{17}	a_{17}
a_2	a_6	a_2	a_9	a_{14}	a_{18}	a_{18}
a_{13}	a_7	a_3	a_{10}	a_{15}	a_{19}	a_{15}
a_4	a_4	a_4	a_{11}	a_{16}	a_{20}	a_{16}
			a_{12}		a_3	
					a_4	

TABLE 1: Paths between buildings and shelters

l	L_l (Km)	$\hat{\phi}_l$ $(veh/\frac{1}{4}h)$	l	L_l (Km)	$\hat{\phi}_l$ $(veh/\frac{1}{4}h)$
a_1	1.5	500	a_{11}	2	700
a_2	1.4	350	a_{12}	3	700
a_3	1.2	350	a_{13}	3	350
a_4	1.3	700	a_{14}	1	500
a_5	3	350	a_{15}	3	600
a_6	1.4	500	a_{16}	2	600
a_7	2	500	a_{17}	2	600
a_8	3	500	a_{18}	1.5	600
a_9	2	350	a_{19}	1	500
a_{10}	1.7	350	a_{20}	1	350

TABLE 2: Road links caracteristics

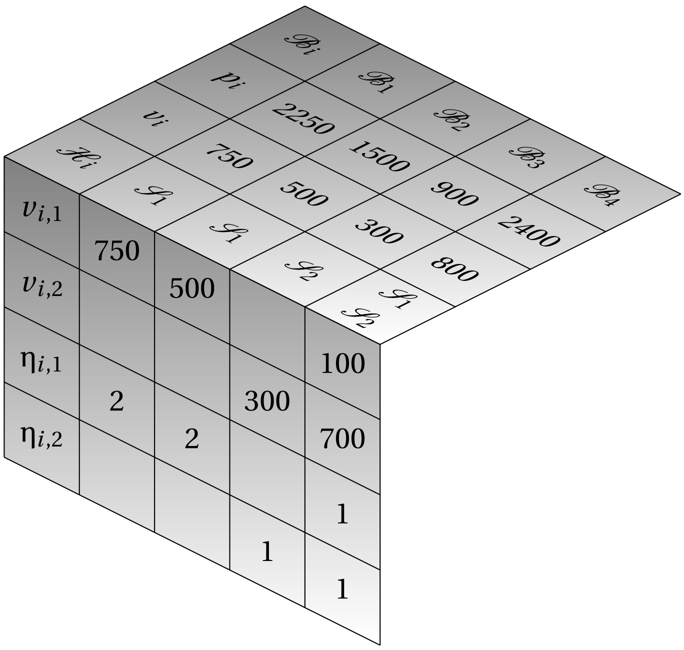


TABLE 3: Number of vehicles to be evacuated from buildings to shelters

FIGURE 4: Evacuation model input

attributed to the no use of a backward strategy i.e. once the flow is assigned it cannot be changed.

Algorithm 1 Greedy heuristic

```
1: procedure ASSIGNMENT OF FLOW
2:     Let {B_1,...,B_n} the evacuation priorities list of buildings.
3:     i* = 1
4:     θ = 0                                        ▷ Initialization of number of time slots
5:     while i* ≤ n do
6:         i = i*, modif = false, θ = θ+1            ▷ Opening of a new slot
7:         while i ≤ n do
8:             for S_j ∈ H_i do
9:                 r_{i,j} = v_{i,j} − Σ_{θ_1=1}^{θ−1} Σ_{k=1}^{η_{i,j}} x_{i,j,k,θ_1}   ▷ Residual number of vehicles
10:                if r_{i,j} > 0 then
11:                    for k = 1 to η_{i,j} do       ▷ Assignment of flow from B_i to S_j
12:                        if r_{i,j} > 0 then
13:                            r* = r_{i,j}
14:                            if Min_{φ̂−φ}(Arc^k_{i,j})-r_{i,j}>0 then
15:                                r_{i,j} = 0    ▷ Residual capacity of Arc^k_{i,j} is greater than r_{i,j}
16:                            else
17:                                r_{i,j} = r_{i,j} - Min_{φ̂−φ}(Arc^k_{i,j})
18:                            end if
19:                            x_{i,j,k,θ} = r* − r_{i,j}   ▷ Assigned flow to path Arc^k_{i,j} at slot θ
20:                            for l ∈ Arc^k_{i,j} do
21:                                φ^θ_l = φ^θ_l + x_{i,j,k,θ}   ▷ Increase the flow crossing arc l at slot θ
22:                            end for
23:                        end if
24:                    end for
25:                    if r_{i,j} > 0 and modif = false then
26:                        i* = i   ▷ First building not completely evacuated at slots {1,...,θ}
27:                        modif = true
28:                    end if
29:                end if
30:            end for
31:            i = i+1
32:        end while
33:        if modif = false then                     ▷ Stop condition
34:            i* = i
35:        end if
36:    end while
37: end procedure
```

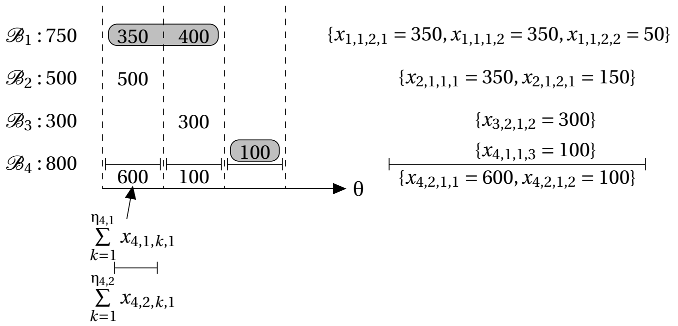


FIGURE 5: Evacuation scheduling using greedy algorithm

This algorithm seeks to evacuate people, to some extent, fairly by the assignment of vehicles from each building to the first path then to the second, etc. This fairness is the consequence of the high probability to evacuate each building on the first best path, then on the second one, etc.

5.0.2. Maximum flow approach

To fill the gap of the previous approach, namely non-maximization of flow, the push-relabel algorithm that maximizes the flow in the network between an origin and a destination is used. This algorithm is integrated in a transshipment approach which takes into account not only the capacity of network, but in addition the number of vehicles to be evacuated from origins and the safe points capacities. At each time slot in this approach we activate only the arcs corresponding to buildings not yet totally evacuated. The capacity of each activated arc corresponds to the maximum number of vehicles that may be evacuated from a building to the associated safe point. This maximum number represented in the graph (figure 2) by $\hat{\phi}_{s,\mathscr{M}_{j,i}}$ is equal to the minimum value between $r_{i,j}$ and $\hat{\mathscr{K}}_{i,j}$ the capacity of paths connecting $\mathscr{B}_i$ to $\mathscr{S}_j$. Also the maximum number of vehicles that may be evacuated through $Arc^k_{i,j}$ (and which is represented by $\hat{\phi}^k_{\mathscr{M}_{j,i},\mathscr{B}_i}$ in the graph) is equal to the minimum value between $r_{i,j}$ and $Min_{\hat{\phi}}Arc^k_{i,j}$ the capacity of k-th path (Lines 24-28). The capacity of each shelter at each time slot corresponds to the sum of the maximum numbers of vehicles that may be evacuated (depending on network capacity) from all buildings associated to this shelter (lines 30-32).

The transshipment approach as we will show (see tables 4 and 5) does not respect neither paths nor destinations. It is therefore subjected simultaneously to the following constraints :

- The number of vehicles to be evacuated from each building is not necessarily equal to the remaining number of vehicles of this building but takes the value of :

  $$\hat{\phi}_{s,\mathscr{M}_{j,i}} = Min(r_{i,j},\ \ \hat{\mathscr{K}}_{i,j})$$

  We shall see later (see table 4) this constraint does not completely prevent the problem of non-compliance of paths.
- At each time slot, we build a sub-graph containing only paths of buildings not yet completely evacuated.

  $$G^{\theta} = \cup_{i=1}^{n} \cup_{\mathscr{S}_j \in \mathscr{H}_i} ((\cup_{k=1}^{\eta_{i,j}} Arc^k_{i,j}) * (\left\lceil \frac{\sum_{\theta'=1}^{\theta-1} \sum_{k=1}^{\eta_{i,j}} x_{i,j,k,\theta'}}{v_{i,j}} \right\rceil)),$$

  where $(\cup_{k=1}^{\eta_{i,j}} Arc^k_{i,j}) * 0 = \{\emptyset\}$.

  This constraint ensures, in some extent, that each building will be evacuated through its own minimum K-paths. The step of constructing the sub-graph is not illustrated in the algorithms for reasons of simplification.
- The capacity of each safe point is function of remaining vehicles to be evacuated.

  $$\hat{\phi}_{\mathscr{S}_j,p} = \sum_{i=1}^{n} \hat{\phi}_{s,\mathscr{M}_{j,i}} \qquad \forall \mathscr{B}_i \in \mathbf{B}_{ext}, \mathscr{S}_j \in \mathscr{H}_i$$

  where $\mathbf{B}_{ext} \subseteq \mathbf{B}$ is the set of buildings not yet completely evacuated.

The push-relabel algorithm [2] is called at each time slot in this approach. The result of this approach is the vector $\mathbf{x} = \{x_{i,j,k,\theta} / 1 \le i \le n, \forall \mathscr{R}_j \in \mathscr{H}_i, 1 \le k \le \eta_{i,j}, 1 \le \theta \le \Theta\}$ (Lines 36-43). This algorithm terminates (line 44) when all vehicles from all buildings were all evacuated.

The figure 6 shows the evacuation scheduling of the set of buildings $\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4\}$ using the maximum flow approach. The number of time slots Θ for evacuating these buildings is reduced to 2. At each time slot $\theta \in \{1, \ldots, \Theta\}$ we know $x_{i,j,k,\theta} =$

2. http ://www.boost.org/

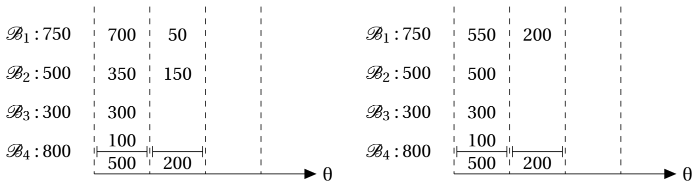
Algorithm 2 Maximum flow approach

```
1:  procedure CAPACITY OF K-BEST PATHS (I,J)
2:      cφ̄_l = φ̄_l, ∀k ∈ {1,···,η_{i,j}}, ∀l ∈ Arc^k_{i,j}          ▷ Copy the variable capacity
3:      𝒦_{i,j} = 0                 ▷ Capacity of paths between ℬ_i and 𝒮_j initialized to zero
4:      for k = 1 to η_{i,j} do                            ▷ All roads that connect ℬ_i to 𝒮_j
5:          y = Min_φ̄ Arc^k_{i,j}                        ▷ Variable capacity of path Arc^k_{i,j}
6:          𝒦̂_{i,j} = 𝒦_{i,j} + y
7:          for l ∈ Arc^k_{i,j} do
8:              φ̄_l = φ̄_l − y   ▷ Remove the variable capacity of path Arc^k_{i,j} from all its arcs
9:          end for
10:     end for
11:     φ̄_l = cφ̄_l, ∀k ∈ {1,···,η_{i,j}}, ∀l ∈ Arc^k_{i,j}         ▷ Reset the variable capacity
12: end procedure

13: procedure MAXIMIZATION OF FLOW
14:     Let {ℬ_1,...,ℬ_n} the evacuation priority list of buildings
15:     r_{i,j} = v_{i,j}
16:     θ = 0                                          ▷ Initialization of number of slots

17:     repeat
18:         θ = θ + 1                                                ▷ Opening a new slot
19:         φ̄_l = φ̂^θ_l, ∀l ∈ 𝒜                                 ▷ Copy the constant capacity
20:         for i = 1 to n do
21:             for 𝒮_j ∈ ℋ_i do
22:                 if r_{i,j} > 0 then
23:                     φ̂_{s,ℳ_{j,i}} = Min{r_{i,j}, 𝒦̂_{i,j}}     ▷ Compute the capacity of arc (s, ℳ_{j,i})
24:                     for k = 1 to η_{i,j} do
25:                         φ̂^k_{ℳ_{j,i},ℬ_i} = Min{r_{i,j}, Min_φ̂(Arc^k_{i,j})}   ▷ Capacity of (ℳ_{j,i}, ℬ_i)^k
26:                     end for
27:                 else
28:                     φ̂_{s,ℳ_{j,i}} = 0
29:                 end if
30:             end for
31:         end for
32:         for j = 1 to m do
33:             φ̂_{𝒮_j,p} = Σ_{i=1}^{n} φ̂_{s,ℳ_{j,i}}            ▷ Compute the capacity of each shelter
34:         end for
35:         Boost.push_relabel_max_flow()                          ▷ Calling of push − relabel
36:         for i = 1 to n do
37:             for 𝒮_j ∈ ℋ_i do
38:                 for k = 1 to η_{i,j} do
39:                     x_{i,j,k,θ} = φ^k_{ℳ_{j,i},ℬ_i}
40:                 end for
41:                 r_{i,j} = r_{i,j} − φ_{s,ℳ_{j,i}}
42:             end for
43:         end for
44:     until Σ_{i=1}^{n} Σ_{𝒮_j∈ℋ_i} r_{i,j} = 0
45: end procedure
```

$\phi^k_{\mathcal{M}_{j,i},\mathcal{B}_i}$ the number of vehicles evacuated by path $Arc^k_{i,j}$, $1 \leq k \leq \eta_{i,j}$ and $\phi_{s,\mathcal{M}_{j,i}}$ the number of vehicles evacuated from $\mathcal{B}_i$ to $\mathcal{S}_j$ at this time slot, $1 \leq i \leq 4, \mathcal{S}_j \in \mathcal{H}_i$.

Given the automatic construction of paths according to the numbering of nodes, the maximum flow algorithm (push-relabel) does not necessarily observe the priority of each building. For example, the figure 6 shows two results for two different numbering of nodes of the graph in figure 2. Recall that the priorities list is $\{\mathcal{B}_2, \mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3\}$. It should be noted that the maximum flow assigned by push-relabel in the graph is independent of numbering of nodes.

Furthermore, constraints (lines 23, 32-34, algorithm 2) does not completely evacuate the non-compliance to predefined paths and destinations by push-relabel (automatic construction of paths as described above). Figure 6-b and table 4 show an example of non-compliance to paths ($\mathcal{B}_4 \to \mathcal{S}_1$) and to destinations ($\mathcal{B}_3 \to \mathcal{S}_1$). Indeed, push-relabel built two new paths in order to maximize the flow : the first one $\{a_{17}, a_{18}, a_{19}, a_{20}, a_{11}, a_{12}\}$ which allowed the evacuation of 100 vehicles from $\mathcal{B}_4$ to $\mathcal{S}_1$ and the second path $\{a_{13}, a_{14}, a_{19}, a_{20}, a_{11}, a_{12}\}$ which formed, for 150 vehicles from $\mathcal{B}_3$, the crossing path towards $\mathcal{S}_1$ (see table 4).



a: Priority not respected b: Priority respected

FIGURE 6: Maximum flow approach

l	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}
$\hat{\phi}_l$	500	350	350	700	350	500	500	500	350	350	700	700	350	500	600	600	600	600	500	350
$\mathcal{B}_2 \to \mathcal{S}_1$								150	0	0	350	350								
$\mathcal{B}_2 \to \mathcal{S}_1$		200	200	550				0												
$\mathcal{B}_1 \to \mathcal{S}_1$	300	0	0	350																
$\mathcal{B}_1 \to \mathcal{S}_1$				0	0	150	150													
$\mathcal{B}_4 \to \mathcal{S}_1$											250	250					500	500	400	250
$\mathcal{B}_4 \to \mathcal{S}_2$															100	100	0	0		
$\mathcal{B}_3 \to \mathcal{S}_1$											100	100	200	350					250	100
$\mathcal{B}_3 \to \mathcal{S}_2$													50	200	50	50				

TABLE 4: Flow maximum approach : Assignment of flow at first time slot (subjected to capacity constraint of safety points)

The table 5 applied to the same example, shows how the flow may be increased of 100 units from $\mathcal{B}_3$ ($\mathcal{B}_3 \to \mathcal{S}_1$) if the constraints on the capacities of shelters (lines 32-34, algorithme 2) are relaxed (assigned an infinite value).

l	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}
$\hat{\phi}_l$	500	350	350	700	350	500	500	500	350	350	700	700	350	500	600	600	600	600	500	350
$\mathcal{B}_2 \to \mathcal{S}_1$								150	0	0	350	350								
$\mathcal{B}_2 \to \mathcal{S}_1$		200	200	550				0												
$\mathcal{B}_1 \to \mathcal{S}_1$	300	0	0	350																
$\mathcal{B}_1 \to \mathcal{S}_1$				0	0	150	150													
$\mathcal{B}_4 \to \mathcal{S}_1$											250	250					500	500	400	250
$\mathcal{B}_4 \to \mathcal{S}_2$															100	100	0	0		
$\mathcal{B}_3 \to \mathcal{S}_1$											0	0	100	250					150	0
$\mathcal{B}_3 \to \mathcal{S}_2$													50	200	50	50				

TABLE 5: Flow maximum approach : Assignment of flow at first time slot (capacity constraint of safety points is relaxed)

5.0.3. *Maximum flow per-blocks algorithm*

Having regard to the random nature of the adaptation of push-relabel algorithm based on numbering of nodes (see example figure 6), and in order to ensure compliance with the evacuation priority, an other track is to maximize per blocks the flow between buildings and safety points. This new algorithm is here coupled with push-relabel. However, it can be coupled with any other algorithm maximizing the flow in the network (for example, linear program CPLEX).

We define the sub-graph $\mathcal{G}^*_{\theta,i} = \{\cup_{k=1}^{\eta_{i',j}} \;/\; r_{i',j} > 0, \forall i' \leq i, \forall \mathcal{S}_j \in \mathcal{H}_{i'}\}$ which corresponds to the network connecting only the building $\mathcal{B}_i$ and the priority buildings to their destinations. At each step in each time slot θ, we build $\mathcal{G}^*_{\theta,i}$ tagging the capacity of the arc $(s, \mathcal{M}_{j,i})$ with the maximum number of vehicles that may be evacuated from the eligible building $\mathscr{B}_i$ to $\mathscr{S}_j$. As we mentioned before, this maximum number represented in the graph (see figure 3) by $\hat{\phi}_{s,\mathscr{M}_{j,i}}$ is equal to the minimum value between $r_{i,j}$ and $\hat{\mathscr{K}}_{i,j}$. Also the maximum number of vehicles that may be evacuated through $Arc^k_{i,j}$ (and which is represented by $\hat{\phi}^k_{\mathscr{M}_{j,i},\mathscr{B}_i}$ in the graph) is equal to the minimum value between $r_{i,j}$ and $Min_{\hat{\phi}} Arc^k_{i,j}$ the capacity of k-th path. In this approach per-blocks, it should be noted that the value of $\hat{\mathscr{K}}_{i,j}$ is not constant as it depends on variations of arcs capacities $\bar{\phi}_l$ (capacity which may take down values at each time slot).

The construction of a block of buildings is based on the principle of assigning the maximal number of vehicles from the current building while keeping the maximal number of vehicles assigned from all buildings previously treated (highest priorities) (Lines 33-36, algorithm 3). This principle allows the observation of evacuation priorities of buildings and its violation is detected when the sum of vehicles assigned from previous buildings and current building is less than the sum of maximum numbers of vehicles to be removed from these buildings (Lines 18-21, algorithm 3). In a violation, the allocation of the maximum number of vehicles from the current building keeping the maximal flow from previous buildings (line 18, algorithm 3) is impossible (case of $\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4\}$, figure 7). Therefore, we return to the last successful configuration ($\mathbf{a}^*$, line 19, algorithm 3) which then becomes a closed block, namely a block consisting only of buildings with maximal flows (see bloc $\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3\}$, figure 7).

The next step is, first, to subtract the flow of closed block ($\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3\}$, figure 7) from the residual capacity of arcs (line 21, algorithm 3), and, second, to assign the flow from current building ($\mathscr{B}_4$, figure 7) by the greedy algorithm (lines 22-29, algorithm 3). It should be noted that for the first closed block of buildings, the residual capacities ($\bar{\phi}$) of arcs coincide with the initial capacities ($\hat{\phi}$). Flow assigned from current building is obviously equal to residual capacity of K-best paths connecting this building to the safe point associated. The same procedure is repeated for the following buildings in order to build a new block(s) up to the last building in the priorities list. Therefore an additional property is considered here for the construction of $\mathscr{G}^*_{\theta,i}$. This latter contains only buildings present in current block.

We repeat this construction procedure of blocks (see figure 7) by opening a new time slot and resetting the capacities of arcs (line 4, algorithm 3) until the evacuation of all buildings ($back = false$, algorithm 3).

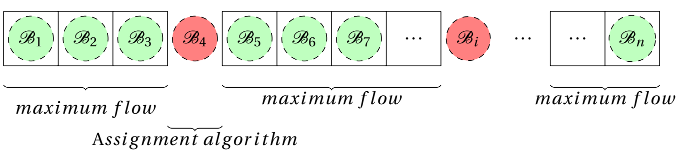


FIGURE 7: Evacuation of buildings at a given time slot according to the Maximum flow per-blocks approach

Algorithm 3 Maximum flow per-blocks approach (with priority)

1: **procedure** FIND MAXIMUM FLOW PER-BLOCKS
2: Let $\{\mathscr{B}_1, \ldots, \mathscr{B}_n\}$ the established priority list of buildings to evacuate.
3: **for** $i = 1$ **to** n **do**
4: $\theta = \theta + 1$, $\bar{\phi}_l = \hat{\phi}_l$, $\forall l \in \mathscr{A}$ ▷ Variable capacity = constant capacity
5: **while** $i \leq n$ **do**
6: **for** $\mathscr{S}_j \in \mathscr{H}_i$ **do**
7: **if** $r_{i,j} > 0$ **then**
8: **if** !entry **then**
9: $i^+ = i, entry = true$
10: **end if**
11: $\mathscr{Q}.add(\mathscr{S}_j)$ ▷ Queue
12: $\bar{\phi}^k_{\mathscr{M}_{j,i},\mathscr{B}_i} = Min\{r_{i,j}, Min_{\bar{\phi}} Arc^k_{i,j}\}, \forall k \in \{1, \ldots, \eta_{i,j}\}$
13: $\bar{\phi}_{s,\mathscr{M}_{j,i}} = Min\{r_{i,j}, \hat{\mathscr{K}}_{i,j}\}$ ▷ Maximum number of vehicles
14: $\bar{\phi}_{\mathscr{S}_{j'},p} = \sum_{i'=i^+}^{i} \bar{\phi}_{s,\mathscr{M}_{j',i'}}, \forall j' \in \{1, \ldots, m\}$ ▷ Shelters capacity
15: $\mathrm{SUM} = \mathrm{SUM} + \bar{\phi}_{s,\mathscr{M}_{j,i}}$
16: $\mathbf{a} = Boost.push\_relabel\_max\_flow()$
17: $\mathrm{FLOW} = \sum_{i'=i^+}^{i-1} \sum_{\mathscr{S}_{j'} \in \mathscr{H}_{i'}} \phi_{s,\mathscr{M}_{j',i'}} + \sum_{\mathscr{S}_{j'} \in \mathscr{Q}} \phi_{s,\mathscr{M}_{j',i}}$
18: **if** FLOW < SUM **then** ▷ Priority may not be respected
19: $\mathbf{a} = \mathbf{a}^*$ ▷ ϕ^θ_l : flow incoming from all buildings
20: $\phi^\theta_l = \phi^\theta_l + \overrightarrow{\phi}_l, \forall l \in \mathscr{A}$ ▷ $\overrightarrow{\phi}_l$: flow from some buildings
21: $\bar{\phi}_l = \bar{\phi}_l - \overrightarrow{\phi}_l, \forall l \in \mathscr{A}$ ▷ Modify the capacity of arcs
22: **for** $k = 1$ to $\eta_{i,j}$ **do** ▷ Assignment algorithm
23: $r_{i,j} = r_{i,j}$ - $\mathrm{Min}_{\bar{\phi}} Arc^k_{i,j}$
24: $x_{i,j,k,\theta} = Min_{\bar{\phi}} Arc^k_{i,j}$
25: **for** $l \in Arc^k_{i,j}$ **do**
26: $\phi^\theta_l = \phi^\theta_l + Min_{\bar{\phi}} Arc^k_{i,j}$
27: $\bar{\phi}_l = \bar{\phi}_l - Min_{\bar{\phi}} Arc^k_{i,j}$
28: **end for**
29: **end for**
30: $\bar{\phi}_{s,\mathscr{M}_{j',i'}} = 0, \forall i' \in \{i^+, \ldots, i-1\}, \forall \mathscr{S}_{j'} \in \mathscr{H}_{i'}$
31: $\bar{\phi}_{s,\mathscr{M}_{j',i}} = 0, \forall \mathscr{S}_{j'} \in \mathscr{Q}$ ▷ Close all opened arcs
32: $entry = false$, SUM = 0
33: **else**
34: $\mathbf{a}^* = \mathbf{a}$ ▷ Successful allocation
35: $x_{i,j,k,\theta} = \bar{\phi}^k_{\mathscr{M}_{j,i},\mathscr{B}_i}, \forall k \in \{1, \ldots, \eta_{i,j}\}$
36: $r_{i,j} = r_{i,j} - \bar{\phi}_{s,\mathscr{M}_{j,i}}$ ▷ Subtract residual number
37: **end if**
38: **if** $r_{i,j} > 0$ **then**
39: **if** !$back$ **then**
40: $i^\# = i, back = true$
41: **end if**
42: **end if**
43: **end if**
44: **end for**
45: $\mathscr{Q}.clear()$ ▷ Clear queue
46: **end while**
47: **if** $back$ **then**
48: $\bar{\phi}_{s,\mathscr{B}_{j,i'}} = 0, \forall i' \in \{i^+, \ldots, i\}, \forall \mathscr{S}_j \in \mathscr{H}_{i'}$ ▷ Close all opened arcs
49: $back = entry = false$, $i = i^\# - 1$
50: **end if**
51: **end for**
52: **end procedure**

In summary, maximum flow per-blocks algorithm, described above, allows to maximize the flow per-blocks while respecting the priorities list of buildings.

Note that the maximal flow that can be evacuated from a building at a given time slot is the minimum between residual number of vehicles and capacity (residual capacity in this approach) of K-best paths. Recall also that the maximum flow by-blocs approach guarantees the evacuation priorities of buildings by constructing blocks of maximal flows (see figure 7). Those blocks are isolated from the view point of assignment. In other words, at each time slot, there is no relationship exists between the current block under construction (bloc $\{\mathscr{B}_5, \mathscr{B}_6, \mathscr{B}_7, \ldots\}$, figure 7) and the previous blocks built by the algorithm (bloc $\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3\}$, figure 7). Therefore the maximization of flow in the current block is independent of the flows of previous blocks : the flows of the previous blocks are already assigned and can not be changed. In addition, it is independent of the allocation of flow, carried out by the greedy heuristic, from each building violating the property of maximal flow in previous blocks (case of $\mathscr{B}_4$, figure 7). On the other hand, the assignment by the greedy heuristic of the flow from a given building (case of $\mathscr{B}_i$, figure 7) does not depend neither on maximum flows of closed blocks (previously constructed, blocks $\{\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3\}$ and $\{\mathscr{B}_5, \mathscr{B}_6, \mathscr{B}_7, \ldots\}$, figure 7) nor on the flows previously assigned by the same heuristic ($\mathscr{B}_4$, figure 7).

Thus, the maximum flow per-blocks approach can cause a loss of flow in the network at certain time slots, which is illustrated in the figure 8.

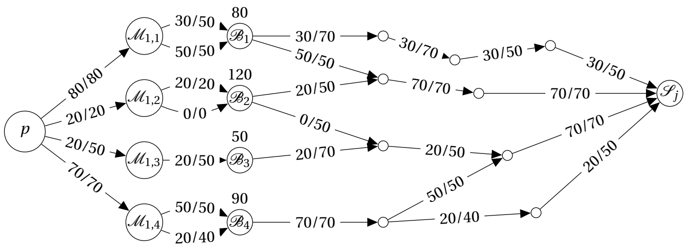


FIGURE 8: Loss of flow using maximum flow per-blocks algorithm ; weight on arcs : flow/capacity

Let $\{\mathscr{B}_i, 1 \le i \le 4\}$ be a set of buildings to be evacuated and $\{\mathscr{B}_1, \mathscr{B}_4, \mathscr{B}_3, \mathscr{B}_2\}$ the priority list established. In this example the Max-Fw (maximal flow) of vehicles from $\mathscr{B}_1$ is assigned to the two paths connecting $\mathscr{B}_1$ to the shelter associated $\mathscr{S}_j$, with 50 vehicles through the first path and 30 vehicles through the second one (see figure 8 ; step 1, table 6). As for $\mathscr{B}_1$, Max-Fw of vehicles from $\mathscr{B}_4$ is assigned to the two paths connecting it to $\mathscr{S}_j$, with 50 vehicles through the first path and 20 vehicles through the second one (see figure 8 ; step 2, table 6). For $\mathscr{B}_3$, the Max-Fw of vehicles cannot be totally assigned : $\{\mathscr{B}_1, \mathscr{B}_4\}$ becomes now a closed block then the greedy algorithm is used to assign only 20 vehicles from $\mathscr{B}_3$ (residual capacity of unique path connecting $\mathscr{B}_3$ to $\mathscr{S}_j$) (see figure 8 ; step 3, table 6).
With the same construction strategy as for the previous block, we move to the construction of the next block with $\mathscr{B}_2$ for which the Max-Fw of 20 vehicles is assigned to the first path.

We note that an additional 20 vehicles from $\mathscr{B}_3$ would could have be evacuated transferring (pushing) 20 vehicles from the first path of $\mathscr{B}_4$ to the second one. However the maximum flow per-blocks approach does not allow this given the lack of relationship between the assignment of flow from $\mathscr{B}_3$, that has broken the maximal flow property in the block $\{\mathscr{B}_1, \mathscr{B}_4\}$, and the maximization of flow in this same block. Similarly, 20 additional vehicles from $\mathscr{B}_2$ could be evacuated shifting an equivalent number of vehicles from the first to the second path of $\mathscr{B}_1$. However, maximum flow per-blocks algorithm does not either allow this transfer given the lack of relationship between the blocks (see table 6).

The principle of maximum flow per-blocks approach stems from the need to observing the priorities of buildings : as long as the flow assigned from buildings is maximal, the priorities are met. However, when the algorithm can not assign the Max-Fw from a building keeping the maximal flows

	$\phi_{s,\mathscr{M}_{j,1}}$	$\phi_{s,\mathscr{M}_{j,2}}$	$\phi_{s,\mathscr{M}_{j,3}}$	$\phi_{s,\mathscr{M}_{j,4}}$	$\bar\phi_{s,\mathscr{M}_{j,1}}$	$\bar\phi_{s,\mathscr{M}_{j,2}}$	$\bar\phi_{s,\mathscr{M}_{j,3}}$	$\bar\phi_{s,\mathscr{M}_{j,4}}$
$\mathscr{B}_1$	80	0	0	0	80	0	0	0

a: Step 1 ; Assignment of maximal flow from $\mathscr{B}_1$ to $\mathscr{S}_j$ is succeeded

	$\phi_{s,\mathscr{M}_{j,1}}$	$\phi_{s,\mathscr{M}_{j,2}}$	$\phi_{s,\mathscr{M}_{j,3}}$	$\phi_{s,\mathscr{M}_{j,4}}$	$\bar\phi_{s,\mathscr{M}_{j,1}}$	$\bar\phi_{s,\mathscr{M}_{j,2}}$	$\bar\phi_{s,\mathscr{M}_{j,3}}$	$\bar\phi_{s,\mathscr{M}_{j,4}}$
$\{\mathscr{B}_1\}$	80	0	0	0	80	0	0	0
$\{\mathscr{B}_1, \mathscr{B}_4\}$	80	0	0	70	80	0	0	70

b: step 2 ; Assignment of maximal flow from $\mathscr{B}_4$ to $\mathscr{S}_j$ is succeeded

Block broken

Bloc 1

	$\phi_{s,\mathscr{M}_{j,1}}$	$\phi_{s,\mathscr{M}_{j,2}}$	$\phi_{s,\mathscr{M}_{j,3}}$	$\phi_{s,\mathscr{M}_{j,4}}$	$\bar\phi_{s,\mathscr{M}_{j,1}}$	$\bar\phi_{s,\mathscr{M}_{j,2}}$	$\bar\phi_{s,\mathscr{M}_{j,3}}$	$\bar\phi_{s,\mathscr{M}_{j,4}}$
$\{\mathscr{B}_1\}$	80	0	0	0	80	0	0	0
$\{\mathscr{B}_1, \mathscr{B}_4\}$	80	0	0	70	80	0	0	70
$\{\mathscr{B}_1, \mathscr{B}_4, \mathscr{B}_3\}$	80	0	50	60	80	0	50	70

c: Step 3 ; Assignment of maximal flow from $\mathscr{B}_3$ to $\mathscr{S}_j$ is failed. The algorithm assigns the maximal flow from $\mathscr{B}_3$ to $\mathscr{S}_j$ but by reducing the flow assigned from $\mathscr{B}_4$ (the value is reduced from 70 (maximal flow) to 60)

Broken block

Greedy algorithm

Block 1

	$\phi_{s,\mathscr{M}_{j,1}}$	$\phi_{s,\mathscr{M}_{j,2}}$	$\phi_{s,\mathscr{M}_{j,3}}$	$\phi_{s,\mathscr{M}_{j,4}}$	$\bar\phi_{s,\mathscr{M}_{j,1}}$	$\bar\phi_{s,\mathscr{M}_{j,2}}$	$\bar\phi_{s,\mathscr{M}_{j,3}}$	$\bar\phi_{s,\mathscr{M}_{j,4}}$
$\{\mathscr{B}_1\}$	80	0	0	0	80	0	0	0
$\{\mathscr{B}_1, \mathscr{B}_4\}$	80	0	0	70	80	0	0	70
$\{\mathscr{B}_1, \mathscr{B}_4, \mathscr{B}_3\}$	80	0	50	60	80	0	50	70
$\{\mathscr{B}_3\}$	80	0	20	70	80	0	20	70

d: Step 4 ; Assignment of flow from $\mathscr{B}_3$. This flow corresponds to residual capacity of path connecting $\mathscr{B}_3$ to $\mathscr{S}_j$ after removing from arcs the flow assigned from closed block $\{\mathscr{B}_1, \mathscr{B}_4\}$

Broken block

Greedy algorithm

Block 1

	$\phi_{s,\mathscr{M}_{j,1}}$	$\phi_{s,\mathscr{M}_{j,2}}$	$\phi_{s,\mathscr{M}_{j,3}}$	$\phi_{s,\mathscr{M}_{j,4}}$	$\bar\phi_{s,\mathscr{M}_{j,1}}$	$\bar\phi_{s,\mathscr{M}_{j,2}}$	$\bar\phi_{s,\mathscr{M}_{j,3}}$	$\bar\phi_{s,\mathscr{M}_{j,4}}$
$\{\mathscr{B}_1\}$	80	0	0	0	80	0	0	0
$\{\mathscr{B}_1, \mathscr{B}_4\}$	80	0	0	70	80	0	0	70
$\{\mathscr{B}_1, \mathscr{B}_4, \mathscr{B}_3\}$	80	0	50	60	80	0	50	70
$\{\mathscr{B}_3\}$	80	0	20	70	80	0	20	70
$\{\mathscr{B}_2\}$	80	20	20	70	80	20	20	70

Block 2

e: Step 5 ; Assignment of maximal flow from $\mathscr{B}_2$ to $\mathscr{S}_j$ is succeeded. As $\mathscr{B}_2$ is the last building in the priorities list, then the second block including only $\mathscr{B}_2$ becomes a closed block

TABLE 6: Construction of blocks by the maximum flow per-blocks algorithm

from previous buildings (highest priorities), the priority may no longer be met (the case of $\{\mathscr{B}_1, \mathscr{B}_4, \mathscr{B}_3\}$, see table 6). This violation is due to an assignment made possible by the algorithm resulting in maximal flow from the current building but impacting the maximal flows from previous buildings.

To address the loss of flow shown above, an algorithm which maximizes flow lexicographically has been developed. The following section presents this algorithm.

5.0.4. Lexicographic maximum flow

The lexicographic maximum flow algorithm is further based on the development of a new concept : lexicographic maximal flow (Lex-Max-Fw). Lex-Max-Fw from a building to a shelter, at a given time slot, is defined as the maximum number of vehicles that can be evacuated from this building without impacting the lexicographic maximal flows from buildings with higher priorities. It should be noted that the Lex-Max-Fw from a building is always less than or equal to Max-Fw from this building.

The principle of the lexicographic maximum flow algorithm is based on the determination of the Lex-Max-Fw from the current building. Initially equal to the Max-Fw of this building, Lex-Max-Fw decreases in the event of impact on the lexicographic maximal flows from previous priority buildings. Push-relabel algorithm is called for each iteration (line 8, algorithm 4) of each time slot (line 5, algorithme 4) with a number of buildings increased by one unit. The greedy algorithm becomes unnecessary in this case.

The difference with the previous approach, at each time slot, concerns the failure of the assignment of the Lex-Max-Fw (which is equal to the Max-Fw) from the current building while maintaining the Lex-Max-Fw (which are equal to Max-Fw) from previous priority buildings.

This failure in the allocation reflects the inequality between firstly, the sum of assigned flows from previous buildings and current building, and, secondly, the sum of their Lex-Max-Fw. Therefore, instead of closing the current block excluding the current building (maximum flow per-blocks algorithm), lexicographic maximum flow algorithm computes a new value of Lex-Max-Fw (certainly less than the Max-Fw) for the current building. This value is the difference between firstly, the sum of assigned flow from current building and previous buildings, and, secondly, the sum of Lex-Max-Fw of previous buildings (lines 19-21, algorithm 4). The same procedure described above is applied to the next building (line 8, algorithm 4).

The table 7 and the figure 9 illustrate the operation of lexicographic maximum flow algorithm (see figure 8 and table 6 for comparison with maximum flow per-blocks algorithm).

Recall that the difference between the maximum flow per-blocks algorithm and the lexicographic maximum flow algorithm regards the failure in the assignment of Lex-Max-Fw (which is equal to Max-Fw) from $\mathcal{B}_3$ while keeping the lexicographical maximal flows (which are equal to the maximal flows) from previous priority buildings $\{\mathcal{B}_1, \mathcal{B}_4\}$.

At this point, push-relabel assigns 80 vehicles from $\mathcal{B}_1$, 60 from $\mathcal{B}_4$ and 50 from $\mathcal{B}_3$. As the total flow assigned from $\{\mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3\}$ is less than the sum of Lex-Max-Fw of these buildings, then a new value of Lex-Max-Fw for $\mathcal{B}_3$ is computed :

$\hat{\phi}_{s,\mathcal{M}_{j,3}} = (\phi_{s,\mathcal{M}_{j,1}} + \phi_{s,\mathcal{M}_{j,4}} + \phi_{s,\mathcal{M}_{j,3}}) - (\hat{\phi}_{s,\mathcal{M}_{j,1}} + \hat{\phi}_{s,\mathcal{M}_{j,4}}) = (80 + 60 + 50) - (80 + 70) = 40$

The assignment at the next step of Lex-Max-Fw from $\mathcal{B}_2$ is not supported by the network. The sum of flow (210) assigned from buildings (current building and previous buildings) is less than the sum of Lex-Max-Fw (280) of these buildings. A new value of Lex-Max-Fw for $\mathcal{B}_2$ is then computed :

$\hat{\phi}_{s,\mathcal{M}_{j,2}} = (\phi_{s,\mathcal{M}_{j,1}} + \phi_{s,\mathcal{M}_{j,4}} + \phi_{s,\mathcal{M}_{j,3}} + \phi_{s,\mathcal{M}_{j,2}}) - (\hat{\phi}_{s,\mathcal{M}_{j,1}} + \hat{\phi}_{s,\mathcal{M}_{j,4}} + \hat{\phi}_{s,\mathcal{M}_{j,3}}) = (70 + 60 + 0 + 100) - (80 + 70 + 40) = 40$ (see table 7, figure 9).

	$\phi_{s,\mathcal{M}_{j,1}}$	$\phi_{s,\mathcal{M}_{j,2}}$	$\phi_{s,\mathcal{M}_{j,3}}$	$\phi_{s,\mathcal{M}_{j,4}}$	$\hat{\phi}_{s,\mathcal{M}_{j,1}}$	$\hat{\phi}_{s,\mathcal{M}_{j,2}}$	$\hat{\phi}_{s,\mathcal{M}_{j,3}}$	$\hat{\phi}_{s,\mathcal{M}_{j,4}}$
$\{\mathcal{B}_1\}$	80	0	0	0	80	0	0	0
$\{\mathcal{B}_1, \mathcal{B}_4\}$	80	0	0	70	80	0	0	70
$\{\mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3\}$	80	0	50	60	80	0	50	70
$\{\mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3, \mathcal{B}_2\}$	70	100	0	60	80	100	40	70
$\{\mathcal{B}_1, \mathcal{B}_4, \mathcal{B}_3, \mathcal{B}_2\}$	80	40	40	70	80	30	40	70

TABLE 7: Construction of one block by the lexicographic maximum flow algorithm

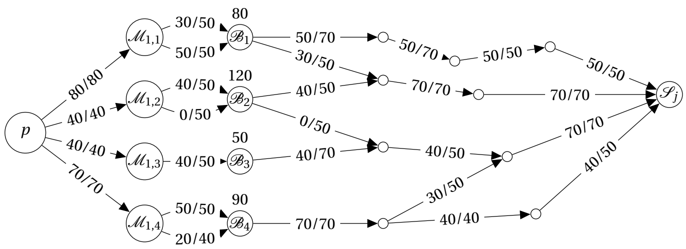


FIGURE 9: Gain of flow using lexicographic maximum flow algorithm, arcs weight : flow/capacity

The complexity of algorithm 4 can be found between the estimated best case $O[\sum_{i=1}^{n} \frac{i^2}{n^2} |\mathcal{N}||\mathcal{A}| \log(\frac{i}{n} \frac{|\mathcal{N}|^2}{|\mathcal{A}|})]$ and the exactly worst case $O[n|\mathcal{N}||\mathcal{A}|\log(\frac{|\mathcal{N}|^2}{|\mathcal{A}|})]$. In the worst case, all buildings are considered they have the same K-best paths. By against, the best case assumes that buildings don't share any arc of the network and the numbers of nodes and arcs in the K-best paths of each building are respectively equals to $\frac{|\mathcal{N}|}{n}$ and $\frac{|\mathcal{A}|}{n}$. Recall that n is the number of buildings (sources), $\mathcal{N}$ and $\mathcal{A}$ are respectively the sets of nodes and arcs in the static graph.

Given a set of n buildings to be evacuated, the lexicographic maximum flow algorithm, as we mentioned above, calls at each iteration the push-relabel algorithm with a number of buildings increased by one unit. Being that the complexity of push-relabel is of $O[|\mathcal{N}||\mathcal{A}|\log(\frac{|\mathcal{N}|^2}{|\mathcal{A}|})]$, so for the worst case the complexity is $O[n|\mathcal{N}||\mathcal{A}|\log(\frac{|\mathcal{N}|^2}{|\mathcal{A}|})] = n \cdot O[|\mathcal{N}||\mathcal{A}|\log(\frac{|\mathcal{N}|^2}{|\mathcal{A}|})]$. About the best case, the complexity of push relabel is equal to $O[\frac{i^2}{n^2}|\mathcal{N}||\mathcal{A}|\log(\frac{i}{n}\frac{|\mathcal{N}|^2}{|\mathcal{A}|})]$ at each iteration $i \in \{1, \cdots, n\}$. It should be noted that neither the best case nor the worst one is close to be possible.

The development of the lexicographic maximum flow algorithm integrating push-relabel has determined the lexicographical maximum flows of buildings according to a priority list established beforehand. In addition, the fact to taking into account the evolving of capacities of shelters (line 14, algorithm 4), at each iteration for each time slot, results in respect of the assignment of safety points (as destinations) to buildings (see figure 3 and table 9 ; see table 4 for comparison).

The non-compliance with K-best paths still present because of the difficulty of adapting the algorithm of push-relabel by the integration of the constraint of respect of paths (see constraint 5, section 4, page 4). Table 8 illustrates the allocation of flow made by the lexicographic maximum flow algorithm

Algorithm 4 Lexicographic maximum flow algorithm

procedure LEXICOGRAPHICAL MAXIMIZATION OF FLOW
Let $\{\mathscr{B}_1, \ldots, \mathscr{B}_n\}$ the evacuation priorities list of buildings
$r_{i,j} = v_{i,j}, \forall 1 \le i \le n, \forall \mathscr{S}_j \in \mathscr{H}_i$
$\theta = 0, \bar{\phi}_l = \hat{\phi}_l^0, \forall l \in \mathscr{A}$

for $i = 1$ **to** n **do**
$\theta = \theta + 1$, SUM $= 0$;
$i^+ = i$
while $i \le n$ **do**
for $\mathscr{S}_j \in \mathscr{H}_i$ **do**
if $r_{i,j} > 0$ **then**
$\mathscr{Q}.add(\mathscr{S}_j)$ ▷ Queue
$\hat{\phi}^k_{\mathscr{M}_{j,i},\mathscr{B}_i} = \mathrm{Min}\{r_{i,j}, \mathrm{Min}_{\hat{\phi}} Arc^k_{i,j}\}, \forall k \in \{1, \ldots, \eta_{i,j}\}$
$\hat{\phi}_{s,\mathscr{M}_{j,i}} = \mathrm{Min}\{r_{i,j}, \hat{\mathscr{K}}_{i,j}\}$ ▷ Maximal number of vehicles to evacuate
$\hat{\phi}_{\mathscr{S}_j,p} = \sum_{i'=i^+}^{i-1} \hat{\phi}_{s,\mathscr{M}_{j,i'}} + \sum_{\mathscr{S}_{j'} \in \mathscr{Q}} \hat{\phi}_{s,\mathscr{M}_{j',i}}, \forall j \in \{1, \ldots, m\}$ ▷ Capacity
SUM $=$ SUM $+ \hat{\phi}_{s,\mathscr{M}_{j,i}}$
$Boost.push\_relabel\_max\_flow()$
FLOW $= \sum_{i'=i^+}^{i-1} \sum_{\mathscr{S}_j \in \mathscr{H}_{i'}} \phi_{s,\mathscr{M}_{j,i'}} + \sum_{\mathscr{S}_j \in \mathscr{Q}} \phi_{s,\mathscr{M}_{j,i}}$ ▷ Assigned flow
if FLOW $<$ SUM **then** ▷ Evacuation priority may not be respected
SUM $=$ SUM $- \hat{\phi}_{s,\mathscr{M}_{j,i}}$
$\hat{\phi}_{s,\mathscr{M}_{j,i}} =$ FLOW $-$ SUM ▷ Lexicographic maximal number
SUM $=$ SUM $+ \hat{\phi}_{s,\mathscr{M}_{j,i}}$
if $\sum_{\mathscr{S}_j \in \mathscr{H}_i \setminus \mathscr{Q}} r_{i,j} + \sum_{i'=i+1}^{n} \sum_{\mathscr{S}_j \in \mathscr{H}_{i'}} r_{i',j} = 0$ **then**
$Boost.push\_relabel\_max\_flow()$
end if
end if
$r_{i,j} = r_{i,j} - \hat{\phi}_{s,\mathscr{M}_{j,i}}$ ▷ Residual number of vehicles
if $r_{i,j} > 0$ **then**
if Not $back$ **then**
$i^\# = i, back = true$
end if
end if
end if
end for
$\mathscr{Q}.clear()$ ▷ Clear queue
end while
$x_{i,j,k,\theta} = \phi^k_{\mathscr{M}_{j,i},\mathscr{B}_i}, \forall 1 \le i \le n, \forall \mathscr{S}_j \in \mathscr{H}_i, \forall k \in \{1, \ldots, \eta_{i,j}\}$
if $back$ **then**
$\hat{\phi}_{s,\mathscr{B}_{i,j}} = 0, \forall i \in \{i^+, \ldots, n\}, \forall \mathscr{S}_j \in \mathscr{H}_i$
$back = false$
$i = i^\# - 1$ ▷ Index of first building not yet completely evacuated
end if
end for
end procedure

l	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}
$\hat{\phi}_l$	500	350	350	700	350	500	500	500	350	350	700	700	350	500	600	600	600	600	500	350
$\mathscr{B}_2 \to \mathscr{S}_1$								150	0	0	350	350								
$\mathscr{B}_2 \to \mathscr{S}_1$		200	200	550				0												
$\mathscr{B}_1 \to \mathscr{S}_1$	300	0	0	350																
$\mathscr{B}_1 \to \mathscr{S}_1$				0	0	150	150													
$\mathscr{B}_4 \to \mathscr{S}_1$											250	250					500	500	400	250
$\mathscr{B}_4 \to \mathscr{S}_2$															100	100	0	0		
$\mathscr{B}_3 \to \mathscr{S}_2$													250	400	0	0				

TABLE 8: Assignment of flow by lexicographic maximum flow algorithm at first time slot

	$\hat{\phi}_{s,\mathscr{M}_{1,1}}$	$\hat{\phi}_{s,\mathscr{M}_{1,2}}$	$\hat{\phi}_{s,\mathscr{M}_{2,3}}$	$\hat{\phi}_{s,\mathscr{M}_{1,4}}$	$\hat{\phi}_{s,\mathscr{M}_{2,4}}$
$\{\mathscr{M}_{1,2}\}$	-	500 ✓	-	-	-
$\{\mathscr{M}_{1,2}, \mathscr{M}_{1,1}\}$	700 ✗	500	-	-	-
$\{\mathscr{M}_{1,2}, \mathscr{M}_{1,1}, \mathscr{M}_{1,4}\}$	550	500	-	100 ✓	-
$\{\mathscr{M}_{1,2}, \mathscr{M}_{1,1}, \mathscr{M}_{1,4}, \mathscr{M}_{2,4}\}$	550	500	-	100	600 ✗
$\{\mathscr{M}_{1,2}, \mathscr{M}_{1,1}, \mathscr{M}_{1,4}, \mathscr{M}_{2,4}, \mathscr{M}_{2,3}\}$	550	500	300 ✓	100	500

TABLE 9: Steps of lexicographic maximum flow algorithm at first time slot

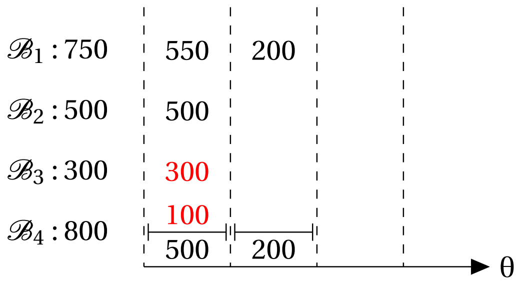


FIGURE 10: Evacuation scheduling according to lexicographic maximum flow algorithm

using push-relabel. This table indicates the residual capacity of each arc at the first slot and its use by push-relabel (construction of paths in residual graph) without considering the K-best paths between buildings and safe points. In addition, table 9 shows the behavior of the lexicographic algorithm incorporating *push-relabel* at the same time slot (see also figure 10).

During the evacuation of $\mathscr{B}_4$ towards $\mathscr{S}_1$, push-relabel has built a new path $\{a_{17}, a_{18}, a_{19}, a_{20}, a_{11}, a_{12}\}$ which allowed to evacuate 100 vehicles. It should be noted that if the variable capacities of shelters are not considered then the push-relabel algorithm will build the path $\{a_{13}, a_{14}, a_{19}, a_{20}, a_{11}, a_{12}\}$ which allows to evacuate 200 vehicles from $\mathscr{B}_3$ to $\mathscr{S}_1$ in addition to 100 vehicles that are evacuated from this building by its minimal path $\{a_{13}, a_{14}, a_{15}, a_{16}\}$ towards the safety point associated $\mathscr{S}_2$ (see figure 4).

The assignment of flow in the network, as we have shown, is subjected to the destinations capacities and to some extent to the evacuation paths (strategy of construction of sub-graph). These two elements (destinations and paths) forming the basis of construction of an evacuation plan are usually predetermined according to several important criteria (vulnerability, capacity, transit time, etc.), and therefore they must be respected. Otherwise the shelters capacities and the evacuation paths are (re-)determined depending on the assignment of flow.

6. Applications and results

The evacuation of buildings is based on a priorities list established according to several criteria. This list, in addition to its importance in assessing the vulnerability of the study site, allows to plan better the management of evacuation during a flood crisis.

The respect of priorities list is studied here on a real graph. While the greedy heuristic, first flow assignment algorithm presented in STOM, allows to observe the evacuation priorities, it does not ensure the maximization of flow injected into the network. The second approach developed to address this latter is based on push-relabel algorithm. We have shown on a small theoretical graph the non-compliance with priorities, destinations and evacuation routes, by this approach. We approach in this section the importance of compliance with the above elements in the context of ensuring both the succession of evacuation time slots of each building and the minimization of the number of these time slots.

The study site on which the model is applied and tested is the valley of Tours[3](see figure 11 and [3]). The elaborated evacuation network corresponds to the 3-best paths between buildings and shelters (see [2]) and is characterized by : 13446 nodes, 22676 arcs, 9825 buildings grouped in 2753 buildings[4] and 2 safety points. The figure 11 shows this area and the buildings to be evacuated to two shelters (ZRO[5]), one in the North and other in the South. For reasons of clarity of the figures, we perform the tests here only on the center of this valley : city of Tours. The graph corresponding to the this city is characterized by : 6754 nodes, 11270 arcs, 3975 buildings grouped in 1308 buildings and 2 shelters. The slot time unit is equal to 15 minutes and the evacuation priorities list is established according to several criteria (hazard level, nearby dikes, etc.)[3].

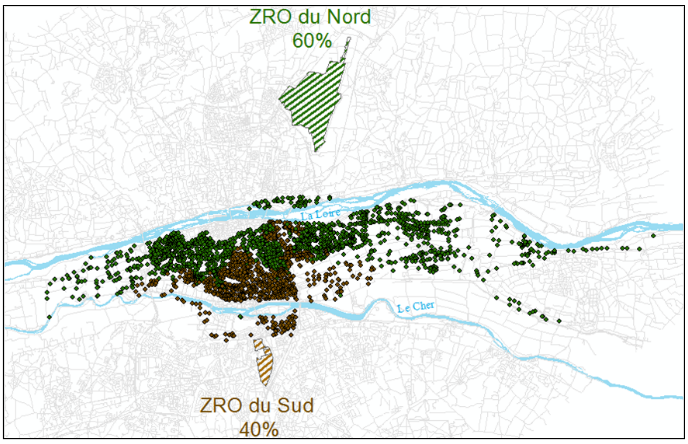


FIGURE 11: The valley of Tours, FR 37

The figure 12 shows the parallel evacuation of the city of Tours (France, 37) to two shelters (North and South). This evacuation is achieved by the greedy heuristic according to a priority list based on the distance from safe points. The size of each point shown in the figures that follow, unless otherwise indicated, refers to the number of evacuated vehicles from each given building at each given time slot.

In this figure (figure 12), buildings are classified by descending order of evacuation priority and each color corresponds to a building. Unlike the previous figure 12, the figure 13 shows a chaotic evacuation carried out by maximum flow (push-relabel) approach. The representation of buildings follows the same classification as that of figure 12. We see clearly in figure 13 that buildings which have near priorities may be evacuated at very different time slots.

3. France 37
4. By assigning each building to the nearest network node
5. Zone de Regroupement et d'Orientation

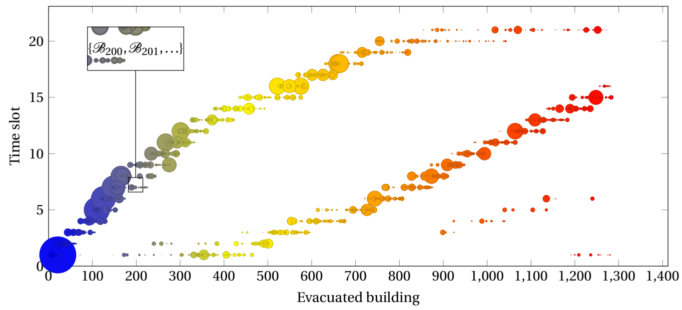


FIGURE 12: Evacuation carried out by the greedy algorithm

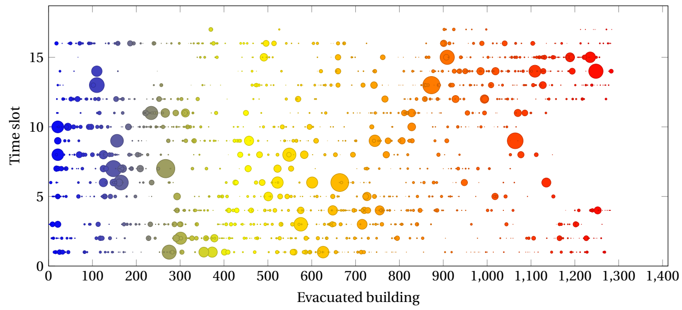


FIGURE 13: Evacuation carried out by the maximum flow approach

It should be noted that the disparity in the number of time slots (see also figure 22) rests solely with the non-compliance of paths and destinations (compliance or non-compliance with priorities have no influence on the maximization of flow).

Moreover, figures 14, 16, 17, 18 illustrate the importance of the respect of priorities, destinations and routes to ensure the succession of time slots of each building and the minimization of number of these time slots.

In figures 14, 16, we respectively represent the continuity and discontinuity of incoming flows from buildings with blue and red lines. For example, if a building is evacuated at time slots 1 and 3, then we draw a blue line between the time slots 1 and 2, another red between the 2^{nd} and 3^{rd}, and finally a blue line between the third and fourth slot. Time slot unit is equal 15 minutes as previously mentioned.

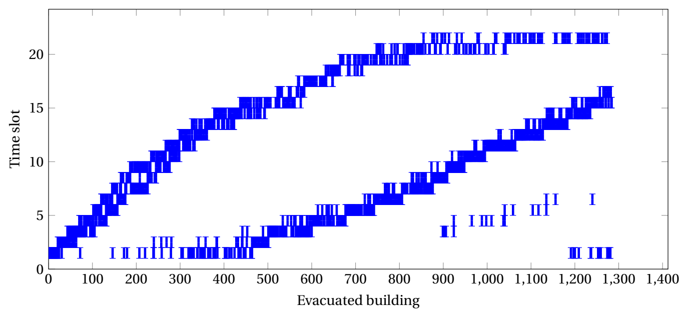


FIGURE 14: Continuity of flows from each building : greedy heuristic

Figure 14 shows the sequence of time slots of each building in the evacuation plan provided by greedy heuristic. This sequence, which may not be well illustrated in Figure 12 due to the density of information (some buildings appear to be evacuated in several non-consecutive time slots), is shown in figure 15. In this latter, buildings evacuated to the safe point in the north are presented first, then those that are evacuated to the south.

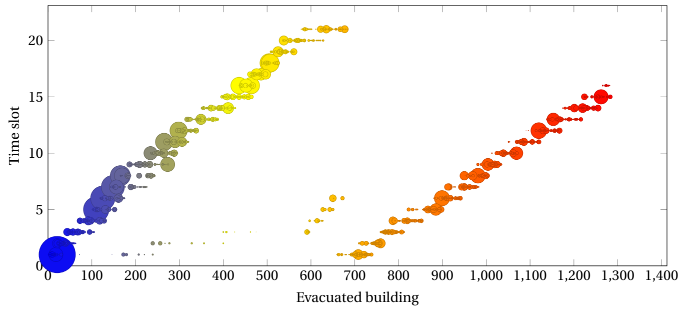


FIGURE 15: Parallel evacuation of buildings to shelters : greedy algorithm

In contrast, the sequence of evacuation time slots of each building is not guaranteed by the maximum flow (push-relabel) approach. Figure 16 illustrates this discontinuity showing non-consecutive evacuation of several buildings.

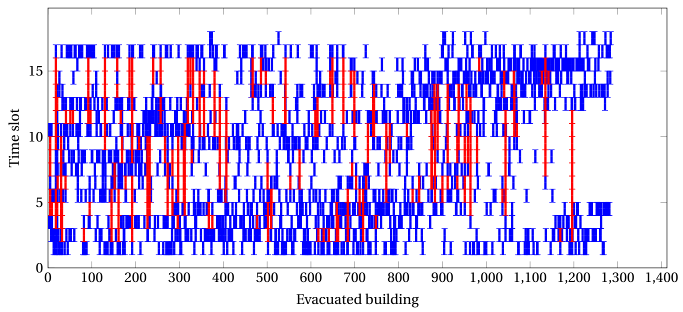


FIGURE 16: Discontinuity of flow from the majority of buildings : maximum flow approach

On the other hand, the disparity in the number of evacuation slots of each building is seen in the results of the greedy heuristic and the maximum flow (push-relabel) approach. the two figures 17 et 18 illustrate this disparity.

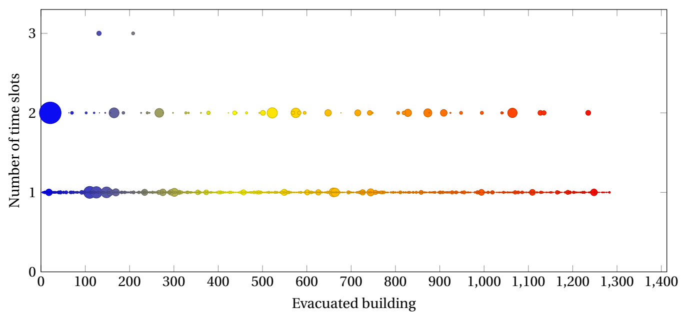


FIGURE 17: Number of time slots during which each building is evacuated : greedy algorithm

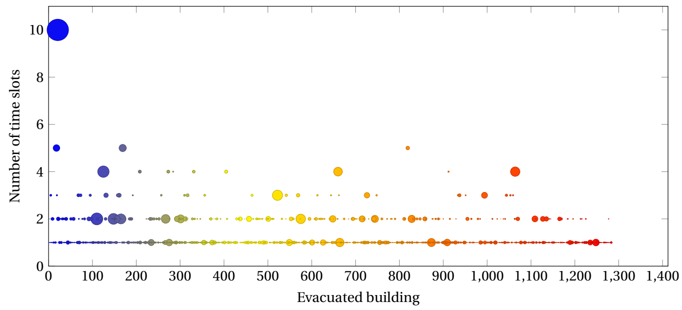


FIGURE 18: Number of time slots during which each building is evacuated : maximum flow approach

To ensure compliance with priorities and destinations, a lexicographic maximum flow algorithm coupled with push-relabel algorithm was developed (see section 5.0.4). However escape routes are not always respected as shown theoretically. This problem leads to the discontinuity of time slots of some buildings (see figure 20). Therefore respect of priorities, destinations and paths are necessary complementary elements to ensure the succession of time slots of all buildings (see [4]). Conversely, the minimum number of time slots of each building can be provided by the lexicographic maximum flow algorithm as shown in figure 21.

It should be recalled that compliance with paths automatically leads to the compliance with destination but the reverse is not necessarily true.

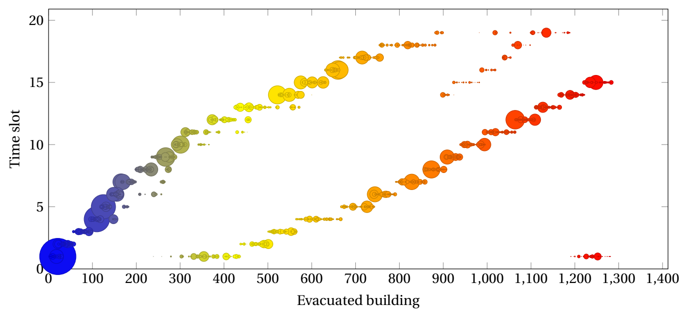


FIGURE 19: Evacuation carried out by the lexicographic maximum flow algorithm

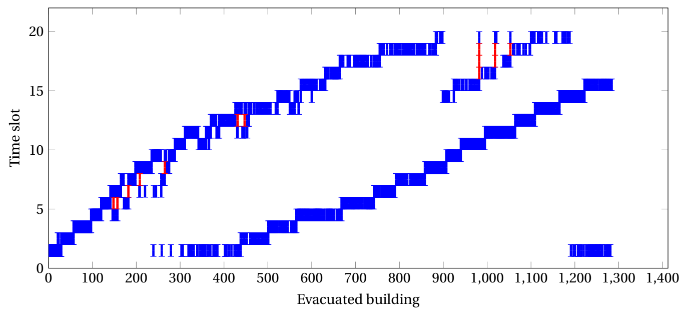


FIGURE 20: Semi-continuity : lexicographic maximum flow algorithm

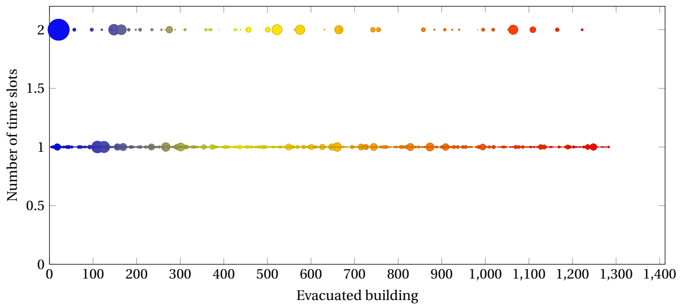


FIGURE 21: Evacuation time slots number of each building : lexicographic maximum flow algorithm

The last figure 22 compares the number of vehicles evacuated by time slots according to the approaches developed : maximum flow approach, lexicographic maximum flow algorithm and greedy heuristic.

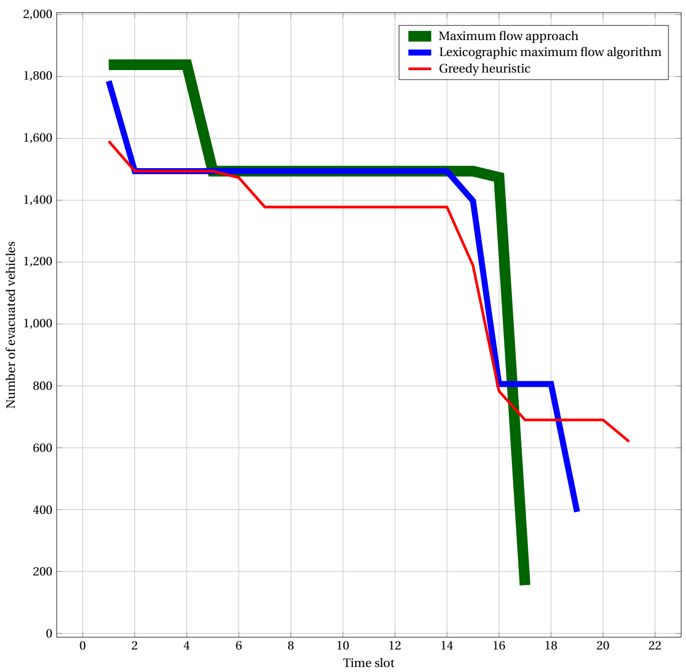


FIGURE 22: Comparison of the number of vehicles evacuated per slots between the different approachs developped

Conclusion

In this paper we have presented a novel polynomial algorithm for the lexicographic maximum dynamic flow problem with several sources applied to evacuation planning. Several approaches were developed and compared here. We have shown in a real graph the importance of compliance with the evacuation priority, the destinations and the evacuation paths in order to minimize the evacuation time slots of each building and to ensure the succession of these slots.

Acknowledgement

We would like to thank LA REGION CENTRE (France) and FEDER (Europe) for funding this research.

References

[1] Alaeddine, H., Maïzia, M., Serhini, K., and Néron, E. (2014a). A mezoscopic vehicles pursuit model for managing traffic during a massive evacuation. *portail.scd.univ-tours.fr/iii/encore/?lang=eng.*

[2] Alaeddine, H., Serhini, K., Maïzia, M., and Néron, E. (2014b). Finding the k-best paths in evacuation network. *portail.scd.univ-tours.fr/iii/encore/?lang=eng.*

[3] Alaeddine, H., Serhini, K., Maïzia, M., and Néron, E. (2014c). A spatio-temporel optimization model for the evacuation of the population exposed to natural disasters. *portail.scd.univ-tours.fr/iii/encore/?lang=eng.*

[4] Alaeddine, H., Serhini, K., Néron, E., and Maïzia, M. (2014d). Traffic assignment algorithms for planning a mass vehicular evacuation. *portail.scd.univ-tours.fr/iii/encore/?lang=eng.*

[5] Bretschneider, S. (2012). Mathematical models for evacuation planning in urban areas. *Lecture Notes in Economics and Mathematical Systems 659.*

[6] Bretschneider, S. and Kimms, A. (2012). Pattern-based evacuation planning for urban areas. *European Journal of Operational Research*, 216 :57–69.

[7] Burkard, R., Dlaska, K., and Klinz, B. (1993). The quikest flow problem. *ZOR-Methods and Models of Operations Research*, 37 :31–58.

[8] Catell, K., Dinneen, M., and Fellows, M. (1994). A simple linear-time algorithm for finding paths-decompsitions of small width. *Department of Computer Science, University of Victoria, Victoria, British Columbia, V8W 3P6, Canada.*

[9] Dijkstra, E. (1959). A note on two problems in connection with graphs. *Numerische Mathematik 1*, page 83–89.

[10] Fleischer, L. (2001). Universally maximum flow with piecewise-constant capacities. *Networks*, 38(3) :115–125.

[11] Fleischer, L. and Skutella, M. (2003). Quickest flows over time.

[12] Fleischer, L. and Tardos, E. (1996). Efficient continuous-time dynamic network flow algorithms.

[13] Ford, L. and Fulkerson, D. (1955). Maximal flow through a network. *Canadian Journal of Mathematics*, 8 :399–404.

[14] Ford, L. and Fulkerson, D. (1958a). Constructing maximal dynamic flows from static flows. *Oper. Res*, 6 :419–433.

[15] Ford, L. and Fulkerson, D. (1958b). Constructing maximal dynamic flows from static flows. *Operations Research*, 6 :419–433.

[16] Ford, L. and Fulkerson, D. (1962). Flows in networks. *Princeton University Press, Princeton, NJ.*

[17] Gale, D. (1959). Transient flows in networks. *The Michigan Mathematical Journal*, 6 :59–63.

[18] Goldberg, A. and Tarjan, R. (1988). A new approach to the maximum flow problem. *Journal of the ACM*, 35(4) :921–940.

[19] Hamacher, H., Pedersen, C., and Ruzika, S. (2007). Multiple objective minimum cost flow problems : A review. *European Journal of Operational Research*, 176 :1404–1422.

[20] Hamacher, H. and Tjandra, S. (2001). Mathematical modelling of evacuation problems : A state of art. *Beritche des Fraunhofer Istitut Techno- und Wirtschaftsmathematik, Nr. 24*.

[21] Hamacher, H. and Tufekci, S. (1987). On the use of lexicographic min cost flows in evacuation modeling. *Department of Industrial and System Engineering, University of Florida, Gainesville, Florida 32611*.

[22] Hoppe, B. and Tardos, E. (1994). Polynomial time algorithms for some evacuation problems. *Proc. of 5th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 433–441.

[23] Kamiyama, N. and Katoh, N. (2014). The universally quickest transshipment problem in a certain class of dynamic networks with uniform path-lengths. *Discrete Applied Mathematics*, 178 :89–100.

[24] Lim, G., Zangeneh, S., Baharnemati, M., and Assavapokee, T. (2012). A capacitated network flow optimization approach for short notice evacuation planning. *European Journal of Operational Research 223*, pages 234–245.

[25] Minieka, E. (1973). Maximal, lexicographic, and dynamic network flows. *Operations Research*, 21 :517–527.

[26] Orda, A. and Rom, R. (1995). On continuous network flows. *Operations Research Letters*, 17 :27–36.

[27] Saadatsresht, M., Mansourian, A., and Talaei, M. (2009). Evacuation planning using multiobjective evolutionary optimization approach. *European Journal of Operational Research 198*, pages 305–314.

[28] Sleator, D. and Tarjan, R. (1983). A data structure for dynamic trees. *Journal of Computer and System. Sciences*, 26 :362–391.

[29] Stepanov, A. and Smith, J. (2008). Multi-objective evacuation routing in transportation networks. *European Journal of Operational Research*.

[30] Wilkinson, W. (1971). An algorithm for universal maximal dynamic flows in a network. *Operations Research*, 19 :1602–1612.