Robust Flows with Losses and Improvability in Evacuation Planning *

Marc Goerigk†1 and Ismaila Abderhamane Ndiaye2

1Fachbereich Mathematik, Technische Universität Kaiserslautern, Germany
2Université François-Rabelais de Tours, LI EA 6300, OC ERL CNRS 6305, Tours, France

Abstract

We consider a network flow problem, where the outgoing flow is reduced by a certain percentage in each node. Given a maximum amount of flow that can leave the source node, the aim is to find a solution that maximizes the amount of flow which arrives at the sink.

Starting from this basic model, we include two new, additional aspects: On the one hand, we are able to reduce the loss at some of the nodes; on the other hand, the exact loss values are not known, but may come from a discrete uncertainty set of exponential size.

Applications for problems of this type can be found in evacuation planning, where one would like to improve the safety of nodes such that the number of evacuees reaching safety is maximized.

We formulate the resulting robust flow problem with losses and improvability as a mixed-integer program for finitely many scenarios, and present an iterative scenario-generation procedure that avoids the inclusion of all scenarios from the beginning. In a computational study using both randomly generated instance and realistic data based on the city of Nice, France, we compare our solution algorithms.

Keywords: network flow; flow with losses; robust optimization; network design

1 Introduction

Network flow with losses (and also gains) is a well-known and fruitful object of study, see, e.g., [Old01, Rad98, Way99]. Applications can typically be found, e.g., in telecommunication networks, electrical networks, exchange markets, machine loading, lot-sizing or the boolean satisfiability problem.

Also the field of evacuation planning has seen rising interest in the application of operations research models to help the decision maker assessing a

*Partially supported within the project DSS_Evac_Logistic, by the Federal Ministry of Education and Research Germany as FKZ 13N12229, and by the French National Research Agency as ANR-11-SECU-002-01 (CSOSG 2011).

†Corresponding author. Email: goerigk@mathematik.uni-kl.de

critical situation, and making the right choices to potentially safe lives, see, e.g., [OMH09, LKN11]. For general overviews, we refer to [HT01, AG06]. Also, network flows are a standard modeling technique in the field, see, e.g., [CFS82, Yam96, CHT88].

In this paper, we extend the concept of flows with losses to also include improvability (i.e., the amount of loss can be reduced at a limited number of nodes) as well as robustness (i.e., the exact amount of loss is not known exactly). Both model extensions are motivated by applications in evacuation planning.

As an example, we consider the situation that several evacuees need to leave an endangered region on foot (e.g., after an earthquake or a flooding strikes an urban area). Depending on the path they choose, they face different estimated degrees of dangerousness, which lead to the potential death of evacuees. Such a situation can be captured as a network flow with losses, where the number of evacuees reaching safety is to be maximized (see [NNL$^+$14]).

From a short-term perspective, security forces can be used to reduce the risk in the network (e.g., by extinguishing fires with the help of airplanes, or by removing debris) during an evacuation. From a long-term perspective, the structural safety of an endangered area can be improved (e.g., by stabilizing buildings with a relatively high probability of collapse during an earthquake), which will also result in a reduced risk value during an emergency.

Furthermore, as risk values are only an estimation, they are considered as being uncertain. We present a robust optimization approach with network improvability to include both these points.

In the following, we present some further literature in which related aspects are analyzed.

The basic idea of the well-known "Contraflow"-setting is to make better use of the given infrastructure in the case of an evacuation. As an example, if a highway has two lanes entering the endangered area, and two lanes leaving the endangered area, it makes sense to reverse at least one of the two entering lanes to facilitate the outgoing flow. The contraflow problem has been considered by [XLW10, XT11] an many others. Similar to our setting, the lane reversal may be interpreted as the distribution of improvements in the network, where the number of such improvements is bounded. However, in our setting, improvements are not on arc capacities, but on vertex safety instead.

There are several papers considering network improvement problems (such as [SK98, KMN$^+$98, DNW04, OZ07, DLG11, LM13, CLZ06]). However, the problem of improving vertex safety has not been considered yet.

Regarding the field of robust optimization as a means to handle optimization problems affected by uncertainty, we refer to the surveys [KY97, ABV09, BS04, BS03, BTGN09] and [GS13].

Contributions and overview. In Section 2, we introduce the nominal (i.e., non-uncertain) max flow problem with losses and improvements, which we use to model pedestrian movements during an evacuation. This model is extended to include uncertainty in Section 3, where we also present an iterative solution algorithm. As already the nominal problem is NP-hard, we also consider heuristic solution approaches in Section 4, and compare these algorithms in a computational study in Section 5. Section 6 concludes the paper and points out further research directions.

2 Flow with Losses and Improvability

2.1 Flow with Losses

Let a directed graph $G = (V, A)$ be given. We start with considering the maximum flow problem, where in each vertex $i \in V$, the flow leaving node i is multiplied with a fixed factor $p_i \in [0, 1]$. Given a bound on the value of outflow of a (super-)source, the *flow with losses* (FL) problem is to find the largest possible amount of flow entering a (super-)sink.

The problem can be easily modeled as a linear program using the variables f_{ij} to denote the flow along arc $(i, j) \in A$. Let $s \in V$ denote the source node, and $t \in V$ the sink node. We write $V' := V \setminus \{s, t\}$. Furthermore, let $S \in \mathbb{R}^+$ denote the maximum amount of flow leaving the source node, and let the capacity of each arc $(i, j) \in A$ be denoted as $u_{ij} \in \mathbb{R}^+$.

The flow with losses problem can then be formulated as:

$$
\begin{aligned}
\text{(FL)} \quad & \max \sum_{(i,t)\in A} f_{it} && && (1)\\
& \text{s.t.} \sum_{(s,i)\in A} f_{si} \le S && && (2)\\
& p_j \sum_{(i,j)\in A} f_{ij} = \sum_{(j,k)\in A} f_{jk} && \forall j \in V' && (3)\\
& 0 \le f_{ij} \le u_{ij} && \forall (i,j) \in A && (4)
\end{aligned}
$$

The objective function (1) is to maximize the amount of flow entering the sink t. Constraint (2) ensures that at most S evacuees leave the source s, while Constraints (3) are a modification of the usual flow constraints, capturing the relative loss of flow in each node. Finally, Constraints (4) ensure that arc capacities u_{ij} are respected.

In evacuation planning, the model can be used to calculate the percentage of pedestrian evacuees that could reach safe places, starting from an endangered region. Nodes can correspond to street crossings or important points along a street, and arcs to streets or street sections connecting these points. Note that this model follows a *macroscopic* point of view, i.e., individual or selfish behaviour of evacuees is ignored. Thus, a resulting optimal solution may be considered as a upper bound on the actual percentage of evacuees reaching safe places, which is only attainable if every evacuee acted according to the system optimum. Typically, macroscopic optimization results are complemented with microscopic simulation results to give a more realistic description of an emergency situation (see [HHK$^+$11]).

2.2 Flow with Losses and Improvability

We now extend the simple model (FL) to include possible improvements on the vertices. We assume that along with the loss p_i at every node $i \in V'$, we are also given an amount of possible improvement $\hat{p}_i$, where $p_i + \hat{p}_i \le 1$. The improvement of any node i has costs c_i, and there is a budget B on the total possible improvement costs. We denote the problem as FLI (flow with losses and improvements).

To model FLI as a mathematical program, we start with the following non-linear model:

$$\max \sum_{(i,t)\in A} f_{it} \tag{5}$$

$$\text{s.t.} \sum_{(s,i)\in A} f_{si} \leq S \tag{6}$$

$$\sum_{j\in V'} c_j z_j \leq B \tag{7}$$

$$(p_j + \hat{p}_j z_j) \sum_{(i,j)\in A} f_{ij} = \sum_{(j,k)\in A} f_{jk} \qquad \forall j \in V' \tag{8}$$

$$0 \leq f_{ij} \leq u_{ij} \qquad \forall (i,j) \in A \tag{9}$$

$$z_j \in \{0,1\} \qquad \forall j \in V' \tag{10}$$

As before, the objective function (5) models the number of evacuees entering the safety node, while Constraint (6) bounds the number of evacuees leaving the start node. We use new binary variables z_i to model if node $i \in V'$ is improved or not. Thus, Constraint (7) is a knapsack-constraint on the spendable budget. Finally, Constraints (8) model the loss of flow along the nodes, where the loss factor is $p_i + \hat{p}_i$, if i is improved, and p_i otherwise.

We linearize the products $z_j f_{ij}$ to gain a mixed-integer linear program. To this end, we introduce additional variables $h_j := \sum_{(i,j)\in A} z_j f_{ij}$. The resulting model is then the following:

$$\text{(FLI)} \qquad \max \sum_{(i,t)\in A} f_{it} \tag{11}$$

$$\sum_{(s,i)\in A} f_{si} \leq S \tag{12}$$

$$\sum_{j\in V'} c_j z_j \leq B \tag{13}$$

$$h_j \leq M z_j \qquad \forall j \in V' \tag{14}$$

$$h_j \leq \sum_{(i,j)\in A} f_{ij} \qquad \forall j \in V' \tag{15}$$

$$p_j \sum_{(i,j)\in A} f_{ij} + \hat{p}_j h_j = \sum_{(j,k)\in A} f_{jk} \qquad \forall j \in V' \tag{16}$$

$$0 \leq f_{ij} \leq u_{ij} \qquad \forall (i,j) \in A \tag{17}$$

$$z_j \in \{0,1\} \qquad \forall j \in V' \tag{18}$$

$$h_j \geq 0 \qquad \forall j \in V' \tag{19}$$

The new Constraints (14) and (15) model that $h_j = \sum_{(i,j)\in A} f_{ij}$ if and only if $z_j = 1$. To this end, M must be a large enough constant (e.g., $M \geq \max_{j\in V'} \sum_{(i,j)\in A} u_{ij}$ suffices).

Note that due to Constraint (13), one can easily show weak NP-completeness of FLI by a reduction from the one-dimensional knapsack problem, where we use one node for each item. Each node has $p_i = 0$ and $\hat{p}_i$ equal to the profit of that item. Each node has also a size c_i and the overall size of the knapsack is equal to B. However, we can even show strong NP-completeness via the 3-partition problem. A proof is given in Appendix A

Theorem 1. *The decision problem of FLI is strongly NP-complete.*

3 Robust Flows

We now extend the FLI problem to include uncertainty in the loss values p_i. The motivation is to better model that these values can only be estimates, and will never reflect the actual risk on a vertex. Instead, we assume to know only an uncertainty set $\mathcal{U} \subseteq \mathbb{R}^{|V|}$ that contains all possible realizations of p.

We have to decide where to put our improvement resources before we know the realization of p. Then, the actual scenario becomes revealed and the evacuees take the best possible route with respect to our improvements, and the scenario. Following a worst-case approach, the question is: Where should we put the improvements, so that the evacuee flow is maximal in the worst case that may happen?

3.1 Finite Uncertainty Sets

We first consider the case of a finite set of scenarios $\mathcal{U}^f = \{p^1, \dots, p^N\}$, and write $\mathcal{N} := \{1, \dots, N\}$. We assume that $p_i^k + \hat{p}_i \leq 1$ for all $i \in V$, $k \in \mathcal{N}$. The resulting robust two-stage problem can be modeled in the following way:

$$\begin{aligned}
\text{(RFLI)} \quad \max \min_{k \in \mathcal{N}} & \sum_{(i,t)\in A} f_{it}^k && && (20)\\
& \sum_{(s,i)\in A} f_{si}^k \leq S && \forall k \in \mathcal{N} && (21)\\
& \sum_{j \in V'} c_j z_j \leq B && && (22)\\
& h_j^k \leq M z_j && \forall j \in V', k \in \mathcal{N} && (23)\\
& h_j^k \leq \sum_{(i,j)\in A} f_{ij}^k && \forall j \in V', k \in \mathcal{N} && (24)\\
& p_j^k \sum_{(i,j)\in A} f_{ij}^k + \hat{p}_j h_j^k = \sum_{(j,k)\in A} f_{jk} && \forall j \in V', k \in \mathcal{N} && (25)\\
& 0 \leq f_{ij}^k \leq u_{ij} && \forall (i,j) \in A, k \in \mathcal{N} && (26)\\
& z_j \in \{0,1\} && \forall j \in V' && (27)\\
& h_j^k \geq 0 && \forall j \in V', k \in \mathcal{N} && (28)
\end{aligned}$$

We shall also write RFLI($\mathcal{U}^f$) when the uncertainty set underlying this formulation needs to be specified.

For every scenario $k \in \mathcal{N}$, we introduce a new set of flows f^k, h^k, which depend on the risk of the respective scenario. The minimum in the objective function can be removed by writing

$$\max \alpha$$

$$\alpha \leq \sum_{(i,t)\in A} f_{it}^k \qquad \forall k \in \mathcal{N}$$

(21–28)

instead. Note that we may consider only the variables z (i.e., the improvements) as decision variables; then, to evaluate the objective $val(z)$, one needs to solve a nominal FLI problem for every scenario to determine the resulting worst-case.

3.2 Bounded Uncertainty Sets

3.2.1 Model

We now extend the previous model by considering more realistic uncertainty sets. For every loss value p_i, an interval of possible outcomes $[\underline{p}_i, \overline{p}_i]$ is given. However, as it would be too pessimistic to assume that the worst-case $\underline{p}_i$ happens at all nodes simultaneously, we follow the spirit of [BS04] and bound the number of vertices that may deviate from their best-case $\overline{p}_i$. We model this as the following uncertainty set:

$$\mathcal{U} = \mathcal{U}(K) = \left(\left[\underline{p}_1, \overline{p}_1\right] \times \ldots \times \left[\underline{p}_{|V|}, \overline{p}_{|V|}\right]\right) \cap \left\{p \in \mathbb{R}^{|V|} \;\middle|\; \#\{i \in V : p_i < \overline{p}_i\} \leq K\right\}$$

As before, we assume $\overline{p}_i + \hat{p}_i \leq 1$. Here, $K \in \mathbb{N}$ is a parameter that can be used to control the degree of conservatism of a solution. For $K = n$, the uncertainty set $\mathcal{U}$ becomes the Cartesian product of intervals, resulting in a large possible set of outcomes. A resulting robust solution will have a relatively small nominal objective value, but a high degree of robustness. For $K = 0$ on the other hand, the uncertainty set $\mathcal{U}$ becomes a best-case singleton, meaning that uncertainty is ignored and a nominal FLI problem needs to be solved. It is in the hands of the practitioner to find a value for K that suits the application, and the outcomes of multiple alternatives may be compared.

Note that when writing RFLI($\mathcal{U}$) directly as a mixed-integer program like (20–28), one would require infinitely many constraints and variables. However, whenever we have $p_i < \overline{p}_i$ for a loss parameter, we may as well assume that $p_i = \underline{p}_i$, as such a scenario would always dominate any scenario with less risk. This makes it possible to enumerate all relevant scenarios.

Lemma 2. *RFLI($\mathcal{U}$) can be solved by solving a problem RFLI($\mathcal{U}^f$), where* $|\mathcal{U}^f| = \binom{|V|}{K}$

That is, we can reduce the bounded uncertainty set to a finite uncertainty set, which contains all possible scenarios where K nodes are "bad" and the others are "good". As this number grows exponentially, a better option to solve RFLI is required.

3.2.2 Evaluating a solution

We first consider the following subproblem: Given a fixed choice of improvements z, which scenario $p \in \mathcal{U}$ is responsible for the worst output flow? We may model this question as an optimization problem over the losses p.

$$WC(z) := \min f^*(p)$$

$$\begin{aligned}
& p_i = \overline{p}_i + (\underline{p}_i - \overline{p}_i)w_i + \hat{p}_i z_i && \forall i \in V' \\
& \sum_{i \in V'} w_i \leq K \\
& w_i \in \{0,1\} && \forall i \in V' \\
& p_i \geq 0 && \forall i \in V'
\end{aligned}$$

We decide for each node i if it is “bad” with the help of the binary variable w_i. We would like to make a choice in a way that $f^*(p)$ is as small as possible, where $f^*(p)$ denotes the optimal outflow under scenario p. This is itself simply a flow with losses problem.

$$f^*(p) := \max \sum_{(i,t) \in A} f_{it} \tag{29}$$

$$\sum_{(i,s) \in A} f_{si} \leq S \tag{30}$$

$$p_j \sum_{(i,j \in A)} f_{ij} \geq \sum_{(j,k) \in A} f_{jk} \qquad \forall j \in V' \tag{31}$$

$$0 \leq f_{ij} \leq u_{ij} \qquad \forall (i,j) \in A \tag{32}$$

Note that using “$\geq$” in Constraint (31) instead of “=“ does not change the optimal solution of the flow with losses problem. To integrate the computation of $f^*(p)$ into $WC(z)$, we dualize the problem and get:

$$\begin{aligned}
\min \ & \sum_{(i,j) \in A} u_{ij}\alpha_{ij} + S\beta \\
& \alpha_{ij} + \gamma_i - p_j\gamma_j + \chi_s(i)\beta \geq \chi_t(j) && \forall (i,j) \in A \\
& \alpha_{ij} \geq 0 && \forall (i,j) \in A \\
& \beta \geq 0 \\
& \gamma_i \geq 0 && \forall i \in V' \\
& \gamma_s = \gamma_t = 0
\end{aligned}$$

where

$$\chi_i(j) = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{else} \end{cases}$$

We now use this dual formulation to rewrite $WC(z)$.

$$\begin{aligned}
\min \ & \sum_{(i,j) \in A} u_{ij}\alpha_{ij} + S\beta \\
& \sum_{i \in V'} w_i \leq K \\
& \alpha_{ij} + \gamma_i - (\overline{p}_j + (\underline{p}_j - \overline{p}_j)w_j + \hat{p}_j z_j)\gamma_j + \chi_s(i)\beta \geq \chi_t(j) && \forall (i,j) \in A \\
& \alpha_{ij} \geq 0 && \forall (i,j) \in A \\
& \beta \geq 0 \\
& \gamma_i \geq 0 && \forall i \in V' \\
& \gamma_s = \gamma_t = 0
\end{aligned}$$

$$w_i \in \{0,1\} \qquad \forall i \in V'$$

Due to the product $w_j\gamma_j$, this is a non-linear program (note that z is fixed, and the product $z_j\gamma_j$ is linear here). Substituting $\gamma'_i = w_i\gamma_i$ yields

$$\min \sum_{(i,j)\in E} u_{ij}\alpha_{ij} + S\beta \tag{33}$$

$$\sum_{i\in V} w_i \le K \tag{34}$$

$$\alpha_{ij} + \gamma_i - (\overline{p}_j + \hat{p}_j z_j)\gamma_j + (\overline{p}_j - \underline{p}_j)\gamma'_j + \chi_s(i)\beta \ge \chi_t(j) \quad \forall (i,j)\in A \tag{35}$$

$$\gamma'_i \le \gamma_i \quad \forall i \in V' \tag{36}$$

$$\gamma'_i \le M w_i \quad \forall i \in V' \tag{37}$$

$$\alpha_{ij} \ge 0 \quad \forall (i,j)\in A \tag{38}$$

$$\beta \ge 0 \tag{39}$$

$$\gamma_i, \gamma'_i \ge 0 \quad \forall i \in V' \tag{40}$$

$$\gamma_s = \gamma_t = 0 \tag{41}$$

$$w_i \in \{0,1\} \quad \forall i \in V' \tag{42}$$

where M is a constant large enough ($\prod_{(i,j)\in A} \underline{p}_j^{-1}$ suffices).

4 Solution Algorithms

4.1 Scenario Generation

Using formulation (33–42) for $WC(z)$, we can evaluate the objective value for a choice of improvements z and also produce a scenario w where this objective value is attained. In the following, this is used as part of a solution algorithm.

We start with any finite scenario set $\mathcal{U}^0 \subseteq \mathcal{U}$, e.g., $\mathcal{U}^0 = \emptyset$ or $\mathcal{U}^0 = \{\overline{p}\}$. Solving RFLI($\mathcal{U}^0$) yields some solution for the improvements z^0 and an objective value OBJ^0. Solving $WC(z^0)$ determines a new scenario given by w^0 and an objective value WC^0. Setting $\mathcal{U}^1 := \mathcal{U}^0 \cup \{w^0\}$ gives a new scenario set that can be used to determine the next choice of improvements, which is then iteratively repeated.

The algorithm stops with an improvement z^k, when the objective value of the worst-case problem $WC(z^k)$ equals the objective value of RFLI($\mathcal{U}^k$). As each problem RFLI($\mathcal{U}^k$) for a scenario set $\mathcal{U}^k \subseteq \mathcal{U}$ is a relaxation of the problem RFLI($\mathcal{U}$), the optimal objective value of each solution z^k is an upper bound on the optimal objective value of problem RFLI($\mathcal{U}$). Therefore, optimality is proved when the worst-case objective equals the estimated upper bound.

The algorithm is summarized as Algorithm 1.

Note that the values OBJ^k are monotonically decreasing, being the optimal objective values to problems that have increasingly more constraints and variables. However, their actual objective values WC^k are not necessarily monotonically increasing. Figure 1 provides an example for a typical run of Algorithm 1. Encircled are solutions which are better than solutions from previous iterations.

Algorithm 1 (Exact Algorithm for RFLI($\mathcal{U}$))

```
Require: An instance of RFLI($\mathcal{U}$).
 1: $k \leftarrow 0$
 2: $\mathcal{U}^0 \leftarrow \{\overline{p}\}$
 3: Solve RFLI($\mathcal{U}^k$). Let $z^k$ be the resulting solution for the improvements, and
    $OBJ^k$ the resulting objective value.
 4: Solve $WC(z^k)$. Let $p^k$ be the resulting scenario, and $WC^k$ the resulting
    objective value.
 5: if $OBJ^k = WC^k$ then
 6:     return Optimal improvements $z^k$ and objective value $OBJ^k$.
 7: else
 8:     $\mathcal{U}^{k+1} \leftarrow \mathcal{U}^k \cup \{p^k\}$
 9:     $k \leftarrow k + 1$
10:     Goto 3
11: end if
```

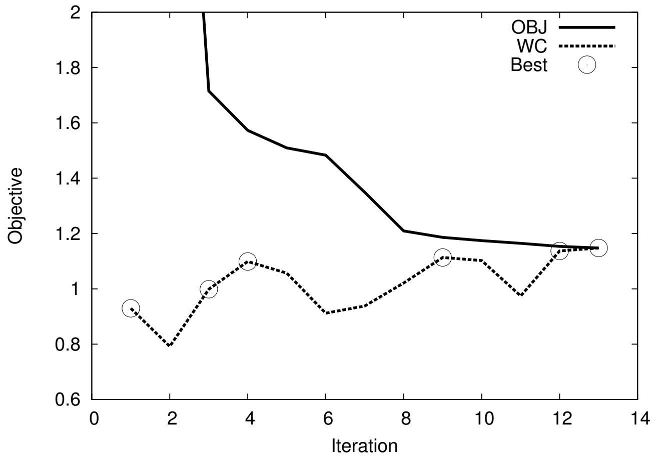


Figure 1: Example run of Algorithm 1. Optimality is reached when OBJ^k and WC^k coincide.

4.2 Bounding the Scenario Set

Note that problem (33–42) to compute the objective value of a choice of improvements has $O(|A|)$ many variables and $O(|A|)$ many constraints. Problem (20–28) however has in the kth iteration $O(k|A|)$ many variables and $O(k|A|)$ many constraints. Thus, while the computation time to solve $WC(z)$ stays roughly the same over all iterations, the solution time for RFLI($\mathcal{U}$) increases with increasing iteration number of Algorithm 1.

Therefore, if there was some method to remove redundant scenarios from $\mathcal{U}^k$, computation times will improve. To this end, we consider the following variant of Algorithm 1: Let $L \in \mathbb{N}$ be an algorithm parameter that controls the desired size of $\mathcal{U}^k$. When a scenario p^k is added to $\mathcal{U}^k$ in Step 8 of the algorithm, we check if $|\mathcal{U}^k| \leq L - 1$. If this is the case, the scenario is included as usual. If, however, $|\mathcal{U}^k| = L$ (i.e., the desired scenario set size is reached), we first choose and remove a scenario from $\mathcal{U}^k$.

If L is chose too small, this may result in a cycling of the solution algorithm. To prevent this, we save all scenarios that have been generated in the solution process so far in an additional set $\overline{\mathcal{U}}^k$. If a scenario is generated twice (i.e., $p^k \in \overline{\mathcal{U}}^{k-1}$), we increase the parameter L by one.

Concerning the choice which scenario should be removed from the current uncertainty set, different approaches would be possible. In the end, every such approach will lead to an optimal solution of RFLI($\mathcal{U}$), but will have an impact on computation times.

In the following, we use the strategy to remove a scenario $p^j \in \mathcal{U}^k$ for which $\sum_{(i,t)\in A} f^j_{it}$ is maximal in the most recent solution of the RFLI problem. In other words, the scenario is removed which allows the largest amount of flow to reach the sink. However, due to the max-min structure of the objective function, an optimal solution of RFLI does not necessarily maximize the flow in every scenario. To ensure this, a lexicographic objective function of the form

$$\max \alpha + \varepsilon \sum_{k\in\mathcal{N}} \sum_{(i,t)\in A} f^k_{it}$$

is used, with $1 \gg \varepsilon > 0$.

4.3 Artificial Scenarios

As the nominal problem FLI is already NP-hard and the robust version further increases its computational difficulty, we now introduce heuristic solution approaches.

Note that to evaluate a choice of improvements z, the mixed-integer program $WC(z)$ needs to be solved. This renders heuristic procedures ineffective, which rely on the evaluation of many solutions (e.g., local search or genetic algorithms).

Instead, we aim at the solution of one FLI problem with a single scenario $p \in \mathbb{R}^{|V|}$ that "represents" the whole uncertainty set $\mathcal{U}$ appropriately. It is not necessary that p is an actually possible scenario, i.e., $p \notin \mathcal{U}$ is possible. While this problem can be solved comparatively fast, its quality depends on how p is determined.

We consider the following cases:

1. We set $p = \lambda\underline{p} + (1-\lambda)\overline{p}$ for some constant $\lambda \in [0,1]$. This includes the best-case scenario $p = \overline{p}$, and the worst-case scenario $p = \underline{p}$.

2. As a special case of this convex combination, we use a value for λ that depends on K and n. If one expects "bad" nodes to be randomly uniformly distributed, then the expected loss is given as $p = \lambda\underline{p} + (1-\lambda)\overline{p}$ with $\lambda = K/n$.

3. Finally, assumption that "bad" nodes are uniformly distributed may be too optimistic, given that only nodes are relevant which actually carry some flow. Estimating that in a graph with n nodes, only $\sqrt{n}$ nodes have a flow which is larger than zero, we get the more pessimistic value $\lambda = \min\{1, K/\sqrt{n}\}$. If the graph class is known, such an estimate may be further refined.

Any of the above approaches yield a heuristic set of improvements z. To evaluate these solutions, a problem $WC(z)$ still needs to be solved. For large instances, this evaluation may take more computation time than for finding the improvements.

5 Experiments

We present two sets of experiments. In the first one, we use randomly generated grid graphs to evaluate the algorithms presented in this paper. In the second experiment, we use realistic data based on the city of Nice, France, to compare our heuristic approaches.

All experiments were conducted on a computer with a 16-core Intel Xeon E5-2670 processor, running at 2.60 GHz with 20MB cache, and Ubuntu 12.04. Mixed-integer programs were solved using Gurobi v. 5.5 using C++ programs compiled with gcc v. 4.5.4. and flag -O3. For the first experiment, 4 processors were used per algorithm, and the complete experiment was conducted four times to remove solver variance in solution times. For the second experiment, all 16 processors were used. Reported computation times always refer to wall clock measurements.

5.1 Random Instances

Instances. We created 7 sets $\mathcal{I}_\ell$, $\ell = 3, \dots, 9$ of random instances to compare our solution algorithms for the RFLI problem. Each graph is a grid of size $\ell \times \ell$, where evacuees start in the lower left corner and need to reach the upper right corner. S is always equal to 100, B and K are chosen uniformly randomly from $\{1, \dots, \ell\}$. Arc capacities are generated in $\{1, \dots, 100\}$, and we generate losses $\underline{p}_i \in [0.001, 0.999]$, $\overline{p}_i \in [\underline{p}_i, 1]$, and $\hat{p}_i \in [0, 1 - \overline{p}_i]$ (in this order). Improvement costs c_i are chosen randomly from $[0.5, 1.5]$. For each $\ell = 3, \dots, 9$, we generated 20 such instances (i.e., a total of 140 instances).

Setting. Every instance is solved using the iterative Algorithm 1, and its variant with bounded desired uncertainty size from Section 4.2 with $L = 5$ and $L = 10$. When referring to these solutions, we write OPT-F ("F" like "full"), OPT-5, and OPT-10, respectively. Furthermore, we determine heuristic solutions using artificial scenarios as described in Section 4.3 with $\lambda \in \{0.00, 0.25, 0.50, 0.75, 1.00, K/n, K/\sqrt{n}\}$. We refer to these solutions as W(λ).

We record the objective value and the computation time for each of these algorithms.

Results. We summarize our results in Tables 1, 2, 3 and 4.

The average relative objective value of the heuristic algorithms is compared in Table 1. We normalized values such that 1.00 corresponds to an optimal solution, e.g., the value 0.96 in the top left corner of the table means that on average over the 20 instances of size 3×3, the best-case solution reaches an objective value that is 96% of the optimal objective value. Generally, the quality of the heuristic solutions declines with increasing instance size. Furthermore, for the considered instances, the "more robust" solutions (i.e., $\lambda \in \{0.75, 1.00, K/\sqrt{n}\}$) tend to perform better. These three algorithms perform similarly, with no clear indicator which should be preferred in general.

Instance	W(0.00)	W(0.25)	W(0.50)	W(0.75)	W(1.00)	W(K/n)	W($K/\sqrt{n}$)
$\mathcal{I}_3$	0.96	0.98	0.99	0.97	0.99	0.98	0.99
$\mathcal{I}_4$	0.89	0.91	0.92	0.93	0.95	0.90	0.93
$\mathcal{I}_5$	0.89	0.90	0.92	0.94	0.93	0.88	0.89
$\mathcal{I}_6$	0.76	0.76	0.83	0.89	0.88	0.75	0.85
$\mathcal{I}_7$	0.80	0.81	0.83	0.83	0.84	0.80	0.85
$\mathcal{I}_8$	0.78	0.78	0.79	0.82	0.81	0.78	0.82
$\mathcal{I}_9$	0.73	0.73	0.83	0.82	0.83	0.74	0.84

Table 1: Average relative objective value.

Instance	W(0.00)	W(0.25)	W(0.50)	W(0.75)	W(1.00)	W(K/n)	W($K/\sqrt{n}$)
$\mathcal{I}_3$	0.02	0.02	0.02	0.02	0.02	0.02	0.02
$\mathcal{I}_4$	0.03	0.03	0.03	0.03	0.03	0.03	0.03
$\mathcal{I}_5$	0.05	0.05	0.05	0.05	0.05	0.05	0.05
$\mathcal{I}_6$	0.10	0.09	0.09	0.10	0.11	0.10	0.11
$\mathcal{I}_7$	0.18	0.16	0.17	0.16	0.27	0.17	0.23
$\mathcal{I}_8$	0.69	0.77	0.77	0.70	0.66	0.73	0.66
$\mathcal{I}_9$	1.96	1.89	1.22	1.67	1.90	2.00	1.67

Table 2: Average computation times in seconds for heuristics (total).

Also considering the average computation times of these heuristics as presented in Table 2, results are very similar. All algorithms require the solution of one RFLI problem, and one worst-case problem for the evaluation.

While these computation times scale well with the problem size, this is not the case for the iterative solution algorithms, see Table 3. We report computation times for the lower bound problems $WC(z)$ and the upper bound problems RFLI separately. As these results indicate, most computation time is spent solving the increasingly difficult RFLI problems.

Comparing the three exact solution methods, we find that the bounding the desired uncertainty size does generally not help to decrease computation times. As expected using this variant, the number of iterations increases (see Table 4), while the computation time per iteration can be reduced in most cases.

A direct comparison of computation times between OPT-F and OPT-5 for instances where OPT-F needs at least 6 upper bound iterations is shown in Figure 2. Please note the double-log scale. While the reduced desired uncertainty size is able to improve some computation times, it is slightly worse on most of

	LB			UB			Total		
Instance	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10
$\mathcal{I}_3$	0.02	0.01	0.01	0.02	0.02	0.02	0.04	0.03	0.03
$\mathcal{I}_4$	0.03	0.03	0.03	0.05	0.04	0.04	0.08	0.07	0.07
$\mathcal{I}_5$	0.08	0.08	0.08	0.18	0.17	0.17	0.26	0.25	0.25
$\mathcal{I}_6$	0.46	0.56	0.45	2.69	2.64	2.44	3.15	3.20	2.89
$\mathcal{I}_7$	2.03	2.78	2.04	14.41	18.29	13.93	16.44	21.07	15.97
$\mathcal{I}_8$	9.49	14.59	10.73	153.89	192.99	180.56	163.38	207.58	191.29
$\mathcal{I}_9$	47.56	98.39	66.94	370.96	884.88	521.70	418.53	983.27	588.64

Table 3: Average computation times in seconds for exact algorithms.

	#UB			Time per UB		
Instance	OPT-F	OPT-5	OPT-10	OPT-F	OPT-5	OPT-10
$\mathcal{I}_3$	2.05	2.05	2.05	0.01	0.01	0.01
$\mathcal{I}_4$	2.30	2.30	2.30	0.02	0.02	0.02
$\mathcal{I}_5$	3.15	3.15	3.15	0.05	0.05	0.05
$\mathcal{I}_6$	5.25	6.20	5.30	0.30	0.22	0.27
$\mathcal{I}_7$	6.00	7.48	6.10	1.29	1.09	1.21
$\mathcal{I}_8$	8.85	12.90	10.00	12.90	10.24	12.70
$\mathcal{I}_9$	7.70	12.15	8.90	24.63	26.58	24.29

Table 4: Average number of upper bound subproblems and computation times per problem in seconds.

the instances. Here, future research may focus on better strategies to determine the scenario that should be removed in each iteration.

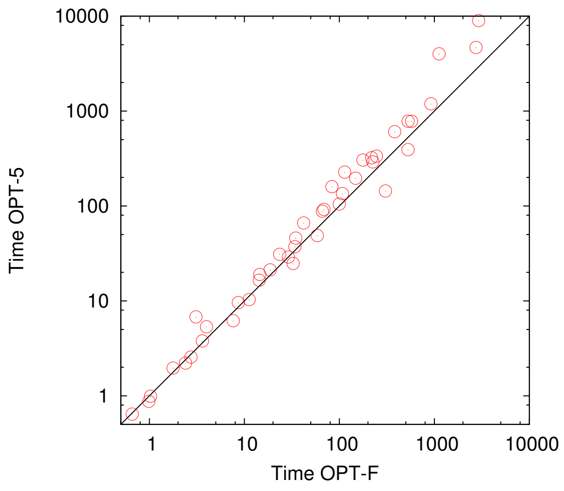


Figure 2: Comparison of computation times in seconds for OPT-F and OPT-5.

Summarizing these results, we find that exact computation times for the considered grid instances do not scale well with the instance size, as both the required number of iterations and the solution time per iteration increase. The heuristic algorithms scale well with the instance size and yield results within around 80% of an optimal solution for the more conservative scenario estimates.

5.2 Realistic Instances

Instances. We now consider problem instances which are based on real-world data modeling the city of Nice, France. The city is situated in a seismically active region, and has encountered several earthquakes and tsunamis in its history. Figure 3 shows the underlying graph, which is generated from OpenStreetMap data. We used 500 nodes and 1124 edges.

To generate loss values, we used seismic simulation data from [L$^+$14] that simulates an earthquake with similar characteristics as the Ligure earthquake of 1887 with a magnitude of approximately $6.9M_W$. Based on these simulation results, loss values were estimated corresponding to the expected damage to infrastructure. Figure 4 shows the level of damages considering the Ligure earthquake scenario.

The population data come from the French National Institute for Statistics and Economic Studies *(INSEE)* who estimate the population of each city including Nice and the repartition of the population over the area. Their estimation can also take into account foreigners during holidays.

In order to know the population impacted by the earthquake scenario, the population repartition given for the city of Nice by *INSEE* in 2011 is mapped with the Ligure damages scenario, e.g., Figure 4 in order to know the number of persons that have to be evacuated. We consider a set of 17 starting points form where evacuees can have informations as their affectation shelters and the safest way to reach them.

The shelter locations are outside the endangered areas (i.e. area in red) In this case of study, we consider 7 of them near the living places that have to be evacuated and each of them has a capacity.

No data regarding the costs to improve the safety of locations was available. We therefore generated random values from $[0.5, 1.5]$ per node corresponding to "unit improvement costs", e.g., 10,000 Euros.

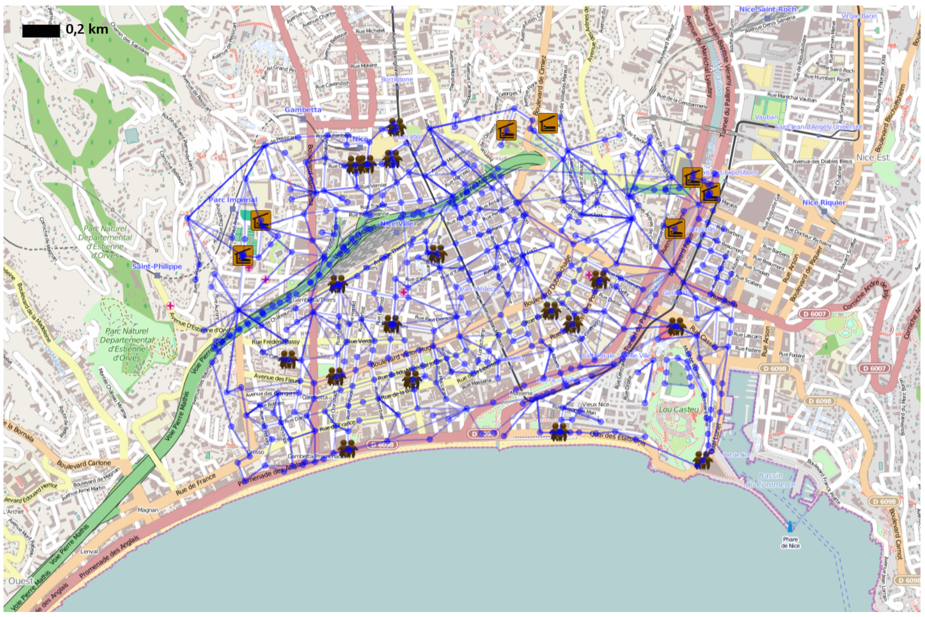


Figure 3: Graph for Nice.

Setting. We consider the two scenario sets $\mathcal{U}(10)$ (i.e., only 2% of all nodes may have their worst-case loss value) and the more conservative set $\mathcal{U}(50)$ (i.e., 10% of nodes). We solved these instances heuristically using $\lambda \in \{0.00, 0.50, 1.00, K/n, K/\sqrt{n}\}$ for varying improvement budget $B \in [10, 30]$. To limit computation times, we imposed a timelimit of 120 seconds for each lower bound problem.

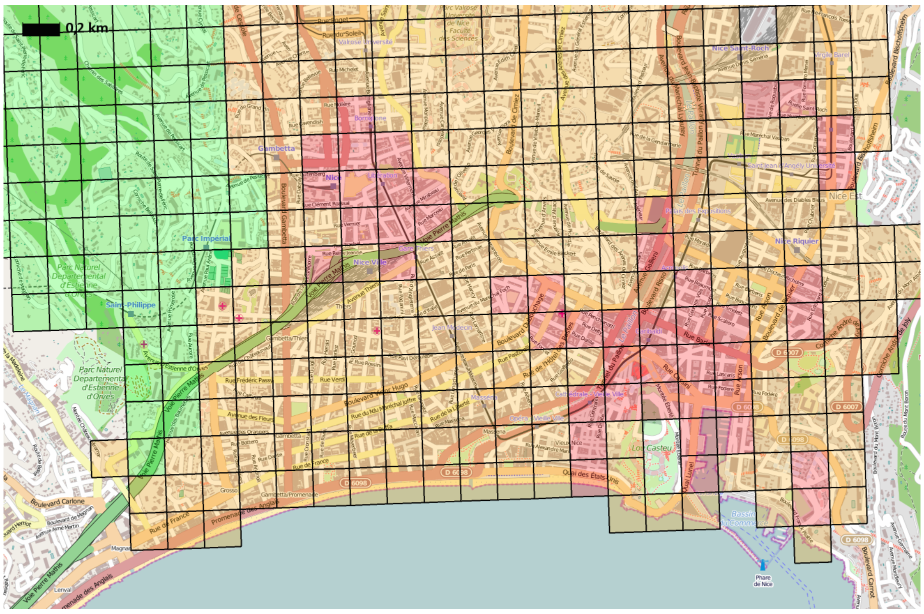
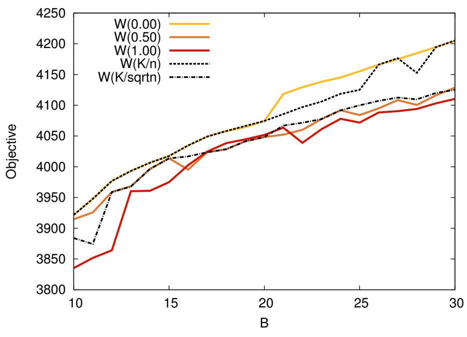
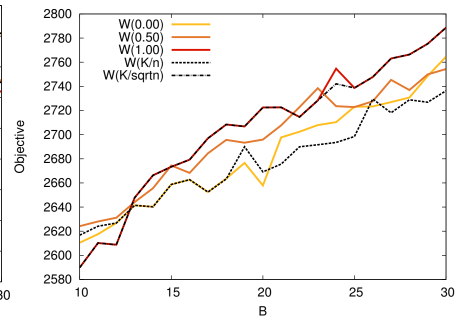


Figure 4: Ligure damages scenario for Nice

Results. We compare the heuristic objective values for $K = 10$ and $K = 50$ in Figures 5(a) and 5(b), respectively. While the more optimistic estimates $\lambda = 0.00$ and $\lambda = K/n$ outperform the others for $K = 10$, this is turned around for $K = 50$. Overall, the value $\lambda = K/\sqrt{n}$ provides a good compromise choice for both settings. Computation times for the upper bound problems were in the order of a few seconds, while the lower bound problems always required the full 120 seconds available.



(a) $K = 10$



(b) $K = 50$

Figure 5: Heuristic objective values for varying improvement budget B.

Using computational results as presented in Figure 5, the planner is provided a helpful tool to decide two questions: Firstly, how many evacuees can be estimated to reach shelter locations under different degrees of damage severeness? Using our population data, estimates lie between ~ 2500 and ~ 4200

evacuees. Secondly, what is the safety gain by increasing the spending on security measures? These results indicate between 8 evacuees per improvement unit for the heavy-damage case, and around 15 evacuees per improvement for the light-damage case. These results may serve as indicators to better prepare for an emergency situation.

6 Conclusion

In this paper we contributed to the current literature on network flow evacuation planning models, by introducing a new evacuation model that includes losses along nodes, improvability, and uncertainty. As the scenario set and thus the model is of exponential size, it cannot be solved directly using a mixed-integer programming solver. We therefore developed an algorithm that iteratively increases the problem size by finding the worst-case scenario for the current (partial) solution.

In computational experiments we compared the performance of variants for the iterative algorithm to heuristic approaches using only a single scenario. Both randomly generated and realistic instances based on the city of Nice, France, were used.

References

[ABV09] H. Aissi, C. Bazgan, and D. Vanderpooten. Minmax and minmax regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427 – 438, 2009.

[AG06] N. Altay and W. G. Green III. OR/MS research in disaster operations management. *European Journal of Operational Research*, 175(1):475 – 493, 2006.

[BS03] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming Series B*, 98:2003, 2003.

[BS04] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[BTGN09] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton University Press, Princeton and Oxford, 2009.

[CFS82] L.G. Chalmet, R.L. Francis, and P.B. Saunders. Network models for building evacuation. *Fire Technology*, 18(1):90–113, 1982.

[CHT88] W. Choi, H.W. Hamacher, and S. Tufekci. Modeling of building evacuation problems by network flows with side constraints. *European Journal of Operational Research*, 35(1):98 – 110, 1988.

[CLZ06] A. M. Campbell, T. J. Lowe, and L. Zhang. Upgrading arcs to minimize the maximum travel time in a network. *Networks*, 47(2):72–80, 2006.

[DLG11] B. Dilkina, K. J. Lai, and C. P. Gomes. Upgrading shortest paths in networks. In T. Achterberg and J. C. Beck, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 6697 of *Lecture Notes in Computer Science*, pages 76–91. Springer Berlin Heidelberg, 2011.

[DNW04] I. Demgensky, H. Noltemeier, and H.-C. Wirth. Optimizing cost flows by edge cost and capacity upgrade. *Journal of Discrete Algorithms*, 2(4):407 – 423, 2004. The 26th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2000).

[GS13] M. Goerigk and A. Schöbel. Algorithm engineering in robust optimization. Technical report, Preprint-Reihe, Institut fr Numerische und Angewandte Mathematik, Universitt Göttingen, 2013. Submitted.

[HHK+11] H.W. Hamacher, S. Heller, W. Klein, G. Köster, and S. Ruzika. A sandwich approach for evacuation time bounds. In R. D. Peacock, E. D. Kuligowski, and J. D. Averill, editors, *Pedestrian and Evacuation Dynamics*, pages 503–513. Springer US, 2011.

[HT01] H. W. Hamacher and S. A. Tjandra. Mathematical modeling of evacuation problems: A state of the art. *In Pedestrian and Evacuation Dynamics (Schreckenberg, M. and Sharma, S. D. eds)*, 1964:227–266, 2001.

[KMN+98] S. O. Krumke, M. V. Marathe, H. Noltemeier, R. Ravi, and S.S. Ravi. Network improvement problems. *Network Design: Connectivity and Facilities Location, AMSDIMACS Volume Series in Discrete Mathematics and Theoretical Computer Science*, 40:247–268, 1998.

[KY97] P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications.* Kluwer Academic Publishers, 1997.

[L+14] A. Lemoine et al. Pligurian earthquake: Seismic and tsunami scenario modeling, from hazard to risk assessment towards evacuations planning. In *Proceedings of the Second European Conference on Earthquake Engineering and Seismology*, 2014.

[LKN11] G. Lämmel, H. Klüpfel, and K. Nagel. Risk minimizing evacuation strategies under uncertainty. In R. D. Peacock, E. D. Kuligowski, and J. D. Averill, editors, *Pedestrian and Evacuation Dynamics*, pages 287–296. Springer US, 2011.

[LM13] Y. Lin and K. Mouratidis. Best upgrade plans for large road networks. In M. A. Nascimento, T. Sellis, R. Cheng, J. Sander, Y. Zheng, H.-P. Kriegel, M. Renz, and C. Sengstock, editors, *Advances in Spatial and Temporal Databases*, volume 8098 of *Lecture Notes in Computer Science*, pages 223–240. Springer Berlin Heidelberg, 2013.

[NNL$^+$14] I. A. Ndiaye, E. Neron, A. Linot, N. Monmarche, and M. Goerigk. A new model for macroscopic pedestrian evacuation planning with safety and duration criteria. *Transportation Research Procedia*, 2(0):486 – 494, 2014. The Conference on Pedestrian and Evacuation Dynamics 2014 (PED 2014), 22-24 October 2014, Delft, The Netherlands.

[Old01] J. D. Oldham. Combinatorial approximation algorithms for generalized flow problems. *Journal of Algorithms*, 38(1):135 – 169, 2001.

[OMH09] S. Opasanon and E. Miller-Hooks. The safest escape problem. *Journal of the Operational Research Society*, 60:1749–1758, 2009.

[OZ07] F. Ordez and J. Zhao. Robust capacity expansion of network flows. *Networks*, 50(2):136–145, 2007.

[Rad98] T. Radzik. Faster algorithms for the generalized network flow problem. *Mathematics of Operations Research*, 23(1):69–100, 1998.

[SK98] S. Schwarz and S.O. Krumke. On budget-constrained flow improvement. *Information Processing Letters*, 66(6):291 – 297, 1998.

[Way99] K. D. Wayne. *Generalized Maximum Flow Algorithms.* PhD Thesis. Cornell University, New York, United States, 1999.

[XLW10] C. Xie, D.-Y. Lin, and S. T. Waller. A dynamic evacuation network optimization problem with lane reversal and crossing elimination strategies. *Transportation Research Part E: Logistics and Transportation Review*, 46(3):295 – 316, 2010.

[XT11] C. Xie and M. A. Turnquist. Lane-based evacuation network optimization: An integrated lagrangian relaxation and tabu search approach. *Transportation Research Part C: Emerging Technologies*, 19(1):40 – 63, 2011.

[Yam96] T. Yamada. A network flow approach to a city emergency evacuation planning. *International Journal of Systems Science*, 27(10):931–936, 1996.

A Proof of Theorem 1

Theorem 3. *FLI is strongly NP-complete, even if $c_i = 1$ for all $i \in V'$.*

Proof. Given a decision instance of FLI, it requires a polynomial number of operations to verify each constraint in order to answer *Yes* or *No*. Therefore, FLI is in NP.

To show NP-hardness, we consider the 3-partition problem, which is known as strongly NP-hard: Given a (multi)set M of $3N$ natural numbers a_i with $B/4 < a_i < B/2$ for all $i = 1, \dots, 3N$, where $B := 1/N \sum_{i=1}^{3N} a_i$. Find a partition of M into N sets $M_1, \dots, M_N$ such that the sum of numbers a_i in each set is equal to B.

Given an instance of 3-partition, we create an instance of FLI in the following way:

There is a source node s where NB evacuees start. Additionally, there is one node v_i for each $i = 1, \dots, 3N$, one node v'_k for each $k = 1, \dots, N$, nodes v''_{ik} for each $i = 1, \dots, 3N$ and $k = 1, \dots, N$, and one sink node t.

We have $p_i = 1$ and $\hat{p}_i = 0$ for all nodes v_i and v'_i; and $p_{ik} = 0$, $\hat{p}_{ik} = 1$ for all nodes v''_{ik}.

Furthermore, there are arcs from s to v_i for all $i = 1, \dots, 3N$ with capacity a_i; arcs from v_i to v''_{ik} and from v''_{ik} to v'_k with capacity a_i for all $i = 1, \dots, 3N$ and $k = 1, \dots, N$; and arcs from v'_k to t with capacity B for all $k = 1, \dots, N$. An example is presented in Figure 6.

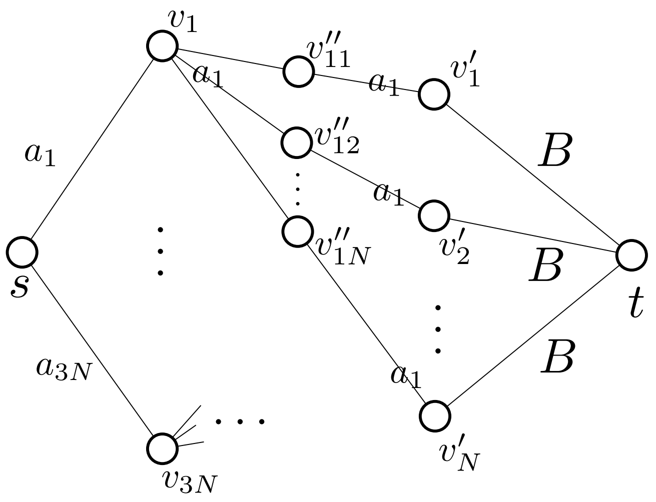


Figure 6: FLI problem built from 3-partition instance.

The improvement costs c_i are equal to 1 for all nodes, and the improvement budget equals N. We have that there exists a 3-partition if and only if the FLI problem has a solution with objective value NB.

To see this, let a 3-partition be given. If value a_i is in set M_k, then we improve node v''_{ik}, meaning the a_i units of flow can go from v_i to v'_k. As we have a 3-partition, there is exactly one possible path between every pair of nodes v_i and v'_k, and the total inflow in the nodes v'_k equals B. Thus, we have a choice of improvements that yields a flow with value NB.

On the other hand, let an optimal solution to the FLI instance with value NB be given. As NB units of flow start from s, there is no loss in the network. This means that for every node v_i, $i = 1, \dots, 3N$, there is exactly one improved node v''_{ik} (as there needs to be at least one such node to prevent loss, and the improvement budget enforces that not more than one such node can be improved). The node improvements can thus be considered as a partitioning of the elements $a_1, \dots, a_{3N}$ into N sets. As each node v'_k has B units of outgoing flow, this partition is a 3-partition with equal sums.

$\square$