A Capacitated Network Flow Optimization Approach for Short Notice Evacuation Planning

Gino J. Lim[a,*], Shabnam Zangeneh[b], M. Reza Baharnemati[a], Tiravat Assavapokee[c]

[a]*Department of Industrial Engineering, University of Houston, 4800 Calhoun Road, Houston, TX 77204*
[b]*Department of Industrial and Systems Engineering, University of Washington*
[c]*Walmart Co.*

Abstract

We present a capacity constrained network flow optimization approach for finding evacuation paths, flows and schedules so as to maximize the total evacuees for short notice evacuation planning (SNEP). Due to dynamic nature of this optimization problem, we first construct a time-expanded network that expands the static network over the planning horizon for every time interval. Since the resulting evacuation networks become extremely large to solve, we have developed Evacuation Scheduling Algorithm (ESA) to expedite the solution process. ESA utilizes Dijkstra's algorithm for finding the evacuation paths and a greedy algorithm for finding the maximum flow of each path and the schedule to execute the flow for each time interval. We show that the complexity of ESA is $O(|\mathcal{N}_c| \cdot n^2) + O(|\mathcal{N}_c| \cdot m \cdot T)$. Numerical experiments show a tremendous advantage of ESA over an exact algorithm (CCEP) in computation time by running up to 41,682 faster than CCEP. In many test network instances, CCEP failed to find a solution within 12 hours while ESA converged to a solution in less than 0.03 seconds.

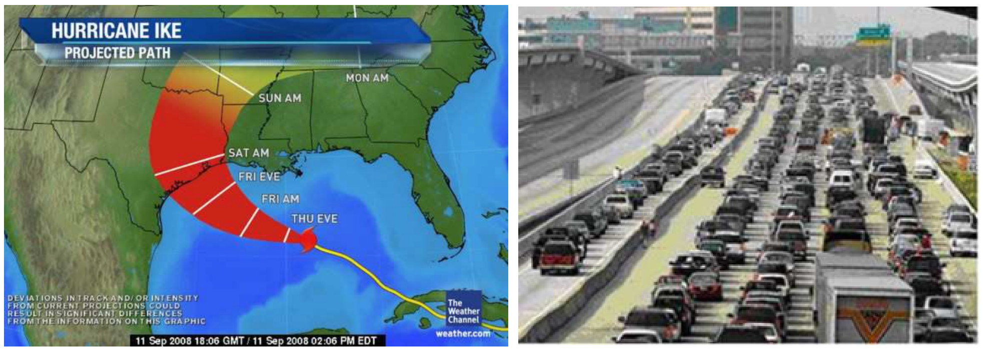
Keywords: Capacitated Network Flow Problem, Evacuation Planning, Time-expanded Network, Shortest Path.

*Corresponding author

Email addresses: `ginolim@uh.edu` (Gino J. Lim), `shzangeneh@hotmail.com` (Shabnam Zangeneh), `mbaharnemati@uh.edu` (M. Reza Baharnemati), `tiravata@gmail.com` (Tiravat Assavapokee)

1. Introduction

Hurricanes are cyclones that develop over the warm tropical oceans and have a sustained wind speed over 74 miles per hour. Hurricanes typically produce dangerous winds, torrential rains and flooding. All of them may result in tremendous property damage and loss of life in coastal populations. Recent hurricanes such as Ike (See Figure 1(a)), Gustav, Rita, and Katrina have given real situations to local and federal agencies for testing their ability in evacuating and safely relocating their residents. The massive traffic congestion (See Figure 1(b)) resulting from simultaneous evacuation of several million residents coupled with significant shortages of fuel and other basic necessities once again underscored the importance of a well-planned strategy in effectively evacuating large metropolitan areas.


HURRICANE IKE
PROJECTED PATH
MON AM
SUN AM
SAT AM
FRI EVE
FRI AM
THU EVE
DEVIATIONS IN TRACK AND/OR INTENSITY FROM CURRENT PROJECTIONS COULD RESULT IN SIGNIFICANT DIFFERENCES FROM THE INFORMATION ON THIS GRAPHIC
The Weather Channel
weather.com
11 Sep 2008 18:06 GMT / 11 Sep 2008 02:06 PM EDT

(a) A Hurricane Ike Path Trajectory captured from weather.com on September 11, 2008

(b) An Evacuation Traffic

Figure 1: Hurricane Evacuation

The evacuation route planning is a complex problem which has several aspects. One is analyzing the effects of different behavioral and managerial factors on evacuation (Perry, 1985; Vogt and Sorensen, 1992; Dow and Cutter, 1998; Drabek, 1999). Defining evacuation zones (Sorensen et al., 1992) is another aspect of the problem. These zones should be established before the disaster happens. Furthermore, shelters and safe places should be allocated (Sherali et al., 1991) and notified to evacuees in advance to reduce evacuation related risks and costs. One of the key aspects of the problem, which is investigated in this paper, is to determine the evacuation paths, flows and schedules in an efficient way. Depending on the types of evacuation networks (Smith, 1991; Southworth, 1991), two main approaches have been identified in the literature to develop mathematical formulation for evacuation route planning. The first one is based on the static network while the second one is based on the time-expanded network that considers network flow over time. In Cova and Johnson (2003), the routing plans for a regional evacuation are provided on a static network while minimizing the crossing conflicts at intersections because they believe that most traffic delays in regional evacuations occur at intersections. However, their approach does not provide the evacuation schedule, *i.e.,* how many times a specific route can be used during evacuation and when to evacuate. Lu et al. (2005) also use the static network to avoid the computational burden of the time-expanded network. They present a heuristic iterative algorithm Capacity Constrained Route Planner (CCRP) that produces a sub-optimal solution for the evacuation planning problem. Based on the static network, CCRP finds the minimum time horizon that ensures 100% evacuation. However, resulting evacuation paths are not necessarily useful in practice because the evacuation paths from CCRP allow intersection nodes to hold flow for some periods of time. In practice, intersection nodes should not hold flow.

Time-expanded networks more realistically describe the dynamic evacuation problem compared to the static mode. However, the resulting size of the network is often too large to solve in a reasonable time. In Hamacher and Tjandra (2002), the evacuation problem is formulated as a network flow optimization model. But its slow solution time is a major drawback of their approach for real size evacuation networks. Ford and Fulkerson (1958, 1962) first proposed a linear programming model for the maximal dynamic network flow problem with one source node and one destination node. But it was not designed to find the paths. Therefore, they proposed a *temporally repeated flows* algorithm that finds an optimal solution for the maximal dynamic network flow problem. The algorithm provides a set of paths and corresponding flows that maximize the total output flow from a single source node in a given period of time. This algorithm can be expanded to a network with multiple source nodes and destination nodes. However, it is not applicable to networks with limited amounts of supplies in the source nodes and limited capacities in the destination nodes such as in evacuation networks (Hoppe, 1995).

Another approach is to formulate the evacuation planning problem as a transshipment problem (Hoppe and Tardos, 2000) or more specifically as a dynamic transshipment problem (Herer and Tzur, 2001). However, in the dynamic transshipment problem, a specific demand in each time period for each destination node is required, which is different than the evacuation problems.

In this paper, we formulate the evacuation problem as a capacitated network flow problem on the time-expanded network. We determine the evacuation paths, their flows, and evacuation schedule (when to evacuate with how much flow via which path). We first attempt to develop an exact solution approach. However, evacuation networks are typically large. If we add the time component to the optimization problem, the corresponding model becomes extremely large scale that there are no known polynomial algorithms for solving such problems. Therefore, a heuristic algorithm is also developed for solving the evacuation problem that can generate high-quality solutions for large-scale evacuation network problems.

Our goal is to develop a decision making tool that deals with assigning evacuation routes and schedules to evacuees in different evacuation areas. We assume that decision makers have complete information on the number of evacuees in each area, the capacity and topology of transportation networks, and the path forecasts of approaching hurricanes. Based on the available information, we facilitate the evacuation process by providing clear temporal and spatial schedules and routes to evacuation vehicles by utilizing network optimization techniques.

The rest of the paper is organized as follows. Section 2 describes the evacuation planning problem, mathematical notation, and the network optimization formulations of the problem. Since the traditional maximal dynamic network flow formulation is designed to find optimal flows, it does not fit well for our evacuation planning problem. Therefore, we formulate SNEP as a *capacitated network flow problem* (CNFP) in Section 2.3 that determines optimal evacuation path(s), flow(s), and schedule(s) for each centroid in the network. In Section 3, Capacity Constrained Evacuation Planning algorithm (CCEP) is proposed to find an optimal solution to our problem. Since CCEP is computationally expensive for large-scale problems, a heuristic solution approach called *Evacuation Scheduling Algorithm* is developed in Section 4. Our approach is based on the shortest path algorithm for path generation and a greedy algorithm for flow generation to facilitate the solution time. In Section 5, numerical experiments are conducted for testing the performance of the heuristic algorithm on a number of different cases. Finally, we conclude our paper with a short summary in Section 6.

2. Methodology

2.1. Notation

In this section, we start by introducing mathematical notation used in this paper (see Table 1). We first consider our problem on a static network $G = (\mathcal{N}, \mathcal{A})$ that represents the transportation network in the area of interest. Because of the nature of the evacuation problem, we divide the set

Table 1: Static network notation

Notation	Description
$\mathcal{N}_d$	set of all impact nodes
$\mathcal{N}_S$	set of all safe nodes
$\mathcal{N}_C$	set of centroids, $\mathcal{N}_C \subset \mathcal{N}_d$
$\mathcal{N} = \{\mathcal{N}_d \cup \mathcal{N}_S\}$	set of all nodes
$\mathcal{A}$	set of all arcs in the network
t_i	impact time at node i
s_i	initial number of evacuation vehicles located at node i
u_i	maximum number of evacuation vehicles which can be located at node i per time period
$\vec{t}_{ij}$	travel time on the connecting road between node i and node j, ($\vec{t}_{ii} = 1$)
ϑ_{ij}	maximum number of evacuation vehicles which can enter into arc (i, j) per time period
Node $i \in \mathcal{N}$	physical location including impact nodes and safe nodes
Arc $(i, j) \in \mathcal{A}$	connecting road between node i and node j

of nodes $\mathcal{N}$ into two subsets: $\mathcal{N}_d$ (impact nodes) and $\mathcal{N}_S$ (safe nodes). The former includes all the nodes that are declared as evacuation zones. The latter contains all the safe nodes to which evacuees are trying to reach during the evacuation. Impact nodes are further classified as *centroids* ($\mathcal{N}_C$) and *intersections.* Centroids are the nodes with positive supplies whereas intersections do not have supplies. Evacuees from one centroid can pass through another centroid, but they are not allowed to stay in any visiting centroids during evacuation. Furthermore, we are given one parameter, t_i, to denote impact time of node i. *Impact time* is defined as the amount of time that is available for evacuees in node i to evacuate until the impact of the projected hurricane is made to the area. By definition, the impact time for a safe node is equal to infinity because safe nodes are not supposed to be affected by a hurricane. Other parameters are self-explanatory and definitions are given in Table 1.

A few assumptions are made for our model formulation. Since our goal is to find a solution that works well for the worst case scenario, the transit time is estimated based on the peak rush hour travel time on a major highway, which is a conservative estimate of the travel time. A set of impact nodes and a set of safe nodes are given to the model *a priori*. The impact nodes in the network are categorized into different regions based on their priorities for evacuation. The number of evacuation regions depends on the number of nodes in the network as well as the network size.

2.2. Dynamic Network Construction

Due to dynamic nature of this optimization problem, we first construct a *dynamic network* (or *time-expanded network*) that expands the static network over the planning horizon for every time interval. We first explain how traditional time-expanded networks are constructed. Then a more refined network construction approach is described to construct smaller networks.

2.2.1. Traditional time-expanded network construction

Time-expanded networks were introduced by Ford and Fulkerson (Ford and Fulkerson, 1958) to maximize the flow from the source node to the sink node in a network within a given time period T. In this approach, all the nodes and the arcs of the static network are duplicated at each time period. The resulting arcs are called *movement arcs* $\mathcal{A}_M$. There are also *holdover arcs* $\mathcal{A}_H$ for each node which hold the flow of that node for one time period. Figure 2 shows a small example of a static network to explain the construction of the corresponding time-expanded network. We slightly modify the mathematical notation for constructing the time-expanded network. Let $G^\tau = (\mathcal{N}^\tau, \mathcal{A}^\tau)$ represent a time-expanded network of a static network $G = (\mathcal{N}, \mathcal{A})$ over a given evacuation planning horizon T. Where, $\mathcal{N}^\tau = \{i_t | i \in \mathcal{N}; t = 0, \ldots, T-1\}$ is the set of nodes in the time-expanded network, $\mathcal{A}_M = \{(i_t, j_{\bar{t}}) | (i,j) \in \mathcal{A}; \bar{t} = t + \vec{t}_{ij} \leq T; t = 0, \ldots, T-1\}$ is the set of all movement arcs while $\mathcal{A}_H = \{(i_t, i_{t+1}) | i \in \mathcal{N}; t = 0, \ldots, T-2\}$ is the set of all holdover arcs. Therefore, the set of arcs in the time-expanded network is the union of the two arc sets, i.e., $\mathcal{A}^\tau = \{\mathcal{A}_M \cup \mathcal{A}_H\}$. In addition, the capacity of an arc (a, b),

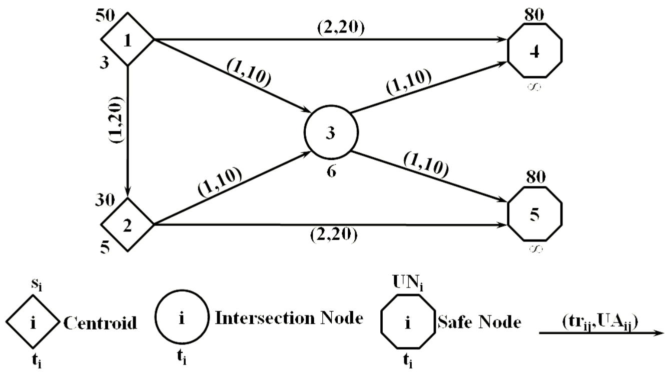


Figure 2: Static network $G = (\mathcal{N}, \mathcal{A})$

$\bar{\mathcal{U}}_{a,b}$, in the time-expanded network is

$$\bar{\mathcal{U}}_{ab} = \begin{cases} \vartheta_{ij}, & \text{if } (a,b) \in \mathcal{A}_M \text{ and } a = i_t,\ b = j_{t+\vec{t}_{ij}} \text{ for some } t \in \{0,\dots,T-1\} \\ u_i, & \text{if } (a,b) \in \mathcal{A}_H \text{ and } a = i_t,\ b = i_{t+1} \text{ for some } t \in \{0,\dots,T-2\} \end{cases}$$

Note that the flow of an arc (i,j) is bounded by its own arc capacity ϑ_{ij} if it is a movement arc at node i at time t. If the arc is a holdover arc, it is just an imaginary arc that represents the supply of the node from time t to $t+1$. Therefore, the flow of such an arc (i_t, i_{t+1}) is bounded by either its supply (if it is a centroid) or the capacity (for a safe node) at node i at time t, u_i. Note that we assume zero capacity for any nodes that are not either a source (or starting) node or a destination node, i.e., an intersection node is not allowed to hold any supplies from other impact nodes.

2.2.2. Reduced network construction

In order to minimize unnecessary computational burden, we take the following steps to a reduced version of the traditional time-expanded network. We first add two imaginary nodes J^* (*Super-safe* node) and J' (*Unsafe* node) to G^τ. Then, each safe node i_{T-1}, $\forall i \in \mathcal{N}_S$, in G^τ is connected to node J^* through the arc (i_{T-1}, J^*), with capacity $\bar{\mathcal{U}}_{i_{T-1},J^*} = u_i$, and each centroid i_0, $\forall i \in \mathcal{N}_C$, is connected to node J' through arc (i_0, J'), with capacity $\bar{\mathcal{U}}_{i_0,J'} = s_i$. The *Super-safe* node is assumed to be the final destination for all the evacuees, while *Unsafe* node is considered as a shelter location for evacuees who could not evacuate for various reasons. Note that we use J^* in the static network as its final destination for path generation. Thus, all safe nodes, $\mathcal{N}_S$, are connected to J^* in G.

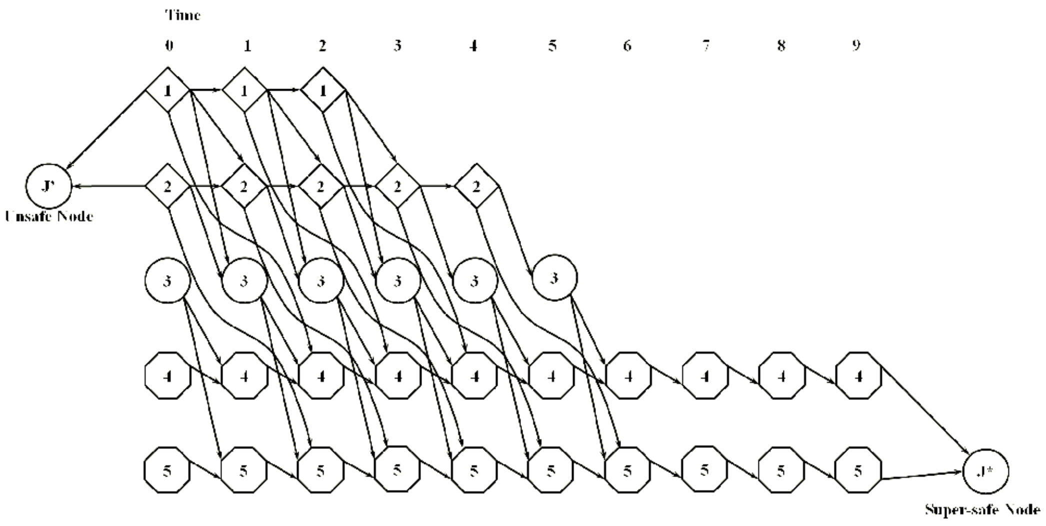


Figure 3: A modified time-expanded network construction $G^\tau = (\mathcal{N}^\tau, \mathcal{A}^\tau)$

In practice, our evacuation network is smaller than the traditional network. Since evacuees have t_i time to evacuate, some nodes and arcs should be omitted in the network construction; (1) node i_t, $\forall i \in \mathcal{N}_d$, such that $t \geq t_i$, (2) arcs $(i_t, j_{t+\vec{t}_{ij}})$ such that $t + \vec{t}_{ij} \geq t_j$, and (3) arcs $(i_t, j_{t+\vec{t}_{ij}})$ such that $t \geq t_i$. Furthermore, since the intersection nodes do not hold any population, holdover arcs are not included if they are associated with an intersection node. An example of our proposed time-expanded network with two imaginary *Super-safe* and *Unsafe* nodes are depicted in Figure 3.

2.3. Capacitated Network Flow Problem (CNFP) Formulation

Our primary optimization model can be formulated as a capacitated network flow problem that sends flows from a set of origin locations to a set of destinations while sharing the capacity of the arcs with other paths. Note that each centroid is allowed to have multiple paths.

2.3.1. Evacuation prioritization for centroids

Suppose that we have R regions in the network each of which consists of a subset of impact nodes. Each centroid in region $r \in \{1, \cdots, R\}$ has one or more evacuation paths. We define η_r as the total number of centroids in region r. Parameter w_r is defined as the weight of region r. The concept of region weights is linked to the priority of the regions for evacuation. There are several factors which affect this priority such as distance of the region from the center of hurricane, flood elevation, and population density. We first assign the region numbers based on this priority, i.e. the region with the lowest region number will have the highest priority for evacuation. In addition, we normalize the weights of the regions so that the summation of all the weights equals one. The formulation that embraces both the logic of prioritization and normalization is as follow

$$w_r = \frac{R - r + 1}{R(R+1)/2}, \; r \in \{1, \cdots, R\}. \tag{2.1}$$

Furthermore, we impose the normalized weights not only for each region, but also for each centroid in each region. This further normalization is obtained by including the number of centroids in each region into the above formulation. For each region r, the following new formulation calculates the weight for its centroids

$$\hat{w}_i = \frac{w_r}{\sum_{j=1}^{R}(w_j \cdot \eta_j)}, \;\; \forall i \in \mathcal{N}_r \subseteq \mathcal{N}_C \text{ and } \mathcal{N}_C = \bigcup_{r=1}^{R} \mathcal{N}_r, \tag{2.2}$$

where $\mathcal{N}_r$ is a set of centroids that belong to region r. Note that equation (2.2) implies that all centroids in one region has the same weight (i.e., evacuation priority).

2.3.2. Mathematical Model

Our decision variables to the optimization model are x_{itjk}, $\xi_{iJ'}$, and y_{ijk}, and they are defined as:

$x_{itjk} =$	The number of evacuation vehicles leaving node i to node j at time t using path k,
$\xi_{iJ'} =$	The number of evacuation vehicles leaving node i to node J',
$y_{ijk} =$	1, if arc (i, j) belongs to the path k; 0, otherwise.

In addition to notation defined in Table 1, we introduce δ_i as a set of paths originating from centroid i, $\forall i \in \mathcal{N}_C$, and subsequently $\Omega = \bigcup_{i \in \mathcal{N}_c} \delta_i$ as total outgoing paths from all centroids. Our objective is to maximize the total number of evacuation vehicles (or flow) for all centroids from the R

regions to safe destinations. Depending on the urgency of the evacuation, each region is assigned with a different priority weight. If a region r needs to be evacuated first, the highest weight will be assigned to the centroids in the region. Therefore, our objective function is defined as

$$\text{Maximize } z_{cnfp} = \sum_{i \in \mathcal{N}_C} \sum_{k \in \delta_i} \sum_{j | (i_0, j_{0+\vec{t}_{ij}}) \in \mathcal{A}^\tau} \hat{w}_i \cdot x_{i0jk} \tag{2.3}$$

This objective function is to maximize the total weighted outgoing flows from all centroids at time zero. In reality, some evacuees will depart a centroid at time zero and others will evacuate based on their schedule. The former follows the movement arcs while the latter follows the hold-over arcs to the next time interval. Therefore, all evacuees will leave the centroids at time zero in our time-expanded evacuation network.

The next set of constraints are the flow balance equations:

$$\sum_{k \in \delta_i} \sum_{j | (i_t, j_{t+\vec{t}_{ij}}) \in \mathcal{A}^\tau} x_{itjk} + \xi_{iJ'} = s_i, \ \forall i \in \mathcal{N}_C, t = 0, \tag{2.4}$$

$$\sum_{k \in \delta_i} \sum_{j | (i_t, j_{t+\vec{t}_{ij}}) \in \mathcal{A}^\tau} x_{itjk} = 0, \ \forall i \in \mathcal{N} \setminus \mathcal{N}_C, t = 0, \tag{2.5}$$

$$\sum_{j | (i_t, j_{t+\vec{t}_{ij}}) \in \mathcal{A}^\tau} x_{itjk} - \sum_{j | (j_{t-\vec{t}_{ji}}, i_t) \in \mathcal{A}^\tau} x_{j(t-\vec{t}_{ji})ik} = 0, \ \forall i \in \mathcal{N}, t \in \{1, \dots, T-1\},$$

$$, \forall k \in \Omega \tag{2.6}$$

$$\sum_{k \in \Omega} \sum_{i \in \mathcal{N}_s} x_{itJ^*k} + \sum_{i \in \mathcal{N}_C} \xi_{iJ'} - \sum_{i \in \mathcal{N}_C} s_i = 0, \ t = T-1. \tag{2.7}$$

Constraint 2.4 ensures that at time 0, all evacuees at each centroid are either sent to the intersection nodes or to the unsafe node, J'. Constraint 2.5 states that the total outgoing flow from intersection nodes is equal to zero because they do not hold evacuees. Constraint 2.6 guarantees that the total incoming flow to node i at each time period per evacuation path is equal to the total outgoing flow from the same node at the same time period on the same path. Finally, Constraint 2.7 satisfies the conservation of flow, i.e., all evacuees eventually either reach the super-safe node or they are sent to the unsafe node. However, in the objective function, the goal is to maximize the number of evacuees who can be safely evacuated to the super-safe node.

We add a bundle constraint for the movement arcs and the hold-over arcs to the model:

$$\sum_{k\in\Omega} x_{itjk} \leq \bar{\mathcal{U}}_{i_t, j_{t+\vec{t}_{ij}}}, \forall (i_t, j_{t+\vec{t}_{ij}}) \in \{\mathcal{A}^\tau \cup \{(i_{T-1}, J^*) | i \in \mathcal{N}_S\}\},$$
$$t \in \{0, \ldots, T-1\}. \quad (2.8)$$

The path generation constraints are defined as:

$$\sum_{j|(i,j)\in\mathcal{A}} y_{ijk} - \sum_{j|(j,i)\in\mathcal{A}} y_{jik} = 1, \qquad \forall i \in \mathcal{N}_C, \forall k \in \delta_i, \quad (2.9)$$

$$\sum_{j|(i,j)\in\mathcal{A}} y_{ijk} - \sum_{j|(j,i)\in\mathcal{A}} y_{jik} = 0, \qquad \forall i \in \mathcal{N}, \forall k \in \Omega, \text{if } i \neq \text{source}(k), \quad (2.10)$$

$$\sum_{j|(i,j)\in\mathcal{A}} y_{ijk} - \sum_{j|(j,i)\in\mathcal{A}} y_{jik} = -1, \quad \forall k \in \Omega, i = J^*. \quad (2.11)$$

These constraints guarantee that flow of each path starts form its origin, continues along the path, and eventually ends at its destination. The constraint

$$x_{itjk} \leq \vartheta_{ij} \cdot y_{ijk}, \quad \forall (i,j) \in \{\mathcal{A} \cup \{(i, J^*) | i \in \mathcal{N}_S\}\}, t \in \{0, \ldots, T-1\}, \forall k \in \Omega \quad (2.12)$$

ensures that flow of a path can exist on an arc if and only if the arc belongs to the path. There is a practical issue with the current formulation that a flow may occur from a source node to another source node through a path in time-expanded networks. This becomes a problem only if the flow stops at the second source node for a time period. It means that some evacuees move from one centroid to another centroid and stay there for a while without any movements. This defeats the purpose of the evacuation planning: "non-interrupted movement". In Section 2.2.1, we mentioned that holdover arcs of intersection nodes are not allowed to hold flow, but the source nodes. In order to have a practical solution as mentioned above, we add further limitation on this constraint that flow on holdover arcs allowed only for the source node of path k, source(k). The corresponding constraint is

$$x_{itik} = 0, \forall k \in \Omega, \forall t \in \{0, 1, \ldots, T-1\}, i \in \{\mathcal{N}_C \setminus \text{source}(k)\}. \quad (2.13)$$

Finally, we add the non-negativity constraints for all variables to the optimization model:

$$\begin{array}{ll} x_{itjk} \in Z_+, & (i_t, j_{t+\vec{t}_{ij}}) \in \{\mathcal{A}^\tau \cup \{(i_{T-1}, J^*) | i \in \mathcal{N}_S\}\}, t \in \{0, 1, \ldots, T-1\}, \\ & , \forall k \in \Omega, \\ \xi_{iJ'} \in Z_+, & i \in \mathcal{N}_C, \\ y_{ijk} \in \{0, 1\}, & \forall (i, j) \in \mathcal{A}, t \in \{0, \ldots, T-1\}, \forall k \in \Omega. \end{array}$$

This completes the model formulation. As an input to CNFP, we need to assign an upper bound on the number of paths to each centroid to select a subset of the paths and their optimal flows. However, this bound is not known in advance and it affects the size of the model. Therefore, we attempt to find an optimal number of paths to use for each centroid using Capacity Constrained Evacuation Planning algorithm in Section 3. A stopping criterion of CCEP is the maximum number of vehicles that can be safely evacuated from all centroids within a given evacuation time horizon. This maximum number is obtained by relaxing CNFP on path, *i.e.*, it only optimizes for the maximum flow. The resulting model becomes a maximal dynamic network flow model that optimizes the number of vehicles without concerning the evacuation paths (see Section 2.4) and it is an input to CCEP. The final output of CCEP will be the evacuation plan that includes evacuation paths, their flows, and schedules.

2.4. A priority based maximal dynamic network flow optimization model for generating an upper bound of the maximum weighted sum of evacuees

We denote *the maximum weighted sum of evacuees* as the total sum of weights (i.e., priority values) times the maximum number of vehicles that can be safely evacuated from all centroids to their destinations within a given time horizon. First, let x_{itj} be a non-negative integer variable that represents the number of evacuation vehicles leaving node i to node j at time t. Then, the optimization model (IP_{ubd}) for generating an upper bound on the maximum number of evacuation vehicles can be formulated as follows:

$$\max \quad z_{ubd} = \sum_{i \in \mathcal{N}_C} \sum_{j | (i_0, j_{0+\vec{t}_{ij}}) \in \mathcal{A}^\tau} \hat{w}_i \cdot x_{i0j}, \tag{2.14}$$

$$\text{s.t.} \sum_{j | (i_t, j_{t+\vec{t}_{ij}}) \in \mathcal{A}^\tau} x_{itj} + \xi_{iJ'} = s_i, \forall i \in \mathcal{N}_C, t = 0, \tag{2.15}$$

$$\sum_{j|(i_t, j_{t+\vec{t}_{ij}}) \in \mathcal{A}^\tau} x_{itj} - \sum_{j|(j_{t-\vec{t}_{ji}}, i_t) \in \mathcal{A}^\tau} x_{j(t-\vec{t}_{ji})i} = 0, \forall i \in \mathcal{N}, t \in \{1, \ldots, T-2\}, \tag{2.16}$$

$$\sum_{i \in \mathcal{N}_S} x_{itJ^*} + \sum_{i \in \mathcal{N}_C} \xi_{iJ'} = \sum_{i \in \mathcal{N}_C} s_i, t = T-1, \tag{2.17}$$

$$x_{itj} \leq \bar{\mathcal{U}}_{i_t, j_{t+\vec{t}_{ij}}}, \forall (i,j) \in \{\mathcal{A} \cup \{(i, J^*) | i \in \mathcal{N}_S\}\}, t \in \{0, \ldots, T-1\}, \tag{2.18}$$

$$x_{itj} \in Z_+, \quad t \in \{0, \ldots, T-1\}, (i_t, j_{t+\vec{t}_{ij}}) \in \{\mathcal{A}^\tau \cup \{(i_{T-1}, J^*) | i \in \mathcal{N}_S\}\}$$

$$\xi_{iJ'} \in Z_+, i \in \mathcal{N}_C.$$

The objective function (2.14) is to maximize the weighted sum of evacuees. Constraints (2.15), (2.16), and (2.17) form our flow balance equations. Constraint (2.15) implies that the supply of each centroid consists of two groups of vehicles that evacuate either to safe destinations or the shelters. Constraint (2.16) describes the balance of flow for time $t = 1, \ldots, T-2$. Constraint (2.17) states that the total evacuees at time $T-1$ must be equal to the total supplies at the beginning of evacuation. Constraints (2.18) is the capacity constrains for the flows of the movement arcs and holdover arcs, respectively.

Theorem 2.1. *The optimal solution of IP_{ubd} in Section 2.4 gives an upper bound to the optimal objective value of the CNFP model, i.e., $z^*_{cnfp} \leq z^*_{ubd}$.*

Proof. Part I: Objective Function – We first show that both CNFP and IP_{ubd} models have essentially the same objective function. A key component of this proof is based on the following definition

$$\sum_{k \in \Omega} x_{itjk} = x_{itj}, \quad \forall (i_t, j_{t+\vec{t}_{ij}}) \in \{\mathcal{A}^\tau \cup \{(i_{T-1}, J^*) | i \in \mathcal{N}_S\}\}. \tag{2.19}$$

By this definition, we make the claim as follows.

$$\begin{aligned} z_{cnfp} &= \sum_{i\in\mathcal{N}_C}\sum_{k\in\delta_i}\sum_{j|(i_0,j_{0+\vec{t}_{ij}})\in\mathcal{A}^\tau} \hat{w}_i\cdot x_{i0jk} \\ &= \sum_{i\in\mathcal{N}_C}\sum_{j|(i_0,j_{0+\vec{t}_{ij}})\in\mathcal{A}^\tau} \hat{w}_i\cdot\left(\sum_{k\in\delta_i} x_{i0jk}\right) \\ &= \sum_{i\in\mathcal{N}_C}\sum_{j|(i_0,j_{0+\vec{t}_{ij}})\in\mathcal{A}^\tau} \hat{w}_i\cdot x_{ijk} = z_{ubd}. \end{aligned}$$

Part II: Feasible Region – Now, we show that the feasible region of CNFP is tighter than that of IP_{ubd}. Note that IP_{ubd} does not have the path definition. Therefore, we will focus on the CNFP constraints and show that some of the constraints are equivalent to those of IP_{ubd}, some other constraints are simply redundant in IP_{ubd}, and the rest of the constraints make the CNFP model tighter than the IP_{ubd} model.

1. *Flow Balance Equations:* Based on the definition of (2.19), it is easy to see that constraints (2.4) and (2.7) are identical to (2.15) and (2.17), respectively. When we apply a summation over k on both sides of (2.6), it becomes (2.16).
2. *Bundle Constraints:* By the definition given in (2.19), constraints (2.8) is identical to (2.18).
3. *Non-negativity Constraints:* Since $x_{itjk} \geq 0$, $x_{itj} = \sum_k x_{itjk} \geq 0$. Therefore, $x_{i0J'}, x_{itJ^*} \geq 0$ holds true in IP_{ubd}.
4. *Extra Constraints:* Finally, constraints (2.13) is the additional constraint along with the path generation constraints (2.9)–(2.11) to claim that the feasible region of CNFP is tighter than that of IP_{ubd}.

Therefore, $z^*_{cnfp} \leq z^*_{ubd}$. □

3. Capacity Constrained Evacuation Planning Algorithm

The IP_{ubd} model in Section 2.4 finds the maximum weighted sum of evacuees that the current network can provide. Our next task is to find a way to achieve an optimal solution to our evacuation problem using CCEP. Since the number of paths per centroid is assumed to be given in advance, we determine the paths, flows, and schedule in CCEP and its procedure is described below.

First, we solve the IP_{ubd} model discussed in Section 2.4 to obtain the optimal objective value z^*_{ubd}. Based on the solution, we update the remaining supply (RS^i_{ubd}) for centroid $i \in \mathcal{N}_C$, i.e., $RS^i_{ubd} = \xi_{iJ'}$. The optimal objective value z^*_{ubd} is used in the algorithm termination criterion while the value of RS^i_{ubd} is used to check the progress of the algorithm.

Step 0 Solve the IP_{ubd} model in Section 2.4

- Obtain z^*_{ubd};
- $RS^i_{ubd} \leftarrow \xi_{iJ'}, \forall i \in \mathcal{N}_C$;

Next, model parameters are initialized. We start by assuming that one path is allowed per centroid in the CNFP model, ($\Omega = \{\omega_1, \omega_2, \ldots, \omega_{|\mathcal{N}_C|}\}$ and $\delta_i = \{\omega_i\}, \forall i \in \mathcal{N}_C$), where ω_i represents the memory space of the i^{th} path that is defined in Step 1 and generated in Step 2. As an input parameter to CNFP, the counter *cnt* is the maximum number of paths allowed in the model. The value of *cnt* is initially set to the cardinality of the centroids, $cnt = |\Omega|$.

Step 1 Initialization

- $\Omega = \{\omega_1, \omega_2, \ldots, \omega_{|\mathcal{N}_C|}\}$;
- $\delta_i = \{\omega_i\}, \forall i \in \mathcal{N}_C$;
- $cnt \leftarrow |\Omega|$;

In Step 2, given the value of *cnt*, we solve the CNFP model by assuming that there are $|\delta_i|$ paths for node i, i.e., $cnt = \sum_{i \in \mathcal{N}_C} |\delta_i|$. Based on the solution, we update the remaining supply (RS^i_{cnfp}) for every centroid in $\mathcal{N}_C$. The parameters RS^i_{ubd} and RS^i_{cnfp} are essentially used for deciding which of the centroids will need more paths in the next step.

In Step 3, we compare the objective values z^*_{cnfp} and z^*_{ubd}. We use two stopping criteria for termination.

Criterion 1:. If the total numbers of evacuees of both models are same, it means that we already have sufficient number of paths for evacuation. Therefore, the current solution is optimal and the algorithm terminates. However,

Step 2 Solve the CNFP model to generate paths and flows

```
    INPUT: $cnt, |\delta_i|, \forall i \in \mathcal{N}_C$
    OUTPUT: $z^*_{cnfp}$ and $\delta_i, \forall i \in \mathcal{N}_C$
    Update: $RS^i_{cnfp} \leftarrow \xi_{iJ'}, \forall i \in \mathcal{N}_C$
```

Step 3 Update path information

```
if Stopping criteria are satisfied then
    Stop! Accept the current solution.
else
    for $i \in \mathcal{N}_C$ do
        if $RS^i_{cnfp} > RS^i_{ubd}$ then
            $cnt \leftarrow cnt + 1$
            $\delta_i = \delta_i \cup \{\omega_{cnt}\}$
            $\Omega = \Omega \cup \{\omega_{cnt}\}$
        end if
    end for
    Go to Step 2.
end if
```

if the weighted sum of evacuees in CNFP is less than that of the IP_{ubd} model ($z^*_{cnfp} < z^*_{ubd}$), then some of the centroids still have positive remaining supply, i.e, $\exists i \in \mathcal{N}_C$ such that $RS^i_{cnfp} > 0$. Since RS^i_{ubd} provides a lower bound to RS^i_{cnfp}, a centroid whose RS^i_{cnfp} is larger than RS^i_{ubd}, one path is added to ensure a higher evacuation. As a result, *cnt* gets updated and a space is reserved for this new path ω_{cnt} that is to be determined in Step 2. Then, it goes back to Step 2.

Criterion 2:. Since z^*_{ubd} is an upper bound of z_{cnfp}, CCEP may be repeated without making progress in CNFP. Therefore, the algorithm also terminates if there are no improvements in CNFP in γ consecutive times, $\gamma > 1$, and it is the second stopping criterion.

Theorem 3.1. *CCEP finds an optimal set of evacuation paths that maximize the weighted sum of evacuees.*

Proof. CCEP terminates when one of two stopping criteria is met. If it is terminated by the first criterion (i.e., $z^*_{cnfp} = z^*_{ubd}$), the minimum number of paths to have the maximum weighted sum of evacuees is found. As described in proof of Theorem 2.1, CNFP has a tighter feasible region than that of IP_{ubd}, i.e., a feasible solution of IP_{ubd} may not be feasible for CNFP. Hence, when the algorithm is terminated by the second criterion, adding extra paths to the centroids that still has remaining supply does not improve the objective function value in the future iterations. Therefore, the solution found in the current iteration is optimal. □

4. Evacuation Scheduling Algorithm (ESA)

Finding the optimal solution using CCEP can be computationally challenging due to the size of the network. To overcome the computational burden, we propose Evacuation Scheduling Algorithm. ESA utilizes the *Dijkstra's algorithm* (Edsger, 1959) to find the shortest path and the *greedy heuristic algorithm* discussed in Section 4.1 for finding the maximum flow of the path on the network.

4.1. A greedy algorithm for flow generation

Ford-Fulkerson algorithm (Ford and Fulkerson, 1962) is a typical choice for finding the maximum flow over a network. We simplify this algorithm to find the maximum flow over a path, $\mathbb{P}$, in the static network for each time interval. The maximum flow of $\mathbb{P}$ in the static network is the capacity of $\mathbb{P}$, which is the minimum capacity of the arcs associated with $\mathbb{P}$. This method can be generalized for finding the maximum flow of $\mathbb{P}$ over the time-expanded network. In $\mathbb{P}$, there is no connection between two nodes from one time period to the next in the time-expanded network. This rule does not apply to the source and the destination nodes. Therefore, the maximum flow of $\mathbb{P}$ over the time-expanded network will be the total sum of the path capacity over different time periods.

The greedy algorithm for finding the maximum flow of $\mathbb{P}$ over the time-expanded network consists of the following steps. First, the maximum flow value, MFV, is set to zero. Next, for each time period, $t \in \{0, 1, \dots, T-1\}$, the path capacity, $\text{Cap}^t_{\mathbb{P}}$, is calculated such that the path capacity at time t is defined as the minimum flow capacity among the arcs associated with $\mathbb{P}$ at t. Finally, we add $\text{Cap}^t_{\mathbb{P}}$ to MFV.

Comparing the greedy algorithm to Ford-Fulkerson algorithm, we do not need to consider two-way arcs and augment the augmenting paths. So, the

```
Greedy Algorithm
$MFV \leftarrow 0$;
for $t \in \{0, 1, \ldots, T-1\}$ do
    find the capacity of $\mathbb{P}$ at time $t$, $\text{Cap}^t_{\mathbb{P}}$.
    $MFV \leftarrow MFV + \text{Cap}^t_{\mathbb{P}}$
end for
```

complexity of the greedy algorithm can be estimated as $O(m \cdot T)$ comparing to the complexity of Ford-Fulkerson algorithm, $O(m \cdot f)$, where m is the number of arcs and f is the maximum flow in the time-expanded network.

4.2. ESA Solution Procedure

The objective of our heuristic is to find the shortest evacuation path(s) from each centroid to the safe area, and to push the maximum possible flow through this path(s). A dummy safe node $< d >$ is added to the static network with an infinite capacity, an infinite impact time, and a zero supply. All safe nodes are connected to this dummy safe node through arcs with the capacity of the safe node and zero traveling time. These dummy nodes and arcs are used to find the shortest distance from centroids to the closest safe node. Given the shortest path, the greedy algorithm is applied on the time-expanded network to obtain the maximum possible number of evacuees that can flow through the path. If this path does not have enough capacity to evacuate all the evacuees coming out from the centroid, a second shortest path will be generated, and so on. Finally, all the evacuation paths, flows, and leaving times for each centroid are reported as the output of the algorithm.

Step 1 Initialization

- Assign unique identification numbers to all nodes $\mathcal{N}$ in the network G.
- Create a time-expanded network based on the static network.
- Cluster G into R regions so that each centroid belongs to a region (or a set).
- $r \leftarrow 1$.

We now describe the steps of ESA. First, the network G is divided into R regions based on the evacuation priority (See equation (2.2)). Each centroid of the network is assigned to one region based on the region definition discussed in Section 2.3. We then construct the corresponding time-expanded network. We select region 1 as the first region for evacuation route planning as described in Step 1. In Step 2, we select region r and move to Step 3.

Step 2 Region Selection

- Select set $< r >$ and go to Step 3.

In Step 3, if the selected region still contains at least one centroid (i.e., set $< r >$ is not empty), select a centroid (v) and move to Step 4. Otherwise, a region with the next highest priority is selected, i.e., $r = r + 1$. If there is another region to consider, $r \leq R$, then move to Step 2; else, the algorithm terminates. In this step, centroids are selected in the same way that we did in CNFP model. As described in Section 2.3.1, all centroids in one region has the same priority. Therefore, centroid v is selected randomly from region r. Note that regions are selected based on their priority.

Step 3 Node Selection

```
if set $< r >$ is not empty then
  select a centroid $< v >$ from set $< r >$ and go to Step 4.
else
  $r \leftarrow r + 1$
  if $r \leq R$ then
    go to Step 2.
  else
    stop!
  end if
end if
```

In Step 4, a counter UC is reset to 0. UC is to count how many times already existing paths were re-generated in Step 5. This is to prevent the path generation step from generating the same path over and over again. More details about the use of UC are given in Step 6.

A shortest path $\mathbb{P}_v$ is generated using the *Dijkstra's algorithm* from the current node v to *Super-Safe* node d in Step 5. We use the travel times of

Step 4 Reset Counter

- $UC \leftarrow 0$

all arcs connecting those two nodes as the distance measure. Once a path is found, we update travel times by adding a penalty term $\Delta_{ij} > 0$ to the original $\vec{t}_{ij}$ as follow

$$\Delta_{ij} = \left\lceil \mu \cdot \frac{\varsigma_v}{m_{\mathbb{P}_v}} \right\rceil, \forall (i,j) \in \mathbb{P}_v, \tag{4.1}$$

where the parameter $\mu > 0$ is a penalty factor to the travel time of the arcs in the generated path $\mathbb{P}_v$, $m_{\mathbb{P}_v}$ is the number of arcs in the path $\mathbb{P}_v$, and $\varsigma_v = s_v/\bar{\mathcal{U}}_{\mathbb{P}_v}$ is a *path utilization factor* in which s_v is the supply of node v and $\bar{\mathcal{U}}_{\mathbb{P}_v}$ is the capacity of the path $\mathbb{P}_v$. A large ratio of ς_v means that the utilization of path $\mathbb{P}_v$ will be substantially lower in the next iteration because arcs in this path are highly utilized in the current iteration. Therefore, a higher penalty will be imposed on all arcs in this path. Suppose now that two paths have the same ratio ς_v, then a path with a higher $m_{\mathbb{P}_v}$ will have a lower penalty so that some arcs in the path may be re-utilized in the future paths.

The reason for this travel time update is because the same path may be discouraged to be used in the future path generation. Note that, in Step 8, we aim to push as much flow as possible through the path. Therefore, the remaining capacity of the same path in the next iteration will likely to be low once it is used.

Step 5 Path Generation

- Find the shortest path from centroid $< v >$ to node $< d >$ in static network using the travel time $\vec{t}_{ij}$ as distance between nodes i and j.
- Update $\vec{t}_{ij}$ for each arc in the path in the static network: $\vec{t}_{ij} \leftarrow \vec{t}_{ij} + \Delta_{ij}$.

A path is called *unique* if it is not previously recorded as one of the evacuation paths of node v. In Step 6, we check if the generated path $\mathbb{P}_v$ in Step 5 is unique. If it is, we find the corresponding flow for the path in Step 8. If $\mathbb{P}_v$ is not unique, attempts are made to generate a unique path for a finite number of times κ. If it fails, then we remove node v from the current set r. If the value of κ is large, there will be a higher chance to find a unique path. But it will increase the algorithm running time. Based on our experiments, $\kappa = 10$ seems to be working reasonably well. Note that there can be a situation where a node in region r may not have a feasible path. This means that the supply on this node cannot be evacuated. In this case, evacuees in those nodes need to be evacuated to nearest shelters.

Step 6 Unique Path Identification

```
if the path is unique then
    go to Step 7.
else
    $UC \leftarrow UC + 1$.
    if $UC > \kappa$ then
        remove node $< v >$ from set $< r >$.
        go to Step 3.
    else
        go to Step 5.
    end if
end if
```

In Step 7, given the path $\mathbb{P}_v$ from node v to node d, a maximum flow of the path is obtained using the *greedy algorithm*. In this step, we ensure that the flow value cannot exceed the supply of the node.

Step 7 Maximum Flow Generation

- Find the maximum flow of path $\mathbb{P}_v \leftarrow$ *greedy algorithm.*

  Output: $flow(\mathbb{P}_v, t)$, $\forall t$

$$\text{Maximum Flow}(\mathbb{P}_v) := f_{max} = \sum_t flow(\mathbb{P}_v, t)$$

Our final step is to check if the current path can push a positive flow. If it can, the solution of the current iteration is saved, that includes the path information, evacuation starting times, and their corresponding flows. We then update the capacity of the arcs in the time-expanded network by reducing the capacity by the amount of $flow(\mathbb{P}_v, t)$. Furthermore, the supply of node v is also reduced by f_{max}. If the resulting supply of node v becomes zero, the node is removed from the set r, and the process continues in Step 3. Otherwise, another path will be generated in Step 5. On the other hand, if $\mathbb{P}_v$ cannot hold a positive flow, then we assume that no more feasible path can be generated. Therefore, node v is removed from the set r.

Step 8

```
if $f_{max} > 0$ then
    • Record the path, leaving times, and flows.
    for $\forall(i,j) \in \mathbb{P}_v$ and $t \in \{0,\ldots,T-1\}$ do
        $\bar{\mathcal{U}}_{i_t,j_{t+\vec{t}_{ij}}} \leftarrow (\bar{\mathcal{U}}_{i_t,j_{t+\vec{t}_{ij}}} - flow(\mathbb{P}_v,t))$.
    end for

    • $s_v \leftarrow s_v - f_{max}$.

    if $s_v = 0$ then
        Remove node $< v >$ from set $< r >$.
        go to Step 3.
    else
        go to Step 5.
    end if
else
    Remove node $< v >$ from set $< r >$.
    go to Step 3.
end if
```

Complexity of ESA. We analyze the worst case complexity of ESA. There are eight steps in this algorithm. ESA spends most time in three steps: network construction, shortest path algorithm, and the greedy algorithm. All other steps are performed in linear time or faster. The time-expanded network generation is done once and can be done in $O((n+m)\cdot T)$. In the worst case, the shortest path algorithm can be done in $O(n^2)$ (Ahuja et al., 1993), the flow assignment can be done in $O(T\cdot m)$ in our greedy algorithm, and the network updates will be performed and it is done in $O(T \cdot m)$. The iteration repeats $|\mathcal{N}_c|$ times. Therefore, the complexity of ESA is $O(|\mathcal{N}_c|\cdot n^2)+O(|\mathcal{N}_c|\cdot m\cdot T)$. Typically, $|\mathcal{N}_c|$ is much smaller than n in practice.

5. Computational Results

5.1. Experiment Setup

We provide numerical experiments to show the performance of CCEP and ESA for solving SNEP. Our base evacuation network is shown in Figure 4. This is the map of the Greater Houston area that includes the City of Galveston, TX. Three regions are classified as R_1, R_2, and R_3 that represent the evacuation priority such that $R_1 \succ R_2 \succ R_3$. Both algorithms, CCEP and ESA, are developed in a C++ environment. CNFP was solved using CPLEX 12.1. All experiments are made on a workstation with 2.83GHz Intel Xeon Quad CPU and 16GB RAM running Windows Server 2008.

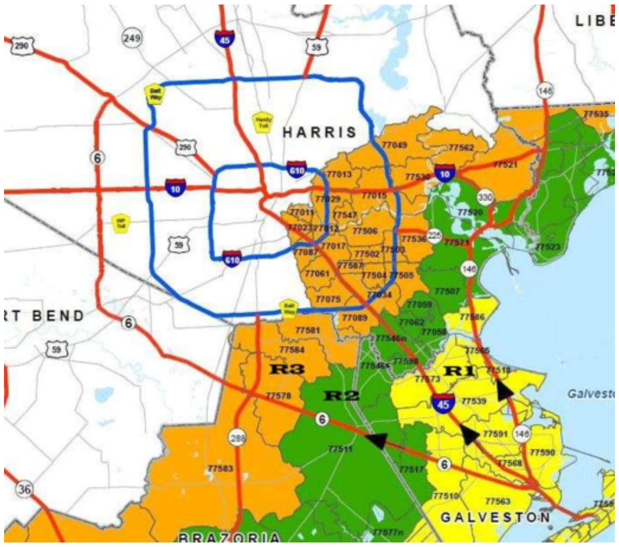


Figure 4: Houston-Galveston Evacuation Map

5.2. Numerical Results

First, we illustrate the procedure of CCEP on a small network (Figure 5) that has 3 centroids ($\mathcal{N}_C = \{1, 2, 3\}$), 5 intersections, and 2 safe nodes ($\mathcal{N}_S = \{9, 10\}$). The numbers of evacuation vehicles (supplies) in centroids 1, 2, and 3 are 350, 185, and 200, respectively. The evacuation time horizon is assumed to be 30 units ($T = 30$). The IP_{ubd} model (Section 2.4) is first solved to generate an upper bound of z_{cnfp}. In this example, IP_{ubd} provides an optimal solution of 100% evacuation for each centroid, *i.e.* $RS^i_{ubd} = 0,\ \forall i \in \mathcal{N}_C$.

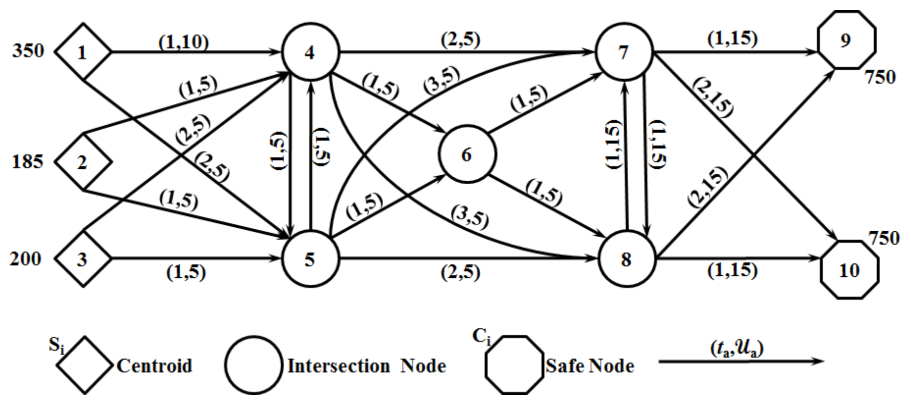


Figure 5: A sample evacuation network

Table 2 shows the results of CCEP for the problem instance in Figure 5. At the first iteration, one path is allowed for each of the three centroids. Then, the CNFP model is solved. Since the remaining supply after solving

Table 2: An illustration of CCEP

iter.	NP†			RS‡			% gap
	C1	C2	C3	C1	C2	C3	
1	1	1	1	225	60	75	49.98
2	2	2	2	100	0	0	13.61
3	3	2	2	0	10	0	1.36
4	3	3	2	0	0	0	0.00

† Number of Paths for each centroid

‡ Remaining Supply in each centroid

CNFP is greater than zero for all the centroids ($RS^i_{cnfp} > 0$), we increase the number of paths by one for all three centroids in the second iteration. At the end of the second iteration, Centroid 1 still has positive remaining supply. Therefore, another path is allowed to this centroid in the third iteration. However, this has caused that Centroid 2 has a positive remaining supply as a result. Note that since we only specify how many paths each centroid can have, CCEP is free to choose paths within the given limitation in order to optimize the total flow. Having zero remaining supply in one iteration does not guarantee the same result in the following iterations. Thus, one more path is added to Centroid 2 from the fourth iteration which finds the optimal flow that is identical to the solution obtained by IP_{ubd}.

In the second set of experiments, a sample network is constructed based on the evacuation network of Greater Houston area (Figure 4). In the sample network, we have 42 nodes which are connected by 111 arcs. Impact areas are categorized into three regions. Region 1 has 6 centroids while regions 2 and 3 have 4 and 3, respectively. We have four safe nodes in the network. Arcs in the network are either unidirectional or bidirectional. Arcs connecting intersection nodes are bidirectional and the remaining arcs are unidirectional. To test the sensitivity of CCEP and ESA on T and the total supply ($\sum_{i=1}^{13} s_i$), we considered four different scenarios in which the total supply are 300, 600, 900, and 1200, respectively. In each scenario, we increased T by 1 from time 24 until it reaches the minimum total number of evacuation paths, which in this example is 13. For each scenario and each T, the total number of completed evacuees (TNCE) and the total number of evacuation paths (TNP) are recorded after running CCEP and ESA.

Table 3: Numerical results for comparison between CCEP and ESA

Time	Scenario 1 (Supply=300)					Scenario 2 (Supply=600)				
	TNCE (%)			TNP		TNCE (%)			TNP	
	IP_{ubd}	CCEP	ESA	CCEP	ESA	IP_{ubd}	CCEP	ESA	CCEP	ESA
24	96.0	96.0	94.3	13	12	73.3	73.3	64.7	25	13
25	99.3	99.3	96.0	13	13	81.7	81.7	76.7	22	18
26	100.0	100.0	99.3	13	16	90.0	90.0	83.7	19	16
27	100.0	100.0	100.0	13	15	95.0	95.0	94.2	17	18
28	100.0	100.0	100.0	13	14	96.7	96.7	95.8	15	16
29	100.0	100.0	100.0	13	13	98.3	98.3	96.7	13	17
30	100.0	100.0	100.0	13	13	100.0	100.0	97.5	13	17
31	100.0	100.0	100.0	13	13	100.0	100.0	98.3	13	17
32	100.0	100.0	100.0	13	13	100.0	100.0	99.2	13	17
33	100.0	100.0	100.0	13	13	100.0	100.0	100.0	13	15
34	100.0	100.0	100.0	13	13	100.0	100.0	100.0	13	14
35	100.0	100.0	100.0	13	13	100.0	100.0	100.0	13	13

Table 3 and Table A.5 in Appendix Appendix A show the results of experiments. In Table A.5, some of the entries are missing values because optimal solutions of CNFP were not found in 12 hours running time and CCEP requires optimal solution of CNFP. As described in *Theorem 2.1*, CCEP produced solutions that reached the theoretical upper bound of TNCE given by $IPubd$ in all cases. This result is important because it confirms *Theorem 3.1* that CCEP finds the minimum number of evacuation paths that maximize the weighted sum of evacuees. Note that the major difference between CCEP and IP_{ubd} is that the latter only generates the flows on arcs, not the paths.

It is easy to see from these tables that as the value T increases, TNCE monotonically increases while TNP monotonically decreases when CCEP is applied. As described in Section 3, CNFP is a module of CCEP. The goal of CCEP is to minimize the number of paths while CNFP is to maximize the total evacuees. This explains the decreasing pattern of TNP as T increases. Furthermore, a higher TNCE is expected when more time is available for evacuation. The results are similar for ESA except that TNP fluctuates with an overall decreasing pattern as T increases. It can also be seen from the results that TNCE of CCEP is always higher than that of ESA in all scenarios tested. However, the major drawback of CCEP is that it cannot handle large networks when the centroids have high supply and the value of T is large.

For a performance comparison between CCEP and ESA, we generated 36 random networks that are variants of Figure 4. Each network may have different network structure such as different number of nodes and arcs. The different network configurations are shown in Table 4. The first column includes problem case identification number. The second, third, and fourth columns have the information related to the number of centroids in each region. Note that Region 1 has the highest priority for evacuation followed by regions 2 and 3. The next four columns show the number of nodes (n) and arcs (m) in the static and dynamic networks. The total number of evacuees (total supply) in each experiment is in the next column. Finally, the last four columns include the comparison between CCEP and ESA on the percent of evacuation and the CPU run time in seconds. Note that the planning time horizon is assumed to be 2 days. Since the length of each time interval for the dynamic network is 15 minutes, the total time interval is 192 units for all 36 experiments, $T = 192$.

The first 9 cases are to evacuate 600 vehicles of total supply and both ESA and CCEP achieved 100% evacuation. However, the CPU run times for CCEP are substantially higher than those of ESA in all cases. The next set of data includes 27 evacuation networks with 5,660 vehicles of total supply. In all cases, CCEP failed to provide a solution within 12 hours while ESA achieved 100% evacuation in less than a second except for Case 10.

Based on the experiments performed in this paper, it is clear that ESA

Table 4: Numerical results for performance comparison between CCEP and ESA

ID	$\mathcal{N}_C$			SNW†		DNW‡		Total	% Evacuation		Time (sec)	
	r_1	r_2	r_3	n	m	n	m	Supply	ESA	CCEP	ESA	CCEP
1	1	2	4	27	78	5185	16945	600	100	100	0.010	45.91
2	2	3	5	31	74	5953	16762	600	100	100	0.010	118.69
3	3	4	6	37	93	7105	20964	600	100	100	0.012	500.19
4	4	2	1	30	77	5761	16749	600	100	100	0.010	45.99
5	5	3	2	34	96	6529	20951	600	100	100	0.011	166.97
6	6	4	3	42	111	8065	24389	600	100	100	0.015	137.98
7	2	1	4	29	82	5569	17704	600	100	100	0.010	68.99
8	3	2	5	34	103	6529	22288	600	100	100	0.013	425.84
9	4	3	6	38	102	7297	22670	600	100	100	0.013	279.86
10	1	2	4	27	78	5185	16945	5660	93.2	–	0.011	–
11	2	3	5	31	74	5953	16762	5660	100	–	0.010	–
12	3	4	6	37	93	7105	20964	5660	100	–	0.013	–
13	4	2	1	30	77	5761	16749	5660	100	–	0.010	–
14	5	3	2	34	96	6529	20951	5660	100	–	0.012	–
15	6	4	3	42	111	8065	24389	5660	100	–	0.015	–
16	2	1	4	29	82	5569	17704	5660	100	–	0.011	–
17	3	2	5	34	103	6529	22288	5660	100	–	0.013	–
18	4	3	6	38	102	7297	22670	5660	100	–	0.014	–
19	4	5	7	46	141	8833	30692	5660	100	–	0.018	–
20	5	6	8	49	130	9409	29164	5660	100	–	0.019	–
21	6	7	9	52	153	9985	34130	5660	100	–	0.022	–
22	7	5	4	44	124	8449	27445	5660	100	–	0.017	–
23	8	6	4	47	122	9025	27636	5660	100	–	0.017	–
24	9	7	6	51	140	9793	31647	5660	100	–	0.020	–
25	5	4	7	43	133	8257	29164	5660	100	–	0.018	–
26	6	5	8	49	136	9409	30310	5660	100	–	0.019	–
27	7	6	9	52	146	9985	32793	5660	100	–	0.021	–
28	7	8	10	53	146	10177	33366	5660	100	–	0.022	–
29	8	9	12	61	167	11713	38141	5660	100	–	0.025	–
30	9	10	15	67	186	12865	42725	5660	100	–	0.029	–
31	10	8	7	54	141	10369	32411	5660	100	–	0.022	–
32	12	9	8	57	141	10945	33175	5660	100	–	0.022	–
33	15	10	9	71	186	13633	42720	5660	100	–	0.029	–
34	8	7	10	55	155	10561	35085	5660	100	–	0.023	–
35	9	8	12	60	159	11521	36613	5660	100	–	0.025	–
36	10	9	15	66	187	12673	42916	5660	100	–	0.030	–

† Static Network ‡ Dynamic Network

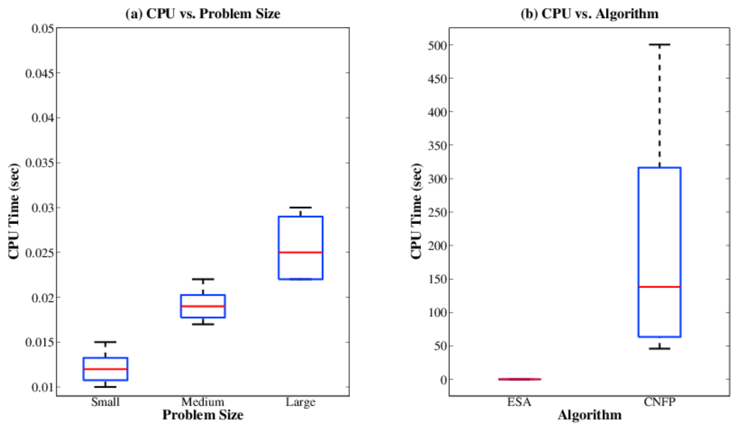


Figure 6: Boxplots for CPU time comparison

is able to find solutions fast even as the network size increases. Figure 6(a) is a boxplot that shows CPU run time distribution for 9 small (6,023 nodes in average), 9 medium (8,517 nodes in average), and 9 larger (10,694 nodes in average) cases. There is an up trend in computation time as the network size increases. The median computation time increased from 0.012 seconds to 0.019 seconds and 0.025 seconds from the small case to medium and large cases, respectively. However, all problems were solved significantly faster than CCEP as can be seen in Figure 6(b).

6. Conclusion

We have developed a capacitated network flow model for short notice evacuation planning. This model is used to determine the starting time for executing the evacuation process, recommended evacuation paths, and their flow for priority based evacuation networks. Since an evacuation occurs over time, the original (static) network is extended to a time-expanded (dynamic) network to capture the dynamics of the evacuation networks over time. There is no known polynomial algorithm for solving SNEP. Therefore, a capacity constrained evacuation planning algorithm is developed to find an optimal solution to our problem. However, due to the exponentially growing size of the dynamic network as the size of the static network increases, CCEP fails to find a solution for large networks. To overcome this computational complexity, we have developed a heuristic algorithm *Evacuation Scheduling*

Algorithm to expedite the solution process. We have shown that the complexity of ESA is $O(|\mathcal{N}_c| \cdot n^2) + O(|\mathcal{N}_c| \cdot m \cdot T)$. Numerical experiments show superior performance gain of ESA over CCEP. In small test networks, ESA ran up to 41,682 times faster than CCEP (0.012 vs 500.19 CPU seconds). To medium to large test network instances, CCEP failed to generate a solution while ESA obtained optimal solutions (except for one instance) within 0.03 seconds on all network instances.

In any emergency situation, there is uncertainty associated with how the situation will progress. Therefore, future work of this research should consider uncertainty of the model input parameters such as road capacity, traversal times, and the number of evacuees coming from each impact area.

Acknowledgment. This research was supported in part by the U.S. Department of Homeland Security.

References

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

Cova, T. and Johnson, J. (2003), 'A network flow model for lane-based evacuation routing', *Transportation Research Part A* **37**(7), 579–604.

Dow, K. and Cutter, S. (1998), 'Crying wolf: Repeat responses to hurricane evacuation orders', *Coastal Management* **26**(4), 237–252.

Drabek, T. E. (1999), 'Understanding disaster warning responses', *The Social Science Journal* **36**(3), 515 – 523.

Edsger, W. (1959), 'A note on two problems in connexion with graphs', *Numerische Mathematik* **1**, 269–271.

Ford, L. and Fulkerson, D. (1958), 'Constructing maximal dynamic flows from static flows', *Operations Research* **6**(3), 491–433.

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

Hamacher, H. and Tjandra, S. (2002), 'Mathematical modelling of evacuation problems–a state of the art', *Pedestrian and Evacuation Dynamics* pp. 227–266.

Herer, Y. and Tzur, M. (2001), 'The dynamic transshipment problem', *Naval Research Logistics* **48**(5), 386–408.

Hoppe, B. E. (1995), Efficient dynamic network flow algorithms, PhD thesis, Cornell University.

Hoppe, B. E. and Tardos, E. (2000), 'The quickest transshipment problem', *Mathematics of Operations Research* pp. 36–62.

Lu, Q., George, B. and Shekhar, S. (2005), 'Capacity constrained routing algorithms for evacuation planning: a summary of results', *Lecture Notes In Computer Science* **3633**, 291.

Perry, R. (1985), *Comprehensive Emergency Management: Evacuating Threatened Populations*, Jai Press, London.

Sherali, H. D., Carter, T. B. and Hobeika, A. G. (1991), 'A location-allocation model and algorithm for evacuation planning under hurricane/flood conditions', *Transportation Research Part B: Methodological* **25**(6), 439 – 452.

Smith, J. M. (1991), 'State-dependent queueing models in emergency evacuation networks', *Transportation Research Part B: Methodological* **25**(6), 373 – 389.

Sorensen, J. H., Carnes, S. A. and Rogers, G. O. (1992), 'An approach for deriving emergency planning zones for chemical munitions emergencies', *Journal of Hazardous Materials* **30**(3), 223 – 242.

Southworth, F. (1991), Regional evacuation modeling: a state-of-the-art review, Technical report, Oak Ridge National Laboratory.

Vogt, B. and Sorensen, J. (1992), Evacuation research: A reassessment, Technical report, Oak Ridge National Lab., Tennessee, United States.

Appendices

Appendix A. Comparison between CCEP and ESA

Table A.5: Numerical results for comparison between CCEP and ESA

Time	Scenario 3 (Supply=900)					Scenario 4 (Supply=1200)				
	TNCE (%)			TNP		TNCE (%)			TNP	
	IP_{ubd}	CCEP	ESA	CCEP	ESA	IP_{ubd}	CCEP	ESA	CCEP	ESA
24	48.9	48.9	47.8	25	13	36.7	36.7	35.0	25	16
25	54.4	54.4	54.4	25	16	40.8	40.8	39.2	25	16
26	60.0	60.0	60.0	25	17	45.0	45.0	43.3	25	16
27	65.6	65.6	65.6	23	17	49.2	49.2	49.2	24	18
28	71.1	71.1	68.9	21	18	53.3	53.3	53.3	24	18
29	76.7	76.7	73.4	24	19	57.5	57.5	57.5	24	18
30	82.2	82.2	79.7	20	16	61.7	61.7	60.4	22	19
31	87.8	–	83.9	–	17	65.8	65.8	64.6	21	20
32	93.3	–	91.6	–	16	70.0	70.0	70.0	20	20
33	98.1	–	94.9	–	16	74.2	–	72.9	–	16
34	99.2	–	95.9	–	17	78.3	–	74.6	–	17
35	100.0	–	96.4	–	16	82.5	–	81.7	–	16
36	100.0	–	97.0	–	16	86.7	–	85.8	–	16
37	100.0	–	97.6	–	16	90.8	–	90.0	–	16
38	100.0	–	98.1	–	15	95.0	–	91.6	–	18
39	100.0	–	98.7	–	15	99.2	–	93.8	–	17
40	100.0	–	99.2	–	15	100.0	–	96.7	–	17
41	100.0	–	99.8	–	15	100.0	–	97.1	–	17
42	100.0	–	100.0	–	14	100.0	–	97.5	–	16
43	100.0	–	100.0	–	13	100.0	–	97.9	–	16
44	100.0	–	100.0	–	13	100.0	–	98.3	–	16
45	100.0	–	100.0	–	13	100.0	–	98.8	–	15
46	100.0	–	100.0	–	13	100.0	–	99.2	–	15
47	100.0	–	100.0	–	13	100.0	–	99.6	–	15
48	100.0	–	100.0	–	13	100.0	–	100.0	–	15