

# Capacity Constrained Routing Algorithms for Evacuation Planning: A Summary of Results\*

Qingsong Lu\*\*, Betsy George, and Shashi Shekhar

Department of Computer Science and Engineering,  
University of Minnesota, 200 Union St SE,  
Minneapolis, MN 55455, USA  
{lqingson, bgeorge, shekhar}@cs.umn.edu  
<http://www.cs.umn.edu/research/shashi-group/>

**Abstract.** Evacuation planning is critical for numerous important applications, e.g. disaster emergency management and homeland defense preparation. Efficient tools are needed to produce evacuation plans that identify routes and schedules to evacuate affected populations to safety in the event of natural disasters or terrorist attacks. The existing linear programming approach uses time-expanded networks to compute the optimal evacuation plan and requires a user-provided upper bound on evacuation time. It suffers from high computational cost and may not scale up to large transportation networks in urban scenarios. In this paper we present a heuristic algorithm, namely Capacity Constrained Route Planner(CCRP), which produces sub-optimal solution for the evacuation planning problem. CCRP models capacity as a time series and uses a capacity constrained routing approach to incorporate route capacity constraints. It addresses the limitations of linear programming approach by using only the original evacuation network and it does not require prior knowledge of evacuation time. Performance evaluation on various network configurations shows that the CCRP algorithm produces high quality solutions, and significantly reduces the computational cost compared to linear programming approach that produces optimal solutions. CCRP is also scalable to the number of evacuees and the size of the network.

**Keywords:** evacuation planning, routing and scheduling, transportation network.

---

\* This work was supported by Army High Performance Computing Research Center contract number DAAD19-01-2-0014 and the Minnesota Department of Transportation contract number 81655. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by the AHPARC and the Minnesota Supercomputing Institute.

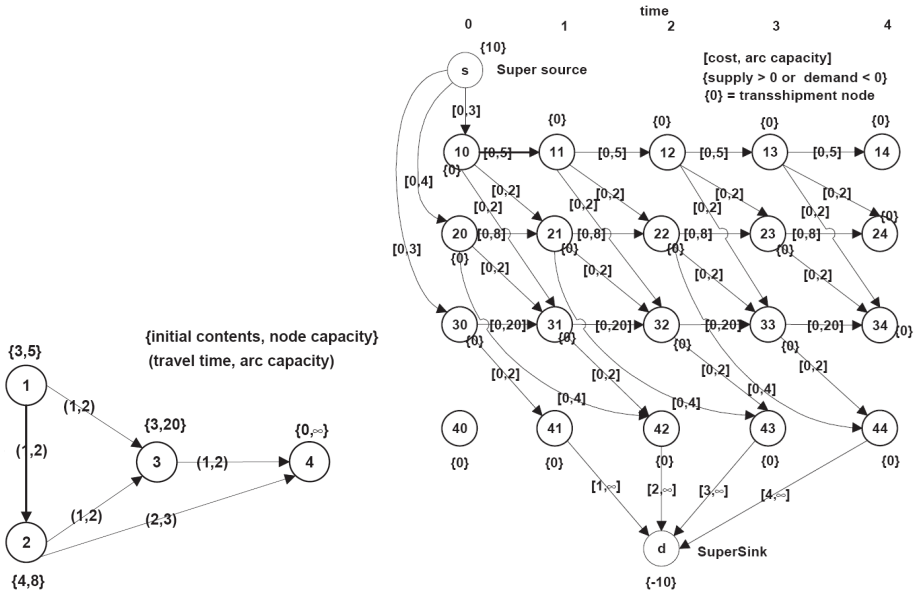
\*\* Corresponding author.

## 1 Introduction

Evacuation planning is critical for numerous important applications, e.g. disaster emergency management and homeland defense preparation. Traditional evacuation warning systems simply convey the threat descriptions and the need for evacuation to the affected population via mass media communication. Such systems do not consider capacity constraints of the transportation network and thus may lead to unanticipated effects on the evacuation process. For example, when Hurricane Andrew was approaching Florida in 1992, the lack of effective planning caused tremendous traffic congestions, general confusion and chaos [1]. Therefore, efficient tools are needed to produce evacuation plans that identify routes and schedules to evacuate affected populations to safety in the event of natural disasters or terrorist attacks [12,14,7,8].

The current methods of evacuation planning can be divided into two categories, namely traffic assignment-simulation approach and route-schedule planning approach. The traffic assignment-simulation approach uses traffic simulation tools, such as DYNASMART [27] and DynaMIT [5], to conduct stochastic simulation of traffic movements based on origin-destination traffic demands and uses queuing methods to account for road capacity constraints. However, it may take a long time to complete the simulation process for a large transportation network. The route-schedule planning approaches use network flow and routing algorithms to produce origin-destination routes and schedules of evacuees on each route. Many research works have been done to model the evacuation problem as a network flow problem [15,4] and to find the optimal solution using linear programming methods. Hamacher and Tjandra [17] gave an extensive literature review of the models and algorithms used in these linear programming methods. Based on the triple-optimization results by Jarvis and Ratliff [20], linear programming method for evacuation route planning works as follows. First, it models the evacuation network into a network graph, as shown by network  $G$  in Figure 1, and it requires the user to provide an estimated upper bound  $T$  of the evacuation egress time. Second, it converts evacuation network  $G$  to a time-expanded network, as shown by  $G_T$  in Figure 2, by duplicating the original evacuation network  $G$  for each discrete time unit  $t = 0, 1, \dots, T$ . Then, it defines the evacuation problem as a minimum cost network flow problem [15,4] on the time-expanded network  $G_T$ . Finally, it feeds the expanded network  $G_T$  to minimum cost network flow solvers, such as NETFLO [21], to find the optimal solution. For example, EVACNET [9,16,22,23] is a computer program based on this approach which computes egress time for building evacuations. It uses NETFLO code to obtain the optimal solution. Hoppe and Tardos [18,19] gave a polynomial time bounded algorithm by using ellipsoid method of linear programming to find the optimal solution for the minimum cost flow problem. Theoretically, ellipsoid method has a polynomial bounded running time. However, it performs poorly in practice and has little value for real application [6].

Linear programming approach can produce optimal solutions for evacuation planning. It is useful for evacuation scenarios with moderate size networks, such as building evacuation. However, this approach has the following limita-



**Fig. 1.** Evacuation Network  $G$ , **Fig. 2.** Time-expanded Network  $G_T$ , with  $T=4$ , (source: [17])

tions. First, it significantly increases the problem size because it requires time-expanded network  $G_T$  to produce a solution. As can be seen in Figures 1 and 2, if the original evacuation network  $G$  has  $n$  nodes and the time upper bound is  $T$ , the time-expanded network  $G_T$  will have at least  $(T + 1)n$  nodes. This approach may not be able to scale up to large size transportation networks in urban evacuation scenarios due to high computational run-time caused by the tremendously increased size of the time-expanded network. Second, linear programming approach requires the user to provide an upper bound  $T$  of the evacuation time in order to generate the time-expanded network. It is almost impossible to precisely estimate the evacuation time for an urban scenario where the number of evacuees is large and the transportation network is complex. An under-estimated time bound  $T$  will result in failure of finding a solution. In this case, the user will have to increase the value of  $T$  and re-run the algorithm until a solution can be reached. On the other hand, an over-estimated  $T$  will result in an over-expanded network  $G_T$  and hence lead to unnecessary storage and run-time.

Heuristic routing and scheduling algorithms can be used to find sub-optimal evacuation plan with reduced computational cost. It is useful for evacuation scenarios with large size networks and scenarios that do not require an optimal plan, but need to produce an efficient plan within a limited amount of time. However, old heuristic approaches only compute the shortest distance route from a source to the nearest destination without considering route capacity constraints. It cannot produce efficient plans when the number of evacuees is large and the

evacuation network is complex. New heuristic approaches are needed to account for capacity constraints of the evacuation network. Lu, Huang and Shekhar [26] proposed prototypes of two heuristic capacity constrained routing algorithms, namely SRCCP and MRCCP, and tested its performance using small size building networks. SRCCP assigns only one route to each source node. It has very fast run-time but the solution quality is very poor and hence has little value for real application. MRCCP assigns multiple routes to each source node and produces high quality solution with much less run-time compared to that of linear programming approach. However, its scalability to large size networks is unsatisfactory because it has a computational cost of  $O(p \cdot n^2 \log n)$  (where  $n$  is the number of nodes and  $p$  is the number of evacuees). In this paper, we present an improved algorithm called Capacity Constrained Route Planner (CCRP). CCRP can reduce the run-time to  $O(p \cdot n \log n)$  by conducting only one shortest path search in each iteration instead of the multiple searches used in MRCCP. We also present the analysis of its algebraic cost model and provide the results of performance evaluation using large size transportation networks.

In the CCRP algorithm, we model capacity as a time series because available capacity of each node and edge may vary during the evacuation. We use a generalized shortest path search algorithm to account for route capacity constraints. This algorithm can divide evacuees from each source into multiple groups and assign a route and time schedule to each group of evacuees based on an order that is prioritized by each group's destination arrival time. It then reserves route capacities for each group subject to the route capacity constraints. The quickest route available for one group is re-calculated in each iteration based on the available capacity of the network. Performance evaluation on various network configurations shows that the CCRP algorithm produces high quality solutions, and significantly reduces the computational cost compared to linear programming approach. CCRP is also scalable to the number of evacuees and the size of the network. A case study using a nuclear power plant evacuation scenario shows that this algorithm can be used to improve existing evacuation plans by reducing evacuation time.

We also explored the possibility of formulation of a new optimal algorithm using A\* search [28,29]. It addresses the limitations of linear programming approach by using only the original evacuation network to find the optimal solution and it does not require the user to provide an upper bound of the evacuation time. Details of the A\* search formulation and the proof of monotonicity and admissibility of this A\* search algorithm are available in [25]. It is not included in this paper due to space constraints.

**Outline:** The rest of the paper is organized as follows. In Section 2, the problem formulation is provided and related concepts are illustrated by an example evacuation network. Section 3 describes the Capacity Constrained Route Planner (CCRP) algorithm and the algebraic cost model. In Section 4, we present the experimental design and performance evaluation. We summarize our work and discuss future directions in Section 5.

## 2 Problem Formulation

We formulate the evacuation planning problem as follows:

**Given:** A transportation network with non-negative integer capacity constraints on nodes and edges, non-negative integer travel time on edges, the total number of evacuees and their initial locations, and locations of evacuation destinations.

**Output:** An evacuation plan consisting of a set of origin-destination routes and a scheduling of evacuees on each route. The scheduling of evacuees on each route should observe the capacity constraints of the nodes and edges on this route.

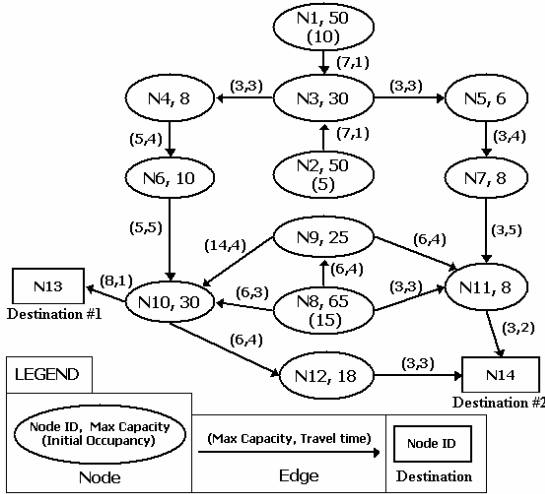
**Objective:** (1) Minimize the evacuation egress time, which is the time elapsed from the start of the evacuation until the last evacuee reaches the evacuation destination. (2) Minimize the computational cost of producing the evacuation plan.

**Constraint:** (1) Edge travel time preserves FIFO (First-In First-Out) property. (2) Edge travel time reflects delays at intersections. (3) Limited amount of computer memory.

We illustrate the problem formulation and a solution with an example evacuation network, as shown in Figure 3. In this evacuation network, each node is shown by an ellipsis. Each node has two attributes: maximum node capacity and initial node occupancy. For example, at node N1, the maximum capacity is 50, which means this node can hold at most 50 evacuees at each time point, while the initial occupancy is 10, which means there are initially 10 evacuees at this node. In Figure 3, each edge, shown as an arrow, represents a link between two nodes. Each edge also has two attributes: maximum edge capacity and travel time. For example, at edge N4-N6, the maximum edge capacity is 5, which means at each time point, at most 5 evacuees can start to travel from node N4 to N6 through this link. The travel time of this edge is 4, which means it takes 4 time units to travel from node N4 to N6. This approach of modelling a evacuation scenario to a capacitated node-edge graph is similar to those presented in Hamacher [17], Kisko [23] and Chalmet [9].

As shown in Figure 3, suppose we initially have 10 evacuees at node N1, 5 at node N2, and 15 at node N8. The task is to compute an evacuation plan that evacuates the 30 evacuees to the two destinations (node N13 and N14) using the least amount of time.

*Example 1 (An Evacuation Plan).* Table 1 shows an example evacuation plan for the evacuation network in Figure 3. In this table, each row shows one group of evacuees moving together during the evacuation with a group ID, source node, number of evacuees in this group, the evacuation route with time schedule, and the destination time. The route is shown by a series of node number and the time schedule is shown by a start time associated with each node on the route. Take source node N8 for example; initially there are 15 evacuees at N8. They are divided into 3 groups: Group A with 6 people, Group B with 6 people and



**Fig. 3.** Node-Edge Graph Model of Example Evacuation Network

Group C with 3 people. Group A starts from node N8 at time 0 to node N10, then starts from node N10 at time 3 to node N13, and reaches destination N13 at time 4. Group B follows the same route of group A, but has a different schedule due to capacity constraints of this route. This group starts from N8 at time 1 to N10, then starts from N10 at time 4 to N13, and reaches destination N13 at time 5. Group C takes a different route. It starts from N8 at time 0 to N11, then starts from N11 at time 3 to N14, and reaches destination N14 at time 5. The procedure is similar for other groups of evacuees from source node N1 and N2. The whole evacuation egress time is 16 time units since the last groups of people (Group H and I) reach destination at time 16. This evacuation plan is an optimal plan for the evacuation scenario shown in Figure 3.

In our problem formulation, we allow time dependent node capacity and edge capacity, but we assume that edge capacity does not depend on the actual flow amount in the edge. We also allow time dependent edge travel time, but we require that the network preserve the FIFO (First-In First-Out) property.

Alternate problem formulations of the evacuation problem are available by changing the objective of the problem. The main objective of our problem formulation is to minimize the evacuation egress time. Two alternate objectives are: (1) Maximize the number of evacuees that reach destination for each time unit; (2) Minimize the average evacuation time for all evacuees. Jarvis and Ratliff presented and proved the *triple optimization theorem* [20], which illustrated the properties of the solutions that optimize the above objectives of the evacuation problem. A review of linear programming approaches to solve these problem formulations was given by Hamacher and Tjandra [17].

**Table 1.** Example Evacuation Plan

Group of Evacuees			Route with Schedule	Dest. Time
ID	Source	Number		
A	N8	6	N8(T0)-N10(T3)-N13	4
B	N8	6	N8(T1)-N10(T4)-N13	5
C	N8	3	N8(T0)-N11(T3)-N14	5
D	N1	3	N1(T0)-N3(T1)-N4(T4)-N6(T8)-N10(T13)-N13	14
E	N1	3	N1(T0)-N3(T2)-N4(T5)-N6(T9)-N10(T14)-N13	15
F	N1	1	N1(T0)-N3(T1)-N5(T4)-N7(T8)-N11(T13)-N14	15
G	N2	2	N2(T0)-N3(T1)-N5(T4)-N7(T8)-N11(T13)-N14	15
H	N2	3	N2(T0)-N3(T3)-N4(T6)-N6(T10)-N10(T15)-N13	16
I	N1	3	N1(T1)-N3(T2)-N5(T5)-N7(T9)-N11(T14)-N14	16

### 3 Proposed Approach

Linear programming approach can produce optimal solutions for evacuation planning. It is useful for evacuation scenarios with moderate size networks, such as building evacuation. However, it may not be able to scale up to large size transportation networks in urban evacuation scenarios due to high computational cost caused by the tremendously increased size of the time-expanded network. Heuristic routing and scheduling algorithms can be used to find sub-optimal evacuation plan with reduced computational cost. It is useful for evacuation scenarios with large size networks and scenarios that do not require an optimal plan, but need to produce an efficient plan within a limited amount of time.

In this section, we present a heuristic algorithm, namely Capacity Constrained Route Planner (CCRP), that produces sub-optimal solutions for evacuation planning. We model edge capacity and node capacity as a time series instead of fixed numbers. A time series represents the available capacity at each time instant for a given edge or node. We propose a heuristic approach based on an extension of shortest path algorithms [13,11] to account for capacity constraints of the network.

#### 3.1 Capacity Constrained Route Planner (CCRP)

The Capacity Constrained Route Planner (CCRP) uses an iterative approach. In each iteration, the algorithm first searches for route  $R$  with the earliest destination arrival time from any source node to any destination node, taking previous reservations and possible waiting time into consideration. Next, it computes the actual amount of evacuees that will travel through route  $R$ . This amount is affected by the available capacity of route  $R$  and the remaining number of evacuees. Then, it reserves the node and edge capacity on route  $R$  for those evacuees. The algorithm continues to iterate until all evacuees reach destination. The detailed pseudo-code and algorithm description are shown in Algorithm 1.

The CCRP algorithm keeps iterating as long as there are still evacuees left at any source node (line 1). Each iteration starts with finding the route  $R$  with the

**Algorithm 1.** Capacity Constrained Route Planner (CCRP)**Input:**

- 1)  $G(N, E)$ : a graph  $G$  with a set of nodes  $N$  and a set of edges  $E$ ;  
 Each node  $n \in N$  has two properties:  
      $Maximum\_Node\_Capacity(n)$  : non-negative integer  
      $Initial\_Node\_Occupancy(n)$  : non-negative integer  
 Each edge  $e \in E$  has two properties:  
      $Maximum\_Edge\_Capacity(e)$  : non-negative integer  
      $Travel\_time(e)$  : non-negative integer
- 2)  $S$ : set of source nodes,  $S \subseteq N$ ;
- 3)  $D$ : set of destination nodes,  $D \subseteq N$ ;

**Output:** Evacuation plan: Routes with schedules of evacuees on each route**Method:**Pre-process network: add super source node  $s_0$  to network,link  $s_0$  to each source nodes with an edge which

$$Maximum\_Edge\_Capacity() = \infty \text{ and } Travel\_time() = 0; \quad (0)$$

while any source node  $s \in S$  has evacuee do { (1)    find route  $R < n_0, n_1, \dots, n_k >$  with time schedule  $< t_0, t_1, \dots, t_{k-1} >$         using one generalized shortest path search from super source  $s_0$   
         to all destinations, (where  $s \in S, d \in D, n_0 = s, n_k = d$ )        such that  $R$  has the earliest destination arrival time among                routes between all  $(s, d)$  pairs,        and  $Available\_Edge\_Capacity(e_{n_i n_{i+1}}, t_i) > 0, \quad \forall i \in \{0, 1, \dots, k-1\}$ ,        and  $Available\_Node\_Capacity(n_{i+1}, t_i + Travel\_time(e_{n_i n_{i+1}})) > 0,$ 

$$\forall i \in \{0, 1, \dots, k-1\}; \quad (2)$$

 $flow = \min(\text{number of evacuees still at source node } s,$          $Available\_Edge\_Capacity(e_{n_i n_{i+1}}, t_i), \quad \forall i \in \{0, 1, \dots, k-1\},$          $Available\_Node\_Capacity(n_{i+1}, t_i + Travel\_time(e_{n_i n_{i+1}})),$ 

$$\forall i \in \{0, 1, \dots, k-1\}; \quad (3)$$

    ); (3)    for  $i = 0$  to  $k - 1$  do { (4)         $Available\_Edge\_Capacity(e_{n_i n_{i+1}}, t_i)$  reduced by  $flow$ ; (5)         $Available\_Node\_Capacity(n_{i+1}, t_i + Travel\_time(e_{n_i n_{i+1}}))$  reduced by  $flow$ ; (6)    } (7)} (8)Output evacuation plan; (9)

earliest destination arrival time from any sources node to any destination node based on the current available capacities (line 2). This is done by generalizing Dijkstra's shortest path algorithm [13,11] to work with the time series node and edge capacities and edge travel time. Route  $R$  is the route that starts from a source node and gets to a destination node in the least amount of time and available capacity of the route allows at least one person to travel through route  $R$  to a destination node.

Compared with the earlier MRCCP algorithm [26], major improvements in CCRP lie in line 0 and line 2. In MRCCP, finding route  $R$  (line 2) is done by

running generalized shortest path searches from each source node. Each search is terminated when any destination node is reached. In CCRP, this step is improved by adding a super source node  $s_0$  to the network and connecting  $s_0$  to all source nodes (line 0). This allows us to complete the search for route  $R$  by using only one single generalized shortest path search, which takes the super source  $s_0$  as the start node. This search terminates when any destination node is reached. Since the super source  $s_0$  is connected to each source nodes by an edge with infinite capacity and zero travel time, it can be easily proved that the shortest route found by this search is the route  $R$  we need in line 2. This improvement significantly reduces the computational cost of the algorithm by one degree of magnitude compared with MRCCP. We give a detailed analysis of the cost model of CCRP algorithm in the next section.

### 3.2 Algebraic Cost Model of CCRP

We now provide the algebraic cost model for the computational cost of the proposed CCRP algorithm. We assume that  $n$  is the number of nodes in the evacuation network,  $m$  is the number of edges, and  $p$  is the number of evacuees.

The CCRP algorithm is an iterative approach. In each iteration, the route for one group of people is chosen and the capacities along the route are reserved. The total number of iterations equals the number of groups generated. In the worst case, each individual evacuee forms one group. Therefore, the upper bound of the number of groups is  $p$ , i.e. the number of iterations is  $O(p)$ . In each iteration, the computation of the route  $R$  with earliest destination arrival time is done by running one generalized Dijkstra's shortest path search. The worst case computational complexity of Dijkstra's algorithm is  $O(n^2)$  for dense graphs [11]. Various implementations of Dijkstra's algorithm have been developed and evaluated extensively [4,10,32]. Many of these implementations can reduce the computational cost by taking advantage of the sparsity of the graph. Transportation road networks are very sparse graphs with a typical edge/node ratio around 3. In CCRP, we implement Dijkstra's algorithm using heap structures, which runs in  $O(m + n \log n)$  time [4,10]. For sparse graphs,  $n \log n$  is the dominant term. The generalization of Dijkstra's algorithm to account for capacity constraints affects only how the shortest distance to each node is defined. It does not affect the computational complexity of the algorithm. Therefore, we can complete the search for route  $R$  with  $O(n \log n)$  run-time. The reservation step is done by updating the node and edge capacities along route  $R$ , which has a cost of  $O(n)$ . Therefore, each iteration of the CCRP algorithm is done in  $O(n \log n)$  time. As we have seen, it takes  $O(p)$  iterations to complete the algorithm. The cost model of the CCRP algorithm is  $O(p \cdot n \log n)$ . CCRP is an improved algorithm based on the same heuristic method of MRCCP [26] which has a run-time of  $O(p \cdot n^2 \log n)$ . CCRP reduces the computational cost of MRCCP by one degree of magnitude.

The computational cost of linear programming approach depends on the method used to solve the minimum cost flow problem. Hoppe and Tardos [18] showed that this problem can be solved using ellipsoid method which is theoretically polynomial time bounded. However, the computational complexity of

**Table 2.** Comparison of Computational Costs ( $n$ : number of nodes,  $p$ : number of evacuees,  $T$ : user-provided upper-bound on evacuation time)

Algorithm	Computational Cost	Solution Quality
CCRP	$O(p \cdot n \log n)$	Sub-optimal
MRCCP	$O(p \cdot n^2 \log n)$	Sub-optimal
Linear Programming Approach	at least $O((T \cdot n)^6)$	Optimal

ellipsoid method is at least  $O(N^6)$ [6](where  $N$  is the number of nodes in the network). Since linear programming approach requires a time-expanded network,  $N$  equals to  $(T + 1)n$  (where  $n$  is the number of nodes in the original evacuation network,  $T$  is the user-provided evacuation time upper bound).

Table 2 provides a comparison of CCRP, MRCCP, and the linear programming approach. As can be seen, linear programming approach produces optimal solutions but suffers from high computational cost. Both CCRP and MRCCP reduce the computation cost by producing sub-optimal solution, while CCRP gives better computational cost than MRCCP.

**Lemma 1:** CCRP is strictly faster than MRCCP.

The computational costs of CCRP and MRCCP are  $O(p \cdot n \log n)$  and  $O(p \cdot n^2 \log n)$  respectively, as shown in Table 2.

## 4 Experiment Design and Performance Evaluation

Performance evaluation of the CCRP algorithm was done by conducting experiments using various evacuation network configurations. In this section, we present the experiment design and an analysis of the experiment results.

### 4.1 Experiment Design

Figure 4 describes the experiment design to evaluate the performance of the CCRP algorithm. The purpose is to compare the algorithm run-time and solution quality of the proposed CCRP algorithms with that of MRCCP [26] and NETFLO [21] which is a popular linear programming package used to solve minimum cost flow problems.

First, we used NETGEN [24] to generate evacuation networks with evacuees. NETGEN is a program that generates transportation networks with capacity constraints and initial supplies based on input parameters. In our experiments, the following three were selected as independent parameters to test their impacts on the the performance of the algorithms: number of evacuees initially in the network, number of source nodes, and network size represented by number of nodes. Number of edges is treated as a dependent parameter as we set the number of edges to be equal to 3 times the number of nodes because 3 is the typical edge/node ratio for real transportation road networks. Next, the same

evacuation network generated by NETGEN was fed to the CCRP and MRCCP algorithms. Before feeding the network to NETFLO, we used a network transformation tool to transform the evacuation network into a time-expanded network, which is required by minimum cost flow solvers as NETFLO to solve evacuation problems [17,9]. This process requires an input parameter  $T$  which is the estimated upper-bound on evacuation egress time. If the evacuation cannot be completed by time  $T$ , NETFLO will return no solution. In this case, we must increase  $T$  to create a new time-expanded network and try to run NETFLO again until a solution can be reached. Finally, after CCRP, MRCCP and NETFLO produced a solution for each test case, the evacuation egress time, which represents the solution quality, and the algorithm run-time were collected and analyzed in the data analysis module.

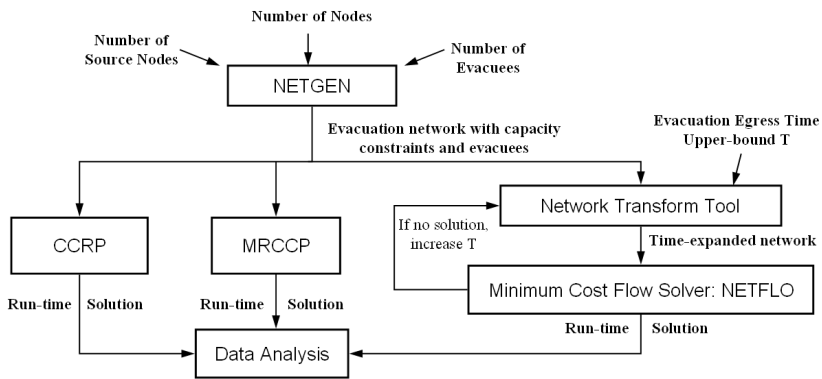


Fig. 4. Experiment Design

The experiments were conducted on a workstation with Intel Pentium IV 2GHz CPU, 2GB RAM and Debian Linux operating system.

## 4.2 Experiment Results and Analysis

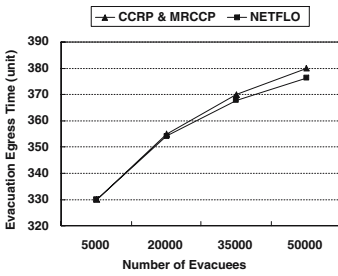
We want to answer three questions: (1) How does the number of evacuees affect the performance of the algorithms? (2) How does the number of source nodes affect the performance of the algorithms? (3) Are the algorithms scalable to the size of the network, particularly will they handle large size transportation networks as in urban evacuation scenarios?

*Experiment 1: How does the number of evacuees affect the performance of the algorithms?*

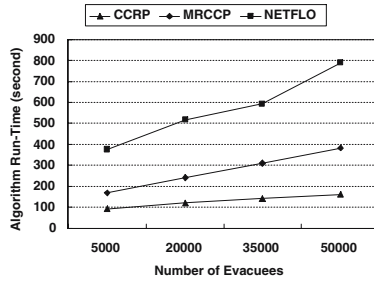
The purpose of the first experiment is to evaluate how the number of evacuees affects the performance of the algorithms. We fixed the number of nodes and the number of source nodes of the network, and varied the number of evacuees

to observe the quality of the solution and the run-time of CCRP, MRCCP and NETFLO algorithms.

The experiment was done with four test groups. Each group had a fixed network size of 5000 nodes and fixed number of source nodes at 1000, 2000, 3000, and 4000 respectively. We varied the number of evacuees from 5000 to 50000. Here we present the experiment results of the test group with number of source nodes fixed at 2000. We omit the results from the other three groups since this group shows a typical result of all test groups. Figure 5 shows the solution quality represented by evacuation egress time and Figure 6 shows the run-times of the three algorithms.



**Fig. 5.** Quality of Solution With Respect to Number of Evacuees



**Fig. 6.** Run-time With Respect to Number of Evacuees

Since CCRP and MRCCP use the same heuristic method to find solution, it is expected that CCRP and MRCCP produced solutions with the same evacuation egress time for each test case. As seen in Figure 5, CCRP and MRCCP produced very high quality solution compared with the optimal solution produced by NETFLO. The solution quality of CCRP and MRCCP drops slightly as the the number of evacuees grows. In Figure 6, we can see that, in each case, the run-time of CCRP remains half that of MRCCP and less than 1/3 that of NETFLO. In addition, the CCRP run-time is scalable to the number of evacuees while the run-time of NETFLO grows much faster.

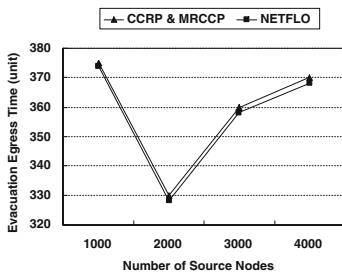
This experiment shows: (1) CCRP produces high quality solutions with much less run-time than that of NETFLO. (2) The run-time of CCRP is scalable to the number of evacuees.

*Experiment 2: How does the number of source nodes affect the performance of the algorithms?*

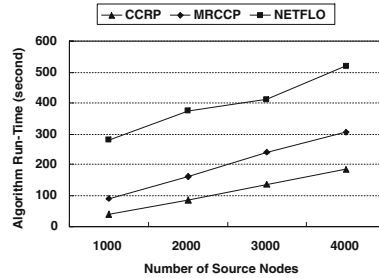
In the second experiment, we evaluate how the number of source nodes affects the performance of the algorithms. We fixed the number of nodes and the number of evacuees in the network, and varied the number of source nodes to observe the quality of the solution and the run-time. In this experiment, by varying the number of source nodes, we actually create different evacuee distributions in the

network. A higher number of source nodes means that the evacuees are more scattered in the network.

Again, the experiment was done with four test groups. Each group had a fixed network size of 5000 nodes and fixed number of evacuees at 5000, 20000, 35000, and 50000 respectively. We varied the number of source nodes from 1000 to 4000. Here we present the experiment results of the test group with number of evacuees fixed at 5000. It shows a typical result of all test groups. Figure 7 shows the solution quality represented by evacuation egress time and Figure 8 shows the run-times of the three algorithms.



**Fig. 7.** Quality of Solution With Respect to Number of Source Nodes



**Fig. 8.** Run-time With Respect to Number of Source Nodes

As seen in Figure 7, in each test case, CCRP and MRCCP produced high quality solution (within 5 percent longer evacuation time) and the number of source nodes has little effect on the solution quality. It is also noted that the evacuation time is non-monotonic with respect to the number of source nodes and we plan to explore the potential reasons in future works.

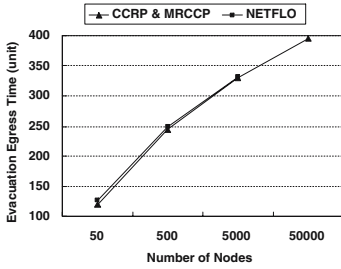
Figure 8 shows that the run-time of all three algorithms are scalable to the number of source nodes. However, the run-time of CCRP remains less than half that of NETFLO.

This experiment shows: (1) The solution quality of CCRP is not affected by the number of source nodes. (2) The run-time of CCRP is scalable to the number of source nodes.

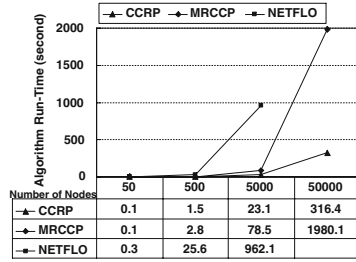
### *Experiment 3: Are the algorithms scalable to the size of the network?*

In the third experiment, we evaluate how the network size affects the performance of the algorithms. We fixed the number of evacuees and the number of source nodes in the network, and varied the network size to observe the quality of solution and the run-time of the algorithms.

The experiment was done with a fixed number of evacuees at 5000 and the number of source nodes at 10. We varied the number of nodes from 50 to 50000. Figure 9 shows the solution quality represented by evacuation egress time and Figure 10 shows the run-times.



**Fig. 9.** Quality of Solution With Respect to Network Size



**Fig. 10.** Run-time With Respect to Network Size

Note: x-axis(number of nodes) in Figure 9 and 10 is on a logarithmic scale rather than linear. Run-time of CCRP and MRCCP in Figure 10 grow in small polynomial.

There is no data point for NETFLO at network size of 50000 nodes. We were unable to run NETFLO for this setup because the size of the time-expanded network became too large (more than 20 million nodes and 80 million edges)that NETFLO could not produce solution.

As seen in Figure 9, in each of the first three test case, CCRP and MRCCP produced high quality solution (within 5 percent longer evacuation time) and the solution quality becomes closer to optimal solution as the network size increases. Figure 10 is shown with a data table of each run-time. The x-axis(number of nodes) of Figure 10 is on a logarithmic scale rather than linear and the run-time of CCRP and MRCCP grow in small polynomial. It can be seen that the run-time of CCRP is scalable to the network size while the NETFLO run-time grows exponentially.

This experiment shows: (1) Given a fixed number of evacuees and source nodes, the solution quality of CCRP increases as the network size increases. (2) The run-time of CCRP is scalable to the size of the network.

We also conducted experiments using a real evacuation scenario. The Monticello nuclear power plant is about 40 miles to the northwest of the Twin Cities. Evacuation plans need to be in place in case of accidents or terrorist attacks. The evacuation zone is a 10-mile radius around the nuclear power plant as defined by Minnesota Homeland Security and Emergency Management [3].

The experiment was done using the road network around the evacuation zone provided by the Minnesota Department of Transportation [2], and the Census 2000 population data for each affected city. The total number of evacuees is about 42,000. The old hand-crafted evacuation plan has an evacuation egress time of 268 minutes. CCRP algorithm produced a much better plan with evacuation time of only 162 minutes. This experiment shows that our algorithm is effective in real evacuation scenarios to reduce evacuation time and improve existing plans.

Our approach was presented in the UCGIS Congressional Breakfast Program on homeland security[30], and the Minnesota Homeland Security and Emergency Management newsletter[31]. It was also selected by the Minnesota Department

of Transportation to be used in the evacuation planning project for the Twin Cities Metro Area, which involves a road network of about 250,000 nodes and a population of over 2 million people.

## 5 Conclusions and Discussions

In this paper, we proposed a new capacity constrained routing algorithm for evacuation planning problem. Existing linear programming approach uses time-expanded network and requires user provided upper bound on evacuation time. To address these limitations, we presented a heuristic algorithm, namely Capacity Constrained Route Planner (CCRP), which produces sub-optimal solution for evacuation planning problem without using time-expanded networks. We provided the algebraic cost model and the performance evaluations using various network configurations. Experiments show that CCRP algorithm produces high quality solution and significantly reduces the computational cost compared to linear programming approach which produces optimal solution. It is also shown that the CCRP algorithm is scalable to the number of evacuees and the size of the transportation network. A case study using real evacuation scenario shows that CCRP algorithm can be used to improve existing evacuation plans by reducing total evacuation time.

The limitation of CCRP algorithm remains the follows. First, we assume that maximum capacity of an edge does not depend on traffic flow amount on the edge. We understand that it is a challenging task to accurately model the capacity of each road segment in a real evacuation scenario as the actual traffic flow rate may depend on vehicle speed as well as road occupancy. Second, the generalized shortest path algorithm we used in CCRP requires that the edge travel time reflects traffic delays at intersections. For future work, we plan to incorporate existing research results, such as Ziliaskopoulos and Mahmassani [33], to better address this problem.

To address the sub-optimality issue of the CCRP algorithm, we also explored the possibility of formulating the evacuation problem as a search problem using A\* algorithm. Our A\* search formulation addresses the limitations of linear programming approach by only using the original evacuation network to find optimal solution. Thus, it does not require prior knowledge of evacuation time. We proved that the heuristic function used in our A\* formulation is monotone and admissible thus guaranteeing the optimality of the solution. Details of the A\* search formulation can be found in [25]. It is not included in this paper due to space constraints.

## Acknowledgment

We are particularly grateful to members of the Spatial Database Research Group at the University of Minnesota for their helpful comments and valuable discussions. We would also like to express our thanks to Kim Koffolt for improving the readability of this paper.

This work is supported by the Army High Performance Computing Research Center (AHPCRC) under the auspices of the Department of the Army, Army Research Laboratory under contract number DAAD19-01-2-0014 and the Minnesota Department of Transportation under contract number 81655. The content does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. AHPCRC and the Minnesota Supercomputer Institute provided access to computing facilities.

## References

1. *Hurricane Evacuation web page*. <http://i49south.com/hurricane.htm>, 2002.
2. *Minnesota basemap web site*. <http://www.dot.state.mn.us/tda/basemap/>, Minnesota Department of Transportation, 2004.
3. *Monticello evacuation planning web site*. <http://www.hsem.state.mn.us/>, Minnesota Homeland Security and Emergency Management, 2004.
4. R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
5. M. Ben-Akiva et al. *Development of a Deployable Real-Time Dynamic Traffic Assignment System: DynaMIT and DynaMIT-P User's Guide*. Intelligent Transportation Systems Program, Massachusetts Institute of Technology, 2002.
6. D. Bertsimas and J.N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.
7. S. Brown. Building America's Anti-Terror Machine: How Infotech Can Combat Homeland Insecurity. *Fortune*, pages 99–104, July 2002.
8. The Volpe National Transportation Systems Center. Improving Regional Transportation Planning for Catastrophic Events(FHWA). *Volpe Center Highlights*, pages 1–3, July/August 2002.
9. L. Chalmet, R. Francis, and P. Saunders. Network Model for Building Evacuation. *Management Science*, 28:86–105, 1982.
10. B.V. Cherkassky, A.V. Goldberg, and T. Radzik. Shortest Paths Algorithms: Theory and Experimental Evaluation. *Mathematical Programming*, 73:129–174, 1996.
11. T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001.
12. The Homeland Security Council. Planning Scenarios, Executive Summaries, Created for Use in National, Federal, State, and Local Homeland Security Preparedness Activities. July 2004.
13. E.W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1:269–271, 1959.
14. ESRI. GIS for Homeland Security, An ESRI white paper, November 2001.
15. L.R. Ford and D.R. Fulkerson. *Flows in Network*. Princeton University Press, 1962.
16. R. Francis and L. Chalmet. A Negative Exponential Solution To An Evacuation Problem. *Research Report No.84-86, National Bureau of Standards, Center for Fire Research*, October 1984.
17. H.W. Hamacher and S.A. Tjandra. Mathematical Modeling of Evacuation Problems: A state of the art. *Pedestrian and Evacuation Dynamics*, pages 227–266, 2002.
18. B. Hoppe and E. Tardos. Polynomial Time Algorithms For Some Evacuation Problems. *Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 433–441, 1994.
19. B. Hoppe and E. Tardos. The Quickest Transshipment Problem. *Proceedings of the 6th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 512–521, January 1995.

20. J.J. Jarvis and H.D. Ratliff. Some Equivalent Objectives for Dynamic Network Flow Problems. *Management Science*, 28:106–108, 1982.
21. J.L. Kennington and R.V. Helgason. *Algorithm for Network Programming*. Wiley and Sons, 1980.
22. T. Kisko and R. Francis. Evacnet+: A Computer Program to Determine Optimal Building Evacuation Plans. *Fire Safety Journal*, 9:211–222, 1985.
23. T. Kisko, R. Francis, and C. Nobel. *EVACNET4 User's Guide*. University of Florida, 1998.
24. D. Klingman, A. Napier, and J. Stutz. NETGEN: A Program for Generating Large Scale Capacitated Assignment, Transportation, and Minimum Cost Flow Network Problems. *Management Science*, 20:814–821, 1974.
25. Q. Lu, B. George, and S. Shekhar. Capacity Constrained Routing Algorithm for Evacuation Planning: A Summary of Results. *Technical Report, Department of Computer Science and Engineering, University of Minnesota, (05-023)*, 2005.
26. Q. Lu, Y. Huang, and S. Shekhar. Evacuation Planning: A Capacity Constrained Routing Approach. *Proceedings of the First NSF/NIJ Symposium on Intelligence and Security Informatics*, pages 111–125, June 2003.
27. H.S. Mahmassani, H. Sbayti, and X. Zhou. *DYNASMART-P Version 1.0 User's Guide*. Maryland Transportation Initiative, University of Maryland, September 2004.
28. N.J. Nilsson. *Principles of Artificial Intelligence*. Tioga Publishing Co., 1980.
29. J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison Wesley, 1984.
30. S. Shekhar. Evacuation Planning: Presentation at UCGIS Congressional Breakfast Program on Homeland Security. <http://www.ucgis.org/winter2004/program.htm>, February 2004.
31. S. Shekhar and Q. Lu. Evacuation Planning for Homeland Security. *Minnesota Homeland Security and Emergency Management newsletter*, October 2004.
32. F.B. Zhan and C.E. Noon. Shortest Paths Algorithms: An Evaluation Using Real Road Networks. *Transportation Science*, 32:65–73, 1998.
33. A.K. Ziliaskopoulos and H.S. Mahmassani. A Note on Least Time Path Computation Considering Delays and Prohibitions for Intersection Movements. *Transportation Research B*, 30(5):359–367, 1996.