Evacuation Planning: A Capacity Constrained Routing Approach [†]

Qingsong Lu, Yan Huang, Shashi Shekhar
Department of Computer Science, University of Minnesota
200 Union St SE, Minneapolis, MN 55455
[*lqingson*, *huangyan*, *shekhar*]@cs.umn.edu

February 10, 2003

Abstract

Evacuation planning is critical for applications such as disaster management and homeland defense preparation. Efficient tools are needed to produce evacuation plans to evacuate populations to safety in the event of catastrophes, natural disasters, and terrorist attacks. Current optimal methods suffer from computational complexity and may not scale up to large transportation networks. Current naive heuristic methods do not consider the capacity constraints of the evacuation network and may not produce feasible evacuation plans. In this paper, we model capacity as a time series and use a capacity constrained heuristic routing approach to solve the evacuation planning problem. We propose two heuristic algorithms, namely Single-Route Capacity Constrained Planner and Multiple-Route Capacity Constrained Planner to incorporate capacity constraints of the routes. Experiments on a real building dataset show that our proposed algorithms can produce close-to-optimal solution, which has total evacuation time within 10 percent longer than optimal solution, and also reduce the computational cost to only half of the optimal algorithm. The experiments also show that our algorithms are scalable with respect to the number of evacuees.

Keywords: evacuation planning, disaster management, capacity constraint

---

[†] This work was partially supported by Army High Performance Computing Research Center contract number DAAD19-01-2-0014 and by NASA grant No. NCC 2 1231. 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 AHPCRC and the Minnesota Supercomputing Institute.

1 Introduction

Evacuation planning is critical for numerous important applications, e.g. emergency building evacuation, disaster management and recovery, and homeland defense preparation. Efficient tools are needed to produce evacuation plans which identifies routes and schedules to evacuate populations to safety in the event of catastrophes, natural disasters, and terrorist attacks [7, 3, 14, 4].

The current methods of evacuation planning can be divided into three categories, namely warning systems, linear programming approaches, and heuristic approaches. Warning systems simply convey threat descriptions and the need of evacuation to the affected people via mass media communication methods. Such systems can have unanticipated effects on the evacuation process. For example, when Hurricane Andrew was approaching Florida and Louisiana in 1992, the affected population was simply asked to leave the area as soon as possible. This caused tremendous traffic congestion on highways and led to great confusion and chaos [1]. The second type of evacuation planning, uses network flow and linear programming approach. EVACNET [8, 11, 12] computes optimal solution using linear programming methods. It has exponential running time and cannot be applied to large transportation networks. Hoppe and Tardos [9, 10] gave the first polynomial algorithm to compute optimal solution for evacuation problem. However, their algorithm uses ellipsoid method which suffers from high computational complexity and therefore is not practical to implement. The third type of evacuation planning uses heuristics approaches to find evacuation plans. However, current naive heuristic approaches only compute the shortest distance path to the nearest exit without considering route capacity constraints and traffic from other sources. It cannot produce efficient plans when the number of people to be evacuated is large and the route network is complex.

New heuristic approaches are needed to account for capacity constraints of the evacuation network. A capacity constrained routing approach reserves route capacities subject to capacity constraints in an order specified by heuristics. We propose two new heuristic algorithms for capacity constrained routing, namely single-route approach and multiple-route approach. The first algorithm evacuates all the people from the same source via a single route by reserving route capacity based on an order determined by pre-computed shortest path lengths. The second algorithm can assign multiple routes to groups of people from the same source based on an order prioritized by shortest travel time path lengths re-calculated in each iteration. The multiple-route approach produces close-to-optimal solutions with significantly reduced computational time compared to optimal solution algorithms. It outperforms the single-route approach in solution quality because of its flexibility in choosing routes although it is computationally more expensive since the single-route approach can produce solution for large network in seconds. Experimental results on a large building dataset show that our proposed algorithms can produce sub-optimal solution, which has total evacuation time within 10% longer than optimal solution, and at the same reduce the computational cost to only half of the optimal algorithm. Our algorithms are also scalable with respect to the total number of people to be evacuated. To the best of our knowledge, this is the first paper exploring heuristic algorithms using capacity constrained routing for evacuation planning.

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. Section 3 proposes two capacity constrained heuristic algorithms. The algorithm comparison and cost models are given in Section 4. In Section 5, we presents the experimental design and results. We summarize our work and discuss future directions in Section 6.

Scope: The proposed approaches can not be applied directly to routing models which have intersection queuing delays common in vehicle routing.

2 Problem Formulation

The capacity constrained routing problem can be formulated as follows. Given a transportation network with capacity constraints, the initial number of people to be evacuated, their locations, and evacuation destinations, we need to produce evacuation route plans consisting of a set of origin-destination routes and a scheduling of people to be evacuated via the routes. The objective is to minimize the total time needed for evacuation. The scheduling of people onto the routes should observe the route capacity constraints and other application dependent constraints, e.g. total time available for evacuation, etc. A secondary objective is to minimize the computational overhead of producing the evacuation plan.

We illustrate the problem formulation and a solution with an example. Suppose we have a simple two-story building, as shown in Figure 1 (floor map from [12]). In this building, there are two rooms on the second floor, two staircases, and one room and two exits on the first floor. This building will be modeled as a node-edge graph, as shown in Figure 2.

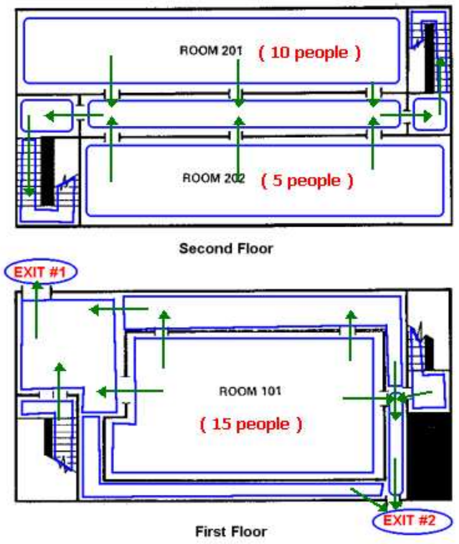

Figure 1: Building Floor Map with Node and Edge Definition

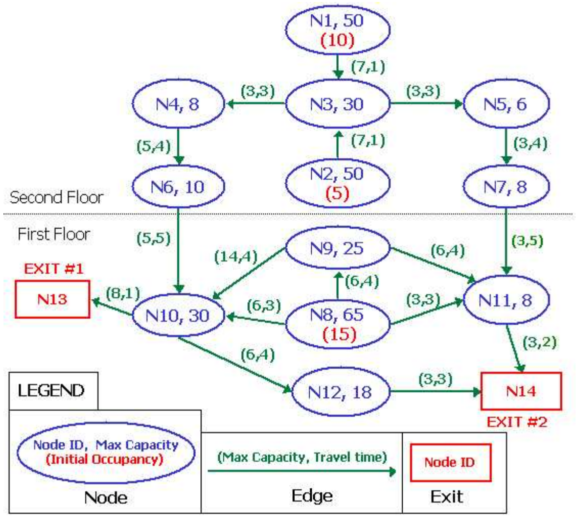

Figure 2: Node-Edge Graph Model of Example Building

In this model, each room, corridor, staircase, and exit of the building is represented as a node, shown by an ellipsis. Each node has two attributes: maximum node capacity and initial node occupancy. For example, at node N1, which represents Room 201 in the building, the maximum capacity is 50, which means Room 201 can hold at most 50 people, while the initial occupancy is 10, which means there are initially 10 people in this room that are to be evacuated. Each pathway from one node to another node is represented as an edge, shown by arrows between the two nodes in Figure 2. Each edge also has two attributes: maximum edge capacity and travel time. For example, at edge N1-N3, which represents the path linking Room 201 and the corridor, the maximum capacity is 7, which means at most 7 people can travel from Room 201 to the corridor simultaneously, while the travel time is 1, which means it takes 1 time unit to travel from the room to the corridor. This approach to model building floor-map with capacity to node-edge graph is similar to those presented in [12, 5].

As shown in Figure 2, suppose we initially have 10 people at node N1, 5 at node N2, and 15 at node N8. The task is to compute an evacuation plan that evacuates the 30 people to the exits (N13 and N14) using the least amount of time.

Example 1 (An Evacuation Plan) Table 1 shows an evacuation plan. In the table, each row shows one group of people moving together during the evacuation with a group ID, number of people in this group, origin node, the start time, the evacuation route, and the exit time. Take node N8 for exmaple, initially there are 15 people at N8. They are divided into 3 groups: Group A with 6 people, Group B with 6 people and Group C with 3 people. Group A starts at time 0, follows route N8-N10-N13 and reaches EXIT1(N13) at time 4. Group B starts at time 1, also follows route N8-N10-N13 and reaches EXIT2(N13) at time 5. Group C start at time 0, follows route N8-N11-N14 and reaches EXIT2(N14) at time 4. The procedure is similar for people from N1 and N2. The whole evacuation takes 16 time units since the last group of people (Group F and J) reaches the exit at time 16.

Group of People			Start Time	Route	Exit Time
ID	Origin	No. of People			
A	N8	6	0	N8-N10-N13	4
B	N8	6	1	N8-N10-N13	5
C	N8	3	0	N8-N11-N14	5
D	N1	3	0	N1-N3-N4-N6-N10-N13	14
E	N1	3	1	N1-N3-N4-N6-N10-N13	15
F	N1	3	2	N1-N3-N4-N6-N10-N13	16
G	N1	1	0	N1-N3-N5-N7-N11-N14	16
H	N2	3	0	N2-N3-N5-N7-N11-N14	15
I	N2	2	1	N2-N3-N5-N7-N11-N14	16

Table 1: Evacuation Plan Example

3 Capacity Constrained Routing Approach

We use a capacity constrained routing approach to conduct the evacuation planning. We model available edge capacity and available node capacity as a time series instead of a fixed number. A time series represents the available capacity at each time instant for a given edge or node. We propose an approach based on the extension of shortest path algorithms [6] to account for route scheduling with capacity constraints. We propose two heuristic algorithms to compute the evacuation plan.

3.1 Single-Route Capacity Constrained Routing Approach

In the Single-Route Capacity Constrained Planner (SRCCP) algorithm, first, all the shortest routes from sources to any destination are pre-computed. Next, capacities are reserved along the pre-computed routes by reducing available node and edge capacities at certain time points along the route. The detailed pseudo-code and algorithm description are as follows.

Algorithm 1 Single-Route Capacity Constrained Planner

```
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:
             $Maximum\_Node\_Capacity(n)$, $Initial\_Node\_Occupancy(n)$: non-negative integer
          Each edge $e \in E$ has two properties:
             $Maximum\_Edge\_Capacity(e)$, $Travel\_time(e)$: non-negative integer
       2) $S$: set of source nodes, $S \subseteq N$;
       3) $D$: set of destination nodes, $D \subseteq N$;
Output: Evacuation plan
Method:
for each source node $s \in S$ do
   find the shortest time route $R_s < n_0, n_1, \ldots, n_k >$ among routes
      from $s$ to all destination nodes $d \in D$, ( where $n_0 = s$ and $n_k = d$ );
Sort routes $R_s$ by total travel time, increasing order;
for each route $R_s$ in sorted order do {
   Initialize next start node on route $R_s$ to move: $st = 0$;
   while not all evacuees from $n_0$ reached $n_k$ do {
      find next available time $t$ to start move from node $n_{st}$;
      find the furthest node $n_{end}$ that can be reached from $n_{st}$ without stopping;
      $flow = \min($ number of evacuee at node $n_{st}$,
                 $Available\_Edge\_Capacity($all edges between node $n_{st}$ and $n_{end}$ on route $R_s)$,
                 $Available\_Node\_Capacity($all nodes from node $n_{st+1}$ to $n_{end}$ on route $R_s)$,
               $)$;
      for $i = st$ to $end - 1$ do {
         $t' = t + Travel\_time(e_{n_i n_{i+1}})$;
         $Available\_Edge\_Capacity(e_{n_i n_{i+1}}, t)$ reduced by $flow$;
         $Available\_Node\_Capacity(n_{i+1}, t')$ reduced by $flow$;
         $t = t'$;
      }
      next start node $st$ =closest node to destination on route $R_s$ with evacuee;
   }
}
Postprocess results and output evacuation plan;
```

Group of People			Start Time	Route	Exit Time
ID	Origin	No. of People			
A	N8	6	0	N8-N10-N13	4
B	N8	6	1	N8-N10-N13	5
C	N8	3	2	N8-N10-N13	6
D	N1	3	0	N1-N3-N4-N6-N10-N13	14
E	N1	3	0	N1-N3(W1)-N4-N6-N10-N13	15
F	N1	1	0	N1-N3(W2)-N4-N6-N10-N13	16
G	N1	2	1	N1-N3(W1)-N4-N6-N10-N13	16
H	N1	1	1	N1-N3(W2)-N4-N6-N10-N13	17
I	N2	2	0	N2-N3(W3)-N4-N6-N10-N13	17
J	N2	3	0	N2-N3(W4)-N4-N6-N10-N13	18

Table 2: Result Evacuation Plan of the Single-Route Capacity Constrained Planner

In the first step(line 1-2), for each source node s, we find the route R_s with shortest total travel time among routes between s and all the destination nodes. The total travel time of route R_s is the sum of the travel time of all edges on R_s. For example, in figure 2, R_{N1} is N1-N3-N4-N6-N10-N13 with a total travel time of 14 time units. R_{N2} is N2-N3-N4-N6-N10-N13 with a total travel time of 14 time units. R_{N8} is N8-N10-N13 with total travel time of 4 time units. This step is done by a variation of Dijkstra's [6] algorithm in which edge travel time is treated as edge weight and the algorithm termintes when the shortest route from s to one destination node is determined.

The second step(line 3), is to sort the routes we obtained from step 1 in increasing order of the total travel time. Thus, in our example, the order of routes will be R_{N8},R_{N1},R_{N2}.

The third step(line 4-18), is to reserve capacities for each route in the sorted order. The reservation for route R_s is done by sending all the people initially at node s to the exit along the route in the least amount of time. The people may need to be divided into groups and sent by waves due to the constraints of the capacities of the nodes and edges on R_s.

For example, for R_{N8}, the first group of people that starts from N8 at time 0 is at most 6 people because the available edge capacity of N8-N10 at time 0 is 6. The algorithm makes reservations for the 6 people by reducing the available capacity of each node and edge at the time point that they are at each node and edge. This means that available capacities are reduced by 6 for edge N8-N10 at time 0 because the 6 people travel through this edge starting from time 0; for node N10 at time 3 because they arrive at N10 at time 3; for edge N10-N13 at time 3 because they travel through this edge starting from time 3. They finally arrive at N13(EXIT1) at time 4. The second group of people leaving N8 has to wait until time 1 since the first group has reserved all the capacity of edge N8-N10 at time 0. Therefore, the second group leaves N8 at time 1 and reaches N13 at time 5. Similarly, the last group of 3 people leaves N8 at time 2 and reaches N13 at time 6. Thus all people from N8 are sent to exit N13. The next two routes, R_{N1} and R_{N2}, will make their reservation based on the available capacities that the previous routes left with.

The final step of the algorithm is to output the entire evacuation plan, as shown in Table 2, which takes 18 time units.

3.2 Multiple-Route Capacity Constrained Routing Approach

The Multiple-Route Capacity Constrained Planner (MRCCP) is an iterative approach. In each iteration, the algorithm re-computes the earliest time route from any source to any destination taking the previous reservations and possible on-route waiting time into consideration. Then it reserves the capacity for this route in the current iteration. The detailed pseudo-code and algorithm description are as follows.

Algorithm 2 Multiple-Route Capacity Constrained Planner

```
   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:
                 $Maximum\_Node\_Capacity(n)$, $Initial\_Node\_Occupancy(n)$:  non-negative integer
              Each edge $e \in E$ has two properties:
                 $Maximum\_Edge\_Capacity(e)$, $Travel\_time(e)$:  non-negative integer
           2) $S$:  set of source nodes, $S \subseteq N$;
           3) $D$:  set of destination nodes, $D \subseteq N$;
   Output:  Evacuation plan
   Method:
 while any source node $s \in S$ has evacuee do {                                         (1)
    find the route $R < n_0, n_1, \ldots, n_k >$ with earliest destination arrival time
       among routes between all $s$,$d$ pairs, $s \in S$,$d \in D$, (where $n_0 = s$ and $n_k = d$);   (2)
    $flow = \min($ number of evacuee still at source node $s$,
                 $Available\_Edge\_Capacity($all edges on route $R)$,
                 $Available\_Node\_Capacity($all nodes from node $n_1$ to $n_k$ on route $R)$,
               );                                                                          (3)
    for $i = 0$ to $k - 1$ do {                                                            (4)
       $t' = t + Travel\_time(e_{n_i n_{i+1}})$;                                            (5)
       $Available\_Edge\_Capacity(e_{n_i n_{i+1}}, t)$ reduced by $flow$;                   (6)
       $Available\_Node\_Capacity(n_{i+1}, t')$ reduced by $flow$;                          (7)
       $t = t'$;                                                                           (8)
    }                                                                                      (9)
 }                                                                                         (10)
 Postprocess results and output evacuation plan;                                           (11)
```

The MRCCP algorithm keeps iterating as long as there are still evacuees at any source node (line 1).

Each iteration starts with finding the route R with the earliest destination arrival time from any sources node to any any exit node based on the current available capacities (line 2). This is done by generalizing Dijkstra's shortest path algorithm [6] to work with the time series capacities and edge travel time. Route R is the route that reaches an exit in the least amount of time and at least one person can be sent to the exit through route R. For example, at the very first iteration, R will be N8-N10-N13, which can reach N13 at time 4. The actual number of people that will travel through R is the smallest number among the number of evacuees at the source node and the available capacities of each of the nodes and edges on route R (line 3). Thus, in the example, this amount will be 6, which is the available edge capacity of N8-N10 at time 0.

The next step is to reserve capacities for the people on each node and edge of route R (lines 4-9). The algorithm makes reservation for the 6 people by reducing the available capacity of each node and edge at the time point that they are at each node and edge. This means that available capacities are reduced by 6 for edge N8-N10 at time 0, for node N10

Group of People			Start Time	Route	Exit Time
ID	Origin	No. of People			
A	N8	6	0	N8-N10-N13	4
B	N8	6	1	N8-N10-N13	5
C	N8	3	0	N8-N10-N14	5
D	N1	3	0	N1-N3-N4-N6-N10-N13	14
E	N1	3	1	N1-N3-N4-N6-N10-N13	15
F	N1	3	0	N1-N3-N5-N7-N11-N14	15
G	N1	1	2	N1-N3-N4-N6-N10-N13	16
H	N1	3	1	N2-N3-N5-N7-N11-N14	16
I	N2	2	2	N2-N3-N5-N7-N11-N14	17

Table 3: Result Evacuation Plan of the Multiple-Routes Capacity Constrained Planner

at time 3, and for edge N10-N13 at time 3. They finally arrive at N13(EXIT1) at time 4. Then, the algorithm goes back to line 2 for the next iteration.

The iteration terminates when the occupancy of all source nodes is reduced to zero, which means all evacuee have been sent to exits. Line 11 outputs the evacuation plan, as shown in Table 3.

4 Comparison and Cost Models of the Two Algorithms

It can be seen that the key difference between the two algorithms is that the SRCCP algorithm only produces one single route for each source node, while the MRCCP can produce multiple routes for groups of people in each source node. MRCCP can produce evacuation plan with shorter evacuation time than SRCCP by the flexibility of adapting to the available capacities after previous reservations. Yet, MRCCP needs to re-compute the earliest time route in each iteration which incurs more computational cost than SRCCP.

We then provide simple algebraic cost models for the computational cost of the two proposed heuristic algorithms. We assume the total number of nodes in the graph is n, the number of source nodes is n_s, and the number of groups generated in the result evacuation plan is n_g.

The cost of the SRCCP algorithm consists of three parts: the cost of the computing the shortest time route from each source node to any exit node is denoted by C_{sp}, the cost of sorting all the pre-computed routes by their total travel time is denoted by C_{ss}, and the cost of reserving capacities along each route for each group of people is denoted by C_{sr}. The cost model of the SRCCP algorithm is given as follows:

$$Cost_{SRCCP} = C_{sp} + C_{ss} + C_{sr} = O(n_s \times nlogn) + O(n_s logn_s) + O(n \times n_g) \tag{1}$$

The MRCCP 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 is determined by the number of groups generated. In each iteration, the route with earliest destination arrival time from each source node to any exit node is re-computed with the cost of $O(n_s \times nlogn)$. Reservation is made for the node and edge capacities along the chosen route with the cost of $O(n)$. The cost model of the MRCCP algorithm is given as follows:

$$Cost_{MRCCP} = O((n_s \times nlogn + n) \times n_g) \tag{2}$$

In both cost models, the number of groups generated for the evacuation plan depends on the network configuration which include maximum capacity of nodes and edges, and the number of people to be evacuated at each source node.

5 Solution Quality and Performance Evaluation

In this section, we present the experiment design, our experiment setup, and the results of our experiments on a building dataset.

5.1 Experiment Design

Figure 3 describes the experimental design to evaluate the impact of parameters on the algorithms. The purpose is to compare the quality of solution and the computational cost of the two proposed algorithms with that of EVACNET which produces optimal solution. First, a test dataset which represents a building layout or road network is chosen or generated. The dataset is a evacuation network characterized by its route capacities and its size (number of nodes and edges). Next, a generator is used to generate the initial state of the evacuation by populating the network with a distribution model to assign people to source nodes. The initial state will be converted to EVACNET input format to produce optimal solution via EVACNET and converted to node-edge graph format to evaluate the proposed two heuristic algorithms. The solution qualities and algorithm performance will be analyzed in analysis module.

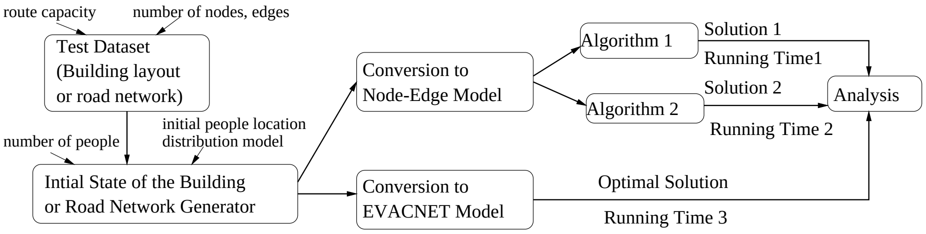

Figure 3: Experiment Design

5.2 Experiment Setup and Results

The test dataset we used in the following experiments is the floor-map of Elliott Hall, a 6-story building on the University of Minnesota campus. The dataset network consists of 444 nodes with 5 exits nodes, 475 edges, and total node capacity of 3783 people. The generator produces initial states by varying source node ratio and occupancy ratio from 10% to 100%.

The experiment was conducted on a workstation with Intel Pentium III 1.2GHz CPU, 256M RAM and Windows 2000 Professional operating system.

The people distribution generator distribute p_n people to n_s randomly chosen source nodes among all the nodes. The source node ratio is defined as $\frac{s_n}{\text{total number of nodes}}$ and the occupancy ratio is defined as $\frac{p_n}{\text{total capacity of all nodes}}$.

We want to answer two questions: (1)How does people distribution affect the performance and solution quality of the algorithms? (2) Are the algorithms scalable with respect to the number of people to be evacuated?

Experiment 1: Effect of People Distribution The purpose of the first experiment is to evaluate how the people distribution affects the quality of the solution and the performance of the algorithms. We fixed the occupancy ratio and varied the source node ratio to observe the quality of the solution and the running time of the two proposed algorithms and EVACNET.

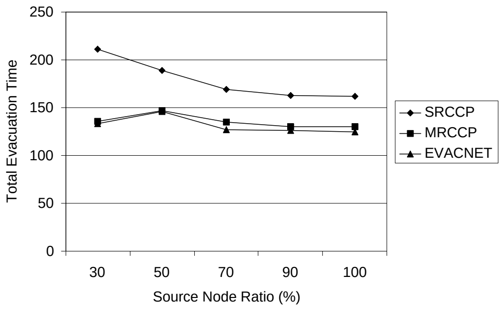

Figure 4: Quality of Solution With Respect to Source Node Ratio

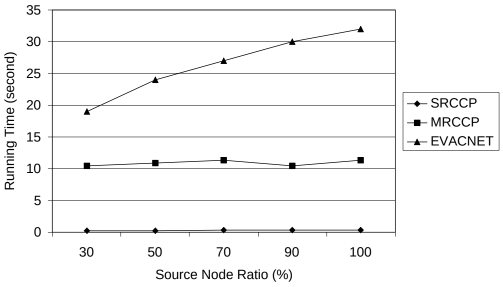

Figure 5: Running Time With Respect to Source Node Ratio

The experiment was done with fixed occupancy ratio from 10% to 100% of total capacity. Here we present the experiment results with occupancy ratio fixed at 30% and source node ratio varying from 30% to 100% which shows a typical result of all test cases. Figure 4 shows the total evacuation time given by the three algorithms and Figure 5 shows their running time.

As seen in Figure 4, at each source node ratio, MRCCP produces solution with total evacuation time that is within 10% longer than optimal solution produced by EVACNET. The quality of solution of MPCCP is not affected by the distribution of people when the total number of people is fixed. For SRCCP, the solution is 59% longer than EVACNET optimal solution when source node ratio is 30% and drops to 29% longer when source node ratio increases to 100%. It shows that the solution quality of SRCCP increases when source node ratio increases. In Figure 5, we can see that the running time of EVACNET grows much faster then the running time of SRCCP and MRCCP when source node ratio increases.

This experiment shows: (1)SRCCP produces solution closer to optimal solution when source node ratio is higher. (2)MRCCP produces close to optimal solution (less than 10%

longer than optimal) with less than half of running time of EVACNET. (3) The distribution of people does not affect the performance of two proposed algorithms when total number people is fixed.

Experiment 2: Scalability with Respect to Occupancy Ratio In this experiment, we evaluated the performance of the algorithms when the source node ratio is fixed and the occupancy ratio is increasing.

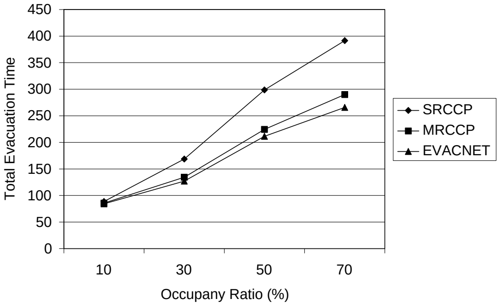

Figure 6: Quality of Solution With Respect to Occupancy Ratio

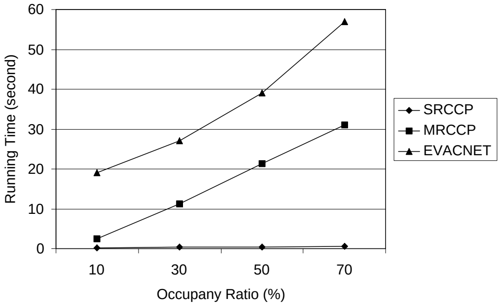

Figure 7: Running Time With Respect to Occupancy Ratio

Figure 6 and Figure 7 show the total evacuation time and the running time of the 3 algorithms when the source node ratio is fixed at 70% and occupancy ratio varies from 10% to 70% which is a typical case among all test cases.

As seen in Figure 6, compared with the optimal solution by EVACNET, solution quality of SRCCP decreases when occupancy ratio increases, while solution quality of MRCCP still remains within 10% longer than optimal solution. In Figure 7, the running time of EVACNET grows significantly when occupancy ratio grows, while running time of MRCCP remains less than half of EVACNET and only grows linearly.

This experiment shows: (1)The solution quality of SRCCP goes down when total number of people increases. (2) MRCCP is scalable with respect to number of people.

6 Conclusion and Future Work

In this paper, we proposed and evaluated two heuristic algorithms of capacity constrained routing approach. Cost models and experimental evaluations using a a real building dataset are presented. The proposed SRCCR algorithm can produces plan instantly but the quality of solution suffers when evacuee number grows. The MRCCR algorithm produces solution within 10% of optimal solution while the running time is scalable to number of evacuees and is reduced to half of the optimal algorithm. Both algorithms are scalable with respect to the number of evacuees.

Currently, we choose the shortest travel time route without considering the available capacity of the route. In many cases, a longer route with larger available capacity may be a better choice. In our future work, we would like to explore heuristics with route ranking method based on weighted available capacity and travelling time while choosing best routes.

We also want to extend and apply our approach to vehicle evacuation in transportation road networks. Modelling vehicle traffic during evacuation is a more complicated job than modelling pedestrian movements in building evacuation because modelling queuing at intersections and the cost of taking turns are challenging tasks. Current vehicle traffic simulation tools, e.g. DYNASMART [13], DYNAMIT [2], uses an assignment-simulation method to simulate the traffic based on origin-destination routes. We plan to extend our approach to work with such traffic simuation tools to address vehicle evacuation problems.

Acknowledgment

We are particularly grateful to Spatial Database Group members 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 partially supported by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014 and by NASA grant No. NCC 2 1231, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

References

[1] Hurricane evacuation web page. *http://i49south.com/hurricane.htm*, 2002.

[2] M. Ben-Akiva et al. *Deveopment of Dynamic Traffic Assignment System for Planning Purposes: DynaMIT User's Guide.* ITS Program, MIT, 200.

[3] S. Browon. Building America's Anti-Terror Machine: How Infotech Can Combat Homeland Insecurity. *Fortune*, pages 99–104, July 2002.

[4] The Volpe National Transportation Systems Center. Improving Regional Transportation Planning for Catastrophic Events(FHWA). *Volpe Center Highlights*, pages 1–3, July/August 2002.

[5] L. Chalmet, R. Francis, and P. Saunders. Network Model for Building Evacuation. *Management Science*, 28:86–105, 1982.

[6] C. Corman, T. Leiserson and R. Rivest. *Introduction to Algorithms.* MIT Press, 1990.

[7] ESRI. GIS for Homeland Security, An ESRI white paper. *ESRI White Paper*, November 2001.

[8] 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.

[9] 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.

[10] 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.

[11] T. Kiosko and R. Francis. Evacnet+: A Computer Program to Determine Optimal Building Evacuation Plans. *Fire Safety Journal*, 9:211–222, 1985.

[12] T. Kiosko, R. Francis, and C. Nobel. *EVACNET4 User's Guide.* University of Florida, 1998.

[13] H.S. Mahmassani et al. *Development and Testing of Dynamic Traffic Assignment and Simulation Procedures for ATIS/ATMS Applications.* Technical Report DTFH6 1-90-R-00074-FG, CTR, University of Texas at Austin, 1994.

[14] A. Moore. Data Mining technologies for Homeland Defense. *Tutorial on Mining Massive Data Sets and Streams: Mathematical Methods and Algorithms for Homeland Defense*, June 2002.