

A Simple Pedestrian Simulator Using Node-Edge Graphs for Floorplan Models

Constantine K. Christakos
Cambridge, MA
Email: dean@alum.mit.edu

Abstract—Some simulation environments may wish to evaluate effects on flow and movement, such as in a pedestrian simulation. Analysis of the flow is particularly useful when the map in which the flow occurs is modeled as a node-edge graph. I create a simple pedestrian simulation in which pedestrian movement occurs in a node-edge graph and the pedestrians direct themselves towards an “escape” node as their goal. The node-edge graph allows designers to analyze the performance results in terms of number of nodes and connectivity and to run trials in layouts created by random graph generators. I base pedestrian behavior in the simulation on physics-based models developed in other pedestrian simulations. The result is an extremely simple pedestrian simulation that allows the user to estimate the effect of unevenly-distributed crowds, floorplan size, and number of pedestrians on escape time. Furthermore, the simulation can be augmented with different methods of providing instructions to pedestrians to see the effect of those instructions on escape time. The focus on simplicity allows for quick analysis and extension as well as easy integration with other simulations.

I. INTRODUCTION

This project was motivated by the need to create a simple simulator that would measure escape times of pedestrians in a changing environment. Specifically, I was concerned with the problem of pedestrian escape during a situation in which there are rapidly changing conditions within the environment and where passageways may be blocked and reopened during an emergency. These blocked passageways may occur due to crowding, due to physical collapses of the passageway itself, or due to other hazards in the area.

With such a model, one can analyze the system in terms of node size and pedestrian density and how those features affect performance. When integrated with a network simulation, relationships can be between network performance and features such as graph size and graph connectivity and corresponding pedestrian performance can be examined.

First, this paper goes over some brief background of pedestrian simulations and previous work in that field and covers some work in geographical representation of layouts. Next, I describe the architecture of the simulator I built and explain the design decisions made. Finally, I present some basic performance results and conclude with discussion of applications of the simulator as well as further extensions that can be made to it.

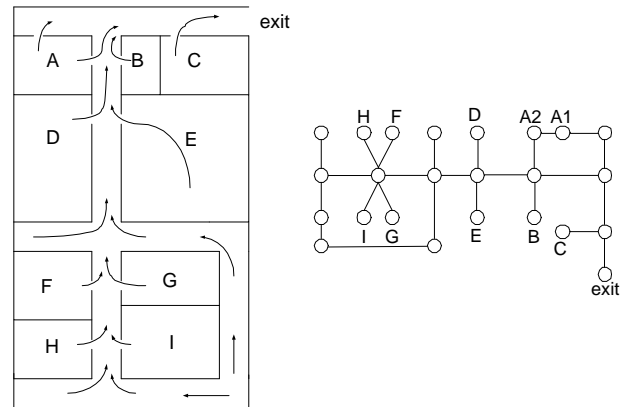


Fig. 1. **Left:** The paths to which users will be directed in order to reach the exit the fastest. **Right:** That layout as it corresponds to a graph, with individual rooms maintaining their same label

II. BACKGROUND

A. Representation of Layouts

Solving the problem of escape from a building can be mapped to a pathfinding problem in a graph (Figure 1). The sink in the graph can be represented as the exit. Each node represents a point at which a user can make a choice of which direction to go, aided by a sensor indicating the direction of the shortest path to the exit. The goal is to find the shortest path in a graph and to find a new path when the current one becomes unavailable.

Representing floorplans as graphs allows others to more keenly analyze constraints of a floorplan. Tools of graph theory have been applied to the geometry of architecture. Applications of graph theory to architectural floorplans fall into two categories: constraint analysis and path finding, both of which have been explored in detail within the field.[1] Analysis of geographical layouts such as mazes (Figure 2) has been aided by its representation as a node-edge model (Figure 3).[2]

Within the field of pedestrian simulation itself, representation of the pedestrian environment has adopted the node-edge model in applications such as park management[3] and representation of urban traffic.[4] The latter example of [4] is particularly relevant to this work because it deals specifically with wireless network simulations, albeit with respect to seeing the network would act if pedestrians could serve as wireless

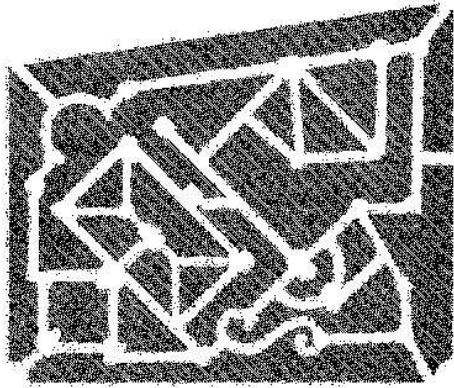


Fig. 2. The hedge maze at Versailles.

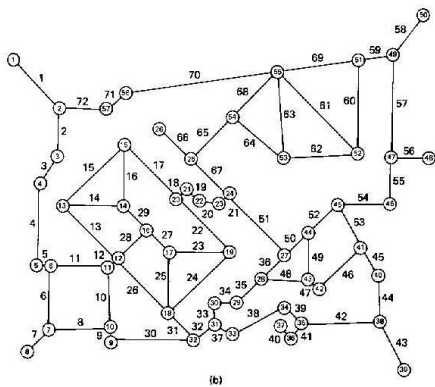


Fig. 3. The hedge maze at Versailles represented as a graph.

nodes, rather than using wireless communications to direct pedestrians. That pedestrian simulation work was developed simultaneously as the work presented here and could easily be integrated in the same area in which this work was applied.[5]

B. Movement of Pedestrians

To make a simple simulator, I needed an accurate – or at least believable – model for pedestrian movement.

Simulating the behavior of pedestrians is an active research area, whose diverse applications include areas from park management [6] to room escape [7] to traffic crossings.[8] Pedestrian behavior is affected by the speed and direction at which the pedestrian wishes to travel, and the sum of all forces exerted upon him by other pedestrians and the walls. This force model is given by

$$m_i \frac{dv_i}{dt} = m_i \frac{v_i^0(t)e_i^0(t) - v_i(t)}{\tau_i} + \sum_{j(=i)} f_{ij} + \sum_w f_{iW} \quad (1)$$

where a pedestrian i has a mass m_i and a desired velocity v_i^0 in the direction e_i^0 . v_i is the pedestrian's actual velocity, and f_{ij} and f_{iW} are the forces of each other adjacent pedestrian, j , and the walls adjacent to pedestrian i , respectively. In some cases, the sum of the forces exerted on the pedestrian is sufficient to cause injury or death, and some pedestrian simulations [7]

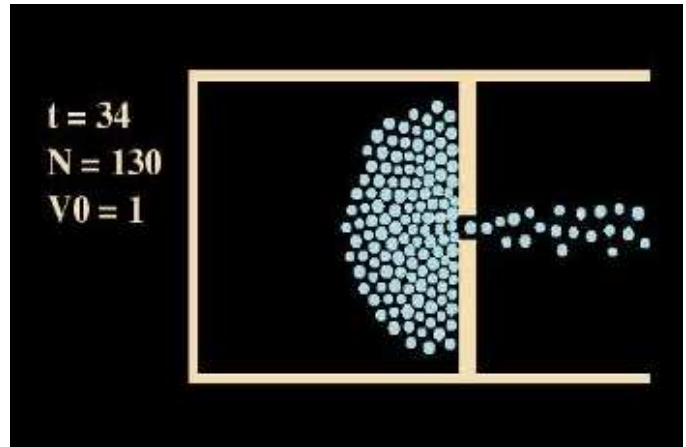


Fig. 4. A simulation of pedestrians escaping through a room in the Panic simulator.

take these possibilities into account in order to predict how dangerous an environment is.

Modeling the precise physical interactions between pedestrians is beyond the scope of this work. In fact, many projects have decided to model pedestrian traffic probabilistically rather than dealing with specific physical interactions.[8], [9] I seek to come up with a simple means of modeling pedestrian traffic within a floorplan and to gain an understanding about how crowds of pedestrians affects their ability to travel. To this end, I examined two different pedestrian scenarios – pedestrian travel through hallways (in my model, the edges of the graph) and pedestrian travel through rooms (in my model, the nodes of the graph). The initial hypothesis was that pedestrians traveled with a velocity v_0 in isolation, but their velocity would change according to a recognizable function of the number of other pedestrians in that hallway or room. Using the pedestrian simulator developed by [7], I ran a number of simulations of pedestrian escape times out of a simulated room to see how pedestrian throughput was affected by the number of other pedestrians in the room. (Figure 4) Looking at the overall throughput of pedestrians given different initial starting values for the number of pedestrians in the room, one sees that the time it takes for a room to completely empty is roughly linear, and the average throughput of pedestrians in all cases is relatively constant (Figure 5). This provided an indication that there was no real difference in velocity as a function of the number of pedestrians. Rather, pedestrians only took a longer time to escape out of a crowded room because there were so many other pedestrians in front of them. The more pedestrians in the room, the longer a pedestrian would have to wait to leave, similar to if he was standing in a queue. To confirm my suspicions, I re-analyzed the data to examine the average amount of time it took a single pedestrian to escape a room given that there were X pedestrians in the room at that time. For example, in the case of how long it took a single pedestrian to leave when there were 50 pedestrians in the room, I looked at all simulations of between 50 and 100 starting pedestrians in the room and took the average of the

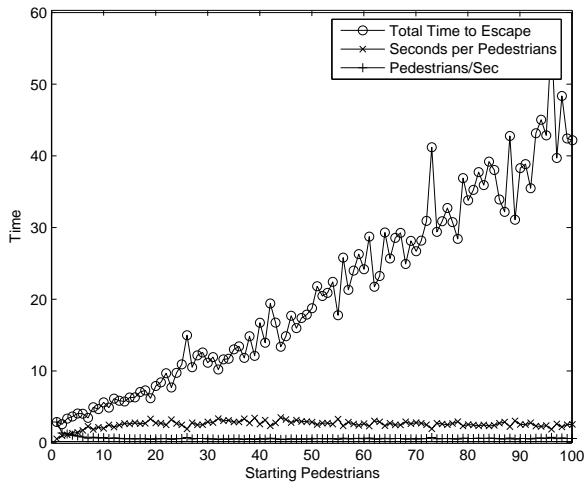


Fig. 5. An examination of escape times out of a room, given the starting number of pedestrians for that simulation.

amount of time it took for the number of pedestrians in the room to go from 50 to 49. The results are similar (Figure 6) in the case of these “marginal” escape times—the time to escape—about half a second—is unaffected by the number of pedestrians in the room at that time. The only difference is at the ends of the graph—for 1 pedestrian and 99-100 pedestrians. In the case of 1 pedestrian, the simulation ran longer because the single pedestrian starts at the middle of the room, adding simulation time while the pedestrian travels to the exit. In the case of 99-100 pedestrians, there were only a few cases in which there were this many pedestrians in the room, and there were few data points to average. Once again, the time it took for the initial pedestrian to leave dominated the average, distorting the results, whereas in all other cases, the average escape time was relatively constant, regardless of how many pedestrians were in the room.

I concluded that a simple yet effective way to model pedestrian escape from a room was as a queue. The pedestrian at the head of the queue waits before escaping, removing himself from the queue and entering the hallway. The next pedestrian in the queue then begins to wait a certain interval before he, too, is removed, and so on, until the queue is empty (Figure 7).

As opposed to escaping from rooms, I did not find many crowding effects in the hallways using the Helbing simulator. In addition, the act of leaving a room one by one means that the pedestrians will have minimal physical interactions once they are in the hallway. However, it is possible that in a situation where there are two separate crowds of pedestrians moving in a hallway in opposite directions that pedestrian interaction effects will be significant and affect movement speed. I discuss possible ways of dealing with that situation in the last section.

By designating nodes and edges with a fixed amount of “bandwidth” that pedestrians must abide by, which has been

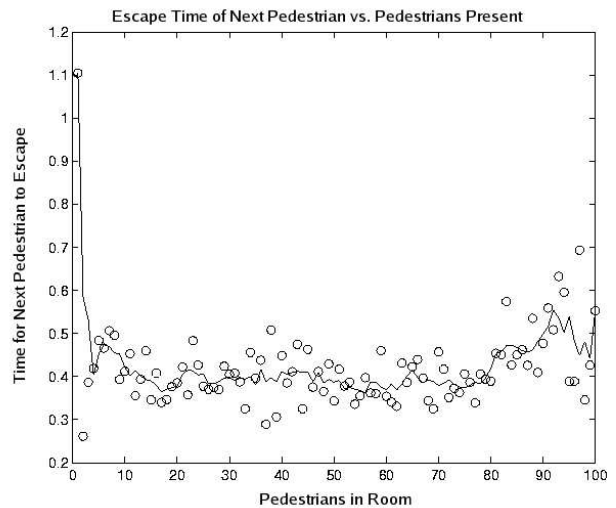


Fig. 6. The average escape time for a single pedestrian to leave a room, given the number of pedestrians in the room at the time, averaged over all simulations, with moving average

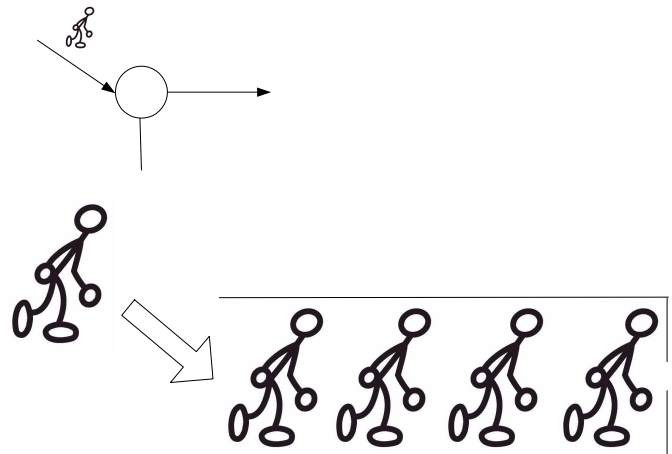


Fig. 7. A pedestrian entering a node over in incoming edge (top) waits in a queue before leaving the node on the outgoing edge (bottom).

called a “flow-based” approach to modeling.[10] One prominent example of a such a system is EVACNET4.[11]

III. BEHAVIOR AND IMPLEMENTATION

As described above, the pedestrian-escape simulations showed that most of the delay in room escape due to people in the room occurs because each pedestrian needs to squeeze through the doorway. Instead of modeling the precise physical interactions between pedestrians, I simply place each new, arriving pedestrian into a queue for a certain edge. The arriving pedestrians must get into a queue behind the other pedestrians and wait until they leave the node over that edge before that pedestrian can exit. This queue models the crowding around an exit, as shown in the Panic simulator.[7] Regardless of how long the queue is, pedestrians leave the node through the exit edge at the same rate. When the simulation is initialized, a

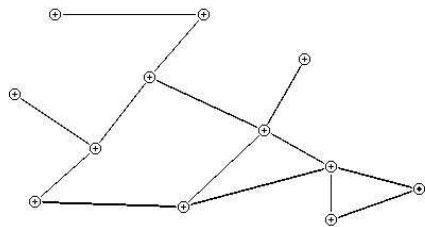


Fig. 8. A Sample scenario of the pedestrian simulator.

shortest path spanning tree is calculated with the exit node as the root. The default behavior of the pedestrians when they arrive at a node is to choose the edge in that spanning tree that leads to that exit node.

However, if too many pedestrians are waiting in a queue over an exit edge, then the doorway to an edge is considered “blocked”; the pedestrian will then decide to “flee” and seek out an alternate route to the exit. The pedestrian does so by choosing random edges from each node for 5 steps from one node to the next. At this point, the pedestrian will either have reached the exit, or the pedestrian will return to always choosing the designated “exit edge” each time he enters a node.

The consequences of crowding to the problem of escape are clear – if no edges are ever blocked, then pedestrians proceed normally through the graph, taking the shortest path from their start nodes to the exit node. On the other hand, if edges are consistently blocked, some pedestrians will flee throughout the graph at random, over and over again, stymied by both the blocked passageways and their inability to find an alternate path to the exit node.

The system itself is implemented in Java and can run with a simple visual presentation (Figure 8). This was originally constructed as a multi-threaded system with each of the pedestrians acting independently, while all of their movements were tied to a global clock. However, more consistent results were gathered when the global clock controlled each individual pedestrian, and the former approach was abandoned in favor of the latter. Pedestrians are assumed to travel at a rate of 12 feet/sec. The distance between any two connected nodes is always considered to be equal within the graph, assumed to be 10 feet. In addition, there is the wait time of half a second to leave a node after it is entered. Support could be added to model distances of different lengths and how this would affect the calculation of the “best path” between a node and the exit, but this is tangential to the evaluation. The current model’s results remain valid without loss of generality; the

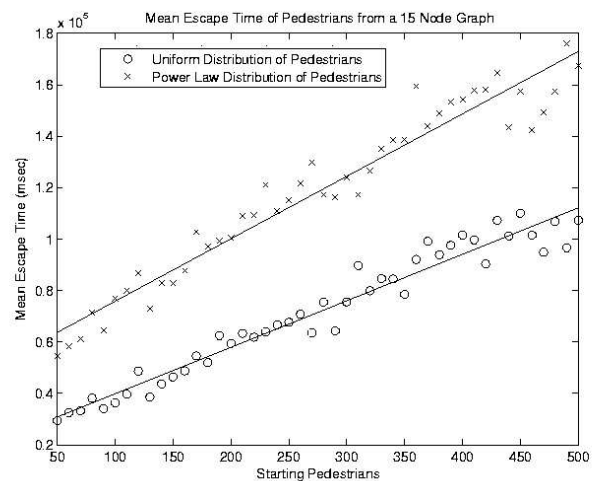


Fig. 9. A comparison mean escape times of pedestrians in both a power-law and uniform distribution escaping through a 15 node graph.

floorplan would simply be modeled as a weighted graph, and shortest paths to the resource node would be calculated using a minimum-weight spanning tree rather than a shortest-path spanning tree.

IV. RESULTS

The results of some basic experiments with this similar follow. Specifically, I am interested in the effects of crowding and pedestrian density on the escape time performance and, ultimately, whether such performance can be improved by augmenting the decisions made by the pedestrians. For now, however, I will examine consequences of crowding are under the standard model of pedestrian behavior described in the previous section.

To get an idea of how the pedestrian simulation performed as the number of pedestrians rose, I ran several trials with randomly generated graphs of a fixed size, and seeded the graph with an increasing number of pedestrians. One of the possible scenarios considered was how building escape would be affected in the event that pedestrians within the building were holding a meeting, and thus crowded in a specific area. To simulate this scenario, I seeded the pedestrians in the graph according to a power-law distribution and compared that escape time to a uniform distribution of pedestrians in the graph (Figure 9). As one can see, the power-law distribution of pedestrian results in consistently higher escape times than a uniform distribution of pedestrians through the graph.

Looking at the effects as the number of pedestrians remains constant while the graph expands (Figure 11), we still see consistently higher escape times due to power-law distribution of pedestrians, but the crowding effects (or rather, lack of crowding effects) appear to offset the increasing floorplan size.

V. DISCUSSION AND CONCLUSION

This paper has presented a simple pedestrian simulator that mimics the effects of other simulators that examine the effects

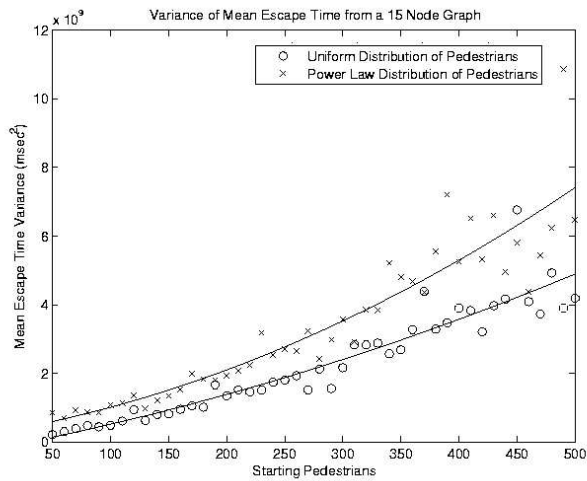


Fig. 10. A comparison of mean escape time variance for pedestrians in both a power-law and uniform distribution escaping through a 15 node graph.

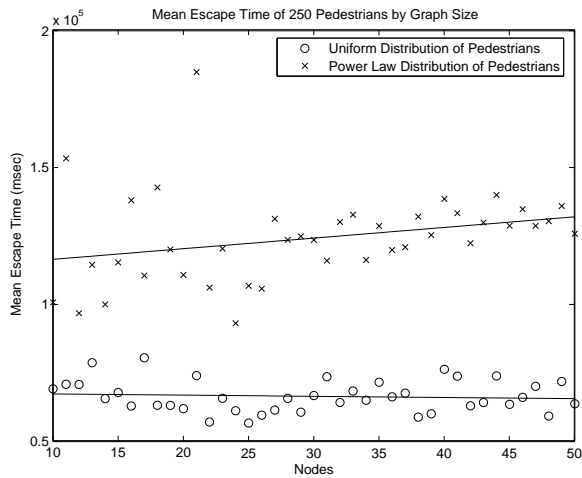


Fig. 11. A comparison of mean escape time for by graph size for both a power-law and uniform distribution of 250 pedestrians.

of pedestrian crowds. While the evaluation here is simple, the floorplan model—using simple nodes and edges—lends itself to evaluation of performance based on graph properties such as connectivity and graph size.

This model is very simple, treating all nodes in the graph as identical. A more realistic model of floorplans has been developed which treats some nodes as “transport nodes” that pedestrians would seamlessly pass through and other nodes as “room nodes” in which crowding effects would predominate when traveling in and out of. These features allow one to create more realistic models based on actual floorplans.

While pedestrians moving on edges do not interact with each other, such interactions could be abstracted into a simple model in the same way crowding through a doorway was abstracted into the simple queue model. One example is to assign a probability p of a direct collision which would force

both pedestrians to stop moving and then re-accelerate past each other to keep moving, resulting in a delay of time t_d while moving down the edge. The overall slowdown of a pedestrian traveling over an edge would be proportional to the number of other pedestrians in front of a traveling pedestrian and influenced by the probability of a head-on collision. For example, one pedestrian crossing the path of another could then be modeled as a slowdown by a factor of $p(1 + \frac{t_d}{t_0})$, where t_0 is the base time for an unencumbered pedestrian to travel over an edge. Progressively more pedestrians on the edge repeat this slowdown factor, in keeping with the “flow” model of pedestrian simulation.

The architecture of this system is open to such modifications and analysis and can be easily integrated with other simulators in order to examine the performance effects of changes in pedestrian behavior. [5] Its simplicity allows for easy extension and the use of a node-edge model makes it applicable to and reminiscent of network communication simulators.

ACKNOWLEDGMENT

This work was completed with support from the MIT Media Laboratory. The author would like to thank Andrew Lippman, Joseph Paradiso, and Samuel Madden for contributing to this work.

REFERENCES

- [1] L. March and P. Steadman, *The Geometry of the Environment*. Cambridge, MA: MIT Press, 1971.
- [2] W. J. Mitchell, *Computer-Aided Architectural Design*. New York, NY: Petrocelli/Charter, 1977.
- [3] C. Gloor, P. Stucki, and K. Nagel, “Hybrid techniques for pedestrian simulations,” in *Proceedings of the 4th Swiss Transport Research Conference*, Monte Verità / Ascona, March 2004.
- [4] K. Maeda, K. Sato, K. Konishi, A. Yamasaki, A. Uchiyama, H. Yamaguchi, K. Yasumoto, and T. Higashino, “Getting urban pedestrian flow from simple observation: Realistic mobility generation in wireless network simulation,” in *Proceedings of MSWiM 2005*, Montreal, Quebec, Canada, October 2005.
- [5] C. K. Christakos, “Sensor Networks Applied to the Problem of Building Evacuation: An Evaluation in Simulation,” in *Proceedings of the 15th Annual IST Mobile and Wireless Conference*, Mykonos, Greece, June 2006.
- [6] C. Gloor, D. Cavens, E. Lange, K. Nagel, and W. Schmid, “A pedestrian simulation for very large scale applications,” *A. Koch und P. Mandl (Hrsg.): Multi-Agenten-Systeme in der Geographie*, 2003.
- [7] D. Helbing, I. J. Farkas, and T. Vicsek, “Simulating dynamical features of escape panic,” *Nature*, vol. 407, pp. 487–490, 2000.
- [8] H. Oh and V. Sisiopiku, “Probabilistic models for pedestrian capacity and delay at roundabouts,” in *Proceedings of the Fourth International Symposium on Highway Capacity*, Maui, Hawaii, 2000.
- [9] R. L. Hughes, “The flow of human crowds,” *Annual Review of Fluid Mechanics*, vol. 35, pp. 169–182, January 2003.
- [10] G. Santos and B. E. Aguirre, “Critical review of emergency evacuation simulation models,” University of Delaware Disaster Research Center, Tech. Rep., 2004, preliminary Paper # 339.
- [11] T. Kisko, R. Francis, and C. Nobel, “EVACNET4 User’s Guide,” 1998, <http://www.ise.ufl.edu/kisko/files/evacnet/EVAC4UG.HTML>.