



Simulating optimum egress time

Kardi Teknomo^{*}, Proceso Fernandez

Department of Information Systems & Computer Science, Ateneo de Manila University, Katipunan Avenue, Quezon City, Metro Manila, Philippines

ARTICLE INFO

Article history:

Available online 11 February 2012

Keywords:

Minimum egress simulation
Route choice self-organization
Shortest path

ABSTRACT

A standard evacuation map posted in a room shows the location of the current room and the path to the nearest exit. If the number of occupants in the building is only small, then the shortest paths to the exits may enable all the occupants to evacuate in minimum possible time. Previous studies have shown, however, that the shortest path configuration does not lead to minimum egress time for large crowds in public facilities such as those in a school, theater or gym.

We extend the previous studies by determining the minimum egress time for different crowd sizes on a fixed network graph. We apply optimization search on a mesoscopic multi-agent pedestrian simulation that employs the concept of Route-Choice Self-Organization (RCSO). We show that the egress time gap between the shortest path configuration and RCSO configuration increases very quickly with respect to crowd size. Thus, for crowded pedestrian facilities, there may be a need to revise the standard evacuation map so that evacuation behavior that approximates the RCSO becomes possible, and this can lead to a much better egress time.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

There are many standards required by government for pedestrian facilities. These standards are enforced to safeguard the lives of the people inside such facilities. One important standard is on the facility's egress time, defined as the amount of time needed to evacuate everyone (Tubbs and Meacham, 2007).

To lower the egress time, evacuation maps are normally posted in rooms and other selected areas. Each of these maps shows the location of the current area and the path to the nearest exit. The purpose of these maps is to make people aware of where they should go in case of evacuation so that when the evacuation is needed, they would not spend unnecessary time searching for a way out.

The premise of using the shortest path is that the shorter the distance to be traveled, the less time is needed to cover that distance. However, this fails to consider the fundamental density–flow relationship, which establishes that if more people are on the same path (such as the case when everyone uses the shortest path) then the increased density actually results to decreased speed, and consequently, higher egress time.

Therefore, if the number of occupants in a building is only small, then the shortest paths to the exits may enable all the occupants to evacuate in minimum possible time. However, as shown in previous studies (e.g., Teknomo, 2008; Schneider and Könecke, 2010), the shortest path configuration does not lead to minimum egress

time for large crowds. This is not difficult to imagine, since the decrease in speed brought by congestion offsets the advantage of short distance towards the egress point. Therefore, the evacuation of large crowds in a public facility such as a school, theater or gym requires special consideration.

We extend the previous studies by determining the minimum egress time for different crowd sizes. We apply optimization search on a mesoscopic multi-agent pedestrian simulation on a fixed network graph that employs the concept of Route-Choice Self-Organization (RCSO). We show that the egress time gap between the shortest path configuration and RCSO configuration increases very quickly with respect to crowd size. Thus, for crowded pedestrian facilities, there may be a need to revise the standard evacuation map so that an evacuation behavior that approximates the RCSO becomes possible. Doing this may consequently lead to much better egress times.

This paper is organized as follows. Sections 2 and 3 cover preliminary discussions about the mesoscopic multi-agent pedestrian simulation and the route-choice self-organization, including related literature about these. Section 4 discusses the methodology used in the study. This is followed by a presentation and analysis of the derived results in Section 5. Finally, Section 6 summarizes the main contributions of this paper and proposes some improvements that can lower egress times.

2. Mesoscopic multi-agent simulation

Pedestrian movement can be modeled in the microscopic, macroscopic and mesoscopic levels. A microscopic pedestrian

^{*} Corresponding author. Tel.: +6324266001x5660.

E-mail addresses: teknomo@gmail.com (K. Teknomo), pfernandez@ateneo.edu (P. Fernandez).

simulation treats each pedestrian as an individual agent that moves about a virtual environment. Each agent is capable of independent action, including pursuit of specific goals and interaction with other agents. This model involves computation-intensive processes and is able to produce very detailed information about the pedestrian flow in some virtual environment. The studies of Blue and Adler (2000), Helbing and Molnár (1995), Hoogendoorn and Bovy (2004), Kretz and Schreckenberg (2006); Schadschneider (2001), to name a few of them, use this type of model.

On the other extreme, macroscopic pedestrian simulations model the environment using simpler details, and aggregate the pedestrian into groups that follow the fundamental flow–density relations. These models perform fewer computations than microscopic simulation models and are able to produce aggregate information only about the pedestrian system being modeled. Such modeling is the approach used for example by Henderson (1974), Kachroo and Ozbay (1999) and Lovas (1994).

Our study uses a model strategy that is between the two extremes described earlier. Called mesoscopic pedestrian simulation, our model involves aggregating environment information into a network graph, while still treating each pedestrian as an individual agent in the system. The result is a model with fewer computations than in microscopic models, but with more detailed results than in macroscopic models. Mesoscopic models have been used for studying pedestrian behavior in several papers also (Teknomo and Millonig, 2007; Teknomo et al., 2008). Slightly more aggregated and different mesoscopic models are also used in other studies (Tolujew and Alcalá, 2004; Florian et al., 2001).

In our mesoscopic model, a pedestrian environment is represented by a graph $G = (V, E)$ where V is the set of nodes, each representing a critical area of the environment (e.g., door, start or end point of a stair, elevator, escalator, ramp etc.) and E is the set of edges, each representing a direct path from one node to another. Fig. 1 illustrates a sample room layout with the corresponding network graph structure.

A *source node* in the network graph represents an area where the pedestrians are coming from. The two blue nodes on the left of the network graph in Fig. 1 are source nodes. A *sink node* (e.g., the brown colored nodes on the right most in Fig. 1) corresponds to a target egress point such as an exit door.

Each edge in the graph is assigned an equivalent length and an equivalent width, effectively limiting the maximum number of pedestrians that can simultaneously occupy the space represented by the edge. In this paper, this limit is called the *space capacity* of the given edge. For a given edge with length l , width w and maximum density ρ_{max} , the space capacity c is computed using the following equation:

$$c = l \cdot w \cdot \rho_{max} \quad (1)$$

A node in our model is not associated with a capacity value. Since a door is such a crucial element of a pedestrian evacuation and should have an associated capacity, it is modeled by an edge connecting two nodes that represent the opposite sides of a door (see Fig. 1). Other crucial elements of the environment, such as stairs, are represented similarly.

In our pedestrian simulation, an agent moves from a source node to a sink node through a valid path in the network graph. The actual path that is selected by an agent, among possibly many path options, is based on the rules described in the next section. By using a network graph instead of a virtual environment, some details are aggregated. This results to a simplified model where there are fewer computations to make.

3. Route choice self-organization

The mesoscopic pedestrian simulation that we employ in our study uses a multi-agent system that involves agents moving within a network graph. The path that an agent follows during the simulation is called the agent's *origin–destination path*, or simply OD path.

Traditional multi-agent simulations assign OD paths using either all-or-nothing (Dafermos and Sparrow, 1969) or multi-path (Dial, 1971) assignment. For a given origin and destination, the all-or-nothing scheme computes the best possible path, i.e., one that minimizes some utility cost function. Then, every agent that starts and ends on the same origin–destination pair is assigned to this minimum path. This is the model that is implicitly used in current evacuation maps where only the shortest path to an exit is diagrammed.

The multi-path assignment scheme is a bit more complex. For a given OD pair, the utility costs of all OD paths are computed in order to evaluate the relative attractiveness of each path. Then a probability distribution is computed among the paths, based on this attractiveness. Finally, the agents are stochastically assigned to a specific OD path using the generated distribution.

In both types of the traditional multi-agent simulations, the path of the agent is assigned prior to movement. Once the paths have been assigned, none of the agents ever deviate from his assigned path. This, however, does not accurately reflect actual pedestrian movement. For instance, a pedestrian who sees a congested path might decide to use an alternative less congested paths. Such a dynamic path decision-making is modeled better by what is called *Route-Choice Self-Organization* (RCSO).

An RCSO model enables an agent to dynamically update the route choice based on prevailing circumstances. Although this is done individually, an emergent group behavior is observed, thus the term self-organization. This produces a load-balancing phenomenon on the network, thereby lowering the egress time. The RCSO model used in this study is described in the succeeding paragraphs.

Each agent at a node v_i decides which edge e to take in order to get to which next node v_j , based on the values for *permission*, *interaction* and *navigation*, as discussed below:

- *Permission*: This is a binary value indicating whether or not there is a direct link from v_i to v_j . If such an edge e exists, then *permission* = 1, otherwise *permission* = 0.
- *Interaction*: This is a real number from the interval $[0, 1]$ indicating the attractiveness of a given edge, based on a measure of how congested the edge is. A value close to 0 implies the edge

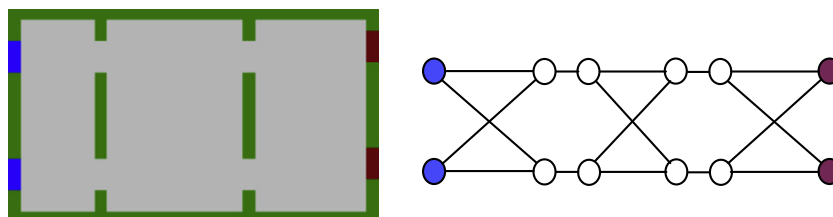


Fig. 1. A sample floor layout, and the corresponding network graph.

is not attractive (congested) while a value close to 1 indicates high desirability. Since it is a measure of congestion, the interaction value is therefore affected by the actual locations of the other agents in the system.

The formula that we use for the interaction value $I_{e,t}$ for the edge e at time t is given by

$$I_{e,t} = 1 - \text{BetaCDF}\left(\frac{\sigma_{e,t}}{c_e}; \theta_1, \phi_1\right) \quad (2)$$

where σ and c are the pedestrian volume and space capacity, respectively, of the given edge, while θ_1 and ϕ_1 are parameters to the Beta Cumulative Distribution Function. The Beta function was selected because the graph of its cumulative distribution is flexible enough to handle different varieties of expected volume interaction relationship. For example, when $\theta_1 = \phi_1 = 1$, the function is simply linear.

- **Navigation:** The range of values for this also comes from the interval $[0, 1]$. If the attraction component measures the attractiveness of an edge, the navigation component measures the attractiveness of a node at the other end of the edge.

First, every node v is assigned a value indicating a measure of some utility distance. We have called such assignment as *Sink Propagation Value* (SPV) since the values normally emanate from the sink nodes. Correspondingly, the assignment function is $SPV(v)$. One possible assignment function is the measure of actual distance of a node from a nearest sink. In this case, $SPV(v) = 0$ if v is a sink node, or $SPV(v) > 0$ otherwise. Fig. 2 illustrates a simple SPV computation for a given network where the edge lengths have been pre-assigned.

The navigation value N_{ij} from node v_i to an adjacent node v_j is a normalized value that also uses the Beta function, as described below.

$$N_{ij} = \begin{cases} 0 & \text{if } SPV(v_i) \leq SPV(v_j) \\ \text{BetaCDF}\left(\frac{SPV(v_i) - SPV(v_j)}{\max_{v_k \in \Gamma(v_i)} \{SPV(v_i) - SPV(v_k)\}}; \theta_2, \phi_2\right) & \text{otherwise} \end{cases} \quad (3)$$

Here, $\Gamma(v_i)$ is the set of nodes adjacent to v_i and having lower SPV than v_i , i.e., nearer to some sink. The parameters θ_2 and ϕ_2 are used for the Beta distribution function. Note that for static environments such as those where the doors do not open and close during the simulation, the navigation values need to be computed only at the start of the simulation. This is because the computed SPV for each node is not changed during any part of the simulation.

The movement of the agents in the network graph is modeled using discrete time steps by specifying behavioral update rules at given events. The main types of events are as follows:

1. **Agent insertion:** An agent is inserted into the simulation at a specific time and at a specific source node. For this study, all agents are inserted at the (same) source node at the start of the simulation.

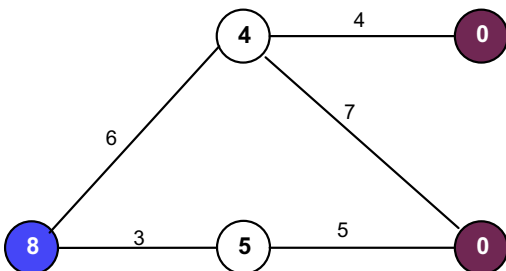


Fig. 2. SPV computation may be based on shortest distance from a node to a nearest sink.

2. **Movement from node to edge:** When an agent is on a node, it must decide which incident edge to take. Let v_i be the current node of the agent at time t and let v_j be an adjacent node from $\Gamma(v_i)$. The specific edge $e = (v_i, v_j)$ that is selected is the one that maximizes

$$k_e = \arg \max_{j \in \Gamma(v_i)} P_{e,t} \cdot I_{e,t} \cdot N_{ij} \quad (4)$$

Note that although the permission value $P_{e,t}$ and the navigation value N_{ij} do not change for a static environment the interaction value $I_{e,t}$ is dynamic, and is dependent on the current situation in the network.

3. **Movement within edge:** Once an agent enters an edge, its speed is computed based on the number of agents ahead and on the same edge, using the fundamental density–flow relationship.

$$\text{speed} = \text{speed}_{\max} \cdot \left(1 - \text{BetaCDF}\left(\frac{\sigma_{e,t}}{c_e}; \theta_3, \phi_3\right)\right) \quad (5)$$

where speed_{\max} is fixed at 1.2 m/s for all pedestrians.

The agent is assumed to move at this computed speed throughout its stay in the edge. Since the length of the edge is known, then the time that the agent exits the edge and arrives at the next node can be easily determined. The parameters θ_3 and ϕ_3 are used to adjust the cumulative Beta distribution function to control the speed–density relationship. The speed–density follows the [Green-shield's \(1934\)](#) linear model when $\theta_3 = \phi_3 = 1$. Knowing the speed–density function, the computation of the density–flow relationship is a straightforward application of the fundamental traffic flow formula.

4. **Termination:** An agent is removed from the system once it reaches a sink node. As such, its corresponding pedestrian is considered evacuated from the facility.

The algorithm for the RCSO model that we use in this study is summarized in Fig. 3. The main output is an NTXY table, which indicates the XY position (in this case, which specific node or edge) of an agent with id N at a given time T .

4. Methodology

The RCSO model produces a load-balancing effect on a given network. As such, it can be viewed as approximating the minimum possible egress time for a facility. To give us a better understanding of the dynamics of the (approximate) minimum egress time, the numerical experiments for this study were divided into two parts, as shown in Fig. 4.

The first set of experiments simulates the minimum egress time using RCSO. However, since the performance of the RCSO depends on the parameter values, we estimate the values that give the minimum egress time using Simultaneous Perturbation Stochastic Approximation (SPSA by [Spall \(1998\)](#)) integrated with Tabu list ([Glover, 1989](#)). The set of parameter values that produces the minimum egress time defined what we call the *minimum egress RCSO*, in order to differentiate it from *normal RCSO*, which uses the parameter values $\theta_1 = \phi_1 = \theta_2 = \phi_2 = \theta_3 = \phi_3 = 1$. The last two parameter values yield a linear relationship for the fundamental diagrams of traffic flow (in term of relationship between space mean speed and density).

The SPSA executed the simulation several times and terminated only when there was no improvement in the results after 500 iterations. For a fixed (relatively simple) network graph and an OD pair set, one simulation may require several seconds to several minutes of run-time. Thus an SPSA optimization search for the same input could run from a few hours to many days.

Algorithm: RCSO
Input: Network Graph $G=(V, E)$, and a set of OD pairs for the agent population
Output: NTXY Table

Pre-compute the navigational values $N_{i,j}$
 Clear the NTXY Table
 Insert the agents to the network

For each time step
 If there is no more agent in the network Then
 return NTXY table

For each agent A still in the network
 update A's position
 record entry to NTXY table

Fig. 3. Overview of the RCSO algorithm.

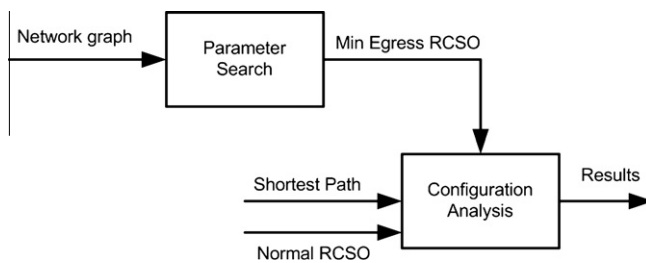


Fig. 4. Experiment design.

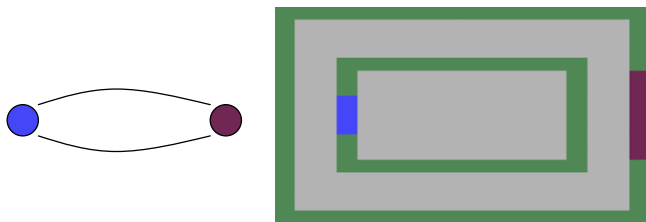


Fig. 5. Basic network graph used in the experiments, and a possible corresponding floor layout.

To derive results within reasonable time, we have decided to use a very simple and yet interesting network graph in this study. The graph contains only two nodes (one source and the other sink) connected using two edges so as to give pedestrians the chance to select a path (see Fig. 5).

After setting up the network graph structure, the lengths and widths of each of the edges, and also the total number of agents in the system were varied in order to determine the relationships among these. The many combinations of variable values also enabled better analysis of the three configurations being compared.

The aim in the first set of experiments is to elucidate the relationship of the independent variables to the egress time. To help in this cause, the NTXY table (produced by the simulation with minimum egress time, as returned by the SPSA optimization search) was used to generate the path-usage statistics. The various graphs for the experiments are shown in the section on results.

For the second set of experiments, the results of the previous set of experiments were used to compare with the results of simulations involving normal RCSO and the shortest path configuration. The aim of this part of the study is to validate the hypothesis that as the number of pedestrians in a facility increases, the gap

between the egress time of the shortest path configuration and the RCSO configuration increases at a very fast rate.

5. Results and analysis

As mentioned in the previous section, the study is divided into two parts. The first part searches for the set of values (for the six parameters of the RCSO model) that minimizes the egress time. The second part then compares the egress time of the following configuration models:

1. Minimum egress RCSO
2. Normal RCSO
3. Minimum Distance Configuration.









5.1. The minimum egress RCSO

After determining the minimum egress RCSO by running the RCSO algorithm on the network graph and then searching for the parameter values that minimize the egress time, the path usage statistic was also collected for each of the various combinations of values for the length, width and number of agents. In particular, these statistics were collected by first fixing the values of L_1 and L_2 (the lengths of the two edges) as well as the widths W_1 and W_2 in order to define the exact space capacities ($C_1 = L_1 W_1$ and $C_2 = L_2 W_2$) of the edges, and then observing how the ratio of the path usage between the two edges varies with increasing number of agents. Graphing the results on these path usage ratios reveals four different patterns emerging. These four general patterns are shown in Fig. 6.

In these graphs, the x-axis is for the number of agents while the y-axis captures the ratio of the path usage. The blue and red plots correspond to the path usage ratios of the two edges. Since there are only two edges in the network, the plots exhibit symmetry with respect to the line $y = 0.5$ (the mean ratio).

5.1.1. Pattern 1: One line

The first pattern revealed involves a straight line. Interestingly, this pattern emerges if and only if the dimensions of the two edges are exactly equal (i.e., $L_1 = L_2$ and $W_1 = W_2$). Technically, the pattern is not really a straight line, especially on the left part of the plot where there seems to be a large variation. This is actually due to the sensitivity brought about by the small number of agents. For example, if there are only three agents, then the best assignment would one edge having 2/3 path usage ratio while the other having only 1/3.

One line	A line with interlinked branches	Two lines	Two lines with interlinked branches
			
$c1=20, c2=20$	$c1=60, c2=60$	$c1=100, c2=50$	$c1=100, c2=140$
			
$c1=50, c2=50$	$c1=48, c2=48$	$c1=400, c2=200$	$c1=60, c2=120$

$c1=100, c2=100$

$c1=96, c2=96$

$c1=100, c2=110$

In general, if the number n of agents is even, then each of the edges will have exactly $n/2$ agents passing through it, to yield a usage ratio of 0.5 for both edges. However, if n is odd, then there will be exactly $(n + 1)/2$ and $(n - 1)/2$ agents passing through the first and second edges respectively. These yield ratios of $(n + 1)/$

$(2n)$ and $(n - 1)/(2n)$ which approximate 0.5 when n is sufficiently large. Thus, the plot of the path usage ratio approximates that of a line on the mean ratio.

The first pattern strongly demonstrates the load-balancing phenomenon observed in RCSO models. Whenever the two edges have

equal lengths and equal widths, then even if the number of agents increases, the ratio is still (approximately) 0.5 for each of the edges. Note that the actual optimal configuration for this type of network graph has the same ratio values!

5.1.2. Pattern 2: One line with interlinked branches

Here is where it becomes more interesting. There were plots where the two edges have the same ratios of 0.5, until the number of agents reaches a certain threshold. After the threshold, the path usage ratio for one edge starts to increase, while that for the other edge starts to decrease with increasing agent numbers. Two questions come to mind:

- (1) What is the explanation for this behavior in the plot?
- (2) Can the threshold value be predicted?

The behavior where the two edges have the same ratios corresponds to the “filling-up” period. Each of the two edges during this period can still accommodate more agents, and thus receive agents at approximately the same rate. At the threshold point, the two edges are filled up completely, and then afterwards one edge starts to release agents at the other end already.

The two edges are filled up (almost) simultaneously even though they have different lengths and widths because their space capacities (length \times width) are the same. This concept also helps answer the second question. To determine the threshold point, we only need to compute the maximum total number of agents occupying the two edges. Since the length and width values in the experiments are in linear units, then multiplying this with the maximum density for a space will provide the threshold value. The actual formula for the threshold is therefore given by equation

$$\text{threshold} = (L_1 \cdot W_1 + L_2 \cdot W_2) \rho_{\max} \quad (6)$$

where L is the length, W is the width, and ρ_{\max} is the maximum density.

In our numerical experiments, we use meters for the length and widths of the edges. Our model also assumes that the maximum density is four persons per square meter. The computed threshold in each configuration agrees with that found in each of the plots. To account for the divergence in the plots we give the following explanation.

At the time when the threshold is reached, the two edges are completely filled up. The shorter edge, however, will start to release agents sooner than the other edge, thus becoming available again for accepting a few agents. Consequently, more agents pass through this edge than in the other. Until the longer edge starts to release agents, incoming agents will use the shorter one. This explains why the gap between the path usage ratios continues to increase immediately after the threshold point.

At some point, the change in path usage ratio for the two edges is inverted. That is, the edge that was previously experiencing an increase in the path usage ratio will start to experience a decrease because of congestion, while the other edge starts to have an increase in the ratio (as a result of the congestion on the other edge).

This next critical point actually corresponds to when the second edge starts to release some agents on the other end, and becomes available for incoming agents. These agents will therefore prefer this (unsaturated) edge over the other (saturated) one.

Numerically, this second critical point can be computed by deriving some pieces of information.

1. First, the time t_{out} it takes for the first agent in the longer edge to come out can be computed by $t_{out} = L_2/u_{init}$ where L_2 is the length of the longer edge and u_{init} is the speed of the first agent that entered this edge.

2. Next, the time it takes for each of the first k agents in the shorter edge can be computed using a similar formula, i.e., $t_k = L_1/u_k$. Then we evaluate the maximum k such that $t_k < t_{out}$. This k value is the second critical point.

Two other important aspects of the second pattern should be emphasized:

1. For a single (base) line to be observed on the left part of the plots, the two edges should have the same space capacity values. This ensures that both have equal attractiveness at the start. If the space capacities are not the same, then there will be two lines on the left part of the plot, as explained in the third pattern.
2. After initially filling up the two edges, incoming agents will have to wait for agents coming out of one of the edges. For each edge, there will be no simultaneous exit (i.e., only maximum one agent exits at a given time). Consequently, each future incoming agent sees exactly $c - 1$ agents in the edge, where c is the space capacity. Thus every such agent will have the same initial speed. Considering this fact on agents of both edges, the increase-decrease pattern in path usage ratios is guaranteed to be rhythmic.

5.1.3. Pattern 3: Two lines

In the cases covered by this pattern, the plots are asymptotic to the two (base) lines at ratios r_1 and r_2 . Such cases occur *only* when the lengths of the edges are equal, while the widths are unequal. The explanation for this observation is given as follows.

Let l_1, l_2, w_1 and w_2 be the lengths and the widths of the two edges, with $l_1 = l_2$, and, without loss of generality, let $w_1 < w_2$. If we let $k = w_2/w_1$ be the ratio of the two widths, then it follows that the ratio of the space capacities of the two edges is also equal to k . That is, $c_2/c_1 = k$.

To simplify the discussion, assume first that k is an integer. At the beginning, both (empty) edges are equally attractive, in terms of their interaction value $I_{e,t}$. The first edge gets the first agent, making it less attractive after. Succeeding k agents will select the second edge, after which the interaction values for the two edges would have equalized again. In other words, for every agent that goes to the first edge, k succeeding agents choose the second edge. This leads to the following values for r_1 and r_2 .

$$r_1 = \frac{1}{k+1}, \quad r_2 = \frac{k}{k+1} \quad (7)$$

Writing these in terms of the space capacities, then since $c_2/c_1 = k$, we have the following results.

$$r_1 = \frac{c_1}{c_1 + c_2}, \quad r_2 = \frac{c_2}{c_1 + c_2} \quad (8)$$

In the case that k is not an integer, the plots are just not as straight (since ceiling values have to be accounted for) but are still asymptotic to the computed r_1 and r_2 values.

5.1.4. Pattern 4: Two lines with interlinked branches

Cases for this type actually combine properties from the two previous cases. In particular, they cover the cases when the lengths are unequal (Pattern 2), and at the same time the space capacities are unequal (Pattern 3).

At the start of the simulation, the phenomenon observed in pat-

point at which the branching starts to occur is given at the point when the number of agents is $n = c_1 + c_2$, the sum of the space capacities of the two edges. As in the second case, the interlinking is periodic (sometimes covering several increase–decrease combinations). The reason for this is after the edges have been filled up, the succeeding agents that successfully enter a given edge will have exactly the same velocity as the last agent that entered the same edge.

In summary, the four patterns may be described in four quadrants, determined by the relationship between the lengths and also the space capacities between the two edges, as illustrated in the table below (See Table 1).

The summarized information clearly shows that the usage is not only dependent on the distance (which the shortest path configuration assumes as the only factor), but also on the space capacity. This is the main reason why, as will be shown later, the normal RCSO is better than the shortest path configuration.

It should be noted further that there are four different patterns for the simple network in this study because in each of the two factors (i.e., length and capacity), there are two possibilities: equality and inequality.

If we extend the network to one with three edges from the source to the sink, then there will be more scenarios for each of the factors. In particular, for the length, if L_1, L_2 and L_3 are the measurements for the edges, then, without loss of generality, there are four possible conditions:

1. $L_1 = L_2 = L_3$
2. $L_1 = L_2 < L_3$
3. $L_1 < L_2 = L_3$
4. $L_1 < L_2 < L_3$

All other possible configurations would be equivalent to one of these. For example, the condition $L_1 = L_2 > L_3$ is actually equivalent to the 3rd item in the above list since the labeling of the edges may be interchanged. The four possible cases imply that the number of path usage patterns is $4^2 = 16$.

Similarly, for four edges connecting two nodes (the source and the sink), the number of possible cases are 2^3 , as shown below, and therefore the number of possible patterns is $(2^3)^2 = 64$.

1. $L_1 = L_2 = L_3 = L_4$
2. $L_1 = L_2 = L_3 < L_4$
3. $L_1 = L_2 < L_3 = L_4$
4. $L_1 < L_2 = L_3 = L_4$
5. $L_1 = L_2 < L_3 < L_4$
6. $L_1 < L_2 < L_3 = L_4$
7. $L_1 < L_2 = L_3 < L_4$
8. $L_1 < L_2 < L_3 < L_4$

It can be generalized that for a network on two vertices with n edges connecting the source to the sink, there would be exactly $(2^{n-1})^2$ possible patterns. This can be derived by fixing the order of the edges. Then there are $n - 1$ possible adjacent binary relations, each with two possibilities (i.e., '=' and '<'). Thus there are 2^{n-1} possible length configurations. Similarly, there are 2^{n-1} possi-

ble capacity configurations. The number of possible patterns is therefore $(2^{n-1})^2$.

Determining the number of possible patterns further becomes a lot more complex when a path from a source to a sink involves more than 1 edge, and especially if different paths overlap (i.e., there is a common edge). In any case, what has been clearly established in this part of the study is that to achieve the minimum egress time for a given network, the distance should not be the only factor; the space capacity of each edge is also a very important consideration.

5.2. Egress time comparisons

Before the egress times of the three different configurations are analyzed, we first examine the average speeds of the agents in a specific edge. Fig. 7 shows a revealing result.

Observe that for the minimum egress RCSO, all the agents have the same speed, which is the maximum possible speed of 1.2 m/s. This will lead to the (theoretical) minimum egress time. However, this is also clearly not possible since it violates the flow–density relationship. The ordinary RCSO yields a more realistic flow–density diagram, and is further shown to be superior to the shortest path configuration.

Showing now the actual egress time, which is the main focus of this study, we see an interesting pattern as well (see Fig. 8).

There appears to be jumping levels, unlike the expected linear increase in egress time. Such behavior actually reflects the

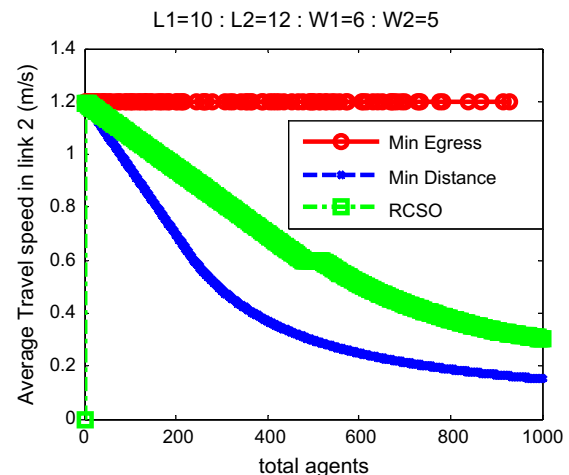


Fig. 7. Comparing the average speed of the agents in the three configurations.

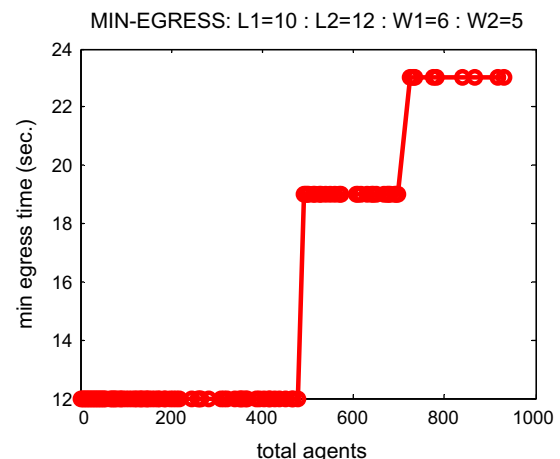


Fig. 8. Total agents vs. egress time for the minimum egress RCSO model.

Table 1
Four basic patterns summarized in quadrants.

	Equal capacities	Unequal capacities
Equal lengths	Pattern 1: One line	Pattern 3: Two lines
Unequal lengths	Pattern 2: One line with interlinked branches	Pattern 4: Two lines with interlinked branches

n of the model used, since an agent at a given discrete step is only allowed to enter an edge if that edge is not yet occupied because the agents' speeds are not affected by the density (inferred from Fig. 7), then if c is the space capacity of the edge, c agents are allowed to pass through that edge synchronously from start to end.

The second level in the plot corresponds to the next batch of agents synchronously passing through that edge. If there are b batches of agents, then the plot will have exactly b level portions. The division of the algorithm can have the agents not simultaneously processed in a single discrete time-step. If lag time between agents is placed, then the plot would be more consistent with the theoretical egress time, as shown in Fig. 9.

Finally, we compare the results of the egress times from the different models. The plot for this is shown in Fig. 10.

In the plot, it is clear that ordinary RCSO yields better egress times than the shortest path configuration. Although the egress time of the ordinary RCSO is bigger than that of the minimum egress RCSO, it still gives much improvement compared to the shortest path configuration. In fact, the gap in the performance of ordinary RCSO compared with the shortest path generally increases with increasing crowd size.

An important implication of this last result is that when evacuation facilities are not too crowded, then the shortest path configuration may yield good egress times (i.e., near the optimal egress time). However, as the size of the crowd increases, such configuration leads to egress times that are too far from the theoretical

optimal. Ordinary RCSO, in this case can approximate the theoretical optimal much better.

6. Conclusion

This study aims to simulate optimum egress time. By running SPSA with Tabu list to find the best parameter set for the (minimum egress) RCSO, a theoretical lower bound of the egress time was determined. For such configuration, it was observed that there are four emerging patterns on the relative path usage. More importantly, the emerging relative path usage is a function mainly of two edge attributes: the length and space capacity. This gives us hints that the shortest path need not always be the preferred evacuation route because it fails to consider the other important edge attribute (space capacity).

Comparing the egress times of ordinary RCSO and shortest path configuration reveals that the ordinary RCSO approximates the minimum egress configuration better. RCSO relates to pedestrians adapting their behavior to local conditions whereas static evacuation maps provide a snapshot view of the routes/exits. One practical application of this result is to revise egress-helping items, such as evacuation maps, in order to encourage an evacuation that follows ordinary RCSO instead of the simple shortest path. For instance, instead of displaying only the shortest path in an evacuation map, a primary exit route and a secondary exit route may be displayed. Alternatively, the shortest path in some of the maps may be replaced by slightly longer (but more spacious) paths so that congestion on some paths may be decreased, leading to the emergence of an RCSO-like evacuation behavior.

When multiple routes are displayed on evacuation maps, the occupants can have options to alter their route choice. However, the exercise of such options by the occupants is not guaranteed. It is also not guaranteed that large crowds will follow the routes suggested on evacuation maps. It is not yet clear how influential the "following the crowd" behavior is against evacuation map information. Such possible deviations should be addressed in some further studies.

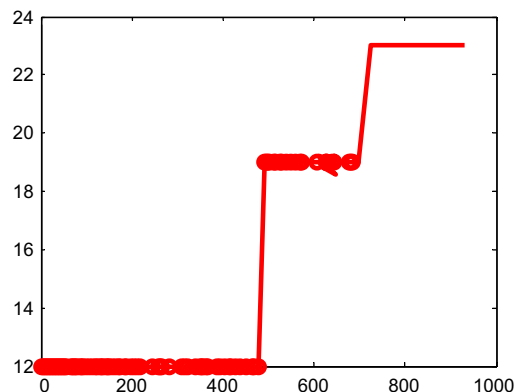
As a final note, the insights gathered in this study resulted from a simple network. It was shown that, as the number of edges increases, the number of possible path usage patterns increases exponentially. It would be very interesting to find some emerging patterns for these as well. Another direction, therefore, for further studies can investigate some major patterns for more complex networks.

Acknowledgements

This research was funded by the Loyola Schools of the Ateneo de Manila University. The authors are grateful to the anonymous reviewers and Prof. Ed Galea for their useful comments and corrections.

References

- Blue, V.J., Adler, J.L., 2000. Cellular automata microsimulation of bidirectional pedestrian flows. *Transportation Research Board* 1678, 135–141.
- Dafermos, Stella.C., Sparrow, F.T., 1969. The traffic assignment problem for a general network. *Journal of Research of the National Bureau of Standards* 73B, 91–118.
- Dial, R., 1971. A probabilistic multipath traffic assignment model which obviates path enumeration. *Transportation Research* 5 (2), 83–111.
- Florian, M., Mahut, M., Tremblay, N., 2001. A hybrid optimization-mesoscopic simulation dynamic traffic assignment model. In: *Proceedings of the 2001 IEEE Intelligent Transport Systems Conference*, pp. 120–123.
- Greenshields, B.D., 1934. A study of traffic capacity. *Proceedings of Highway Research Board* 14, 448–477.
- Glover, F., 1989. Tabu Search – Part I. *ORSA Journal on Computing* 1 (3), 190–206.
- Helbing, D., Molnár, P., 1995. Social force model for pedestrian dynamics. *Physical Review E* 51, 4282–4286.



- Henderson, L.F., 1974. On the fluid mechanic of human crowd motions. *Transportation Research* 8, 509–515.
- Hoogendoorn, S.P., Bovy, P.H.L., 2004. Pedestrian route-choice and activity scheduling theory and models. *Transportation Research Part B: Methodological* 38 (2), 169–190.
- Kachroo, P., Ozbay, K., 1999. *Feedback Control Theory for Dynamic Traffic Assignment*. Springer-Verlag, London.
- Kretz, T., Schreckenberg, M., 2006. F.A.S.T.- Floor field and agent-based simulation tool. In: *Proceedings of International Symposium of Transport Simulation*.
- Lovas, G.G., 1994. Modeling and simulation of pedestrian traffic flow. *Transportation Research* 28B, 429–443.
- Schadschneider, A., 2001. Cellular automaton approach to pedestrian, dynamics-theory. In: Schreckenberg, M., Sharma, S.D. (Eds.), *Pedestrian and Evacuation Dynamics*. Springer, Berlin.
- Schneider, V., Könnecke, R., 2010. Egress route choice modelling – concepts and applications. In: Klingsch, W., Rogsch, C., Schadschneider, A., Schreckenberg, M. (Eds.), *Pedestrian and Evacuation Dynamics*, 2008, Springer, Berlin.
- Spall, J.C., 1998. An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins APL Technical Digest* 19, 482–492.
- Teknomo, K., Millonig, A., 2007. A navigation algorithm for pedestrian simulation in dynamic environments. In: *Proceeding of the 11th World Conference on Transport Research (WCTR)*, University of California, Berkeley, USA, June 24–28, 2007.
- Teknomo, K., 2008. Modeling mobile traffic agents on network simulation. In: *Proceeding of the 16th Annual Conference of Transportation Science Society of the Philippines (TSSP)*, Metro Manila, Philippines, September 19, 2008.
- Teknomo, K., Bauer, D., Matyus, T., 2008. Pedestrian route choice self-organization. In: *Proceeding of the 3rd International Symposium on Transport Simulations*, Queensland, Australia August 6–8th, 2008.
- Tolujew, J., Alcalá, F., 2004. A mesoscopic approach to modeling and simulation of pedestrian traffic flows. In: *Proceedings 18th European Simulation Multiconference*.
- Tubbs, J.S., Meacham, B.J., 2007. Egress design solutions– a guide to evacuation and crowd management planning. *ARUP* 55, 55–56.