12-2022

Simulating Emergency Evacuation Response in an Auditorium Space

Anna Lee

Simulating Emergency Evacuation Response in an Auditorium Space

A thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Industrial Engineering with Honors

By

Anna Lee

Fall 2022

University of Arkansas
Department of Industrial Engineering

Thesis Advisor: Manuel D Rossetti, Ph.D.

Committee Member: Kelly M. Sullivan, Ph.D.

Abstract

The successful execution of emergency evacuations is very important for the protection of the public. Some emergency events, such as fires, can occur with very little warning and turn into a dangerous situation in less than a minute. With high population densities, universities have increased risk involved with evacuations. One specific area that presents high risk is auditorium spaces such as lecture halls with high densities combined with added barriers such as tables and chairs. The ability to assess a building's emergency preparedness is necessary for keeping the public safe. Simulation is a way to conduct a theoretical event and gain insight into the effects of the event in a safe, no-impact environment. This research seeks to use simulation modeling to assess the rate at which individuals travel through an evacuation process in an auditorium space and the impact capacity has on evacuation time.

Table of Contents

1. Introduction

1.1 Background

Emergency events can happen suddenly at any moment or at any place. Fires can grow from small, manageable flames to out of control in as little as 30 seconds [1]. Rapid response is necessary to ensure the safety of everyone involved in a fire emergency or similar high-risk situations. University campuses are uniquely susceptible to emergency events due to the high-density student population [2]. One campus location with particularly high risk is lecture halls. These spaces are designed to hold the greatest number of students possible while also allowing for individual room for access to class materials.

There are two designs a lecture hall will usually have. The first consists of long tables with multiple students sitting at each table in moveable chairs. The second common design has stationary chairs that students sit in with tablet arms that serve as small desks for school supplies. Both have various advantages and disadvantages, but the second design allows for a higher room capacity and is most similar to other auditorium spaces not used as classrooms. For these reasons, this design will be used as the basis for this research.

There are three common steps in an evacuation process [3]. First, there is the detection and alarming of the emergency. Secondly, a person will process the alarm and make decisions to prepare themselves for action. Finally, that person will take actions to ensure their safety. The final step is what takes the longest amount of time in the evacuation of an auditorium space. It can be highly varied depending on the distance from the seat to the exit and the number of people acting as barriers to the exit. Understanding how these two factors impact evacuation time is important in designing auditorium spaces to have a high level of safety even with a high capacity.

1.2 Literature Review

The topic of emergency evacuations is a highly researched one. Multiple publications have been written to try to address or understand the issues that arise in evacuations. Designing high-density spaces in accordance with standard safety regulations reduces the amount of risk that comes with random emergency events. To test layout designs and safety regulations, simulation has been a go-to method. Simulation has allowed evacuation procedures to be assessed in a no-impact environment for low costs [4].

Discrete-event and agent-based modeling have been the common simulation methods for emergency evacuations. Agent-based modeling is a branch of the more common discrete-event simulation and is agent behavior driven whereas discrete-event modeling is process-driven [5]. The goal of this research is to observe and understand an evacuation process in an auditorium space and gain insights. For this goal, a discrete-event simulation model was concluded to be the best option to produce the most impactful results while not creating an overly complex model. The Arena simulation software was chosen for this research due to it being widely used in the simulation community [6] and the author's level of knowledge utilizing the software.

2. Conceptual Model

The primary focus of this research is to understand the risks of having a large number of people in an auditorium space when in need of an evacuation. The model designed will simulate the movement of individuals from the moment people start to move to the last person leaving the auditorium. The space used in the model was influenced by a common lecture hall design as shown below (Figure 1).

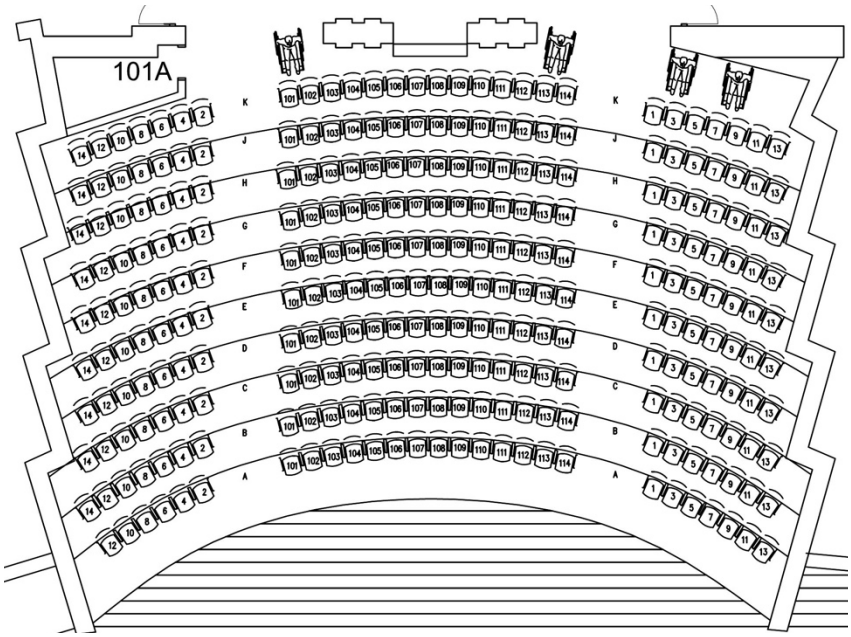

Figure 1. A widely used lecture hall layout. [7]

The layout was then redesigned as a rectangular grid, with each area a person could occupy represented as a cell (Figure 2). These cells are created using resources in the simulation and can be seized and released by entities. Each cell is approximately 4 - 6ft^2, although exact measurements are not assigned because the amount of space a person occupies varies slightly. Each cell is considered 1 unit. The grid space contains 3 primary different cell types: the seating cells, the aisles or stairs, and the free-walking cells. Since the simulation represents a random

emergency event and is starting at the moment people begin to move from their starting location, everyone must start out in seating cells, then they are able to move into whichever type of cell they choose. The grid is also a coordinate system, with the x-values starting in the lower left-hand side and increasing horizontally. The y-values also start in the lower left-hand side but increase vertically. One of the goals of the model is to have a dynamic layout that can be changed to the preference of the user. The base model has a maximum area dimension of 40x20 cells. At the beginning of the simulation, the size of the room and each section is defined and initialized.

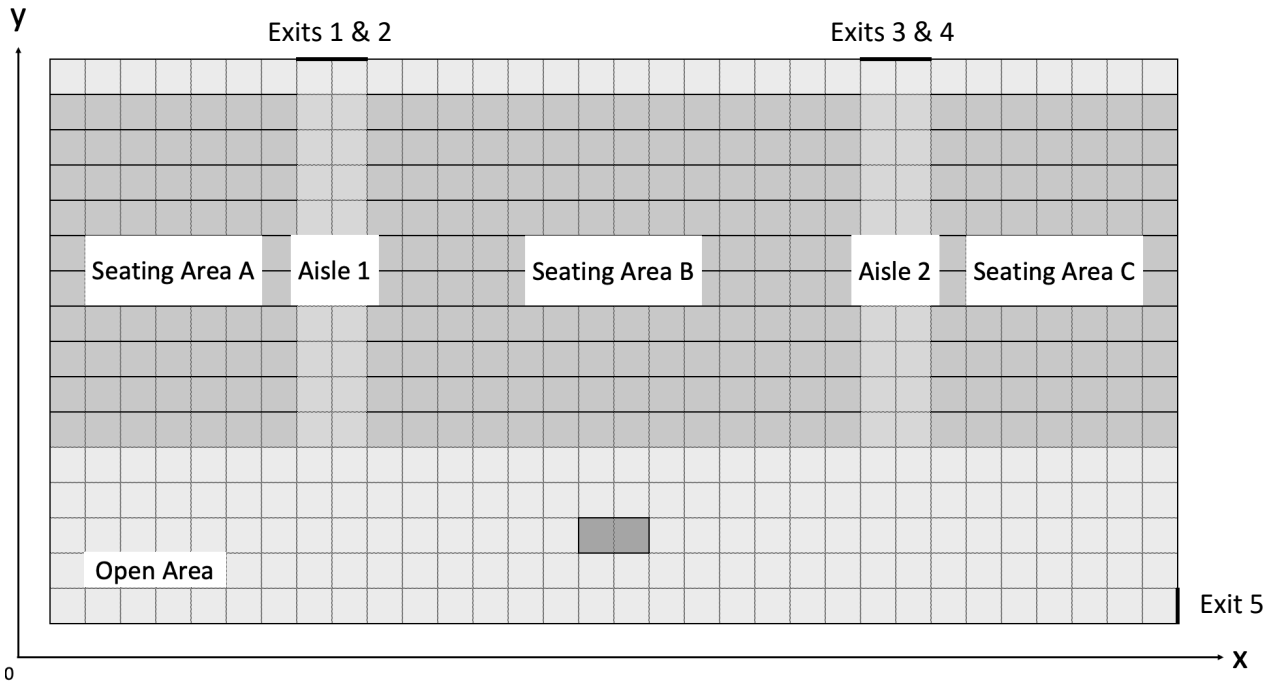

Figure 2. A grid layout of a lecture hall.

As individuals travel from their starting seat to an exit, they move from cell to cell. To move to a new cell, a person has the ability to choose between the 8 adjacent cells of their current cell and to transfer to the next cell. Each of these cells is represented as zones. To determine which cell to travel to, an entity must first find the direction it needs to travel towards

the exit. Once the direction angle is found, the zone that the angle falls into is the zone that the entity wants to travel to (Figure 3).

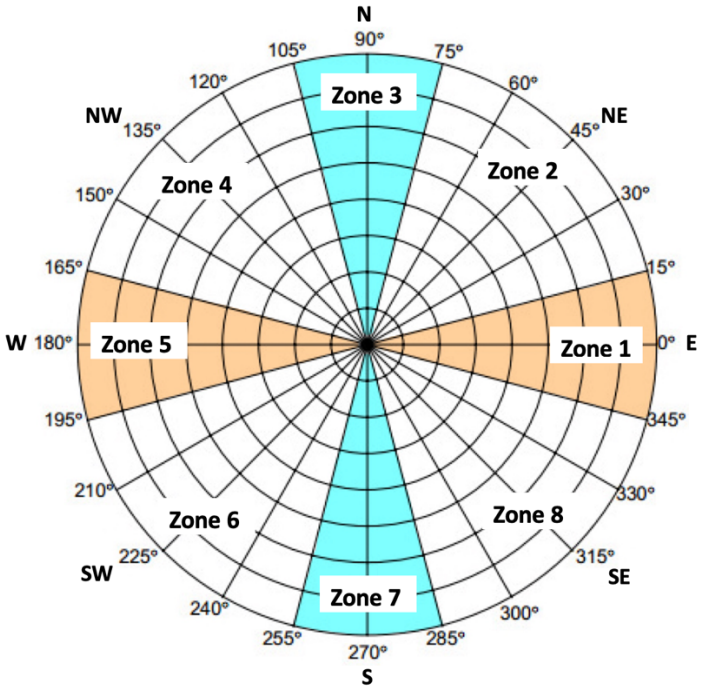

Figure 3. The zones and the direction angles they represent

Within each cell type, the potential movement of a person changes. Within the rowed seating area, people can only travel left or right towards the nearest aisle, having to wait if the cell they want to travel to is occupied by another person. Within the aisles, a person can only travel up or down. In the free area, a person can travel to any of the 8 cells directly adjacent to their current cell. If the cell a person wants to travel to is occupied by someone else, they will either wait for the space to become free or find an alternative route. The blocked cells are barriers that a person cannot travel through; the primary example of this is desks at the front of the room. Figure 4 below depicts a flowchart of the movement a person will make from the start of the evacuation until they leave the room.

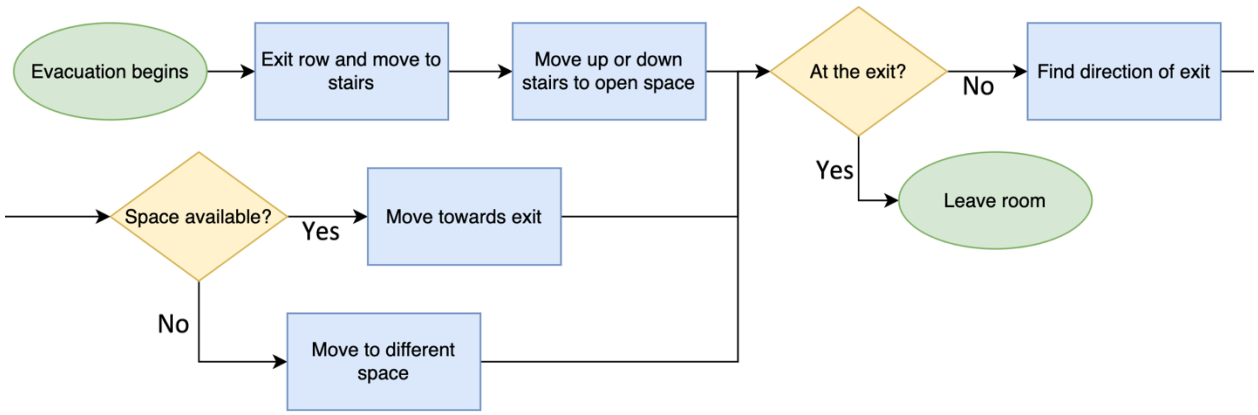

Figure 4. A flowchart of movement in an evacuation

3. Simulation Model

Using Arena simulation software, a discrete-event simulation model was built to represent an evacuation event in an auditorium space. From this research, a user will be able to analyze the evacuation effectiveness of auditorium spaces of various sizes using a base model with dynamic input parameters.

3.1 Input parameters

At the beginning of the model, two Excel files are read in, one containing layout sizing parameters of the grid space and the second containing all the resource coordinates. The purpose of these files is to allow the end user to adjust what size to make the auditorium space. The base model is designed with maximum values of x = 40 and y = 20, with the input parameters scaling down those values as seen fit.

Table 1. Layout parameters that can be changed before the initialization of the model.

Parameters	Values	Area	X value range
Minimum X	1	Section A	1 - 7
Maximum X	32	Section B	10 - 23
Minimum Y	1	Section C	26 - 32
Maximum Y	16	Aisle A	8 - 9
Y Section/Aisle Range	6 - 15	Aisle B	24 - 25

The file layoutdat.csv is read by one entity at the beginning of the simulation. It reads in 16 variables that adjust the size of the overall model as well as the seating sections and aisles (Table 1). Having many different variables read in by a file allows for more control over the design of the model and allows for quicker changes. This file is read first because the sizing of the overall model and seating areas impacts which resources are used for seating and which entities can be initialized into the model.

Table 2. The two attributes depicting a location and their corresponding resource.

myCurrentX	1	1	1	1	1	…	40	40	40
myCurrentY	1	2	3	4	5	…	18	19	20
Resource	r11	r21	r31	r41	r51	…	r1840	r1940	r2040

The cell that an entity or person occupies at any point during the simulation is controlled by two attributes, myCurrentX and myCurrentY, which represent the x and y-coordinates of a cell in the grid respectively. At the start of the simulation, an entity for each cell of the base grid is created, 800 in total. The file resourcedat.csv is read by each entity and assigns each a value for myCurrentX and myCurrentY (Table 2). These attributes then correspond to a resource that the entity claims. Each entity will have a unique combination of x and y, resulting in only one entity holding each resource.

An additional input that a user can define is the number of people that run through the simulation. The input can either be a number or a distribution, as the number of people in an auditorium space will typically fall into a range rather than a set known number. As the layout is parameterized and seating sections are defined, the maximum number of entities that can start within seating and therefore start in the simulation, changes. The number of people to be added to the model is initialized as Uniform(50, 250).

3.2 Model Statistics

There are 2 main statistics recorded from the model. The first is the time it takes for each person to leave the room; it is recorded after the model is initialized and the entity starts traveling through the movement modules to when the entity is disposed. The goal of recording this metric

is to learn the fastest and slowest exit times and the average exit time for all entities. From this statistic, the probability that all people will exit the room within a specified amount of time can also be recorded and analyzed.

The second statistic recorded is the average time that it takes to exit the room from every seat. The goal of this is to understand which seats have a higher risk associated with choosing to sit in them due to the distance or length of time it takes to get to any exit door.

3.3 Model Overview

The overall model can be broken down into multiple subparts that run consecutively. The first is the setup of the simulation which involves the parameterizing of the layout and reading in of resources. The second is the movement of people from rows to aisles and then from aisles to free space. The third and final is the determining of direction towards the exit and movement towards the exit.

3.3.1 Simulation Setup

To prepare the model to work with any layout design, many resources, attributes, variables, and expressions must be defined before it is run. The foremost of this is the grid that represents the auditorium space that people or entities travel through to exit. This grid is created using 800 resource elements that each represent a cell. Each resource is then a part of an arrayed expression, eGridResources. Within Arena, double arrayed expressions are read row number first, then column number. Resource elements in an expression are placed in the position of their corresponding coordinates, as seen in Figure 5.

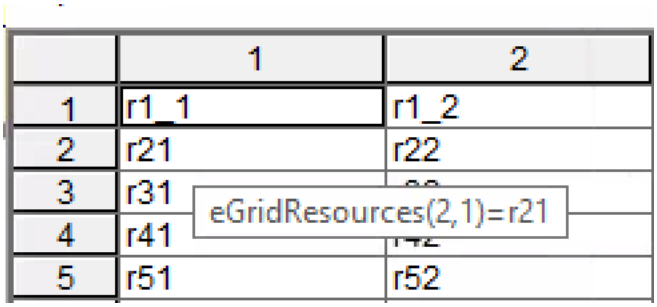

Figure 5. The expression eGridResources and some of its elements.

At the start of the model, two CREATE modules are used to generate entities. The first, as shown in the pseudo-code in Figure 6, creates one entity that reads in the input variables to modify the layout. It then travels through a while loop that picks the entities to be initialized into the grid to act as people. The entity loops through lines 7 – 11 for every person that is to be initialized in the model. During the loop, it looks through the HoldStart queue (Figure 7) and removes a random entity. That removed entity then travels to seize its starting resource.

```
1    Create 1 initializing entity
2    Begin ReadWrite
            vMinX
            vMaxX
            vMinY
            vMaxY
            vSectionMinY
            vSectionMaxY
            vSectionAMin
            vSectionAMax
            vAisleAMin
            vAisleAMax
            vSectionBMin
            vSectionBMax
            vAisleBMin
            vAisleBMax
            vSectionCMin
            vSectionCMax
3    End ReadWrite
4    Delay for total entity generation
5    Begin Assign
            vNumPeople = ANINT(UNIF(50,250))
            myStartCount = 1
            vExit1X = vAisleAMin
            vExit1Y = vMaxY
            vExit2X = vAisleAMax
            vExit2Y = vMaxY
            vExit3X = vAisleBMin
            vExit3Y = vMaxY
            vExit4X = vAisleBMax
            vExit4Y = vMaxY
            vExit5X = vMaxX
            vExit5Y = vMinY
6    End Assign
7    Begin While
            myStartCount <= vNumPeople
8           Remove entities from HoldStart.Queue
                  ANINT(UNIF(1,NQ(HoldStart.Queue)))
9    Begin Assign
            myStartCount = myStartCount + 1
10   End Assign
11   EndWhile
```

Figure 6. Pseudo-code for initialization the grid layout

The second CREATE module is used to generate 800 entities to represent each cell in the auditorium grid. Each entity is assigned an x and y coordinate value from an input file and from those, a resource. The entity travels through a DECIDE module to determine if the resource it was assigned is in the seating area. If in the seating area, it is placed in a hold queue (line 7 of Figure 7) where it will remain until either the singular starting entity from Figure 6 will remove it and initialize it into the rest of the model, or once the specified number of entities are removed, it will be disposed. If the entity's resource is in a non-seating area it is disposed. It also alters the state of its resource to inactive if outside of the layout's boundaries. After all the simulation setup modules have been fully processed, only entities that represent people exiting the auditorium space will be left to travel through the rest of the model.

```
1     Create 800 people
2     Begin ReadWrite
            myCurrentX
            myCurrentY
3     End ReadWrite
4     Begin Assign
            myVelocity = UNIF(0.4, 0.8333)
            myCurrentRes = eGridResources(myCurrentY, myCurrentX)
5     End Assign
6     Decide with condition
            If ((myCurrentX >= vSectionAMin && myCurrentX <= vSectionAMax) ||
            (myCurrentX >= vSectionBMin && myCurrentX <= vSectionBMax) ||
            (myCurrentX >= vSectionCMin && myCurrentX <= vSectionCMax)) &&
            (myCurrentY >= vSectionMinY && myCurrentY <= vSectionMaxY)
7                 Hold in HoldStart.Queue
            Else
8                 Hold in Dispose.Queue
9                 Decide with condition
                        If myCurrentX > vMaxX || myCurrentY > vMaxY
10                            Alter resource capacity
                                    myCurrentRes = 0
11    Dispose
```

Figure 7. Pseudo-code for initializing all entities as resources

3.3.2 Row and Aisle Movement

Once all of the entities are transferred to the beginning of the movement section or disposed, the entities will begin traveling through the rest of the model, from cell to cell. Before any movement through the grid is done, an entity must first find the exit. There 5 possible exit points an entity can travel towards. Four of them are at the top of the aisles grouped in pairs and the last is in the bottom right-hand corner. For each entity, the distance from its current resource to each exit point is calculated, as seen in line 1 of Figure 8. The exit with the shortest distance is then assigned as the exit for that entity. The only exception to this rule is if the entity is in the first and second rows. If so, it will directly be assigned the fifth exit. Once an exit is assigned, all grid movement of an entity will then be in an effort to reach that target exit.

```
1    Begin Assign
         myDistance1 = SQRT(((vExit1X - myCurrentX)* 2) + ((vExit1Y - myCurrentY) ** 2))
         myDistance2 = SQRT(((vExit2X - myCurrentX)* 2) + ((vExit2Y - myCurrentY) ** 2))
         myDistance3 = SQRT(((vExit3X - myCurrentX)* 2) + ((vExit3Y - myCurrentY) ** 2))
         myDistance4 = SQRT(((vExit4X - myCurrentX)* 2) + ((vExit4Y - myCurrentY) ** 2))
         myDistance5 = SQRT(((vExit5X - myCurrentX)* 2) + ((vExit5Y - myCurrentY) ** 2))
2    End Assign
3    Decide with conditions
         myDistance1 < myDistance2 && myDistance3 && myDistance4 && myDistance5
4            Begin Assign Exit 1
                 myExitX = vExit1X
                 myExitX = vExit1Y
5            End Assign
```

Figure 8. Partial pseudo-code for determining the exit for an entity

After an entity's exit is determined, the entity will begin grid movement. An entity must first travel through its seating row to an aisle and then from the aisle to free movement space, and finally to the exit. The only exception to this is seats on the front row, where entities can travel directly to free movement space.

```
1   Decide with conditions
        If myCurrentX <= vSectionAMax && myCurrentX >= vSectionAMin &&
        myCurrentY <= vSectionMaxY && myCurrentY >= vSectionMinY
2           Assign myNextZone = 1
        Else If myCurrentX <= vSectionBMax && myCurrentX >= vSectionBMin &&
        myCurrentY <= vSectionMaxY && myCurrentY >= vSectionMinY
3           Decide with condition
                If myExitX > myCurrentX
4                   Assign myNextZone = 1
                Else
5                   Assign myNextZone = 5
        Else If myCurrentX <= vSectionCMax && myCurrentX >= vSectionCMin &&
        myCurrentY <= vSectionMaxY && myCurrentY >= vSectionMinY
6           Assign myNextZone = 5
        Else
7           Decide with condition
                If ((myCurrentX <= vAisleAMax && myCurrentX >= vAisleAMin) ||
                (myCurrentX <= vAisleBMax && myCurrentX >= vAisleBMin)) &&
                (myCurrentY <= vSectionMaxY && myCurrentY >= vSectionMinY)
8                   Decide with condition
                        If myCurrentY > myExitY
9                           Assign myNextZone = 7
                        Else
10                          Assign myNextZone = 3
                Else
11                  Assign Direction
```

Figure 9. Pseudo-code for seating and aisle movement

In a seating section, an entity is only able to travel left (zone 5) or right (zone 1) towards the aisles (lines 1 – 6 of Figure 9). Entities cannot travel between rows of seats because the seats act as barriers and it is assumed that people cannot travel over them. Once in an aisle or on the stairs, an entity will then only travel up (zone 3) or down (zone 7) depending on the location of the exit cell. If the exit is at the top, the entity will travel directly from aisle to exit. If an entity is traveling to the exit at the bottom of the grid, it will enter free movement space after the bottom of the aisle.

3.3.3 Exit Movement

Once an entity is no longer in the seating or aisle area it has an expanded range of movement. An entity can move to any of the 8 adjacent cells from its current cell. The direction an entity needs to travel towards to reach the exit is first determined. Once found, the zone that the angle falls

into is considered the preferred zone (Figure 3). To capture the randomness of human movement, once the preferred zone is picked, the actual zone that the entity will move to is picked using discrete probability distributions within the expression, eZoneCDF (Figure 10). These distributions have a high probability of choosing the preferred zone and a small probability of choosing adjacent zones.

1	DISC(0.8,vEast, 0.9, vNE, 1, vSE)
2	DISC(0.6, vNE, 0.8, vNorth, 1, vEast)
3	DISC(0.8, vNorth, 0.9, vNW, 1, vNE)
4	DISC(0.6, vNW, 0.8, vWest, 1, vNorth)
5	DISC(0.8, vWest, 0.9, vSW, 1, vNW)
6	DISC(0.6, vSW, 0.9, vSouth, 1, vWest)
7	DISC(0.8, vSouth, 0.9, vSE, 1, vSW)
8	DISC(0.6, vSE, 0.8, vEast, 1, vSouth)

Figure 10. Discrete probabilities for every zone

Once the next zone is picked, the entity must check to see if the resource of the desired cell to move into is free. If it is, the entity will move into the cell and claim the resource and then release the resource representing their previous cell. If the chosen cell is occupied by another entity or if it is outside of the bounds of the grid layout, then the moving entity will have to find a new cell to move into.

There are two main reasons why an entity would not be able to transfer to the new grid cell and claim its resource. The first is if the entity is along an edge in the grid and the chosen cell to move into is past the edge. The second is if an entity wants to go in a certain direction, but cannot due to a different entity already occupying that space. In both instances, the direction or zone the entity can move in is limited.

To keep an entity from trying to move into a cell that doesn't exist because it is past a wall, an entity determines if it is along a wall before choosing the desired zone. As shown in lines 2 – 8 of Figure 11, if an entity is found to be on a wall, it is statically given a direction to

follow along the wall to make it closer to the exit. One potential issue to arise from this method, is having a build-up of entities along the wall that are unable to move. This is overcome though because the access to wall space within the model is limited, and entities being along a wall in the free movement space is a rare occurrence.

```
1       Decide with conditions
            If myCurrentX <> vMinX && myCurrentX <> vMaxX &&
            myCurrentY <> vMinY && myCurrentY <> vMaxY
2               Find Direction Angle
            Else If myCurrentX == vMinX || myCurrentX == vMaxX
3               Decide If myCurrentY > myExitY
4                   Assign myNextZone = 7
                Else myCurrentY < myExitY
5                   Assign myNextZone = 3
            Else If myCurrentY == vMinY || myCurrentY == vMaxY
6               Decide If myCurrentX > myExitX
7                   Assign myNextZone = 5
                Else myCurrentX < myExitX
8                   Assign myNextZone = 1
```

Figure 11. Pseudo-code for determining if an entity is along a wall

When an entity wants to claim a resource and move into a cell that another entity already has, it has to choose a new cell to transfer to. To do this, the entity checks all adjacent cells and picks one of the free ones to travel into. An entity travels through a while loop for each zone. During the loop, the entity checks the state of the resource from the zone to see if it is idle (line 4 of Figure 12). If open, the entity separates into 2, with the duplicate representing the free cell space and going into a hold. If busy, the entity goes to the end of the loop and either restarts the loop to check the next zone or leaves the while loop. Once all 8 zones have been checked, the original entity will then search the queue of duplicates and randomly select one (line 12 of Figure 12). The zone that the selected duplicate represents will then become the zone that the original entity travels to (lines 13 & 14 of Figure 12).

```
1       Begin While
            myCounter <= vZones
2           Begin Assign
                myPossibleNewX = eNewLocation(myCounter, 1)
                myPossibleNewY = eNewLocation(myCounter, 2)
                myPossibleNewRes =eGridResources(myPossibleNewY, myPossibleNewX)
3           End Assign
4           Decide with condition
                If STATE(myPossibleNewRes) == IDLE_RES
5                   Separate, Duplicate Original
6                       Original, line 10
                        Duplicate
7                           Assign myPossibleZone = myCounter
8                           Hold in Search Zone Queue
9                           Dispose
                Else
10                  Assign myCounter = myCounter + 1
11      EndWhile
12      Search, Search Zone Queue, from 1 To NQ(Search Zone)
            Condition: AQUE(Search Zone, ANINT(UNIF(1, NQ(Search Zone))),
            NSYM(myPossibleZone))
                Return J = myPossibleZone
13      Assign myNextZone = J
14      Begin Assign
            myNewY = eNewLocation(myNextZone, 2)
            myNewX = eNewLocation(myNextZone, 1)
            myNewRes = eGridResources(myNewY, myNewX)
15      End Assign
```

Figure 12. Pseudo-code for selecting an idle resource to claim.

Once an entity picks a new cell to travel to and the resource for that cell is free or idle, the entity then travels into the new cell and leave its old one (Figure 13).

```
1       Begin Assign
            myOldX = myCurrentX
            myOldY = myCurrentY
            myOldRes = myCurrentRes
            myCurrentX = myNewX
            myCurrentY = myNewY
            myCurrentRes = myNewRes
2       End Assign
3       Seize myCurrentRes
4       Delay
            SQRT(((myCurrentX - myOldX)**2) + ((myCurrentY - myOldY)**2)) * myVelocity
5       Release myOldRes
```

Figure 13. Pseudo-code for moving from the current resource cell to a new one

The entity first seizes the new resource cell (line 3 of Figure 13), then delays for the time it takes to walk from the current cell to the new one (line 4 of Figure 13). To realistically simulate the time it takes to travel from one cell to another, the delay time is calculated using the distance formula multiplied by the attribute myVelocity. The distance formula finds the unit distance traveled from an entity's old cell to a new one. The distance will either be 1 or 1.41 units depending on if the cell the entity moved into is next to or diagonal to the original cell. Once in the new cell, the entity releases the old one and assigns the new cell attributes as its current ones. The velocity of each attribute is represented as a uniform probability distribution, UNIF(0.45, 0.8). The bounds of the distribution were found using time studies. 10 people of various heights walked the length of 10 cells multiple times, with each time walking at a different velocity. The fastest and slowest velocities of each person were then averaged to find the values used in the model.

As an entity is traveling through the grid space toward the exit, it will continuously check to see if the cell it is in is the desired exit point. If it is, the entity releases the resource cell and leaves the model. If it is not at the exit, it loops through the movement modules discussed in sections 3.3.2 and 3.3.3 until it finally reaches the exit.

3.4 Verification and Validation

To verify the model and the approach taken to simulate movement in an auditorium space, several iterations were created and tested until the final model worked as intended. The model began as a single entity in a smaller 8x8 grid to confirm the exit movement from section 3.3.3 worked correctly. Once the entity was able to locate and move toward the exit properly, blocked resources were added to test the code from Figure 12. After debugging and verifying that the

entity could move around blockages, more entities were added. From then on, the grid space was expanded to its full size of 40x20 cells, and subparts of the model were added one by one and tested multiple times until the working model was achieved. The dynamic ability was finally tested by changing the input parameters multiple times and checking for any fault in the model.

To validate the model, the exit times produced by the simulation were partially compared to real-world results from a time study. In the time study, participants began at different seats in a lecture hall and traveled to an exit. Times were taken for participants to go directly from their seats to the exit with no obstructions and with partial obstructions caused by a few people in front of them. Since the velocity of each entity in the model was assigned from real-world data, the exit times in the model aligned with those in the time study. Due to the large number of people exiting the auditorium space in the model, a full-scale evacuation was not able to be replicated in a time study. The accuracy of a smaller model to the time study is still assumed to apply in the same way to a larger-scale model.

3.5 Animation

To visualize the model while running, animation was used in Arena to recreate the grid layout. Each resource cell was animated and as the model is run, the cells change color to represent different resource states. Figure 14 below is the animation of the beginning of the model running.

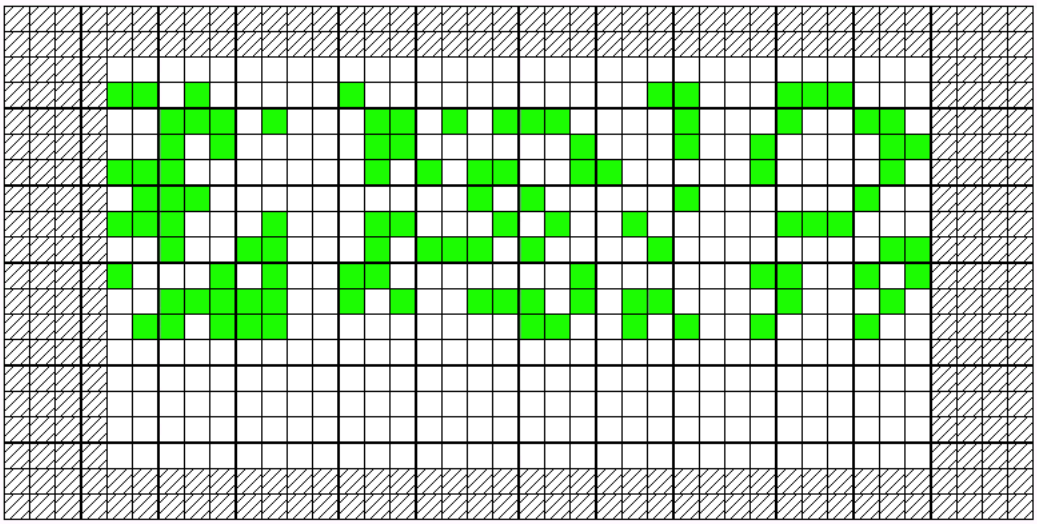

Figure 14. Animation of the beginning of the evacuation

Each of the 800 resources are animated and represented by squares in the grid. Each resource cell is white when idle and an entity can seize it. When the model starts and the input parameters are initialized, the resources that are outside of the specified grid bounds become inactive. Once inactive, those resources can never be claimed by an entity and act as walls in the model. They are the striped squares in the animation. Alternatively, when a resource is seized by an entity, it will turn green. When the resource is released, it turns white again. Also, the entities that are seizing a resource in the grid and moving towards an exit are animated with people icons and will be on top of the resource they are using.

Due to the layout of the grid being dynamic, the aisles and exits are unable to be highlighted. Animation elements are statically created before the model is run and cannot be adjusted mid-run without the use of Excel VBA.

4. Results

The evacuation model was run, and statistics were collected on the time it takes for all entities to leave the simulation and the time it takes to exit from each starting resource cell. The model was run using 25 replications, with each replication having between 50 and 250 starting entities. During each replication, the time it takes for each entity to exit is recorded and the results are shown below (Figure 15).

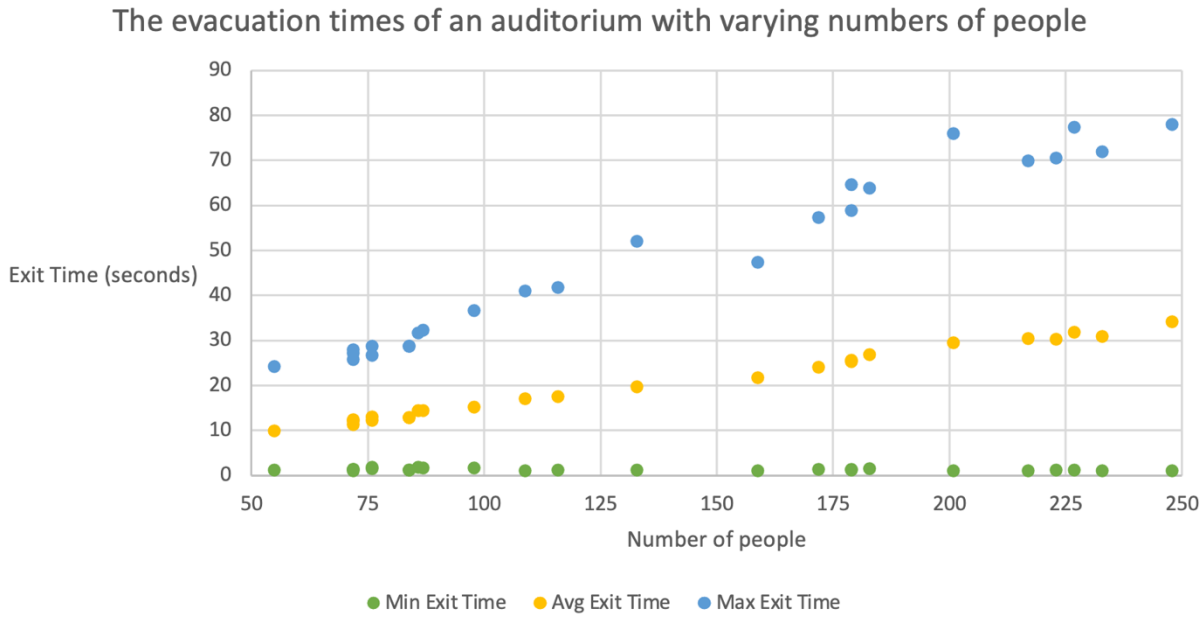

Figure 15. Graph of evacuation times for various amounts of people

The graph shows that the time it takes for the first entity to exit the model remains consistent at around 1 second. As the number of people in the room increases, the total time it takes for all individuals to exit significantly increases. At the lowest occupancy observation with 55 people, the exit time is 24.12 seconds. At the highest occupancy observation, there are 248 people in the room and the total exit time is 77.86 seconds (Table 3). The half-width was 3.11. The growth trends of the average and maximum exit times are observed to be linear.

Table 3. Exit times of the highest and lowest occupancy observations

Number of people	Minimum Exit Time	Maximum Exit Time	Average Exit Time
55	1.09	24.12	9.7
248	0.95	77.86	34.02

These values were then compared across models of different layout sizes. The trends seen in the initial 25 replications were also observed in the new experiments. The first person to exit any simulation consistently left the model around 1 second (unless there were very few entities in the model and they did not start close to an exit, in which case the exit time was slightly higher). The last person to exit left in increasingly longer times dependent on the number of people in the grid. At the maximum capacity with the largest layout size possible of 40x20, the total exit time was 148 seconds, which is approximately 2.5 minutes.

Another goal of simulating an auditorium evacuation is to gain insight into the risk associated with each seat. Seats closer to the edge of the seating sections have a smaller distance to travel to get to the exit and seats in the middle have a farther distance to travel and more potential barriers in the form of people. Even with these observations, it is still unclear which seats impose the most risk to people sitting in them due to the length of time it takes to travel to the exit and leave the room. Tallies are used in the model to track the amount of time it takes to exit the space from every initial resource cell. Using the maximum parameters for the model, a 40x20 space with all seating cells occupied, the model was run with the exit time tracked for each initial seized resource. Figure 16 below shows the resources with the highest and lowest exit times. The rows with the highest exit times are the 3rd and 4th row from the bottom in the middle section. The resources with the fastest exit times are the ones closest to the exits. From

these values, the seat with the longest exit time and subsequently the highest risk is the seat with the coordinates (20, 9). This seat is in the exact center of the middle seating section and the entity occupying that seat must wait for all entities left, right, and above it to exit the model before it can also exit, causing the exit time to be high.

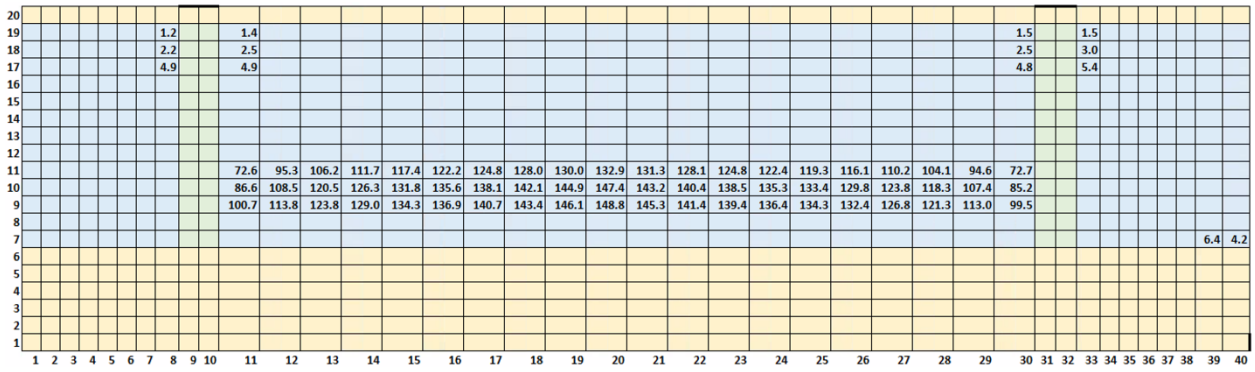

Figure 16. The resources that take the longest and shortest time to exit from

This method was done with other layouts that were smaller and with layouts with less than 100% capacity and the results were the same. The middle of the 3rd and 4th rows of the center seating section produced the longest exit times.

5. Conclusion

Based on the results of this simulation model, having an auditorium space with a high capacity increases the risk involved with an evacuation event. Although large auditoriums are designed to have a high occupancy limit, keeping the capacity low as often as possible increases the efficiency of a possible evacuation. The exit times of the model continued to increase as people were added because each person is an added barrier for others and obstructs the exit path.

Within high-capacity auditoriums, the risk during an evacuation is not equal across all seating areas. Seats in the middle of sections or along walls with no aisle, take a much longer time to get to an exit than seats closer to aisles. The seats with the highest amount of risk are those in the lower rows of the middle section. This is due to those seats having the highest number of possible obstructions in their paths. A way that this model could be expanded on is by allowing entities to change their desired exit dependent on the amount of traffic flowing to each one. This could potentially reduce the evacuation time of the riskiest seats.

From building this simulation model, and completing this research, it has become evident how important this topic is to study. Further expansion on this topic will help increase the safety of auditorium spaces and reduce the negative impacts that come from evacuation events.

References

[1] M. Kobes, I. Helsloot, B. de Vries, and J. Post, “Building safety and human behaviour in fire: A literature review,” *Fire Safety Journal*, vol. 45, no. 1. pp. 1–11, Jan-2010.

[2] R. Parker, National Center for Campus Public Safety, rep., Nov. 2016.

[3] P. Kucera and E. Strakosova, “Assessment of safety evacuation of persons in the design of assembly areas,” *Safety and Security Engineering V*, 2013.

[4] X. Zhang, G. Coates, & X. Ni, (2017). Agent-based Modelling and Simulation for Lecture Theatre Emergency Evacuation. *ISCRAM*.

[5] W. K. V. Chan, Y. -J. Son and C. M. Macal, "Agent-based simulation tutorial - simulation of emergent behavior and differences between agent-based simulation and discrete-event simulation," Proceedings of the 2010 Winter Simulation Conference, 2010, pp. 135-150, doi: 10.1109/WSC.2010.5679168.

[6] M.D. Rossetti (2021). Simulation Modeling and Arena, 3rd and Open Text Edition. Retrieved from https://rossetti.github.io/RossettiArenaBook/ licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

[7] *Neff Hall; Room 101 Seating Chart*, 2021. [Online]. Available: https://www.pfw.edu/offices/special-events/boxoffice/neff101seating.html. [Accessed: 12-Oct-2021].

Appendix

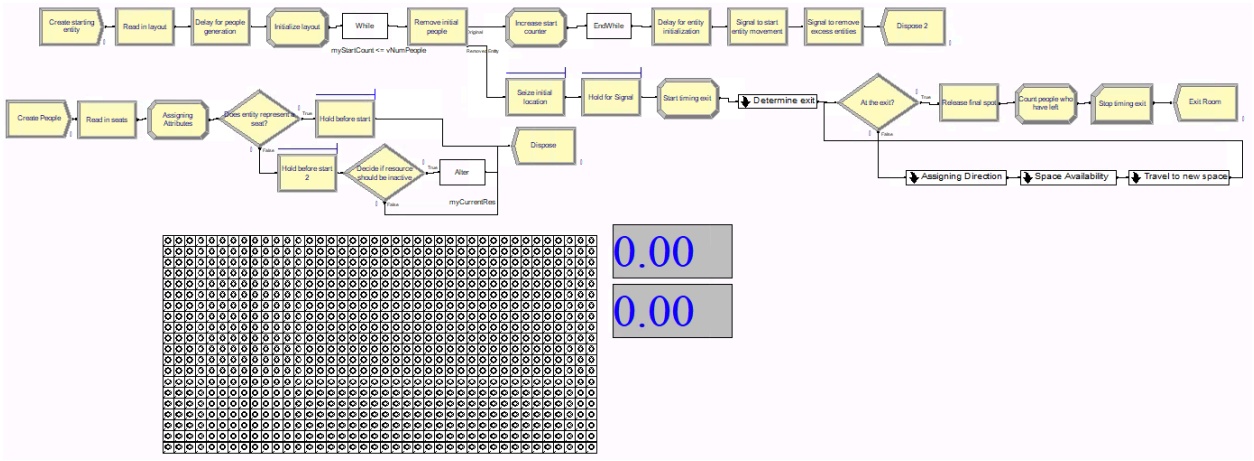

Figure 17. Picture of the entire model in Arena

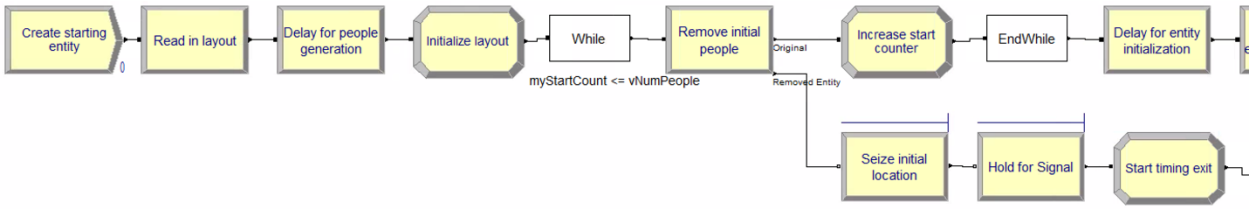

Figure 18. Modules used for initializing the grid layout

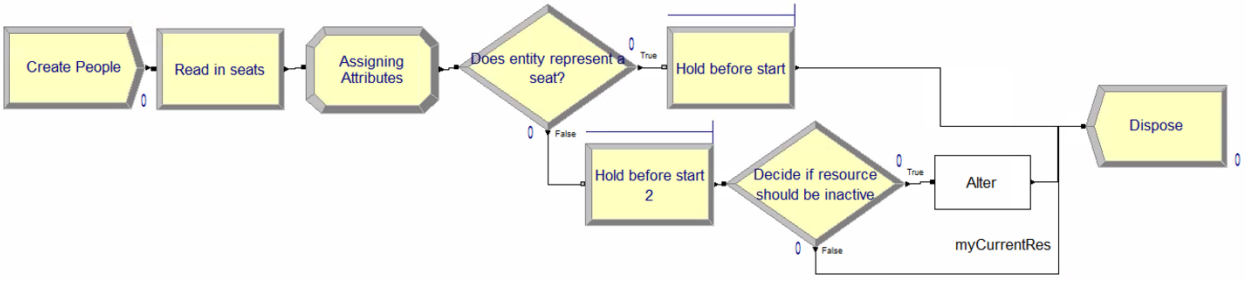

Figure 19. Modules used for initializing resources

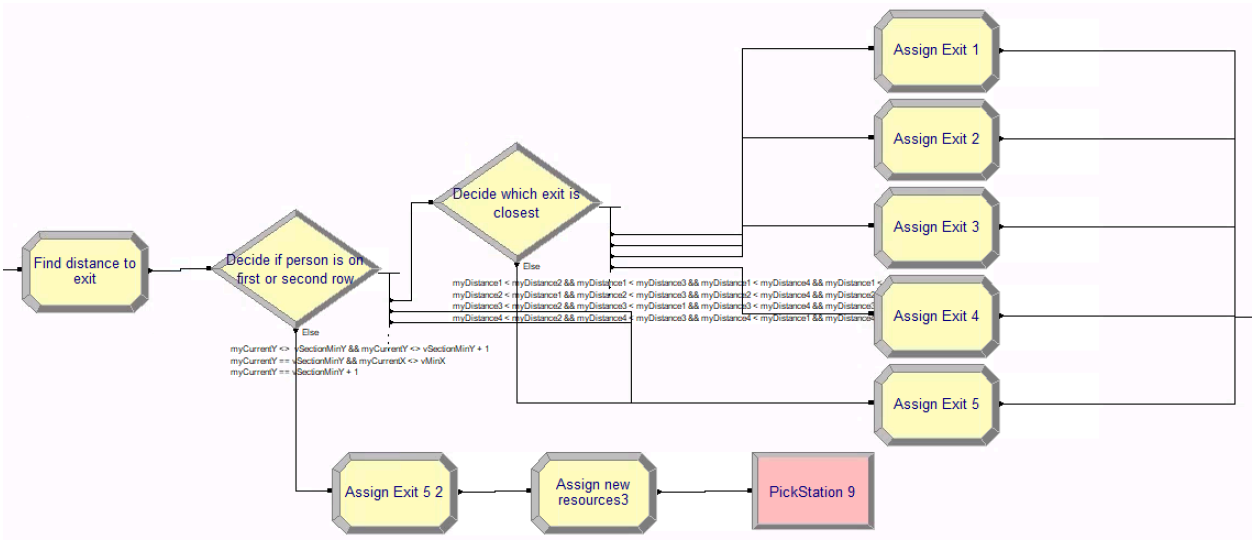

Figure 20. Modules used for determining exit for each entity

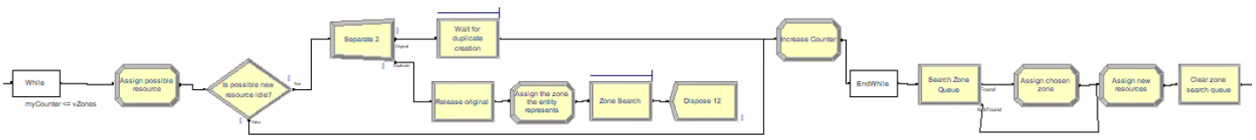

Figure 21. Modules used for determining which cell to move into

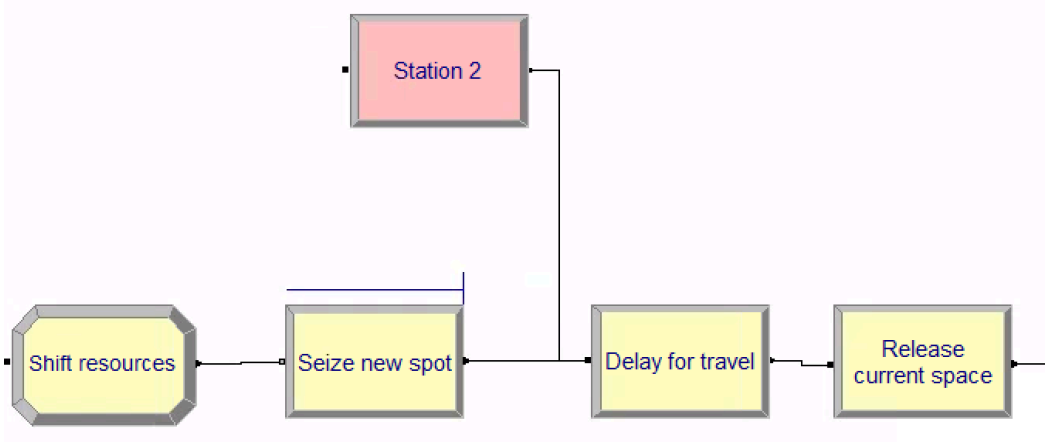

Figure 22. Modules used for transferring from one resource to another

Table 4. Exit times observed during 25 replications

Number of people	Exit Time Min	Exit Time Max	Exit Time Average
109	0.91	40.86	16.86
76	1.39	26.59	12.16
84	0.98	28.55	12.74
227	1.06	77.29	31.66
223	1.11	70.33	30.09
159	0.91	47.19	21.52
72	0.95	27.8	11.2
72	1.18	25.63	11.87
76	1.59	28.61	12.91
183	1.34	63.64	26.74
217	0.95	69.83	30.26
179	1.04	64.42	25.21
248	0.95	77.86	34.02
172	1.27	57.15	23.86
98	1.44	36.54	15.06
84	0.98	28.55	12.74
233	0.95	71.79	30.81
55	1.09	24.12	9.7
133	0.98	51.88	19.49
87	1.48	32.13	14.32
201	0.96	75.78	29.29
116	1.03	41.7	17.35
86	1.61	31.53	14.3
72	1.13	27.06	12.25
179	1.15	58.78	25.39

Table 5. Time study values of velocity over 20 cell spaces

Person	Time 1 (sec)	Time 2 (sec)	Time 3 (sec)
1	9.24	12.32	16.18
2	9.36	13.98	14.62
3	10.92	12.02	14.08
4	10.14	13.66	16.5
5	10.3	12.8	14.68
6	9.42	11.48	15.66
7	7.48	12.46	16.74
8	9.08	13.18	17.66
9	7.16	12.34	17.86
10	7.12	11.34	16.24