

**ASSIGNMENT NO. 2**

**Cell DEVS -Pedestrian**

**Evacuation MODEL**

**Course Title:**

**Methodologies for Discrete Event**

**Modelling and Simulation**

**Course Code: SYSC 5104**

**SUBMITTED TO:**

**Dr. Gabriel A. Wainer**

**Professor, Ph.D.**

**Systems and Computer Engineering**

**Carleton University, Ottawa, Canada**

**SUBMITTED BY:**

**Harshita Ghushe**

**Student ID: 101289614**

**DATE OF SUBMISSION:**

**21/03/2023**

**PART I - Conceptual Model Description:**

An extended cost potential field cellular automaton model for the study of pedestrian dynamics is fundamental to managing pedestrian evacuations and designing pedestrian facilities. This understanding is crucial for developing appropriate evacuation protocols and regulations. It is essential to comprehend the characteristics of pedestrian flow in advance to effectively manage emergency situations. Different physical methods have been used to observe and replicate the dynamic properties of pedestrian crowds, including various self-organizing behaviors.

Simulating the movement of pedestrians during emergencies is challenging when their visibility is affected. Most simulations assume favorable visibility conditions, but in reality, visibility can be reduced by factors such as smoke, fire, power outages, or panic. When visibility is reduced, pedestrians behave differently than they would in ideal conditions. Understanding these behaviors is crucial for accurately simulating pedestrian evacuation in adverse sight conditions.

The conceptual model involves simulating the evacuation reactions of people in a room using cellular automata (CA) equipped with different categories such as choosing to go in multiple directions that determine their evacuation strategy such as left, right, straight, or back, based on the reference of the wall until they find an exit. Each person is represented as a cell in the 2-dimensional grid of cellular automata and is assigned a category number that defines their evacuation strategy. The model involved people updating their states to evacuate the room through the exit(s). The CD++ tool is used to simulate and test multiple exit scenarios.

**Rules:**

This section outlines the behavior of pedestrians in a simulation. The rules described below are written in pseudo-code and specifically apply to individuals walking upwards, but the same logic can be applied to those walking downwards. In addition to basic movement, to enhance the realism of the simulation, the simulation also includes rules to handle collisions in open spaces and towards the walls.

**Rule 1:** The first rule applies to pedestrians moving straight up, represented by state 1. According to this rule, they continue moving upward when the simulation starts until they reach the wall, represented by state 5. When they reach the wall, they switch to state 3 and begin moving left in reference to the wall.

**Rule 2:** The second rule applies to pedestrians moving straight down, represented by state 2. They continue moving downward when the simulation starts until they reach the wall, represented by state 5. When they reach the wall, they switch to state 4 and begin moving right in reference to the wall.

**Rule 3:** The third rule is a default rule, which applies when a walker does nothing and waits in their current location.

While there are other potential rules for different types of movement, such as people moving left, right, collisions in the other three sides of walls or diagonally to find the wall or various combinations of movements(people going up and right when they find a wall or people going down and left when they find a wall, etc.), these rules are sufficient to understand pedestrian behavior during evacuation. It is important to note that evacuating a room during no visibility cannot be predicted, as the location of individuals is unknown, and they must find their own way out. Therefore, the evacuation method cannot be generalized.

It is also worth noting that this model considers individuals who prefer both left and right turns when they encounter a wall. An interesting aspect of this model is that it has been tested with multiple exits, and the end results have been recorded.

**PART II – Cell DEVS Specification**

**Model Concepts :**

CELL-DEVS :

Cell DEVS is extended DEVS formalism, in which we have implementation of cellular automata to improvise the execution performance of cellular models using a discrete event approach. Here, each cell is defined as atomic model using timing delays, which is later merged to a coupled model representing a cell space.

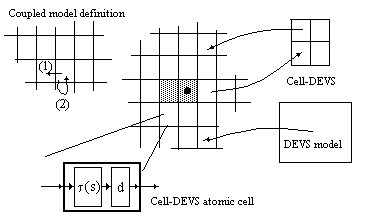
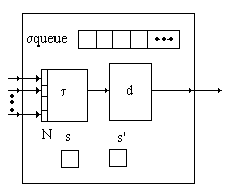


Fig 1: Cell-DEVS Coupled model Fig 2: Cell-DEVS Atomic Model

CD++ Toolkit:

CD++ (Wainer, 2002) is a modelling and simulation toolkit that implements DEVS and Cell-DEVS theory. Atomic models can be defined using a state-based approach (coded in C++ or an interpreted graphical notation), while coupled and Cell-DEVS models are defined using a built-in specification language. We will show the basic features of the tool through an example of application. CD++ also includes an interpreter for Cell-DEVS models. The model specification includes the definition of the size and dimension of the cell space, the shape of the neighbourhood and borders.

The cell’s local computing function is defined using a set of rules with the form:

{POSTCONDITION}{DELAY}{PRECONDITION}.

These indicate that when the PRECONDITION is satisfied, the state of the cell will change to the designated POSTCONDITION, whose computed value will be transmitted to other components after consuming the DELAY.

CELLULAR AUTOMATA :

A red and white flag

Description automatically generated with medium confidenceTo simulate cellular automata, there is a neighbourhood defined for each cell. Each cell has its 2 possible states that is whether it is empty or full. The neighbourhood is generally the adjacent residing cells. We classify the neighborhoods into 2 parts :  [von Neumann neighborhood](https://en.wikipedia.org/wiki/Von_Neumann_neighborhood) and the [Moore neighborhood](https://en.wikipedia.org/wiki/Moore_neighborhood)

Fig 3: The red cells are the [neighborhood](https://en.wikipedia.org/wiki/Moore_neighborhood) for the blue cell

**Cell DEVS Formal Specification**

To specify the model, we are utilizing the following Moore neighborhood..

A picture containing text, crossword puzzle

Description automatically generated

As the above figure shows, the neighborhood consists of 9 cells:

{(-1,-1), (-1,0), (-1,1), (0,-1), (0,0), (0, 1), (1, -1), (1,0), (1, 1)}

The core cell in the model is represented by (0,0), and pedestrians are allowed to move to adjacent cells in the north (-1,0), south (1,0), east (0,1), or west (0,-1) directions according to the rules specified above.

**M = <Xlist, Ylist, I, X, Y, ƞ, N, {r,c}, C, B, Z, SELECT>**

* Xlist = { Ø };
* Ylist = { Ø };
* I = { Ø };
* X = Y = {0,1,2,3,4,5};

The different states of the cells and their corresponding meanings are:

* + 0: Empty cell
  + 1: Cell/pedestrian occupied by an up walker
  + 2: Cell/ pedestrian occupied by a down walker
  + 3: Cell/pedestrian with state 1 has reached a wall and is moving left
  + 4: Cell/ pedestrian with state 2 has reached a wall and is moving right
  + 5: Wall cell
* Ƞ = 9;
* N = {(0,0), (0,1), (0,-1), (1,-1), (1,0), (1,-1), (-1,-1), (-1,0), (-1,1)};
* r = 20; c = 20;
* C = {Cij | i ϵ [0,19], j ϵ [0,19]};
* B = {Ø}; % wrapped
* Z = Inverse neighbourhood of N
* SELECT = {(0,0), (1,0), (0,1), (0,-1), (-1,0)};

**Rules Implementation:**

|  |  |  |  |
| --- | --- | --- | --- |
| **State Names** | **Values** | **Colors** | **Description** |
|  |
| Blank Cell | 0 | Silver | Empty cell. |  |
| Pedestrian Evacuating from Upwards direction | 1 | Red | This cell indicates individuals moving upwards on the grid. |  |
|  |
| Pedestrian Evacuating from downwards direction | 2 | Green | This cell indicates individuals moving downwards on the grid. |  |
|  |
| The Pedestrian occupying the cell with state 1 has now reached a wall and is moving towards the left. | 3 | Yellow | This cell represents a pedestrian who was previously moving up on the grid and has now reached a wall. The person is now changing direction and moving towards the left, relative to the wall. |  |
| The Pedestrian occupying the cell with state 2 has now reached a wall and is moving towards the right. | 4 | Blue | This cell represents a pedestrian who was previously moving down on the grid and has now reached a wall. The person is now changing direction and moving towards the right, relative to the wall. |  |
| Wall | 5 | Brown | Denotes a wall. |  |

* Pedestrian 1 - Going up

rule : 0 100 {(0,0)=1 and (-1,0)=0 and (1,0)!=5 and (-1,1)!=3}

rule : 1 100 {(0,0)=0 and (1,0)=1 and (0,-1)!=5 and (0,1)!=5 and (1,1)!=5 and (0,1)!=3}

* Pedestrian 1 has reached the top wall. State changes to 3 and going left

rule : 3 0 {(0,0)=1 and (statecount(5)>=1)}

* Pedestrian reached a wall at the top and cell to the left is empty

rule : 0 100 {(0,0)=3 and (0,-1)=0 and (-1,0)=5}

* Pedestrian is in current state 0, its right cell is in state 3 with wall at the top (Top to Left top)

rule : 3 100 {(0,0)=0 and (0,1)=3 and (-1,0)=5}

* Pedestrian has reached a wall to left top (0,-1) , cell below is empty
* rule : 0 100 {(0,0)=3 and (1,0)=0 and (0,-1)=5}
* Pedestrian is in current state 0, its up cell is in state 3 with wall to the left top (Left top to Left bottom)
* rule : 3 100 {(0,0)=0 and (-1,0)=3 and (0,-1)=5}
* Pedestrian has reached a wall to the left bottom(1,0), cell to the right is empty
* rule : 0 100 {(0,0)=3 and (0,1)=0 and (1,0)=5}
* Pedestrian is in current state 0, the person to the left is in state 3 with wall to left bottom (Left bottom to right bottom)
* rule : 3 100 {(0,0)=0 and (0,-1)=3 and (1,0)=5}
* Pedestrian has reached a wall to right bottom (0,1) , cell above is empty
* rule : 0 100 {(0,0)=3 and (-1,0)=0 and (0,1)=5}
* Pedestrian is in current state 0, the person to the bottom is in state 3 with wall to right bottom (Right bottom to right top)
* rule : 3 100 {(0,0)=0 and (1,0)=3 and (0,1)=5}
* Pedestrian 2 - Going down
* rule : 0 100 {(0,0)=2 and (1,0)=0 and (-1,0)!=5 and (1,1)!=4}
* rule : 2 100 {(0,0)=0 and (-1,0)=2 and (0,1)!=5 and (0,-1)!=5 and (-1,-1)!=5 and (0,1)!=4}
* Pedestrian 1 has reached the bottom wall. State changes to 4 and going right
* rule : 4 0 {(0,0)=2 and (statecount(5)>=1)}
* Pedestrian has reached to bottom wall (1,0) and cell to the left is empty
* rule : 0 100 {(0,0)=4 and (0,-1)=0 and (1,0)=5}
* Pedestrian is in current state 0 and person the right is in state 4 with wall at the bottom (bottom to left bottom)
* rule : 4 100 {(0,0)=0 and (0,1)=4 and (1,0)=5}
* Pedestrian has reached the left bottom wall (0,-1) and cell above is empty
* rule : 0 100 {(0,0)=4 and (-1,0)=0 and (0,-1)=5}
* Pedestrian is in state 0 and cell to the bottom is in state 4 with wall at the left bottom (Left bottom to left top)
* rule : 4 100 {(0,0)=0 and (1,0)=4 and (0,-1)=5}
* Pedestrian has reached the left top wall (-1,0) and cell to the right is empty

rule : 0 100 {(0,0)=4 and (0,1)=0 and (-1,0)=5}

* Pedestrian is in state 0 and the cell to the right is in state 4 with wall the left top (Left top to right top)

rule : 4 100 {(0,0)=0 and (0,-1)=4 and (-1,0)=5}

* Pedestrian has reached the right top and the cell below is empty

rule : 0 100 {(0,0)=4 and (1,0)=0 and (0,1)=5}

* Pedestrian is in current state 0, the top cell is in state 4 with wall at the right top (Right top to right bottom)

rule : 4 100 {(0,0)=0 and (-1,0)=4 and (0,1)=5}

* Prevent collisions in the open area of the cells:
  + Pedestrian is in current state 2(down) and a cell is in state 1(up), dont change the position

rule : 0 100 {(0,0)=2 and (1,0)=1}

* + Pedestrian is in current state 0, the cell to top right is in state 2(down) and either of right and bottom cell is in state 1(up)

rule : 2 100 {(0,0)=0 and (-1,1)=2 and ((0,1)=1 or (1,1)=1)}

* Prevent collisions in bottom right to bottom left

rule : 4 100 {(0,0)=3 and (0,1)=4 and (1,0)=5}

rule : 3 100 {(0,0)=0 and (0,-1)=3 and (0,1)=4 and (-1,-1)=0 and (-1,0)!=2 and (1,0)=5}

* Default - don't move

rule : {(0,0)} 100 {t}

**Model Specification:**

[top]

components : pedestrianevacuation

[pedestrianevacuation]

type : cell

width : 20

height : 20

delay : transport

defaultDelayTime : 100

border : wrapped

neighbors : pedestrianevacuation(-1,-1) pedestrianevacuation(-1,0) pedestrianevacuation(-1,1)

neighbors : pedestrianevacuation(0,-1) pedestrianevacuation(0,0) pedestrianevacuation(0,1)

neighbors : pedestrianevacuation(1,-1) pedestrianevacuation(1,0) pedestrianevacuation(1,1)

%%%% state values %%%%

% 0 - Empty Cell

% 1 - cell occupied by an up walker

% 2 - cell occupied by a down walker

% 3 - cell with state 1 has reached a wall and is going left

% 4 - cell with state 2 has reached a wall and is going right

% 5 - wall cell

initialvalue : 0

initialrowvalue : 0 55555555555555555555

initialrowvalue : 1 50000000000000000105

initialrowvalue : 2 50000000000000000005

initialrowvalue : 3 50000000000000000005

initialrowvalue : 4 50010000000000000005

initialrowvalue : 5 50000000000000000005

initialrowvalue : 6 00000000000000000005

initialrowvalue : 7 50000000000000000005

initialrowvalue : 8 50000000000000000005

initialrowvalue : 9 50000000000020000005

initialrowvalue : 10 50000000000012000005

initialrowvalue : 11 50000000000000000005

initialrowvalue : 12 50100000000000000005

initialrowvalue : 13 00000000000000200005

initialrowvalue : 14 50000000000000000005

initialrowvalue : 15 50000012000000000005

initialrowvalue : 16 50000000000000000005

initialrowvalue : 17 50000000000000000005

initialrowvalue : 18 51000000000000000025

initialrowvalue : 19 55555555555555555555

localtransition : pedestrianevacuation-rule

[pedestrianevacuation-rule]

%Pedestrian 1 - Going up

rule : 0 100 {(0,0)=1 and (-1,0)=0 and (1,0)!=5 and (-1,1)!=3}

rule : 1 100 {(0,0)=0 and (1,0)=1 and (0,-1)!=5 and (0,1)!=5 and (1,1)!=5 and (0,1)!=3}

%Pedestrian 1 has reached the top wall. State changes to 3 and going left

rule : 3 0 {(0,0)=1 and (statecount(5)>=1)}

%Pedestrian reached a wall at the top and cell to the left is empty

rule : 0 100 {(0,0)=3 and (0,-1)=0 and (-1,0)=5}

%Pedestrian is in current state 0, its right cell is in state 3 with wall at the top (Top to Left top)

rule : 3 100 {(0,0)=0 and (0,1)=3 and (-1,0)=5}

%Pedestrian has reached a wall to left top (0,-1) , cell below is empty

rule : 0 100 {(0,0)=3 and (1,0)=0 and (0,-1)=5}

%Pedestrian is in current state 0, its up cell is in state 3 with wall to the left top (Left top to Left bottom)

rule : 3 100 {(0,0)=0 and (-1,0)=3 and (0,-1)=5}

%Pedestrian has reached a wall to the left bottom(1,0), cell to the right is empty

rule : 0 100 {(0,0)=3 and (0,1)=0 and (1,0)=5}

%Pedestrian is in current state 0, the person to the left is in state 3 with wall to left bottom (Left bottom to right bottom)

rule : 3 100 {(0,0)=0 and (0,-1)=3 and (1,0)=5}

%Pedestrian has reached a wall to right bottom (0,1) , cell above is empty

rule : 0 100 {(0,0)=3 and (-1,0)=0 and (0,1)=5}

%Pedestrian is in current state 0, the person to the bottom is in state 3 with wall to right bottom (Right bottom to right top)

rule : 3 100 {(0,0)=0 and (1,0)=3 and (0,1)=5}

%Pedestrian 2 - Going down

rule : 0 100 {(0,0)=2 and (1,0)=0 and (-1,0)!=5 and (1,1)!=4}

rule : 2 100 {(0,0)=0 and (-1,0)=2 and (0,1)!=5 and (0,-1)!=5 and (-1,-1)!=5 and (0,1)!=4}

%Pedestrian 1 has reached the bottom wall. State changes to 4 and going right

rule : 4 0 {(0,0)=2 and (statecount(5)>=1)}

% Pedestrian has reached to bottom wall (1,0) and cell to the left is empty

rule : 0 100 {(0,0)=4 and (0,-1)=0 and (1,0)=5}

% Pedestrian is in current state 0 and person the right is in state 4 with wall at the bottom (bottom to left bottom)

rule : 4 100 {(0,0)=0 and (0,1)=4 and (1,0)=5}

%Pedestrian has reached the left bottom wall (0,-1) and cell above is empty

rule : 0 100 {(0,0)=4 and (-1,0)=0 and (0,-1)=5}

%Pedestrian is in state 0 and cell to the bottom is in state 4 with wall at the left bottom (Left bottom to left top)

rule : 4 100 {(0,0)=0 and (1,0)=4 and (0,-1)=5}

%Pedestrian has reached the left top wall (-1,0) and cell to the right is empty

rule : 0 100 {(0,0)=4 and (0,1)=0 and (-1,0)=5}

%Pedestrian is in state 0 and the cell to the right is in state 4 with wall the left top (Left top to right top)

rule : 4 100 {(0,0)=0 and (0,-1)=4 and (-1,0)=5}

%Pedestrian has reached the right top and the cell below is empty

rule : 0 100 {(0,0)=4 and (1,0)=0 and (0,1)=5}

%Pedestrian is in current state 0, the top cell is in state 4 with wall at the right top (Right top to right bottom)

rule : 4 100 {(0,0)=0 and (-1,0)=4 and (0,1)=5}

%Prevent collisions in the open area of the cells

%Pedestrian is in current state 2(down) and a cell is in state 1(up), dont change the position

rule : 0 100 {(0,0)=2 and (1,0)=1}

%Pedestrian is in current state 0, the cell to top right is in state 2(down) and either of right and bottom cell is in state 1(up)

rule : 2 100 {(0,0)=0 and (-1,1)=2 and ((0,1)=1 or (1,1)=1)}

%avoid clash in bottom right to bottom left

rule : 4 100 {(0,0)=3 and (0,1)=4 and (1,0)=5}

rule : 3 100 {(0,0)=0 and (0,-1)=3 and (0,1)=4 and (-1,-1)=0 and (-1,0)!=2 and (1,0)=5}

% Default - don't move

rule : {(0,0)} 100 {t}

**Testing and Simulation Results:**

To test multiple simulations, the values defined in **initialrowvalues** on .ma file can be changed and the results are observed.

1. **Simple Scenario having 4 pedestrians moving upwards and 4 pedestrians moving downwards**

Figure 1 below has 1 evacuation path and consists of 4 pedestrians moving upwards [Red cells], 4 moving downwards[Green cells], 1 up walker who has reached the wall and following it[Yellow cells], 1 down walker who has reached the wall and following it[Blue cell].

Chart, waterfall chart

Description automatically generated

Figure 2 below displays that since there are no walkers or obstacles directly on the path of all the pedestrians, they move smoothly in their direction towards the wall.

Chart, waterfall chart

Description automatically generated

Figure below shows that in the next step, once the upward(Red) and downward(Green) pedestrians reach the wall they change their state to 3(Yellow) and 4(Blue) respectively following the wall.

Chart, waterfall chart

Description automatically generated

Moving further with the simulation, it can be seen below that the pedestrians are evacuating smoothly.

Chart, waterfall chart

Description automatically generated

The simulation ended at 00:00:03:200 with the smooth evacuation of pedestrians as follows:

A picture containing application

Description automatically generated

1. **Complex Scenario handling collisions in open space and towards the wall**

Figure 1 below has 2 evacuation paths and consists of 4 pedestrians moving upwards [Red cells], 4 moving downwards[Green cells], 2 up walkers who have reached the wall and following it[Yellow cells], 1 down walker who has reached the wall and following it[Blue cell].

Chart, waterfall chart

Description automatically generated

Figure 1

In the next step, collision is prevented in an open space and providing a way to both the up walker and downwalker.

Chart, waterfall chart

Description automatically generated

Figure 2

Moving further with the simulation, once the down walker reaches the wall, it changes the state to 4(Blue). Similarly, once the up walker reaches the wall, it changes the state to 3(Yellow). [Figure 3]

Chart, waterfall chart

Description automatically generated

Figure 3

Moving further with the simulation step, we can see that there is a collision in the next step with yellow(state 3) and blue(state 4) cell[Figure 4]. In this case, as per the rules the pedestrian with state 4 changes its state to 3 and look for the path to evacuate [Figure 5].

Additionally, the pedestrian with green cell(state 2) remains in the same position if there is a cell ahead and allow the pedestrian with blue cell(state 4) to move further. Once a path is available, the green cell continues to move.[Figure 4 and 5]

Chart, waterfall chart

Description automatically generated

Figure 4

Chart, waterfall chart

Description automatically generated

Figure 5

According to Figure 6, pedestrians moving in both upward and downward directions have encountered a wall whose states are now changed to 3 and 4 respectively. Now, they are following a designated path to evacuate.

Chart, waterfall chart

Description automatically generated

Figure 6

Figure 7 displays that the pedestrian are evacuating properly following the walls from the two paths defined.

Chart

Description automatically generated

Figure 7

Figure 8 displays the successful evacuation through simulation with two paths. The simulation ended at 00:00:02:800.

A picture containing graphical user interface

Description automatically generated

Figure 8

The rules in the simulation perform as anticipated throughout its duration.

**Conclusion**

In conclusion, the rule set implemented in the simulations performed well, without any observed collisions in open space or across the walls. Pedestrians moved smoothly and naturally throughout the simulations, indicating the effectiveness of the rules. This assignment provided a significant opportunity for learning and resulted in a successful outcome overall.

**Further Considerations for Term Project**

In this assignment, I have limited the complexity by considering only certain conditions for the simulation. The simulation includes a pedestrian movement upwards and downwards but not left, right, or diagonally. If a person were to move left, right, or diagonally, additional rules would need to be considered. Additionally, there are possible combinations of movements that could be added to simulate evacuation behavior, such as moving down and left once reaching a wall or up and right once reaching a wall. However, it is difficult to account for every person's mental state during the evacuation, as it is unpredictable and subject to change. This lack of visibility adds to the complexity of the simulation. However, I am planning to consider such scenarios and add more rules to the same model as part of an enhancement for the Term project.

**Reference**

**[**1] An extended cost potential field cellular automaton model for pedestrian evacuation considering the restriction of visual field, 2019. By Xingli Li, Fang Guo, Hua Kuang, Zhongfei Geng, Yanhong Fan.