ASSIGNMENT 2

SYSC-5104

**WORM-FOOD IN CELL-DEVS**

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

CARLETON UNIVERSITY

2016

SUBMITTED BY:

NAVINDER KAUR BRAR (101026788)

SAMANEH AHMADVAND (101021682)

**1. INTRODUCTION TO MODEL**

In this assignment we focus on genetic programming, which is a search heuristic and changes its behaviour (weight in this case) with change in state. The main idea of this assignment is based on the "snake game" and instead of snake we have “worm” and “food”. There are two main points which makes this game of particular interest. First, the worm eats the available piece of food, which are placed on the game board by moving in different directions and making its way to food. Second, as worm eats something its weight increases.

The game begins with a worm having a fixed number of body segment which is represented as single cell confined to a rectangular board of 10 x 10 cells. We consider three planes for this game-plane 0 represents the movement of worm, plane 1 is the position where food is placed and plane 2 is the direction in which worm move to get food. In movement plane or plane 0 there is a worm or an empty cell and for food it looks in direction. There are five different situations for movement:

* Not going to move
* North
* South
* East
* West

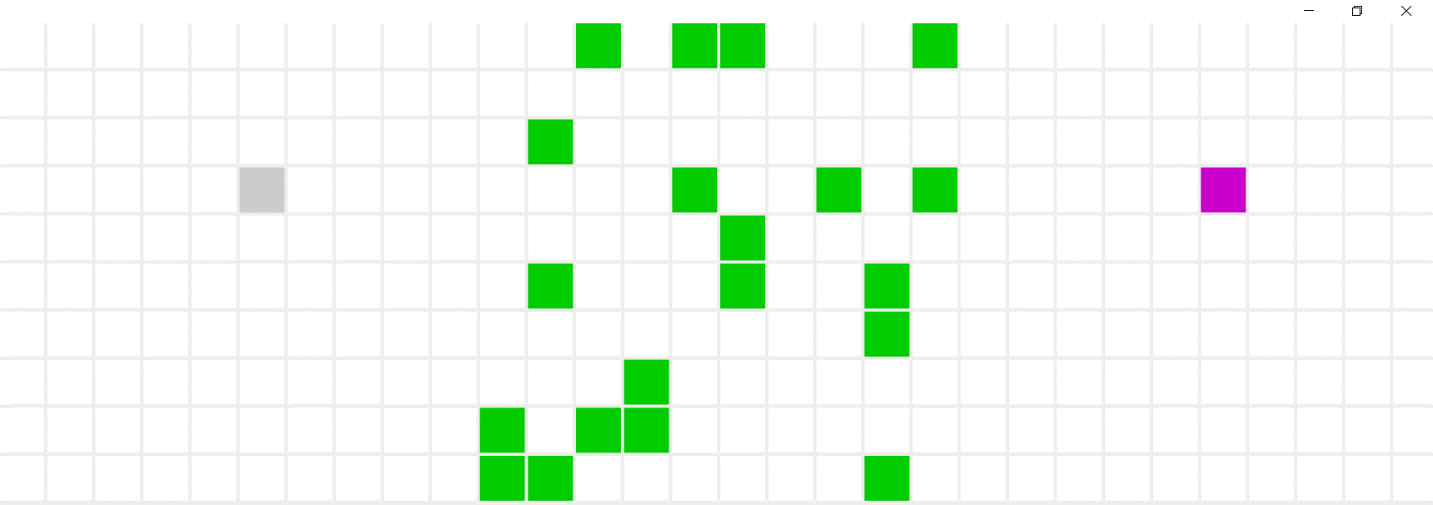
In a food plane or plane 1, “0” means “No Food” while “600” means “Food”.

Also various other representations in direction plane or plane 2 are named below:

* 500- No movement
* 501- North
* 502-South
* 503-East
* 504-West

In movement plane value of cell can range from 0 to 99.If the value of movement plane of cell is “0” this means that cell does not have any worm. The cell with worm can range according to its weight or it increments after eating food available every time. With each time step that passes, the worm can either change direction to North, South, East, and West or remain at same position. The movement of worm or direction id chosen randomly. The initial value of worm is considered as 1 or its weight is 1 but after each step of consuming food it increases.

Movement Food Direction



*Figure 1: Initial seen of simulation as seen in cell-DEVS animation*

From the above figure we can see that it consists of three planes as already described where in zero plane, the grey cell represents the position of worm. In plane 1, all the green entities are food which are initialised in the worm.val file. In second plane the pink colour describes the south direction, to which the worm will move in next step. The direction part is randomly selected and thus describes the next position of the worm and it is not sure that it would always land in the cell which have food for worm. For plane movement and direction only one cell in each plane would remain active while in that of food plane, initially they could even occupy the whole area of 10 x10 but here we have only considered 20 cells with food.

**2.1 FORMAL SPECIFICATION**

Formal specification of the **Atomic Cell-DEVS** model and **Coupled Cell-DEVS** is as following. The more detail about the function is on the next section.

= as defined in worm.ma

= as defined in worm.ma

= as defined in worm.ma

Formal specification of the **Coupled Cell-DEVS** model is as following:

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

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

**2.2FUNCTIONS AND RULES**

In this part of report we focus on the initialization of simulations i.e. how it is written in cd++. The rules are described in this part which determines the behaviour of each cell. This part is extracted from the worm.ma and more information about the rules and its results can be found from there. The following is the definition of model. In this we have defined the dim which represents the dimensions of the cellular model. Delay represent the delay type used in every cell. Delay Time during external events is initialized as integer and is time in milliseconds. All the initial values for each cell are described in “worm.val”. Zone is declared with the transition function name which allows to define an alternative behavior for the group of cells understood inside the specified range. Each range is defined as describing a unique cell, describing an area of cells, or a list that can combine both of them.

**[top]**

**components : worm**

**[worm]**

**type : cell**

**dim : (10,10,3)**

**delay : transport**

**defaultDelayTime : 100**

**border : wrapped**

**neighbors :....**

**initialvalue : 0**

**initialCellsValue : worm.val**

**localtransition : Movement**

**zone : Movement { (0,0,0)..(9,9,0) }**

**zone : Food { (0,0,1)..(9,9,1) }**

**zone : Direction { (0,0,2)..(9,9,2) }**

The language is defined as a new group inside the specification, where each component of the group is a rule and each rule is composed of three elements –condition, delay and result. In movement plane we have set of rules for worm. One of the basic rule is if worm moves from one cell to its neighboring cell, the previously occupied cell has to be empty. These rules also change and effect the other planes as in the first rule we have described if the worm wants to stay in the cell or there is no movement. Also for each rule in plane we have two cases, one when there is food in cell or there is no food. In movement plane the rules are described for every direction with availability of food.

**[Movement]**

**%There is a worm in the cell and it wants to stay in this cell**

**rule : {(0,0,0)} 100 { (0,0,0) > 0 and (0,0,2) = 500 }**

**%There is a worm in the cell, and it wants to move. Because, we just have one worm, when it moves from a cell, that cell will be empty.**

**rule : 0 100 { (0,0,0) > 0 and (0,0,2) != 500 }**

**%A cell is empty, and there is worm in the north neighbour that wants to come to the south (come to this cell).**

**rule : {(-1,0,0)} 100 { (0,0,0) = 0 and (0,0,1) = 0 and (-1,0,0) > 0 and (-1,0,2) = 502} %This cell has no food.**

**rule : {(-1,0,0) + 1} 100 { (0,0,0) = 0 and (0,0,1) = 600 and (-1,0,0) > 0 and (-1,0,2) = 502} %This cell has food.**

**%A cell is empty, and there is worm in the south neighbour that wants to come to the north (come to this cell).**

**rule : {(1,0,0)} 100 { (0,0,0) = 0 and (0,0,1) = 0 and (1,0,0) > 0 and (1,0,2) = 501} %This cell has no food.**

**rule : {(1,0,0) + 1} 100 { (0,0,0) = 0 and (0,0,1) = 600 and (1,0,0) > 0 and (1,0,2) = 501} %This cell has food.**

**%A cell is empty, and there is worm in the east neighbour that wants to come to the west (come to this cell).**

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

**(0,1,2) = 504} %This cell has no food.**

**rule : {(0,1,0) + 1} 100 { (0,0,0) = 0 and (0,0,1) = 600 and (0,1,0) > 0 and (0,1,2) = 504} %This cell has food.**

**%A cell is empty, and there is worm in the west neighbour that wants to come to the east (come to this cell).**

**rule : {(0,-1,0)} 100 { (0,0,0) = 0 and (0,0,1) = 0 and (0,-1,0) > 0 and (0,-1,2) = 503} %This cell has no food.**

**rule : {(0,-1,0) + 1} 100 { (0,0,0) = 0 and (0,0,1) = 600 and (0,-1,0) > 0 and (0,-1,2) = 503} %This cell has food.**

**%If we reach to this rule it means:**

**%A cell is empty, and there is NOT a worm in the neighbours**

**%or**

**%A cell is empty, and there is a worm in the one of the neighbours, but this worm does not want to come to this cell.**

**%So, we set this cell value to 0.**

**rule : 0 100 { (0,0,0) = 0}**

The following box describes about the rules of plane 1 or food plane. If there is no food in the cell its value is 0 if the value of cell =600 or there is food in the cell then various rules applies. If there is food in cell and worm wants to eat it then after eating the value of that cell will change into 0.This rule is described for each direction considered in the assignment. Another case is when there is food available in neighbor but worm does not want to consume it and for this the worm would move to some other neighboring place and the food would remain in the same place, also the weight of worm cannot be increased.

**[Food]**

**%There is no food in this cell, so its value remains 0.**

**rule : 0 100 { (0,0,0) = 0 }**

**%The following rules of the Food section are about the cells that have food.**

**%A worm wants to go to the correspondent cell in the movement plane. This food is eaten by the worm, and there will be no food in this cell.**

**rule : 0 100 {(0,0,0) = 600 and (-1,0,-1) > 0 and (-1,0,1) = 502} %There is worm in the north neighbour that wants to come to the south.**

**rule : 0 100 {(0,0,0) = 600 and (1,0,-1) > 0 and (1,0,1) = 501} %There is worm in the south neighbour that wants to come to the north.**

**rule : 0 100 {(0,0,0) = 600 and (0,1,-1) > 0 and (0,1,1) = 504} %There is worm in the east neighbour that wants to come to the west.**

**rule : 0 100 {(0,0,0) = 600 and (0,-1,-1) > 0 and (0,-1,1) = 503} %There is worm in the west neighbour that wants to come to the east.**

**%There is food in the cell, but it is not going to be consumed by the worm.**

**rule : 600 100 {(0,0,0) = 600}**

Rules for Direction plane is generated randomly and is considered only for the cells which have worm in its correspondent cell. The next position of worm is selected randomly and the other case is when no worm wants to come to that cell.

**[Direction]**

**%For this plane cells, set the next movement direction if there is going to be a worm in the correspondent cell of them in the movement plane.**

**% randInt(n) returns a random integer from [0,n]**

**rule : {randInt(4) + 500} 100 {(0,0,-2) > 0 and (0,0,0) = 500} %There is worm in the cell that wants to stay.**

**rule : {randInt(4) + 500} 100 {(-1,0,-2) > 0 and (-1,0,0) = 502} %There is worm in the north neighbour that wants to come to the south.**

**rule : {randInt(4) + 500} 100 {(1,0,-2) > 0 and (1,0,0) = 501} %There is worm in the south neighbour that wants to come to the north.**

**rule : {randInt(4) + 500} 100 {(0,1,-2) > 0 and (0,1,0) = 504} %There is worm in the east neighbour that wants to come to the west.**

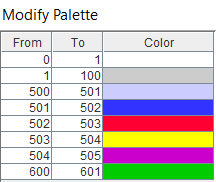
**rule : {randInt(4) + 500} 100 {(0,-1,-2) > 0 and (0,-1,0) = 503} %There is worm in the west neighbour that wants to come to the east.**

**%There is no worm that wants to come to this cell.**

**rule : 0 100 { (0,0,0) >= 0 }**

**3. CELL-DEVS ANIMATION**

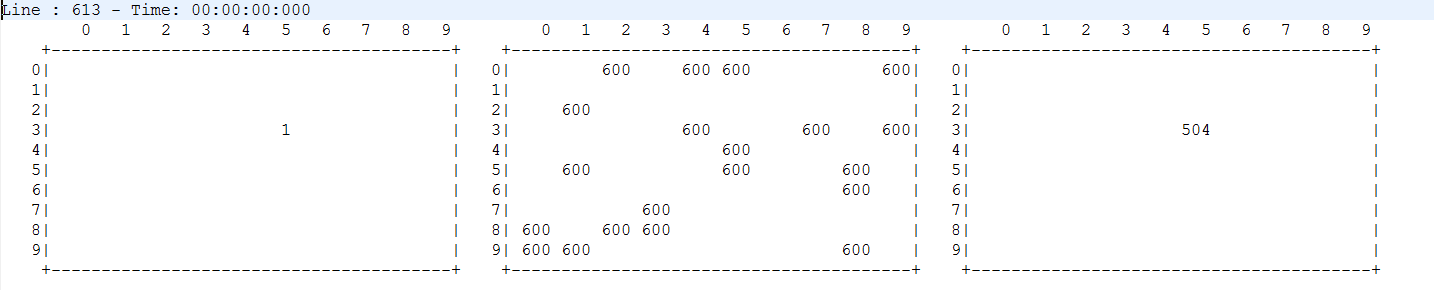
We use the animation tool to see properly how things work in the model and in here we choose different colour according to the change in rules and the colour scheme is loaded in file “worm.pal”.



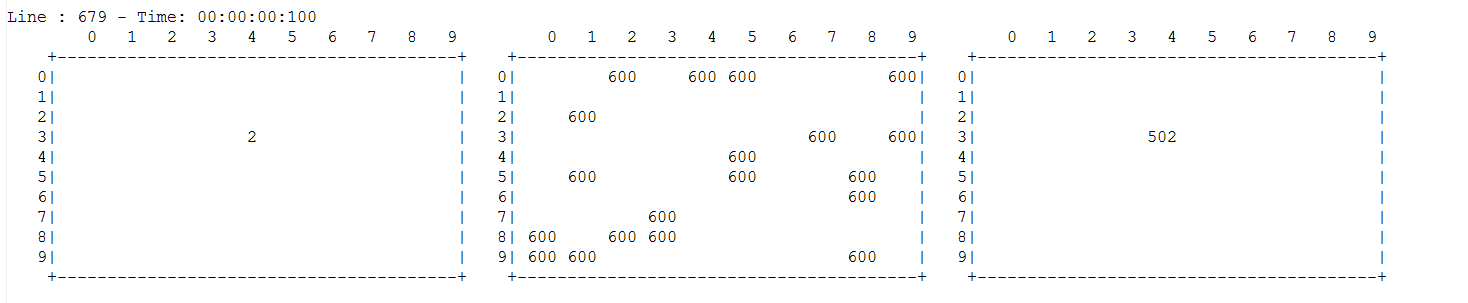
*Figure 2 Used Colour Scheme*

**4. RESULTS**

In this section we will discuss about the results that we got after running the simulation. Here we have shown various slots during simulation which are generated in worm.drw file. The captures consists of simulation results from start, somewhere in between the simulation and at the end of the simulation. The time for which the simulation lasted is 18:600.Each figure consist of three planes as described earlier-movement, food and direction. Initial value which describes or shows us the existence of worm in movement plane is 1 and after consuming food which is shown in plane food or plane 1, the weight increases from 1 to 2 or so on. The third plane or direction plane shows the movement and have different values when moving in neighbours. As for figure 1 it contains all the initial values as described in worm.val while for the next step of simulation it checks the rule from worm.ma file and sees the food and moves to the west and so in next step i.e. in Figure 2 we could see the weight of worm increased or changing from 1 to 2 while its direction or position also changes or is now same as where the food was available. For the next movement it checks its neighbouring cells if no food is available it randomly chooses the direction and changes its position.

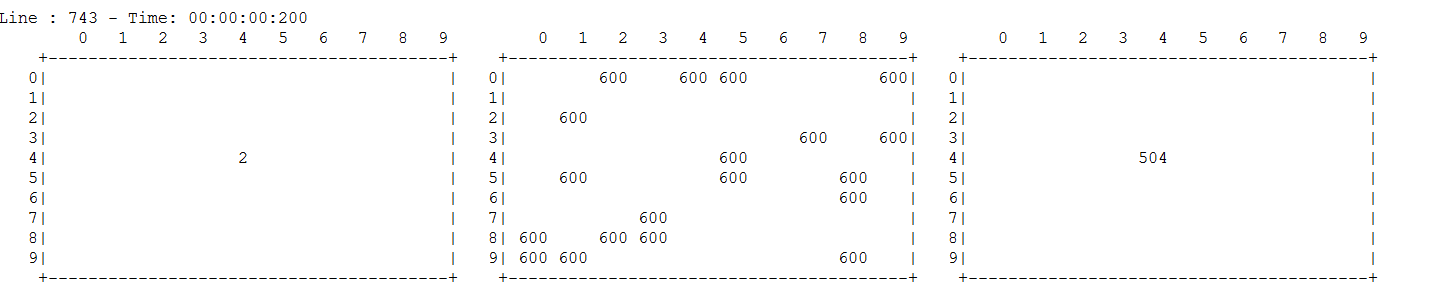


*Figure 3 Initial Values or First Step in Simulation*

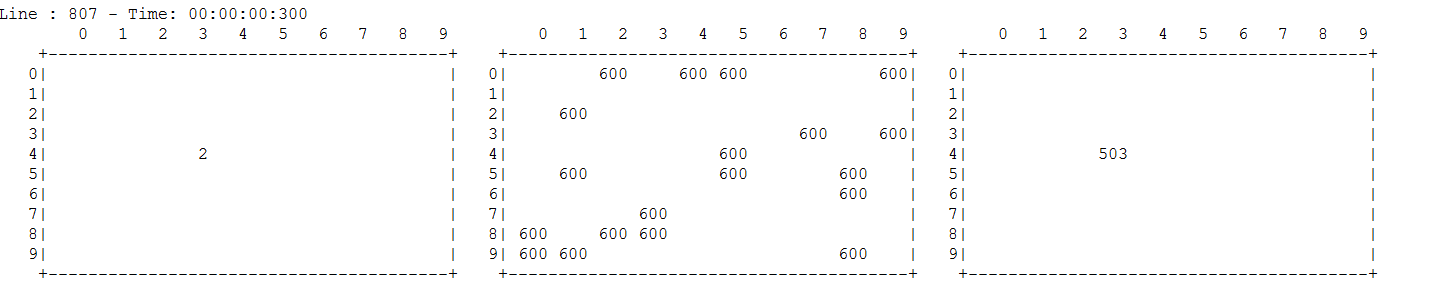


*Figure 4 Second Step in Simulation (Increase in weight for worm)*

In third step of simulation movement of worm was seen but as there was no food available its weight remained same.

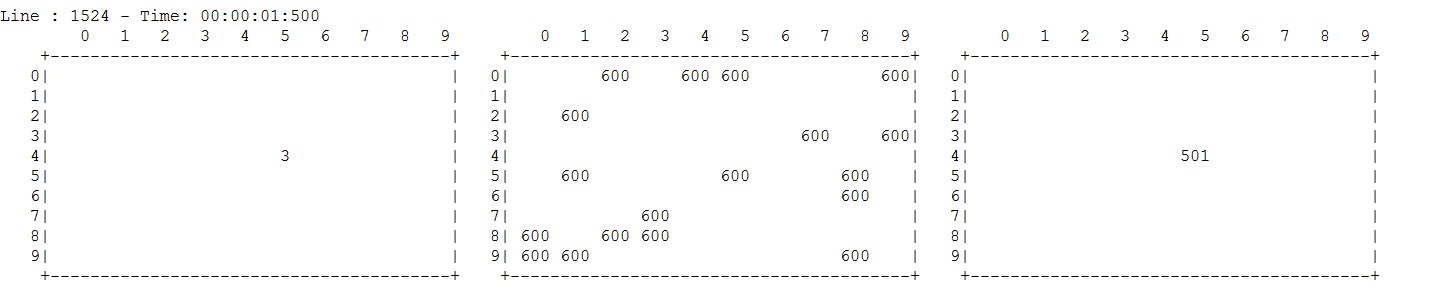


*Figure 5 Third Step in Simulation*



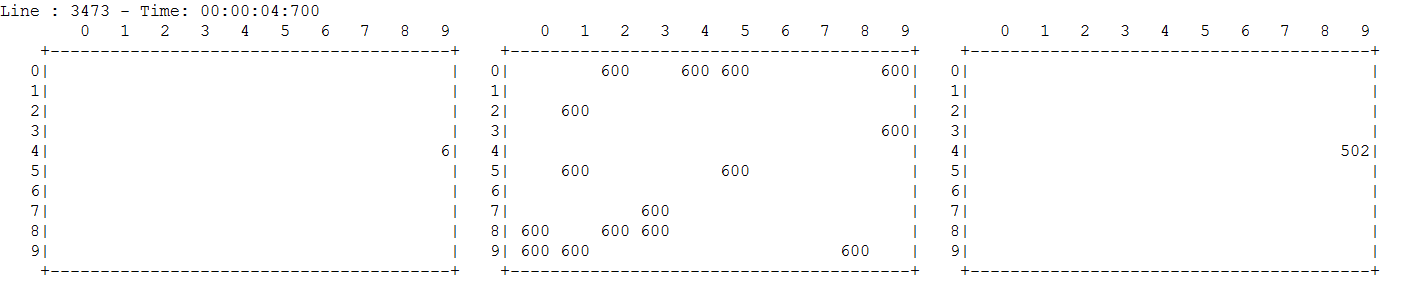
*Figure 6 Fourth step in Simulation*

After skipping to various steps, the following step again shows the movement of worm and consuming food available in food plane and it changes from 2 to 3 and food cell is empty in food plane and worm is seen occupying that very cell in movement plane.

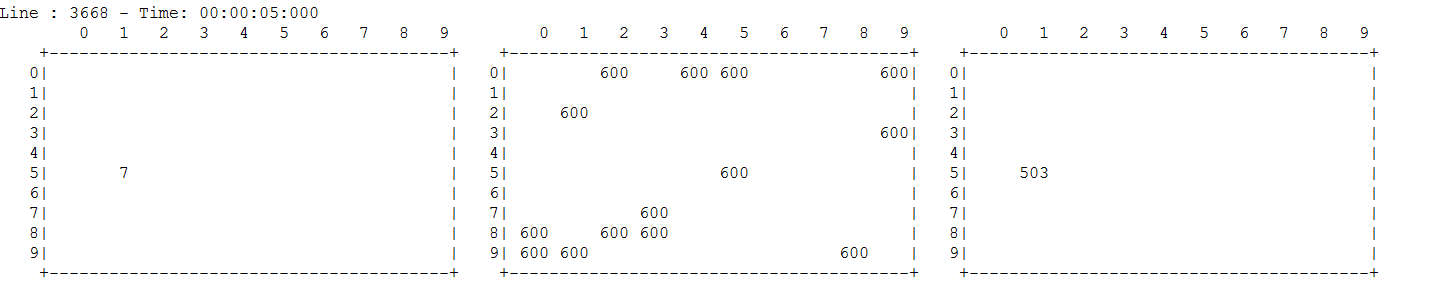


*Figure 7 Middle step in simulation*

The following steps in simulation are taken somewhere in between and again represents some changes in the movement, weight of snake as well as in food plane as food is decreasing and weight of worm is increasing.

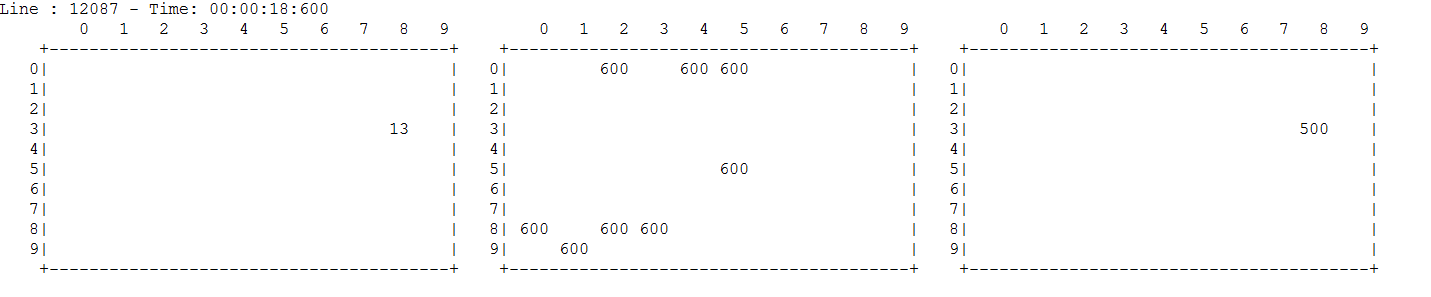


*Figure 8 Nth step in simulation*



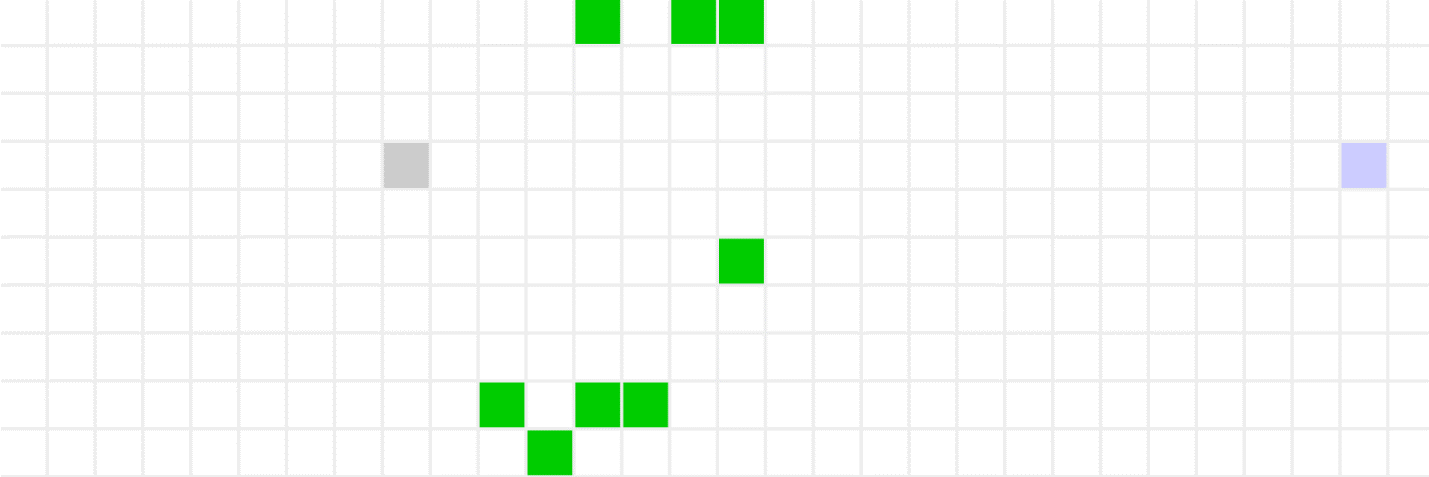
*Figure 9 View after few seconds of Nth step*

The following is the last step in simulation where we can see that there are very less number of food available in food plane and are consumed by worm. Weight of worm is increased from 1 to 13.This step is performed at the 18:600 second of the simulation. At the end value of (3,8,0) is 13.In food plane food is only left at (0,2,1),(0,4,1),(0,5,1),(5,5,1),(8,0,1),(8,2,1),(8,3,1),(9,1,1) for the rest of cells in food plane the value is 0.



*Figure 10 Final Step in Simulation*

Movement Food Direction



*Figure 11 Final Step as seen in animation*