Studying Swarm Mechanics through Cell-DEVS
SYSC5104 Assignment 2
Author: James Baak, 101002918

Part I: Introduction

Cell-DEVS can be used to represent entities travelling through space that depend on their relative neighbours. An application of Cell-DEVS can be applied to the swarm mechanics of insects, fish, and other animals. In the context of this study, a swarm is defined as a large group of animals, usually insects, that form a dense group to perform a certain action like breeding, defence, attack patterns, etc.

The presented material in [1, 2] have used a discrete time-step cellular automata model to represent the movement of bees on a 2-D plane. The groups in [1, 2] have constructed a model using experiments using actual bees and have validated the results between the model and the actual movement of the bees. The project has introduced a series of robots that also control the temperature of certain spaces in an attempt to steer the bees and move them to certain points. These papers do not describe how a whole swarm acts together, but have provided a model for a bee's random walk through space using temperature and the location of other bees (bee-to-bee collision) to determine a walking path. I believe the model presented in [1, 2] can be applied to other research questions like the one in [3] on bee cluster thermoregulation or modified slightly to represent different insect swarm patterns. Once these insect or prey swarm models are defined, predator swarm patterns can then be studied to determine how certain animals interact with the original prey swarm.

I propose to model a 2-D Cell-DEVS model that represents the walking movement of many bees across a grid using a Moore or Hexagonal topology neighbour. The grid will represent movement within the hive. Instead of using robots to control local temperatures in the grid, the local temperature will be dependent on the number of bees within the local area and global temperature. The cellular model can then be used to examine the clusters of bees in varying conditions, such as number of bees and external hive temperature. Other potential research parameters could be the introduction of a hostile foriegn entity, such as a wasp, or introducing tasks to the different bee types within the hive; worker, drone, and queen.

Part II: Model and Simulation

The following project has built off the work presented in [1]. Stefanec et al have designed a cellular automata model of a bee and CASU robots. The bees were modelled with a random walking pattern and wait times when other bees are within proximity of the bee. The CASU robots were used to control the temperature around the bees to change the behaviour of the bees by modifying temperatures in the bee's environment. The model and the modifications that have been applied are described in the sections below.

Encoding

As the currently used version of CD++ can only hold one state for each cell, the multiple cell state values for representing the bee model must be encoded in a single real or integer value. The state will be encoded in a 4 digit integer by the following:

TTMD - each letter represents a digit, therefore the state is 4 digits

TT - The temperature of cell, 28 <= TT <= 36 (bounded by [1], since waiting function depends on T)

M - A state var to keep state of movement across cells (0 - 8)
- Bee cell: M holds the movement state of the bee: {0 -> rest, 1 -> change direction, 2 -> move indent, 3 -> moving }
- Blank cell: M holds the number of bees that want to move to that cell (max 8)
- Not used for walls

D - The direction of the Bee, 1 <= D <= 8, 0 for blank cell and 9 for wall, therefore indication of a bee is represented by D being between or equal to 1 and 8

Layout:

1	2	3
8	B	4
7	6	5

1 -> NW, 2 -> N, 3 -> NE, ..., 8 -> W

Walls have not been implemented in this version as the interaction of bees in a wrapped space is considered instead. They can be implemented with ease during the move indent state of the bee, like the bee collision detection.

Cell DEVS Definition

```
IDC = <X, Y, I, S, θ, N, d, δint, δext, τ, λ, D>
X = Y = {}
I = <η = 9, μ = 0, Px = {}, Py= {}>
S = {s ∈ Z | 2800 <= s <= 3700}
θ = {s, phase, f, σ} // Inertial delay
d = { d ∈ R | 0 < d < ∞ }
τ = {
   // Refer to encoding section for state meaning
   // TTMD - temperature, movement, direction
   // here - the neighbour (0,0), or self

   // Blank cell
```

```
if (isBlank) { // D = 0
   if (bee moving here) { // Bee neighbours with M = 3 and direction pointing here
      TT = calculateTemp()
      M  = 1
      D  = direction of bee moving here
      s  := encode(TT, M, D)
      delay(0)
   } else { // Be not moving to this cell
      TT = calculateTemp()
      M  = number of bees with M = 2 and direction pointing here
      D  = 0 // Stay blank
      s  := encode(TT, M, D)
      delay(0)
   }
}
// Bee cell
else if (isBee) { //  1 <= D <= 8
   if (bee waiting) { // M = 0
      TT = calculateTemp()
      M  = 1
      D  = D
      s  := encode(TT, M, D)
      delay(waitTime() * 1000) // function returns in seconds, so convert to ms
   } else if (bee changing direction) { // M = 1
      TT = calculateTemp()
      M  = 2
      if (uniform(0,1) < w) { // w = random walk probability (0.1 [1])
         D = randint(1,8) // Choose new direction
      } else {
         D = D
      }
      s  := encode(TT, M, D)
      delay(500) // Time to travel
   } else if (bee move indent) { // M = 2
      TT = calculateTemp()
      if (noCollision AND not socialize()) { // Move available and bee doesn't start socializing
         M = 3 // Bee moves to new cell
      } else {
         M = 0 // Bee can't move to the cell at D
      }
      D  = D
      s  := encode(TT, M, D)
      delay(500) // Time to travel
   } else if (bee moving) { // M = 3
      TT = calculateTemp()
      M  = 0
      D  = 0 // Set to blank, bee moving out
      s  := encode(TT, M, D)
      delay(0)
   }
```

```
    }
    else {
       // keep the state the same
       s := s
       delay(0)
    }

    // Functions ======
    encode(TT, M, D) = (TT * 100) + (M * 10) + D

    calculateTemp() {
       temp = 28 // Ambient temp
       for each bee in neighbour:
          temp++ // +1 degree celsius
       return temp
    }

    //Get wait time in seconds using function provided in [1]
    waitTime() {
       // Taken from Eq. 1 in [1]
       (power(3.09 - 0.0403 * TT,-27)/
       (power(3.09 - 0.0403 * TT,-27) + power(1.79,-27))) * 22.5 + 0.645
    }

    // Whether the bee will stop to socialize with other bees; Returns boolean
    // Stopping power parameter can be either 0.4 or 0.8 and bees have to be present
    // Bees have to be present in front of the moving bee (this)
    // Select 3 contacts infront of this bee as per [1]
    // Ex. Direction = 2 (N), neighbour bees to socialize are at NW, N, and NE
    socialize() {
       return (uniform(0,1) < 0.4) AND (bee in front of bee)
    }
}
```

Model Description

The cellular automata model presented in [1], was taken to further study the interaction between bees within an environment. Instead of controlling the bees through temperature, the cellular bee model is used without feedback from the CASU robots. As temperature is still important for the interactions of the bees and how long they socialize, the temperature of the environment is controlled by the bees themselves. The simulated bees emit their own change by modifying the temperature of their immediate surroundings (neighbourhood) by one degree celsius. Another important change that definitely affects the results, is the change of the social parameter and its function. Before the social parameter was used to pick a random number of neighbours around the bee, always including the bee infront if there be. The social parameter used in [1] was set to three, meaning the cell in front and two other cells in the neighbourhood were chosen to determine if the bee is capable of socializing with its neighbours. In the case of this simulation, the three neighbour cells chosen for examination of the presence of a bee will always be the cell the bee is planning to travel to and the cells clockwise and counterclockwise of the front cell. These three cells will be used to determine whether the bee should stop to socialize with the stopping probability of the bee. This decision was made because choosing a set of random neighbours would be difficult in CD++ without the chance of duplicates. Therefore, the bee checks the cells directly in front of it.

If the bee begins to socialize, then the bee will wait inside the current cell for a period of time that is dependent on the temperature of the cell. The equation to represent this time is given by equation 1 in [1]:
`wait time = (power(3.09 - 0.0403 * TT,-27)/(power(3.09 - 0.0403 * TT,-27) + power(1.79,-27))) * 22.5 + 0.645 `
TT represents the double digit temperature of the cell. The values that make up this equation come from parameters presented in [1]. The function is a sigmoid function which results in lower delays (1s) for temperatures closer to 28 degrees celsius and higher delays (25s) for temperatures around 38 degrees celsius. Therefore, as the temperature of the cells increases the socialization time of the bees increase in that region. The only way in this model for the temperature to increase is for bees to group together. Clusters of bees form during simulation that support this claim.

While the bee is not socializing it will be performing a random walk through the 2D plane. This random walk is quite simple. The bee will continue to move in a direction without disruption and with a small chance to change its direction. Most of the time the bee will continue its walk until it runs into another bee and begins to wait. Again, this will form clusters of bees in the plane which is similar to what the bees did in [1] around the CASU robots with higher temperature.

Simulation and Results

The parameters used in the simulation to study the behaviour of the the bees are as follows:

1. Simulation time: 2 minutes
2. Number of bees: 25
3. Grid: 50x50 cells

To place the bees on the grid, the `generate-vals.py` script was used to generate the bees at random coordinates. From there, the simulation was executed using the simu.exe file provided with CD++ to generate the necessary outputs to draw and graph the simulation using the online viewer and the graphing executable.

In the simulation described above, the bees start in random places on the plane, but then begin to run into each other early on (<10 seconds). Clusters of bees begin to form and the bees begin to socialize. At this point bees that are still performing their random walk will run into the originally formed clusters. Once the social time is over, bees will wander off, beginning their random walk again, but usually with other bees very close, so the chance of another social event is high. Clusters on the plane seem to either move across the plane or are stationary if the cluster is large enough, since there is a large 'net' of bees that catch other wandering bees. The movement of the clusters of bees is the main interest of the simulation because the formation of a swarm is the main objective.

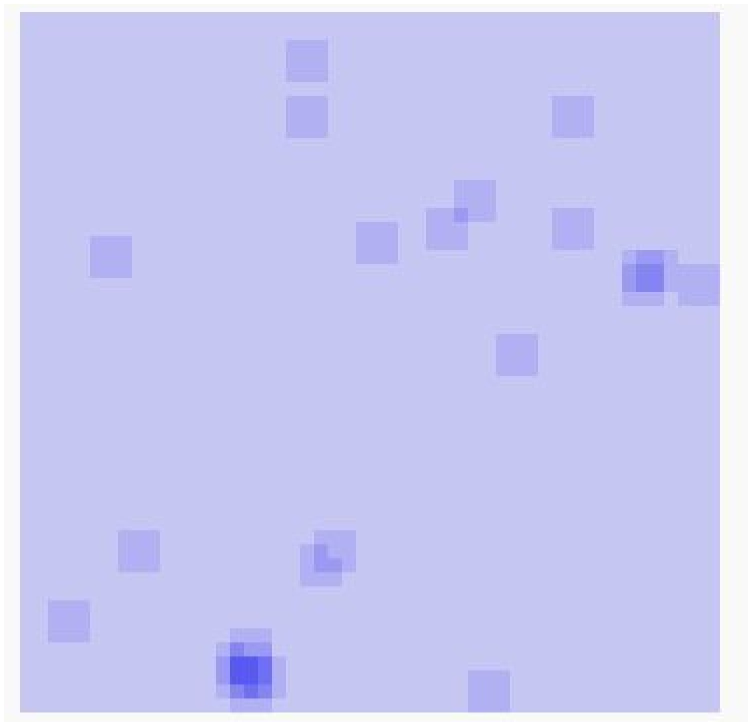

Figure 1: The output of the CD++ online viewer

Figure 1 shows an example timestep of the simulation. Each darker square represents a bee as the palette depends on the temperature of the cell. Since each bee emits a one degree difference in all of its neighbours, the bee can be seen from the difference of the ambient temperatures. In the bottom left, a large cluster can be seen forming. The darker colours represent a higher temperature since the group of bees produce a larger change on their environment together.

Future Work
Now that the simple bee nature has been captured in a 2D environment, like a hive, the project has two more potential directions:
1. Study the movement of swarms with a higher number of entities in 2D and even 3D space
2. Attempt to replicate the hive structure of an actual bee hive, including workers, drones, queens, larva, food, and an entrance/exit for the hive.

If the movement of swarms is further expanded upon, then interesting experiments can be done like prey versus predator analysis, since the movement of a predator greatly depends on the movement of its prey. In the case of this simulation and model, the clustering of bees was partially controlled by the temperature of the surrounding. This temperature factor could also be exchanged for several others like mating and food swarms. The presence of other materials, such as food, or entities may control the swarms behaviour.

Simulating a beehive would also provide valuable information into the structure of the hive and how bees interact within it. Studying the beehive could provide insight into the bee's structure and behaviour, and aid beekeepers and scientists in their understanding of the creatures they work with.

References

[1] M. Stefanec, M. Szopek, T. Schmickl and R. Mills, "Governing the swarm: Controlling a bio-hybrid society of bees & robots with computational feedback loops," *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, 2017, pp. 1-8, doi: 10.1109/SSCI.2017.8285346.

[2] Szopek, M.; Stefanec, M.; Bodi, M.; Radspieler, G. & Schmickl, T. “A cellular model of swarm intelligence in bees and robots,” *10th EAI International Conference on Bio-Inspired Information and Communications Technologies,* 2017, pp11-18, doi: 10.4108/eai.22-3-2017.152396

[3] Ocko Samuel A. and Mahadevan L. “Collective thermoregulation in bee clusters,” J. R. Soc. Interface.1120131033, 2014, http://doi.org/10.1098/rsif.2013.1033