ABM – Week 6 – Seminar – LVL I

Purpose

This task will allow you to explore the concepts of agent interaction and agent memory. You will write code to detect agent interactions, use lists to represent agents' memory of these interactions, and allow agents to change their strategy based on their memory.

Model

Open the model Ringroad.nlogo. This model represents cars driving around a closed circuit of road in both directions.

Take 5-10 minutes to look at the code and think about the following things:

- How does the model differentiate road patches and non-road patches?
  Do the patches have a variable for this? Or is it done some other way?

- Which netlogo primitive (code word) is used to create turtles in this model?
  You have often used crt or create-turtles; is that used here?_

- The choose-direction procedure allows turtles to follow the road.
  What would happen if you replaced "(neighbors4 in-cone 1 300)" with:

  a) "(neighbors in-cone 1 300)" ?
  b) "neighbors4" [i.e. removing in-cone 1 300]

  In each case, try changing the code and running the simulation slowly to see how it is different. Perhaps reduce the number of turtles to make their behaviour clearer.

Task

Give the agents a variable defining whether they drive on the left or the right (their 'driving side'. If two agents who have chosen different driving sides meet, while travelling in opposite directions, they will 'crash' (but in this simulation, they should continue their journey); e.g. a "left" driver meeting a "right" driver will 'crash'.

Adapt the code to detect such 'crashes'. Give the agents a memory of the driving sides of all agents they have encountered. Allow agents to adapt their own driving side based on this memory. Plot a graph of the proportion of agents using each driving side over time.

Simple Extension

Change the road layout and investigate how this affects the behaviour of the system.

Step-By-Step Guide

After each change you make, check that the code runs before you move on to the next step…

Giving all turtles a "left" or "right" driving side

1. Add a turtle variable to turtles-own to store this information

2. Change the turtles-setup procedure to:

    a. Set all of the turtles driving side to "left"
    [Useful primitives: ask, set.]

    b. Select half of the turtles and reset their driving side to "right"
    [Useful primitives: ask, n-of, set]

Your simulation should now run as before, but all turtles will have a "left"/"right" driving side variable. Run the simulation and inspect some turtles to see if this is the case.

Identifying crashes

3. Create a new procedure called check-crash.

4. Add this procedure to the 'ask turtles' block in the go procedure.

In the new procedure…

5. … write a line that creates a set of turtles called 'potential-crashers'. This should be the set of all turtles in the cell whose direction (heading) is not equal to the direction of the current turtle.
[The line should begin 'let potential-crashers '
Useful primitives: turtles-here, with, heading, of, myself]

6. … write a line that creates a set of turtles called 'actual-crashers'. This should be the set of all potential-crashers in the cell whose driving side is not equal to the driving side of the current turtle.
[The line should begin 'let actual-crashers ' and should be very similar to the line written at the previous step.]

7. … use an if block to check whether there are any actual-crashers. If there are, ask them and the current turtle to turn yellow.
[Useful primitives: if, any?, set, color]

8. Go back to the 'ask turtles' block in the go procedure. Make the turtles turn red before they do anything else (otherwise crashers stay yellow forever).

Your simulation should now run as before, but crashes will be visible, because all turtles that crash will briefly turn yellow.

Giving turtles memory of encounters

9. Add a turtle variable to turtles-own to represent their memory.

10. Set this memory as an empty list (just like in Python) in the turtles-setup procedure.

11. In the check-crash procedure, write an 'ask potential-crashers' block to make all the potential-crashers add the driving side of the current turtle to their memory.
[Look at the documentation for lput, which adds an element to a list.
Useful primitives: set, lput, of, myself]

12. (*A little tricky*) Within the same 'ask potential-crashers' block, write an 'ask myself' block to make the current turtle add the driving side of each of the potential-crashers to its own memory.
[The code in this block will be identical to the code written in the previous step.]

Your simulation should now run as before, but turtles will build up a list of terms, either "left" or "right", recording the driving side of all the turtles they have encountered.

Making turtles adapt their driving side based on their memory

13. Create a new procedure called choose-drive-side.

14. Add this procedure to the 'ask turtles' block in the go procedure.

15. In the new procedure, use an if block to check whether the turtle's memory is empty. If not, set the turtle's driving side to the most common driving side that it has encountered.
[You will need to use the 'modes' primitive, which finds the mode of a list. However, this produces a list (because sometimes there is more than one mode, e.g. if the turtle has encountered an equal number of "left" drivers and "right" drivers. This means you need to set the driving side to a randomly chosen one of these modes.
Useful primitives: if, not, empty?, set, one-of, modes]

Your simulation should now run as before, but turtles will change their driving side over time.

Creating a graph to monitor the proportion of turtles driving on each side

16. Create a plot in the Interface. Use a pen update command to plot the proportion of turtles driving on the left.
[You need to count the turtles driving on the left and divide by the total number of turtles.
Useful primitives: plot, count, turtles, with]

17. Give the plot an appropriate title and axis labels. Set the axes to an appropriate scale.

Your simulation should now run as before, but the plot will track the proportion of turtles driving on the left as time passes.