# Multi Agent Systems

# Assignment 2

In this assignment you will implement a simple "Deductive Reasoning Agent", as introduced in Chapter 3 of the book. For now, this agent will only show 'reactive behaviour' (see Section 5.1), i.e., it will base its actions directly on its observations, without making use of any internal states.

Using NetLogo, implement the 'Vacuum World' and the behaviour of the 'Cleaning Robot' described in Chapter 3. This agent follows a fixed, pre-defined path through the grid and cleans all dirt found in the grid. Each square of the grid is a place that may contain dirt or not. For each new simulation, the cells with dirt should be randomly defined. At the end of the simulation the robot must have cleaned all the dirt. For more details, check the description on p.52-54 of the book (or, in case you are using the first edition, p.51-53). Note that Tutorial #3 from the NetLogo user manual can be used as inspiration.

As a basis for the assignment, a NetLogo template is provided via Blackboard. This file includes some setup methods (to initialize your model) and some go methods (to execute it). However, you are free to include more methods if you consider this necessary.

### 2.1 Specific Solution [2 points]

Implement the specific solution given on page 54 for a grid of 3x3 cells. Here, the movements of the agent are a sequence of hard-coded commands to cover the entire area in the prescribed order (this solution could be specific for a 3x3 grid). Once the agent reaches position (2,2), it can stop moving (hence, it does *not* have to travel back to position (0,0)). Make use of the predefined slider interface item to control the random creation of dirt and the monitor interface item to display the number of remaining pieces of dirt (i.e., showing how many cells are grey).

Tips:

- Use the property pcolor of patches (grid) to visualise which cells contain dirt (by making them grey) and which do not;
- For this assignment you can move the agent with a sequence of commands setxy.

## 2.2 Generic Solution [3 points]

Extend Assignment 2.1 in such a way that it works for any arbitrary grid of NxM cells. Test this, for instance, by setting the grid size to 20x30. The agent must stop when it has visited (and cleaned) all the cells.

Tips:

- When you create the agent (create-turtle command), you must define its coordinates on the grid, but you could also define the direction that its head is facing. One way to do this is using the command facexy as a parameter of the create-turtles command. For example: create-turtles number [setxy min-pxcor min-pycor facexy min-pxcor min-pycor + 1].
- There are many alternatives to determine when the agent should stop. One of them is by checking its distance to the end of grid (command distancexy), or by checking if it is located on the last cell (command turtles-here). Another alternative is to count the visited cells and compare this with the total number of cells (see variables max-pxcor and max-pycor).

## 2.3 Avoiding Obstacles [5 points]

Now part of the grid contains *obstacles* (i.e., cells that the agent cannot access) that are generated randomly at the start. These cells should be given a different color (e.g., black). You can create a slider interface item to control the percentage of obstacle cells. Next, change the strategy from Assignment 2.2 into the following strategy:

- If the cell in front of you is an accessible cell (with or without dirt), then:
    - in 80% of the cases: go to that cell
    - in 20% of the cases: turn either left or right (determined randomly)
- If the cell in front of you is an obstacle (or a wall), then either turn left or right (determined randomly)

You will see that, as long as there are not so many obstacles that the agent can be 'locked in', this strategy will eventually lead to all dirt being cleaned. Nevertheless, you will see that this is not the most intelligent strategy, and it may take a long time to clean all cells. Moreover, note that this simulation will never stop, because the model does not have any memory to keep track of visited cells or the amount of pieces of dirt cleaned.

Tips:

- The cell where the agent starts must be a free cell, not an obstacle!

***What to hand in?***

You have to develop three separate NetLogo models, which correspond to the following assignments:

- Assignment 2.1 [2 points]
- Assignment 2.2 [3 points]
- Assignment 2.3 [5 points]

Please include the three NetLogo models within one .zip-file, and submit this file (as a group) via Blackboard.