Knowledge and Agent Systems
Assignment 1

Deadline: September 11th 22:00

Report requirements

- Hand in your updated *KAS_assignment1.nlogo* code, the *sugar-map.txt* file, and a PDF report with your answers to the questions below.
- Include your student numbers and names in the code and the report itself.
- If you have to write code, also add the small snippets of code to the report. You can use the *listings* package for this. In addition, explain the design choices of your code and how it solves the associated problem.

Prerequisites

For this lab you will need NetLogo [1]. It can be downloaded at `https://ccl.northwestern.edu/netlogo/download.shtml`. You do not have to fill in the form on the download page - clicking 'download' will work regardless. On Linux, you will have to extract the NetLogo archive and run the NetLogo executable inside. On Windows, you will have to run the installer.

How to run the code

After opening NetLogo, click on File > Open... and then select the *KAS_assignment1.nlogo* code. You will see a user interface. On the left side you can change the parameters and on the right side you can see output graphs. Click *setup* to initialize the model. Click *go* or *go once* to run the model. In order to see and change the code, go to the *Code* tab on the top left.
Please note: the code only runs if the *sugar-map.txt* file is in the same folder.

Introduction to the assignment

The goal of this assignment is to explore a simple multi-agent system that models hunter-gatherer behavior. The model has been implemented in NetLogo [2]. For programming pointers, look at the quick introduction to NetLogo on Brightspace or the NetLogo dictionary at `https://ccl.northwestern.edu/netlogo/docs/dictionary.html`.
In this model, the environment consists of squares, called *patches*, each with a certain amount of food. The *turtles* in the environment move around, consume the food that they find and use energy for 'living' (in other words: they have a metabolism). Turtles die when their energy level drops below a certain level. Turtles choose where they go. They look around, within their range of vision, and pick the best site.

1 Exploring Sugarscape

1.1 Turtle Variables (1 point)

The turtles in Sugarscape can be considered to have three primary custom turtle variables: `sugar`, `metabolism`, and `vision`.

Describe and explain for each of these variables individually their effect on the simulation with lower and higher values. Include low-level explanations for the high-level observations. You can modify the range of these turtle variables inside the `turtle-setup` procedure.

1.2 Estimating the Carrying Capacity (1 point)

As resources (sugar) are limited, there is a limit to the number of turtles that can be sustained by the environment. In ecology, this limit is called the *carrying capacity* of the environment.

Use an initial population of 400 turtles. Initialize the turtle variables as in Table 1 using the custom `random-in-range` procedure.

turtle variable	range
sugar	[5,25]
metabolism	[1,4]
vision	[1,6]

Table 1: Parameter ranges to use for estimating the carrying capacity.

Determine the number of turtles that can stably survive in the environment using the parameters above. Provide a mean, standard deviation, and the number of runs you performed.
Also, explain why it is important to aggregate the carrying capacity over multiple runs. What are the sources of randomness in this model?

2 Sugarscape, in theory

This part of the assignment contains some exercises related to the first lecture to help you reflect on Sugarscape.

2.1 Reactivity vs. Proactivity (1 point)

There are three proposed behavioral properties that are desired of agents in multi-agent systems: reactivity, proactivity, and social ability.

To what extent do the turtles in Sugarscape balance reactive and proactive (planning-based) behavior?

2.2 BDI (1 point)

Agents can be modeled as having beliefs **B**, desires **D**, and intentions **I**.

What are the turtles' **BDI** in Sugarscape?

2.3 Agents? or Objects? (1 point)

One key difference between agents (in agent-based programming) and objects (in object-oriented programming) is that agents are assumed to have more autonomy, in the sense that they choose for themselves whether to perform an action when requested to do so (possibly making use of their **BDI**). In contrast, objects will always execute their methods when requested. For instance, when calling `model.fit(...)` on your machine learning model, it will always fit its parameters to the given data.

Are the turtles in Sugarscape more like objects or actual agents? You can support your answer using your answers from previous questions.

3 Create Your Own Society

In this part of the assignment, you will extend the provided model. Keep in mind:

- Each extension should be independent and self-contained, so answer the questions separately.
- Interface items have been added which you should use to enable or disable your extensions. They work as global variables.

3.1 Emergence? (1.5 points)

The turtles are currently initialized all over the grid. Add code to initialize turtles in the third quadrant instead. Each agent should still occupy a unique patch. Example output is visible in Figure 1.

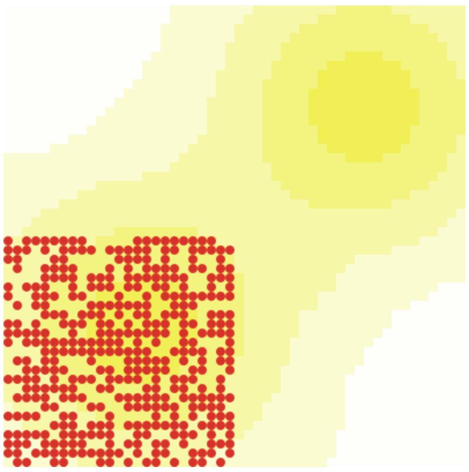

Figure 1: Turtles initialized in the third quadrant.

Use parameters as in Table 1 and experiment with the initial population size. What emergent behavior do you observe? Include a figure of the environment that highlights the behavior. Also, what effect does the initial population size have on the emergent behavior? Provide an explanation.

3.2 Socializing Agents (1.5 points)

The turtles in Sugarscape lack social ability.

Describe a possible way how the turtles could share their resources (sugar). Implement your idea inside the `turtle-share` procedure. Consider how much a turtle can share, and with whom they choose to share their resources. Make sure to motivate your design choices. For example, a turtle should probably not be able to share sugar with a turtle outside their vision.
How does your resource sharing strategy change the model dynamics (e.g., equilibrium population size, wealth distribution, spatial distribution)? Support your answer with relevant output obtained from the model, and mention the size of the initial population you used.

3.3 Aging Agents (2 points)

A biologist looks at your model and correctly points out that people are not (yet) immortal. To address their concerns, we want to extend the model such that turtles may not only die of starvation, but also of old age.

When a turtle is born, it is randomly assigned a maximum age at which it dies. Each turtle has a maximum age between 60 and 100 years, where one tick corresponds to one year. To keep the population size constant, a new turtle is born as a replacement when a turtle dies. This newborn turtle should be born on a random unique patch without inheriting any traits or wealth (i.e., it should be initialized with `turtle-setup`).

Use an initial population of 250 turtles and initialize the other parameters as in Table 1. Include a histogram of a typical resulting wealth distribution. Describe what changes have occurred to the wealth distribution with this extension. Also, provide a brief explanation for these changes.

References

[1] Wilensky, U. (1999). NetLogo. `https://ccl.northwestern.edu/netlogo/`. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

[2] Li, J. and Wilensky, U. (2009). NetLogo Sugarscape 2 Constant Growback model. `https://ccl.northwestern.edu/netlogo/models/Sugarscape2ConstantGrowback`. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.