Manual: Application for Design of Multi-Agent Systems (Beta)

Group B14

October 4, 2020

Preamble

For this project we decided to set the goal for the Beta to be including most of the core functionality. For the final, upcoming version, we will focus mainly on refactoring of the existing code without expansion of features (outside those that focus on making the interaction easier/more useful for a user). We are well aware of some inadequacies within the code, such as the lack of separation between physics simulation and information spread simulation within the nodes and the graph itself, and will resolve for the final deadline.

1 Introduction

This document details how to use the application designed by group B14 for the course Design of Multi-Agent Systems. The goal of the application is to model the spread of information through a network (such as a social media website), modelled using insights from psychology, where we integrate simulated cognitive dissonance in the decision making of agents.

2 Compiling & running the program

On a linux-based operating system the application can be compiled from the terminal using the provided build shellscript, located in the root. For this, it is required that the java development kit (JDK) has been already installed on the computer. After creating the jar file, it is advised to run the program from the terminal using:

```
java -jar graph.jar
```

The reason for this, is that it will then treat the location of the jar as the root folder for the application, which then will create a folder there for data output. If it is ran as a standalone application, the file root used will be somewhere else entirely.

3 Using the program

By default the program opens in headless mode. This mode allows the execution of simulations without requiring a visual representation, which saves some resources.

3.1 Showing the simulation & view controls

Headless mode can be toggled via the "Toggle headless" option in the view menu. If toggled, it will switch to a view of the current model present in the simulation. In order to allow the nodes to space out better, the physics can be toggled from the physics menu. This will force the nodes to space out properly by applying varying forces such as a gravitational pull, spring forces on the edges, and a push force in-between nodes. Furthermore, the gravitational pull force can also be changed from the physics menu if so desired.

3.1.1 View control

The following actions are available for interacting with a view of the model itself:

1. *Scrolling:* Zooms in or out.
2. *Left-clicking & dragging (on background):* pans the view
3. *Left-clicking on node:* Displays information about the selected node, and colours it black
4. *Right-clicking on node:* Displays the same information that a left-click would give, but in addition, also isolates the clicked node & its neighbours in the view.

In addition to this, a couple of options are also available in the view menu:

1. *Toggle show IDs:* Toggles whether or not to draw node IDs. Best left on off to reduce visual clutter.
2. *Show belief:* Colours nodes based on the belief they currently hold. Dark colours indicate a belief close to 0.5. If the belief is above that threshold, it will become coloured red. Below, it will become coloured blue. The more intense the colour, the stronger the belief (i.e. closer to the extreme values for belief of 0 and 1).
3. *Show dissonance:* Shows the nodes that are experiencing dissonance above their threshold as green nodes, and the nodes that are not experiencing dissonance as black. Note that the edges still retain their regular colour.
4. *Reset zoom:* Resets zoom to default.
5. *Pull camera to node:* In the case of having lost the graph, this option will pull the camera to a node in the graph.

3.1.2 Edge Colours

Let $b(n)$ denote the belief of node n. Given an edge e, which connects nodes n_i and n_j, the colour c is decided as follows:

$$c(e) = \begin{cases} \text{Red} & \text{if b}(\text{n}_i) \geq 0.5 \text{ and b}(\text{n}_j) \geq 0.5, \\ \text{Blue} & \text{if b}(\text{n}_i) < 0.5 \text{ and } b(n_j) < 0.5, \\ \text{Black} & \text{otherwise.} \end{cases}$$

3.2 Model simulation

Of course, performing simulations is the most important feature of the application. Under the Sim Control menu, the following options are available:

1. *Reset Model:* Allows the user to create a new randomly initialized model. Will ask the user how many nodes the new model needs to have, and creates it accordingly.
2. *Update dissonance network wide:* Applies the same dissonance update to the entire network. Will ask the user to specify what value to add to the dissonance for each agent. Mainly used for testing purposes.
3. *Update recommendation strategy and set size:* Sets the recommendation strategy that should be used to fill every agent's recommendation set. At every step agents consider information not only from other agents directly connected to them but also from agents that are in their recommendation set. Possible strategies are "polarize" and "random". The former fills the recommendation set with agents that have a belief close to the agent's belief for which the recommendation set is currently created. The strategy ensures that the agents to be recommended are neither directly nor indirectly (via another agent) connected to the current agent. The second strategy ("random") selects agents randomly from the network. Additionally determines the maximum size of the recommendation set. Note that an agent will also connect to all agents within the recommendation set that have an belief close enough to the belief of the current agent and if doing so does not violate the total number of connections per agent permitted by the connection limit.

4. *Update connection limit for nodes:* Sets the maximum number of connections that each node is permitted to have. If it is changed such that the new limit is lower than the old one, nodes above this limit will remove random connections until they fit within the new limit again. It should be noted that this behaviour mainly exists to guarantee that at any point the number of neighbours of each node is less than or equal to the limit, and typically this limit should be kept constant for the entire duration of a simulation.

5. *Step simulation (spacebar):* Performs 1 simulation step on the network, consisting of spreading information, network pruning and finding new potential nodes to connect with. Can also be performed by pressing the spacebar key.

6. *Perform N steps on network:* Same as above, but will then execute n steps directly, where n is requested from the user when the option is selected.

3.3 Data output

Next to running the model, it may be desirable to collect data for further analysis. Options that deal with this can be found under the Log generation menu. By default, data output is off, to allow the user to play around with the application without generating large data dumps. Data is stored in the data_out folder that is generated in the root of the application.

The data that is generated is stored in a folder whose name is decided by the current time, which houses a csv file with relevant variables, to be analysed further, and an image folder that holds images if image data is generated.

1. *Enable data output:* Will enable data output. When enabled, data will be saved on the simulation step performed.

2. *Disable data output:* Disables data output. Disabling & re-enabling will cause the generation of a new folder for data output.

3. *Enable/configure image output:* Turns on image output generation. The user will be requested to provide per how many frames an image should be captured. When this option is selected, be patient! The image capture logic first waits for the physics to be settled to create a neater image, and in the first time steps this might take quite some time. There is a time-out in place, so it is guaranteed to terminate.

4. *Disable image output:* Stop generating images as output.

5. *Set. max avg. movement (capture):* As aforementioned, the image capture first waits for the physics to settle a bit before capturing any output. It does this by checking whether or not the average velocity per node in the network falls under a pre-specified threshold. Currently, we have not managed to find a rule to generalize such a rule that works well for each size of network, so the user can change this if need be. The current initial max avg. velocity under which the network must fall has been set to a value that works well with networks of around one thousand nodes. If a larger network is simulated, this threshold needs to increase, and if a smaller network is simulated, decrease.