Distributed Artificial Intelligence - Intelligent Termites

Daniel Ruiz, Albert Royo

November 2020

UNIVERSITAT DE BARCELONA

Contents

1 Introduction

In this practical we make the transition from StarLogo to NetLogo. As an introduction to the new framework, we are comissioned with implementing a population of ants (the quantity of which will be variable, regulated by means of a slider) that group together using pheromones.

Said simulation must allow the user to configure the following parameters:

- Population: defines the number of ants at the beginning of the experiment.
- Evaporation: percentage of the pheromones that will disappear each iteration.
- Diffusion: percentage of the pheromones that travel to the neighbouring patches.
- Smell-range: maximum amount of patches at which the agents can detect the pheromones.

2 Implementation explanation

In this section we will explain the code, separated in functions, using screenshots of the NetLogo framework.

2.1 Reset

```
to reset
  clear-all
  create-turtles numberofants [ setxy random-xcor random-ycor ]
  ask turtles[set shape "bug" set size 2]

  reset-ticks

end
```

The reset function, linked to the *reset* button, allows us to both set up the environment for the first time and restart it as many times as we want.

It starts by clearing all of the elements of the simulation, leaving it clean. Then it creates the amount of turtles specified by *numberofants* (controlled by the slider of the same name), modifies their size and appeareance, and resets the tick count.

After this process, the simulation is born anew.

2.2 Go

```
to go
  move-ants
  leave-pheromones
  pheromones-dispersion
  pheromone-evaporation
  color-patch
  tick
end
```

The go function, linked to both *Go (1 step)* button ans *Go forever* button, regulates the behaviour of the model.

Following the indications of the practical document that was provided to us, the simulation works as it follows: first, the ants move; after moving, they leave two pheromones on the patch that they are standing on; said pheromones will first disperse and then dissipate (amount of both events regulated by means of a slider); then, the patches will have to be recolored accordingly.

To make the code more readable, intuitive and modificable, each of these individual steps has been placed in a different function which will be called by the main one, Go, very iteration of the simulation. Their behaviour and implementation is explained below.

2.3 Move-ants

```
to move-ants
  ask turtles[
    let possible_patches
      (patch-set (patch-ahead smell-range)
      (patch-left-and-ahead 45 (smell-range + sqrt 2))
      (patch-right-and-ahead 45 (smell-range + sqrt 2))) ;;Create a list with the 3 patches
    face max-one-of possible_patches [pheromones];; go to the patch which has more pheromones,
    forward 1
  ]
end
```

This function determines the movement of the ants based on the amount of pheromones they are able to detect.

First, we create a list with the pheromone value of the three relevant patches placed at *smell-range* distance (in front of the ant, in diagonal towards its right and in diagonal towards its left, using $\sqrt{2}$ for that purpose since that value times the squares side will provide the diagonal). Then, we pick the square with the highest value, and make the ant face it. After that, the ant moves one square in that direction.

2.4 Leave-pheromones

```
to leave-pheromones ;; controls the pheromone increase in turtle occupied patches
 ask patches [
   if any? turtles-here[
    set pheromones pheromones + 2
    ]
 ]
end
```

After handleing the ants' movement, it is time to handle the pheromones left by the ants presence. We simply as all the patches with an ant on top of them to increase their pheromone count by two.

2.5 Pheromone-dispersion

```
to pheromones-dispersion ;; should be called each tick to disperse the pheromones
    diffuse pheromones diffusion
end
```

Then it's time to handle the pheromone dispersion. Pheromones will disperse to their neighbouring patches, with the dispersion amount regulated by the *diffusion* slider (1 meaning full diffusion and 0 no diffusion), meanening that pheromones are able to travel through space on their own.

For that purpose we use the command *diffuse*, a NetLogo native tool, that implements said spreading given a porcentual number. We only have to ask every patch to apply that function, and the pheromone dispersion would be complete.

2.6 Pheromone-evaporation

```
to pheromone-evaporation ;; controls pheromone evaporation
  ask patches [
    set pheromones pheromones - (pheromones * evaporation)
  ]
end
```

A percentage of the pheromones, modulated again by a slider, is meant to evaporate through every iteration of the simulation. With that purpose in mind, we ask every patch to lower its pheromone value according to the percentge stated by the *evaporation* slider. The higher the *evaporation* valie is, the shorter the pheromones will linger in the air. A value of 1 will mean no pheromones survive each iteration.

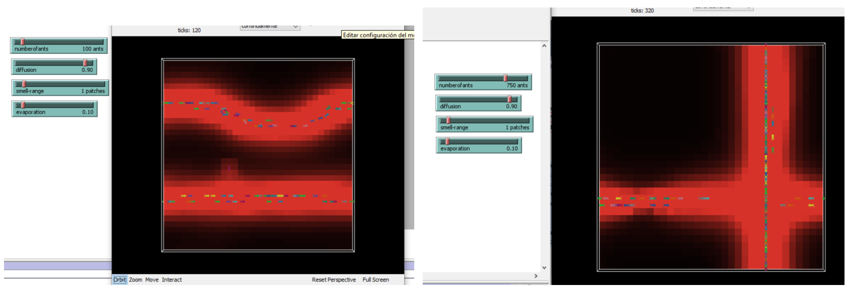
2.7 Color-patch

```
to color-patch ;; recolors the patches given their new pheromone value
  ask patches [
    ;; We create a variable "red_channel" which has the value of 200 times pheromones. This is just so we can see it better.
    let red_channel pheromones * 200;;(1000 / count turtles)
    if red_channel > 300
    [
      set red_channel 300 ;;If the intensity is higher of 300 just set it to 300
    ]
    set pcolor 10 + (5 * (red_channel / 300)) ;; We get one of the 5 red tones deppending on the intensity
  ]
end
```

This function is called in last to recolor the patches according to their new pheromone value. It has no effect on the ants behaviour, but it helps the user understand what is going on in the simulation.

3 Experimentation

When the implementation was finally done, we soon realized that ants tend to group in one or two diferent paths and then just go into a infinite loop. We simbolize this as the objective of grouping themselves is completed and mesasuring the ticks to get a good metric, preferring, a little bit more, the results where the there is only one path or, two crossed paths, not the ones with parallels paths, as seen below:



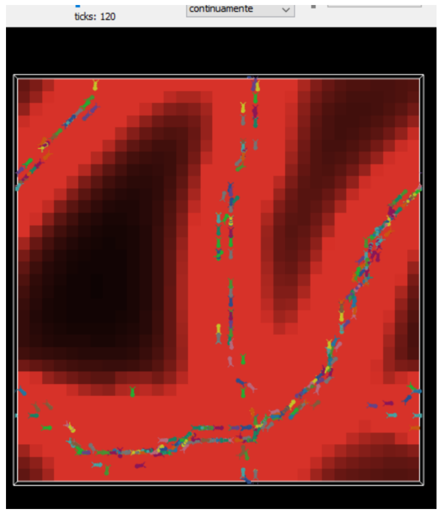
Then we tried the default parameters varying only the population variable and emulating until an infinite loop, 10 times with each value and measured the number of ticks.

We started with with only 20 ants, and as we spected, the loop didn't came the loop until arround tick 600, and in the majority of cases, with some of ants lost alone around the grid. This is quite a bad result.

Then, we proceed with 50 ants and start seeing the expected result around tick 320.

Now, the ants will start forming this groups soon and this will be deffining in the results. With 100 of population, the loop comes around tick 210. Then, with 200 population we get the best result that is a greate mean of 195 ticks to arrive at the infinite loop.

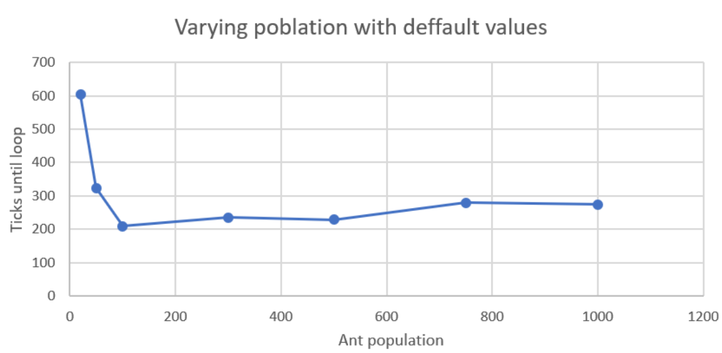
Apart from now, the number of tick will increase becouse the big numbre of ants makes various big formations that vary their paths separating and joining each other in very a dinamic way, but not defining a fix path.



By the way, we can also consider the previous a good result, as "the chain" is alrredy formed.

3.1 Conclusions

With this results ploted in the next graph we can see that probably with a bigger populations the ticks won't stop increasing. It is true, but, on 100 and 200 cases there were many results where we ended up with two different parallel paths and, as loong as we increased the population this was not very common.



Also, we tried varying othe parameters rather than the population with the perfect one(200) and also with the topest(1000).

For example, lowering the *evaporation* or riseing the *diffusio* always helped. We could observe that the result was almost always a unique thin path and a little bit better(less ticks).

On the other hand if we rose the *evaporation* rate, it unexpectedly went very well with large amount of population.

The most noutorios variable is the smell range. With almost every amount of population and other vairable, if the smell range is at 4 or 5 paches, the result only takes less than 100 ticks to become *static*, making it the best ability to upgrade.

4 Future work

We also tried making a bigger grid with non-warpped enviroment to measure the values in a different metric but did not went very well with the border colisions. We could, but, apreciate a more swarm behavior in the bigger grid.

Another good implementation would be making the ants go to a concret spot/or not, on an enviroment and make them come back with resources, for example.