Practical work of IMAS

Practice goal

The main goal is to put in practice the basic concepts of agent technology using a real framework (JADE).

Practice description

You have to design and develop a **decision making system** for real complex situations based on a multi-agent system**.**

This exercise aims to simulate an efficient cleaning of a city. There are two kinds of entities: (i) scouts, that detects where the garbage is, and (ii) harvesters that are responsible for harvesting the garbage and disposing it into recycling centers.

To make the problem finite, we locate this scenario in a city, seen as a grid of variable size, where a limited number of elements and allowed interactions appear. Both scouts and harvesters will have a visual range surrounding their position. We will consider the city divided into a set of grid cells. Within this area there are **buildings** (which may contain garbage), streets where scouts and harvesters may be located, and **recycling centers**. We will add the constraint that in a certain street cell there can only be a unique vehicle (scout or harvester) in each moment.

There are three basic kinds of **actions** that will have to be done: **garbage detecting**, **garbage harvesting** and **garbage recycling**. A certain building may contain some units of garbage. This **garbage** may be of different kinds: **plastic**, **glass** or **paper**. A concrete building only contains one kind of garbage. For example, it could contain 5 units of plastic garbage. This garbage must be detected by exploring the map. Once detected, it must be harvested and then disposed in one of the available recycling centers. Each recycling center only deals with some of the kinds of garbage. The recycling centers give a number of points for each unit of garbage disposed. For example, there could be a recycling center that gives 2 points for each unit of plastic garbage and 3 points for each unit of paper. This recycling center does not deal with plastic and glass garbage. For each kind of garbage, there will be at least one recycling center in the city that accepts it.

The the location of the garbage is **not known** a priori. Each vehicle will have a visual range limited to the cell where it is located plus the 8 cells surrounding it. The vehicles will have to move, horizontally or vertically, through the street cells in order to **explore the city**. They will communicate to combine the information found so that the position of the garbage is known as soon as possible.

Each scout and harvester will have its corresponding agent (**scout agent** and **harvester agent**, respectively). These agents keep their current position and their internal state (for example, a harvester keeps the number of garbage units it is carrying and their kind) and make their own decisions or wait for orders from their managers (depending on the design of the multi-agent system). Scouts and harvesters have different tasks to perform:

- **Scouts:** They explore the map. These vehicles will move through the city discovering the buildings that have garbage. They cannot harvest garbage. They can move, horizontally or vertically, 1 cell per turn.
- **Harvesters:** They harvest garbage and bring it to a recycling center. They can move, horizontally or vertically, 1 cell per turn. In order to harvest garbage, they must be situated in a cell adjacent to the building containing garbage (horizontally, vertically or diagonally) and remain there for some time (**1 turn per garbage unit**). Each harvester can harvest one or more kinds of garbage but it can only carry one kind of garbage at the same time. Moreover, harvesters will have a maximum number of units of garbage that they can carry. When they have harvested garbage, they can go to harvest in another building if the maximum number of units has not been reached or they can go to recycle this garbage. To do this, a harvester has to be situated in a cell adjacent to a recycling center (horizontally, vertically or diagonally) that allows the kind of garbage it is carrying and remain there for some time (**1 turn per garbage unit**). Several harvesters can be harvesting garbage from the same building or disposing garbage in the same recycling center at the same time.

The agents representing the scouts and the harvesters can be coordinated using at least a coordinator, that is, a **scout coordinator** and a **harvester coordinator**. However you can decide, if necessary, design other coordination methods based on several coordinators of different levels (e.g., **harvester coordinator**, **plastic harvester coordinator**, **glass**

harvester coordinator, etc.). The idea behind these coordinators is to have a map of the explored area and use it to make decisions.

For example, harvester coordinator, knowing that there is a limited number of harvesters, must determine how to assign the different harvesters to each one of the buildings with garbage, depending on the kind and amount of garbage, the capacity of the harvesters and their location in the map. Several strategies can be followed, like assigning 1 harvester/building, sending some harvesters to the same building in order to harvest its garbage earlier, sending a harvester to a recycling center which gives less points than another one but it is very much closer, decide routes for the scouts that do not interfere in the harvesting tasks, etc.

There is always a **coordinator agent** which centralizes the orders to be executed in each turn. This coordinator agent knows which are the changes that dynamically happen in the city (e.g., movement of the vehicles, garbage harvested, etc.). In order to simulate and control what is happening, there is also a **system agent** which executes orders to update the state of the world. This agent is the one that, in a nutshell, keeps the state of the city and shows it to the user using a graphical interface.

Technical requirements

Your practical work has to address this problem in two phases:

1. **Initialization.** This is the first phase in which your whole solution is correctly initialized, loading the initial configuration settings from a file, creating all the agents and showing them in their initial positions in the graphical interface.
2. **Simulation.** Once all agents are ready, the evolution of all events will make agents move through the map simulated by steps. At the end of each step, all mobile agents know their next position in the grid.

Architecture

To address garbage harvesting, you have to design and develop the following agents addressing the responsibilities being detailed:

- **SystemAgent.** This is agent is responsible for:
  - Loading the configuration settings at the initialization phase.
  - Spreading the necessary configuration settings and current status of the mountain to the CoordinatorAgent.
  - Providing a GUI where to show the map grid and all its static and mobile elements.
  - Adding new bargage in the map in a random and dynamic way:
    - Deciding if garbage has to appear in the next step. Probability of new garbage.
    - Deciding the number of buildings that will have new garbage.
    - Deciding the type of garbage per building (recall, a building contains only a single type of garbage at a time).
    - Deciding the amount of garbage units put in that building.
- **CoordinatorAgent.** This agent can comunicate with the ScoutCoordinator and the HarvesterCoordinator, providing them the information given by the SystemAgent.
- **ScoutCoordinator.** This agent is responsible for coordinating ScoutAgents (you can decide to add additional levels of hierarchy), and communicate with CoordinatorAgent.
- **HarvesterCoordinator.** This agent is responsible for coordinating HarvesterAgents (you can decide adding additional levels of hierarchy as described above), and communicate with CoordinatorAgent.
- **ScoutAgent.** These agents are responsible to discover where garbage is and inform about it. Since garbage appear dinamically, they have to be moving continuously to discover it as soon as possible.
- **HarvesterAgent.** These agents can bring several types of garbage, but only a single type of garbage at a time. They also have a maximum capacity of garbage units that they can bring.

In the following diagram you can see the basic architecture of the described multiagent system:

```
SystemAgent (loads initial configuration settings)
   ||
   ||
   \/
```

```
  CoordinatorAgent
  ||            ||
  ||            ||
  \/            \/
Scout          Harvester
Coordinator    Coordinator
  ||              ||
  ||              ||
  ??              ??  You have to decide if you add additional coordinators.
  ||              ||
  \/              \/

  N x             M x                                                    ^
ScoutAgent       HarvesterAgent
```

Information flow

The information will flow as described in the next lines:

- The SystemAgent will load the configuration settings, such as the number of agents of any kind (ScoutAgent and HarvesterAgents) and the information related to any single harvester (types of garbage that can bring and total capacity). In addition, configuration settings will define the whole simulation properties, including the size of the grid, the number of any kind of elements, and the total number of simulation steps, among others. When this step is done, simulation can start. SystemAgent will also communicate to the CoordinatorAgent the current status dynamically, such as which elements currently exist and their specific status.
- HarvesterCoordinator will be responsible for a good average quality of service for all harvesters.
- HarvesterAgents and HarvesterCoordinator will implement the chosen coordination model for harvesters and they will decide the actions for each harvester, including which harvesters goes for which garbage and to which recycling center every harvester has to go to dispose its current type of garbage.
- HarvesterAgents have to find the fastest path to every target place (i.e., for a building to clean it from garbage or for a recycling point). Harvesters have to inform to HarvesterCoordinator about the number of steps taken to get to a building and also to get to the recycling point, for statistical purposes. You can decide if a HarvesterAgent with a stated plan can change it to collect some other garbage before continuing its own path.
- The ScoutCoordinator and ScoutAgents will be responsible to decide the path to follow each scout. ScoutCoordinator also gathers the positions of the new garbage. This information will be passed to the HarvesterCoordinator and appended to the current set of detected garbage.
- ScoutAgents have to remain moving continuously to discover new garbage. When they discover some new garbage at a building, they inform to the ScoutCoordinator.
- When all coordination tasks are completed and statistical information sent, both HarvesterCoordinator and ScoutCoordinator will send a message OK to the CoordinatorAgent, which will forward it to the SystemAgent. Then, the SystemAgent knows that it can proceed with the next simulation step.

Any protocol message not described in this document will be defined by each student to make the system work as expected.

Statistical information

For analysis purposes, the SystemAgent shows the statistics gathered along the whole simulation process. They will show the quality of your multiagent system design as well as the life quality on the city. These statistics include:

- **Current benefits:** amount of points received for recycling garbage.
- **Average time for discovering bargabe:** the amount of time spent from garbage appearance until an scout discovers it.
- **Average time for collecting garbage:** the amount of time spent from gargabe discoverage until the first harvest gets to that point.
- **Ratio of discovered garbage:** the ratio of the garbage that is already discovered by scounts from the total.

- **Ratio of collected garbage:** the ratio of the garbage that is already collected by harvesters, including that in the harvesters and that already recycled in recycling centers.

In conclusion, your design should try to maximise the benefits from recycling at the minimum time to recycle garbage. There is a clear tradeoff between proximity and number of iterations for disposals.

Deadline and delivery

You have to deliver the solution to this exercise in two parts:

- **First delivery:** Mandatory delivery in all cases to pass the subject.
  - December 4th, 2016, with the initialization phase developed.
- **Second delivery:** You will have two evaluation opportunities for the whole practical work:
  - January 18th, 2017
  - February 2th, 2017

The student has to upload a zip archive with the solution to Moodle into the corresponding task. To do so, check the deadline explicitly stated into the corresponding Moodle tasks.

Also check that the zip archive contains the whole NetBeans project with clear instructions on how to run it. In particular, **professors will only proceed this way to evaluate each delivery**:

1. Download the zip archive.
2. Unzip it.
3. Load into NetBeans.
4. Run the project from within NetBeans.
5. Both graphical interfaces with the map and the RMA have to be present once started.

Important! Once you have build your zip, follow the above steps to be sure you fulfill them. This will reduce evaluation problems.

Last modified: Tuesday, 27 September 2016, 8:19 PM

Return to: Practice