Assignment 2: Communication, Cooperation and Negotiation

CSE5022 Advanced Multi-Agent Systems

DDL: 23:59, Wednesday, May 15, 2024

1 Overview

In this assignment, you must implement Multi-Agent System concepts related to communication, cooperation and negotiation among agents within a simulated environment. For this purpose, you will extend the code based on the laboratory sessions for the Robot Agents exploring and collecting materials on a remote planet. The overall objective of the agents is to collect ”Precious Materials” to trade them in exchange for survival resources. Furthermore, they can create Alliances with other agents to ensure cooperative operations and survival. For this purpose, you are required to use the open-source Repast Symphony Simulation Multi-agent Toolkit[1], implementing the appropriate classes in *Java* language (considering Object-oriented programming principles).

2 Objectives

- **Communication:** Implement a direct communication mechanism between agents, where agents can communicate to create alliances, request cooperation or negotiate resources.
- **Cooperation:** Implement a simple cooperation mechanism between agents in the same Alliance to communicate the location of materials and increase overall Alliance utility.
- **Negotiation:** Agents can negotiate resources to survive or maximise the agent's utility.
- **Performance Evaluation:** The agents' performance will be based on the Alliance utility where they belong. Generate simulation results in a .csv file to identify the performance in different time steps.

[1] https://repast.github.io/

3 Requirements

1. Classes suggested for this assignment and essential attributes required for the simulation are listed below.

   - *Explorer*: The main agent in the simulation. The goal is to find rocks of the Precious Material. Initially, all agents act as *self-interested* entities. However, during the simulation, they can form alliances with other Agents to maximise alliance utility.
   - *Trader*: Special agents responsible to negotiate with the *Explorers* by giving some survival resources in exchange of *Precious Materials*. Survival resources will restore the survival level of the agent. They appear after the first 100 time steps and can move randomly to be detected by the explorers.
   - *Message*: Encodes the information transmitted between agents.
   - *Precious Material*: All agents aim to collect this material to obtain some utility (when they are traded). A group of them makes a mine.
   - *Base Station*: The origin where all Explorers start operations in the simulation.
   - *Destination*: The ”storage area” where the agents can download the collected rocks. Obtain additional survival resources and increase it’s *utility*. Once the materials are downloaded, the agent can return to the mining spot (origin point).

2. **Initialisation**

   - The *Explorers* are initialized with a 100 % *survival level*, an initial utility of 0 units and some *extra survival resources* **only for negotiation purposes**. The Explorer’s capacity is specified with a *maximum capacity* (randomly initialised).
   - *Precious Material*: A random *weight* is assigned. This Explorer can take only up to its maximum capacity. Moreover, the number of rocks (to make a mine) is randomly specified, together with the weight of each rock. This is important since Explorers can communicate the location to cooperative Explorers.
   - The *Traders* number should be initially specified (for example, 3 Traders). The fewer traders, the more competitive the environment you create. The initial *survival resources* amount can be randomly specified.

3. **Simulation**

   - The Explorers start from the Base Station, looking for Precious Materials that will be used later for negotiation.
   - At every time step, the survival level decreases by 0.1 units.

  - If one Explorer finds a ”Precious Material” spot (possibly a mine). It can collect as much material as its maximum capacity and store it in this location. If the materials exceed the agent capacity, it can contact another agent in a ratio of 10 units to *Request* for cooperation. The agents contacted *directly* can *Response* to the message and decide to make an *Alliance*. An **Alliance** is a group of explorers cooperating within the environment. An agent can belong to only 1 Alliance.

  - The decision to form an Alliance will imply that the Explorer will always have a list of cooperators who can benefit by communicating the Precious Material locations. Once an Alliance is made, the agents make a group and combine the individual utility. The Alliance utility is the average of all agents' utility in the same group.

  - **Utility:** The explorers can increase their utility by trading precious rocks with traders and other agents or downloading materials in the Destination spot. You must decide how much utility they will receive for performing each operation. Please explain your decision in the report.

  - In addition, each Explorer seeks to survive in the simulation; therefore, they need to *Negotiate* survival resources with Precious Materials or be part of an Alliance to know where they can potentially find Precious Materials to trade. You have to decide how much (in terms of Precious Rocks) the survival resources will be. Include this consideration in the report.

  - The environment where the agents operate should be dynamic, which changes every 300 timesteps. The change involves creating new PreciousMaterials objects in the Context Builder (environment). **You don't need to implement this method since it will be provided as part of the Precious Material class. However, you have to change the indicator for the environment to create new spots of Precious Materials**. This is done in the updateMinesSpots() function.

  - The *Traders* presence should be dynamic as well; they should appear after the first 100 time steps and then disappear after 100-time steps (or if they don't have more survival resources to trade with agents) and repeat this cycle during the entire simulation. You can decide if the agents will move randomly or be static in the environment.

  Finally, the simulation must report **Results** based on alliance performance. If explorers at the end of the simulation are not in any Alliance, they must also be reported.

- **Results:** Enable Data collection process in the simulation interface [2].

[2] https://repast.github.io/docs/RepastJavaGettingStarted.pdf

4 Hints

- All classes and essential code needed to execute this simulation are provided. This assignment is an extension of the lab sessions. Consider having a List of the cooperative Explorers' IDs for direct agent communication as an Explorer attribute.
- For the **Alliances**, consider using a boolean variable (inAlliance) to control the communication.
- The decision for each agent to join an **Alliance** is up to you. It can be, for example, considering how far the Requesting Explorer is from a Precious Material spot (similar to the lab practice).
- Consider that the `Context` class allows access to all Objects within the simulation, and `@ScheduledMethod` specify when to trigger any custom method.

5 What to Submit

1. **A report in PDF format** describing each aspect of the simulation, including Communication methods, cooperation and negotiation mechanism and decisions you made. Please write it with as much detail as possible using screenshots of the simulation runtime GUI and code, illustrating the solution step by step.
2. **All source code files**, since it will be evaluated by executing the simulation during different time intervals.

Pack all files into `SID_NAME_A1.zip`, where `SID` is your student ID and `NAME` is your name (e.g., 11710106_ 张三 _A1.zip).