UG3/4 Agent-Based Systems First Assessed Practical

InfoSpeak Environment
Part 1 - Environment exploration and communication

Use the `submit` *program to submit your solution by executing the command*

```
submit abs 1 professor.asl attentive.asl reactive.asl
```

on any DICE machine. The deadline for submission is

Thursday, 11th February 2016, at 4pm

Late submission policy: Normally, you will not be allowed to submit coursework late. If you have a good reason to need to submit late, you should submit an ITO support form as soon as possible at http://www.inf.ed.ac.uk/teaching/contact/index.html
Only in exceptional circumstances, e.g. illness that stopped you getting to email, would an extension be granted after a deadline has passed. You can find a description of what constitutes "good reason" to submit late at

http://web.inf.ed.ac.uk/infweb/student-services/ito/admin/coursework-projects/late-coursework-extension-requests

Introduction

In this practical you are asked to design code for various agents and have them interact with the InfoSpeak environment. The environment represents a simulation of the campus, with various buildings being integrated. Figure 1 below shows an example of the grid world the agents must operate in. The agents are the (in this case blue) ovals, the park is drawn with green in the middle and buildings are represented as labelled black boxes. The buildings are described in Table 1.
Agents already know where every building is positioned and how to get there. They have lectures and other events that they have to attend, which are all described in the configuration file for the InfoSpeak environment, *infospeak.config*.

Before you attempt the assignment, please read *Intro to Jason.pdf* and *SlidesJason.pdf* for Jason's basic syntax and logic, and *Intro to Environment.pdf* for a more thorough explanation to the InfoSpeak environment and list of functions specific to it. The book *Programming Multi-Agent Systems in AgentSpeak using Jason* (Google books link) is available in the library and online if you wish to delve deeper into the language.

Grid name	Building	Environment name
OUT	Homes	out1, out2, ...
LIB	Library	lib1
LT	Lecture Theatres	lt1, lt2, lt3
FOR	Informatics Forum	frum1
LAB	Laboratory	lab1
BAR	Pub/Bar	bar1
GYM	Gym	gym1

Table 1: Buildings table

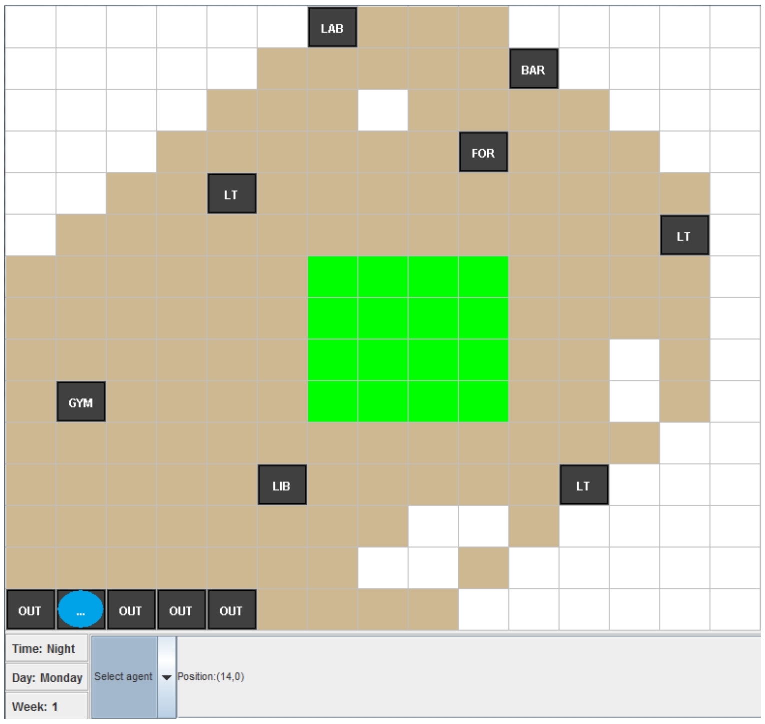

Figure 1: The Gridworld

Intro to Jason.pdf explains how to set up the jEdit and Jason jEdit plugin, which is required to run the program. However, if you find jEdit heavy and uncomfortable to edit with, a package for the *atom* text editor with syntax highlighting for AgentSpeak is available (https://atom.io/packages/language-agentspeak). Note that you still need jEdit to run the files.

Finally, for any questions please email the Teaching Assistant (TA) for the module, Michalis Michaelides, at mic.michaelides@ed.ac.uk. If you are uncertain about how to start or how to proceed at any point, if something is broken or instructions are unclear, do not hesitate to email the TA for clarification or assistance.

Questions

Your task is to modify existing agent code, or implement new agent types, so that they exhibit the behaviour described against each item below in parts a), b), and c). The marks that can be allocated for each question and question item are listed next to them.
It is important to comment your code fully. Comments should clearly and concisely annotate all plans, or groups of plans, and identify which rules are part of answers to which part of the question. If the purpose of function of a plan is unclear, then you may not gain credit for it!

a) Professor agent - 20 marks

In the first section you will familiarize yourself with the already existing agent, the professor agent. You should extend the agent's implementation by the following behaviours:

1. If the professor agent is going to a class, use the .print(Place) function to print the name of the lecture theatre in which the class is held (2 marks)
2. The professor agent will go home to sleep by default at 7pm. (5 marks)
3. The professor gets sick with 20% probability per day. Add the sick belief to its belief base. You can use the .random(X) function which binds X to a random number, $X \in [0, 1)$, to implement the probability. (5 marks)
4. If the professor agent is sick, it will stay at home for 2 days to recover. Upon recovery the sick belief should be removed. (5 marks)
5. The professor agent has to print a message announcing that it got sick or that it has recovered whenever these happen. Use the .print("message") function. (3 marks)

b) Attentive agent - 32 marks

In this section you will implement an attentive agent. The agent will do the following:

1. It has to attend all lectures. (5 marks)
2. It must revise whenever there is spare time at the library between 9am and 5pm. (5 marks)
3. It will go to the gym from 5 to 6pm every day, provided no lectures will be missed, and go home when done to sleep. (4 marks)
4. It gets sick with 30% probability per day, and recovers after a day. It will not go to the gym if it is sick (otherwise its schedule is unchanged). (5 marks)

However, if an assignment is due and not finished the agent will exhibit the following behaviour:

5. It must attend all lectures, unless it is sick (in which case look at next point). (3 marks)
6. It must work on assignments whenever there is spare time at the lab between 10am and 9pm, or 7pm if sick. (5 marks)
7. It must go home at 9pm every day, or at 7pm if sick. (5 marks)

c) Reactive agent - 25 marks

In this part, you should implement an agent who only has a basic schedule and reacts to messages it receives from the attentive agent. The reactive agent behaves according to the following rules:

1. It must go the the library at 9am. (5 marks)
2. It must go home at 8pm. (5 marks)

3. It must react to messages it receives from the other agent by performing the appropriate task (e.g. if it is told a lecture is happening, it must attend the lecture; if it is invited for lunch, it must go). It doesn't matter how this implemented in the agent's internal logic as long as it attends the events in the messages. The messages expected to be received are described in part d), points 1. 2. and 3. Whenever the reactive agent is done with any activity, it needs to return to the library, unless it is 8pm, when it will return home. (15 marks)

d) Messaging - 23 marks

The remaining questions should be answered by modifying the design of the attentive agent in b) to behave as follows, unless the attentive agent in b) is sick, in which case it will NOT perform these tasks. Only this final version of the attentive agent, which includes messaging, should be submitted.

1. It must inform the reactive agent when a lecture is happening. (9 marks)
2. When no lectures are scheduled, it must have lunch at the bar at 2pm. (6 marks)
3. It must invite the reactive agent at the bar to have lunch together. (8 marks)

Messages must be sent so that the reactive agent will arrive at the required locations in time. Use the .send(receiving-agent, tell, goal(X)) function in sender-agent to ask receiving-agent to consider performing plan goal(X).

The behaviours of all agents should be tested on their own, and, where required, when interacting with each other. If you are unable to implement some of the required functionality, it is preferable to submit code that can be compiled and runs normally in the environment, and you may comment in the code or through printed messages on functionality that is missing. You should expect that a large proportion of marks will be lost for solutions that either do not compile or produce serious run-time errors.

Plagiarism

Good Scholarly Practice: Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

http://www.ed.ac.uk/academic-services/students/undergraduate/discipline/academic-misconduct

and at:

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (generally permitting access only to yourself, or your group in the case of group practicals).