Barış Palaska October 21, 2015 2010401045

**CMPE 540: Principles Of Artificial Intelligence**

**Assignment I: Vacuum Cleaner**

On this assignment, the task was to build an intelligent vacuum cleaner. In order to implement this, I created a simple environment that consists of 4 adjacent zones. These zones can be either clean or dirty. Our agent, a vacuum cleaner that has location and dirtiness sensors travels in these zones and cleans them if it gets a “dirty” signal.

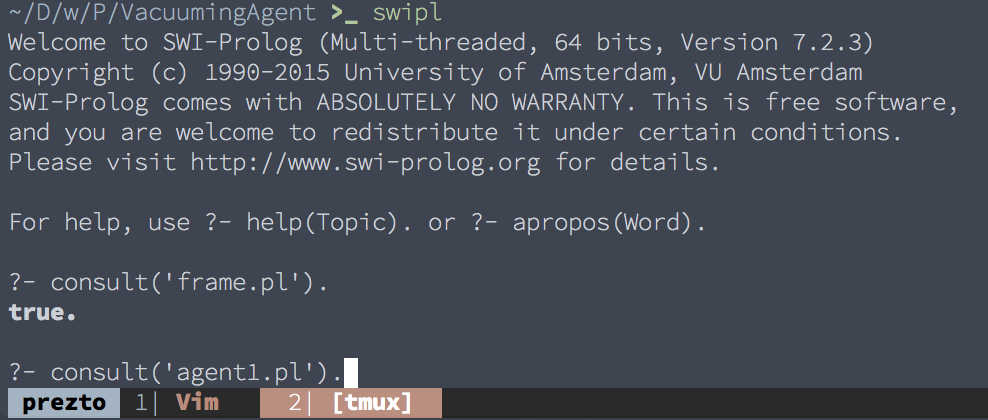
In the project directory, there are four *.pl* files (frame.pl, agent1.pl, agent2.pl, initialstate.pl). agent1.pl and agent2.pl are the agent codes which has different specs from each other. frame.pl defines the dynamic predicates needed and defines the iterations. initialstate.pl defines the dirtiness of the environment and sets the agent’s initial position.

Agent1 is a simple reflex agent which has no memory and has no idea about the history. It can take three different actions; go left, go right and clean. Its movement is random; a random function runs and picks a number between 0 and 1. If the number is in the lower half portion, it moves right and if it’s in the upper portion, it moves left. If it senses dirt in the room, it cleans it. This agent never stops and always searches for a dirty zone moving randomly.

Agent2 is more intelligent compared to 1. It can take an additional action which is ‘do nothing’. In order to implement ‘do nothing’ correctly, it should have memory. The agent must realize that all the zones in the environment are clean. A ‘knowClean(...)’ predicate is asserted when a zone is cleaned or a clean zone is discovered for the first time. If all the zones have knowClean predicate, that means the environment is entirely clean so it can stay idle and take the action “do nothing” until the end of the iterations.

You can find the comments inside the codes which explains which part does what job. Below, the test cases and different agent behaviours are shown in the pictures. In order to run the program, you should follow this prosedure:

* Run SWI Prolog application / run “swipl” in terminal
* Run the following commands on the console:
  + consult(‘frame.pl’).
  + consult(‘agent1.pl’). (or consult(‘agent2.pl’).)
  + simulateMore.

Figure 1: Commands to run the program

**Test cases**

* **Test 1**

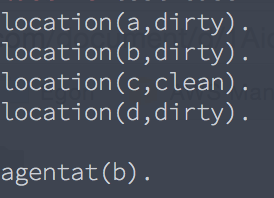
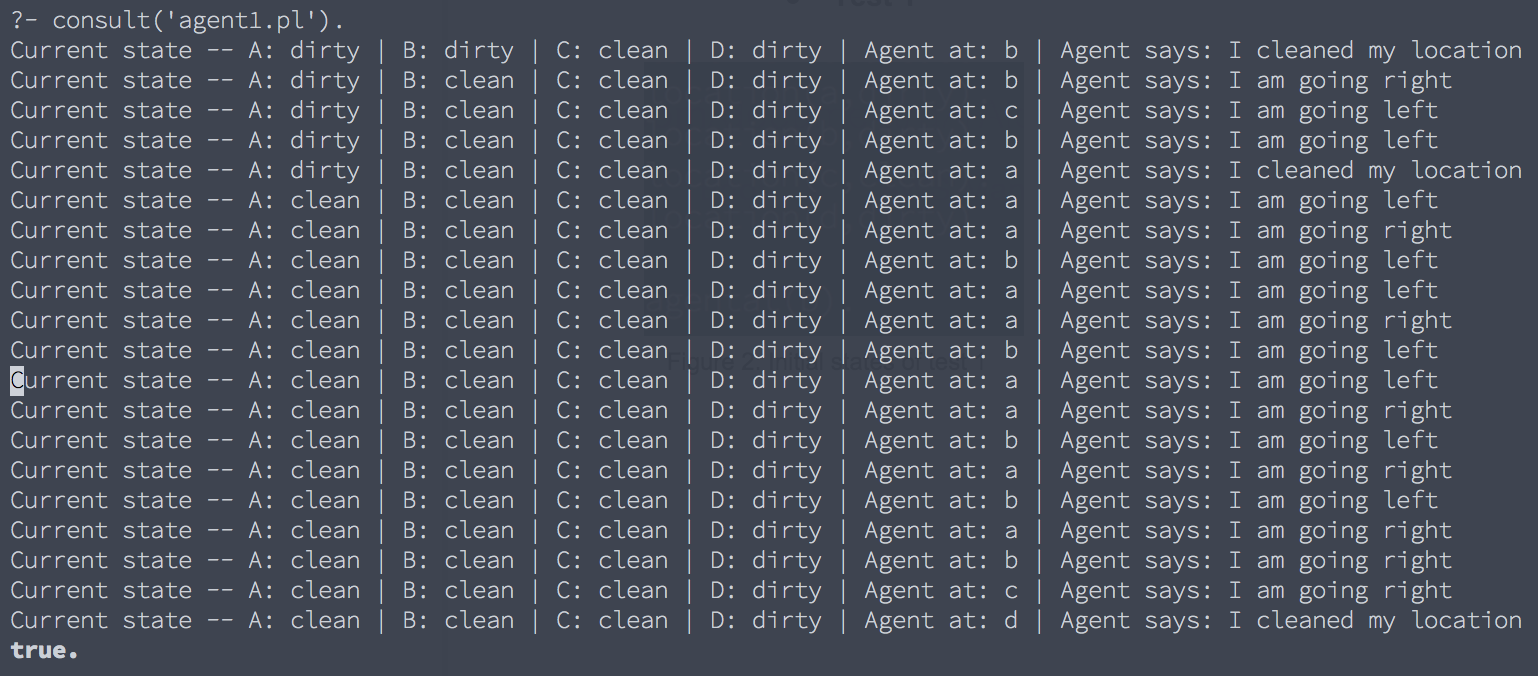
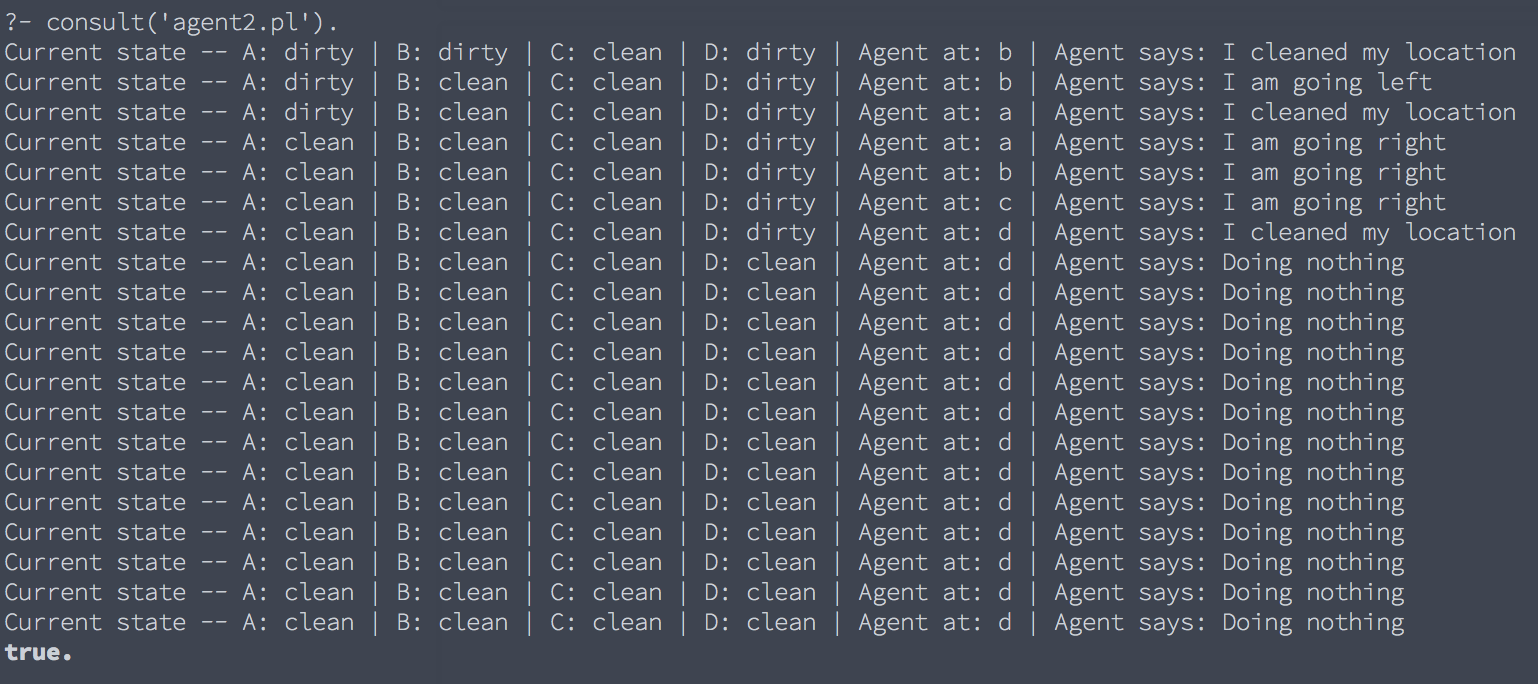


Figure 2: Initial state of test1

Figure 3: Behaviour of agent1 at test1

Figure 4: Behaviour of agent2 at test1

* **Test 2**

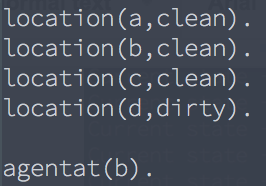
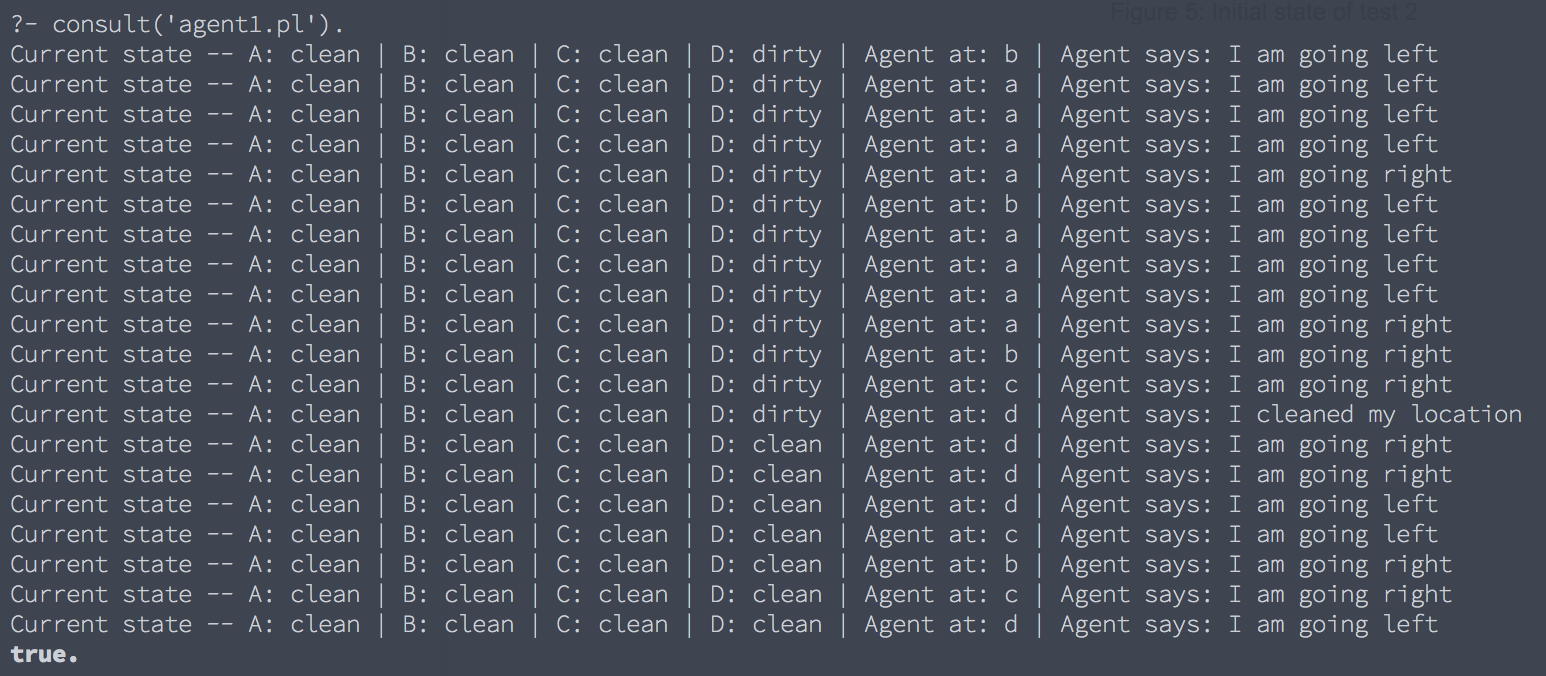
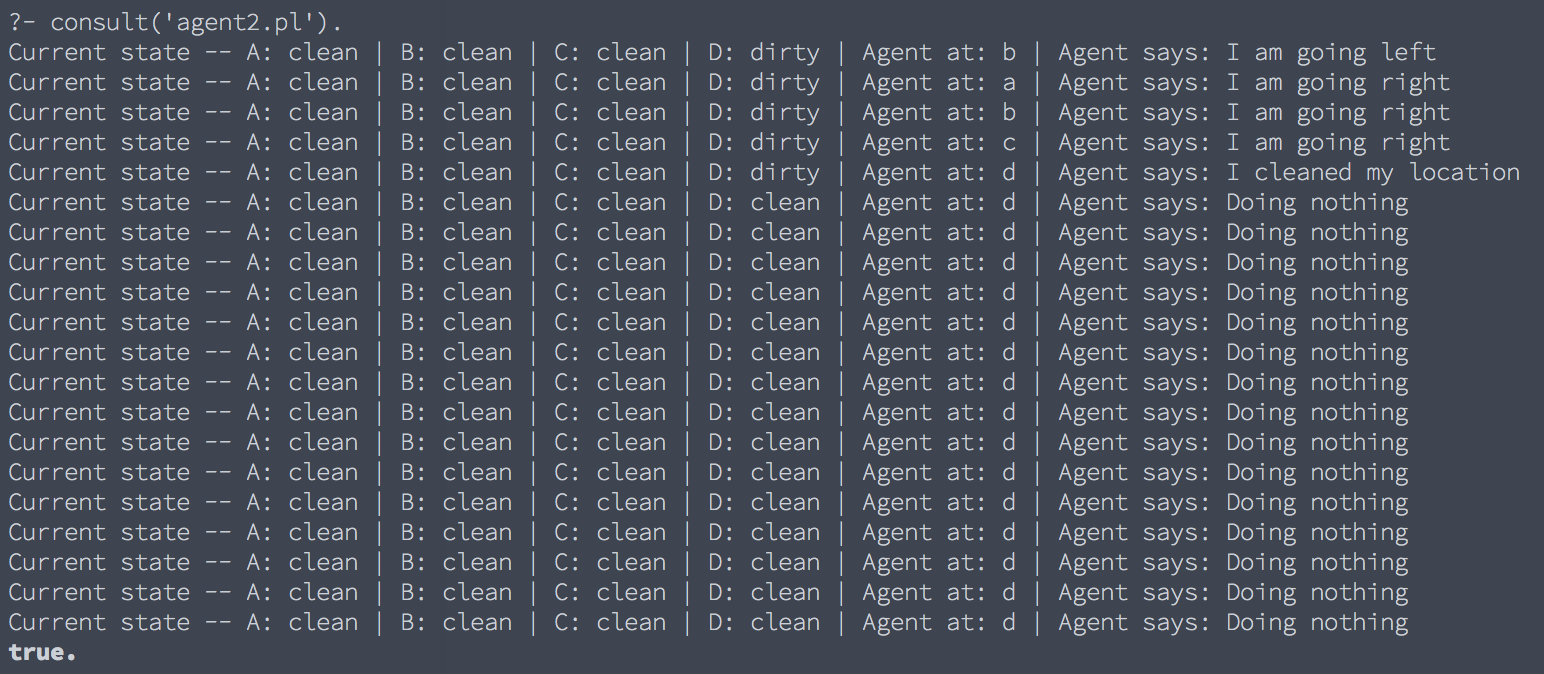


Figure 5: Initial state of test 2

Figure 6: Behaviour of agent1 at test2

Figure 7: Behaviour of agent2 at test2