Artificial Intelligence
2I1AE1
Regis.Clouard@ensicaen.fr
Eric.Ziad-Forest@ensicaen.fr
Loic.simon@ensicaen.fr
2016-2017

Assignment. Rational Agent: Wumpus World

Due Date : 08:00 pm Friday, December 30

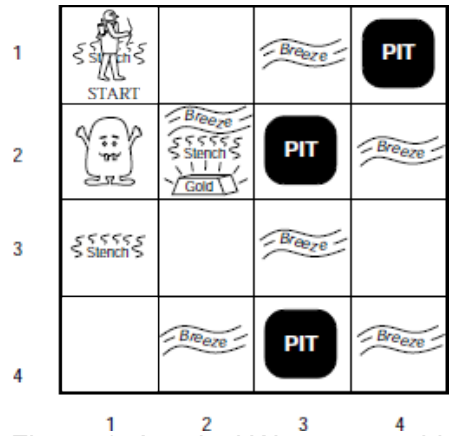

Figure 1: A typical Wumpus world.
The agent is in the top left corner facing right.

In this lab you will familiarize yourself with the concept of rational agent. The goal is to program an agent that can survive in a hostile world.

1. Instructions

What to submit: You have to submit two files via the Moodle platform **by the specified date**.

1. You will fill in portions of agent.py during the lab.
2. You must hand in a report with the response to questions in Section 5.

Evaluation: Your code will be assessed in two ways.

1. The **correctness and clarity** of your python implementation. No comment is needed if your implementation is the mirror of the algorithms. Use additional comments only when you deem necessary to explain special implementation.
2. The **behavior** of your agent against the command given in the provided file `commands.txt`. *Please do not change the names of any provided functions or classes within the code*. You only have to fill in the module agents.py. However, you can add whatever methods or data needed for your implementation.

Academic Dishonesty: We will be checking your code against other submissions in the class for logical redundancy. If you copy someone else's code and submit it with minor changes, we will know. We trust you; *please* don't let us down. If you do, we will pursue the strongest consequences available to us.

2. The Wumpus World

A Wumpus world as shown in Figure 1 is a cave consisting of rooms connected by passageways. This world is surrounded by walls, and walls can only appear in the boundary. Lurking somewhere in the cave is the Wumpus, a beast that eats anyone who enters its room. The Wumpus can be shot by an agent. The agent can fire an arrow in a straight line in the direction the agent is facing. The arrow continues until it either hits (and hence kills) the Wumpus or hits a wall. The agent only has one arrow, so only the first shoot action has any effect. Some rooms contain bottomless pits (*puits*) that will trap anyone who wanders into these rooms. The only mitigating feature of living in this environment is the possibility of finding a heap of gold.

The agent always starts in the square labeled [1, 1], facing to the right (ie. to case [2, 1]). The locations of the gold and the Wumpus are chosen randomly, with a uniform distribution, from the squares other than the start square. In addition, each square other than the start can be a pit, with probability 0.1.

The goal of the game is to achieve the highest score. The score is computed as follows:

Each move costs one point

- Shooting the arrow (irrespective of whether it killed the Wumpus) costs 10 points
- Grabbing the gold (i.e., having previously successfully '`Grab`'bed the Gold) earns 1000 points.
- Killing the Wumpus earns 500 points.
- Dying, by entering a square with the Wumpus or a pit, costs 1000 points.
- Finally, leaving the cave (accomplished by executing '`Climb`' in the same location that the hunter agent started in the cave 'entrance') earns 500 points.

This world is plunged into the dark, so the agent can only perceive its environment through percepts. The neighborhood of a node consists of the four squares north, south, east, west of the given square. In a square the agent gets a vector of percepts, with components:

{ Stench, Breeze, Glitter, Bump, Scream }.

- **Stench** (*puanteur*) is perceived at a square iff the Wumpus is at this square or

in its neighborhood.

- **Breeze** (*brise*) is perceived at a square iff a pit is in the neighborhood of this square.
- **Glitter** (*lueur vive*) is perceived at a square iff gold is in this square.
- **Bump** (*coup*) is perceived at a square iff the agent goes forward into a wall.
- **Scream** (*cri*) is perceived at a square iff the Wumpus is killed anywhere in the cave.

An agent can do the following actions (one at a time):
{ Right, Left, Forward, Shoot, Grab, Climb }.

- The agent can go **Forward** in the direction it is currently facing, or Turn **Right**, or Turn **Left**. Going Forward into a wall will generate a **Bump** percept.
- The agent has a single arrow that it can **Shoot**. The arrow will go straight in the direction faced by the agent until it hits (and kills) the Wumpus, or hits a wall.
- The agent can **Grab** the gold at the current square. The agent can **Climb** out of the cave iff it is at the start square of coordinates [1,1].

3. Resources

Clone the project lab5 into your file system using:

```
git clone http://www.ecole.ensicaen.fr/~rclouard/AI/lab5.git
```

This archive contains all the code and supporting files. There you will find four Python modules (wumpus.py, wumpusworld.py, utils.py, agent.py) and a number of image files (GIF) that are used to generate a visualization of your program. This program also needs the TkInter GUI toolkit.

First, try running the module wumpus.py:

```
./wumpus.py -w 4 -g 0
```

Option '-w *x*' is used to specify the size of the world, and '-g *x*' is used to produce the same world each time *x* as the same value.

Note that `./wumpus.py -h` displays the list of possible arguments.

By default, the agent acts randomly.

4. Exercise. Goal-Based Agent

Your goal is to program a clever agent that navigates in an unknown square environment. We only suppose that the environment size is known through the parameter `gridSize` of the method `init()`.

```
def init( self, gridSize ):
```

4.1 Percept

The method `think( self, percept )` has one parameter '`percept`' that represents the percept values to the agent. The class `Percept` provides the following boolean attributes:

- `percept.stench`
- `percept.breeze`
- `percept.glitter`
- `percept.bump`
- `percept.scream`

4.2 Actuators

The method `think()` should return a string representing the action the agent should perform. The six following actions are available:

- `'left'`
- `'right'`
- `'forward'`
- `'grab'`
- `'shoot'`
- `'climb'`

You can use the priority queue given in the module utils.py (same as lab 1).

Test your code with the command:

```
./wumpus.py -a GoalBasedAgent -w 10 -g 0
```

5. Questions

The question should be answered in a separate report and submitted on the Moodle platform with the module agent.py.

❑ **Question 1.** Write a complete and formal description of the task environment (PEAS).

❑ **Question 2.** Describe your implementation of goal-based agent in **reference to classical IA methods**.