Introduction to AI: Assignment 1

Irek Nazmiev

Task 1

Performance measure

- Legal trip - make sure in user's affiliation to the appropriate flight(s) in order to get rid of the cheaters;
- Fast verification procedure - save the user's time and prevent long queues in front of the terminals in order to speed up the registration process for all airport visitors;
- Accurate documents recognition, baggage checking and passes painting - minimize the amount of mistakes in data processing and output in order to decrease congestions and save the time;
- Minimum human interaction (with the System and staff) - require as less as possible number of actions from user to perform in order to make the process quick and easy, and lessen the airport staff's busyness;
- The most comfortable user experience - make the check-in counter's interface and the registration as easy as possible in order to make the whole process comfortable and explicit.

Environment

- Airport:
  - Partially observable - agent's sensors allow it to see only a small piece of the environment: documents by scanner, user's face by camera and baggage by X-ray scanner; the airport can't be fully seen by the agent;
  - Multiple agents - there must be a lot of check-in counters in the environment by the reason of permanently high flow load of people at the airport;
  - Stochastic - we can't predict all the possible situations that may happen in the airport, and despite the fact that next state of the environment is specified by the agent, everything may happen because of the nature of human;
  - Sequential - each stage of registration process is completely dependent on the previous ones, and all the user's decisions affect on the final result;
  - Dynamic - airport is a really chaotic place, its state is always changing, especially while the user is registering by the agent;
  - Continuous - the registration process is always going on, it continues over the limitless time and has an infinite number of steps;

- Known - the designer of a check-in counter must know everything about the airport organization.

Actuators

- Display - to show the information about each step of registration process;
- Speaker - to pronounce the information about each step of registration process specially for people with disabilities;
- Printer - to print all required tags and passes;
- Baggage conveyor - to receive the baggage and move it through the X-ray scanner.

Sensors

- Touch screen - to get the actions to perform and read the information to process;
- Camera - to verify the user's face with documents and use it for safety preservation;
- Scanner - to scan the documents for their further processing;
- X-ray scanner - to scan the baggage on danger or forbidden stuff.

Task 2

Scorchbeast

Performance measure

- Expand the population - to be fruitful, to reproduce themselves, to look for the places for new colonies, take care of the brood;
- Satisfy hunger - after the colony is overcrowded, they dig to the surface to find the food for the brood and for themselves;
- Survive - typical purpose of all primitive creatures;

Environment

- Appalachia (Fallout 76 world, 2102 year):
  - Partially observable - scorchbeast's sensors allow it to see only a small piece of the environment because of the following reasons:
    - Most of the time it lives underground and dig to the surface only for looking for the new places to live in;
    - The area of its habitat is strictly limited. No agent is able to cover the whole world or at least the Appalachia location.
    - It was evolved from an unknown species of bat, that's why it doesn't have really good vision and eyes;
  - Multiple agents - scorchbeasts live in huge colonies underground, leaving them only for making new ones. In the environment it may probably happen that several scorchbeasts spawn together;
  - Stochastic - we can't predict all the possible situations that may happen in the fallout world, this is a place containing a lot of creatures and random events;
  - Episodic - scorchbeast has a fixed set of actions that are done randomly and independently without any consequences;

    - Dynamic - fallout world is a really chaotic place, its state is always changing while the agent is deliberating;
    - Continuous - fallout 76 has a breathing world, it has infinitely many variations of actions, and state of the environment will be changing continuously.

Actuators

- Wings - fly on the air, see the ground from high places looking for the players and places for new colonies;
- Jaws - chew the food, attack players;
- Throat - create a powerful sonic blast, douse acidic radioactive mist for attacking the players;
- Paws - attack players, move on the ground, keep the balance while flying.

Sensors

- Eyes - look for the food and players to attack, new places for colonies;
- Ears - react on sounds made by players' actions;
- Skin - feel the pain, react on touch.

Task 3

Check the file *wumpus.pl.*

The file is completely documented and commented.

Query structure:

- Start = [X, Y] - starting coordinate of an agent,
- Gold = [X, Y] - coordinate of a gold,
- Wumpus = [X, Y] - coordinate of a Wumpus
- Pits = [[X, Y], ...] - coordinates of pits
- Size = [W, L] - size of a field,
- Route = [[X, Y], ...] - route to the gold
- Alive = (0 or 1) - Wumpus is Alive
- Arrow = (0 or 1) - Agent has arrow

?- start(Start, Gold, Wumpus, Pits, Size, Route, Alive, Arrow).

Initially, agent has to have an arrow (Arrow = 1) and Wumpus has to be alive (Alive = 1).

Key features:

The agent looks for the gold by brute-forcing all the possible paths and backtracking in case of death.

Agent is able to kill the Wumpus - attempt is made only on a cell with stench by the optimizational reasons. If agent is in at the cell with stench, there's a higher probability to hit the Wumpus - it will be killed by shooting the arrow on one of the 4 directions (directly adjacent) for sure. That's why agent recursively shoots the arrows in all 4 directions, and as a result Wumpus is killed.

The cells with stench are found automatically by the Wumpus's position.

In my implementation the arrows are shooted in all 4 directions (not just horizontally and vertically) in order to make it possible to track the agent's action precisely.

There's a rule called *start* needed just for running the program and for convenience in describing the variables.

It's possible for agent to miss the shot. There is a distinct rule for this situation.

There is a rule called *breeze,* but it's not used at all. In my implementation there's no need for such functionality.

Task 4

Test 1: decreasing the area to 3x3 cells.

P	P	P	P	
	W	G	P	
			P	
			P	

Query:

?- start([1,1], [3,3], [2,3], [[1,4], [2,4], [3,4], [4,4], [4,3], [4,2], [4,1]], [5,5], [], 1, 1).

Result:

[[1, 1], [2, 1], [3, 1], [3, 2], [3, 3]]

Wumpus is killed from [2, 2]
[[1, 1], [2, 1], [3, 1], [3, 2], [2, 2], [2, 3], [3, 3]]

[[1, 1], [2, 1], [3, 1], [3, 2], [2, 2], [1, 2], [1, 3], [2, 3], [3, 3]]

Wumpus is killed from [2, 2]
[[1, 1], [2, 1], [2, 2], [3, 2], [3, 3]]

[[1, 1], [2, 1], [2, 2], [2, 3], [3, 3]]

[[1, 1], [2, 1], [2, 2], [1, 2], [1, 3], [2, 3], [3, 3]]

Wumpus is killed from [2, 2]
[[1, 1], [1, 2], [2, 2], [3, 2], [3, 3]]

[[1, 1], [1, 2], [2, 2], [2, 3], [3, 3]]

[[1, 1], [1, 2], [2, 2], [2, 1], [3, 1], [3, 2], [3, 3]]

Wumpus is killed from [1, 3]
[[1, 1], [1, 2], [1, 3], [2, 3], [3, 3]]

Test 2: make it impossible to achieve the goal.

				G
P	P	P	P	
	W		P	
			P	
			P	

Query:

?- start([1,1], [5,5], [2,3], [[1,4], [2,4], [3,4], [4,4], [4,3], [4,2], [4,1]], [5,5], [], 1, 1).

Result:

Wumpus is killed from [3, 3]

Wumpus is killed from [2, 2]

Wumpus is killed from [2, 2]

Wumpus is killed from [2, 2]

Wumpus is killed from [1, 3]

Comments:

Agent didn’t get the gold, but it has killed the Wumpus on 5 paths passed during the gold searching.

Test 3: make the gold achievable only by killing the Wumpus.

		G		
P	P	W	P	
			P	
			P	
			P	

Query:

?- start([1,1], [3,5], [3,4], [[1,4], [2,4], [4,4], [4,3], [4,2], [4,1]], [5,5], [], 1, 1).

Result:

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [3, 1], [3, 2], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [3, 1], [3, 2], [2, 2], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [3, 1], [3, 2], [2, 2], [1, 2], [1, 3], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [2, 2], [3, 2], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [2, 2], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [2, 1], [2, 2], [1, 2], [1, 3], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [2, 2], [3, 2], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [2, 2], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [2, 2], [2, 1], [3, 1], [3, 2], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [1, 3], [2, 3], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [1, 3], [2, 3], [2, 2], [3, 2], [3, 3], [3, 4], [3, 5]]

Wumpus is killed from [3, 3]
[[1, 1], [1, 2], [1, 3], [2, 3], [2, 2], [2, 1], [3, 1], [3, 2], [3, 3], [3, 4], [3, 5]]

Comments:

As it could be seen, all the paths are passed by killing the Wumpus.

Test 4: keep only one possible way to achieve the goal.

			P	W
			P	G
			P	
P	P	P	P	

Query:

?- start([1,1], [5,4], [5,5], [[1,2], [2,2], [3,2], [4,2], [4,3], [4,4], [4,5]], [5,5], [], 1, 1).

Result:

[[1, 1], [2, 1], [3, 1], [4, 1], [5, 1], [5, 2], [5, 3], [5, 4]]

Comments:

No Wumpus kill is needed. There’s just one route to the gold.