Ismail Ben seddik

Project 2

This project is a simple knowledge-based agent constrained with different model configurations with the purpose of killing the Wumpus. The agent tries not to fall into pits, not to get out of the model bounds, not to get eaten by the Wumpus and potentially tries to grab the gold for score augmentation.

Running the Project

- ✓ Upon consulting the knowledge base (WumpusWorld.pl), please type: `run.` to load it and run it.
- ✓ World configurations can be modified in the knowledge base. It already encompasses three models (with 2 commented out). To experiment using any model, please comment out the other models. Creating new configurations can simply be done through altering model predicate values under the "init" predicate.

Primary Predicates

- **position(R, [X,Y]):** indicates different configurations and positions of model components (pits, Wumpus and gold). Position(R, [X,Y]) simply means that room R is located in [X,Y].
- **adjacent([X,Y], Z):** room [X,Y] is adjacent to room Z.
- **breezy([X,Y]):** room [X,Y] is breezy ⇔ a pit is in an adjacent room.
- **smelly([X,Y]):** room [X,Y] is smelly ⇔ the Wumpus is in an adjacent room.
- **glittery([X,Y]):** room [X,Y] is glittery ⇔ gold is in an adjacent room.
- **killWumpus(AgentCell):** kill the Wumpus (if facing it) and win the game.
- **grabGold():** grab the gold and earn a 1000 points bonus.
- **search(Cell, LeadingPath):** main predicate representing agent actions towards attaining its objective and, actually, appending navigated cells to the path leading towards the goal. Through it, the game score is maintained and all essential moves are realized.

Model Experiments

The following are 4 different configurations: the first 2 being 4x4 models and the other 2: 3x3 models.

Model 1

```
+ Wumpus in cell [1,3] was shot with an arrow from cell [1,2] and killed!
+ Path to killing the wumpus is [[1,1],[2,1],[2,2],[1,2]]
+ Game score is -4
true
```

The Wumpus, in this model, is at [1, 3]. It includes three pits: one at [3,1], the other at [3,3] and the last one at [4,4]. The gold is at [2,3].

Model 2

```
+ Wumpus in cell [2,4] was shot with an arrow from cell [1,4] and killed!
+ Path to killing the wumpus is [[1,1],[2,1],[2,2],[1,2],[1,3],[1,4]]
+ Game score is 1989
true
```

The Wumpus, in this model, is at [2, 4]. It includes 3 pits: one at [3,1], the other at [3,3], the other at [4,4]. The gold is at [1,2].

Model 3

```
+ Wumpus in cell [2,2] was shot with an arrow from cell [1,2] and killed!
+ Path to killing the wumpus is [[1,1],[1,2]]
+ Game score is -3
false.
```

In this model, the Wumpus is at [2,2]. The pits are at [1,2], [2,3] and [3,3]. The gold is at [3,1].

Model 4

```
+ Wumpus in cell [2,1] was shot with an arrow from cell [1,1] and killed!
+ Path to killing the wumpus is [[1,1]]
+ Game score is -1
false.
```

In this 3x3 model, the Wumpus is at [2,1]. The pits are at [1,2], [2,3] and [3,3]. The gold is at [3,1]. In this case, the Wumpus is already facing the agent. This model was merely to diversify testing configurations.

Assessment, Limitations and Future Refinements

Fortunately, the agent can often kill the Wumpus and win the game all with optimal actions and in various configurations as demonstrated through the models above. Yet, since we are constrained with time, especially as a capstone student, the agent still suffers from some limitations as my teammate and myself could not optimize it for every configuration. The limitations and faults include:

- Agent, for some configurations, still considers previous model configurations which leads to faulty actions. For instance, if previous model's Wumpus was at [3,3], the agent would, sometimes, consider 2 Wumpuses (the old one and the one corresponding to the newest model). Computing the score also gets influenced by such fault which results in a wrong game score.
- Agent, sometimes, finds very non-optimal paths to killing the Wumpus.

Hence, with room left for upgrading the agent, future work would touch upon:

- Ensuring that previously defined/inferred facts are totally cleared out.
- Optimizing agent actions through maintaining the maximum possible score.
- Disabling backtracking and searching once the Wumpus has been killed (goal reached) so that even if previous facts weren't retracted, the solution would not yield nonsensical paths.
- Incorporating the probabilistic configuration of model components for a more sophisticated agent.
- Graphically visualizing agent actions through integrating the prologue knowledge base with other programming languages which support graphics.