# CS 511 AI 2 - Project 1

## Model-based Reflex Agent

**Unique Name: Purple**

Problem Statement: *A Model-based Reflex Agent for the Wumpus World Simulator. Chose to forgo nondeterminism;*
*i.e., for this second project assume deterministic forward action. Submit a file or zip directory named nickname.zip with your own unique nickname. Your submission will be tested in 50,000 randomly generated wumpus environments for the average obtained score.*

***Model Based Reflex Agents*** use a model of the world to choose their actions. They maintain an internal state. Model − knowledge about "how the things happen in the world". Internal State − It is a representation of unobserved aspects of current state depending on percept history.

The goal of this assignment is for the agent to maximize the total score while moving around the randomly generated world. As the Agent is a 'model-based reflex agent', it keeps track of the unseen world by maintaining an internal state. The input is the percept which represents the state. Then the rules based on that percept is run and the action is returned.

## *Implementation*

First, we set the (Boolean nonDeterministicMode) to 'false' as we're running our model-based agent on deterministic forward action. The number of trials, maximum number of steps and world size is also set by the WorldApplication.java class function.

### *Model Construction*

The model reflex agent consists of an internal model which is constructed based on the current state and percepts.
The model creates an internal map of the wumpus environment.

Let's say that we start from (0,0) and there is no stench or no breeze. This means that the model will mark the adjacent positions (0,1) and (1,0) as safe.
Thus, it can decide to move towards (1,0) or (0,1) as the next step.

At first, the agent moves to the location (1,0).
If it feels a stench, it means either (2,0) or (1,1) has a Wumpus. It is unsure where the wumpus might be located. So, it will mark these two locations unsafe.
Then, it moves back to (0,0) and chooses the other path to move to (0,1).
Here, it doesn't feel any stench which means that (1,1) is perfectly safe.

Now, the model can update that (1,1) is safe.
This means that (2,0) is the location with wumpus.
The agent can, thus, prevent visiting the location (2,0).

In this way, for each step, the previous action is considered and based on that the current/next action is decided.

### *Agent Movement/Function*

The function 'Agent' decides all parameters, except action taken during a turn, for the Agent during, including deterministic forward action (the forward motion succeeds 80% of the time, and agent slips right or left with probability 10% each). The probability is controlled by a random number generator deciding on one of the following moves:

```
switch (rand.nextInt(10)) {

  case 0: moveDir = 'F'; break;
  case 1: moveDir = 'F'; break;
```

```
  case 2: moveDir = 'F'; break;
  case 3: moveDir = 'F'; break;
  case 4: moveDir = 'F'; break;
  case 5: moveDir = 'F'; break;
  case 6: moveDir = 'F'; break;
  case 7: moveDir = 'F'; break;
  case 8: moveDir = 'L'; break;
  case 9: moveDir = 'R'; break;
}
```

The function that decides the action of the agent for each turn is 'AgentFunction'.
The action table consists of the following,

```
actionTable[0] = Action.NO_OP;
actionTable[1] = Action.GO_FORWARD;
actionTable[2] = Action.GO_FORWARD;
actionTable[3] = Action.GO_FORWARD;
actionTable[4] = Action.TURN_LEFT;
actionTable[5] = Action.TURN_RIGHT;
actionTable[6] = Action.GRAB;
actionTable[7] = Action.SHOOT;
```

***Process***

The initial condition is to check if the if the location has glitter. If 'true', then the agent grabs it.

Next, check if there is any bump, glitter, breeze, stench or scream. If not, then move 'forward'.

If there is a bump, then turn to the right or left. The probability for each is set to 0.5

If there is stench, then that means that the wumpus is near.
Hence, there are two possibilities for the next 'action'. The agent can either 'shoot' or take 'no action'. The model is considered for the previous, current and next action.

If there is scream and there is no breeze, then that means that there is no one on the way and the agent can move forward.

Else, if none of the above conditions are 'true', then there is 'no action' taken condition (in order to cover any default actions or loopholes in the code logic).

## ***Sample Outputs***

Given that the problem statement says to assume that:

- The environment is deterministic
- The location has been set to random
- Duration of time steps is 50
- Size of environment is 4*4

**Number of times the trials were run:** 5
**No. of trials:** 10000

Total Score: 3314105
Average Score: 331.4105

Total Score: 3564930
Average Score: 356.493

Total Score: 3854730
Average Score: 385.473

Total Score: 3366505
Average Score: 336.6505

Total Score: 3117790
Average Score: 311.779

***To Run:***
javac WorldApplication.java to compile
java WorldApplication -n false -t 10000

Or

If you are using eclipse,
Modify the run configurations with arguments as -n false -t 10000

Or
If you are using IntelliJ,
Modify the run configurations with arguments as -n false -t 10000 OR reset variables in WorldApplication.Java