Markov Decision Process Implementation and Policy Iteration

Devin Elmore, Daryl Nguyen, Thomson Nguyen, Kenneth Valero, Steven Yacoub

California State University: Long Beach

CECS 451: Artificial Intelligence

Dr. Mahshid Fardadi

May 1st, 2023

# Abstract

AI operating in any large scale environment that is not completely observed must account for uncertainty. In the Wumpus World Problem [1] the environment is not entirely observed so a series of instructions is created that can be followed at any state. This set of instructions is generated through the use of a Markov Decision Process. However, the decision process continues to generate a new set of instructions based on observations made at each state.The new set of instructions come from a combination of policy evaluation and improvement. After moving to a new state the surrounding states are evaluated based on the current set of instruction or policy. Then during policy improvement a new policy is generated based on the possible moves that could be made from the current state. This allows for the agent to make decisions dynamically creating the most efficient path to the goal.

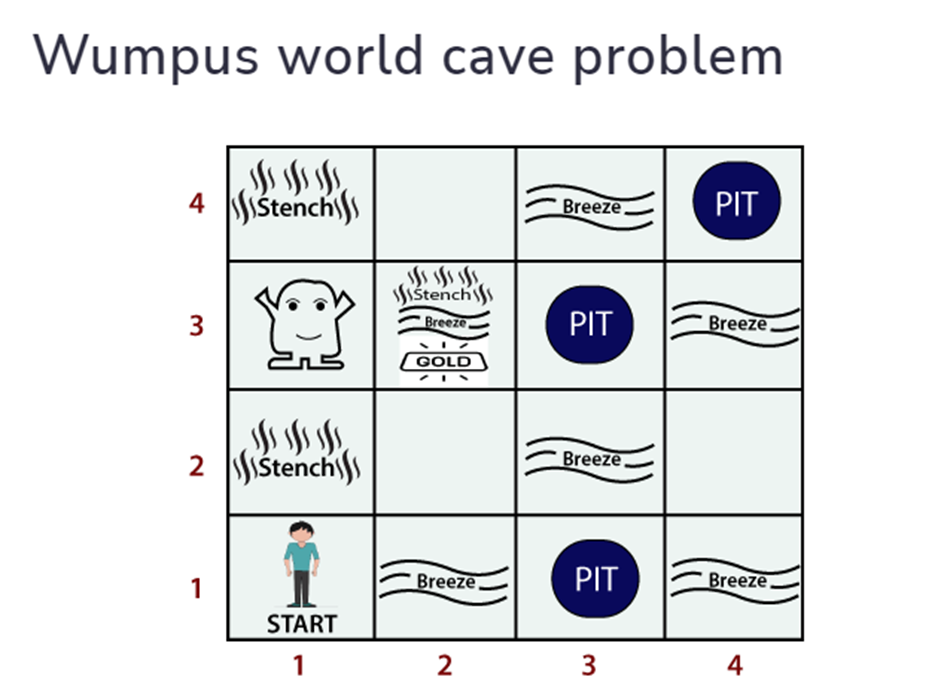
# 1 Introduction

A Markov decision process (MDP) is a sequential decision problem for a stochastic environment that is defined by the following: a set of states, a set of actions, a transition function, a reward function, and a start state [2]. A state describes the current situation that the agent is in whereas an action is an activity that the agent can perform in any of the states. The transition function details the probability of the agent transitioning to the next state from their original state. The reward function denotes the total rewards an agent would receive when undergoing an action at the relevant state. The solution of an MDP is a policy, which essentially consists of a set of actions that an agent should take for any state that they could possibly be in. A policy that maximizes expected utility and is the primary objective to achieve in an MDP is the optimal policy. One of the primary dynamic algorithms used to solve an MDP and calculate the optimal policy is policy iteration. The policy iteration algorithm alternates between policy evaluation and policy improvement until the policy converges. During policy evaluation, the utilities of each state are calculated based on the current policy. Afterwards, a new policy is created during policy improvement with the one-step look-ahead, which focuses on the agent’s next possible action from their future state.

**2 Dataset and Features**

The dataset we are given comes in the form of a four-square by four-square grid, consisting of 16 possible states. [1, 1] is where our agent will start and remain as the default position, while the Wumpus Monster is located in [1, 3]. The “pits” are located in [1, 3], [3, 3], and [1, 4] and act as walls that information will not be entered into, and “Gold” is located at [2, 3]. “Stench”, located in [1, 2], [1, 4], and [2,3] , and “Breeze”, located in [2, 1], [2,3], [3, 2], [3, 4], [4, 1], and [4,3], are to be treated as empty cells which the player can enter information into. For each state that the agent is in, given that they are not the pit cells, the gold cell, or wumpus cell, a valid movement can be attached as an action (up, down, left, right). (Fig. 1)

**Fig. 1: Wumpus world cave problem**

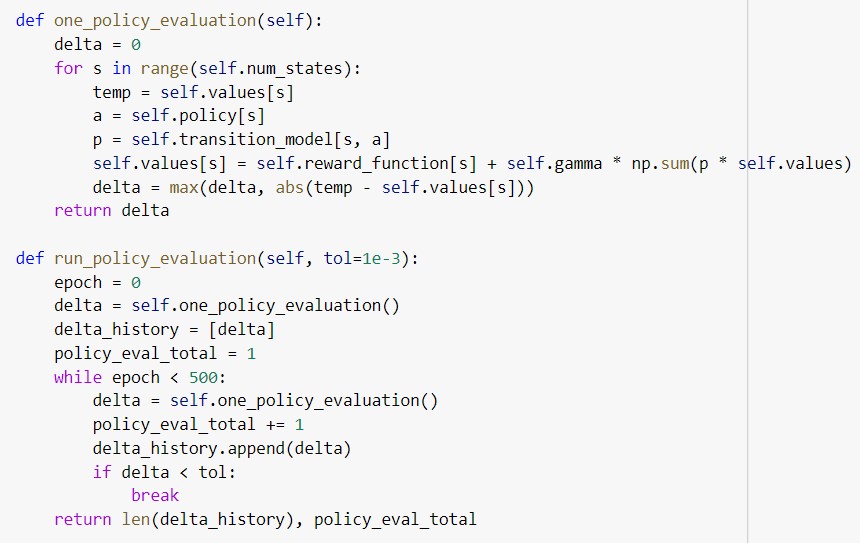
****

**3 Method**

**3.1 Policy Evaluation**

At the beginning of each iteration of our Wumpus World model, we will be evaluating the utility of each cell on the grid excluding the Wumpus cell, the Gold cell, and Pit cells. This is calculated from the sum of the immediate reward of the cell currently being checked and the utility of the successor state that the action established by the policy leads to. (Fig 2)

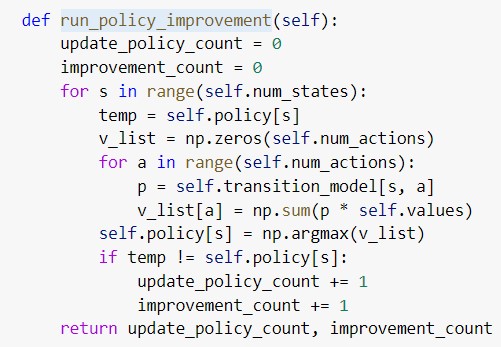
**Fig. 2: Algorithm for Policy Evaluation**



**3.2 Policy Improvement**

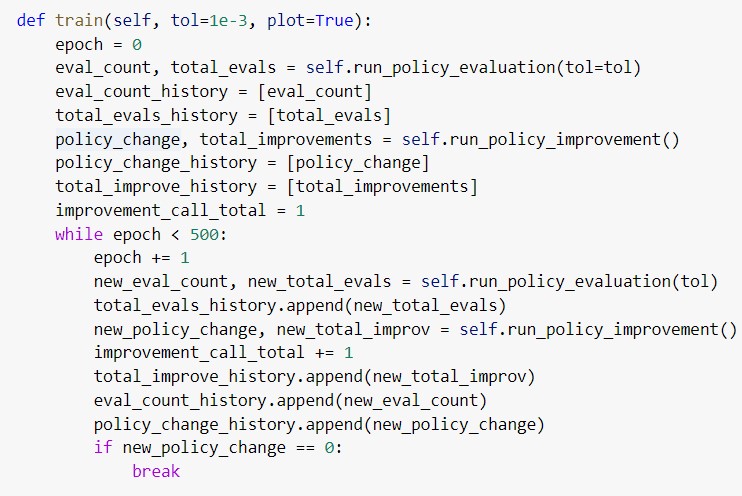
After policy evaluation is carried out, the program will improve upon the current policy, meaning that it will change what actions the agent should take at a given state depending on the reward the agent will receive for carrying out that action. To determine this, the program will calculate the total reward that the agent would receive from carrying out each possible action (up, down, left, right). The policy improvement algorithm will apply the action that produces the most reward to the policy for that given state and update the assigned action for each state accordingly. (Fig. 3)

**Fig. 3: Algorithm for Policy Improvement**



The program will continue the cycle of policy evaluation and improvement until the model reaches convergence, the point where changes to the model become minimal, thus reaching the optimal policy. The total amount of times policy evaluations and policy improvements are conducted are recorded by the training algorithm as well. (Fig. 4)

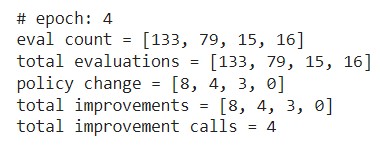
**Fig. 4: Training Algorithm**



**4 Experiments and Results**

Policy evaluation was conducted on the Wumpus World model a total of 243 times; additionally; policy improvement was conducted on the model a total of 4 times. Lastly, updates to actions were conducted a total of 15 times during training. (Fig. 5)

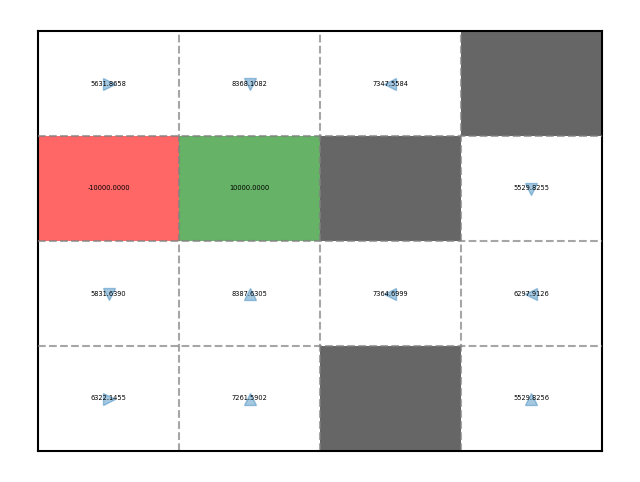
**Fig. 5: Tally of Policy Evaluation/Improvement Calls**



After training, the optimal policy for the model was produced. (Fig. 6) For each empty cell, the agent was assigned actions that would not only draw the agent closer to the gold tile, but would also ensure the agent reaches the Wumpus tile as little as possible.

A notable outlier in this grid is the cell [2,1], where the agent is assigned to go down, which leads them further away from the gold tile, rather than left, which leads them to a tile adjacent to the gold tile. This is likely due to the fact that the predicted reward for going right is significantly reduced to the possibility that the agent may end up going up and reaching the Wumpus tile; meanwhile, if the agent is assigned to go down, then it is certain that the agent will choose an action that will only grant them a net gain.

**Fig. 6: The optimal policy found for the model**



**5 Conclusion & Future Work**

To figure out the optimal policy for the Wumpus World grid, we had implemented an algorithm for policy iteration to be performed on the dataset’s assigned policy. First, the program would evaluate the currently assigned policy by updating utility values until the values converged. Secondly, the program would improve the policy by evaluating which action the player should take for the best reward at each cell. The cycle of evaluation and improvement would continue until the policy improvement results in zero updates. With our algorithms, the program had formed the optimal policy with 243 iterations of evaluation and 4 iterations of improvement; each empty cell was assigned an action that would steer clear from Wumpus and eventually reach the gold tile. In the future, modifications to the improvement algorithm should be made to account for the distance relative to the gold cell so that the optimal policy doesn’t assign actions that, while safely avoiding Wumpus, do not lead the player farther from the goal like with cell [2,1].

# References

[1] Arooj Fatima. What is the Wumpus World in artificial intelligence? Retrieved May 1, 2023 from <https://www.educative.io/answers/what-is-the-wumpus-world-in-artificial-intelligence>

[2] Russell, S. J., & Norvig, P. (2016). Sequential Decision Problems. In Artificial Intelligence: A Modern Approach. essay, Pearson.