Artificial Intelligence

Lab4 Report

Markov Decision Process

**Names & Ids :**

Pola Qulta 7685

Peter Mina 7357

Nour Mohamed Mahmoud 7591

**Reinforcement Learning in GridWorld: Value and Policy Iteration**

**Introduction**

This report details the implementation and analysis of value iteration and policy iteration algorithms for a GridWorld problem. The agent in this problem has four actions (Up, Down, Right, and Left), with a transition model such that 80% of the time, the agent moves in the intended direction, and 10% of the time, it moves at right angles to the intended direction. Collisions with walls result in no movement. The primary goal is to determine the optimal policy for the agent to maximize its rewards, using both value iteration and policy iteration techniques, for different reward values (r = 100, 3, 0, -3).

**Problem Statement**

The agent operates in a 3x3 grid with the following conditions:

* The agent has four actions: Up, Down, Right, and Left.
* The transition model is probabilistic: 80% chance to move in the intended direction, 10% chance to move at right angles to the intended direction.
* Collisions with walls result in no movement.
* The discount factor for rewards is 0.99.

We need to implement value iteration and policy iteration algorithms for different values of the reward parameter 𝑟*r*, and show the resulting policies for each case:

* 𝑟=100
* 𝑟=3
* 𝑟=0
* 𝑟=−3

**Implementation**

**GridWorld Class**

The **GridWorld** class is designed to encapsulate the grid environment and the necessary methods for both value and policy iteration.

**Initialization**

The class is initialized with:

* **grid\_size** (default 3)
* **discount\_factor** (default 0.99)
* **penalty** (default -1)

**Transition Probabilities**

The transition probabilities for each action are defined, considering the probabilistic nature of movement:

* 80% chance to move in the intended direction
* 10% chance to move at right angles

**Grid Initialization**

The **initialize\_grid** method sets up the grid with a specific reward value *r* and a terminal state at (0, 2) with a reward of 10.

**Value Iteration**

The **value\_iteration** method iteratively updates the value function 𝑉*V* for each state, based on the Bellman equation: 𝑉(𝑠)=𝑅(𝑠)+𝛾max⁡𝑎∑𝑠′𝑃(𝑠′∣𝑠,𝑎)𝑉(𝑠′)*V*(*s*)=*R*(*s*)+*γ*max*a*​∑*s*′​*P*(*s*′∣*s*,*a*)*V*(*s*′) The method converges when the change in values between iterations is below a small threshold.

**Policy Evaluation**

The **policy\_evaluation** method calculates the value function 𝑉*V* for a given policy by iteratively updating the values until convergence.

**Policy Improvement**

The **policy\_improvement** method updates the policy by choosing the action that maximizes the value function at each state.

**Policy Iteration**

The **policy\_iteration** method alternates between policy evaluation and policy improvement until the policy converges to an optimal policy.

**Best Action Finder**

The **find\_best\_action** method determines the best action to take from a given state based on the current value function.

**Running the Algorithms**

The **run\_value\_iter\_algo** and **run\_policy\_iter\_algo** methods run the value iteration and policy iteration algorithms respectively for the specified reward values.

**Analysis**

**Intuitive Explanation of Policies**

1. **When 𝑟=100:**
   * **The terminal state at (0, 0) provides a very high reward of 100.**
   * **The terminal state at (0, 2) provides a reward of 10.**
   * **Given the high reward at (0, 0), the policy will strongly favor actions that lead to (0, 0) from most positions, because reaching (0, 0) is more rewarding than reaching (0, 2).**
   * **The policy will likely direct the agent to move towards (0, 0) even if it has to take longer paths, due to the significantly higher reward.**
2. **When 𝑟=3*:***
   * **The terminal state at (0, 0) provides a modest reward of 3.**
   * **The terminal state at (0, 2) provides a reward of 10.**
   * **In this case, the policy will favor reaching (0, 2) since the reward is substantially higher than 3.**
   * **The policy will likely direct the agent towards (0, 2) as it provides a better payoff, despite the initial reward of 3 at (0, 0).**
3. **When 𝑟=0:**
   * **The terminal state at (0, 0) provides no reward (0).**
   * **The terminal state at (0, 2) provides a reward of 10.**
   * **Here, reaching (0, 0) is neutral, offering no benefit.**
   * **The policy will strongly favor reaching (0, 2) as it is the only rewarding terminal state.**
   * **The agent will be directed towards (0, 2) from all positions, as reaching (0, 0) offers no incentive.**
4. **When 𝑟=−3:**
   * **The terminal state at (0, 0) imposes a penalty of -3.**
   * **The terminal state at (0, 2) provides a reward of 10.**
   * **In this scenario, reaching (0, 0) is actively discouraged due to the negative reward.**
   * **The policy will very strongly favor reaching (0, 2) to avoid the penalty.**
   * **The agent will take actions to avoid (0, 0) and move towards (0, 2) to achieve a positive outcome.**

**Sample runs:**

**1-Value iteration**

**A computer screen shot of a black screen

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**A computer screen shot of a computer program

Description automatically generated**2-Policy iteration**

A computer screen shot of a program

Description automatically generated

A computer screen shot of a program

Description automatically generated