CS747-Assignment 2

Vansh Kapoor
200100164

1 Task 1

Value Iteration

For Value iteration, I initialized the the Value function with $\bar{\mathbf{0}}$ (this initialization is used in both Howard Policy Iteration as well as Policy Evaluation) and therefore while implementing the Bellman Optimality Operator for Value iteration, I did not update the value function for terminal states. My process of application terminates when the difference between V_t and V_{t+1} for all state is at most 10^{-12}. That is, successive application of the Bellman operator continues until:

$$\|V_{t+1} - V_t\|_\infty < 10^{-12}$$

Howards Policy Iteration

For policy evaluation in Howard Policy Iteration, I used successive applications of Bellman operator instead of matrix inversion as I felt that would lead to faster convergence. Here I switched the actions for all the states with at least one improvable the actions. The tolerance that I set of value function evaluation for a policy was similar to that for value iteration which was 10^{-12}.

Linear Programming

In the implementation of Linear Programming, for the terminal states I set their lower and upper bounds as 0 and for the rest of states I framed the greater than inequality using the Bellman equation. The objective was to maximize the negative of the sum of the value function variables for all states.

For episodic MDPs my approach fixes the value function for terminal states as 0 and also implements self loops with transition probability 1 and reward 0 for terminal states. Other than this my approach doesn't differentiate between and episodic and continuing MDPs. In other words,

$$T(s_T, a, s_T) = 1 \quad \text{and} \quad T(s_T, a, s) = 0 \quad \forall s \in S \setminus \{s_T\}, \forall a \in A$$
$$R(s_T, a, s) = 0 \quad \forall s \in S, \forall a \in A$$

The default algorithm is set to be Value iteration using its tolerance both its accuracy and computation time can be adjusted to meet the requirements.

2 Task 2

2.1 MDP Formulation

Encoder

`encoder.py` uses the states from the state-opponent policy files along with a ”Win” state and a ”Loss” state. I have defined a function `pos` that when given a player’s position outputs the player’s position in the next time step according to the 4 movement actions possible (Left, Right, Up, Down). It outputs the subsequent position if it is legal and -1 if the player moves outside the grid. My other functions: `prob_pass` and `shoot` give the probabilities that a pass/shoot given the positions of the players and the subsequent position of the opponent.

I iterated over each state in my MDP and updated the transition-probability matrix and the reward matrix using the outgoing transitions from the current state under consideration. Only transition to the Win state awards a reward of one. Thus in such a formulation of the MDP, my Value function of a state is the expected number of goals starting from that state. Here my Win and Loss are my terminal states, and thus my MDP formulation is episodic.

Planner

The output of `encoder.py` is used by is used by `planner.py` to find the optimal policy and Value Function (using default algorithm - Value iteration) of these 8194 states (8192 states from state-opponent policy and 2 Win, Loss states).

Decoder

`decoder.py` uses the state-opponent policy files as a reference to understand the order of optimal Value functions and actions returned by `planner.py`. `decoder.py` ignores the values obtained for last two states, i.e., Win and Lose state.

2.2 Plots

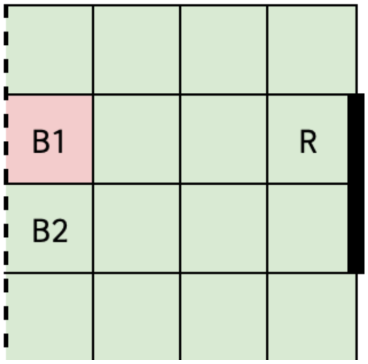

Figure 1: Variation of Expected Number of Goals with p for q=0.7

Variation of Value Function with p for fixed q

Figure 2 shows the variation of the expected number of goals with p for q=0.7 for the positions of the players shown in Figure 1. Consider the case when p=0. There is zero probability of losing possession of the ball by the movement of either players, unless tackling condition occurs which can be easily avoided due the initialization condition and due to the known strategy of the opponent. Therefore the player with the ball can move to the right edge of the grid to maximize the probability of shoot without losing possession of the ball. Thus for p=0 expected number of goals is just probability of goal when the player is at the right edge of the grid which is nothing but p.

Also observe that when p gets closer and closer to 0.5, the probability of losing possession due to movement of player with the ball → 1. Also though passing probability remains high enough, passing the ball gives no advantage as both are equally distant from the right edge. Therefore directly shooting is the optimal move which in fact has a probability of q-0.6 = 0.1 which matches with our observation!

The intermediate probabilities might indicate optimal play such as: movement of player without the ball towards the right edge, passing and shooting directly or movement towards the right edge and then shooting. Intuitively also there should be a rapid decay and then a constant Value function with p. This is because the value function of the state initially should decrease as probability of losing possession increases with movement and later the this probability of losing possession crosses a limit the player changes his policy by directly shooting towards the goal from his current position whose probability of success is independent of *p*.

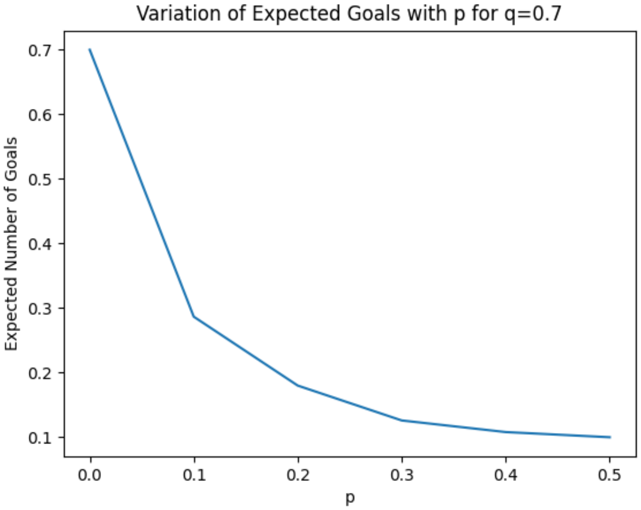

Figure 2: Variation of Expected Number of Goals with p for q=0.7

Variation of Value Function with q for fixed p

Figure 3 shows the variation of the expected number of goals with q for p=0.3 for the positions of the players shown in Figure 1. It is quite predict the increase of Value function and hence the expected number of goals due to increase in q which determines probability of scoring a goal.

By careful observation we can see a point of discontinuity of derivative at $q = 0.8$ and a linear region from q $\in [0.8, 1]$. This probably indicates a shift in policy near q=0.8. From q $\in [0.8, 1]$ we observe that the value function is just q-0.6. This matches our intuition as when the probability of goal increases, the player instead of taking a risk of losing possession by moving, directly shoots from his current position.

Whereas when q is near 6, the player's probability of goal, shooting from the current position $\rightarrow 0$ and hence he moves towards the goal and then shoots. Intuitively here the Value function (hence the policy used) is restricted by the small value of q, whereas in the case above the Value function is restricted sue to the relatively large value of p (losing due to movement)

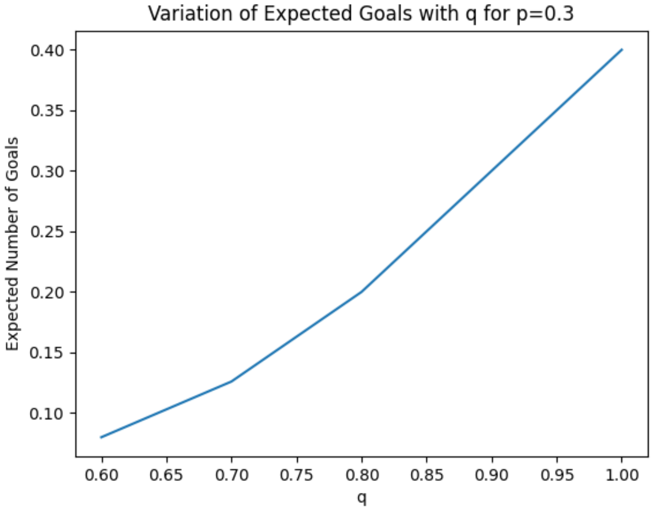

Figure 3: Variation of Expected Number of Goals with q for p=0.3