CS 747 Assignment 2
MDP Planning

Amruta Mahendra Parulekar, 20d070009

October 15, 2023

1 Task 1

First I used argument parsing to get the algorithm, mdp file and policy.

1.1 Value Iteration

For value iteration, the Value function was initialized to zero. A loop was run that calculated the action value function for every state. The updated value function was maximized over all actions to get the best value function for the states. A threshold was set and this process was repeated till the new value function was very close to the one created in its previous loop. Then from this value function, an action value function was created to find the best actions (Policy)(where value of Q was max)

$V_0 \leftarrow$ Arbitrary, element-wise bounded, n-length vector.
$t \leftarrow 0$.
Repeat:
  For $s \in S$**:**
    $V_{t+1}(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T(s,a,s')\,(R(s,a,s') + \gamma V_t(s'))$.
    $t \leftarrow t+1$.
Until $V_t \approx V_{t-1}$ (up to machine precision).

1.2 Howard policy Iteration

Hoe HPI, fist a random policy was chosen by initializing it to random integers for actions. After that a new value function was created using this policy. Now the action value function was maximized over all actions and a threshold was used to compare it to the value function. Basically the aim is to improve all improvable states. Finally, a new policy was also created by taking the best actions. It is a greedy algorithm which switches all improvable states.

$\pi \leftarrow$ Arbitrary policy.
While π has improvable states:
  $\pi' \leftarrow$ PolicyImprovement(π).
  $\pi \leftarrow \pi'$.
Return π.

1.3 Linear Programming

First, I created a model of pulp, and it was given a function to maximize(negative sum of value functions) and n constraint equations were set based on the equation staight in class. The solver gave the best value function, from which beast action was calculated by maximizing the action value function.

$$\text{Maximise} \left(-\sum_{s \in S} V(s) \right)$$

subject to

$$V(s) \geq \sum_{s' \in S} T(s, a, s')\{R(s, a, s') + \gamma V(s')\}, \forall s \in S, a \in A.$$

1.4 Policy evaluation

The actions were extracted from the policy file and then the policy evaluation function was then iteratively improved so that it reached a threshold difference from its previous case and these values were printed along with the policy actions.

2 Task 2

2.1 Encoder

First, I read all the elements in from the given file. Now I made a case for every action.
For the movement actions, I considered the cases where the opponent exchanged places with the player with the ball as a tackle. I printed the successful tackles and actions and then even failures.
For passing, I located the new position of the opponent and checked if it was between the players. If not, I calculated the probability of passing based on the distance between the players and changed the state, else I halved the transition probability. Failure was taken to the end state.
For shooting. I checkedif the opponent after motion was in front of the goalpost. If he was, I haled the probability which I had calculated using teh distance of the player with the ball from the goalpost. Failure was taken to the end state.
For each case, I calculated the probability Reward was taken to be 0 except for the win, where reward was 1. This probability, along with initial state action and the recalculated and encoded final state were printed.

2.2 Decoder

Decoder was made to map the original names and state indices together and print the final policy in the required format.

2.3 MDP planner

This was made for task 1 and its Value iteration algorithm was used for the football question.