CS 747 : Programming Assignment 2
Aayushi Barve, 210010002

1 Task 1: MDP Planning Algorithms

For both value iteration and Howard's policy iteration, I am starting with policy - choose action 0 for every state. For value iteration I have picked a threshold of 0.00000001. I had initially implemented all 3 algorithms with 3 dimensional numpy arrays for transition functions and rewards. This took up a lot of memory for task 2, where there were 8193 states and the size of these arrays was 8193*16*8193, so I changed the structure of value iteration to consider only those ($State$, $Action$, $State$) groups for which the transition function is zero, and only used this set for further calculation rather than also considering all the elements with 0 transition probabilities. This made the computation and encoding for task 2 a lot faster.
I also calculated $\sum_{S'}(T(S,A,S') * R(S,A,S'))$ and stored it as a 2 dimensional array C that I passed to the 3 functions for linear programming, value iteration and Howard's policy iteration rather than recalculating this in every iteration of the algorithms, since C remains unchanged.

2 Task 2: MDP for 2v1 Football

- MDP formulation for the football problem
  - Actions
    The actions are the same as given in the problem statement - actions {0,1,2,3} signify an attempt to move of B1 in {L,R,U,D}, actions {4,5,6,7} signify an attempt to move of B2 in {L,R,U,D}, action 8 signifies an attempt to pass the ball from the player that has the ball to the player that doesn't, action 9 signifies an attempt to shoot a goal.

  - Design of states
    Each state is designed in such a way that it is a 7 digit integer. The first 2, second 2, third 2 digits signify the positions of players B1, B2 and R respectively. The last digit tells us which player is in possession - if it is 1, player B1 is in possession of the ball and if it is 2, player B2 is in possession of the ball. We assign an integer from 0 to 8191 to each of these states in increasing order- starting from 0101011, we first flip possession, then flip positions of R, B2, B1 in that order. The terminal state corresponding to the end of the game (irrespective of whether the end of the game was due to a tackle, an attacker going out of bounds, loss of possession while moving, an unsuccessful pass or a failed goal) is 8192. This forms a base system of sorts - total number of states is 16*16*16*2.
    So if the agent is currently at state S, where S belongs to {1,2,...,8191}, taking actions {0,1,2,3,4,5,6,7} successfully will cause an increment in S by {-512, 512, -4*512, 4*512, -32, 32, -4*32, 4*32} respectively. Similarly if player R moves {L,R,U,D}, S gets incremented by {-2, 2, -8 , 8} respectively.
    If B1 has the ball and a pass is successful, state S gets incremented by 1. If B2 has the ball and a pass is successful, state S gets decremented by 1.

- Transitions

  We set all transition function values to zero initially. We start with current state S, and action that we want to take, A. We are told to assume that R has moved first for any action A our agent wants to undertake. Depending on the current position of R according to state S, R can move to a minimum of 2 and a maximum of 4 intermediate states {R1, R2, R3, R4}. Since R can never go out of bounds, for states where R is at the boundary of the football field, it can move to either 2 places (R is at corner), or to 3 places (R is on an edge but not on a corner).

  Our agent now takes action A from state R_i with some probability r_i. Let us take the example of a move in a particular direction. If a move is successful, the agent transitions from state R_i to state S_i with some probability $P(move is successful|agent is at R_i)$, which is calculated according to the rules of the game. So we increment $T(S, A, S_i)$ by $P(move\, is\, successful|agent\, is\, at\, R_i) * P(agent\, is\, at\, R_i)$ that is $P(move\ is\ successful|agent\ is\ at\ R_i) * r_i$.

  To calculate the probability of the game ending on taking action A from state S, we subtract the sum of the probabilities of the agent ending up at S_1, S_2, S_3 S_4 starting from state S and taking action A from 1.

  $T(S, A, 8192) = 1 - \sum_{S_t \in \{S_1, S_2, S_3, S_4\}} T(S, A, S_t)$

  A similar logic is followed for passing. There are a minimum of 2 and a maximum of 4 intermediate states depending on the movement of R, and a successful pass from these states leads us to 4 possible final states given a pass is attempted from the initial state. The probability of the game ending is calculated as above.

  For shooting however, the probability of the game ending irrespective of whether the goal is successful or not is 1.

  So $T(S, 9, 8192) = 1 \,\forall\, S$-

- Rewards

  The agent gets a reward of 1 only when a player in possession of the ball attempts to shoot a goal and is successful. The reward for all other situations is zero. The effective reward will be the expectation value of the reward obtained given position of the player in possession of the ball and position of the defender. As above we assume R moves first, the probability of the goal being successful depends on where R has moved to. Using the intermediate state, we find the probability of a successful goal, and multiply this with the probability of ending up in that intermediate state using policy of R. We sum these up over all intermediate states to get $R(S, 9, 1892)$.

- Encoding

  We include only the ($State$, $Action$, $State$) tuples for which $T(S, A, S')$ is non-zero in the given format. The type of the MDP is episodic, and the discount factor is 1. Number of states is 8193, number of actions is 10, and there is one terminal state namely 8192.

Graphs

- Keeping q constant and varying p for greedy defence policy

  p is the parameter that controls whether a player is able to move with the ball successfully or not. The probability of the game ending is proportional to p (p if moving without the ball, $2 * p$ moving with the ball). The larger p is, the "weaker" the attackers are in terms of handling the ball. So as p increases, the probability of the attackers winning the game decreases.

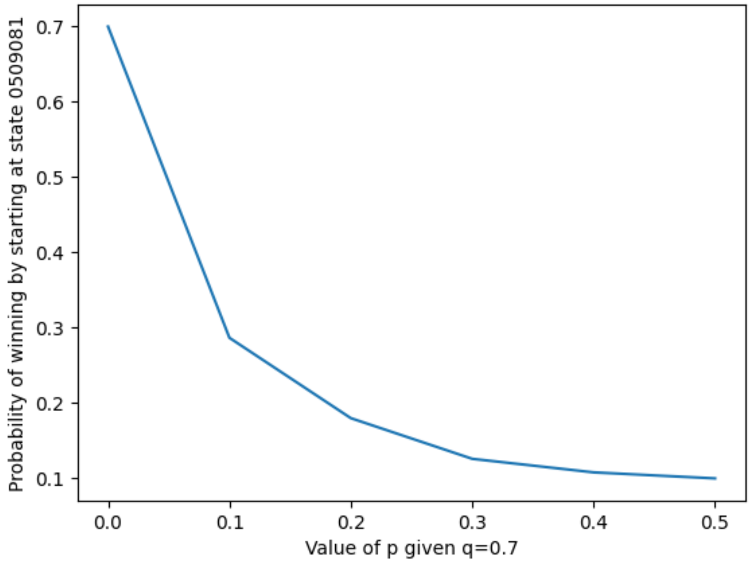

Figure 1: Probability of winning the game vs p

- Keeping p constant and varying q for greedy defence policy
  q is the parameter that controls the probability of a successful pass or a successful goal. The larger q is, the "stronger" the attackers are in terms of passing or shooting. Probability of successful tackle or a failed goal decreases. So as q increases, the probability of the attackers winning the game increases.

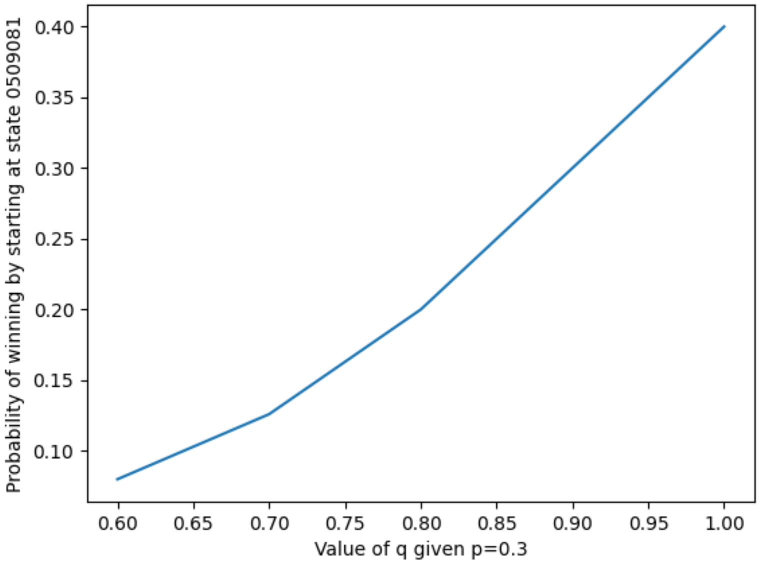

Figure 2: Probability of winning the game vs q

- Expected number of goals for all 3 policies
  For the case $p = 0.1$ and $q = 0.7$, I have evaluated the expected number of goals to be the average of the value function for every possible initial state.
  - Greedy Defense
    Expected number of goals = 0.4743952255859375

- Park the Bus
  Expected number of goals = 0.381978359375
- Random Policy
  Expected number of goals = 0.4713580712890625

The expected number of goals for the second policy is significantly lesser since R shuffles in front of the goal itself, and this causes a significant reduction in the probability of scoring a goal successfully.

The expected number of goals for greedy defense is also more than that for random policy, which is a little bit counter-intuitive. In order to pursue the player with the ball, R may drift away from the goal quite often, leading to a higher probability of scoring a goal successfully for the attackers. Furthermore since p is low and q is high, the attackers are "strong" players, which means they can score a goal or pass successfully quite often, with a lower probability of being tackled, going out of bounds or missing the goal. So for this case, greedy defense gives us a higher number of expected goals. For different values of p and q, which sort of describe the "skill" of the attackers, the expected number of goals for greedy defense and random policy might follow a different trend, since we have a trade-off between R drifting away from the goal and R gaining possession of the ball since the attackers are "weak" players.

- Plotting

  I have included a folder named "plot_policy" with my submission. It has the decoded policy files for the given parameters p and q and greedy defense policy, and the output policy files for $p = 0.1$ and $q = 0.7$ for park the bus and random policy. It also includes the plots, the plotting script used and a text file with the expectation values for the 3 policies of R given $p = 0.1$ and $q = 0.7$