2. Football MDP (*21 points*)

Earlier this season, Stanford's football team rallied from a 29-point deficit to achieve a historic, double-overtime comeback win against Colorado. Imagine you are a statistical analyst for the Stanford football team. Your job is to decide which plays Stanford should run to maximize its points.

We'll use a Markov Decision Process (MDP) to model the game of football. We explicitly define our MDP as follows, but you can also reference the diagram on the next page (which presents the same information visually):

1. **States**: The states are defined as:

   $$\text{States} = \{20 \text{ yards}, 10 \text{ yards}, \text{Touchdown}, \text{Field Goal}, \text{Missed Field Goal}\}$$

   We define our start and terminal states as follows:

   $$s_{\text{start}} = 20 \text{ yards}$$

   $$\text{IsEnd}(s) = \begin{cases} \text{true} & \text{if } s \in \{\text{Touchdown}, \text{Field Goal}, \text{Missed Field Goal}\} \\ \text{false} & \text{if } s \in \{20 \text{ yards}, 10 \text{ yards}\} \end{cases}$$

2. **Actions**: The actions for the non-terminal states are:

   $$\text{Actions}(s) = \{\text{Pass}, \text{Kick}\}$$

3. **Rewards**: We receive a positive reward for scoring a touchdown or making a field goal, but negative rewards otherwise. Formally, our reward function $\text{Reward}(s, a, s')$ is defined as follows:

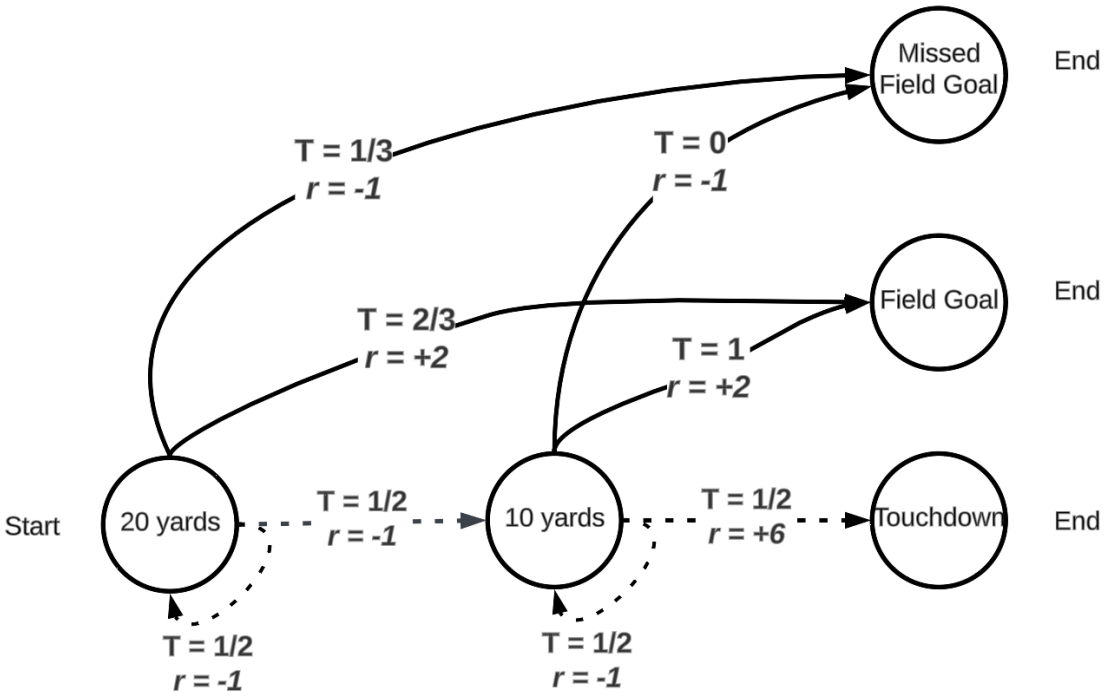
   $$\text{Reward}(s, a, s') = \begin{cases} +6 & \text{if } s' = \text{Touchdown} \\ +2 & \text{if } s' = \text{Field Goal} \\ -1 & \text{otherwise} \end{cases}$$

4. **Transition Probabilities**: The transition probabilities are defined as follows:

s	a	s'	$T(s, a, s')$
20 yards	Pass	10 yards	1/2
20 yards	Pass	20 yards	1/2
10 yards	Pass	Touchdown	1/2
10 yards	Pass	10 yards	1/2
20 yards	Kick	Field Goal	2/3
20 yards	Kick	Missed Field Goal	1/3
10 yards	Kick	Field Goal	1
10 yards	Kick	Missed Field Goal	0

5. **Discount Factor**: $\gamma = 1$.

The diagram below visualizes the MDP we just defined. The **solid lines** correspond to the action Kick, while dashed arrows correspond to the action Pass.



---

a. (*9 points*) The head coach wants to know the policy that yields the most points. Complete three iterations of Value Iteration to identify the optimal action at each non-terminal state. Specifically, please calculate the value of $V_{\text{opt}}^{(t)}(s)$ for $t \in \{0, 1, 2, 3\}$ and $s \in \{20 \text{ yards}, 10 \text{ yards}\}$. We initialize the value of each state at $t = 0$ to be 0, i.e., $V_{\text{opt}}^{(0)}(s) = 0$ for all s. **Fill in your answers in the table below.**

s	$V_{\text{opt}}^{(1)}(s)$	$V_{\text{opt}}^{(2)}(s)$	$V_{\text{opt}}^{(3)}(s)$
20 yards			
10 yards			

Solution Grading rubric: one point per box.

$$V_{\text{opt}}^{0}(10 \text{ yards}) = 0$$
$$V_{\text{opt}}^{0}(20 \text{ yards}) = 0$$
$$V_{\text{opt}}^{1}(10 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[6 + (1)(0)] + (1/2)[-1 + (1)(0)] = \boxed{5/2} \\ \text{Kick} : (1)[2 + (1)(0)] + (0)[-1 + (1)(0)] = 2 \end{cases}$$
$$V_{\text{opt}}^{1}(20 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[-1 + (1)(0)] + (1/2)[-1 + (1)(0)] = -1 \\ \text{Kick} : (2/3)[2 + (1)(0)] + (1/3)[-1 + (1)(0)] = \boxed{1} \end{cases}$$

$$V_{\text{opt}}^{2}(10 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[6 + (1)(0)] + (1/2)[-1 + (1)(5/2)] = \boxed{15/4} \\ \text{Kick} : (1)[2 + (1)(0)] + (0)[-1 + (1)(0)] = 2 \end{cases}$$
$$V_{\text{opt}}^{2}(20 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[-1 + (1)(5/2)] + (1/2)[-1 + (1)(1)] = 3/4 \\ \text{Kick} : (2/3)[2 + (1)(0)] + (1/3)[-1 + (1)(0)] = \boxed{1} \end{cases}$$

$$V_{\text{opt}}^{3}(10 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[6 + (1)(0)] + (1/2)[-1 + (1)(15/4)] = \boxed{35/8} \\ \text{Kick} : (1)[2 + (1)(0)] + (0)[-1 + (1)(0)] = 2 \end{cases}$$
$$V_{\text{opt}}^{3}(20 \text{ yards}) = \max \begin{cases} \text{Pass} : (1/2)[-1 + (1)(15/4)] + (1/2)[-1 + (1)(1)] = \boxed{11/8} \\ \text{Kick} : (2/3)[2 + (1)(0)] + (1/3)[-1 + (1)(0)] = 1 \end{cases}$$

Grading: 1.5 points per box.

b. (*6 points*) After each round of value iteration $t = 1, 2, 3$, what is the optimal policy based on your calculations above.

s	$\pi_{\text{opt}}^{(1)}(s)$	$\pi_{\text{opt}}^{(2)}(s)$	$\pi_{\text{opt}}^{(3)}(s)$
20 yards			
10 yards			

Solution

$$\pi_{\text{opt}}^{(1)}(\text{20 yards}) = \text{Kick}$$
$$\pi_{\text{opt}}^{(1)}(\text{10 yards}) = \text{Pass}$$
$$\pi_{\text{opt}}^{(2)}(\text{20 yards}) = \text{Kick}$$
$$\pi_{\text{opt}}^{(2)}(\text{10 yards}) = \text{Pass}$$
$$\pi_{\text{opt}}^{(3)}(\text{20 yards}) = \text{Pass}$$
$$\pi_{\text{opt}}^{(3)}(\text{10 yards}) = \text{Pass}$$

Grading: 1 point per box.

c. (*6 points*) After running the calculations above, you realize that perhaps value iteration could converge more quickly with the following modification. Define an ordering of the non-terminal states OrderedStates (for example OrderedStates = [10 yards, 20 yards]). At each iteration, we update the value of states in this order; note that the value of one state could depend on the value of another state computed in the same iteration. As an analogy, think about the difference between gradient descent and stochastic gradient descent. Here is the pseudocode for asynchronous value iteration:

Algorithm 1 Asynchronous value iteration

Initialize $V_{\text{opt}}^{(0)}(s) = 0$ for all s
for $t = 0, 1$ **do**
  $V_{\text{opt}}^{(t+1)} \leftarrow V_{opt}^{(t)}$
  for $s = 10$ yards, 20 yards **do**
    $V_{\text{opt}}^{(t+1)}(s) \leftarrow \max_a \left(\sum_{s'} T(s, a, s') \left[\text{Reward}(s, a, s') + V_{\text{opt}}^{(t+1)}(s') \right] \right)$
  end for
end for

Consider the case where the MDP is **acyclic** (for any sequence of actions, the probability of returning to the same state is 0).

1. What is the best OrderedStates such that asynchronous value iteration converges as fast as possible?

2. How would you compute the best OrderedStates?
3. How many iterations does it take asynchronous value iteration to converge to the optimal value function under the best OrderedStates?

Solution Note that when the MDP is acylic, one could just solve the MDP using dynamic programming. Therefore, we should order the states based on a topological sorting of the nodes in the graph. We can compute this ordering by simply running DFS. Given this ordering, we can proceed from the states closest to the terminal states and proceed to the start state. We can achieve convergence in only 1 iteration.