CME241 A2

Yiyun Gu

January 2021

1 Problem 1

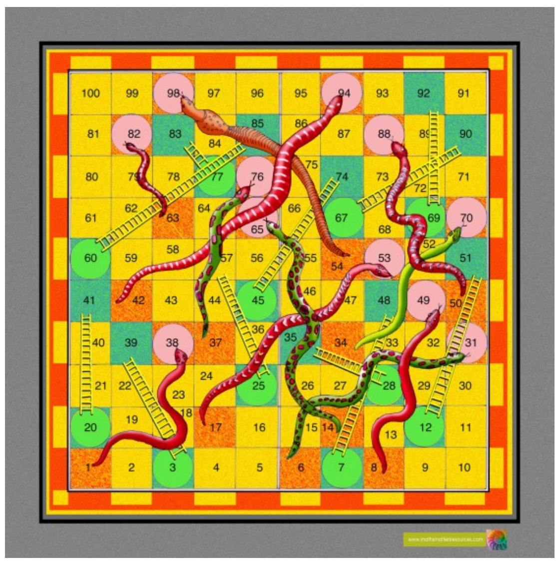

Figure 1: Snakes and Ladders

The state space $\boldsymbol{S} = \{\text{PlayerState}(i) | i = 0, ..., 100\}$ where PlayerState(0)(i) is the start state, PlayerState(i) is at position i and PlayerState(100) is the terminal state.

The transition probabilities:

If $i \leq 94$: At position i, roll a dice and the probability of reaching position $i+1, i+2, ..., i+6$ is all 1/6 if there are no snakes or ladders at position $i+1, i+2, ..., i+6$. If there is a snake or a ladder at position $i+j, j = 1, ..., 6$ which jumps to position k, then replace $P(S_{t+1} = \text{PlayerState(i+j)} | S_t = \text{PlayerState}(i)) = 1/6$ with $P(S_{t+1} = \text{PlayerState(k)} | S_t = \text{PlayerState}(i)) = 1/6$.

If $i > 94$: $P(S_{t+1} = \text{PlayerState}(100) | S_t = \text{PlayerState}(i)) = 1 - (99 - i)/6$ since rolling to a position greater than or equal to 100 means termination.

See SnakesAndLaddersMPFinite in Assignment2/snakesandladders.py

2 Problem 2

The transition map is constructed as described in Problem 1. See SnakesAndLaddersMPFinite in Assignment2/snakesandladders.py

The y-axis should be divided by 100000 since it's histogram plot for 100000 simulations. The x-axis shows the time steps to finish the game.

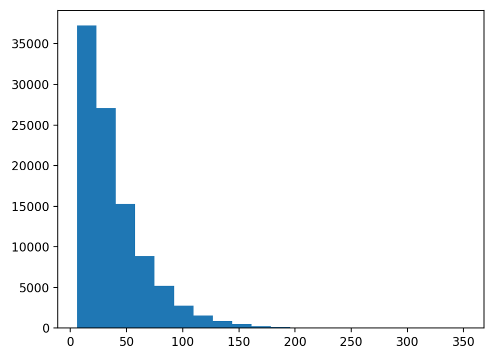

Figure 2: Probability Distribution of Time Steps

3 Problem 3

Construct the finite Markov Process of Frog Puzzle and run 50000 simulations to derive the expected number of jumps to the end. The result is 2.9. See Assignment2/frogpuzzle.py

4 Problem 4

Contruct the finite Markov Reward Process of Snakes and Ladders. Define the value function of s as the expected number of dices to finish the game from the state s. $V(s) = R(s) + \gamma \sum_{s' \in N} P(s, s')V(s')$. Set $R(s) = 1, \forall s$ and $\gamma = 1$. Compared with the next state s', the state s needs 1 more step to finish the game. $\sum_{s' \in N} P(s, s')V(s')$ generates expected number of dices from all the possible next states. The expected number of dices to finish the game is 39.6, which is approximately equal to the result derived from simulations. See SnakesAndLaddersMRPFinite in Assignment2/snakesandladders.py

5 Problem 5

Extend the first Stock Price example to a Markov Reward Process StockPriceMRP1(customized_reward_function, level_param). get_value_function(self, state: StateMP1, gamma: float, time: int) -¿ float returns the value function of the state recursively. Since the process is infinite-states and non-terminating, approximate by calculating the next 20 steps. See Assignment2/stock.py