FILA Assignment 2 Report

Roll No: 15D170013
Huzefa Chasmai

P1) **Linear Programming**:

Key Implementation Choices :

- Used the PULP Library in python to solve the Linear Programming Problem.
- In order to handle the cases of gamma = 1, we enforced the constraint on the state_(S-1) to be equal to 0, for the case of episodic MDPs.

P2) **Howard Policy Iteration**:

Key Implementation Choices :

- Used the np.linAlg package to solve the linear Bellmann equations to obtain the Value functions at each iteration.

- I am using the logic of choosing arbitrarily any improvable policy as opposed to the maximum State action function Policy.

- Also handling special cases where the mdp type is episodic, we make the last row of the matrix A = [0 … 0 .. 0 1] and B= [0] in the AV = B, so that we get the value at last state equal to 0.

- Furthermore there are cases when I get a singular matrix for A, due to equations like V_s = V_s. Those cases where manually handled to give value V_s = 0 and to make the matrix non singular.

- Also while checking the condition Q(s, a) > V(s) we introduced a threshold such that the difference is greater than the threshold. This resolved an infinite loop caused by two policies that differ very slightly in the Q(s,a) but both the policies feeling that the other is better for a case of the Gambler MDP when gamma = 1.

P3) **Gamblers MDP Problem**:

S = 101
A = 51

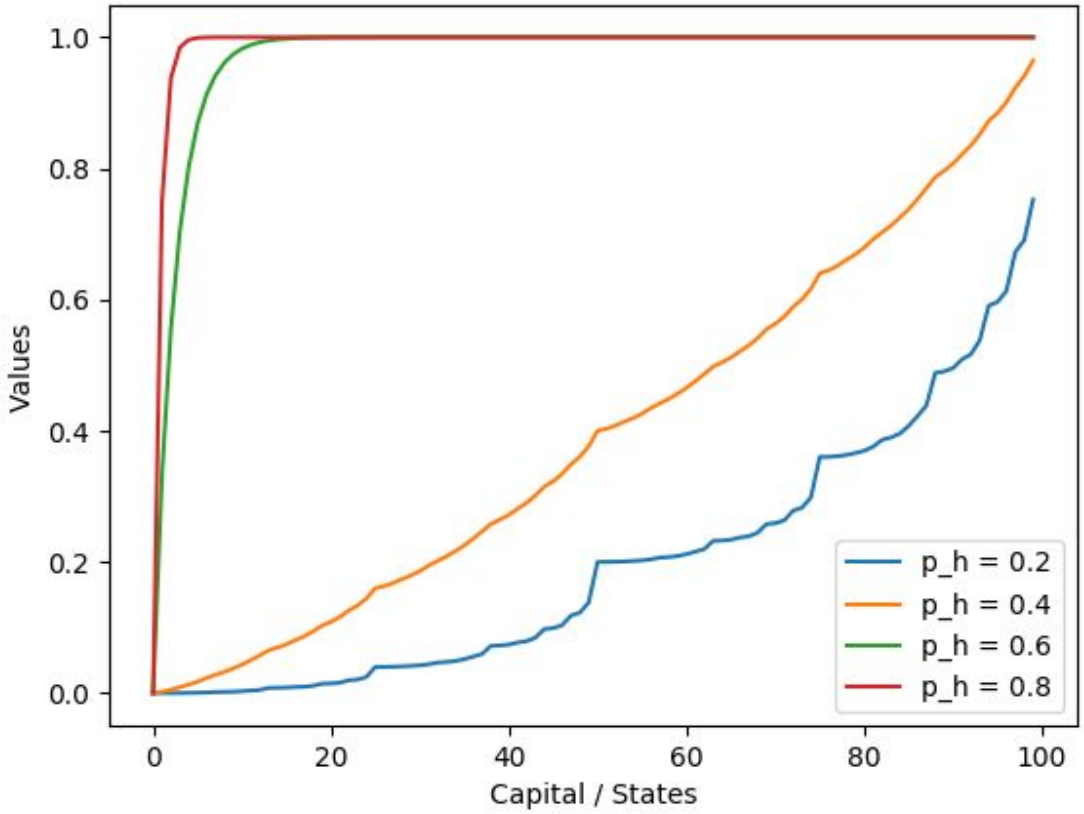
The 101 states correspond to states where the gambler has coins = State_num
51 actions correspond to the money bet, i.e. money bet = action_num. At max it makes sense to bet only min(s, S-s-1) coins since in the first case, that's the amount of coins that the gambler has and hence he cannot bet more than what he has. Also if the gambler is at a state s such that s>50, the gambler needs just S-s-1 coins more so that he can get his coins count = 100 and then leave. Hence the minimum condition and only 50 actions are possible.

Transitions :
For each state, only actions until min(s, S-s-1) are allowed, for action a there are 2 transitions possible “s - > s + a” with a probability of the Prob_Heads and “s -> s - a” with a Probability of 1 - Prob_Heads. For all other (state, action, state) triplets the probability of transition is 0. (Initially had defined transitions for each)

Rewards :
0 for all transitions except transitions of the form “s -> s + a” such that s+a = 100.
For such transitions the reward is set to 1 since this denotes that the gambler has won.

Gamma : 1
Since there is no such notion as immediate reward gets higher weightage and later step rewards get lesser by a factor of gamma and the final goal is to exit.

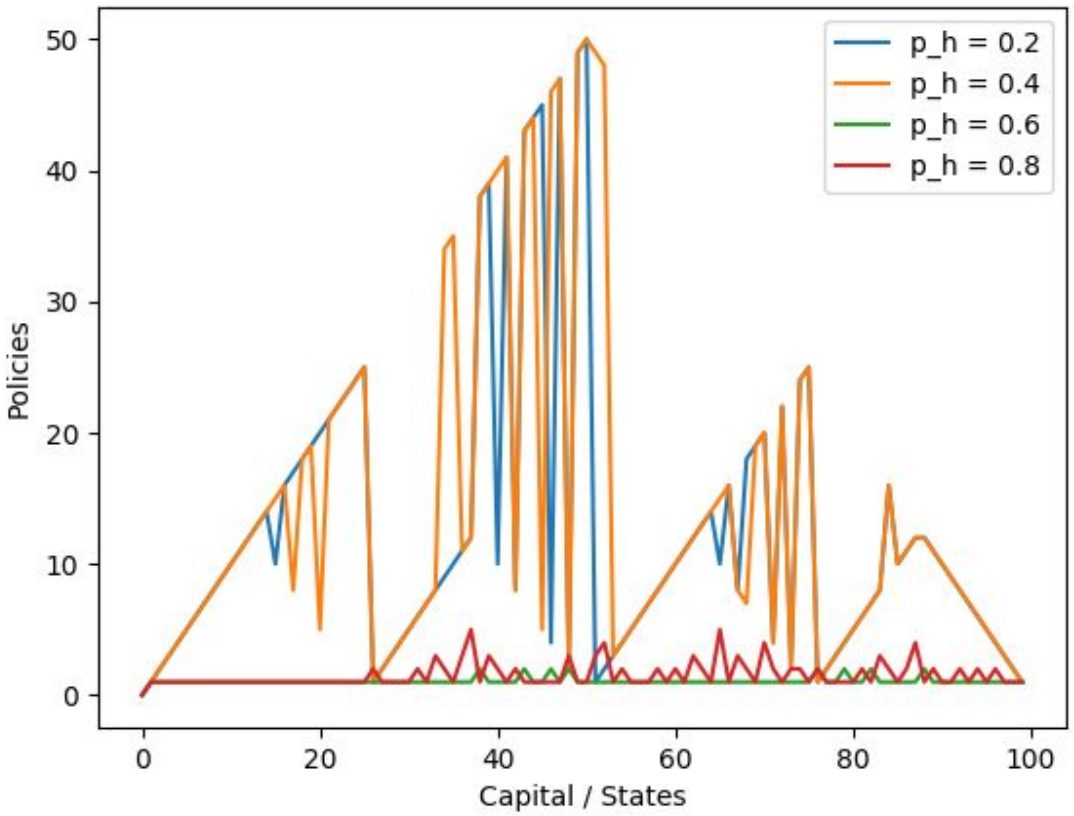
MDP Value Plot for different Values of ph :



Observations :

- The Value Function for the States is plotted in the figure. The value function > 0 as expected since there are no negative rewards.
- The Value functions for each of the ph values is an increasing function since the reward received (when reaching state 100) is definitely higher in the case when the current state is higher. That is, higher the current amount the gambler has more the chance that he will reach state 100 and win.
- Also as the value of ph increases the Value function increases for each of the states. I.e. the graphs for higher ph is above graph for lower ph for each of the states. This is also expected since higher the probability of transitioning to a higher state (getting a heads), higher the chance of the Gambler reaching state 100. The value function is 0 for terminal states (0 &100) since only transitions possible are to the same states and the rewards are 0 since constant term is zero in the Bellmans equations.

MDP Policy Plot for different Values of ph :



Observations:

- This policy plot is a bit random and depends a bit on the way the improvable policy is chosen in the case of hpi So there essentially are multiple policies corresponding to a Value Function. We arbitrarily choose the first such policy and hence the graphs for ph = 0.8 and 0.6 are pretty constant around action 1, since there is no time dependence so constantly betting 1 would give you a good chance of getting to the state 100.
- Also in the case of lp too, there are multiple policies having the same V.
- Hence didn't really bother matching this plot with the one in the book but somewhat similar trends are seen when ph = 0.2.
- Furthermore this plot is different for both lp and hpi.

Key Overall Observations:

- The MDP Value graph above is very smooth when the gamma value < 1. (Tried for values = 0.95, 0.99). This is because the case of gamma = 1 causes some complications which were handled separately. This leads to some of the unexpected spikes. The case of gamma = 1 causes these complications since there are equations of the Form V_s = V_s, in case of action zero. This leads to the action 0 being one possible policy action.
- Also this leads to the solution being unbounded in the LP case and also the matrix being singular in the case of HPI. These cases are handled separately. This case also causes two states with self loops (state 0 and state 100) and hence the LP solver is unable to solve such a problem.
- Also in HPI while checking the condition Q(s, a) > V(s) we introduced a threshold such that the difference is greater than the threshold. This resolved an infinite loop caused by two policies that differ very slightly in the Q(s,a) but both the policies feeling that the other is better for a case of the Gambler MDP when gamma = 1.

Contents :
Folder structure similar to the one asked :
Extras Folder :
data : The gambler mdp files (gamblermdp$ph.txt) and the solutions (sol_algph.txt)
data/plots : Cumulative Plots for the Value and the Policy
encodeGambler.py : The code corresponding to encodeGambler
Planner.py : The code corresponding to planner
Plotter.sh : A Bash script to generate the data and the cumulative plots
Plotting_script.py : Script to plot cumulative plots