CS747 - Assignment 2

Kalpesh Krishna
140070017
kalpeshk2011@gmail.com

1 Implementation

All algorithms were implemented in Python. A main script `planner.py` is called by `planner.sh` which in turns calls algorithms outlined in `algorithms.py`. `numpy` has been used everywhere for random number generation. `pulp` has been used to solve the LP problems.
`generate.py` is used to randomly generate MDPs of 50 states. These MDPs have been place under `./data/MDP50_i.txt`. Three scripts, `run_hpi.py`, `run_bspi.py`, `run_rpi.py` were used for the experiments and detailed results are present in `./logs/`.

- **Linear Programming** - Since `pulp` suffers from a precision issue beyond 7 decimal places, the value function was regenerated from the optimal policy. (ref - `https://github.com/coin-or/pulp/issues/147`)
- RPI - A binomial distribution was used with $p = 0.5$.
- BSPI - Leftmost batches were evaluated first. This is equivalent to evaluating the rightmost batches first. The batch having insufficient elements was the rightmost batch.

2 PI Results

2.1 Howard PI

Howard PI always converged in either 1 or 2 iterations. On an average, it took 1.67 ± 0.47 iterations.

2.2 Randomized PI

10 different random seeds were used while evaluating the 100 MDPs. There was a great variation across seeds. The following table shows the variation in the number of iterations across seeds.

Seed	Mean	Std
0	4.98	0.92
1	5.37	1.19
2	6.87	2.00
3	6.88	1.69
4	7.04	1.43
5	6.96	1.82
6	6.03	1.35
7	7.28	1.68
8	5.14	1.15
9	8.42	2.64
All	6.50	1.95

Table 1: Iterations for Convergence vs Seed across 100 MDP instances

2.3 Batch Switching PI

There was a decrease in the number of iterations taken with an increase in batch size. This decrease was more prominent in the first few incremental steps.

Batch Size	Mean	Std
1	27.73	5.36
2	21.22	3.74
3	16.80	2.88
4	13.89	2.17
5	11.36	2.00
8	8.22	1.57
10	6.37	1.43
15	5.28	1.26
20	4.06	1.04
25	3.09	0.92
50 (hpi)	1.67	0.47
hpi	1.67	0.47
rpi	6.50	1.95

Table 2: Iterations for Convergence vs Batch Size across 100 MDP instances, compared with baselines

2.4 Conclusions

I was quite surprised to see how fast HPI would converge, despite having 50 states in the MDP. However, observing the trend in BSPI, I realized that batchsize $= n$ is equivalent to HPI. Quite clearly for this MDP, HPI is the most suitable choice for rapid convergence. In some sense, this is not very intuitive, since one would not expect a greedy strategy to be optimal. However, since this is a just a 2-action MDP, it seems more likely that a greedy strategy will perform better than a randomised policy.