CS 747 : Foundations of Intelligent Learning Agents
Assignment 2

Arka Sadhu - 140070011

September 5, 2017

1 Linear Programming

The MDP problem is formulated as a linear programming problem. In total there are n variables and nk constraints where n is number of states and k is the number of actions.

1.1 Objective

The objective function is

$$\min \sum_{i=1}^{n} v_i$$

Here v_i is the optimal value function for each state i. These are the n variables

1.2 Constraints

The constraints for the optimal value function is that for all $s \in S$ and $a \in A$, where S is the set of all states, and A is the set of all actions, we have:

$$v_s \geq \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma v_{s'}]$$

Here $T(s, a, s')$ is the transition matrix which gives the probability of transitioning from state s to s' when performing action a. $R(s, a, s')$ is the reward matrix which gives the reward on going from state s to s' on performing action a. γ is a discount factor.

Since the constraint is for all $s \in S$ and for all $a \in A$ there are in total nk constraints.

The objective function, constraints and variables are passed on to PuLP which gives us the optimal value function $V^*(s)$

1.3 Value function to Policy

To get back the best policy we define a function

$$Q^*(s, a) = \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$$

And then get the optimal policy $\pi^*(s)$ as

$$\pi^*(s) = \operatorname*{argmax}_{a \in A} Q^*(s, a)$$

2 MDP Creation

To create an MDP the following was used

- Set total number of states(S) to 50 and total number of actions(A) to 2.
- Create a random reward matrix of size $SxAxS$ with each element in the range $[-1, 1]$.

- Then create a random transition matrix of size $SxAxS$ with each element in the range $[0, 1]$. We note that each eleement of the transition matrix $[s, a, s']$ gives the probability of transition from state s to s' on performing action a. Therefore we would need the sum of elements of the transition matrix from state s and performing action a to be 1. That is the agent can only go to one of the s' states with a finite probability.
- A discount factor γ is created using a random number generator from $[0, 1)$.

3 Comparison between differe Policy Iteration

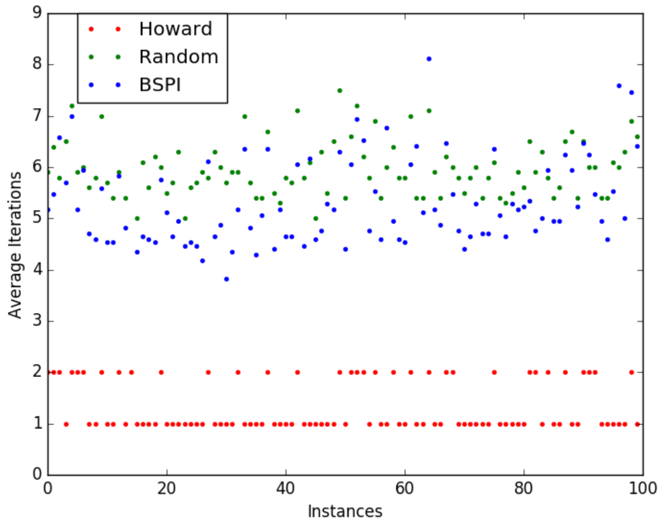

Figure 1: Plot of different Policy Iterations

The plot in 1 is generated in the following manner:

- For Howard PI, directly the number of iterations is plotted.
- For Random PI, random seeds 0-9 are used, and the number of iterations for each of them is noted and then their mean is taken.
- For Batch switch PI, batch sizes in step sizes of 3 were chosen in the range [1, 50] and the number of iterations for each is again noted and their mean is taken.

4 Howard Policy Iteration

For a total of 100 MDP instances with 50 states and 2 action policies.

Table 1: Howard PI

	Howard PI
Mean	1.33
Std deviation	0.47021

5 Random Policy Iteration

For a total of 100 MDP instances with 50 states and 2 action policies.

Table 2: Random PI

	Random PI
Mean	5.977
Std deviation	0.5522

For random seed [0-9]

Table 3: Random PI with seeds

Random Seed	Mean	Std dev
0	5.35	0.841130192063
1	6.04	1.29553077926
2	5.72	1.96509541753
3	4.63	1.04551422755
4	7.39	1.57413468293
5	6.66	1.97595546509
6	6.8	1.75499287748
7	5.13	0.976268405716
8	5.89	1.52901929353
9	6.16	1.32453765518

6 Batch Switching Policy Iteration

For a total of 100 MDP instances with 50 states and 2 action policies.

Table 4: Batch Switch PI

	Batch Switch PI
Mean	5.32764
Std dev	0.86229

For batches in steps of 3 in the range of [1,50]

Table 5: My caption

Batch Size	Mean	Std Dev
1	26.43	4.21011876317
4	13.0	1.8973665961
7	8.23	1.44813673388
10	5.71	1.11619890701
13	4.68	1.00876161703
16	4.3	1.15325625947
19	3.55	0.93139680051
22	3.46	0.89911067172
25	2.53	0.830120473184
28	2.51	0.780960946527
31	2.43	0.681982404465
34	2.46	0.713021738799
37	2.45	0.698212002188
40	2.42	0.602992537267
43	2.41	0.649538297562
46	2.22	0.656962708226
49	1.78	0.742697785105

7 Interesting Observations

- Howard PI didn't take more than 2 iterations on any of the 50 state 2 action MDP. This is quite remarkable since the number of possible policies is 2^{50} and yet it is able to converge extremely fast. This is interesting because the worst case bound of Howard PI is $O(\frac{2^n}{n})$ and it is greater than both of Random PI and Batch Switch PI.

- Random PI and Batch Switch PI both took on an average 5-7 iterations. Interestingly random PI was affected by the random seed quite significantly in a few cases. And in general random PI had a larger std deviation compared to the other two PI.

- Batch Switch PI performs quite bad when the batch sizes are very small. It does much better when batch sizes become larger and the effect on number of iteration seems to be monotonic empirically.

8 Libraries Used

Numpy is used for all matrix operations and sorting methods. PuLP is used for solving linear programming problem and also to get value function for a given policy. Matplotlib is used for generating the plots.