Reinforcement Learning Homework 3

Deadline 5th November 2024
Students Sahiti Chebolu, Surabhi S Nath, Xin Sui

Q1 Recap

(a)

A Markov reward process is a Markov process/chain with values. Like a Markov process, it comprises of a sequence of states and the transition probabilities among the states. It also has the Markov property, which means that, given one state, the transition probability from there to the next state doesn't depend on any proir states. In addition to these properites inherented from being a Markov process, a MRP also has reward values associated with each state as well as a discounting factor for future rewards.

(b)

By fixing a policy (that covers all states).

(c)

Due to the presence of an active agent in the MDP, who interact with the environment and whose actions/policy have an impact on the transition probabilities and the reward function, solving an MDP means finding the optimal policy (which maximizes the value function of the state of reference). The Bellman optimality equation is therefore non-linear (due to the max operatior) and can't be solved in closed form.

Q2

(a)

$$
\begin{aligned}
G_t^{(n)} &= R_{t+1} + \gamma R_{t+2} + ... + \gamma^{n-1} R_{t+n} + \gamma^n G_{t+n}^0 && (1)\\
&= R_{t+1} + \gamma (R_{t+2} + ... + \gamma^{n-2} R_{t+n} + \gamma^{n-1} G_{t+n}^0) && (2)\\
&= R_{t+1} + \gamma G_{t+1}^{(n-1)} && (3)
\end{aligned}
$$

(b)

$$
\begin{aligned}
G_t^{\lambda} &= (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)} && (4)\\
&= (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}(R_{t+1}+\gamma G_{t+1}^{(n-1)}) && (5)\\
&= (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}R_{t+1}+\lambda^{n-1}\gamma G_{t+1}^{(n-1)} && (6)\\
&= R_{t+1}(1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}+\gamma(1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_{t+1}^{(n-1)} && (7)\\
&= R_{t+1}+\gamma(1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_{t+1}^{(n-1)} && (8)\\
&= R_{t+1}+\gamma(1-\lambda)G_{t+1}^{(0)}+\gamma(1-\lambda)\sum_{n=2}^{\infty}\lambda^{n-1}G_{t+1}^{(n-1)} && (9)\\
&= R_{t+1}+\gamma G_{t+1}^{(0)}-\gamma\lambda G_{t+1}^{(0)}+\gamma(1-\lambda)\lambda\sum_{n=1}^{\infty}\lambda^{n-1}G_{t+1}^{(n)} && (10)\\
&= (G_t^{(1)}-\gamma\lambda G_{t+1}^{(0)})+\gamma\lambda G_{t+1}^{\lambda} && (11)
\end{aligned}
$$

Q3

(b)

With default noise = 0.2, the value of the start state (4, 0) becomes non-zero after 1 or 2 policy iterations.
Without noise (n = 0), We need 6 policy iterations for the value of the start state state to become non-zero.

(c)

With noise, we need 1 or 2 policy iterations for convergence depending on the level of noise.
Without noise (n = 0), we need 7 policy iterations for convergence.
This is because with noise, there are non-zero transition probabilities to more states, allowing faster propagation of the values.

(d)

	Value Iteration	**Policy Iteration**
Convergence Speed	Slower convergence due to gradual value updates.	Often converges faster because it optimizes the policy directly.
Computational Cost per Iteration	Less computationally expensive per iteration since it does not require full policy evaluation.	More computationally expensive per iteration due to the policy evaluation step.
Optimal for Large State Spaces	Typically more practical for large state spaces as each iteration is less costly.	Can be computationally intensive for large state spaces due to frequent policy evaluations.
Memory Usage	Often requires less memory as it primarily focuses on value updates.	May require more memory, especially if storing intermediate policies during evaluations.
Suitability	Better for cases where policy stability is less critical or when approximate solutions are acceptable.	Preferred when a stable policy is needed or for exact solutions in smaller state spaces.

Table 1: Comparison of Value Iteration vs. Policy Iteration