CS234 Problem Session Solutions

Week 1: Jan 13

1) **[CA session] Problem 1: MDP Design**

You are in a Las Vegas casino! You have \$20 for this casino venture and will play until you lose it all or as soon as you double your money (i.e., increase your holding to at least \$40). You can choose to play two slot machines: 1) slot machine A costs \$10 to play and will return \$20 with probability 0.05 and \$0 otherwise; and 2) slot machine B costs \$20 to play and will return \$30 with probability 0.01 and \$0 otherwise. Until you are done, you will choose to play machine A or machine B in each turn. In the space below, provide an MDP that captures the above description.

Describe the state space, action space, rewards and transition probabilities. Assume the discount factor $\gamma = 1$. Rewards should yield a higher reward when terminating with \$40 than when terminating with \$0. Also, the reward for terminating with \$40 should be the same regardless of how we got there (and equivalently for \$0).

Solution Example MDP (tables or diagrams containing the same information are equally valid):

States: s_i where $i \in \{0, 10, 20, 30, 40\}$ and s_0 and s_{40} are terminal. Formally, a terminal state can be represented as a state that transitions back into itself and yields reward 0 with probability 1, regardless of the action taken.

Actions: A or B

Rewards: $R(s_i, a, s_j) = j - i$

($R(s, a, s') = 1$ if $s' = s_{40}$ else 0) would also work

Transition probabilities: $P(s_{i-10}|s_i, A) = 0.95$

$P(s_{i+10}|s_i, A) = 0.05$

$P(s_{i-20}|s_i, B) = 0.99$

$P(s_{i+10}|s_i, B) = 0.01$

Where the above transitions are only valid for A in states s_{10}, s_{20}, s_{30} and for B in states s_{20}, s_{30}.

General discussion: An accurate MDP model of the casino scenario would have the following:

States must include all necessary information to judge when we would transition to a terminal state (would have run out of money or reached \$40). This means that states will either directly or indirectly refer to how much money you have left (termination conditions should not rely on additional information from rewards).

Actions could be choosing to use machine A or machine B. Some responses included a special third action to express a loop on terminal states, which is also fine.

There are different ways to specify the reward: we could make the reward the incremental change in our holdings, or we could set a lump-sum reward on transitions to the terminal states.

Transition Probabilities: A transitions to a state with 10 fewer dollars with 95% probability and a state with 10 more dollars with 5% probability. B transitions to a state with 20 fewer dollars with 99% probability and a state with 10 more dollars with 1% probability. Note that these transitions are only defined for states where you have enough money to pay the up-front cost of playing the slot machine. In particular, action B cannot be chosen in a state with only \$10.

2) Problem 2: Contradicting Contractions?

Consider an MDP $M(S, A, P, R, \gamma)$ with 2 states $S = \{S_1, S_2\}$. From each state there are 2 available actions $A = \{stay, go\}$. Choosing *"stay"* from any state leaves you in the same state and gives reward -1. Choosing *"go"* from state S_1 takes you to state S_2 deterministically giving reward -2, while choosing *"go"* from state S_2 ends the episode giving reward 3. Let $\gamma = 1$.

Let $V^*(s)$ be the optimal value function in state s. As you learned in class, value iteration produces iterates $V_1(s), V_2(s), V_3(s), \ldots$ that eventually converge to $V^*(s)$.

(a) [CA Session]

Prove that the ∞-norm distance between the current value function V^k and the optimal value function V^* decreases after each iteration of value iteration.

Solution **Proof:** By definition, since V^* is the optimal value function, we know $BV^* = V^*$. After an iteration of value iteration, we obtain $BV_k = V_{k+1}$. From lecture, we know that the Bellman Backup B is a contraction in ∞-norm, so we know

$||BV_k - BV^*||_\infty \leq ||V_k - V^*||_\infty$.

Thus, we can see that

$||V_{k+1} - V^*||_\infty = ||BV_k - BV^*||_\infty \leq ||V_k - V^*||_\infty$, as required.

(b) [Breakout Rooms]

Now let us consider exactly what forms of convergence are ensured.

For the given MDP, let us initialize value iteration as $V_0 = [0, 0]$. Then $V_1 = [-1, 3]$ and $V_2 = [1, 3]$. We also have $V^* = [1, 3]$.

Is there monotonic improvement in the V estimates for all states? If not, does this contradict the result in Q2(a) and why or why not?

Solution As you run value iteration, it is possible that the error $|V^*(s) - V_k(s)|$ increases in at least one state s while moving from the k-th to the $(k+1)$-th iteration. For example, for state S_1, convergence is clearly not monotonic. The Bellman operator is a contraction. This means that the maximum absolute value across all states of the error (difference of the value function compared to the optimal value function) must not increase between iterations of value iteration. However, for individual states, it is possible that the error increases between iterations. In this example, $||V_0 - V^*||_\infty = 3$, $||V_1 - V^*||_\infty = 2$ and $||V_2 - V^*||_\infty = 0$, so clearly there is no contradiction.

3) **[Breakout Rooms] Problem 3: Stochastic Optimal Policies**

Given an optimal policy that is *stochastic* in an MDP, show that there is always another deterministic policy that has the same (optimal) value.

Solution If a stochastic optimal policy assigns nonzero probability mass to multiple actions in a given state, then those actions must all yield the same expected return; thus, a deterministic optimal policy can be constructed by assigning all probability mass to any single one of these actions, chosen arbitrarily.

4) **[Breakout Rooms] Problem 4: Parallelizing Value Iteration**

During a single iteration of the Value Iteration algorithm, we typically iterate over the states in $\mathcal{S}$ in some order to update $V_t(s)$ to $V_{t+1}(s)$ for all states s. Is it possible to do this iterative process in parallel? Explain why or why not.

Solution The value iteration update for state s is

$$V_{t+1}(s) = \max_{a \in \mathcal{A}} R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V_t(s') \tag{1}$$

Note that $V_{t+1}(s)$ depends only on values in $V_t(\cdot)$, and not on any other entries in V_{t+1}. Therefore, it is possible to parallelize the calculations in Equation 1.