Reinforcement Learning David Silver: Lecture 2 Notes

Name: Eli Andrew

- Reward: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$
  This is a single sample of G_t and therefore contains R rather than $E[R]$.

- Discount factor is useful as a way of dealing with uncertainty in your model. Current rewards not just because they are in the present but because future rewards are more uncertain due to the constraints on our model.

- Discounting also avoids infinte rewards.

- Value function: $v(s)$ gives long-term value of state s. $v(s) = E[G_t | S_t = s]$

- Bellman Equation for MRPs: value function can be decomposed into two parts
  - Immediate reward: R_{t+1}
  - Discounted future reward: $\gamma v(S_{t+1})$
  - Calculation:
    $v(s) = E[G_t | S_t = s]$
    $= E[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s]$
    $= E[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \ldots) | S_t = s]$
    $= E[R_{t+1} + \gamma G_{t+1} | S_t = s]$
    $= E[R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$

- Bellman Equation expressed using matrices: $v = R + \gamma P v$ where v is a column vector with one entry per state.

- Solved directly the Bellman equation solution is $v = (I - \gamma P)^{-1} R$

- Iterative methods for large MDPs: (1) dynamic programming, (2) Monte-carlo simulation, (3) temporal-difference learning

- Markov Decision Process is a Markov Reward Process but with Actions. In other words the reward process is (S, P, R, γ) and the decision process is (S, A, P, R, γ)

- Policy definition: a distribution over actions given states $\pi(a|s) = P[A_t = a | S = s]$

- Because of the Markov property, we do not need to consider R in the policy because s fully characterizes the evolution from this point onwards.

- The policy itself, with the states that it picks, defines a Markov process (S, P^π), and the state and rewards the policy draws defines a Markov reward process, where:
  $P^\pi_{(s,s')} = \sum_{a \in A} \pi(a|s) P^a_{s,s'}$
  $R^\pi_{(s,s')} = \sum_{a \in A} \pi(a|s) R^a_s$

- State value function $v_\pi(s)$ of an MDP is the expected reward starting at state s and then following policy π: $v_\pi(s) = E_\pi[G_t|S_t = s]$

- Action value function $q_\pi(s,a)$ is the expected return starting from state s, taking action a, and then following policy π. $q_\pi(s,a) = E_\pi[G_t|S_t = s, A_t = a]$

- Decomposed state-value function: $v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$

- Decomposed action-value function: $q_\pi(s,a) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a]$

- Bellman equation for V^π: $v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s,a)$ where the policy is giving us the probability of taking the action a given we're in state s and the action-value function is giving us the value of the the action.

- Bellman equation for Q^π: $q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')$ where we are getting our immediate reward R_s^a for the current state and then the discounted reward over all possible states $s' \in S$ we could end up in when taking action a from state s.

- Then taking these two together we can define:
  $v_\pi(s) = \sum_{a \in A} \pi(a|s)(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s'))$
  where this is representing the value from a particular state s using some policy π by: the sum over all actions in the action space of the probability we take that action using this policy given that we are in state s, multiplied by the reward for doing so, which is given as the current reward R_s^a plus the future discounted reward which is calculated as the sum over all possible states in S of the probability of transitioning from the current state s to the new state s' when we perform action a multiplied by the value of being in the next state s' using policy π.

- We can also define $q_\pi(s,a)$ as $R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s') q_\pi(a', s')$

- Optimal value function $v_*(s) = max_\pi(v_\pi(s))$

- Optimal action value function $q_*(s,a) = max_\pi(q_\pi(s,a))$

- For any MDP, there exists an optimal policy π_* that is greater than or equal to all other policies. This optimal policy also acheives the optimal value function $v_*(s) = v_\pi(s)$ and the optimal action value function $q_*(s,a) = q_\pi(s,a)$.

- Bellman optimality equation for $v_*(s) = max_a(q_*(s,a))$

- Bellman optimality equation for $q_*(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_*(s')$

- Putting these two together we get: $max_a(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_*(s'))$
  with which we are looking ahead 2 steps: (1) at the actions we can take and (2) at the ways the environment may move us.

- Bellman optimality equation is non-linear and has no closed form solution in general.

- Iterative solution methods to the Bellman optimality equation include: (1) value iteration, (2) policy iteration, (3) Q-learning, (4) SARSA

- MDP extensions: (1) infinite and continuous MDPs, (2) partially observable MDPs, (3) un-discounted average reward MDPs