CPSC 422

Practice Midterm Exam Solutions

Solution

The exam covers what we did before the midterm break, covering the following sections of the textbook:

- Agents and control: Chapter 1 and Sections 2.1-2.3
- Decision processes: Section 9.5
- Reinforcement Learning: Section 11.3 (except 11.3.7)
- Approximate Inference and time. Section 6.4 and 6.5.

You may bring in one letter sized piece of paper with anything written on it. You may not use calculators, phones, robotic assistants or other electronic aids.

1. (a) A transduction is a function from a percept trace into a command trace. It specifies the action of an agent as a function of what is observed. An agent can only base its action at any time in its previous and current observations (not its future observations), so it can only implement causal transductions which are transductions that only depend on percepts up to the ime the agent is acting.

   (b) An agent does not have access to its history; it can only access what it has remembered. A belief state of an agent encodes what an agent has remembered.

   (c) To implement a causal transduction, an agent needs access to decide what to do based only on what it has remembered and its percepts (this is the command function). It also needs to remember information so it can be sued for future actions. All other functions are just there to make programming easier, and are not strictly needed.

2. $q[34,7] = q[34,7] + \alpha * (3 + \gamma * max_a q[65,a] - q[34,7])$

3. The initial values are not as good estimates as newer values, and so we may not want to weight them as much. It is simpler to ignore the counts (and so keep α fixed). With a fixed α an agent is able to adapt when the environment changes.

4. It gets stuck in non-optimal policies because it does not explore enough to find the best action from each state. To explore, it can pick random actions occasionally. It could also set the initial values high, so that unexplored regions look good.

5. With no discounting the sum of the rewards is often infinite. Discounting means that more recent rewards are more valuable than rewards far in the future.

6. In standard value iteration all of the values are updated from the previous values in a sweep through the values. In asynchronous value iteration, the values are updated from the current value and can be done in any order (it doesn't need to sweep through all of the values). It often works better because the latest values are always used and it can concentrate on updating values where they make the most difference (as it doesn't need to sweep through all of the values each time).

7. Q-learning is like asynchronous value iteration in that it updates the *Q* values, but it uses experience rather than using a model. Thus the average is the average over its experience rather than computing the expected value using a model.

8. There are 15 possible states that could be entered, depending on which direction the robot actually went (up, left or right) and whether the treasure arrived, and where it arrived. Those that have a non-zero immediate reward and/or a future value give:

$$\begin{array}{lll} Q[s13,a2] = & & \\ & 0.8*0.8*(0+0.9*2) & \text{— up, no treasure} \\ + & 0.8*0.2*0.25*(0+0.9*7) & \text{— up, treasure at top right} \\ + & 0.1*0.8*(0.2*-10+0.9*0) & \text{— left, no treasure} \\ + & 0.1*0.2*(0.2*-10+0.9*0) & \text{— left, treasure appears} \\ + & 0.1*0.2*0.25(10+0.9*0) & \text{— right, treasure appears there} \end{array}$$

every other value is 0. Note that $0.2*0.25$ is the probability that a treasure appears at the top right state.

9. (a) $Q(s1, \mathit{right}) = 0$. $Q(s2, \mathit{right}) = -10$. $Q(s3, \mathit{right}) = 0$. $Q(s4, \mathit{right}) = 10$. Each action gets its immediate rewards, as the future value is 0.

   (b) $Q(s1, \mathit{right}) = 0$. $Q(s2, \mathit{right}) = -5$. $Q(s3, \mathit{right}) = 4.5$. $Q(s4, \mathit{right}) = 10$. (Assuming that $s5$ does not have a positive q-value). State $s3$ has no immediate reward, so its average value is $0.5 * 0.9 * 10$. $s2$ gets its average reward, which is -5. Note that $s1$ works differently to $s4$, as it uses the maximum Q-value of $s2$ which is zero.

   (c) In SARSA, $Q(s1, \mathit{right})$ would be -4.5, because it uses $Q(s2, \mathit{right})$, which is -10, rather than $\max_a Q(s2, a)$, which is 0.

10. Reinforcement Learning

   (a) The Q-values need to encode both the immediate reward and the future value. While these do no affect the immediate reward, they are good for modelling the future value, so that other Q-values can have better estimates of the effects of actions.

   (b) Q-learning and SARSA are very wasteful of experiences, so for cases where experience is expensive, we may want a method that allows for useful computation to be carried out between actions, such as the model-based reinforcement learner which collects the relevant statistics about each experience.

   (c) We need $Q[S, A]$ which is of size 4000, $T[S, A, S]$ of size 4000000, and $R[S, A]$ of size 4000.

   (d) Only (ii) and (vii) will find the optimal policy. None of them will follow the optimal policy. Note that (iv) will follow the policy that is optimal out of those that includes 80% exploration.

11. We need $P(S_0)$, the initial state distribution; $P(S_{i+1}|S_i)$ specifying the dynamics; and $P(O_i|S_i)$ specifying the observation model.

    $P(S_0)$ requires 9 parameters, $P(S_{i+1}|S_i)$ requires 90 parameters. $P(O_i|S_i)$ requires 10 parameters. Thus 109 numbers are required to be specified.

12. (a) Variable elimination would maintain a probability distribution over states, and so would not be suitable as there are too many states in a complex environment.

   (b) Rejection sampling would not be suitable as all of the samples will soon be rejected.

   (c) Importance sampling would not be suitable as the weights of all but a few samples would be close to zero (as would soon get to be smaller than representable in a floating point representation, and so would be zero).

(d) Particle filtering would be suitable. It needs to store a set of particles which represent the distribution over the hypotheses of the location of the robot.