CS188 Fall 2013 Section 4: MDPs

1 MDPs: Micro-Blackjack

In micro-blackjack, you repeatedly draw a card (with replacement) that is equally likely to be a 2, 3, or 4. You can either Draw or Stop if the total score of the cards you have drawn is less than 6. Otherwise, you must Stop. When you Stop, your utility is equal to your total score (up to 5), or zero if you get a total of 6 or higher. When you Draw, you receive no utility. There is no discount ($\gamma = 1$).

1. What are the states and the actions for this MDP?

   The state is the current sum of your cards, plus a terminal state:

   $$0, 2, 3, 4, 5, \mathit{Done}$$

   (answers which include 1,6,7,8,9, a *Bust* state, or just "the current sum of your cards plus a terminal state" are acceptable.)

   The actions are $\{\mathit{Draw}, \mathit{Stop}\}$.

2. What is the transition function and the reward function for this MDP?

   The transition function is

   $$T(s, \mathit{Stop}, \mathit{Done}) = 1$$

   $$T(s, \mathit{Draw}, s') = \begin{cases} 1/3 \text{ if } s' - s \in \{2, 3, 4\} \\ 1/3 \text{ if } s = 2 \text{ and } s' = \mathit{Done} \\ 2/3 \text{ if } s = 3 \text{ and } s' = \mathit{Done} \\ 1 \text{ if } s \in \{4, 5\} \text{ and } s' = \mathit{Done} \\ 0 \text{ otherwise} \end{cases}$$

   The reward function is

   $$R(s, \mathit{Stop}, s') = s, s \leq 5$$
   $$R(s, a, s') = 0 \text{ otherwise}$$

3. Give the optimal policy for this MDP.

   In general, for finding the optimal policy for an MDP, we would use some method like value iteration followed by policy extraction. However, in this particular case, it is simple to work out that the opimal policy would be **Draw if $s \leq 2$, Stop otherwise.**

   For completeness, we give below the value iteration steps based on the states and transition functions described above. The optimal policy is given by taking the *argmax* instead of *max*, in the final iteration of value iteration.

V	0	2	3	4	5	Done
V_0	0	0	0	0	0	0
V_1	0	2	3	4	5	0
V_2	3	3	3	4	5	0
V_3	10/3	3	3	4	5	0
Policy Extraction	$10/3_{\mathit{Draw}}$	3_{Draw}	3_{Stop}	4_{Stop}	5_{Stop}	0_{Stop}

4. What is the smallest number of rounds (k) of value iteration for which this MDP will have its exact values (if value iteration will never converge exactly, state so).

   3

2 Pursuit Evasion

Pacman is trapped in the following 2 by 2 maze with a hungry ghost (the horror)! When it is his turn to move, Pacman must move one step horizontally or vertically to a neighboring square. When it is the ghost's turn, he must also move one step horizontally or vertically. The ghost and Pacman alternate moves. After every move (by either the ghost or Pacman) if Pacman and the ghost occupy the same square, Pacman is eaten and receives utility -100. Otherwise, he receives a utility of 1. The ghost attempts to minimize the utility that Pacman receives. **Assume the ghost makes the first move**.

For example, with a discount factor of $\gamma = 1.0$, if the ghost moves down, then Pacman moves left, Pacman earns a reward of 1 after the ghost's move and -100 after his move for a total utility of -99.

Note that this game is not guaranteed to terminate.

1. Assume a discount factor $\gamma = 0.5$, where the discount factor is applied once every time either Pacman or the ghost moves. What is the minimax value of the truncated game after 2 ghost moves and 2 Pacman moves? (Hint: you should not need to build the minimax tree)

   $$1 + 0.5 + 0.25 + 0.125 = 1.875$$

2. Assume a discount factor $\gamma = 0.5$. What is the minimax value of the complete (infinite) game? (Hint: you should not need to build the minimax tree)

   2

3. Why is value iteration superior to minimax for solving this game? Value iteration takes advantage of repeated states to efficiently solve for the optimal policy. Even a truncated minimax tree increases in size exponentially as you increase the search depth.

4. This game is similar to an MDP because rewards are earned at every timestep. However, it is also an adversarial game involving decisions by two agents.

   Let s be the state (e.g. the position of Pacman and the ghost), and let $A_P(s)$ be the space of actions available to Pacman in state s (and similarly let $A_G(s)$ be the space of actions available to the ghost). Let $N(s, a) = s'$ denote the successor function (given a starting state s, this function returns the state s' which results after taking action a). Finally, let $R(s)$ denote the utility received after moving to state s.

   Write down an expression for $P^*(s)$, the value of the game to Pacman as a function of the current state s (analogous to the Bellman equations). Use a discount factor of $\gamma = 1.0$. Hint: your answer should include $P^*(s)$ on the right hand side.

   $P^*(s) =$

   $$P^*(s) = \max_{a \in A_P(s)} R(N(s,a)) + \min_{a' \in A_G(N(s,a))} R(N(N(s,a),a')) + P^*(N(N(s,a),a'))$$