CentraleSupélec

2nd Year
Elective Course

2022 – 2023

Artificial Intelligence – Tutorial Session 4
TS 4: Reinforcement Learning

1 MDPs: short questions

(a) If the only difference between two Markov Decision Process is the value of the discount factor γ, then they must have the same optimal policy. True of false ? Justify.

(b) If we use a feature-based representation for the Q-function, rather than an exact representation with a table, it is possible that the Q-learning algorithm will not find $Q^\star$, the optimal Q-function? True or false? Justify.

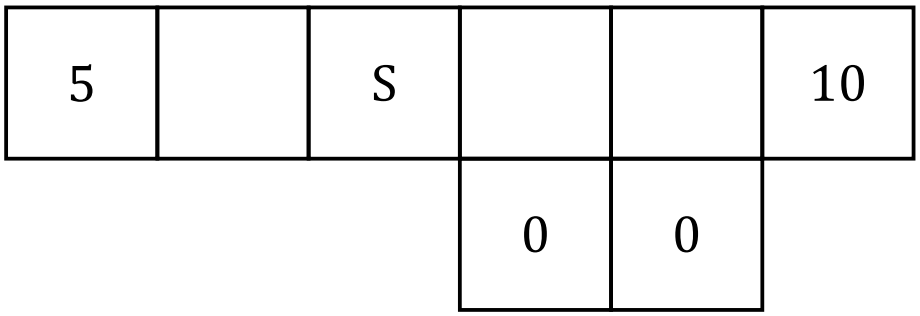
(c) The value iteration algorithm is guaranteed to converge for an MDP with a finite horizon, with finite number of states and actions, and with a discount factor γ such that $0 < \gamma < 1$. True of false?

(d) Recall that for a deterministic policy π where $\pi(s)$ is the action to be taken in state s we have that the value of the policy satisfies the following equations:

$$V^\pi(s) = \sum_{s'} T(s, \pi(s), s')(R(s, \pi(s), s') + \gamma V^\pi(s')).$$

Now assume we have a stochastic policy pi where $\pi(s, a) = P(a \mid s)$ is equal to the probability of taking action a when in state s. Write the equivalent of the above equation for the value of this stochastic policy.

2 Discount MDPs



Consider the above gridworld. An agent is currently on grid cell S, and would like to collect the rewards that lie on both sides of it. If the agent is on a numbered square, its only available action is to Exit, and when it exits it gets reward equal to the number on the square. On any other (non-numbered) square, its available actions are to move East and West. Note that North and South are never available actions.

If the agent is in a square with an adjacent square downward, it does not always move successfully: when the agent is in one of these squares and takes a move action, it will only succeed with probability p. With probability $1 - p$, the move action will fail and the agent will instead move downwards. If the agent is not in a square with an adjacent space below, it will always move successfully.

For parts (a) and (b), we are using discount factor $\gamma \in [0, 1]$.

(a) Consider the policy π_{East}, which is to always move East (right) when possible, and to Exit when that is the only available action. For each non-numbered state x in the diagram below, fill in $V^{\pi_{\text{East}}}(x)$ in terms of γ and p.

(b) Consider the policy π_{West}, which is to always move West (left) when possible, and to Exit when that is the only available action. For each non-numbered state x in the diagram below, fill in $V^{\pi_{\text{West}}}(x)$ in terms of γ and p.

(c) For what range of values of p in terms of γ is it optimal for the agent to go West (left) from the start state (S)?

(d) For what range of values of p in terms of γ is π_{West} the optimal policy?

(e) For what range of values of p in terms of γ is π_{East} the optimal policy?

Recall that in approximate Q-learning, the Q-value is a weighted sum of features: $Q(s,a) = \sum_i w_i f_i(s,a)$. To derive a weight update equation, we first defined the loss function $L_2 = \frac{1}{2}\left(y - \sum_k w_k f_k(x)\right)^2$ and found $dL_2/dw_m = -\left(y - \sum_k w_k f_k(x)\right) f_m(x)$. Our label y in this set up is $r + \gamma \max_a Q(s', a')$. Putting this all together, we derived the gradient descent update rule for w_m as $w_m \leftarrow w_m + \alpha\left(r + \gamma \max_a Q(s', a') - Q(s,a)\right) f_m(s,a)$.

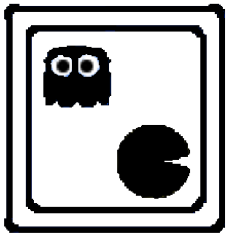
In the following question, you will derive the gradient descent update rule for w_m using a different loss function:

$$L_1 = \left| y - \sum_k w_k f_k(x) \right|$$

(f) Find dL_1/dw_m. Show work to have a chance at receiving partial credit. Ignore the non-differentiable point.

(g) Write the gradient descent update rule for w_m, using the L_1 loss function.

3 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.

(a) 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)

(b) 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)

(c) Why is value iteration superior to minimax for solving this game?

(d) 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^* on the right hand side.