Reinforcement Learning Assignment

Simplified Blackjack / Easy21

Adapted from Easy21 Assignment, David Silver, UCL

Easy21 Assignment Description

The goal of this assignment is to apply reinforcement learning methods to a simple card game that we call Easy21. This exercise is similar to the Blackjack example in Sutton and Barto (Section 5.3), please note, however, that the rules of the card game are different and non-standard.

- The game is played with an infinite deck of cards (i.e., cards are sampled with replacement).
- Each draw from the deck results in a value between 1 and 10 (uniformly distributed) with a colour of red (probability $1/3$) or black (probability $2/3$).
- There are no aces or picture (face) cards in this game.
- At the start of the game both the player and the dealer draw one black card (fully observed).
- Each turn the player may either *stick* or *hit*.
- If the player hits then he/she draws another card from the deck.
- If the player sticks he/she receives no further cards.
- The values of the player's cards are added (black cards) or subtracted (red cards).
- If the player's sum exceeds 21, or becomes less than 1, then the player "goes bust" and loses the game (reward -1).
- If the player sticks then the dealer starts taking turns. The dealer always sticks on any sum of 17 or greater, and hits otherwise.
- If the dealer goes bust, then the player wins; otherwise, the outcome: win (reward $+1$), lose (reward -1), or draw (reward 0) is determined by comparing who has the largest sum.

Submission

- You should submit a single PDF document containing your plots and discussion, and a single archive containing all your source code (Matlab or Python).
- Please organize your source code so that it is easy to follow and apparent how to run your solutions to the assignment's questions, e.g., by naming the relevant files and/or functions `question2`, `question3`, etc.
- Submit your files on Moodle, by the agreed deadline.
- This assignment should be completed individually. A group project will be announced later.
- Any questions or clarifications should preferably be discussed on the Moodle forum or by E-mail.

Implementation of Easy21

You should write an environment that implements the game **Easy21**. Specifically, write a function, named `step`, which takes as input a state s (dealer's first card 1–10 and the player's sum 1–21), and an action a (hit or stick), and returns a sample of the next state s' (which may be terminal if the game is finished) and reward r.

We will be using this environment for model-free reinforcement learning, and you should not explicitly represent the transition matrix for the MDP. There is no discounting ($\gamma = 1$). You should treat the dealer's moves as part of the environment, i.e., calling `step` with a stick action will play out the dealer's cards and return the final reward and terminal state.

Monte-Carlo Control for Easy21

Apply Monte-Carlo control to Easy21. Initialise the value function to zero. Use a time-varying scalar step-size of

$$\alpha_t = \frac{1}{N(s_t, a_t)}$$

and an ϵ-greedy exploration strategy with

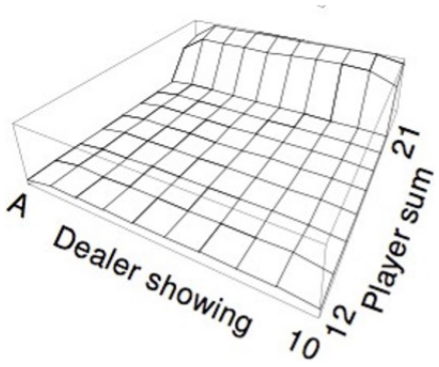
$$\epsilon_t = \frac{N_0}{N_0 + N(s_t)},$$

where $N_0 = 100$ is a constant, $N(s, a)$ is the number of times that state s has been visited, and $N(s, a)$ is the number of times that action a has been selected from state s. Feel free to choose an alternative value for N_0 if it helps produce better results.

Plot the optimal value function

$$V^*(s) = \max_a Q^*(s, a)$$

using similar axes to the figure taken from Sutton and Barto's Blackjack example.



TD Learning in Easy21

Implement Sarsa(λ) in Easy21. Initialize the value function to zero. Use the same step-size and exploration schedules as in the previous section. Run the algorithm with parameter values $\lambda \in \{0, 0.1, 0.2, \ldots, 1\}$. Stop each run after 1000 episodes and report the mean-squared error

$$\sum_{s,a} \left(Q(s, a) - Q^*(s, a)\right)^2$$

over all states s and actions a, comparing the true values $Q^*(s, a)$ computed in the previous section with the estimated values $Q(s, a)$ computed by Sarsa. Plot the mean-squared error against λ. For $\lambda = 0$ and $\lambda = 1$ only, plot the learning curve of mean-squared error against episode number.