Reinforcement Learning Lecture: Homework 13 (Optional)

Ngo Anh Vien

MLR, University of Stuttgart

July 9, 2016

1 Apprenticeship Learning in MountainCar

v You are asked to write an apprenticeship learning program for the Mountaincar domain. Below, we will study a simpler and effective apprenticeship learning algorithm, called the projected max-margin method. You may have to use results from Homework 06.

1.1 Apprenticeship Learning

In our last lecture, we studied one of apprenticeship learning techniques via inverse reinforcement learning. This method is called constraint generation. The main algorithm requires two optimization subroutines at each iteration i: 1) at step 4, you must find w_i (e.g. using quadratic programming, or support vector machine), and 2) at step 5, optimize an optimal policy π_i using the reward function $R_w(s) = w_i^\top \phi(s)$ (e.g. you can use Q-learning, SARSA, etc). We now study another version of that algorithm, called projected max-margin. More specifically, step 4, solving for w_i, is now replaced by the following procedure. At iteration i, the policy set is $\Pi = \{\pi_0, \pi_1, \ldots, \pi_{i-1}\}$ which are associated with a feature vector set $\{\eta(\pi_0), \eta(\pi_1), \ldots, \eta(\pi_{i-1})\}$, respectively. Computing:

- the projected feature vector $\bar{\eta}(\pi_{i-1})$

$$\bar{\eta}(\pi_{i-1}) = \bar{\eta}(\pi_{i-2}) + \frac{\big(\eta(\pi_{i-1}) - \bar{\eta}(\pi_{i-2})\big)^\top \big(\eta(\pi^*) - \bar{\eta}(\pi_{i-2})\big)}{\big(\eta(\pi_{i-1}) - \bar{\eta}(\pi_{i-2})\big)^\top \big(\eta(\pi_{i-1}) - \bar{\eta}(\pi_{i-2})\big)} \big(\eta(\pi_{i-1}) - \bar{\eta}(\pi_{i-2})\big)$$

- $w_i = \eta(\pi^*) - \bar{\eta}(\pi_{i-1})$

1.2 Implementation Note

Some key implementation notes.

- To give demonstration and compute $\eta(\pi^*)$, we use SARSA as done in Homework 06 (the true reward function is assumed to be given) to obtain π^*, then use 50 episodes from π^* to approximate $\eta(\pi^*)$ as in slide 07:13.
- Using linear function approximation with Radial Basis Function features (RBF features) to approximate $R_w(s) = w^\top \phi(s)$, with the same setting as ones in Homework 06.
- Using SARSA as done in Homework 06 to compute π_i and its associated feature vector $\eta(\pi_i)$ (Step 5). Each policy is learnt within 100 episodes whose maximum length is set to 500.
- For stopping criteria, you can rely on the distance $\|\eta(\pi^*) - \bar{\eta}(\pi_i)\|$ or until $i > 50$.

1.3 Results for Report

At each iteration i, you must optimize an optimal policy π_i and $\eta(\pi_i)$ using SARSA and get an averaged number of steps taken to the goal (measure this when you gather 50 episodes in computing η). Plot the results all i and number of steps taken to the goal of policy π_i to see how the performance gets improved.