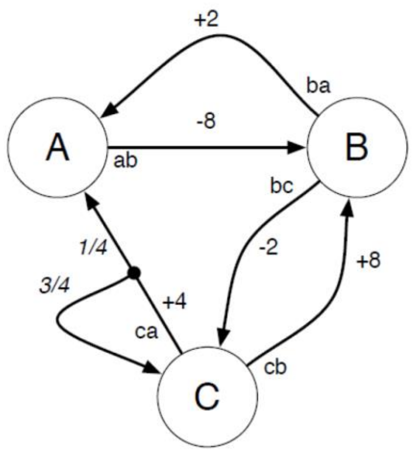
EE 5885 Deep Reinforcement Learning and Control

Homework #2

Due March 13 at 11:59 pm

Problem 1 (35%)

Consider the following Markov Decision Process (MDP) with discount factor γ=0.5. Upper case letters A,B,C represent states; arcs represent state transitions; lower case letters *ab, ba, bc, ca, cb* represent actions; signed integers represent rewards; fractions represent transition probabilities. Consider the *uniform random policy* $\pi(a|s)$ that takes all actions from state s with equal probability. Starting with an initial value function of $V_1(A) = V_1(B) = V_1(C) = 2$, apply *one* iteration of *iterative policy evaluation* (i.e. one backup for each state) to compute a new value function $V_2(s)$.



Problem 2 (40%)

We are now interested in performing linear function approximation in conjunction with a Q-learning algorithm that uses a target network.

In particular, we have a weight vector:

$$w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} \in R^3$$

Given some state s and action $a \in \{-1,0,1\}$, the feature vector of this state-action pair is: $\begin{bmatrix} 2 \cdot s \\ a \\ 0.5 \end{bmatrix}$,

1) Write out the Q value, $q(s, a; w)$, of a state-action pair as a function of the feature vector and the weight vector. (10%)

2) After a single sample (s, a, r, s') is collected, define the loss function that is minimized to update the weights (assuming the weight vector for the target network is represented by w^-). (10%)

3) Suppose we currently have weight vectors (again w^- represents the weight vector for the target network):

$$w = \begin{bmatrix} -2 \\ 1 \\ -1 \end{bmatrix}, w^- = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix},$$

and we collected a sample $((s = 1, a = 0, r = 2, s' = 2)$. Perform a *single* gradient update to the parameter vector w using this sample. Use the learning rate α= 0.2. Write out the gradient, $\nabla_w J(w)$, as well as the new parameters after the update. Show all steps. (20%)

Problem 3 (25%)

The diagram below describes an actor-critic RL architecture. The actor represents the policy $\pi_\theta(a|s)$, parameterized by θ, and the critic represents the action-state value function $Q_w(s, a)$, parameterized by w. The arrows pointing into the actor and critic denote network updates using TD error. If we will implement an *online* actor-critic algorithm, provide the update equations for the actor and critic networks, and use pseudocode to describe the learning algorithm.