CMPUT 366 Reading-Writing Exercise 10

Name: Minghan Li
Student ID: 1561234
CCID: minghan4

We have already learn some basic tools of Reinforcement Learning in the first 11 Chapters, including prediction and control algorithms, function approximation and planning. The next question we will naturally ask is "How can we learn more efficiently?". Luckily, researchers have already come up with several answers to this question, and *Temporal Abstraction* as well as *Eligibility Trace* are two of those solutions.

Temporal abstraction refers to learning temporally abstracted knowledge, i.e. values, from primitive actions, states and rewards over a period of time. Several attempts have been made including *Feudal Reinforcement Learning* and *Options*. However, the former seems to be too specialized in its structure and we want a more general framework to solve this problem, which naturally brings us to the option framework. Options are temporally extended course of actions, which consists three components: a policy π, a termination condition β(which turns out to be equivalent to $1-\gamma$), and an initiation set I. If the option is taken, then actions are selected according to π until the option terminates stochastically according to β. In this way, the agent can not only takes big steps but can select primitive actions in the middle of an option. The foundation theory of options is provided by the existing theory of SMDPs and associated learning methods. With this solid foundation, options can not only speed up the learning process by extended actions but also can provide us with extended perceptions. However, in my personal opinion, one drawback of this framework is that we have to specify the options beforehand. It will be very tempting to let the agent learn the options while its learning the state value, for in some realistic problems we humans don't know what kind of options would be useful.

Temporal abstraction emphasizes on using temporally abstracted knowledge from extended actions to speed up the learnig process, while eligibility traces take a different route to improve learning efficiency. The concept of eligibility traces breaks up into two parts: Forward view and Backward view, where forward view provides the theory and the backward view provides the mechanism. Recall the idea of n-step TD method where the agent use the future n-step return to update its current state value, in the theory of forward view we instead use the weighted sum of all the n-step return as the target, which is known as λ-return, and the algorithm for prediction with it is called TD(λ). However, using λ-return requires accessing to the future, which is impossible. To implement TD(λ), we have to use the backward view, where we use current sum of n-step temporal errors and backpropagate it to all the previous states. This is called offline TD(λ) because we can only update the states at the end of the episodes. We can also implement the online TD(λ) using eligibility traces, but it will only be approximation to the offline TD(λ). TD(λ) algorithms is superior to n-step TD in many aspects: the performance is generally better in similar parameter settings, and it only takes twice as much computational time as TD(0). Moreover, we can combine eligibility trace with classic control methods such as SARSA and function approximation.

One last thing worth mentioning is that there are many different traces in the big trace family, although some of them are not necessarily corresponding to λ-return in the forward view, but they still have similar properties and are able to ahcieve good performance in different tasks. One thought question one would naturally propose at the end is, can we combine eligibility traces and options together with function approximation for control problem? It so, what kind of new problems would arise and we should pay attention to?