Delayed Rewards
[Assignment-2]
(CS 394R: Reinforcement Learning)

Shobhit Chaurasia
Department of Computer Science

September 19, 2016

---

1 Motivation

Inspired by the challenges discussed in the student presentation on lossy reward channels, I was interested in the problem of dealing with delayed rewards in a reinforcement learning setting, often referred to as the "credit assignment problem for learning systems" [1].

2 Problem Statement

The goal is to study a reinforcement learning setting where the rewards perceived by an agent might be occasionally delayed, i.e. reward experienced at time $t+1$ might correspond to a transition $(S_{t'}, A_{t'}) \rightarrow S_{t'+1}$ for some $t' \leq t$, and not necessarily $t' = t$.

3 Modified TD(0) Prediction

Since a comprehensive discussion of the setting of delayed rewards in missing from [2], instead of doing a literature survey on this topic, I set out to tweak the TD(0) prediction to incorporate the notion of reward sharing - a fraction of the reward received at the end of the most recent transition is distributed among the transitions in the past. The tweaked prediction method is referred to as TD(0)-S (for TD(0) with reward **S**haring).

For simplicity of exposition and experimentation, the reinforcement learning setting considered here consists of rewards that can be delayed by at most one time step, i.e., reward R_{t+2} received after the transition $(S_{t+1}, A_{t+1}) \rightarrow S_{t+2}$ can correspond truly either to this transition itself, or the previous transition $(S_t, A_t) \rightarrow S_{t+1}$.

First, the update rules for the TD(0)-S in the simplified setting are discussed. This is followed by a performance comparison to TD(0) prediction. Finally, ideas to generalize the notion of reward sharing to varying degrees of delayed rewards are discussed.

4 TD(0)-S Prediction

Let R'_{t+1} denote the reward received after the transition $\mathcal{T} \doteq (S_t, A_t) \to S_{t+1}$. Since this reward could either correspond truly to $\mathcal{T}$, or to the previous transition $\mathcal{T}' \doteq (S_{t-1}, A_{t-1}) \to S_t$, the state-value update for S_t will use a fraction $(1-\beta)$ of R'_{t+1}, while the remaining fraction β will be added to the state-value estimate for the previous state S_{t-1}. $\beta \in [0, 1]$ controls the degree of reward sharing. The state-value updates at timestep $t+1$ are defined as:

$$\begin{aligned} V_{t+1}(s_t) &\leftarrow V_t(s_t) + \alpha\left[(1-\beta)R'_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)\right] \\ V_{t+1}(s_{t-1}) &\leftarrow V_t(s_{t-1}) + \alpha\beta R'_{t+1} \end{aligned}$$

To get an intuition of how this two step update compares against the update of TD(0), let's write down the update equations for timestep $t = 1$ and $t = 2$, with s_0 as the starting state, reward ben delayed by one timestep, and the following transition sequence. R'_t denotes the reward actually received at timestep t, while R_t denotes the reward intended to be given at timestep t (but, perhaps, was delayed and actually received at $t+1$).

$$\begin{aligned} (s_0, a_0) &\xrightarrow{0} s_1 \\ (s_1, a_1) &\xrightarrow{R'_2 = R_1} s_2 \\ (s_2, a_2) &\xrightarrow{R'_3 = R_2} s_3 \end{aligned}$$

where the numbers above the arrows represent the reward received. R'_2 is the reward received at $t = 2$, but is actually the reward intended to be given after first transition, i.e. R_1.

TD(0) updates at $t = 1$ and $t = 2$ for no-delay case are:

$$\begin{aligned} V_1^{true}(s_0) &\leftarrow V_0(s_0) + \alpha\left[R_1 + \gamma V_0(s_1) - V_0(s_0)\right] \\ V_2^{true}(s_1) &\leftarrow V_1^{true}(s_1) + \alpha\left[R_2 + \gamma V_1^{true}(s_2) - V_1^{true}(s_1)\right] \\ &= V_0(s_1) + \alpha\left[R_2 + \gamma V_0(s_2) - V_0(s_1)\right] \end{aligned}$$

TD(0)-S updates at $t = 1$ and $t = 2$ for one timestep delay are:

$$\begin{aligned} V_1^{tds}(s_0) &\leftarrow V_0^{tds}(s_0) + \alpha\left[(1-\beta)0 + \gamma V_0^{tds}(s_1) - V_0^{tds}(s_0)\right] + \alpha\beta R'_2 \\ &= V_0(s_0) + \alpha\left[\gamma V_0(s_1) - V_0(s_0)\right] + \alpha\beta R_1 \\ &= V_1^{true}(s_0) - \alpha(1-\beta)R_1 \\ V_2^{tds}(s_1) &\leftarrow V_1^{tds}(s_1) + \alpha\left[(1-\beta)R'_2 + \gamma V_1^{tds}(s_2) - V_1^{tds}(s_1)\right] + \alpha\beta R'_3 \\ &= V_0^{tds}(s_1) + \alpha\left[(1-\beta)R_1 + \gamma V_0^{tds}(s_2) - V_0^{tds}(s_1)\right] + \alpha\beta R_2 \\ &= V_0(s_1) + \alpha\left[(1-\beta)R_1 + \gamma V_0(s_2) - V_0(s_1)\right] + \alpha\beta R_2 \\ &= V_2^{true}(s_1) - \alpha R_2 + \alpha(1-\beta)R_1 + \alpha\beta R_2 \\ &= V_2^{true}(s_1) - \alpha(1-\beta)(R_1 + R_2) \end{aligned}$$

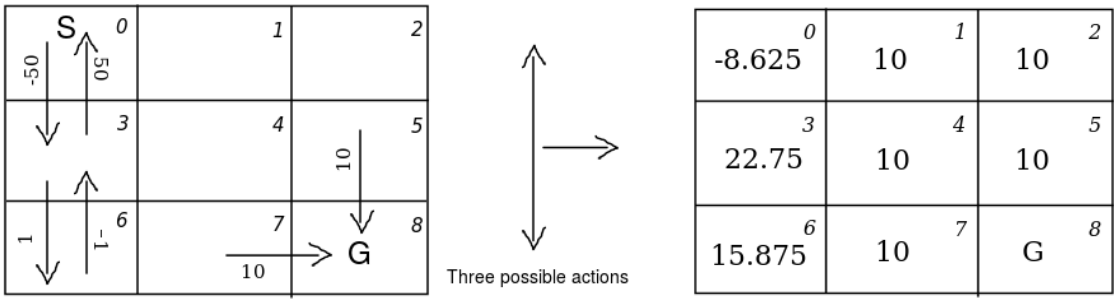
Hence, if there is a deterministic delay of one timestep, then TD(0)-S updates underestimate the state-value estimates. However, this estimate is closer to the true estimate than that of traditional TD(0) on the same task, which can be seen by setting β to 0 in the above equations. In a task with stochastic delay of zero or one time step, β controls the compromise between assigning the credit to either the current state, or to the previous one.

5 Experiments

5.1 Setup

The environment is the gridworld shown in Fig. 1(a). Transitions other than the ones shown incur zero reward. The start and goal states are marked with S and G respectively. States are indexed from 0 to 9. In response to agent's action in a state, the environment gives a reward to the agent. With probability p, the reward given corresponds to the previous transition, and with probability $1 - p$, reward corresponding to the current transition is given. The only exception is the transition to the Goal state, when the reward is given instantaneously. Actions that can take the agent out of the gridworld have the effect of landing the agent in the same state, with a reward of zero. Note that the rewards are setup in such a way to avoid non-zero cycles, and to test the resilience of TD(0) and TD(0)-S to rewards delayed by one timestep (especially in states 0, 3, and 6).

The policy π to be evaluated is the equiprobable random policy starting in state S. The true state-values for this policy are shown in Fig. 1(b)

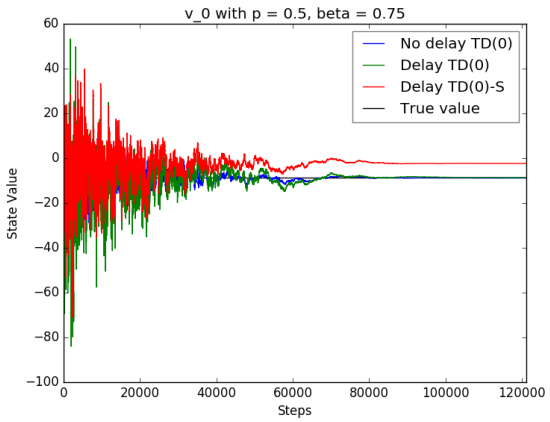


(a) Gridworld (b) State-values

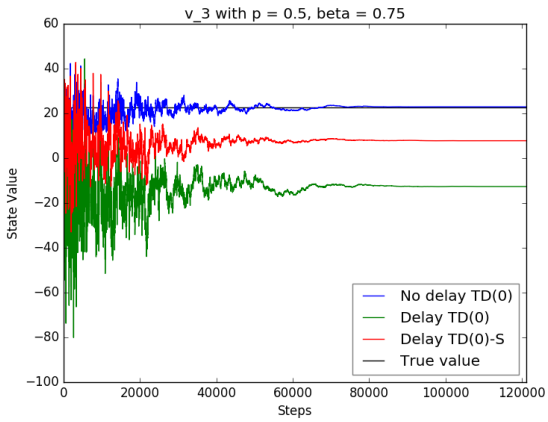
Figure 1: Gridworld with actions, rewards, and true state-values for an equiprobable random policy

5.2 Experiment - 1

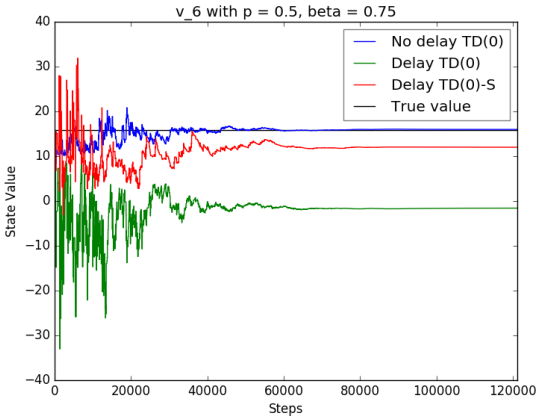
This experiment compares the state-value estimates of TD(0) and TD(0)-S on an un-discounted ($\gamma = 1$) the task with a probabilistic delay of $p = 0.5$. β was set to 0.75; the learning rate α initialized to 0.5, decayed by a factor of 0.99 after each episode.
Fig. 2 shows the state-value estimates for the two algorithms. Progress of TD(0) on a task without any delay is also shown for reference. Estimates are shown only for states 0, 3, and 6 because the other states are not interesting as far as delayed rewards are concerned; all the algorithms have the exact same progress for these states. The corresponding errors in the value-estimates are shown in Fig. 3
As evident from Fig. 2 and Fig. 3, the value estimates computed by TD(0)-S are closer to the true estimates than those of TD(0) on the task with delayed rewards. The exception is the cell 0. I contend that the reason behind this is the special property of cell 0, which is that it's the start state. Being the start state, an initial transition from S either receives the true reward of -50, or no reward at all (because there are no delayed rewards to deliver). In TD(0), this implies that the



(a) Value estimate for cell 0



(b) Value estimate for cell 3



(c) Value estimate for cell 6

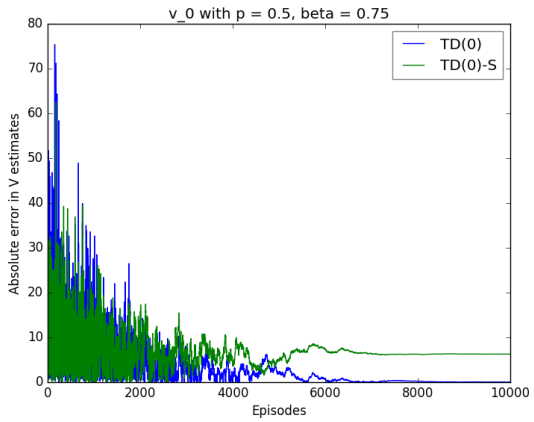
Figure 2: Progress of value estimates for different algorithms. Delay TD(0) and Delay TD(0)-S correspond to the respective algorithms on the delayed reward settings. No delay TD(0) shows TD(0)'s progress on a task without delay. True state-values are also shown for reference.

value estimates are updated with the true reward, albeit fewer number of times (as compared to no-delay case). Thus, eventually, it would converge to the true estimate, although the convergence would be slower than the no-delay case. However, in TD(0)-S, the same case leads the algorithm to update its value estimates based on only a fraction $(1 - \beta)$ of the -50 reward, and hence, it converges to a poorer estimate.

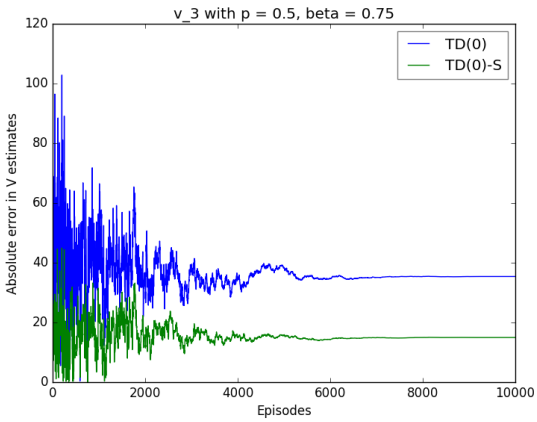
5.3 Experiment - 2

This experiment involves comparing the performance of TD(0)-S in the same delayed reward setting for different values of p and β. Intuitively, lower values of p involve a larger fraction of rewards being delayed, so a higher value of β should be favored to ensure a larger fraction of reward received at time t is awarded to the previous transition. However, this intuition might not hold good for all states, especially in wake of the poor performance of TD(0)-S on certain states (like cell 0).

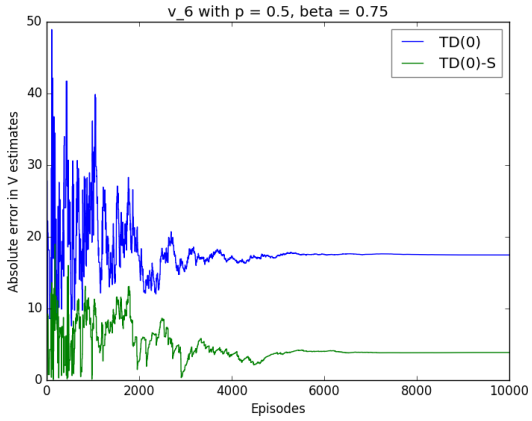
Fig. 4 shows the absolute error in value estimates for the three states after 8000 episodes, averaged



(a) Absolute error in value estimate for cell 0



(b) Absolute error in value estimate for cell 0



(c) Absolute error in value estimate for cell 0

Figure 3: Absolute errors in value estimates for TD(0) and TD(0)-S the delayed reward settings

over 25 repetitions of the experiment. The error in estimates are generally smaller for low values of p with high β. For higher values of p, although the magnitude of error for high values of β increases, the trend is similar to that of lower p values. This counter-intuitive trend is inexplicable. Perhaps, apart from interaction between p and β, the performance of TD(0)-S also depends upon the scale of the values of rewards. Exception for error in estimates of state 0 should be noted. Consistent with the argument presented above for the special case of this state, higher values of β result in larger errors, because the estimates end up being based on increasingly smaller fraction of the true reward -50. $p = 1$ reduces the delayed-reward setting to the no-delay setting, in which case, as expected, $\beta = 0$ works best.

6 Generalized TD(0)-S

The idea of reward sharing can be easily generalized to the setting where the rewards can be arbitrarily delayed. For example, a fraction of the reward $(1 - \beta)$ can be assigned to the current transition, while the remaining is distributed, either equally, or in decreasing amounts, among the previous states. This requires the agent to keep a history of all the state-action pairs it has seen

in the past, and update the value estimates for each one of them at every timestep.

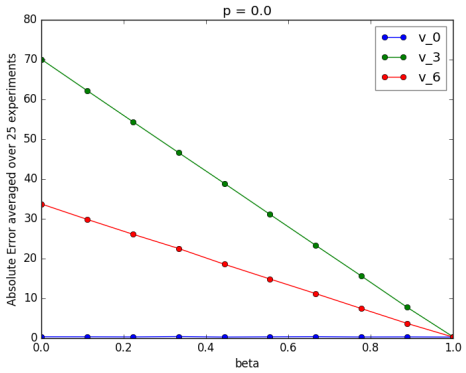
7 Conclusion

A variant of TD(0) prediction method is proposed to allow for distribution of received rewards among previous states. This is useful in reinforcement learning tasks where the agent may experience delayed rewards, leading to the classic "credit assignment problem" [1]. The tweaked prediction method, TD(0)-S, obtains better value estimates in setting where rewards are delayed at most by one timestep. However, the gains are observed to be not consistent across all states. Generalization to settings with arbitrary delays in reward is straightforward.

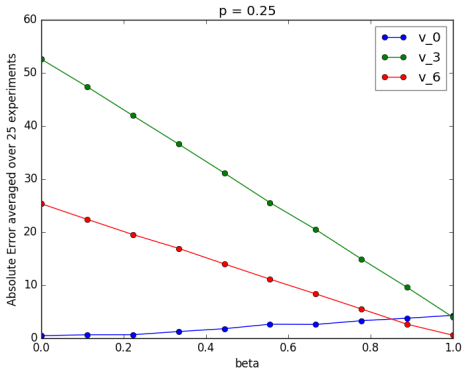
References

[1] Marvin Minsky. "Steps toward artificial intelligence". In: *Proceedings of the IRE* 49.1 (1961), pp. 8–30.

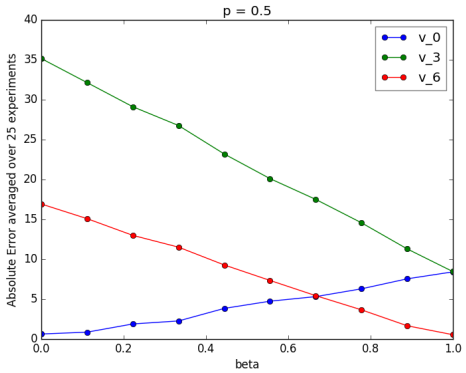
[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Vol. 1. 1. MIT press Cambridge, 1998.



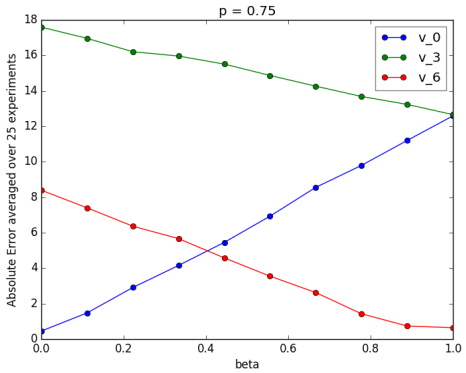
(a) $p = 0.0$



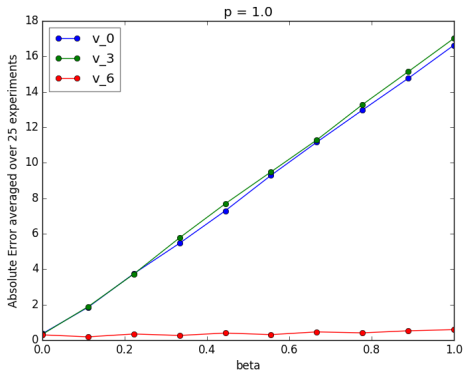
(b) $p = 0.25$



(c) $p = 0.5$



(d) $p = 0.75$



(e) $p = 1.0$

Figure 4: Average absolute error in value estimates for different values of p and β.