Value Driven Representations for Reinforcement Learning - Supplementary Materials

1 Proof of Theorem 5.2

Theorem. *Let M and $\tilde{M}$ be Markov decision processes over the same action space $\mathcal{A}$ and state spaces $\mathcal{S}$, $\tilde{\mathcal{S}}$, respectively. Where $\tilde{\mathcal{S}} = \mathcal{S} \setminus s_0 \cup \{s_0^1, s_0^2\}$ such that s_0^1, s_0^2 are the split of state s_0. Let $\pi^* : \mathcal{S} \to \mathcal{A}$ be the optimal policy in M. Then, $\exists \tilde{\pi} : \tilde{\mathcal{S}} \to \mathcal{A}$ such that $V_{\tilde{M}}^{\tilde{\pi}} = V_M^{\pi}$ where $\tilde{\pi}(s_0^1) = \tilde{\pi}(s_0^2) = \pi(s_0)$, and $\forall \pi' \neq \tilde{\pi} : V_{\tilde{M}}^{\pi'} \leq V_{\tilde{M}}^{\tilde{\pi}}$.*

Proof. Since both M and $\tilde{M}$ are MDP, W.L.G we assume that all polices are deterministic. Setting $\tilde{\pi}(s_0^1) = \tilde{\pi}(s_0^2) = \pi^*(s_0)$ will trivially retrieve the optimal policy of M with the same value $V_{\tilde{M}}^{\tilde{\pi}} = V_M^{\pi}$. Now consider two cases when :

1. $\pi'(s_0^1) = \pi'(s_0^2)$: then π' is also a policy in M with $\pi'(s_0^1) = \pi'(s_0^2) = \pi'(s_0)$, where $V_M^{\pi'} \leq V_M^{\pi^*} = V_{\tilde{M}}^{\tilde{\pi}}$, since π^* is the optimal policy in M.
2. $\pi'(s_0^1) \neq \pi'(s_0^2)$: W.L.G assume $V^{\pi'}(s_0^2) \geq V^{\pi'}(s_0^1)$, we show that the policy can be improved by setting $\pi'(s_0^1)$ equal to $\pi'(s_0^2)$. Then followed by case 1, $V_M^{\pi'} \leq V_M^{\pi^*} = V_{\tilde{M}}^{\tilde{\pi}}$.

$$
\begin{aligned}
&Q^{\pi'}(s_0^1, \pi'(s_0^2)) \\
&= R(s_0^1, \pi'(s_0^2)) + \gamma \sum_{s'} T(s'|s_0^1, \pi'(s_0^2)) V^{\pi'}(s') \\
&= R(s_0^2, \pi'(s_0^2)) + \gamma \sum_{s'} T(s'|s_0^2, \pi'(s_0^2)) V^{\pi'}(s') \\
&= Q^{\pi'}(s_0^2, \pi'(s_0^2)) = V^{\pi'}(s_0^2) \geq V^{\pi'}(s_0^1) = Q^{\pi'}(s_0^1, \pi'(s_0^1))
\end{aligned}
$$

The second equality comes from the fact that in Markovian representation, state is sufficient to determine the transition probabilities and rewards so,

$$
\begin{aligned}
\forall s, a : P(s|s_0^1, a) &= P(s|s_0^2, a) \\
\forall a : R(s_0^1, a) &= R(s_0^2, a)
\end{aligned}
$$

□

2 Pseudo Code for Split Algorithm

In section 4.3, Algorithm 1 is used to determine if a split happens or not for VDR-E.

Algorithm 1 Split($\pi_{old}, graph, B = 10, N = 100$)

1: $v_{old} \leftarrow OPTE(\pi_{old}, graph, N, False)$
2: **for** $o_j \in \mathcal{O}$ **do**
3: $o_j^1, o_j^2 \leftarrow o_j$ // use EM to split observation
4: $\mathcal{O}_j \leftarrow (\mathcal{O} \setminus o_j) \cup (o_j^1, o_j^2)$
5: Construct $graph_j$ for $\mathcal{O}_j$
6: $\pi_j = \max_\pi$ OPTE($\pi, graph_j, N, False$)
7: **for** $b = 1 : B$ **do**
8: Construct $graph_j^b$ for bootstrap b, $\mathcal{O}_j$
9: $V^{\pi_j}(b) =$ OPTE($\pi_j, graph_j^b, N, False$)
10: **end for**
11: $splitScore(j) = \mathbb{1}(\pi_{old} \neq \pi_j) \frac{avg(V^{\pi_j}) - v_{old}}{std(V^{\pi_j})}$
12: **end for**
13: **if** $\max(splitScore) > thresh$ **then**
14: $J \leftarrow argmax_j\ splitScore$
15: $\mathcal{O} \leftarrow \mathcal{O}_J$
16: **end if**

In case of VDR and stochastic policy, line 6 is replaced by a policy optimization method (in our experiments REINFORCE (Williams, 1992)) and line 11 is replaced by

$$D_{KL}(\pi_{old} || \pi_j) \frac{avg(V^{\pi_j}) - v_{old}}{std(V^{\pi_j})}$$

3 Extra Experiments

Riverswim

Riverswim (Osband, Russo, and Van Roy, 2013) is a six state chain MDP used in both PSRL and TCRL showcase. For complete description of the environment, reader may refer to Osband, Russo, and Van Roy (2013). We used episode length of 20 and start with one observation as an initial representation, aliasing all six states. Results in this environment are shown in Figure 1. Our algorithm performs better and is able to learn the optimal policy in relatively few episodes. The optimal

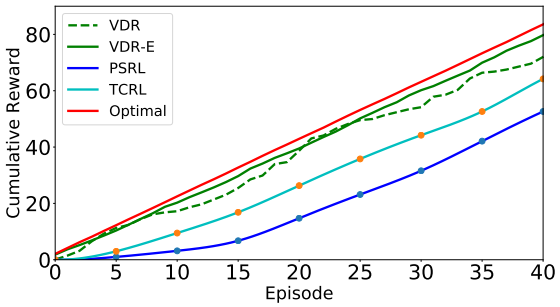

Figure 1: Riverswim. Our algorithm can learn the optimal policy in relatively few episodes. The optimal policy is taking an action toward the goal in all states and is representable using only one observation. (Results of PSRL and TCRL are from Mandel et al. (2016))

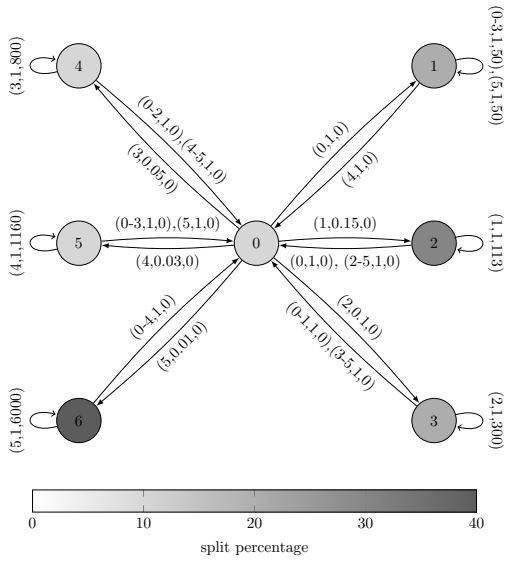

Figure 2: Sixarms' split pattern

policy in this environment is taking an action toward the goal in every state of the true underlying MDP. This simple policy can be represented using only one observation. Our algorithm does not split the initial observation which reflects the fact that the optimal policy can be learned using only one observation.

Split Pattern of Six Arms

Figure 2 shows the split pattern in sixamrs that is described and discussed in section 6.2 of the paper.

Worst Case of Six Arms

VDR can do better than the state-of-the-art Markovian based algorithms when the optimal policy can be represented in a smaller observation space; however, when this assumption is not true our algorithm can still find the optimal policy. We changed the sixarms environments by swapping all actions that yields a reward in

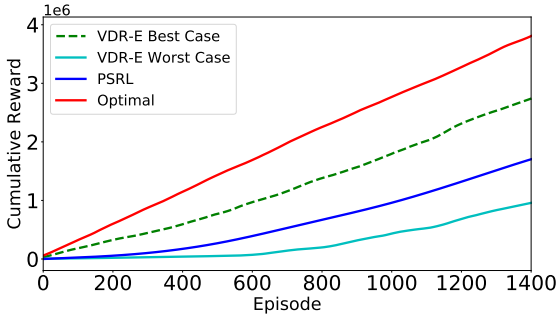

Figure 3: Comparison of the best and the worst case variants of Sixarms

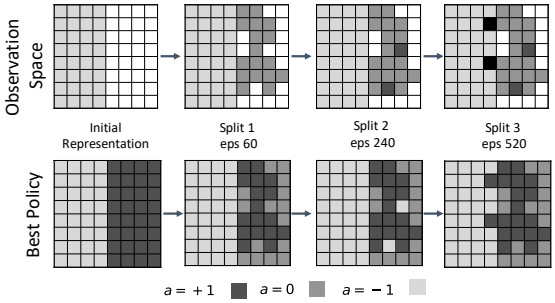

Figure 4: Split sequence of $v_0 = 0.03$ by VDR-E. Above: Observation space representation. Each color represents a distinct observation. Below: Policy found by our algorithm. Each color represents a distinct action

each state (except state 0) to obtain the worst case scenario for VDR, in this case the optimal policy can only be learned if all states are distinct (i.e. Markovian state representation is required). In this situation VDR can still learn the optimal policy but slower than algorithms like PSRL that assumes Markovian representation from the beginning. Figure 3 shows the results of the best and worst case of VDR-E along with PSRL that starts with Markovian representation.

Mountain Car Split Sequence

Figure 4 shows the split sequence for VDR-E along with the optimal policy. The second split is unnecessary to obtain the optimal policy, this is the same phenomenon as in Sixarms environments, where lack of sufficient data suggests a split resulting in a better policy at episode 240; however, running for more episodes algorithm adjusts itself.

References

Mandel, T.; Liu, Y.-E.; Brunskill, E.; and Popovic, Z. 2016. Efficient bayesian clustering for reinforcement learning. In *IJCAI*, 1830–1838.

Osband, I.; Russo, D.; and Van Roy, B. 2013. (more) efficient reinforcement learning via posterior sampling.

In *Advances in Neural Information Processing Systems*, 3003–3011.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.