Revisiting State Augmentation methods for Reinforcement Learning with Stochastic Delays

Somjit Nath, Mayank Baranwal, Harshad Khadilkar

Introduction

- Most real world problems are plagued by delays in learning.
- Most Reinforcement Learning(RL) algorithms fail to learn anything substantial due to the presence of delays.
- Thus handling delay in RL is crucial aspects to enable RL to be used in a variety of real world problems like congestion control, control of robotic systems, distributed computing, medical domains etc.

Delays in Reinforcement Learning

- RL algorithms model the environment as a Markov Decision Process (MDP).
- In the presence of delays, one can equivalently model the underlying problem as partially observable MDP (POMDP).
- POMPDs are generalizations of MDPs, however solving POMDPs without estimating hidden action states leads to arbitrary sub-optimal policies.

Types of Delays

Observation Delay
Delay = 3

$T = t$	$T = t + 1$	$T = t + 2$	$T = t + 3$
s_{t-3}	s_{t-2}	s_{t-1}	s_t
a_t	a_{t+1}	a_{t+2}	a_{t+3}

Action Delay
Delay = 3

$T = t$	$T = t + 1$	$T = t + 2$	$T = t + 3$
s_t	s_{t+1}	s_{t+2}	s_{t+3}
a_{t-3}	a_{t-2}	a_{t-1}	a_t

MDP State Augmentation

- In the presence of delays, it becomes imperative to add the delayed observation with the non-implemented actions to make the states Markov.
- The reformulation allows an MDP with delays to be viewed as an equivalent MDP without delays.
- The resulting Bellman equation with conditional expectation in the equivalent cost structure requires vast computational resources for estimation.

Delay Resolved Algorithms

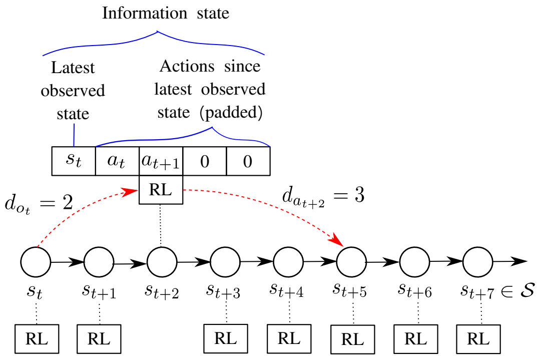


Figure: Graphical illustration of the problem with observation and actions delays.

Delay Resolved Algorithms

- For a delay of d, the time-complexity of computation of the expected cost scales as $O(d|\mathcal{S}|^2)$.
- The size of the "augmented" state grows exponentially with d, i.e., $\mathcal{I} = \mathcal{S} \times \mathcal{A}^d$, and thus tabular learning approaches become computationally prohibitive even for moderately large delays.
- Our algorithm uses neural networks as nonlinear function approximators, the computational complexity scales as $O(|\mathcal{S}| + d|\mathcal{A}|)$.

Prior Work

- Much of the work on delays in RL was analyzed for constant delays. The analysis used augmented states, which results in high computational overhead.

- A memory-less method was also introduced which updated the Q values with the effective action. However for this to work, the exact delay value is needed and this value should be constant.

- Model based approaches were also developed to predict the underlying transition probabilities from the delayed transitions. However, learning these transition dynamics can be tricky.

Recent Advances

- We can also use forward models for predicting the current undelayed state. However, learning forward models are slightly more computationally intensive and can be problematic to learn in complicated environments.
- A recently formalized Real-Time Reinforcement Learning (RTRL), a deep-learning based framework that incorporates the effect of *single-step* action delay.
- Another recent algorithm performs partial trajectory sampling and learning from them. Their algorithm, known as Delay Correcting Actor Critic (DCAC), performs reasonably well, however, at the expense of requiring *complete information of the delay values at each step needed for re-sampling.*

Constant Delay MDP

Definition : CDMDP

A CDMDP, denoted by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma, d_o, d_a \rangle$, augments an MDP with state-space $\mathcal{S} \times \mathcal{A}^{d_o+d_a}$. Note that a policy π is defined as a mapping $\pi : \mathcal{S} \times \mathcal{A}^{d_o+d_a} \rightarrow \mathcal{A}$.

- Information state under observation delay: $I_t = \{s_{t-d}, a_{t-d}, \ldots, a_{t-2}, a_{t-1}\}$
- Information state under action delay: $I_t = \{s_t, a_{t-d}, \ldots, a_{t-2}, a_{t-1}\}$
- CDMDPs with appropriately chosen information states can be treated as equivalent MDP without delays

Stochastic Delay MDP

Definition : SDMDP

A SDMDP, denoted by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma, d_o, d_a, [k], n \rangle$, augments an MDP with state-space $\mathcal{S} \times \mathcal{A}^{d_o+d_a} \times \mathbb{Z}^+$ such that $d_o + d_a \leq n - 1$. A policy π in SDMDP is defined as a mapping $\pi : \mathcal{S} \times \mathcal{A}^{d_o+d_a} \times \mathbb{Z}^+ \to \mathcal{A} \cup \{\emptyset\}$. Here $\emptyset$ represents *no action* and corresponds to the scenario when the MDP *freezes* for the agent.

- Unlike CDMDP, size of information state may vary $\implies$ inclusion of *no action*
- Information state under observation delay: $I_t = \{s_{t-d}, k, a_{t-d}, \ldots, a_{t-1}, \emptyset, \ldots, \emptyset\}$
- Unlike CDMDP, information state also includes the instant k at which the most recent state s_{t-d} was first observed
- Rewards for any unobserved past states are received upon observing the most recent state
- Agent can make decisions at each instant despite stochastic delays

Some Key Theoretical Results

Theorem (Equivalence of SDMDP and undelayed MDP)

The SDMDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma, d_o, 0, [k], n \rangle$ *can be reduced into an equivalent undelayed MDP* $\langle \mathcal{I}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma \rangle$ *with simplified cost structure.*

Lemma (Equivalence of action and observation delays)

The CDMDPs $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma, d, 0 \rangle$ *and* $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma, 0, d \rangle$ *are functionally equivalent to the MDP* $\langle \mathcal{I}, \mathcal{A}, \mathcal{P}_{\mathcal{A}}, r, \gamma \rangle$ *with appropriately chosen information state* $\mathcal{I}$*.*

Environments and Algorithms

Environments: Gym Environments (Acrobot, CartPole, MountainCar) & W-Maze

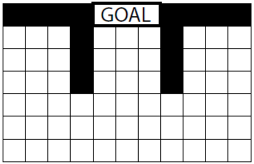
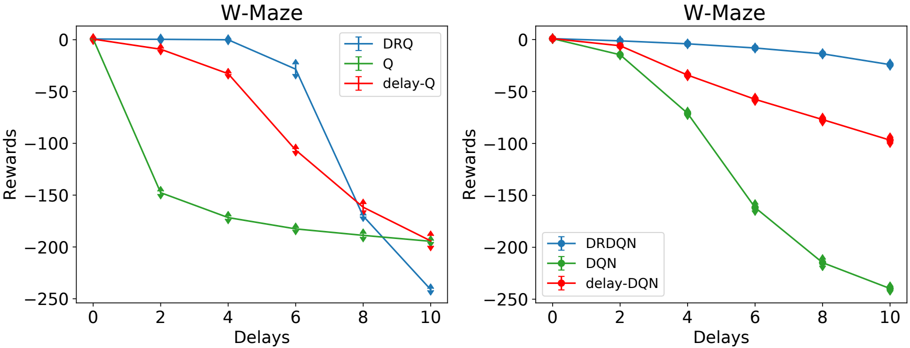


Figure: W-Maze

Algorithms used:

- Q/DQN: Baseline algorithm that does not take any information about delays.
- delay-Q/DQN: Memory-less Algorithm using effective action for learning Q values
- Delayed-DQN: Using Forward Model to predict the current state
- DRQ/DRDQN: Our Algorithm

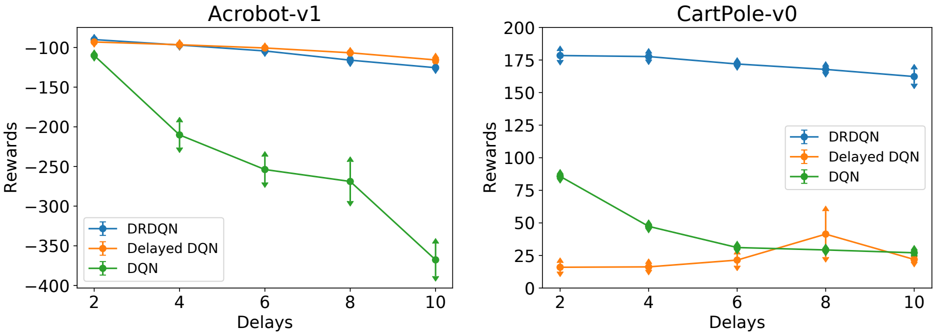
Tabular v/s Function Approximation



(a) (b)

Figure: Using Neural Networks to tackle the problem exploding state space: (a) shows the performance with Tabular Q-learning, whereas (b) uses DQN.

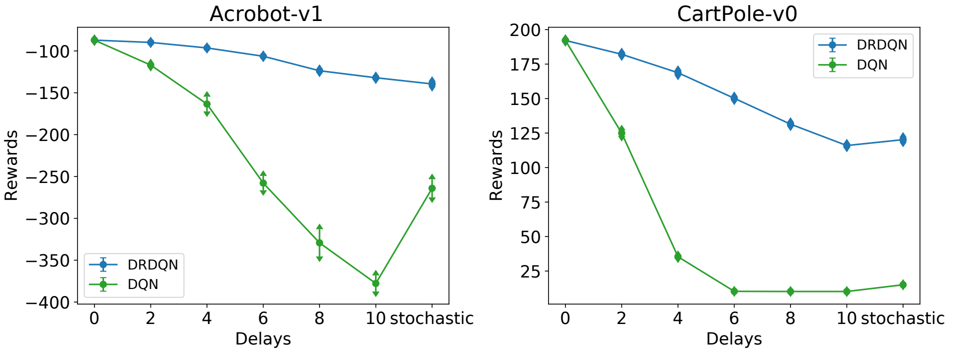
Action Delays



(a) (b)

Figure: Comparison with Baselines on Gym environments for constant action delays.

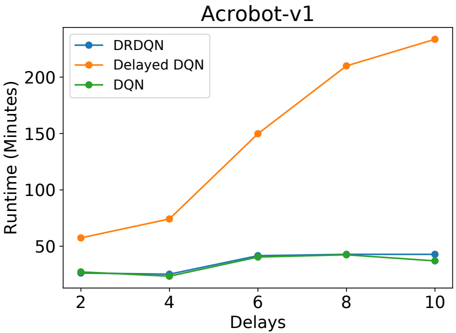
Observation Delays



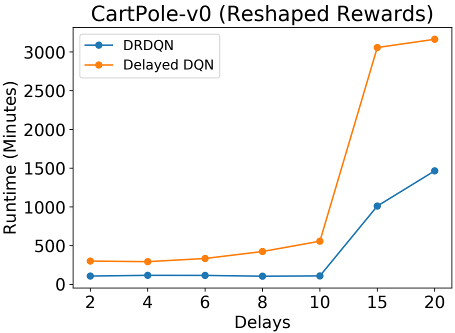
(a) (b)

Figure: Comparison with Baselines on Gym environments for constant and stochastic observation delays. DRDQN learns the optimal policy both for constant and stochastic delays.

Compute Comparison



(a)



(b)

Figure: (a) and (b) show runtimes for DRDQN and Delayed DQN

Equivalence of action and observation delays

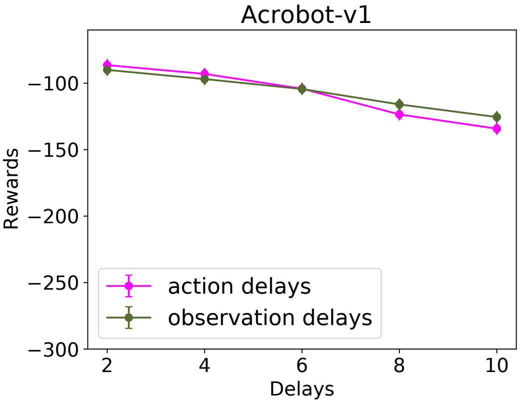


Figure: Equivalence of Action and Observation Delays

Conclusion

- We revisit the idea of using augmented states as a solution for delayed RL problems.
- We formally define both constant and stochastic delays and provide theoretical analysis on why the reformulation still converges to the same optimal policy.
- We adapt state augmentation methods to constant and stochastic delay problems, to formulate a class of algorithms, which can perform well on both constant and random delay tasks.

Thank You!