Multi-Step Bootstrapping Expected SARSA, Tree Backup and $Q(\sigma)$ Algorithms Technical Report

Riashat Islam
McGill University
Reasoning and Learning Lab
riashat.islam@cs.mcgill.ca

February 17, 2017

1 Introduction

In this work, we evaluate experimentally the performance of Expected SARSA, Tree Backup algorithms and the $Q(\sigma)$ algorithm. At first, we will analyse the differences in convergence of these algorithms, and then experimentally demonstrate the differences between them. We will evaluate our algorithms on the simple GridWorld and the CliffWorld MDP environments.

We provide experimental results and analysis of the following algorithms:

- Demonstrate the performance of Expected SARSA on GridWorld and CliffWorld Environments
- Tree Backup Algorithms : In particular, we compare the performance of Two Step and Three Step Tree Backup with Expected SARSA (One-Step Backup)
- Experimentally analyse differences between Expected SARSA and Tree Backup Algorithms with SARSA and Q-Learning
- On-Policy $Q(\sigma)$ and n-step $Q(\sigma)$ algorithms - We evaluate the performance of $Q(\sigma)$ algorithms on Grid World and Cliff World environments
- Compare $Q(\sigma)$ algorithm with Expected SARSA and Tree Backup

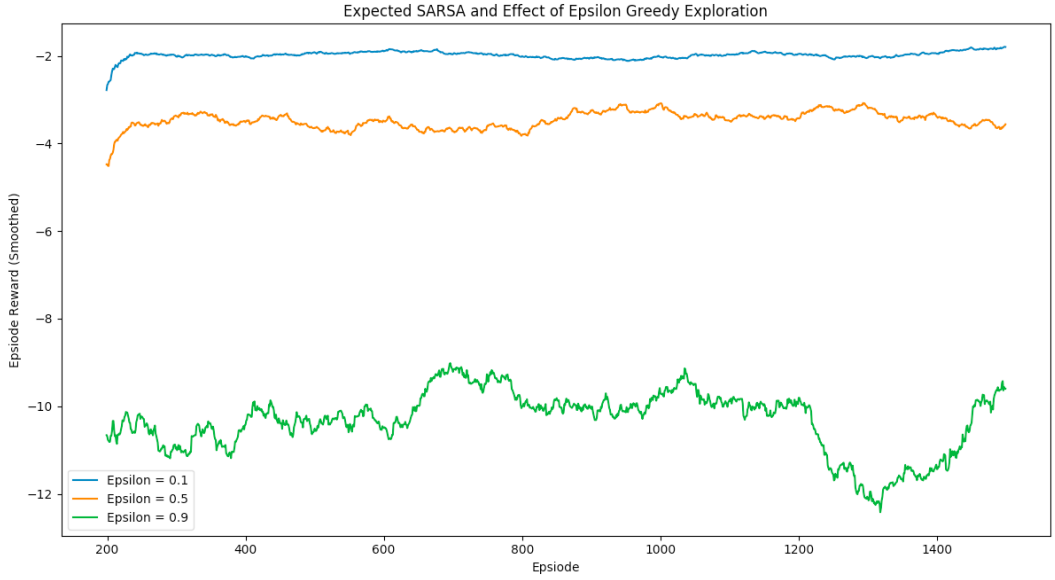
2 Experimental Results

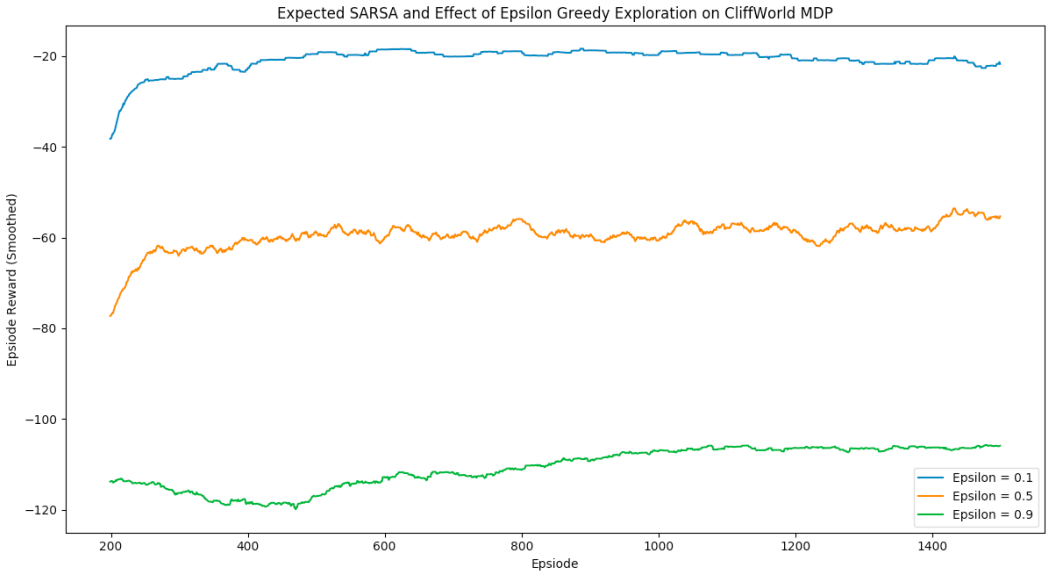
2.1 Expected SARSA

We first demonstrate the performance of Expected SARSA, and it's effect with the ϵ parameter on the GridWorld and CliffWorld MDPs. As expected, since Expected SARSA considers all possible

actions, having a smaller value of ϵ is reasonable, since it encourages less random actions to be action, and more actions with a higher probability to be taken based on greedy maximisation of the Q function.

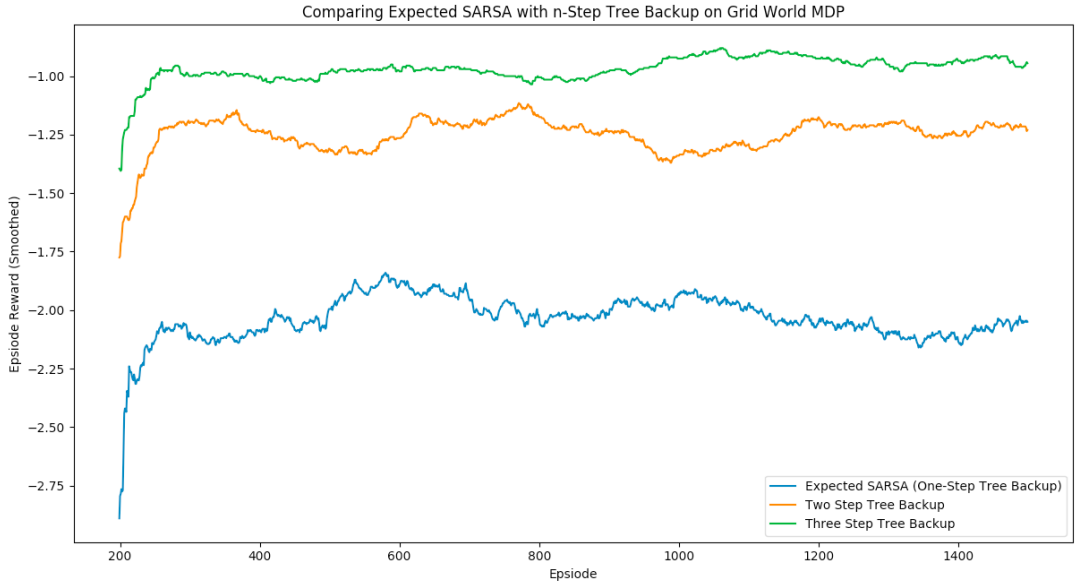
Our conclusion based on the results below is that with smaller value of ϵ, by assigning a lower probability to random actions, the algorithm performs better if actions are taken based on the maximisation of Q function.





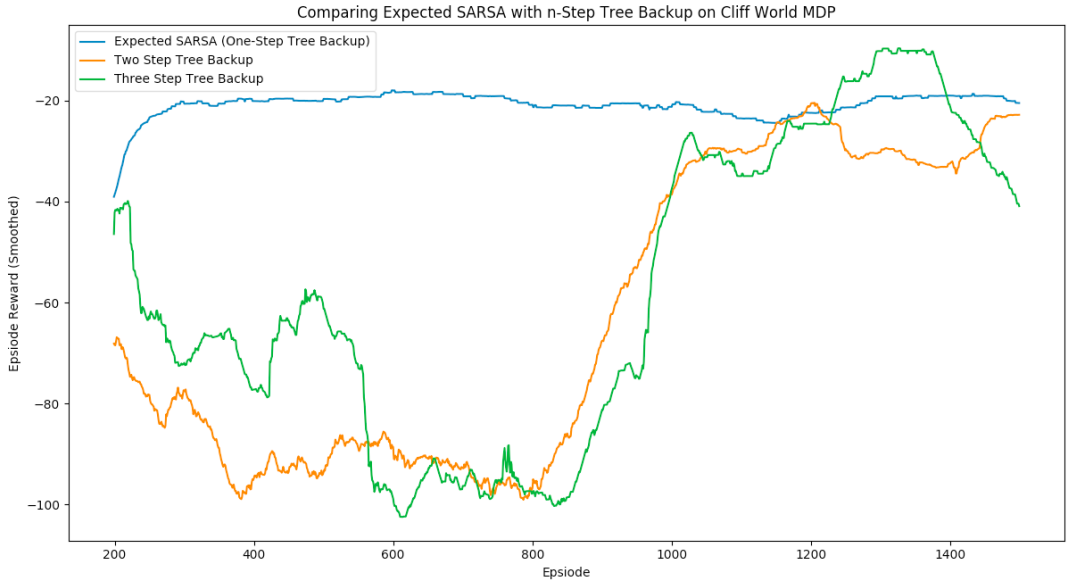
2.2 Tree Backup Algorithms

We first compare the two and three step tree backup algorithms with Expected SARSA on the Grid World environment. Our results show that n-step Tree Backup performs much better than Expected SARSA.



However, when we apply the n-step backup algorithms on a different MDP (CliffWorld in this case), the n-step backup algorithms actually performs much worse.

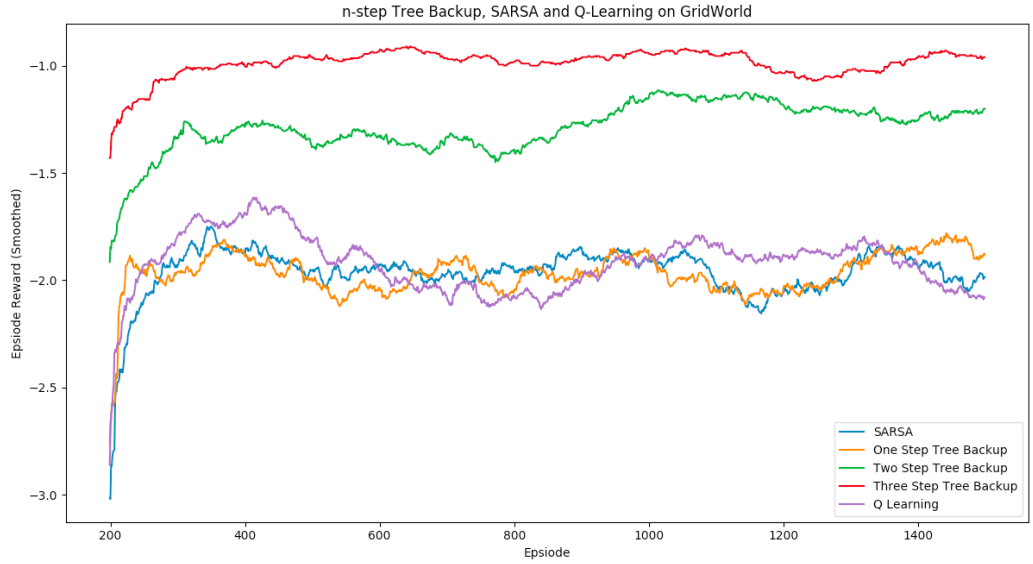
Our hypothesis is that this difference is performance is due to the nature of the Cliff World environment. Since the cliffworld environment assigns large negative rewards for falling off the cliff, it is reasonable for n-step backup algorithms to perform worse in this. This is because we are considering two steps ahead and computing all possible actions at a state two steps ahead. This makes the agent more likely to fall off the cliff and hence the degradation in performance.

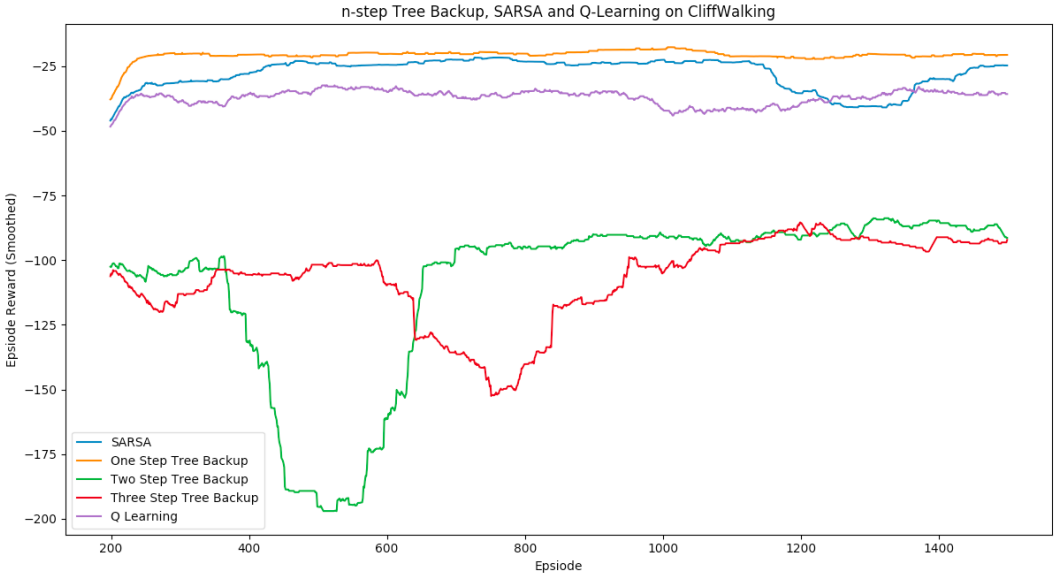


2.3 Compare Expected SARSA and n-step Tree Backup with SARSA and Q-Learning

We further evaluate the differences between the Expected SARSA and Tree Backup algorithms with our old SARSA and Q-Learning algorithm. Surprisingly, the n-step Tree Backup algorithms perform much better in the Grid World environment (where there is no large negative reward for taking bad actions), and in constrast performs much worse in the Cliff World environment.

Based on our experimental analysis, we hypothesize that the performance of n-step Tree Backup algorithms is perhaps largely dependent with the nature of the environment. If there are large penalties associated with taking bad actions in the environment, then n-step backup algorithms are expected to perform much worse. This is because all possible actions are considered (including the bad actions) at a state n-step ahead, which makes the agent more prone to performing worse. However, in environments such as the Grid World, where there are no penalties with bad actions, n-step backup algorithms perform much better.

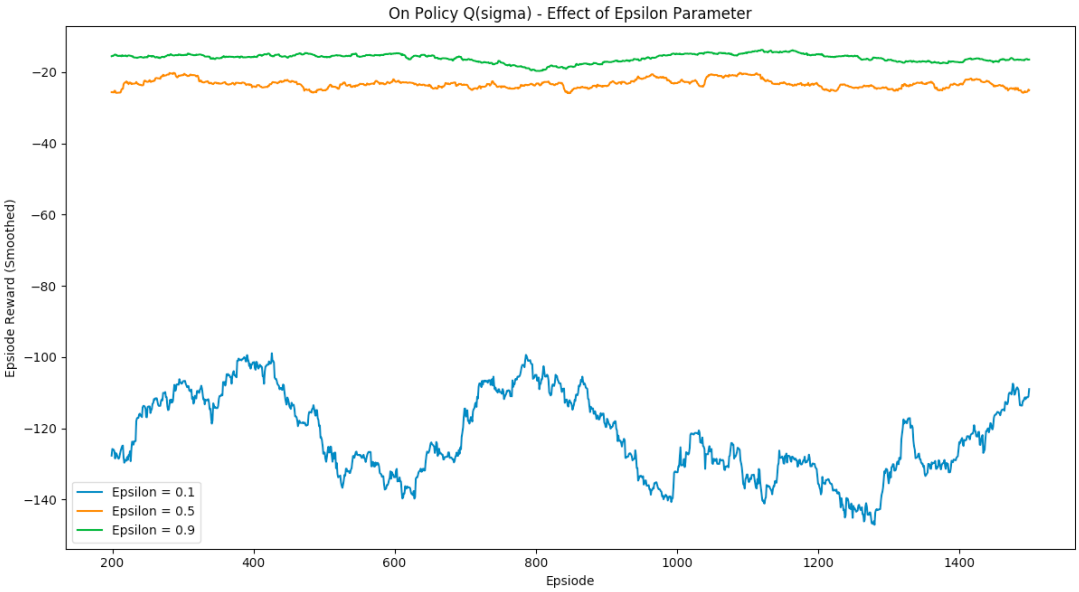




2.4 $Q(\sigma)$ Algorithm

First we demonstrate experimentally the performance of Q(σ) algorithm on our environments. Our results in addition to demonstraing Q(σ) algorithm, analyses the effect of the ϵ parameter in this on-policy algorithm.

Note that we only implemented the on-policy $Q(\sigma)$ algorithm. In a later coursework, we will demonstrate the performance of off-policy $Q(\sigma)$.

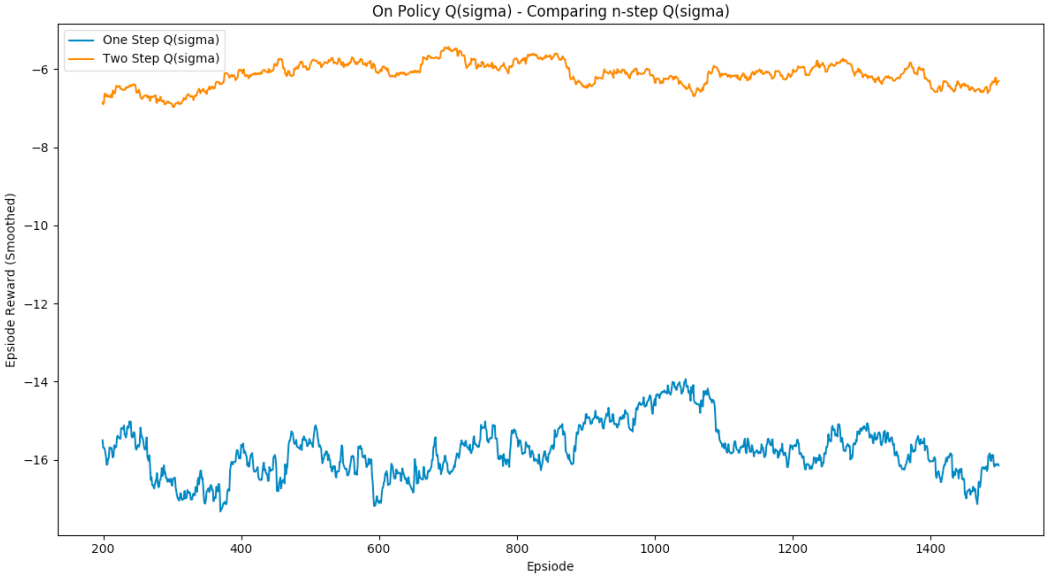


Interestingly, the $Q(\sigma)$ algorithm performs better with a large ϵ parameter. This emphasizes that random actions with a higher probability are more preferred in this algorithm, instead of taking greedy actions with respect to the Q function. In other words, the algorithm seems to perform better with random exploratory action.

In future work, it would therefore be interesting to analyse the effect of behaviour policy on the off-policy $Q(\sigma)$ algorithm, and compare its performance with different off policies (ϵ greedy, Thompson Sampling, Boltzmann policy and others).

n-Step $Q(\sigma)$ Algorithm

We then compare the differences between one step and two step $Q(\sigma)$ algorithm based on a high ϵ paramter of 0.9. Our results on the Grid World environment shows that two step $Q(\sigma)$ algorithm performs much better than one step algorithm.



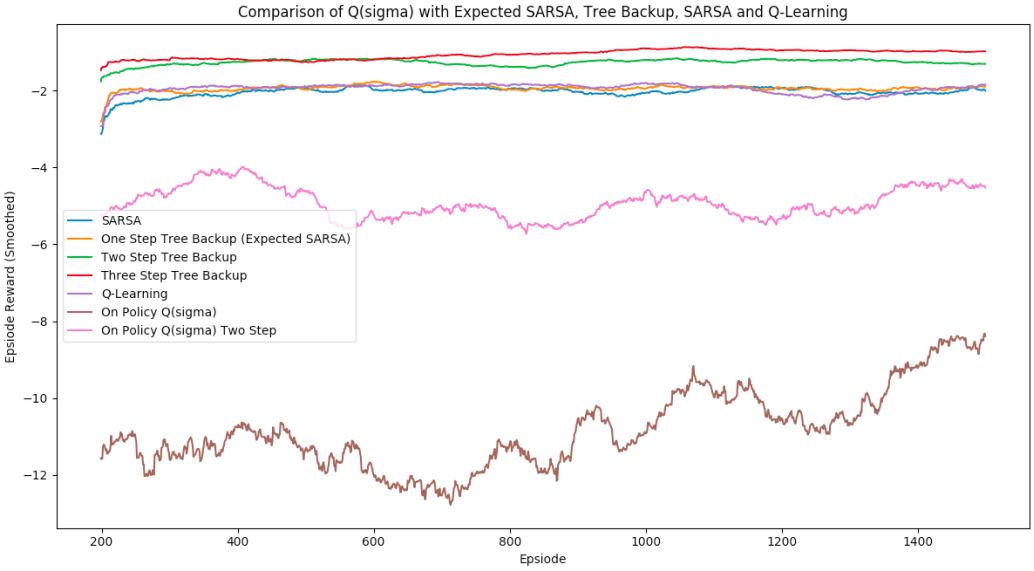
However, the $Q(\sigma)$ algorithm does not perform as well as Expected SARSA or other on or off policy algorithms we analysed previosuly. We furhter demonstrate that experimentally in the section below.

For the implementation of $Q(\sigma$ we choose a random σ parameter of 1 or 0 at every step and perform the Q step updates (which is eitehr Expected SARSA or SARSA). Our hypothesis for this poor performance for $Q(\sigma)$ lies on the fact that the environment may not be suitable for good performance of this algorithm. It may be that $Q(\sigma)$ will only perform good in carefully designed MDPs (which we do not do here). This is therfore left for future work in a furhter coursework or project.

It would also be interesting to see how well the off-policy $Q(\sigma)$ algorithm performs in the grid world MDP that we considered here.

2.5 Comparison of Q(sigma) with Expected SARSA, Tree Backup, SARSA and Q-Learning

Finally, we compare our implemented $Q(\sigma)$ algorithm with Expected SARSA and n-step Tree Backup algorithms on the Grid World MDP.



Our results show that $Q(\sigma)$ algorithm, both one step and two step do not perform as well as Expected SARSA and Tree Backups. In fact, the n-step Tree Backup algorithms seems to dominate and achieve a better cumulative reward than the other algorithms we considered here.

Finally, we further demonstrate our comparison of algorithms on the Cliff World MDP. In such a MDP, where there is a large negative reward assigned on taking bad actions, the Expected SARSA algorithm seems to perform the best compared to n-step Tree Backup. However, our $Q(\sigma)$ algorithm performs worse in both the grid world and cliff world environments.