DynaQ and its Variants
[Assignment-3]
(CS 394R: Reinforcement Learning)

Shobhit Chaurasia
Department of Computer Science

September 29, 2016

---

1 Problem Statement

This assignment is based on Exercise 8.4 in our textbook [1] (2^{nd} edition). Performance of Dyna, DynaQ+, and a variant proposed in this exercise is evaluated on a gridworld in different settings.

2 Introduction

DynaQ+ encourages testing of long-untried actions by awarding a bonus reward during planning (i.e. in the simulated experience) to these actions. In particular, if the modeled reward for a transition is r, and the transition has not been tried for τ time steps, then planning backups are done as if that transition produced a reward of $r + \kappa\sqrt{\tau}$, for some small κ [1]. Note that the exploration bonus are actually added to the Q-learning update in the planning stage, thereby, changing the estimated values of states and actions.
A variant of DynaQ+ (DynaQ+ Variant) uses the same exploration bonus, but instead of adding it to the backups, it is solely used in action selection.

3 DynaQ+ vs DynaQ+ Variant

Both the algorithms encourage continual exploration due to the built-in exploration bonus. In DynaQ+, since exploration bonus changes the estimated Q-values in the planning stage, the agent is able to simulate exploration to potentially far-off regions of the state space, and hence, the exploration is not limited to a local region around the agent's current state. By connecting the effects of simulated explorations around the state-space, DynaQ+ can achieve long range exploration, with the effect that it can potentially discover completely new policies, without necessarily following them.

On the other hand, exploration bonus in DynaQ+ Variant is solely used for action selection in the real-world, and has no effect on the Q-values, thereby limiting the exploration to states surrounding the agent's current state. This makes long range exploration less plausible and far slower since

the agent has to take each and every exploratory action in the real-world, limiting the rate of exploration to one per timestep. Contrast this to DynaQ+ where exploration is done in planning, which could happen multiple times per timestep.

4 Experiments

4.1 Setup

The environment is the gridworld shown in Fig. 1. The shaded cells represent obstacles. The values in cells are the rewards that the agent receives when it lands in those states. Transitions that take the agent out of the gridworld, or bump it against an obstacle have the effect of keeping the agent in the same cell.

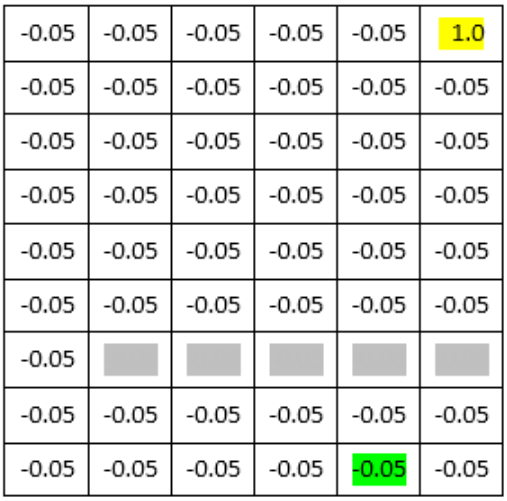


Figure 1: The Gridworld. Values within each cell represents the reward received on landing in that cell. Green cell is the start state; yellow cell is the goal state. Obstacles are shaded in grey.

The three algorithms are tested on the gridworld in different settings. For all the three algorithms, ϵ was set to 0.1, α to 0.1 (decayed by a fraction of 0.99 after each episode), and the number of planning steps per episode was set to 50. 150 episodes of each algorithm were executed. For DynaQ+ and DynaQ+ Variant, κ was set to 0.001. Since the actions taken in the first episode could have an unfair effect on the progress of the three algorithms, it was ensured that the sequence of actions taken in the first episode was the same across the three algorithms.

4.2 Experiment - 1

This experiment compares the performance of the three algorithms on a stationary setting as shown in Fig. 1. Since this is a stationary task, I suspected that the exploration bonus in DynaQ+ might help it find the optimal path faster than DynaQ, but would later lead DynaQ+ to perform slightly poorer than DynaQ (in spite of an evidence to the contrary in the textbook) because of the added exploration which is not useful in the long-term in a stationary setting. Similar

reasoning holds for DynaQ+ Variant.

Fig. 2 plots the cumulative reward received by the agent as a function of timesteps. Since landing in any state except the goal state incurs a negative reward, a higher cumulative reward is indicative of the agent following a shorter path to the goal. As expected, exploration helps DynaQ+ find the optimal path quicker. Moreover, in line with the observations stated in the book, and contrary to my belief, the added exploration does not hurt DynaQ+ in the long run, and it maintains its superior performance over DynaQ after many episodes. What's surprising is the exploration bonus during action selection does not affect DynaQ+ Variant much in either a positive or negative way (at least not until the latter half).

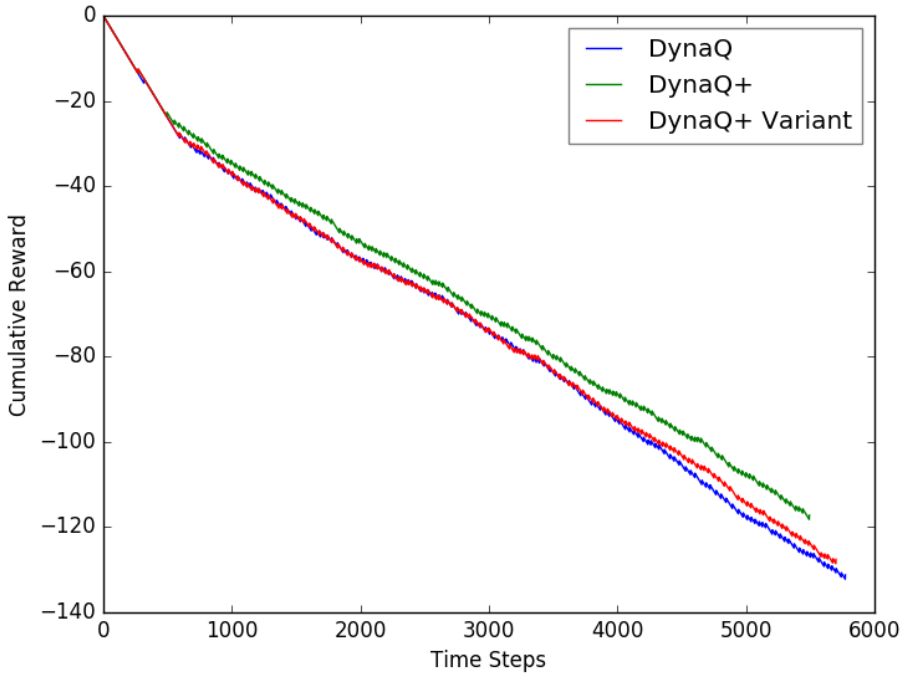
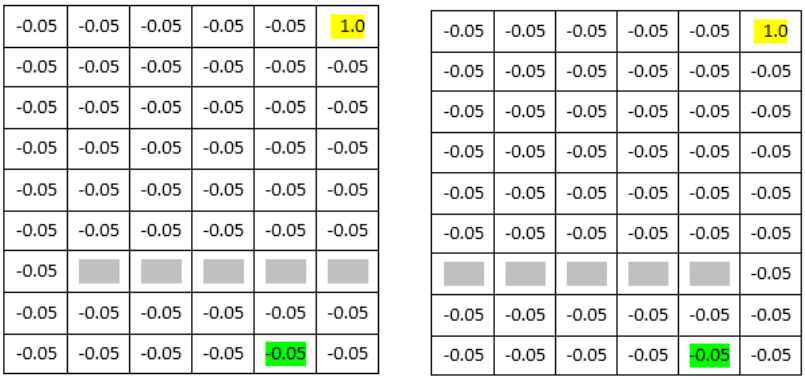


Figure 2: Cumulative reward in the stationary case.

4.3 Experiment - 2

This experiment compares the performance of the three algorithms on a non-stationary setting. After 1500 timesteps, the gridworld is changed from Fig. 3(a) to Fig. 3(b). This amounts to blocking of the optimal path discovered by the agent, and opening up a new (better) path. The added exploration bonus in DynaQ+ and DynaQ+ Variant should prove to be useful here, and I expected the agent to discover the new path faster with these algorithms.

Fig. 4 plots the cumulative reward received by the agent as a function of timesteps. Both DynaQ+ and DynaQ+ Variant discover the new path faster than DynaQ thanks to the extra exploration bonus. However, DynaQ+ discovers the new path significantly faster than DynaQ+ Variant. I believe that the reason behind this is the ability of DynaQ+ to explore 'more globally' as discussed in Section 3, which enables it to 'connect the exploratory dots' more fruitfully. Quite interestingly,



(a) Gridworld before change-point. (b) Gridworld after change-point.

Figure 3: The Gridworld. Values within each cell represents the reward received on landing in that cell. Green cell is the start state; yellow cell is the goal state. Obstacles are shaded in grey.

even though the added exploration gives DynaQ+ Variant an edge over DynaQ in uncovering a new path, it hurts it in long-term after the environment has stabilized (thereby making exploration futile), as seen by DynaQ outperforming DynaQ+ Variant later. This is in line with my hunch stated in Section 4.2. However, this effect was not visible in Fig. 2. A possible reason could be the length of the optimal path being far shorter after change-point in Fig. 3(b) as compared to that in Fig. 1. A longer optimal path gives DynaQ enough opportunity to explore (through its ϵ-soft policy), thereby potentially hurting it nearly as much as the added exploration bonus hurts DynaQ+ Variant.

4.4 Experiment - 3

This experiment compares the performance of the three algorithms on a non-stationary setting. After 1500 timesteps, the gridworld is changed from Fig. 5(a) to Fig. 5(b). This amounts to opening of an alternative and better path to the goal, rendering the original path non-optimal. This setting really seeks to exploit the 'global exploration' ability of DynaQ+ over DynaQ+ Variant. For the reasons discussed in Section. 3, I expected DynaQ+ to discover the new optimal path, while the other two keep following the previous, now sub-optimal path, oblivious to the change in the environment.

Fig. 6 plots the cumulative reward received by the agent as a function of timesteps. The results are in-line with the expectations, with DynaQ+ discovering the new optimal path, while the other two continue following their previous policies.

If we change the start state to the bottommost-rightmost cell, then the new path becomes even more accessible. In this setting, in addition to DynaQ+, both DynaQ+ Variant and DynaQ are able to discover the new optimal path (see Fig. 7). This is because the discovery of this new path requires comparatively less global exploration. Local exploration suffices, and both DynaQ (by chance, through its ϵ-soft policy) and DynaQ+ Variant (through both ϵ-soft policy and exploration bonus) are able to discover it. However, it is not intuitively clear why DynaQ+ Variant is able to discover the new path faster than DynaQ+.

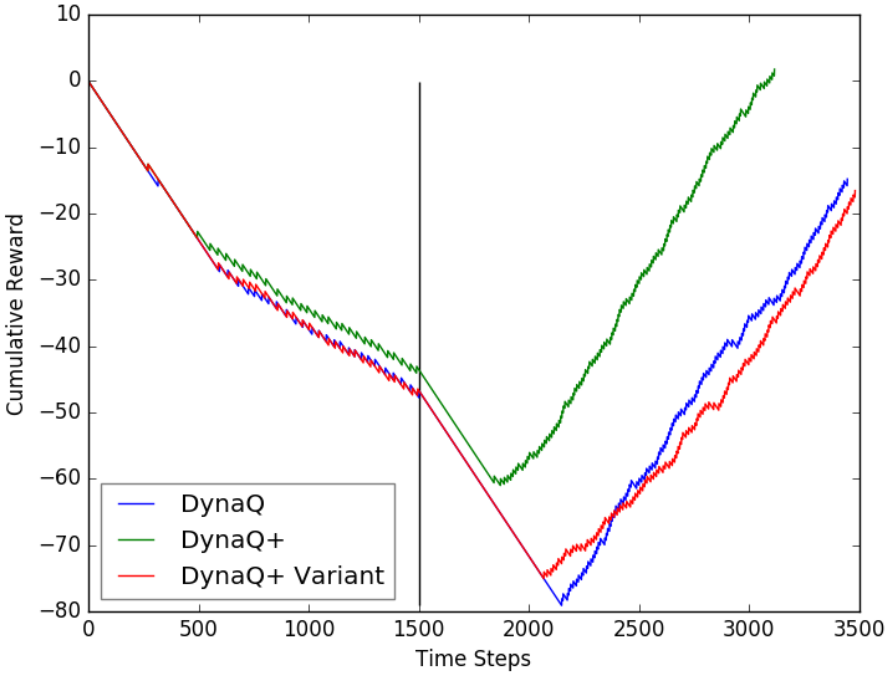


Figure 4: Cumulative reward in the non-stationary case where the original path is blocked to open a new one. The black vertical line denotes the change-point.

5 Conclusion

The added exploration bonus in DynaQ+ encourages exploration, which could come handy in non-stationary environments, giving it an edge over DynaQ. DynaQ+ Variant also leverages this exploration bonus, albeit in a way that does not allow it to simulate global exploration like DynaQ+. An empirical observation from implementing these algorithms is that the value of κ needs to be chosen very carefully. Its scale should be such that the exploration bonus is of the order of the average rewards received at each timestep. However, even a value that is a little too high could render the agent exploring way too aggressively.

References

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

-0.05	-0.05	-0.05	-0.05	-0.05	1.0
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05					
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05

(a) Gridworld before change-point.

-0.05	-0.05	-0.05	-0.05	-0.05	1.0
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05					-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05
-0.05	-0.05	-0.05	-0.05	-0.05	-0.05

(b) Gridworld after change-point.

Figure 5: The Gridworld. Values within each cell represents the reward received on landing in that cell. Green cell is the start state; yellow cell is the goal state. Obstacles are shaded in grey.

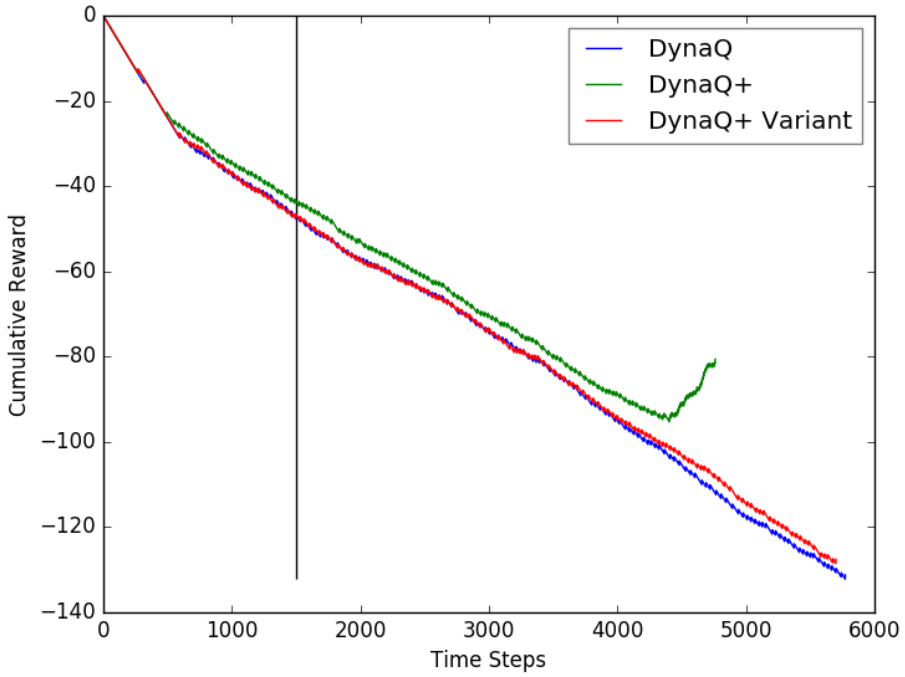


Figure 6: Cumulative reward in non-stationary case where a new optimal path is opened. The black vertical line denotes the change-point.

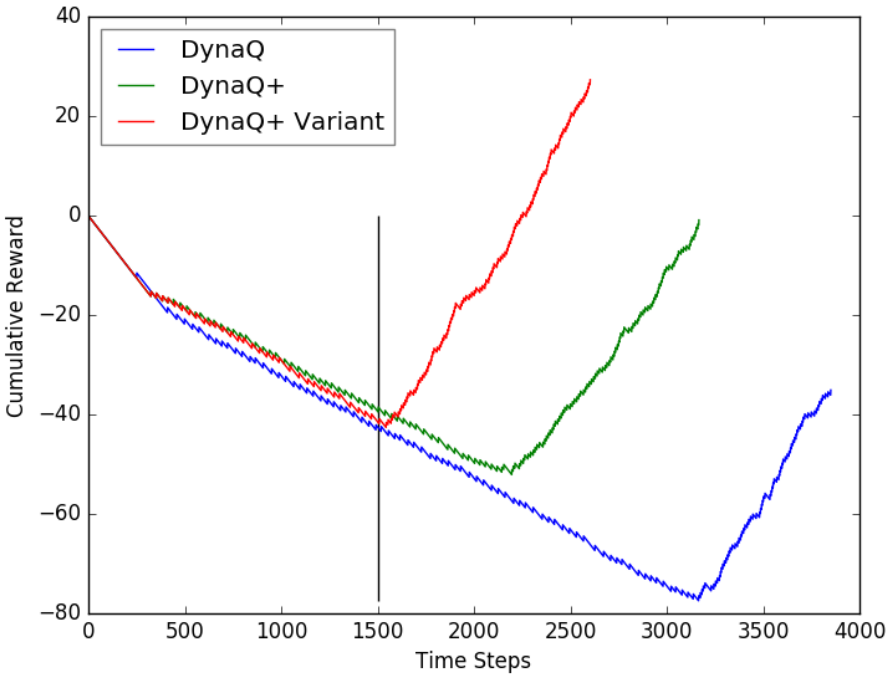


Figure 7: Cumulative reward in non-stationary case with the bottom-right cell being the start state, where a new optimal path is opened. The black vertical line denotes the change-point.