Polytechnics School of ULB
First Master in Computer Engineering
Section: Computational Intelligence

Learning Dynamics

Assignment 4 : Q-Learning

Alexandre Balon-Perin

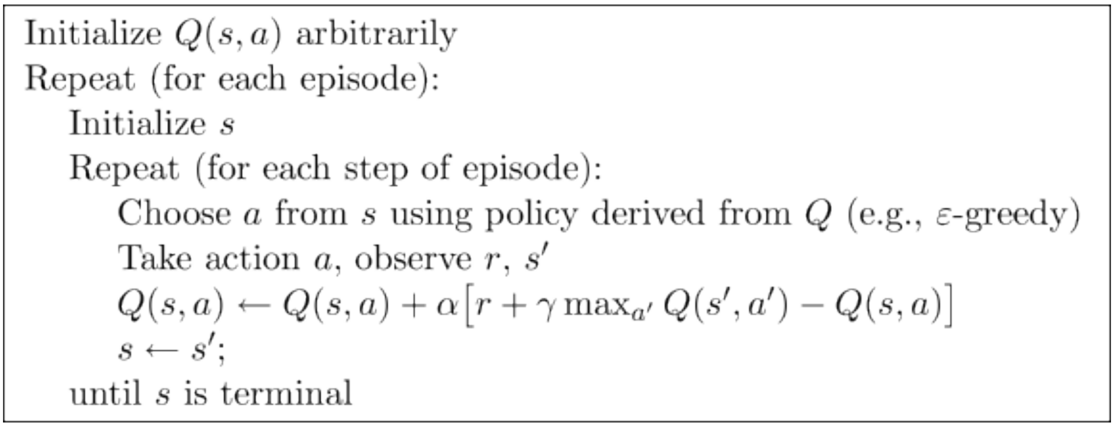
Academic Year 2010-2011

Table of Contents

Introduction

In this assignment we were asked to implement the Q-Learning algorithm with the following specifications:

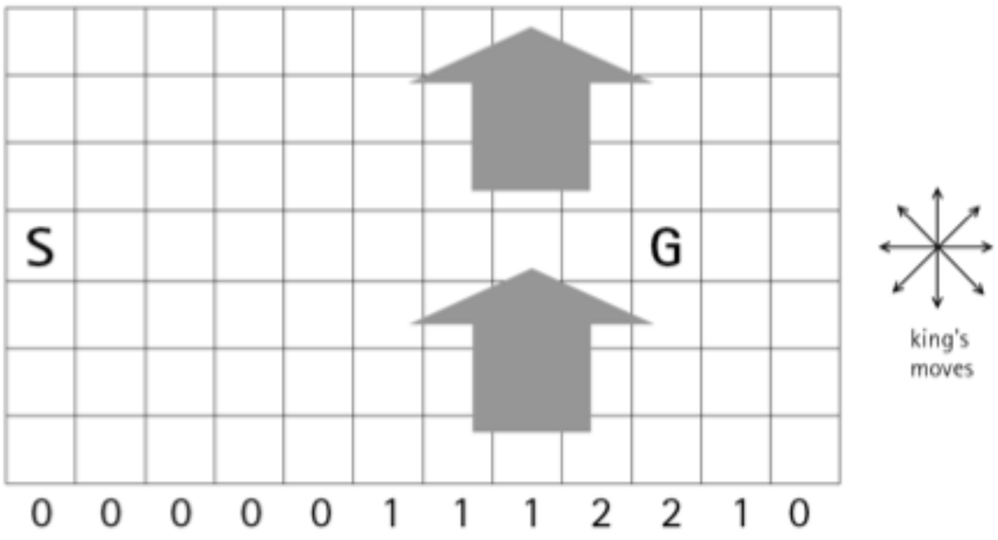
- $\alpha = 0.1$
- $\gamma = 0.9$
- Initial Q(s,a) = 0 $\forall$ a,s
- Rewards:
  - $r_s = -1$ (reward for all actions)
  - $r_g = 10$ (reward for action leading directly to the goal)
- $\varepsilon = 0.2$ (ε-Greedy selection method)



```
Initialize $Q(s,a)$ arbitrarily
Repeat (for each episode):
    Initialize $s$
    Repeat (for each step of episode):
        Choose $a$ from $s$ using policy derived from $Q$ (e.g., $\varepsilon$-greedy)
        Take action $a$, observe $r$, $s'$
        $Q(s,a) \leftarrow Q(s,a) + \alpha\big[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\big]$
        $s \leftarrow s'$;
    until $s$ is terminal
```

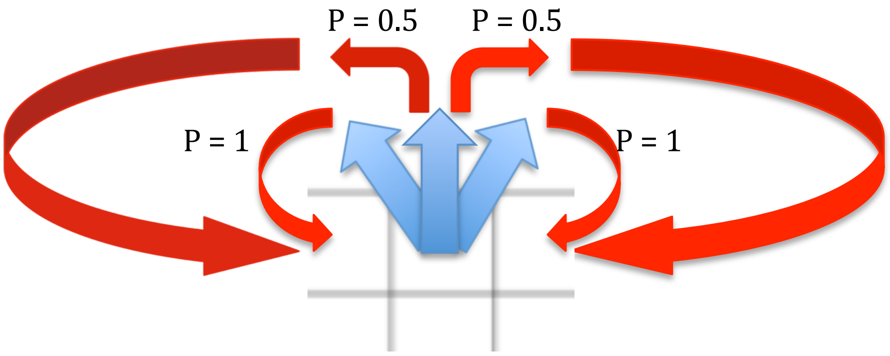
We will use this algorithm on the Windy Gridworld that will be presented in the next chapter.

The Windy Gridworld



The Windy Gridworld is a start grid world of size 12x7 with a start (S) and a goal (G). There is a crosswind upward through the middle of the grid. The available actions in each cell are the king's moves (8 actions).

If any action would bring the agent outside the grid, it ends up in the nearest cell.
Here is an example of how the program manages an "out-of-bond" move:



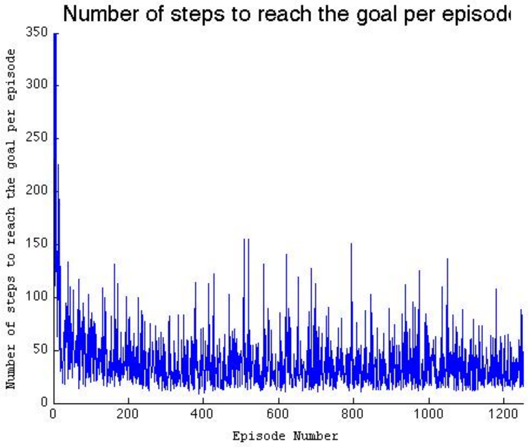
Where P is the probability to follow the path represented by the red arrows.

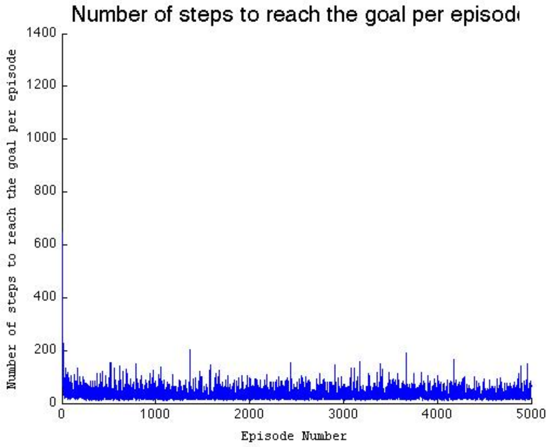
The mean strength of the wind is given below each column, in number of cells shifted upward. Due to stochasticity, the wind sometimes varies by 1 from the mean values given for each column.

Results

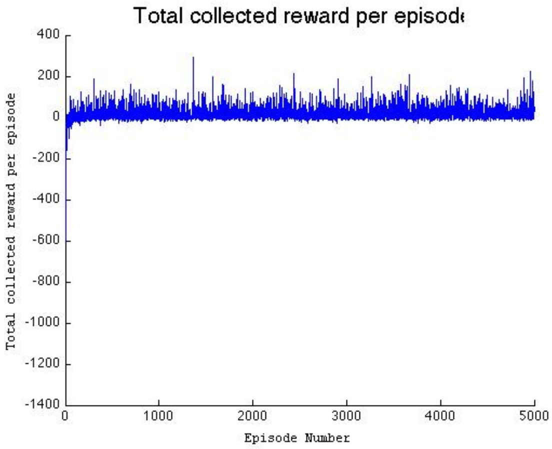
Plots

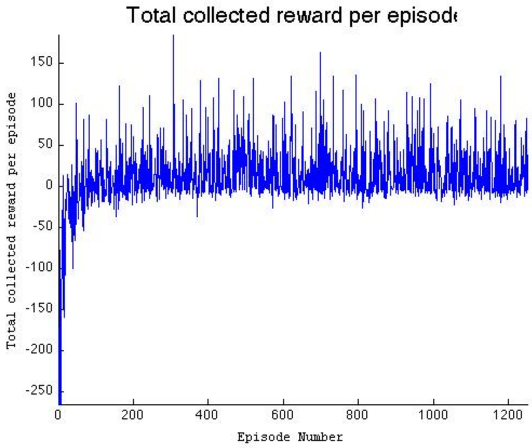
Number of steps to reach the goal per episode



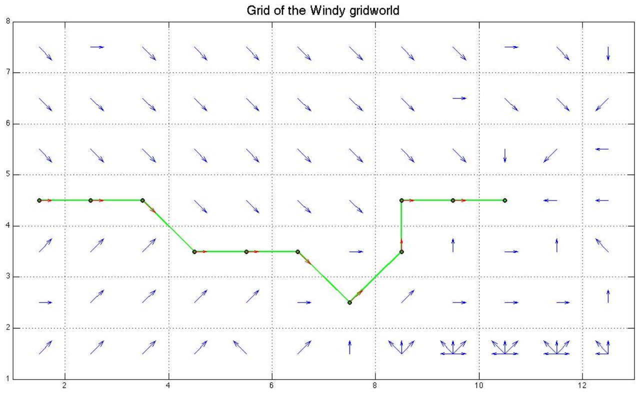


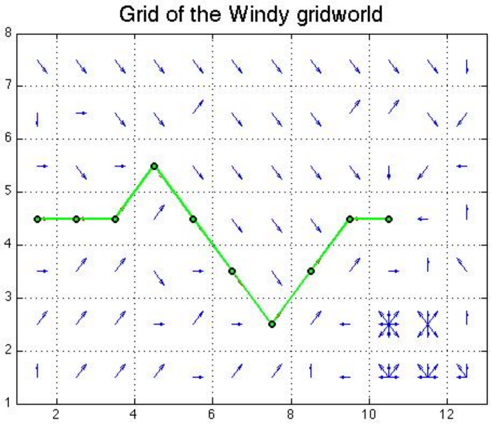
Total reward per episode

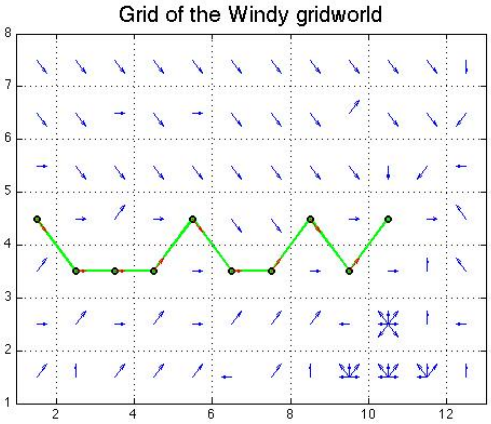




Samples of paths on the Gridworld







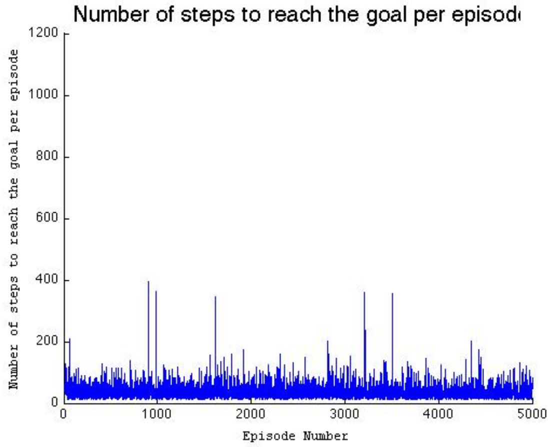
Discussion

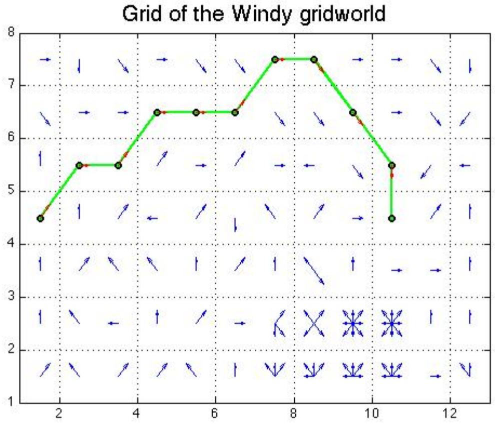
As we expected, these diagrams show that the number of steps to reach the goal per episode decreases exponentially whereas the total collected reward increases exponentially. We can also see that the lower part of the windy columns is not explored because it is difficult for the agent to reach these cells. In fact, it is blown upward almost all of the time before it could reach them.

Discussion

1. $r_g = 1000$

Setting the reward for reaching the goal to 1000 does not help the agent to reach the goal more quickly. Instead, the agent wastes a lot of steps looking around without finding the goal. We can see on the graph below that at some point the number of steps reaches a peak where it should be very low. The path taken by the agent is much longer then the one taken with $r_g = 10$ as well. The arrows indicating the best action do not make much sense either.

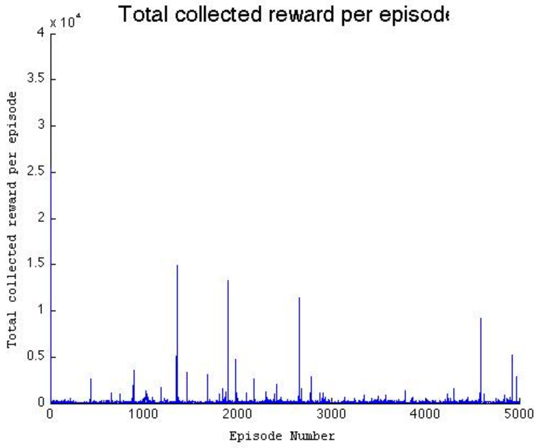


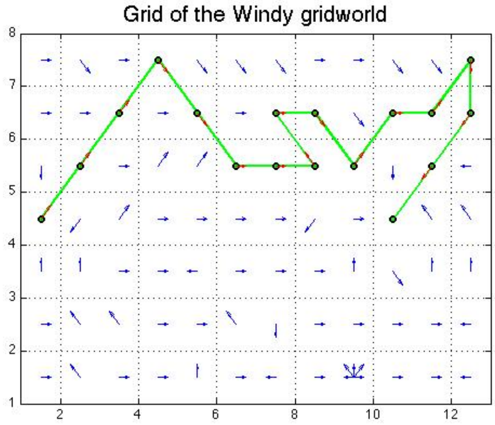


If we change the immediate reward so that it is not negligible in comparison to r_g we can obtain a much better result and the convergence is as fast as before. ($r_S = -100$ for example)

2. $r_S = 1$

Setting the reward of each action to 1 instead of -1 will have great influence on the learning process. Giving a reward for the immediate action closer to the one for the action that directly goes to the goal means that the agent does not really "care" about going to the goal cell. The distinction between going to the goal and doing something else is not so clear now.





3. ε = 1

If we change the value of ε in the ε-Greedy selection method, we can see that the selection method becomes random. In this case the number of steps to reach the goal increases dramatically because the agent does not follow any coherent policy but randomly chooses an action to take without taking the Q function into account. The agent explores all the time but does not exploit the information acquired over the episodes. The number of steps to reach the goal per episodes increases dramatically.

4. Influence of the stochasticity of the wind

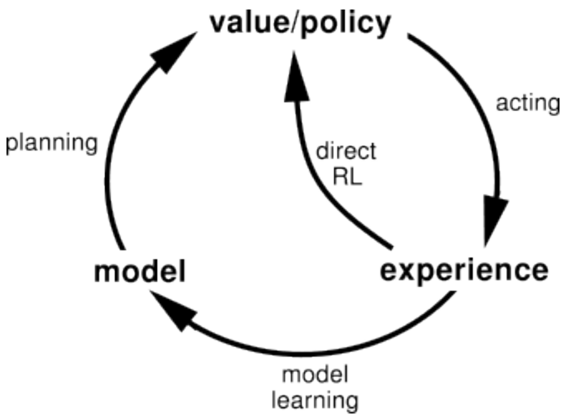
If we add the wind to the gridworld, the Q function will be somehow mistaken at some point. For example, if the agent takes an action a' from the state s' and then because of the wind end up on the goal's cell, the Q value will be updated with a higher value then it should. The next time the agent will be in the same state s', the best action will be again a' but if the wind does not blow that much this time the agent will end up in a cell which is not the goal and maybe just go away instead of going to the goal. Adding stochasticity definitely makes it much more difficult for the agent to learn something.

5. Planning and Learning

NB: This first paragraph is for me to remember the difference. I hope you don't mind :)

"The difference is that whereas planning uses simulated experience generated by a model, learning methods use real experience generated by the environment.

Indirect methods often make fuller use of a limited amount of experience and thus achieve a better policy with fewer environmental interactions. On the other hand, direct methods are much simpler and are not affected by biases in the design of the model."
(Sutton S. Richard and Barto G. Andrew, *Reinforcement Learning: An introduction*)



Consider we would like to incorporate planning in our learning process in order to speed up the learning. How can you extend/change the Dyna-Q algorithm to non-deterministic environments like our Windy Gridworld?

The Dyna-Q algorithm must be change in a way that apply the stochastic behavior the wind before returning s' and r to the Model. The resultant state might be different than the one previously selected by the selection method such as ε-Greedy.