Pierre and Marie Curie University

Experiments

Learning from expert

Author:
Matthieu Zimmer

Supervisors:
Paolo Viappiani
Paul Weng

July 26, 2013

Version 1.0

Contents

1 Environnement

1.1 Mountain car

A standard testing domain in reinforcement learning, is a problem in which an under-powered car must drive up a steep hill. Since gravity is stronger than the car's engine, even at full throttle, the car cannot simply accelerate up the steep slope. The car is situated in a valley and must learn to leverage potential energy by driving up the opposite hill before the car is able to make it to the goal at the top of the rightmost hill.

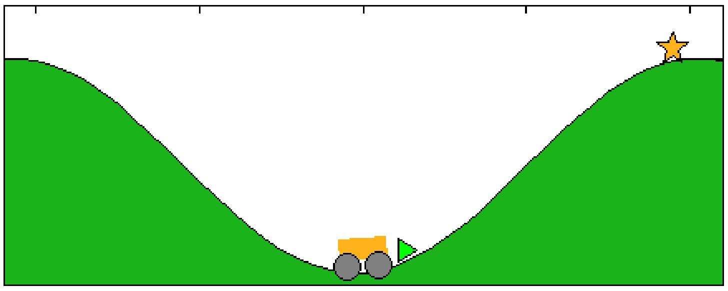

Figure 1: The Mountain Car Problem

1.1.1 State Variables

Two dimensional continuous state space : Velocity = (-0.07,0.07) and Position = (-1.2,0.6)

1.1.2 Actions

One dimensional discrete action space : Motor = (left, neutral, right)

1.1.3 Reward

For every time step: -1, when goal is reached: +1

1.1.4 Update Function

Velocity = Velocity + Action *0.001+cos(3*Position)*-0.0025
Position = Position + Velocity

1.1.5 Starting Condition

Position = -0.5 Velocity = 0.0

1.1.6 Termination Condition

Position >= 0.6

1.2 Grid Game

The mountain car problem gives a -1 reward at every step, and the state allows to determine exactly the best action to take. So we could think that there is no state more important than other (to give feedbacks).

Here, we introduce an other problem where rewards will be different according to our state.

The agent have to reach every green boxes in the given order.

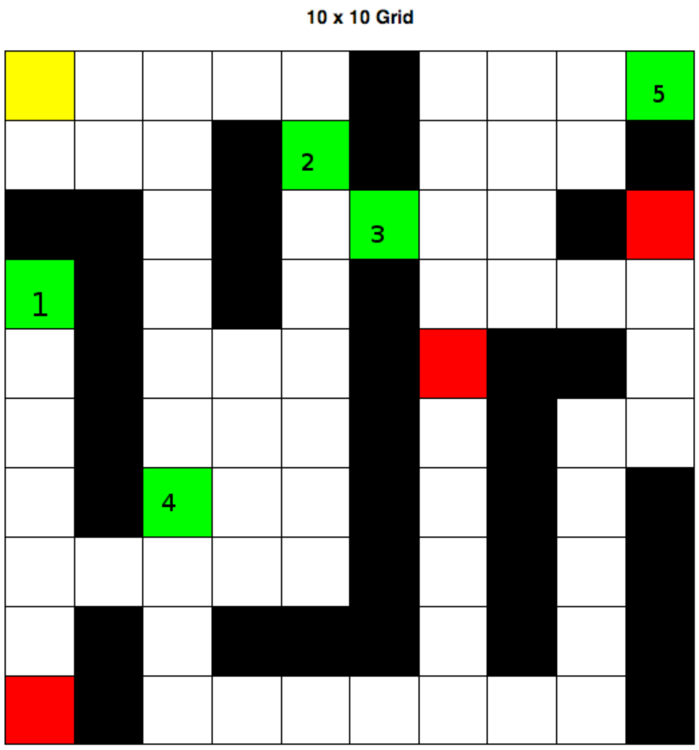

Figure 2: Grid World

1.2.1 State Variables

Four dimensional discretized state space :
X = {0, 9} ; Y = {0, 9} ; XGOAL = {0, 9} ; YGOAL = {0, 9}

1.2.2 Actions

One dimensional discrete action space : Direction = (left, up, right, bottom)

1.2.3 Reward

white boxes : -1
green boxes : +10 (only the first time)
red boxes : -10

1.3 Second Grid Game (LS)

The main difference in this problem, is that, given a state there isn't only one best possible action.

The agent have to reach every green boxes in any order.

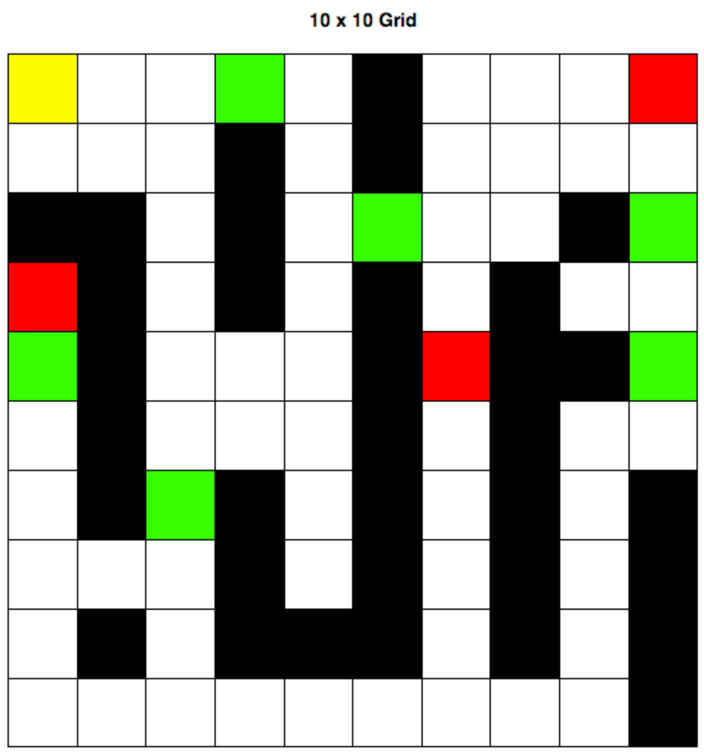

Figure 3: Gris World Insconsistent State

1.3.1 State Variables

Four dimensional discretized state space : X = {0, 9} ; Y = {0, 9}

1.3.2 Actions

One dimensional discrete action space : Direction = (left, up, right, bottom)

1.3.3 Reward

white boxes : -1
green boxes : +10 (only the first time)
red boxes : -10

1.3.4 State inconsistency

Imagine your in the boxe (6,0), you only know that you are in this box, you don't already know if you reach the goal in (4,0). Thus, there is two different reasonable chooses.

2 Comparing strategy : how the advice affects the learner

2.1 Strategies

2.1.1 Max

$Q(s,a) = max(Q(s,\_)) + \delta$

2.1.2 Override (Lucky Exploration)

$\pi(s) = a$
$Q(s,a)$ will be updating according with the reward given the next time in this state.

2.1.3 Max + Override

$\pi(s) = a$
$Q(s,a) = max(Q(s,\_) + \delta$

2.1.4 Others

We could imagine an other strategy as Lucky Exploration only for the next time we encounter this state (not fixed for everytime).

2.2 Experiment A : Advice after

In this experiment, we have two agent, a teacher (QLearning) and a learner (QLearning). The teacher is favored to advice (it is done to compare advice effects) : $+10$ if advice else -10. With this reward function, it is not important to display the algorithm "advice before" because it will advice all the time.

The teacher advices the learner with the best action to take (according to the best policy previously computed). They have the same state representation.

2.2.1 Mountain Car

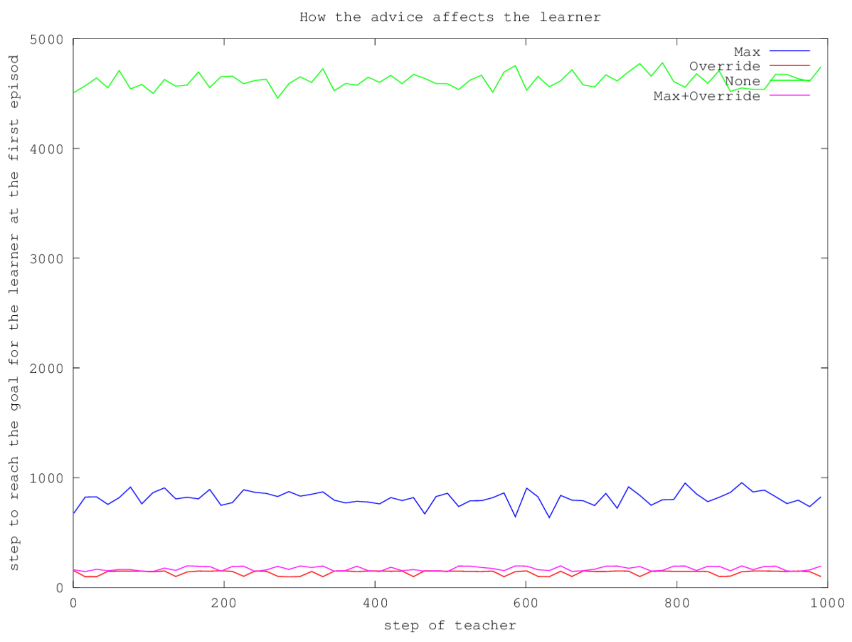

Figure 4: Teacher advices all the time. Number of step to reach to goal at the first episod

The two best strategies are Override and Max+Override. (Max+Override is a little quite better because it updates the Q value directly).

2.2.2 Grid World

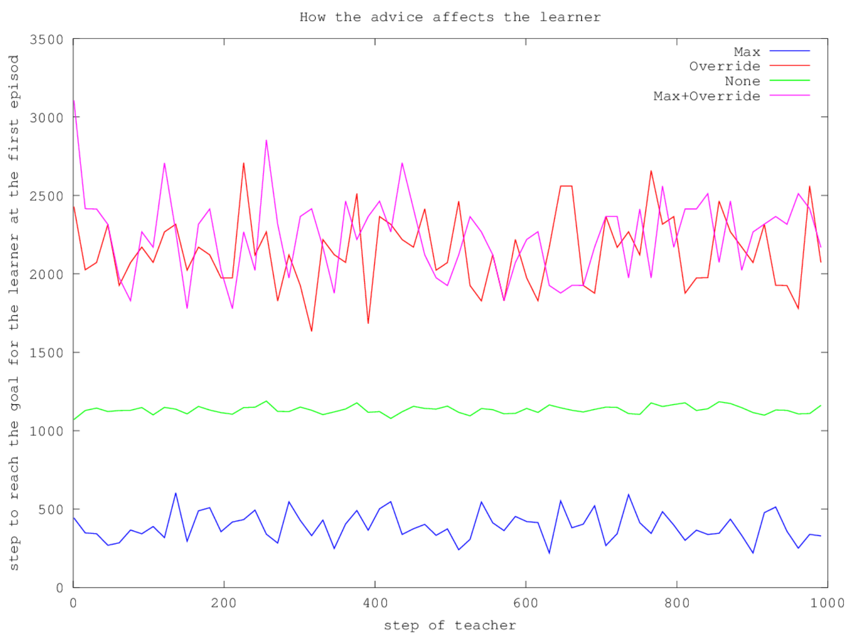

Figure 5: Teacher advices all the time. Number of step to reach to goal at the first episod

This time in the grid world, the best strategy is Max. The fixed policy strat are really bad, more than without advice.

2.2.3 Grid World (LS)

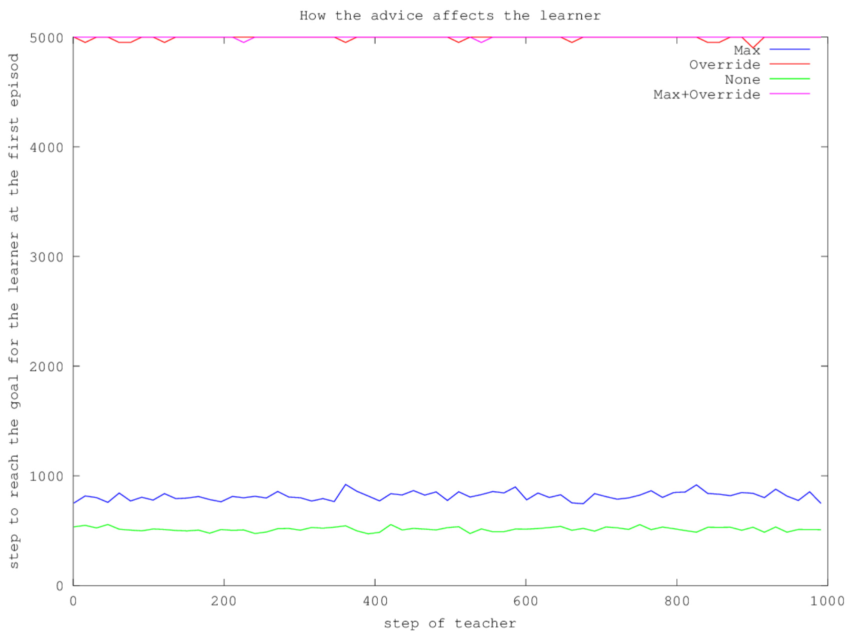

Figure 6: Teacher advices all the time. Number of step to reach to goal at the first episod

In this environnement, all strategies are bad when the teacher advice after because of the different good choose possible in a same state.

2.3 Experiment B : Advice before

This experiment is used to compare the advice effect when the teacher can correct a mistake of the learner before he perform it. Also, we need to use a different reward function (based on a cost when he advice) for the teacher, otherwise he will advice always and all algorithms will be at the same performance.

3 Comparing teaching strategies : teacher reward function

3.1 Reward functions