Transfer Learning in Gridworld

Shun Zhang[*]

April 30, 2013

1 Environment Setup

For this assignment, I used the envrionment [1] used in Jong and Stone [2008]. The agent starts from a random grid in the upper-left room. The terminal state is a grid in the bottom-right room, with a reward of +1. The transition is assumed to be known. A 2×2 rooms example is shown in Figure 1.

This envrionment is scalable. The size and number of rooms can be changed and the changed environment is reasonably related. So I used this for examining the performance of transfer learning.

In the following sections, I'll describe algorithms that can transfer an environment in a small scale to one in a large scale. Their will be compared with the performance of learning the target tasks directly.

2 Learning Algorithm

Policy, reward and Q functions can all be possibly transfered (Taylor and Stone [2011]). *π-reuse Exploration Strategy* can be directly applied only when the source and targe is different in goal, same in states, actions and transitions (Fernández and Veloso [2006]). Regarding the problem in this assignment, states and rewards are different between source and target tasks. Transitions are almost the same, except the ones bumping into the walls.

This problem is solved by states aggregation in the target task. After transfering a policy into a larger domain, I assume the policy of adjacent states is the same (Table 2). There are apparent flaws in this design - the transitions are not correct on the border of the wall, especially when layout changes. This can be ignored as we are using model-free learning.

[*]Programming Assignment 4.

[1]The source code is from Reinforcement Learning project of CS343 Artificial Intelligence (http://www.cs.utexas.edu/ pstone/Courses/343spring12/) course. The framework is provided. The Q-learning part was pair programmed by Paul Nguyen and me. Now I implemented other experiment utility functions based on it for this assignment.

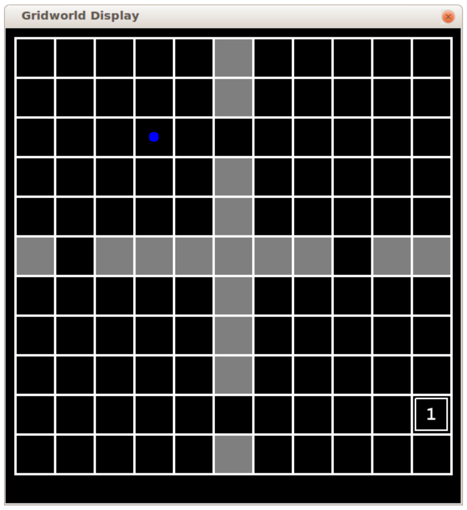

Figure 1: The setup of the envrionment. There are four 5x5 rooms. There is one entry to connect adjacent rooms. The agent starts from a random grid in the upper-left room. The terminal state is a grid in the bottom-right room, with a reward of +1.

Down	Left
Right	Right

Down	Down	Left	Left
Down	Down	Left	Left
Right	Right	Right	Right
Right	Right	Right	Right

Table 1: For example, the upper table is the learned policy in source task, in a 2x2 gridworld. The lower table is the transferred policy in the new task, which is in a 4x4 gridworld. Assume both envrionments have no obstacles. When the goal state is still the same, the new policy should be almost the correct one.

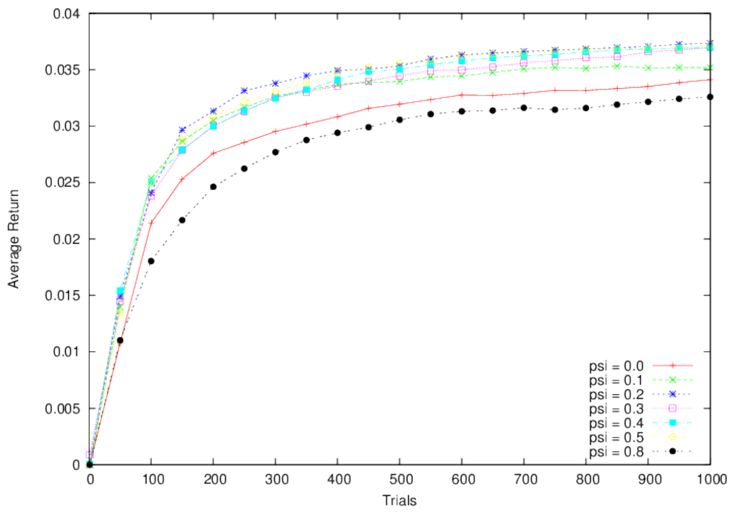

Figure 2: Comparison of different ψ values, transfering from small 4-room world to big 4-room world (Figure 1). 10 runs for each data points.

3 Empirical Results

For comparision purpose, I only need to select an algorithm that can support off-policy learning. So I used Q-learning here. For parameter setting, discounter $\gamma = 0.9$, learning rate $\alpha = 0.3$, exploration rate $\epsilon = 0.3$. These are kept constant for all the experiments.

In the target task learning, the probability of consulting source task, ψ, is a variable in the experiments. With probability of ψ, the agent uses the policy in the source task. With probability of $(1 - \psi)\epsilon$, the agent explores. With probability of $(1 - \psi)(1 - \epsilon)$, the agent exploits using its own policy.

The agent should be more and more confident on its own policy after more episodes. So after one episode, ψ is reduced by a decay factor v, which is $\psi = \psi v$ after one episode. I kept $v = 0.95$ in these experiments. These are refered on Fernández and Veloso [2006].

I ran 1000 episodes Q-learning on a gridworld which has 4 rooms, each of which has 2x2 grids. It's like a smaller version of Figure 1. I used the policy learned in this environment as the one on source task.

The two experiments are transfering policy from a small 4-room world to a big 4-room world (Figure 2), and transfering from a small 4-room world to a 6-room world, with the same size (Figure 3).

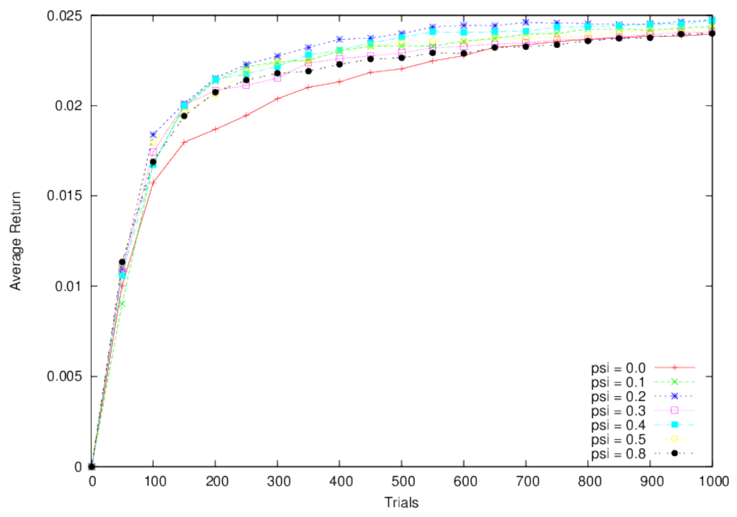

Figure 3: Comparison of different ψ values, transfering from small 4-room world to big 6-room world. 10 runs for each data points.

4 Discussion

Like everything, extremes can lead to bad performance. $\psi = 0$ is the same as Q-learning, which can serves as a baseline here. It transfers nothing from the source task when learning the new task. $\psi = 0.8$, on the opposite, transfers much from the source task.

There is a decay factor υ which reduce ψ after every episode. This not only impelements the confidence of the agent over time (less relience on the source task), but also guarantees the *asymptotic performance* (Taylor and Stone [2011]). Our algorithms will converge as same as Q-learning at last.

Jumpstart can also be observed in every case except when $\psi = 0.8$ in Figure 2. I think the reason is that policy in the source task doesn't provide useful information, and the agent has to adopt them with a large probability. Even though, when ψ decreases, it will converge.

References

Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of AAMAS'06, the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Hakodate, Japan, May 2006.

Nicholas K. Jong and Peter Stone. Hierarchical model-based reinforcement learning: Rmax + MAXQ. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, July 2008.

Matthew E. Taylor and Peter Stone. An introduction to inter-task transfer for reinforcement learning. *AI Magazine*, 32(1):15–34, 2011.