Foundations of Agents: Practical Assignment 1

Kevin Trebing (i6192338)

October 12, 2018

1 Tower of Hanoi problem

In a Tower of Hanoi problem the agent needs to move disks of different sizes from one pin to another pin. Furthermore, the disks need to be in the order that a smaller disk needs to be on top of a bigger one. Only one disk can be moved at a time and only the topmost disk can be moved. In our problem we have 3 pins and two disks. The starting position is pin 1 and the smaller disk is on the bigger disk. The goal is to move the disks to pin 3.

1.1 Description of States

Notation:

- a = small disk,
- b = big disk,
- 1 = pin1,
- 2 = pin2,
- 3 = pin3,
- ab = a is on b

We have 12 different possible states:

State	Pin		
s_0	ab1	2	3
s_1	1	ab2	3
s_2	1	2	ab3
s_3	ba1	2	3
s_4	1	ba2	3
s_5	1	2	ba3
s_6	b1	a2	3
s_7	a1	b2	3
s_8	b1	2	a3
s_9	a1	2	b3
s_{10}	1	a2	b3
s_{11}	1	b2	a3

1.2 Description of Actions

We have 6 different actions that the agent can take.

Action	effect
a_1	move a to pin1
a_2	move a to pin2
a_3	move a to pin3
b_1	move b to pin1
b_2	move b to pin2
b_3	move b to pin3

2 How the agent learns the optimal result for every initial state

Every time the agent ends in the absorbing state the new state is randomly chosen. Given enough resets the agent will start in every state enough times to approximate the optimal result. In my implementation I do 10000 iterations which results in every state being the starting state about 833 times.

3 Optimal policy

The optimal policy describes for every state the best action the agent should take.

- $\pi(s_0) = a_2$
- $\pi(s_1) = a_1$
- $\pi(s_2) = a_1$
- $\pi(s_3) = b_3$
- $\pi(s_4) = b_3$
- $\pi(s_5) = b_1$
- $\pi(s_6) = b_3$
- $\pi(s_7) = b_3$
- $\pi(s_8) = a_2$
- $\pi(s_9) = a_3$
- $\pi(s_{10}) = a_3$
- $\pi(s_{11}) = a_1$

4 Q-values

The q-values of of the states given the optimal policy:

- $u(s_0) = 75.30$
- $u(s_1) = 75.26$
- $u(s_2) = 0.00$
- $u(s_3) = 87.11$
- $u(s_4) = 86.98$
- $u(s_5) = 66.62$
- $u(s_6) = 85.63$
- $u(s_7) = 85.99$
- $u(s_8) = 74.99$
- $u(s_9) = 98.54$
- $u(s_{10}) = 98.83$
- $u(s_{11}) = 74.96$

5 Difference to value and policy iteration

Convergence depended on the learning rate a lot. At first I used $\frac{1}{numVisits}$, but this resulted in bad results. The reason for this is that the initial probability of choosing an action is the same for every action, but as soon as the first q-value for that state is updated with an initial learning rate of 1 the probability to choose this action is way higher than the other actions.

When fixing the learning rate at 0.01 the probability of choosing an action does not change very much, this is what we want in the beginning of learning to encourage exploration. A small learning rate stabilizes learning.

6 Convergence speed

Since I have a fixed learning rate of 0.01 it needs a lot of loops to converge to optimal values (optimal values are the values that were calculated by policy iteration in the earlier assignment). Using 10000 iterations I get very similar q-values compared to the optimal utility values. The time for 10000 iterations are about 2.5s.

7 Note

The Hanoi.py file requires Python 3.6.