**Hanoi Tower using Value and Policy iteration**

**The assignment**

Solve a Tower of Hanoi problem with three pins 1, 2 and 3 and two disks A and B. Disk A is larger than B. The goal is to move the two disks to pin 3 such that the lager disk A is at the bottom and the smaller disk B is at the top. Reaching this goal has a reward of: 100. The agent can move only one disk at the time. We do not forbid to put the larger disk A on top of the smaller disk B, but there is a penalty for doing this; i.e., a reward of: -10. To encourage the agent to solve the problem in a minimal number of steps, all actions that do not result in the goal state or the penalized state have a reward of: -1.

Our agent can make mistakes. When moving a disk from pin i to pin j, the agent may actually put the disk on pin k where k≠i and k≠j. The probability of making a mistake is: 0.1. The discount factor of future rewards: γ=0.9

Model this problem as an MDP and solve it using both Value iteration and Policy iteration.

**State description**

The states have been named with 4 characters, an integer after ‘b’ represents the pin where the **big** disk is placed. An integer after ‘s’ represents the pin where the **small** disk is placed. Therefore, the patter is always letter + number + letter + number. It is important to notice that when both disks are in the same pin, the order of letters indicate which one is on top (the second is on top of the first one), if they are in different pins then the couple will always indicate where the big disk is. Given this policy, the 12 possible states are named:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| b1s1 | b1s2 | b1s3 | s2b2 | s3b3 | b3s3 | b2s3 | b3s3 | b2s2 | b3s1 | b2s1 | s1b1 |

**Actions description**

There are six possible actions, moving the small disk to the pin 1, 2 or 3 and the same for the big disk. With the constraint that if both disks are in the same pin only the disk on the top can be moved, therefore, there is a maximum of 4 possible actions at each state. The second constraint is that null moves are not considering meaning that if the small disk is in the pin 1, moving it again to the pin 1 is not considered as a move.

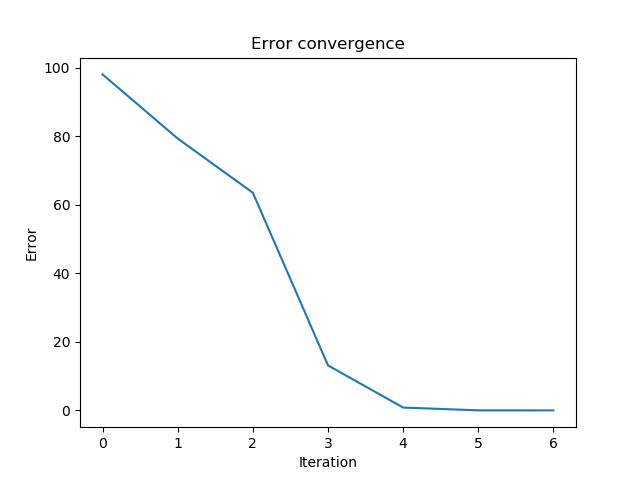
To name the actions, the pattern is similar to the states ‘s1’ means moving the small disk to the pin 1 and ‘b2’ means moving the disk to the pin 2. Therefore, the possible actions are:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| s1 | s2 | s3 | b1 | b2 | b3 |

**Results**

The result with value and policy iterations are the same, to check this, we created an algorithm to run both 1.000 times and compare the output. The result was always the same:

|  |  |  |
| --- | --- | --- |
| **State** | **Optimal policy** | **Utility** |
| b1s1 | s2 | 75.38 |
| b1s2 | b3 | 85.92 |
| b1s3 | s2 | 75.38 |
| s2b2 | b3 | 86.75 |
| s3b3 | b1 | 66.84 |
| b3s2 | s3 | 98.79 |
| b2s3 | s1 | 75.38 |
| b3s3 | Absorbing state | |
| b2s2 | s1 | 75.38 |
| b3s1 | s3 | 98.79 |
| b2s1 | b3 | 85.92 |
| s1b1 | b3 | 86.75 |

In the following graphs it can be seen how every iteration, the error gets smaller every time.