Middle East Technical University Department of Computer Engineering

CENG 567

Design & Analysis of Algorithms

Spring '2016-2017

HOMEWORK 2

Due date: 15 May 2017, Monday, 23:59

REINFORCEMENT LEARNING

Congratulations for discovering the secret teleportation machine located in the secret places of this world. Now, you have the ability to open a gateway to a parallel world ruled by Skynet.

As you unlocked and started the machine, the majestic gateway is opened. Now the question is; *what kind of world awaits* ?

As you guys entered the gateway, you found yourselves in a fairly dark room with transparent pipes on the wall having blue, green and purple liquid inside them, lighting the room. You noticed that there is a high-tech door, "NO MAN or KINGS, ONLY MECHA" written on top of it. As you approached to the door, two little minions, spider-like machines, come front and open the door. As the door opens, you see that the room (you are in) floats in a worm-hole like environment and there are teleportation pads ahead.

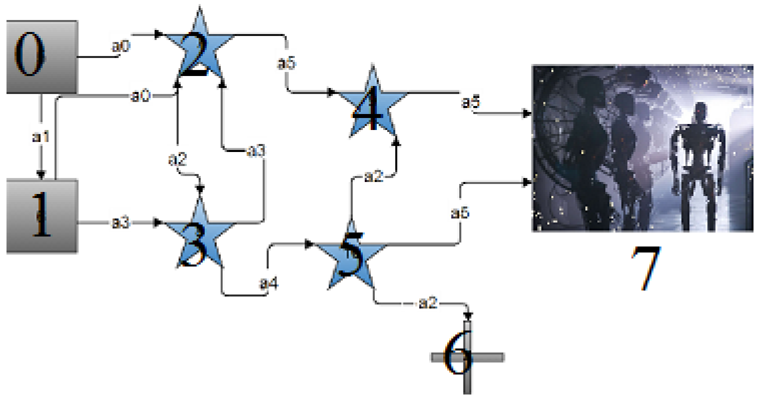
Once you try to step ahead to the pad in front, the pad buzzed you with electricity (remember the sign). Hopefully, the spider has a map to the environment and, of course, you know how to hack a mecha. Thus, you decided to hack one of the spider minions to send to the one of rooms, seen in the far distant. The idea is to pass one of the spiders to one of rooms ahead, so that the spiders can communicate and open portal for you to pass directly.

However, you sense that this is not the end of the story and there is more than meets the eye. You hack into the brain of the spider and gather some further details. After you pass to the other room you will reach the gate of my universe leading you to the environment where Skynet resides. You learn that first tele-pads work deterministically, and the tele-pads in my universe work in a probabilistic manner.

Here are the key points for the problem in figure below. In this graph there will be 4 different types of nodes:

Regular nodes, (star nodes)
Teleportation node, bridges between the universes (square node), initial starting point
Vortex nodes, similar to black holes, they will suck your robot, robot shouldn't move to these nodes (cross nodes)
Goal node, Skynet

Your task is to implement the **Value Iteration** algorithm for the given environment (and upload code



to spider):

```
Procedure Value_Iteration($S$,$A$,$P$,$R$,$\theta$)
    Inputs
        $S$ is the set of all states
        $A$ is the set of all actions
        $P$ is state transition function specifying $P(s'|s,a)$
        $R$ is a reward function $R(s,a,s')$
        $\theta$ a threshold, $\theta>0$
    Output
        $\pi[S]$ optimal policy
        $V[S]$ value function
    Local
        real array $V_k[S]$ is a sequence of value functions
        action array $\pi[S]$
    assign $V_0[S] \leftarrow 0$
    $k \leftarrow 0$
    repeat
        $k \leftarrow k+1$
        for each state $s$ do
            $V_k[s] = max_a \Sigma_{s'} P(s'|s,a) (R(s,a,s')+ \gamma V_{k-1}[s'])$
    until $k =  \theta$
    for each state $s$ do
        $\pi[s] = argmax_a \Sigma_{s'} P(s'|s,a) (R(s,a,s')+ \gamma V_k[s'])$
    return $\pi, V_k$
```

INPUT

Input will be given from file named *mdp.inp*, an example is provided at the end of this homework:

- The first line represents the type of nodes in the system ordered by their ID starting from 0: (V)ortex ; (T)eleport ; (S)tar ; (G)oal

- The second line represents discount factor to be used (given in the algorithm as γ)
- Next, the number of actions is given
- Then, for each node in my plane (except Skynet) the admissible set of actions are given in the form :

  $< node_i >< action\_id_0 >< action\_id_1 > \dots < action\_id_n >$
- After those lines, the effects of each action is represented in the form :

  action : $< action\_id_i >$

  $< node\_id_i >$

    $< reward >$

    $< node\_id_0 >< transition\_probability\_as\_percentile >$

    $< node\_id_1 >< transition\_probability\_as\_percentile >$

    ...

    $

  $< node\_id_j >$

    $< reward >$

    $< node\_id_0 >< transition\_probability\_as\_percentile >$

    $< node\_id_1 >< transition\_probability\_as\_percentile >$

    ...

    $

  #

  action : $< action\_id_j >$

    ... *(same here)*

EXECUTION FLOW & OUTPUT

After building the world, your program should print two lists ; where the first list represents value table while the other list represents your policy for the first iteration. After, your code repeatedly will wait for character *c* for continuing iteration (one step) then print the matrix and the lists again corresponding to that iteration, and wait for the character $ for convergence and hence termination.

mdp.cpp $> c$

$< V - table >$

$< Policy - list >$

mdp.cpp $> c$

.
.
.

mdp.cpp $>$ $

$< V - table >$

$< Policy - list >$

For V table, you should just output a column list, stating the $V(s)$ for each s. For the policy list you should use the following format;

- $< node\_id_0 > \quad < node\_id_1 >, < node\_id_2 >, \dots, < node\_id_n >$

where the first node represents the name of the node the latter represents the best states to go from the $< node\_id >$. For example, for an environment where the policy states that: go to 1 or 2 from state 0, go to 2 from 1, go to 3 from 2, then the output should be;

```
0   1,2
1   2
2   3
```

Example *mdp.inp* :

```
TTSSSSVG
0.1
6
0 0 1
1 0 3
2 2 5
3 3 4
4 5
5 2 5
action : 0
0
0
2 80
3 20
$
1
0
2 80
3 20
$
#
action : 1
0
0
1 100
$
#
action : 2
2
-10
3 50
0 30
1 20
$
5
20
4 40
6 40
3 20
$
#
action : 3
1
-10
3 90
0 10
$
3
100
```

```
2 50
0 50
$
#
action : 4
3
0
5 80
2 20
$
#
action : 5
2
20
4 90
0 10
$
4
200
7 80
5 20
$
5
200
7 80
3 20
$
#
E
```

1 Regulations

1. Your code should be in C++.
2. **Late Submission: Not Allowed**
3. **Cheating: We have zero tolerance policy for cheating**. People involved in cheating will be punished according to the university regulations.
4. **Newsgroup:** You must follow the newsgroup for discussions and possible updates on a daily basis.
5. **Evaluation:** The .cpp file will be checked for plagiarism automatically using "black-box" technique and manually by assistants, so make sure to obey the specifications.

2 Submission

Submission will be done via ODTUCLASS. Upload your code file named "mdp.cpp". Do not send any other files.