Multi Agent Systems

Homework Assignment 5

Dirk Hoekstra, Luisa Ebner, Philipp Lintl

October 17, 2018

6 Reinforcement Learning

6.1 Bellman equation

Given MDP→ modelbased; ∀ s ∈ S: $s \to a \to s'$ is unique.
Bellman equations:

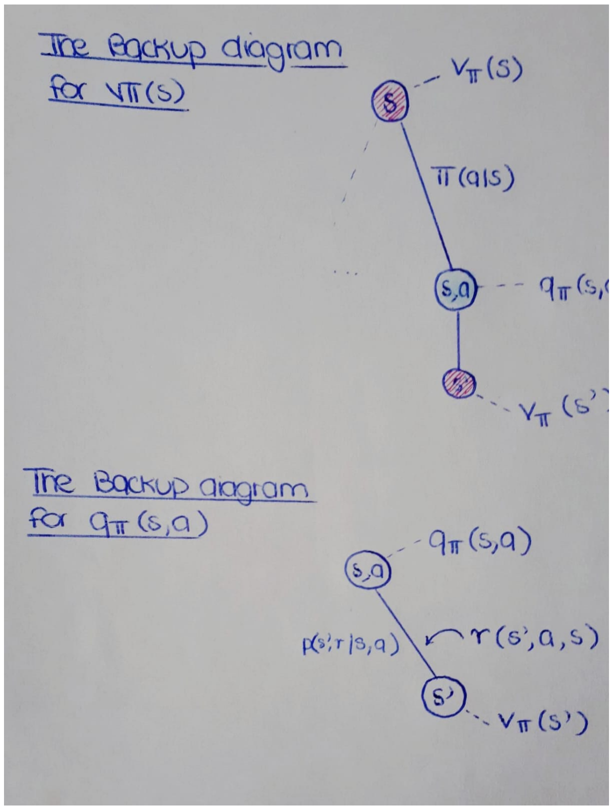
- state value function:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi(G_t | S_T = s) \\ &= \mathbb{E}_\pi(R_{t+1} + \gamma \cdot G_{t+1} | S_t = s) \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \\ &= \sum_a \pi(a|s) \cdot q_\pi(s, a) \end{aligned}$$

As both the policy π and the MDP $= \{S, A, R, P, \gamma\}$ are given, all $v_\pi(s)$ are computable as a system of $|S|$ linear equations in $|S|$ unknowns.

- action value function:

$$\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi(G_t|S_T = s, A_t = a) \\
&= \mathbb{E}_\pi(R_{t+1} + \gamma \cdot G_{t+1}|S_t = s, A_t = a) \\
&= \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] \\
&= \sum_{s',r} p(s',r|s,a)[r + \gamma \sum_{a'} \pi(a'|s') \cdot q_\pi(s',a')]
\end{aligned}$$



6.2 MDP 1

- Given MDP $= \{S, A, R, P, \gamma\}$
- State space $S = \{0, \ldots, n\}$ n even
- Action space A = {clockwise (c), anticlockwise (ac)}

- Reward space

$$r(0, 1, ac) = r(0, n, c) = 10$$
$$\text{all other } r(s', s, a) = 0$$

- circular state space
- odd number of nodes
- 0-node = absorbing, terminal state
- terminal reward $r_t = 10$ (transitioning to 0)
- all non terminal rewards $r_{NT} = 0$
- π equiprobable policy:

$$\pi(a|s) = \begin{cases} 0.5, & \text{a = clockwise (c).} \\ 0.5, & \text{a = anticlockwise (ac).} \end{cases}$$

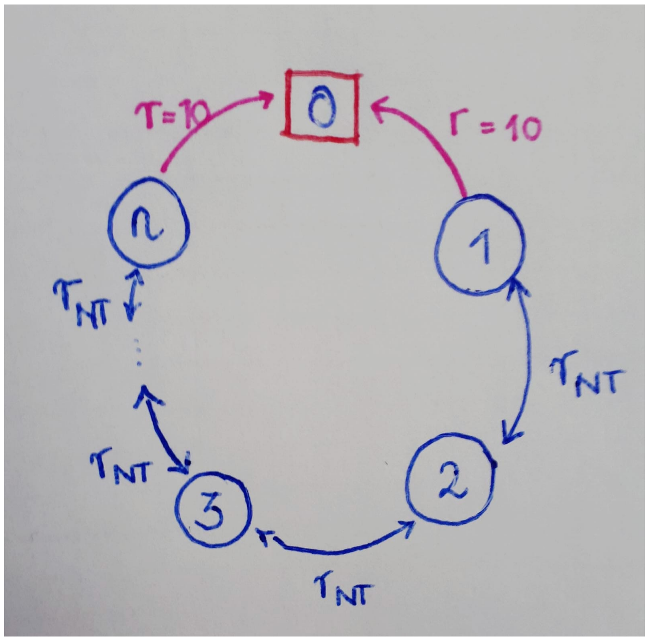


Figure 1: Grpahical representation of the given MDP.

6.2.1 What would be the corresponding values functions v_π and q_π?

(i) $v_\pi(s)$:

$$
\begin{aligned}
v_\pi(0) &= 0 \\
v_\pi(s) &= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r(s',s,a) + \gamma v_\pi(s')] \\
&= 0.5 \cdot \sum_{s',r} p(s',r|s,ac)[r(s',s,ac) + 1 \cdot v_\pi(s')] + 0.5 \cdot \sum_{s',r} p(s',r|s,c)[r(s',s,c) + 1 \cdot v_\pi(s')] \\
&= 0.5 \cdot (\sum_{s',r} p(s',r|s,ac) \cdot r(s',s,ac) + \sum_{s',r} p(s',r|s,ac) \cdot v_\pi(s') \\
&\quad + \sum_{s',r} p(s',r|s,c) \cdot r(s',s,c) + \sum_{s',r} p(s',r|s,c) \cdot v_\pi(s')) \\
&= 0.5 \cdot (r(s-1,s,ac) + v_\pi(s-1) + r(s+1,s,c) + v_\pi(s+1))
\end{aligned}
$$

Important steps in the transformations above:

- All rewards are 0 for $s = 2, \ldots, n-1$
- $p(s-1,r|s,ac) = 1$ and $p(s',r|s,ac) = 0, \forall s' \neq s-1$
- $p(s+1,r|s,c) = 1$ and $p(s',r|s,c) = 0, \forall s' \neq s+1$

Case analysis:

$$
v_\pi(s) = \begin{cases} 0, & \text{s = 0} \\ 0.5 \cdot (10 + v(2)), & \text{s = 1} \\ 0.5 \cdot (v(s-1) + v(s+1)), & \text{s = 2,...,n-1} \\ 0.5 \cdot (v(n-1) + 10), & \text{s = n.} \end{cases}
$$

(ii) $q_\pi(s,a)$:

$$
\begin{aligned}
q_\pi(s,a) &= \sum_{s',r} p(s',r|s,a)[r(s',s,a) + \gamma v_\pi(s')] \\
&= \sum_{s',r} p(s',r|s,a) \cdot r(s',s,a) + \sum_{s',r} p(s',r|s,a) \cdot v_\pi(s')
\end{aligned}
$$

Case analysis a:

$$q_\pi(s,a) = \begin{cases} r(s-1,s,ac) + v_\pi(s-1), & \text{a = ac} \\ r(s+1,s,ac) + v_\pi(s+1), & \text{a = c.} \end{cases}$$

Case analysis s and a:

$$q_\pi(s,a) = \begin{cases} 10, & \text{a = ac, s = 1} \\ v_\pi(s-1), & \text{a = ac, s = 2, \ldots, n.} \\ v_\pi(s+1), & \text{a = c, s = 1, \ldots, n-1} \\ 10, & \text{a = c, s = n} \end{cases}$$

6.2.2 What would be an optimal policy? Is this unique? What are the corresponding value functions $v^\star, q^\star$

The optimal policy emerges as:

$$\pi^\star(s) = \begin{cases} a = ac, & \forall s = \{1, \ldots, \frac{n}{2}\} \\ a = c, & \forall s = \{\frac{n}{2}+1, \ldots, n\} \end{cases}$$

The corresponding value function is:

$$\begin{aligned} v^\star(s) &= \mathbb{E}_\pi(G_t | S_t = s)) \\ &= \mathbb{E}_\pi(\sum_{k=1}^{T-t} R_{t+k} | S_t = s) \\ &= 10 \end{aligned}$$

The value action function $q_\pi(s,a)$ equals the value function of our exempary optimal policy $\pi^\star$, as the state s in $v_\pi(s)$ is sufficient to define a unique action a or in other words $\pi^\star$ is deterministic.
However, this policy is not unique as long as the agent arrives in the terminal state 0 after a finite number of steps. He can arbitrarily often ump between non terminal states. As the reward is 0, this does not cost anything and thus does not decrease $v_\pi(s)$. Another opimal polica would for example be: $\pi^{\star\star} = ac, \forall s = \{1, \ldots, n\}$. Both $\pi^\star$ and $\pi^{\star\star}$ are greedy with respect to $v_{\pi^\star}$ or $v_{\pi^\star\star}$ respectively. And both $v_{\pi^\star}$ and $v_{\pi^\star\star}$ are consistent with $\pi^\star$ and $\pi^{\star\star}$.

6.2.3 How would your answer change if for each non terminal step accrued a reward of $r_{NT} = -1$

Now, $\pi^\star$ would be the unique optimal policy.

$$v_\pi^\star(s) = \begin{cases} (s-1)\cdot(-1)+10, & \forall s = \{1,\dots,\frac{n}{2}\} \\ (n-s)\cdot(-1)+10, & \forall s = \{\frac{n}{2}+1,\dots,n\} \end{cases}$$

Every jump between the non-terminal states would decreas the cumulative expected reward. As such, $\pi^\star$ is the only optimal strategy.
$v^\star = q^\star$ (because the policy $\pi^\star$ is non-probabilistic in terms of which action to take in any state.

6.2.4 How would your answer change if $\gamma < 1$? (Assume $r_{NT} = 0$)

The optimal policy would again be $\pi^\star$:

$$\begin{aligned} v^\star(s) &= \mathbb{E}_{\pi^\star}(G_t|S_t = s) \\ &= \begin{cases} \gamma^{s-1}\cdot 10, & \forall s = \{1,\dots,\frac{n}{2}\} \\ \gamma^{n-s}\cdot 10, & \forall s = \{\frac{n}{2}+1,\dots,n\} \end{cases} \end{aligned}$$

Once again $q^\star(s,a) = v^\star(s)$ as $\pi^\star$ is deterministic. The optimal policy $\pi^\star$ is unique for $\gamma < 1$. It is the unique best strategy to go straight to the terminal node with the smallest number of steps in one direction. Any other pass would decrease the cumulative expected reward as it would increase the exponent of γ multiplied with the reward 10.

6.2.5 How would your answer change if the number of non-terminal states was odd?(Assume $r_{NT} = -1$ and $\gamma = 1$

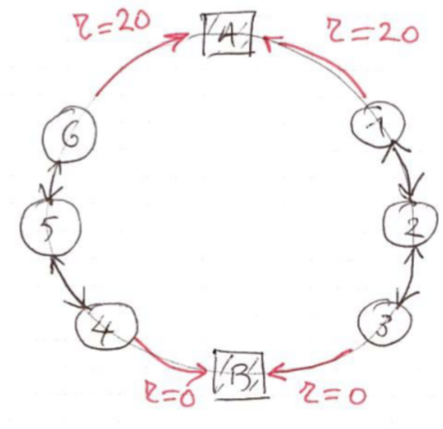
This case equals case 6.2.3, where $\pi^\star$ was the unique opimal policy aside from the fact that n is now odd. This change leaves us with exactly two optimal policies:

$$\pi_1^\star(s) = \begin{cases} a = ac, & \forall s = \{1,\dots,\lfloor\frac{n}{2}\rfloor\} \\ a = c, & \forall s = \{\lceil\frac{n}{2}\rceil,\dots,n\} \end{cases}$$

$$\pi_2^\star(s) = \begin{cases} a = ac, & \forall s = \{1,\dots,\lceil\frac{n}{2}\rceil\} \\ a = c, & \forall s = \{\lceil\frac{n}{2}\rceil+1,\dots,n\} \end{cases}$$

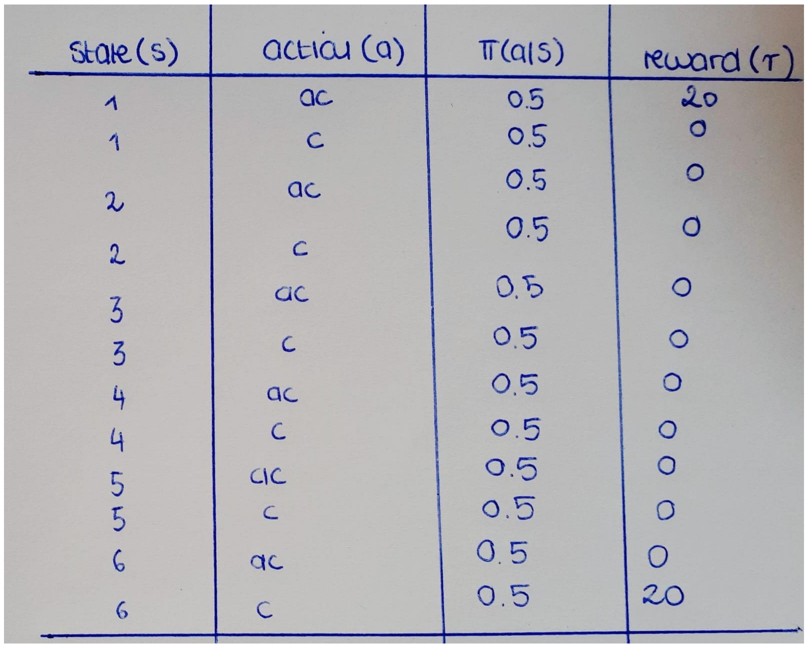
For the state $\lceil \frac{n}{2} \rceil$ it is equally optimal to go clockwise or anticlockwise. This leaves us with two optimal policies $\pi_1^\star$ and $\pi_2^\star$. The value functions $v_1^\star$ and $v_2^\star$ remains the same as $v^\star$ in 6.2.3.

6.3 MDP 2



6.3.1 What would be the values $v_\pi(1)...v_\pi(6)$ i.e. the long-term return for each of the six non-terminal states? Explain.

The chosen policy π is:



State (s)	action (a)	$\pi(a\|s)$	reward (r)
1	ac	0.5	20
1	c	0.5	0
2	ac	0.5	0
2	c	0.5	0
3	ac	0.5	0
3	c	0.5	0
4	ac	0.5	0
4	c	0.5	0
5	ac	0.5	0
5	c	0.5	0
6	ac	0.5	0
6	c	0.5	20

$$\begin{aligned} v_\pi(s) &= E_\pi(G_t|S_t = s) \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r(s, s', a) + v_\pi(s')\right] \end{aligned}$$

$$\begin{aligned} v_\pi(1) &= 0.5 \cdot r(1, 2, cl) + 0.5 v_\pi(2) + 0.5 \cdot r(1, A, ac) + 0.5 \cdot v_\pi(A) \\ &= 0.5 \cdot (v_\pi(2) + 20) \end{aligned}$$

$$\begin{aligned} v_\pi(2) &= 0.5 \cdot r(2, 3, cl) + 0.5 v_\pi(3) + 0.5 \cdot r(2, 1, ac) + 0.5 \cdot v_\pi(1) \\ &= 0.5(v_\pi(3) + v_\pi(1)) \end{aligned}$$

$$\begin{aligned} v_\pi(3) &= 0.5 r(3, B, cl) + 0.5 v_\pi(B) + 0.5 \cdot r(3, 2, ac) + 0.5 \cdot v_\pi(2) \\ &= 0.5 v_\pi(2) \end{aligned}$$

$$\begin{aligned} v_\pi(4) &= 0.5 r(4, 5, cl) + 0.5 v_\pi(5) + 0.5 \cdot r(4, B, ac) + 0.5 \cdot v_\pi(B) \\ &= 0.5 v_\pi(5) \end{aligned}$$

$$\begin{aligned} v_\pi(5) &= 0.5 r(5, 6, cl) + 0.5 v_\pi(6) + 0.5 \cdot r(5, 4, ac) + 0.5 \cdot v_\pi(4) \\ &= 0.5(v_\pi(6) + v_\pi(4)) \end{aligned}$$

$$\begin{aligned} v_\pi(6) &= 0.5 r(6, A, cl) + 0.5 v_\pi(A) + 0.5 \cdot r(6, 5, ac) + 0.5 \cdot v_\pi(5) \\ &= 0.5(20 + v_\pi(5)) \end{aligned}$$

6.3.2 For the setup defined above, what would be an optimal policy $\pi^\star$ and the corresponding optimal values $v^\star(1), \dots, v^\star(6)$. Is $\pi^\star$ unique?

$$\pi^\star(s) = \begin{cases} a = ac, & s = 1,2,3 \\ a = cl, & s = 4,5,6 \end{cases}$$
$$v^\star(s) = 20, \quad \forall s = 1, \dots, 6$$

This optimal policy is not unique, which we show with a counter example:

$$\pi^{\star alt}(s) = \begin{cases} a = ac, & s = 1 \\ \pi(ac|s) = \pi(cl|s) = 0.5, & s = 2 \\ a = ac, & s = 3 \\ a = cl, & s = 4 \\ \pi(ac|s) = \pi(cl|s) = 0.5, & s = 5 \\ a = cl, & s = 6 \end{cases}$$

$v^\star(s) = 20, \quad \forall s = 1, \dots, 6$ as this holds either it is optimal too and thus $\pi^\star$ is not unique.

6.3.3 Now assume $r_N T = -1$. Again, what would be an optimal policy $\pi^\star$ and the corresponding values $v^\star(1), \dots, v^\star(6)$. Is $\pi^\star$ unique?

The optimal policy again is $\pi^\star$ (see 6.2.2). This time $\pi^\star$ is unique: Any jumps between the non-terminal state would cause addditional 'costs', decreasing $v_{\pi^\star}(s)$. Correspondingly,

$$v^\star(s) = \begin{cases} 20, & s = \{1,6\} \\ 19, & s = \{2,5\} \\ 18, & s = \{3,4\} \end{cases}$$

6.3.4 Now assume $r_N T = -10$. Again, what would be an optimal policy $\pi^\star$ and the corresponding values $v^\star(1), \dots, v^\star(6)$. Is $\pi^\star$

In this case $\pi^\star$ is an optimal policy, but it is not unique. In state s=3 and s=4, the agent would be equally well off going clockwise or anticlockwise with a $v^\star(3) = v^\star(4) = 0$. As such, another optimal policy would be $\pi^{\star alt}$:

$$\pi^{\star alt} = \begin{cases} a = ac, & for s = \{1,2\} \\ a = cl, & for s = \{5,6\} \\ \pi(ac|s) = \pi(cl|s) = 0.5, & for s = \{3,4\} \end{cases}$$

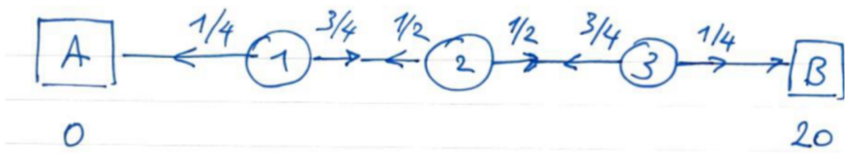
For either $\pi^\star$ and $\pi^{\star alt}$, the state value function $v^\star(s)$ is:

$$v^\star(s) = \begin{cases} 20, & s = \{1,6\} \\ 10, & s = \{2,5\} \\ 0, & s = \{3,4\} \end{cases}$$

6.4 MDP 3

6.4.1 Compute the state value function $v_\pi(s)$ under the policy π for all three states s = 1,2,3.

$state(s)$	$action(a)$	$\pi(a \mid s)$	$reward(r)$
1	L	1/4	0
1	R	3/4	−2
2	L	1/2	−2
2	R	1/2	−2
3	L	3/4	−2
3	R	1/4	20



$$\begin{aligned}
v_\pi(1) &= \frac{1}{4} \cdot 1(0+0) + \frac{3}{4}(-2 + v_\pi(2)) \\
&= -\frac{6}{4} + \frac{3}{4} v_\pi(2) \\
v_\pi(2) &= \frac{1}{2} \cdot (-2 + v_\pi(1)) + \frac{1}{2}(-2 + v_\pi(3)) \\
&= -1 + \frac{1}{2} v_\pi(1) - 1 + \frac{1}{2} v_\pi(3) \\
&= -2 + \frac{1}{2}(v_\pi(1) + v_\pi(3)) \\
v_\pi(3) &= \frac{3}{4} \cdot (-2 + v_\pi(2)) + \frac{1}{4}(20 + 0) \\
&= -1.5 + \frac{3}{4} v_\pi(2) + 5 \\
&= 3.5 + \frac{3}{4} v_\pi(2)
\end{aligned}$$

$\Rightarrow$ System of three linear equations of three unknowns.

$$\begin{aligned}
a &= -\frac{6}{4} + \frac{3}{4} \cdot b \\
b &= -2 + \frac{1}{2}(a + c) \\
c &= 3.5 + \frac{3}{4} \cdot b \\
\Rightarrow \quad & b = -4,\ c = 0.5,\ a = -4.5
\end{aligned}$$

$$v_\pi(s) = \begin{cases} -4.5, & s = 1 \\ -4, & s = 2 \\ 0.5, & s = 3 \end{cases}$$

6.4.2 Compute the state-action values $q_\pi(2, R)$ and $q_\pi(3, L)$

$$\begin{aligned}
q_\pi(2, R) &= r(3, 2, R) + v_\pi(3) \\
&= -2 + 0.5 = -1.5 \\
q_\pi(3, L) &= r(2, 3, L) + v_\pi(2) \\
&= -2 - 4 = -6
\end{aligned}$$

6.4.3 What would be an optimal policy $\pi^\star$ for this MDP? Is it unique?

The optimal policy $\pi^\star$ would be to go right with probability 1 in any state s=1,2,3.

state(s)	action(a)	$\pi(a\|s)$	reward(r)
1	R	1	-2
2	R	1	-2
3	R	1	20

With corresponding state value functions:

$$v_\pi^\star(s) = \begin{cases} 16, & s = 1 \\ 18, & s = 2 \\ 20, & s = 3 \end{cases}$$

This optimal strategy is unique, as going left always means a negative reward, then going right meams a negative reaward too. In order to get a positive reward at all, the agent needs to aim for terminal state B, which leaves this optimal policy.

6.5

6.5.1 Write a programme to check the shown v_π values (right grid in figure above), where π is the equi-probable policy and $\gamma = 0.9$)

In the Java program we created a representation of the grid world. Each state (cell on the grid) is represented by a `GridCell` object. For each `GridCell` object we call the `getRandomPolicyValue()` to calculate the value of this state, for the equi-probable random policy.

However each state calls the `getRandomPolicyValue()` for the state s' it produces. This would result in endless recursion, therefore we have a `MAX_DEPTH` variable in our `Main` class. This is the depth of the recursion. In other words, this is the amount of moves it will 'look ahead'.

When setting the `MAX_DEPTH=8` we get the following values for v

```
[v= 2.99][v= 8.33][v= 4.02][v= 4.68][v= 1.04]
[v= 1.26][v= 2.72][v= 1.97][v= 1.57][v= 0.24]
[v=-0.11][v= 0.60][v= 0.52][v= 0.22][v=-0.54]
[v=-0.98][v=-0.44][v=-0.33][v=-0.56][v=-1.14]
```

```
[v=-1.76][v=-1.24][v=-1.11][v=-1.28][v=-1.82]
```

This looks similar as the v_π values as seen in the assignment. However it is not exactly the same. We argue that this is because we only look 8 moves ahead. If we increase the `MAX_DEPTH` then the values should start to look more similar to those in the assignment.

When setting the `MAX_DEPTH=11` we get the following values for v

```
[v= 3.01][v= 8.23][v= 4.02][v= 4.72][v= 1.12]
[v= 1.33][v= 2.75][v= 2.01][v= 1.63][v= 0.31]
[v=-0.05][v= 0.62][v= 0.57][v= 0.24][v=-0.51]
[v=-1.00][v=-0.46][v=-0.37][v=-0.60][v=-1.19]
[v=-1.83][v=-1.32][v=-1.19][v=-1.38][v=-1.92]
```

The values are now even more similar which is good. However, we didn't increase the `MAX_DEPTH` value any more since our computer would then explode :)

Problem 6.5.2. Apply greedification to π to obtain a better policy. What are the corresponding v-values?

The algorithm (policy) will now pick the best action instead of a random one. We implement this by instead of doing $\sum_a p(a|s)(r + \gamma v(s'))$ we do $max_a(r + \gamma v(s'))$

In the program we set the `MAX_DEPTH=11` and call the `getGreedyPolicyValue()` for each GridCell (state).

```
[v= 1.86][v= 2.21][v= 1.86][v= 1.30][v= 1.06]
[v= 0.68][v= 1.86][v= 0.91][v= 1.06][v= 0.36]
[v= 0.30][v= 0.62][v= 0.45][v= 0.36][v= 0.07]
[v=-0.01][v= 0.14][v= 0.12][v= 0.05][v=-0.12]
[v=-0.22][v=-0.10][v=-0.08][v=-0.13][v=-0.26]
```

Problem 6.5.2. Change the setup in such a way that the MDP becomes episodic. More precisely, we assume that both A and B are terminal, absorbing states. Transition to A (B) yields an immediate reward of 10 (5) and finishes the game. We no longer need discounting, so we assume $\gamma = 1$. Re-assess the previous two bullets

To implement this we create a `boolean IS_EPISODIC` in the `Main` class. If this is true then state A and B will return an empty list in the `getPossibleActions()` method. Furthermore we change `DISCOUNT_FACTOR=1.0` in the `Main` class. And finally, we return a reward of 10 if transitioning into state A, and a reward of 5 if transitioning into state B.

For the random policy with a `MAX_DEPTH=11` we get the following result:

```
[v= 5.52][v= 0.00][v= 5.83][v= 0.00][v= 2.01]
[v= 3.26][v= 4.56][v= 3.58][v= 2.58][v= 1.11]
[v= 0.93][v= 1.63][v= 1.38][v =0.79][v=-0.19]
[v=-0.88][v=-0.31][v=-0.26][v=-0.68][v=-1.42]
[v=-2.24][v=-1.66][v=-1.55][v=-1.85][v=-2.53]
```

For the random greedy with a `MAX_DEPTH=11` we get the following result:

```
[v= 2.50][v= 0.00][v= 2.50][v= 0.00][v= 1.25]
[v= 1.36][v= 2.50][v= 1.46][v= 1.25][v= 0.64]
[v= 0.79][v= 1.13][v= 0.88][v= 0.64][v= 0.27]
[v= 0.22][v= 0.39][v= 0.35][v= 0.19][v=-0.05]
[v=-0.23][v=-0.07][v=-0.07][v=-0.16][v=-0.35]
```

As you can see the terminal states have a value of 0. Furthermore, the difference between the values are more evenly spread out with the greedy policy function.

6.6 Q-learning and SARSA

MDP with linear state space
→ agents ate located on a line → transitions can only be made towards the direct neighbors.
Action space = {Left(L), Right(R)}

$$\pi(a|s) \begin{cases} \frac{1}{2}, & fora = L \\ \frac{1}{2}, & fora = R \end{cases}$$

$\alpha = 0.9$; $\gamma = \frac{2}{3}$

state(s)	action(a)	$\pi(a\|s)$	reward(r)	q(s,a)
2	R	-1	5	
2	L	0	4	
3	R	1	6	
3	L	3		

6.6.1 Compute the next value for $q_2(2, R)$ under one Q-Learning iteration (i.e. only update this state action pair

$$
\begin{aligned}
Q(s,a) &\leftarrow Q(s,a) + \alpha \cdot (R + \gamma \cdot max_a Q(S',a) - Q(S,A)) \\
q_\pi(2,R) &\leftarrow q_\pi(2,R) + \alpha \cdot (r(a,R,3) + \gamma \cdot Q(3,R) - q_\pi(2,R)) \\
&5 + 0.9 \cdot (-1 + \frac{2}{3} \cdot 6 - 5) \\
&3.2
\end{aligned}
$$

6.6.2 Compute the next value for $q_2(2, R)$ under one under one iteration step of expected SARSA (using the equiprobable policy π)

$$
\begin{aligned}
Q(s,a) &\leftarrow Q(s,a) + \alpha \cdot (R + \gamma \cdot \sum_a \pi(a|s') q_\pi(S',a) - Q(S,A)) \\
q_\pi(2,R) &\leftarrow q_\pi(2,R) + \alpha \cdot (r(a,R,3) + \gamma \cdot \sum_a \pi(a|3) \cdot q_\pi(3|a) - q_\pi(2,R) \\
&5 + 0.9 \cdot (-1 + \frac{2}{3}(\frac{1}{2} \cdot 3 + \frac{1}{2} \cdot 6) - 5) \\
&2.3
\end{aligned}
$$