



Laporan Tugas 3

Machine Learning

Q-Learning

Disusun oleh :

Odia Pratama 1301154405

IF-39-13

Universitas Telkom

Bandung

2018

Analisis Masalah

Membangun sebuah sistem Q-Learning untuk menemukan *optimum policy* sehingga Agent yang berada pada posisi (1, 1) dapat menuju ke *goal state* (10, 10) dengan mendapatkan total reward yang maksimum. *Agent* dapat melakukan berbagai aksi diantaranya : N (*North*) atau bergerak ke atas, E (*East*) atau bergerak ke kanan, W (*West*) atau bergerak ke kiri, dan S (*South*) atau bergerak ke bawah. Dengan *grid world* sebagai berikut:

10	-1	-3	-5	-1	-3	-3	-5	-5	-1	100
9	-2	-1	-1	-4	-2	-5	-3	-5	-5	-5
8	-3	-4	-4	-1	-3	-5	-5	-4	-3	-5
7	-3	-5	-2	-5	-1	-4	-5	-1	-3	-4
6	-4	-3	-3	-2	-1	-1	-1	-4	-3	-4
5	-4	-2	-5	-2	-4	-5	-1	-2	-2	-4
4	-4	-3	-2	-3	-1	-3	-4	-3	-1	-3
3	-4	-2	-5	-4	-1	-4	-5	-5	-2	-4
2	-2	-1	-1	-4	-1	-3	-5	-1	-4	-1
1	-5	-3	-1	-2	-4	-3	-5	-2	-2	-2
	1	2	3	4	5	6	7	8	9	10

Desain



Evaluasi Hasil Eksperimen

Tabel R

```
UP : [ 0 0 0 0 0 0 0 0 0 0 0 -1 -3 -5 -1 -3 -3 -5 -5
      -1 100 -2 -1 -1 -4 -2 -5 -3 -5 -5 -3 -4 -4 -1 -3 -5
      -5 -4 -3 -5 -3 -5 -2 -5 -1 -4 -5 -1 -3 -4 -4 -3 -3 -2
      -1 -1 -1 -4 -3 -4 -4 -2 -5 -2 -4 -5 -1 -2 -2 -4 -4 -3
      -2 -3 -1 -3 -4 -3 -1 -3 -4 -2 -5 -4 -1 -4 -5 -5 -2 -4
      -2 -1 -1 -4 -1 -3 -5 -1 -4 -1]
DOWN : [ -3 -5 -1 -3 -3 -5 -5 -1 100 0 -1 -1 -4 -2 -5 -3 -5 -5
          -5 0 -4 -4 -1 -3 -5 -5 -4 -3 -5 0 -5 -2 -5 -1 -4 -5
          -1 -3 -4 0 -3 -3 -2 -1 -1 -1 -4 -3 -4 0 -2 -5 -2 -4
          -5 -1 -2 -2 -4 0 -3 -2 -3 -1 -3 -4 -3 -1 -3 0 -2 -5
          -4 -1 -4 -5 -5 -2 -4 0 -1 -1 -4 -1 -3 -5 -1 -4 -1 0
          -3 -1 -2 -4 -3 -5 -2 -2 -2 0]
LEFT : [ 0 -1 -3 -5 -1 -3 -3 -5 -5 -1 0 -2 -1 -1 -4 -2 -5 -3 -5 -5 0 -3 -4 -4
         -1 -3 -5 -5 -4 -3 0 -3 -5 -2 -5 -1 -4 -5 -1 -3 0 -4 -3 -3 -2 -1 -1 -1
         -4 -3 0 -4 -2 -5 -2 -4 -5 -1 -2 -2 0 -4 -3 -2 -3 -1 -3 -4 -3 -1 0 -4
         -2 -5 -4 -1 -4 -5 -5 -2 0 -2 -1 -1 -4 -1 -3 -5 -1 -4 0 -5 -3 -1 -2 -4
         -3 -5 -2 -2]
RIGHT : [-2 -1 -1 -4 -2 -5 -3 -5 -5 -5 -3 -4 -4 -1 -3 -5 -5 -4 -3 -5 -3 -5 -2 -5
         -1 -4 -5 -1 -3 -4 -4 -3 -3 -2 -1 -1 -1 -4 -3 -4 -4 -2 -5 -2 -4 -5 -1 -2
         -2 -4 -4 -3 -2 -3 -1 -3 -4 -3 -1 -3 -4 -2 -5 -4 -1 -4 -5 -5 -2 -4 -2 -1
         -1 -4 -1 -3 -5 -1 -4 -1 -5 -3 -1 -2 -4 -3 -5 -2 -2 -2 0 0 0 0 0 0
         0 0 0 0]
```

Tabel Q

```
UP : [ 0 0 0 0 0 0 0 0 0 0 0 -1 -7 -13 -1 -11 -3 -5 -13
      -1 496 -6 -1 -1 -4 -2 -5 -3 -13 -17 -5 -7 -16 -8 -1 -11 -5
      -5 -4 -3 -5 -11 -9 -6 -5 -1 -8 -13 -1 -3 -4 -8 -11 -11 -2
      -1 -1 -1 -8 -11 -4 -4 -2 -17 -2 -12 -17 -1 -6 -2 -4 -8 -3
      -6 -3 -1 -11 -4 -7 -1 -3 -16 0 0 0 0 0 0 0 0 0
      0 -1 -1 -8 -1 -3 -13 -1 -16 -1]
DOWN : [ 0 -5 -1 -3 -3 -5 -5 -1 496 0 0 -1 -4 -6 -5 -3 -13 -5
         -17 0 -4 -4 -1 -3 -5 -5 -4 -3 -5 0 -5 -6 -13 -1 -12 -5
         -1 -11 -4 396 -7 -3 -2 -1 -1 -1 -4 -11 -16 0 -6 -17 -6 -4
         -13 -1 -2 -2 -4 0 -11 -6 -7 -1 -3 -8 -11 -1 -3 0 -6 -13
         -12 -1 -4 -5 -5 -6 -12 0 -1 -1 -16 -1 -11 -17 -1 -8 -1 0
         -7 0 0 0 0 0 0 0 0 0 0]
LEFT : [ 0 -1 -7 -5 -1 -11 -3 -17 -5 -1 0 -2 -1 -1 -4 -2 -5 -3
         -5 -5 0 -11 -4 -12 -1 -3 -13 -5 392 -7 0 -3 -5 -2 -5 -1
         -12 -17 -1 -7 0 -8 -3 -11 -2 -1 -1 -1 -4 -11 0 -8 -2 -5
         -6 -12 -5 -1 -2 -6 0 -12 -3 -2 -3 -1 -7 -12 -3 -1 0 -16
         -2 -13 -16 -1 -8 -5 -5 -6 0 -6 -1 -1 -12 -1 -7 -5 -1 -16
         0 -5 -3 -1 -2 -4 -3 391 -2 0]
RIGHT : [-2 0 -1 -4 -6 -5 -3 -13 -5 -17 -3 0 -4 -1 -3 -5 -5 -4
         -3 -5 -3 0 -6 -13 -1 -12 -5 -1 -11 -4 392 0 -3 -2 -1 -1
         -1 -4 -11 -16 -4 0 -17 -6 -4 -13 -1 -2 -2 -4 -4 0 -6 -7
         -1 -3 -8 -11 -1 -3 -4 0 -13 -12 -1 -4 -5 -5 -6 -12 -2 0
         -1 -16 -1 -11 -17 -1 -8 -1 -5 0 -1 -6 -4 -3 -13 -2 -6 -2
         0 0 0 0 0 0 0 0 0 0 396]
```

Pergerakan *Agent* dan Reward maksimal yang didapat

```
[[2, 8, 'up'], [3, 9, 'right'], [3, 7, 'left'], [4, 8, 'down']]
[3, 9, 'right']
-4
[[2, 9, 'up'], [3, 8, 'left'], [4, 9, 'down']]
[4, 9, 'down']
-4
[[3, 9, 'up'], [4, 8, 'left'], [5, 9, 'down']]
[3, 9, 'up']
-4
[[2, 9, 'up'], [3, 8, 'left'], [4, 9, 'down']]
[2, 9, 'up']
-5
[[1, 9, 'up'], [2, 8, 'left'], [3, 9, 'down']]
[2, 8, 'left']
-3
[[1, 8, 'up'], [2, 9, 'right'], [2, 7, 'left'], [3, 8, 'down']]
[2, 9, 'right']
-5
[[1, 9, 'up'], [2, 8, 'left'], [3, 9, 'down']]
[3, 9, 'down']
-4
[[2, 9, 'up'], [3, 8, 'left'], [4, 9, 'down']]
[2, 9, 'up']
-5
[[1, 9, 'up'], [2, 8, 'left'], [3, 9, 'down']]
[1, 9, 'up']
-5
[[0, 9, 'up'], [1, 8, 'left'], [2, 9, 'down']]
[0, 9, 'up']
100
Goal State
Reward : -1876
Max Reward : 32
```

Pada percobaan running program menggunakan jumlah *episode* sebanyak 1000. Dapat disimpulkan bahwa jumlah *episode* dapat memengaruhi jumlah *reward* yang didapat, namun bukan berarti jika semakin banyak *episode* maka akan semakin bagus *reward* yang didapat. Jika semakin banyak *episode* maka komputasi program akan semakin membutuhkan waktu yang lama.

Dari program yang dijalankan akan menghasilkan tabel Q yang berisikan nilai *reward* yang telah dihitung untuk menentukan step selanjutnya dengan formula:

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

Pada program yang dibuat dengan menggunakan nilai gamma = 0,8. Sehingga hasil maksimal *reward* yang didapat adalah 32. Nilai tersebut didapat dengan pergerakan dari *initial state*(1,1) hingga *goal state*(10,10).