Pencarian *Optimum Policy* Menggunakan Metode *Reinforcement Learning (Q – Learning)*

Muh. Hasbi Abdul M – 1301160335

Program Studi S1 Informatika, Fakultas Informatika
Universitas Telkom Bandung
Jl. Telekomunikasi Dayeuhkolot, Bandung 40257

I. PENDAHULUAN

. *Reinforcement Learning* merupakan metode pembelajaran untuk memetakan segala keadaan terhadap aksi yang dilakukan untuk mendapatkan reward yang maksimum. Sedang kan Q – Learning merupakan suatu pengembangan dari *Temporal – Difference* (TD Control). Q – Learning juga merupakan sebuah terobosan yang penting dalam *Reinforcement Learning*. *Q – Learning* melakukan update terhadap tabel Q menggunakan rumus berikut :

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

Gambar 1 : Rumus untuk update tael Q

Dimana $Q(S_t,A_t)$ merupakan fungsi nilai aksi pada keaadan ke – t (S_t) dan aksi ke – t (A_t) sedangkan R_{t+1} merupakan *reward* yang didapat pada langkah yang akan dilakukan (ke-t+1) dan $\max Q(S_{t+1},a)$ merupakan nilai maksimum dari fungsi nilai aksi tujuan (ke-t+1) untuk suatu aksi a (tujuan). Parameter alpha merupakan learning rate yang menyatakan laju perubahan nilai sebelumnya yang akan digantikan nilai baru, alpha bernilai desimal antara 0 sampai 1, sedangkan gamma menentukan nilai reward dari langkah selanjutnya (masa depan).

Q – Learning memperbaharui nilai fungsi berdasar nilai aksi terbesar (maksimum) yang ada pada keadaan / langkah selanjutnya.

II. MASALAH

Menentukan jarak terdekat yang harus dilakukan Agent untuk mencapai *goal* (500) pada *grid world* yang terdapat pada data DataTugas3ML2019.txt sehingga mendapatkan reward se maksimal mungkin (*Optimum Policy)* .

Agent hanya bisa melakukan 4 aksi: Up, Right, Down, dan Left

III. ANALISIS PERMASALAHAN

Pada metode ini terdapat beberapa tahap yang harus dilakukan, yaitu

Sebelum memulai tahap pertama hal yang harus dilakukan adalah memasukkan file txt yang ada ke dalam program (tabel R) lalu memvisualisasikannya

-1.0	-2.0	-3.0	-2.0	-3.0	-3.0	-4.0	-1.0	-4.0	-2.0	-1.0	-2.0	-3.0	-3.0	500.0
-1.0	-3.0	-1.0	-2.0	-4.0	-1.0	-4.0	-1.0	-4.0	-2.0	-4.0	-2.0	-2.0	-2.0	-1.0
-4.0	-2.0	-1.0	-4.0	-2.0	-1.0	-2.0	-4.0	-2.0	-3.0	-2.0	-1.0	-2.0	-4.0	-4.0
-4.0	-2.0	-4.0	-1.0	-3.0	-2.0	-3.0	-2.0	-4.0	-2.0	-4.0	-1.0	-2.0	-4.0	-2.0
-4.0	-2.0	-2.0	-3.0	-2.0	-3.0	-1.0	-1.0	-4.0	-2.0	-1.0	-3.0	-4.0	-2.0	-4.0
-4.0	-3.0	-3.0	-4.0	-2.0	-3.0	-4.0	-2.0	-2.0	-1.0	-1.0	-2.0	-1.0	-2.0	-1.0
-2.0	-3.0	-2.0	-1.0	-1.0	-3.0	-2.0	-1.0	-4.0	-3.0	-1.0	-1.0	-2.0	-3.0	-3.0
-3.0	-1.0	-1.0	-4.0	-4.0	-3.0	-1.0	-2.0	-3.0	-1.0	-1.0	-4.0	-4.0	-3.0	-3.0
-3.0	-1.0	-4.0	-2.0	-3.0	-3.0	-1.0	-4.0	-4.0	-4.0	-2.0	-2.0	-2.0	-2.0	-1.0
-3.0	-4.0	-4.0	-2.0	-3.0	-4.0	-3.0	-3.0	-2.0	-2.0	-3.0	-4.0	-3.0	-4.0	-1.0
-3.0	-4.0	-1.0	-1.0	-1.0	-4.0	-4.0	-4.0	-4.0	-1.0	-2.0	-4.0	-2.0	-2.0	-1.0
-1.0	-3.0	-3.0	-3.0	-3.0	-3.0	-3.0	-3.0	-4.0	-1.0	-2.0	-4.0	-1.0	-2.0	-4.0
-2.0	-2.0	-1.0	-2.0	-2.0	-2.0	-4.0	-3.0	-1.0	-4.0	-1.0	-4.0	-2.0	-2.0	-2.0
-2.0	-1.0	-3.0	-1.0	-4.0	-4.0	-1.0	-3.0	-3.0	-1.0	-1.0	-2.0	-3.0	-4.0	-3.0
-2.0	-2.0	-1.0	-4.0	-4.0	-4.0	-2.0	-2.0	-3.0	-1.0	-2.0	-2.0	-1.0	-1.0	-3.0

Gambar 2 : Visualisasi Grid World

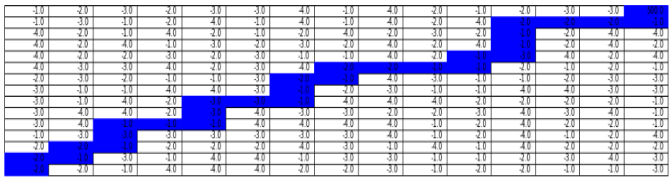
Tahap pertama yaitu menginisialisasi nilai – nilai yang diperlukan (iterasi, state, tabel R, tabel Q, jumlah episode, total reward, dsb)

Tahap kedua definisikan aksi apa yang dapat dilakukan di suatu state, aksi apa yang akan dilakukan dan definisikan juga rumus seperti yang pada gambar 1.

Tahap selanjutnya yaitu proses eksplorasi / *learning* pada kasus ini saya membatasi maksimum step yang dilakukan agar proses learning tidak terlalu lama. Yang dilakukan pada tahap ini adalah melepaskan agen untuk mencari jalan sendiri dengan cara melakukan looping sebanyak jumlah episode dan pada setiap episode dilakukan random posisi dimana agen akan memulai proses *learning*. Setelah agen turun ke dalam *grid world* maka lakukan perulangan kembali hingga agent mencapai *goal* atau mencapai batas maksimum step yang dilakukan dalam perulangan tersebut dilakukan proses random aksi yang dilakukan agent menggunakan fungsi yang telah didefinisikan sebelumnya dan sebelum berganti episode maka tabel Q akan di update sebagai hasil *learning* yang dilakukan agent.

Setelah tahap *learning* dilakukan maka tahap selanjutnya yaitu testing dimana Agent akan melakukan perjalanan dari posisi awal(pojok kiri bawah) ke *goal* berdasarkan tabel Q yang sudah didaptkan dari hasil *learning* sehingga agent dapat mendapatkan reward yang maksimum. Dan memvisualisasikan langkah yang dilakukan oleh agent dalam bentuk tabel.

Berikut merupakan hasil running dari kasus ini :



Gambar 3 : Hasil Optimum Policy yang dilakukan Agen

Berdasarkan tabel berikut maka aksi yang dilakukan yaitu (0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0) dimana 0 merupakan Up (ke atas) dan 1 adalah Right (ke kanan) dan reward maksimum yang didapat yaitu 455.

IV. REFERENSI

[1] Ardiansyah. “Implementasi Q – Learning dan Backpropagation pada Agen yang Memainkan Permainan Flappy Bird”.

[2] Slide Kuliah Pembelajaran Mesin Fakultas Informatika Telkom Umiversity “Reinforcement Learning”