Q-learning ile Yol Planlaması

1-) Giriş

Pekiştirmeli öğrenme (reinforcement learning), öznelerin (agent) bir görevi en yüksek kazançla tamamlayabilmek için hangi eylemleri gerçekleştirmeleri gerektigi ile ilgilenen bir makine öğrenmesi tekniğidir. Bu tür öğrenme algoritmaların girdisi öznelerin görev yapacakları farklı durumlardan oluşan bir ortam *S*, yapabilecekleri eylemler *A*, ortamdaki durumuna göre yapabilecekleri eylemleri belirleyen prensipler, bir durumdan diğer duruma geçtiklerinde elde edecekleri kazançtır.

Q-learning pekiştirmeli bir öğrenme algoritmasıdır. Ortam hakkında hiçbir şeyin bilinmediği durumlarda, Q-learning algoritması ortamı brute-force şeklinde, her ortam için olası tüm aksiyonları takip ederek, problem çözümü için en karlı yolu bulmaya çalışır. Q-learning algoritmasının girdileri kazanç matrisi olarak adlandırılan *R* matrisidir. Bu matrisin satır ve sütunları ortamları temsil etmekte, *R[i][j]* değeri ise *i* durumundan *j* durumuna geçtiğinde elde edilen anlık kazanç değeridir. Eğer *i* durumunda *j* durumuna bir geçiş yoksa *R[i][j]* değeri -1, geçis var ancak *j* durumu hedef durum değilse değeri 0, *j* hedef durum ise değeri kullanıcı tarafından belirlenen bir kazanç değeridir.

Q-learning algoritmasının çıktısı ise öğrenmenin kalitesini gösteren *Q* matrisidir. Q-learning iteratif bir algoritmadır ve tüm değerleri başlangıçta 0 olan *Q* matrisi optimal değerlere yakınsadığı da sona erer. Algoritma her iterasyonda rastgele bir durumdan öğrenmeye baslar, *A*'ya göre durum değiştirir ve *Q* matrisini günceller. *A*'ya göre hedef duruma ulaşıldığında iterasyon sona erer. *A*'ya göre bir durumdan birden fazla duruma geçis olabilir. Boyle bir durumda, olası geçişler den biri rastgele seçilir. Eğer seçilen durum hedef duruma ulaştırmıyorsa, durum rastgele olacak durum olarak belirlenir. Hedef duruma ulaşılana kadar iterasyon devam eder.

Q matrisi aşağıdaki formüle göre güncellenir:

$Q(durum, aksiyon) = R(durum, aksiyon)+\gamma \times Max\{Q(sonraki\ durumlar, tüm\ aksiyonlar)\}$ γ

ogrenme katsayısıdır ve 0 ile 1 arasında bir değer alır.

Aşağıdaki örnek Q-learning algoritmasını çalışmasını kısaca açıklamaktadır. Figür 1'de 6 durumdan oluşan bir ortam verilmistir.Durumlar arası geçişler de oklar ile göstermektedir. Okların üzerindeki değerler anlık kazançları göstermektedir. Buna göre

- Durumlar $S = \{A, B, C, D, E, F\}$
- Eylemler ve kazanç değerleri $A = \{A \xrightarrow{0} B, A \xrightarrow{0} D, B \xrightarrow{0} A, B \xrightarrow{100} C, B \xrightarrow{0} E, D \xrightarrow{0} A, D \xrightarrow{0} E, E \xrightarrow{0} D, E \xrightarrow{0} B, E \xrightarrow{0} F, F \xrightarrow{1-0} C\}$

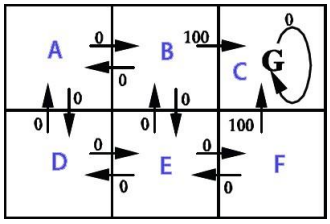

Figure 1: Ortam

Buna göre R matrisi ve Q matrisinin ilk hali aşağıdaki gibi olur.

$$R = \begin{pmatrix} -1 & 0 & -1 & 0 & -1 & -1 \\ 0 & -1 & 100 & -1 & 0 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & 0 & -1 \\ -1 & 0 & -1 & 0 & -1 & 0 \\ -1 & -1 & 100 & -1 & 0 & -1 \end{pmatrix} \qquad Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Başlangıc durumu A olarak seçilsin ve $\gamma = 0.8$ olsun. A durumundan olasi aksiyonlar B durumuna geçiş ve D durumuna geçişidir.. Rastgele olarak B durumuna geçiş seçilsin. Buna göre

$Q(0, 1) = R(0, 1) + 0.8 \times Max\{Q(1, 4), Q(1, 2)\}$

Buna göre Q(0,1) = 0 olarak güncellenir. *B* durumu hedef durum olmadığı için, durum *B*

olarak güncellenir ve iterasyon devam eder. B durumundan olası aksiyonlar *C* durumuna geçiş ve *E* durumuna geçişidir. Rastgele olarak *C* durumuna geçiş seçilsin. Buna göre

$$Q(1, 2) = R(1, 2) + 0.8 \times Max\{Q(2, 2)\}$$

Bu formülünün farklı türevleri olmakla birlikte ödevde bu formulu kullanınız.

Buna göre Q(1,2) = 100 olarak guncellenir ve *C* hedef durum olduğu için iterasyon sonlandırılır. Optimuma yakınsama olu¸sana kadar yeni iterasyonlar işletilir.

2-) Ödev

Burada robotun Q learning algoritması kullanarak engel sütunlarından kaçması ve beyaz alanlardan geçerek doğru yol alması gerekiyor. Aşağıdaki verilen matrisleri gerçek ortamdaki bir yol olarak düşünün. Robotumuz mavi kareden başlayıp kırmızı kutulara çarpmadan bitiş kısmına en kısa(maliyetle) yoldan ulaşırsa başarılı sayılacaktır.

Ajan, herhangi bir beyaz kareden başlayarak sağa, sola, aşağı, yukarı ve çapraz hareket edebilir. Atılan adımlar belirleyici olmalı ve engele çarpışmadıkça başarılı olur. Robot en son duvara geldiğinde robot sadece aşağı hareket ederek istenilen noktaya "37" gelecektir. Sonuç olarak robot başlangıç noktasından istenilen hedefe gelinceye kadar hiçbir engele çarpmadan ve en kısa yolu bularak ödülü alır. Rr: ajan[1,2, 8] karelere çarparsa işlem bitirir. Aksi takdirde, diğer her kareden herhangi bir işlem yapmak rs ödüllendirilir.

1	9	17	25	33
2	10	18	26	34
3	11	19	27	35
4	12	20	28	36
5	13	21	29	37
6	14	22	30	38
7	15	23	31	39
8	16	24	32	40

Örnek matris tablosu.

İndirim faktörü $\gamma = 0.9$, kırmızıya çarparsa -5 ödül puanı,yeşil bitiş noktasına +5, diğer geçişlere +3 ödül puanı olarak hesaplanacaktır.

A) Verilen 50 * 50'lİk matriste her bir kullanıcı kendine özgü engel oluşturup,matristeki değerleri random olarak atayacaktır.Bu matris değerlerini engel.txt dosyasına yazdırılacak.

B) Grafiksel ara yüzde belirlenen yollar, engeller ve duvarlar gösterilecektir.

C) Kullanıcı tarafından bir grafiksel arayüz tasarlanacak, bu ara yüzde ajan başlangıç noktası, hedef noktası istenecektir.

D) Herhangi bir başlangıç noktasından hedef noktaya ulaşıncaya kadar ajanın yaptığı kazançların/maliyetin(episode via cost) ve bölüm adım sayısının (episode via step) grafiği çizdirecek.

E) Sonuç olarak ise başlangıç karesinden hedef kareye giden en kısa yol grafiksel ara yüzde gösterilerek yol planı grafik üzerinde çizdirilecek.