Q-LEARNİNG İLE YOL PLANLAMASI

Vedat YILDIZ - Uğur Muhammed KARAYEL

Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

info@vedatyildiz.net, ugurkarayel41@gmail.com

ÖZET

50x50 boyutlarında hücresel gridlerden oluşan alan içerisinde, Q-learning algoritması kullanılarak en yüksek kazançla gerekli eylemleri gerçekleyen C# ile kodlanmış yazılımdır.

1. GİRİŞ

Başlangıç ve bitiş noktalarının dinamik olarak seçilmesi ve engellerin de rastgele olarak oluşturulması sonrasında Q-learning algoritması kullanılarak en kısa yolun tespit edilerek görsel arabirimde oluşturulması sağlanmaktadır. Buna ek olarak kazançların/maliyetin(episode via cost) ve bölüm adım sayısının (episode via step) grafiklerinin oluşturulması amaçlanmaktadır.

2. TEMEL BİLGİLER

Projeyi gerçeklerken yararlanılan teknolojiler :

- Görsel arabirim için Visual C# Windows form altyapısı, Microsoft Visual Studio 2019 Community.

- Görsel alanın oluşturulması için hücresel grid görünümünü destekleyen 3. parti DLL component,
(flexcell.dll www.grid2000.com)

3. YAKLAŞIMLAR

3.1 Hiyerarşik Yapı:

Projeye ait tüm altyapı, değişkenler ve metotlar, "Form1.cs" dosyası içerisinde **"proje"** namespace'i içerisinde tanımlanmıştır. Q-learning algoritmasına ait tüm işlemler fonksiyonlara bölünerek daha efektif bir kod tasarımı amaçlanmıştır.

3.2 Kullanılan yapı:

Program çalıştırıldığında, 50x50 boyutlarında karesel hücrelerden oluşan alanlar rastgele olarak engeller ile doldurulur ve engellerin bulunduğu hücreler kırmızı renk ile tanımlanır. Buna ek olarak tüm hücrelerin her birine 1 ile 9 arasında rastgele bir rakam atanarak kazançların/maliyetin **(episode via cost)** grafiğinin çıkartılmasında kullanılır. Ayrıca rastgele olarak elde edilmiş olan kırmızı hücreye sahip engeller, engel.txt içerisinde satır ve sütun numaraları olarak örn: (3,15,K) şeklinde gösterilir. Bu işlem, **engeltxt()** fonksiyonu ile gerçekleştirilir. Mouse ile başlangıç ve bitiş hücreleri click yapılarak seçim işlemi dinamik olarak gerçekleştirilir. Başla butonuna basılarak en kısa yol, görsel arabirim üzerine yeşil renk ile çizdirilir. Q-learning algoritması için aşağıdaki formül kullanılarak ilgili yapının oluşturulması sağlanmıştır :

$$NewQ(s,a) = Q(s,a) + \alpha[R(s,a) + \gamma max Q'(s',a') - Q(s,a)]$$

New Q value for that state and that action — Current Q value — Reward for taking that action at that state — Learning Rate — Discount rate — Maximum expected future reward given the new s' and all possible actions at that new state

SetQmatrix() fonksiyonu içerisinde, Learning rate değeri (Alpha) 0.7 olarak tanımlanmış ve discount faktor (indirim faktörü) yani gama değeri, ödev dökümanında belirtildiği üzere 0.9 olarak tanımlanmıştır. Bu değerler ışığında Q matrisi, ilgili iterasyon miktarı boyunca güncellenerek en kısa yola ait path, görsel arabirimde gösterilmiştir. Path belirlenmesi işlemi, **createpath()** fonksiyonu ile sağlanır. Ajanın o anda bulunduğu kutudan, hangi kutulara gidebileceğinin belirlenmesi işlemi, **ListAksiyom()** fonksiyonu kullanılarak belirlenir ve iki boyutlu dizi[] olarak fonksiyondan dönüş elde edilir. double **Maxdegersec()** fonksiyonu ile formülde bulunan **maxQ'** değeri elde edilir.

İterasyon işlemlerinin devamı esnasında Q matrisinin içeriğinin güncellenmesi işlemi, **SetQmatrix()** fonksiyonu içerisindeki

```
Qmatrix[durum, aksiyom] = Math.Round((Qmatrix[durum, aksiyom] + alpha * (Rmatrix[durum, aksiyom] + (gama * Maxdeger - Qmatrix[durum, aksiyom]))), 3, MidpointRounding.AwayFromZero);
```

Satırı ile gerçekleştirilir. **Set_Rmatrix()** fonksiyonu ile ajanın gidebileceği hücreler 0 (sıfır) olarak, engellerin bulunduğu hücreler ise -1 olarak **Rmatrix[,]** isimli iki boyutlu dizi içerisine atanır. Ayrıca hedef konumu gösteren hücreye 100 değeri atanır. **CreateBoard()** fonksiyonu ile flexcell komponentinden faydalanılarak görsel arabirime ait hücrelerin oluşturulması sağlanır.
Bölüm adım sayısı **(episode via step)** grafiğinin oluşturulabilmesi için gerekli olan veriler, SetQmatrix() fonksiyonu içerisindeki sayac değişkeni ile sağlanır ve bu değer, bir iterasyon içerisinde, ajanın hedefi bulmasına kadar geçen adım sayısından faydalanılarak elde edilir ve grafiğe dökülür. Açıklanan işlemler haricindeki geri kalan tüm fonksiyon ve kodlar yapının oluşturulmasına yardımcı olan diğer işlemlerdir(örn: grid1_Click(), btn_reset_Click() vs)

4. *KARŞILAŞILAN PROBLEMLER VE ÇÖZÜM YAKLAŞIMLARI*

50x50 boyutlu bir alan üzerinde çalışılması ve 2500 hücreden oluşan bir yapı bulunması nedeniyle başlangıçta hız konusunda sorunlar yaşanmış, ancak yapılan kod optimizasyonları(R , Q matrislerinin çıktılarının konsola yazdırılması işlemlerinin iptal edilmesi hız anlamında efektiflik sağlamıştır. Zaten R ve Q matrislerinin konsolda gösterilmesi işlemleri, Debug işlemleri yani hesaplamaların doğru olduğunun kontrolü için kullanılmıştır) sayesinde efektiflik elde edilmiştir. Bunun haricinde, başlangıçta görsel arabirime ait hücreler buton komponenti veya label komponeti ile oluşturulmak istenmiş ancak efektif bir sonuç vermemesi ve hız anlamında yavaşlık oluşması nedeniyle aynı işlemi yapabilecek C# destekli komponent araştırmasına gidilmiş ve flexcell isimli komponentin söz konusu ihtiyacı efektif şekilde karşıladığının anlaşılması üzerine bu komponentin kullanılmasına karar verilmiş ve tatmin edici sonuçlar elde edilmiştir.

5. YAZILIM GELİŞTİRME İÇİN HARCANAN SÜRELER (KİŞİ VE SAAT BAZINDA)

Toplam 5 ana aşama gerçekleştirilmiştir.

Aşama 1 : Görsel arabirim için uygun komponentin araştırılması.
(flexcell.dll : www.grid2000.com) 3 gün ve 2 kişi, bağımsız çalışmalar ile…

Aşama 2 : Flexcell komponentinin (www.grid2000.com) kullanımının user manuel üzerinden öğrenilmesi 3 gün ve 2 kişi.

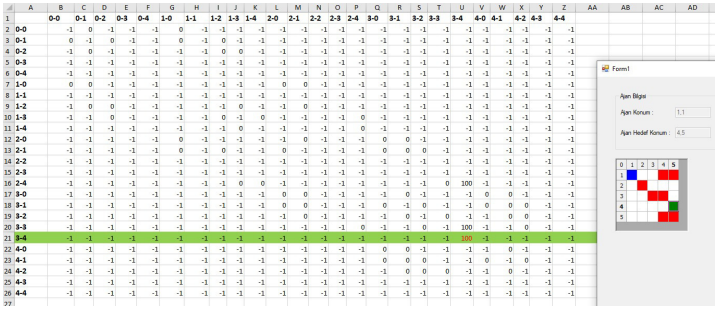
Aşama 3 : Q learnin algoritmasının uygulanması işlemi(R, Q matrisinin doldurulması ve Q matrisinden en kısa yol path'i elde edilmesi) (4 gün ve 2 kişi).

Aşama 4 : kazançların/maliyetin(episode via cost) ve bölüm adım sayısının (episode via step) grafiklerinin oluşturulması ve görsel arabirim üzerinde çizdirilmesi. (4 gün ve 2 kişi)

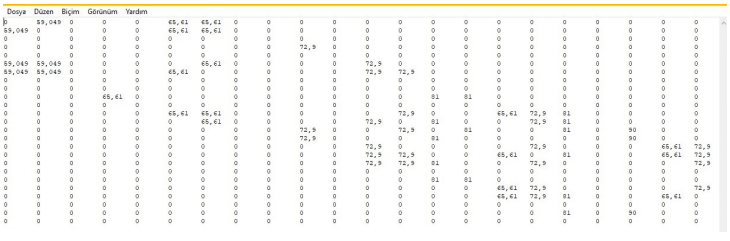
Aşama 5 : Son kontrollerin ve görsel düzenin sağlanarak beta testlerin yapılması (2 gün ve 2 kişi)

6. SONUÇLAR

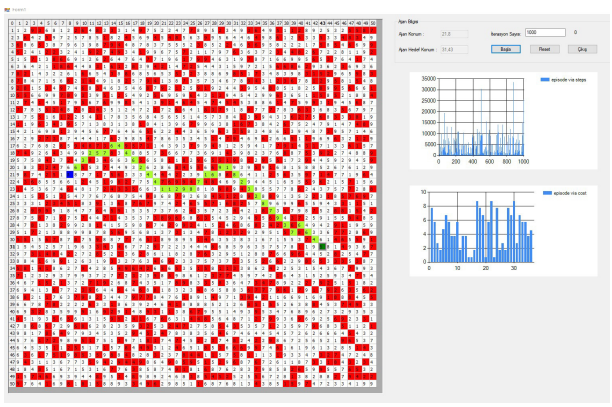
Programın tasarımı esnasında kullanıcı dostu bir tasarım oluşturulması ve kullanıcının minimum efor ile programı kullanabilmesi sağlanmaya çalışılmıştır. Ayrıca tüm isterlerin gerçekleştirilmesi, tüm işlemlerin, belirlenen kuralların eksiksiz olarak gerçekleştirilmesi amaçlanmıştır.



Şekil A : Örnek R matrisinin bir bölümü.



Şekil B : Örnek Q matrisinin bir bölümü.



Şekil C : Uygulanın çalıştırılması sonrasında ekran çıktısı.

7. KAYNAKÇA

[1] Grafiksel arabirim için kullanılan flexcell.dll dosyasına ait user manuel: http://www.grid2000.com

[2] Q-learngin algoritması çalışması-1 : https://www.mygreatlearning.com/blog/simplified-reinforcement-learning-q-learning/

[3] Q-learngin algoritması çalışması-2 : https://www.freecodecamp.org/news/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8/

[4] Episode via cost ve episode via steps işlemleri: https://ichi.pro/tr/q-learning-icin-yeni-baslayanlar-kilavuzu-215378165392700