PETRİ AĞLARINDA EN KISA YOL PROBLEMİNİN PEKİŞTİRMELİ ÖĞRENME METODU İLE ÇÖZÜLMESİ

Mustafa Müjdat ATANAK [1] **Hanife APAYDIN ÖZKAN [2]**

Fatih Onur HOCAOĞLU[3]

[1,2,3]Elektrik-Elektronik Mühendisliği Bölümü
Mühendislik-Mimarlık Fakültesi
Anadolu Üniversitesi, 26470, İki Eylül Kampusu, ESKİŞEHİR

[1] e-posta: mmatanak@anadolu.edu.tr [2] e-posta: hapaydin1@anadolu.edu.tr

[3]e-posta: fohocaoglu@anadolu.edu.tr

Anahtar sözcükler: Petri Ağları, Pekiştirmeli Öğrenme

ABSTRACT

Petri nets are frequently used for analysing, modelling and designing discrete event systems. If it is desired to reach a special state at a Petri net modelled manufacturing and otomation system in shortest possible time interval, the shortest path from the initial state to the desired state must be found.

In this study Reinforcement Learning, which is generally used to solve the so-called Markov decision problems, are considered to find shortest path for a desired state of a Petri net from an initial state and for this purpose some algorithms are developed.

1. GİRİŞ

Teknoloji ilerledikçe sanayide kullanılan sistemler büyümüş gelişmiş ve karmaşıklaşmıştır. Dolayısıyla, sistemlerin en uygun biçimde modellenmesi ve kontrolü gün geçtikçe önemini arttırmaktadır [1].

Günümüzde büyük ölçekli endüstriyel üretim süreçlerinin çeşitli amaçlar doğrultusunda izlenmesi ve denetimine ilişkin tasarımlar ve gerçeklemelerde kesikli olay sistemler kullanılmaya başlanmıştır.

Kesikli olay sistemlerin modellenmesinde kullanılmak üzere markov zincirleri, minimum-maksimum cebir modelleri, Petri ağları gibi çeşitli modelleme yöntemleri geliştirilmiştir. Petri ağı ile modellenen bir sistemin o anki durumu işaretleme vektörleri ile ifade edilir. Sistemin o anki durumu (işaretleme vektörü) ateşlenebilir bir geçişin ateşlenmesiyle değişir.

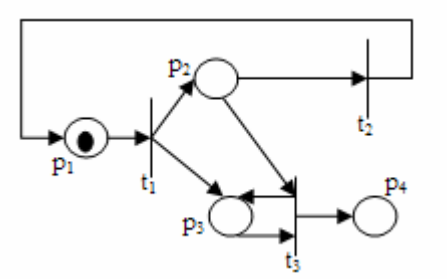
Üretim ve otomasyon sistemlerinde başlangıç durumundan istenen başka bir duruma mümkün olan en kısa yolla ulaşılması, zaman ve işgücü tasarrufu sağlar. Klasik yöntemlerle bu problemin çözülebilmesi için, ilk durumdan gidilebilecek tüm yolların bulunması ve daha sonra en kısa yolun bulunması gerekmektedir. Bu polinomsal zamanda çözülemeyecek bir problemdir. Bu çalışmada Petri ağında bir başlangıç durumundan istenen belirli bir duruma ulaşmak için kısa zamanda kısa bir yol bulan bir pekiştirmeli öğrenme metodu geliştirilmiştir.

2. PETRİ AĞLARI

Bir Petri ağı, daire şeklindeki yerler, çubuk, şeklindeki geçişler ve bunlar arasındaki bağlantıyı sağlayan yönlendirilmiş oklardan oluşur. Yer, bir işlemi veya bir kaynağın durumunu; geçiş ise, bir olayın ya da bir işlemin, başlangıcını ve/veya bitişini göstermektedir [2].

Bir Petri ağının yapısında yerler ve geçişlere ek olarak bulunduğu yerlerde işlemin yapılıyor olduğunu ifade eden "." şeklinde gösterilen belirtiler vardır. Belirtinin olup olmamasına veya sayısına göre ilgili yerlerin gösterdikleri kaynak hakkında bilgi sahibi olunur. Burada kaynak, sistemde gerçekleştirilmek istenen olaylar için gerekli olan koşulları, çalışan personeli, kullanılan makineleri vb. ifade etmektedir.

Bir Petri ağı G(P, T, N, O, m_0) şeklinde gösterilir. Burada T, geçişlerin kümesini; P, yerlerin kümesini göstermektedir ($P \cap T = \emptyset$; $P \cup T \neq \emptyset$). N: $P \times T \rightarrow \{0, 1\}$ girdi matrisidir. Bu matris yerlerden geçişlere doğru olan bağlantıların gösterildiği matristir ve elemanları, bir yerden bir geçişe bir ok ile bağlantı yapılmışsa 1, yapılmamışsa 0 alınarak oluşturulur. O: $P \times T \rightarrow \{0, 1\}$ çıktı matrisidir. Bu matris geçişlerden yerlere doğru olan bağlantıların gösterildiği matristir ve elemanları bir geçişten bir yere bir ok ile bağlantı yapılmışsa 1, yapılmamışsa 0 alınarak oluşturulur. m_0: $P \rightarrow D$ başlangıç işaretleme vektörüdür. Başlangıçta, belirtilerin Petri ağının yerlerindeki dağılımını gösterir.



Şekil 1 Örnek petri ağı

Bir Petri ağının çalışması geçişlerin ateşlenmesiyle belirtilerin yer veya yerlerden başka yer veya yerlere taşınması şeklinde olur. M bir işaretleme vektörünü göstermek üzere, bu işaretleme vektöründen sonra bir "t" geçişinin ateşlenebilirlik şartı: $M(p) \geq N(p, t)$, $\forall p \in P$ şeklindedir [3]. Burada, M(p), M işaretleme vektörünün p yerindeki belirti sayısını göstermektedir. Bir M işaretleme vektöründen bir t geçişinin ateşlenmesiyle bir M_1 işaretleme vektörü oluşuyorsa bu matematiksel olarak, $M_1 = M + O_t - N_t$ şeklinde ifade edilir. Burada, N_t, N matrisinin t geçişine karşılık gelen sütunu; O_t, O matrisinin t geçişine karşılık gelen sütunudur. Petri ağında peş peşe ateşlenen geçişlerin oluşturduğu diziye ateşleme dizisi veya geçiş dizisi denir, g ile gösterilir [3]. Bir Petri ağında bir M işaretleme vektöründen bir g geçiş dizisinin ateşlenmesi ile bir M_1 işaretleme vektörünün oluşması $M_1=p(M,g)$ fonksiyonu ile gösterilir. Ulaşılabilirlik kümesi, R(G, m_0), G ile gösterilen bir Petri ağında, m_0 işaretleme vektöründen ulaşılan tüm işaretleme vektörlerinin oluşturduğu kümedir [4].

3. PEKİŞTİRMELİ ÖĞRENME

Pekiştirmeli öğrenme, ortamda herhangi bir öğretici olmasını gerektirmeyen, sistemin dış dünya ile olan deneysel etkileşimini kullanan deneysel bir öğrenme metodudur. Pekiştirmeli öğrenmede öğrenen ve aktif öğrenme süreci olmak üzere iki nesne vardır. Her bir zaman diliminde öğrenici o anki durumunu değerlendirerek bir olay seçer ve ortamdan aldığı geri beslemeyi sisteme uygular [5], [6]. Öğrenenin amacı herhangi bir durumda en iyi davranışı seçebilecek duruma gelmektir. Ortamdan alınan geri besleme sinyalleri herhangi bir amacın başarılması veya başarısızlık gibi bilgiler olabilir. Öğrenicinin amacı daha önceden belirlenmiş olan bir performans ölçütünü optimize etmektir. Bu performans alınan geri bildirim sinyallerinin bir fonksiyonudur. En iyi olay dizisini belirlemek için her durumda yeterli sayıda güncelleme yapmak gerekir. İdeal olanı sonsuz sayıda güncelleme olduğu halde, bu mümkün olmadığından daha önceden yeterli olacağı varsayılmış sayıda güncelleme yapma yoluna gidilmektedir [7], [8].

Herhangi bir durumda öğrenen:
1-) Ortamı gözlemleyip, yöntem tarafından geri bildirilen sinyalleri algılar.
2-) O anki gözlemlere ve alınan geri bildirimlere dayanarak bir olay tanımlar ve gerçekleştirir.
3-) Yeni bir gözlem yapar ve deneyimlerini günceller.

4. GELİŞTİRİLEN ALGORİTMALAR

Bu çalışmada istenen durum M, ateşlenen geçiş sayısı ags, ve sistemin olaya uyguladığı geri besleme Gb ile ifade edilmek üzere, her bir p(M,t) olayı için bir Q(M,t) değeri (M işaretleme vektöründen t geçişinin ateşlenmesi ile ilgili Q değeri) aşağıdaki algoritmaya göre hesaplanmaktadır.

Algoritma-1

Tüm Q(M,t) değerlerini rastlantısal olarak ata
  for index=1:Döngü_sayısı
  ags=0
  While M ≠ M
    M:=şu_anki_durum
    Q(M,t) değerlerine göre yeni bir "t" geçişi seç.
    p(M,t)=M´ olayını gerçekleştir.
    ags ++
    If p(M,t)=istenen_durum
      Gb=λ^{ags}
    Else
      Gb=0
    End
    Q(M,t):=Q(M,t)+α[Gb+γmax_u·Q(M´,E(G,M´))
          -Q(M,t)]
        M=M´
  End
End

λ, α, γ; değerleri 0 ile 1 arasında olan sabitlerdir. Ateşlenen geçiş sayısı ne kadar fazla ise, sistem o kadar az geri besleme almış olmaktadır. Bu çalışmada en iyi sonuçlar λ=0.7, α=0.3 ve γ=0.1 değerleri ile alınmıştır. Ancak, bu sabitler, probleme bağlı olarak farklı olabilir.

Algoritma, Döngü_sayısı kadar çalıştırıldıktan sonra, en kısa yol bilgisi Q matrisinde saklanmış olur. Döngü_sayısı, problemin büyüklüğüne göre ayarlanabilmektedir. Örnek olarak verilen Petri ağı için Döngü_sayısı için 20 değeri yeterli olmaktadır.

Bu algoritmaya göre her bir p(M,t) olaylarına karşılık gelen Q(M,t) değerleri bulunduktan sonra ilk durumdan istenen duruma ulaşan en kısa yol aşağıdaki algoritma ile bulunur.

Algoritma-2

M=İlk durum
While M ≠ M
  t=argmax $_{t \in E(G,M)}$(Q(M,t))
  M´=p(M,t)
  M = M´
End

5. SONUÇ

Petri ağı ile modellenen bir sistemde istenilen belli bir duruma ulaşabilmek için geleneksel metodların kullanılması durumunda, ağın tüm ulaşılabilirlik kümesinin elde edilmesi, ilk durumdan istenen son duruma kadar tüm yolların değerlendirilmesi gerekir. Bu ise, polinomsal zamanda çözülemeyen bir problemdir. Büyük ölçekli sistemlerde, bu problemin bilgisayar ortamında çözülebilmesi için büyük miktarda hafıza ve çalışma zamanı gerekir. Bu çalışmada öngörülen yöntemde ise daha küçük bir hafıza kullanılarak daha az bir zamanda kabul edilebilir bir çözüme ulaşan sezgisel bir algoritma oluşturulmuştur. Her sezgisel algoritma gibi, amaç en iyi çözümü bulmak değil, kısa zamanda kabul edilebilir bir çözüm bulmaktır. Bu çalışmada istenilen sonuçlar, Algoritma 1' e göre yazılan programın çalıştırılması ile kısa bir zamanda elde edilmektedir.

Geliştirilen algoritma, değişik problem tiplerine çok rahat bir şekilde uyarlanabilir. Örneğin, algoritmada geri besleme fonksiyonunun tanımlandığı koşullar değiştirilerek, algoritma gezgin satıcı problemini çözebilecek hale getirilebilir.

KAYNAKLAR

[1] Aybar A., *Petri Ağlarında Örtüşmeli Ayrıştırma ve Genleştirme Kullanılarak Kontrolör Tasarımı*, Doktora Tezi, Anadolu Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir, 2001.

[2] Apaydın H., *Sınırsız Petri Ağları İçin Tersine Dönüşebilirliği Garanti Eden Sınır Vektörlerinin Bulunması*, Yüksek Lisans Tezi, Anadolu Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir, 2005.

[3] Aybar, A. and İftar, A. (2003b). *Decentralized controller design to enforce boundedness, liveness and reversibility in Petri nets*. In *Proceeding CD-ROM of the European Control Conference*, Cambridge, UK.

[4] Peterson, J.L, *Petri Net Theory and the modelling of systems* Englewood Cliffs, NJ : Prentice-Hall New Jersey,1981

[5] A. Gosavi. *Simulation-based Optimization: Parametric Optimization Techniques and reinforcement Learning*, Kluwer Academic Press, Boston, MA, 2003.

[6] R.S. Sutton and A.G. Barto, *Reinforcement Learning An Introduction*, The MIT Press, Cambridge (1998).

[7] K. Narendra and M.A.L. Thathachar, *Learning automata: an introduction*, Prentice Hall, Englewood Cliffs NJ (1989).

[8] D.P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, Belmont, MA (1995).