SAÜ. Fen Bil. Der. 17. Cilt, 3. Sayı, s. 307-314, 2013 SAU J. Sci. Vol 07, No 3, p. 307-314, 2013

Yarı Markov karar süreci problemlerinin çözümünde çok katmanlı yapay sinir ağlarıyla fonksiyon yaklaşımlı ödüllü öğrenme algoritması

Mustafa Ahmet Beyazıt Ocaktan[1*], Ufuk Kula[2]

[1*]Balıkesir Üniversitesi, Mühendislik Mimarlık Fakültesi, Endüstri Mühendisliği, Balıkesir
[2]Sakarya Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği, Sakarya

24.01.2013 Geliş/Received, 03.04.2013 Kabul/Accepted

ÖZET

Gerçek yaşam problemleri genellikle modellenmesi zor ve büyük ölçekli problemlerdir. Bu nedenle bu tür problemlerin klasik optimizasyon yöntemleriyle çözülmeleri çoğu zaman mümkün değildir. Bu makalede yarı Markov karar süreci olarak modellenebilen gerçek yaşam problemlerine yaklaşık optimal çözüm üreten çok katmanlı yapay sinir ağı yaklaşımlı bir ödüllü öğrenme algoritması geliştirilmiştir. Geliştirilen algoritmanın performansı küçük ölçekli sayısal bir örnek üzerinde ölçülmüş ve klasik ödüllü öğrenme algoritmasıyla kıyaslanmıştır. Sayısal denemelerden elde edilen sonuçlara göre, geliştirilen algoritmanın başarısında yapay sinir ağındaki gizli katman sayısı önemli bir etkendir ve uygun gizli katman sayısına sahip yapay sinir ağı yaklaşımlı ödüllü öğrenme algoritmasıyla elde edilen çözümün ortalama maliyeti, klasik ödüllü öğrenme algoritmasıyla elde edilen çözümün ortalama maliyetiyle yaklaşık aynıdır.

Anahtar Kelimeler: Markov/yarı Markov karar süreci, ödüllü öğrenme, çok katmanlı yapay sinir ağları

A reinforcement learning algorithm using multi-layer artificial neural networks for semi Markov decision problems

ABSTRACT

Real life problems are generally large-scale and difficult to model. Therefore, these problems can't be mostly solved by classical optimization methods. This paper presents a reinforcement learning algorithm using a multi-layer artificial neural network to find an approximate solution for large-scale semi Markov decision problems. Performance of the developed algorithm is measured and compared to the classical reinforcement algorithm on a small-scale numerical example. According to results of numerical examples, the number of hidden layer is the key success factor, and average cost of the solution generated by the developed algorithm is approximately equal to that generated by the classical reinforcement algorithm.

Keywords: Markov/semi Markov decision process, reinforcement learning, multi-layer artificial neural networks

* Sorumlu Yazar / Corresponding Author

1. GİRİŞ (INTRODUCTION)

Gerçek yaşam problemlerinin hemen hemen hepsi büyük ölçekli, karmaşık ve stokastik yapıya sahiptir, ve bu problemlerin önemli bir bölümü Markov/yarı Markov karar süreci (Markov /semi Markov decision process) olarak modellenebilir. Çok ürünlü bir stok sistemindeki yaklaşık optimal stok yenileme ve ürün kombinasyonu politikasının belirlenmesi [1], hava yolu endüstrisindeki koltuk sınıflandırma ve rezervasyon kotalarının belirlenmesi [2], otomatik kontrollü taşıma araçlarının optimal rotalarının belirlenmesi [3], farklı müşteri sınıflarına sahip kuyruk sistemlerinde optimal kontrol politikasının belirlenmesi [4] vb. problemler literatürde çokça karşılaşılan ve Markov/yarı Markov karar süreci olarak modellenebilen gerçek yaşam problemleridir. Bu tür problemler, çözüm uzayının çok büyük olması ve modellemedeki zorluklar nedeniyle geleneksel doğrusal olmayan programlama yada klasik dinamik programlama algoritmalarıyla çözülemez.

Günümüz bilgisayarlarının giderek artan hesaplama ve depolama gücü sayesinde simülasyon tabanlı optimizasyon teknikleriyle, stokastik yapıya sahip karmaşık problemlere yaklaşık optimal çözümler bulunabilmektedir. Son yıllarda oldukça popüler olan simülasyon tabanlı optimizasyon tekniklerinden ödüllü öğrenme (reinforcement learning), dinamik programlamaya dayalı tekniklerin geliştirilmesiyle ortaya çıkmıştır, ve dinamik programlamadaki gibi her bir durum için başlangıçtan sona uzun dönemli olası en küçük maliyeti veren bir değer fonksiyonu hesaplar. Ödüllü öğrenmeyle Markov/yarı Markov karar süreci olarak modellenebilen gerçek yaşam problemleri, yaklaşık optimal olarak çözülebilmektedir. Bertsekas ve Tsitsiklis [5], Sutton [6], Gosavi [7] ve Buşoniu ve arkadaşları [8]'dan ödüllü öğrenme algoritmaları ve ilgili literatür ayrıntılı olarak görülebilir.

Bu çalışmada, uzun dönemde ortalama maliyet kriteri altında yarı Markov karar süreci problemlerinin çözümüne odaklanılmıştır. Markov karar süreçlerinde zincirdeki herhangi bir geçişin süresi sabit bir birim olması gerekirken, yarı Markov karar süreçlerinde bu süre genel dağılımlı bir rassal değişkendir. Yarı Markov karar süreci problemlerinde zincirdeki her bir durum geçişinde sistemde harcanan süre de hesaba katıldığı için, gerçek yaşam problemlerini yarı Markov karar süreci olarak modellemek daha gerçekçidir. Markov ve yarı Markov karar süreci problemleri ve ilgili literatür Puterman [9]'da ayrıntılı olarak tartışılmıştır. Ortalama ödül kriteri altında yarı Markov karar süreci problemlerinin çözümü için Das ve arkadaşları [10] ve Gosavi [11] SMART (semi Markov average reward technique) ile Gosavi [7] Relaxed SMART (relaxed semi Markov average reward technique) algoritmalarını geliştirmiştir. Her iki algoritma da hem Markov ve hem de yarı Markov karar süreci problemlerinin çözümünde kullanılabilmektedir.

Makalede, yarı Markov karar süreci problemleri için SMART algoritmasından hareketle çok katmanlı yapay sinir ağı yaklaşımlı bir ödüllü öğrenme algoritması geliştirilmiştir. Yarı Markov karar süreci olarak modellenmiş çok ürünlü bir stok sistemindeki ürün kombinasyonu problemi geliştirilen algoritmayla çözülmüş, ve klasik ödüllü öğrenme algoritması çözümleriyle kıyaslanarak geliştirilen algoritmanın kullanılabilirliği araştırılmıştır. Makale, stokastik dinamik program olarak modellenebilen büyük ölçekli ve karmaşık gerçek yaşam problemlerinin çözüm yöntemlerine katkı sağlamayı amaçlamaktadır.

Makalenin 2. bölümünde ödüllü öğrenme ve yarı Markov karar süreci problemlerinin çözümü için literatürde yer alan SMART ödüllü öğrenme algoritması verilmiştir. 3. bölümde SMART algoritmasından hareketle geliştirilen yapay sinir ağı yaklaşımlı NeuroSMART algoritması yer almaktadır. 4. Bölümde sayısal uygulamalı örneklerle çok katmanlı yapay sinir ağıyla fonksiyon yaklaşımlı ödüllü öğrenme algoritmasının çözüm kalitesi araştırılmış, ve son bölümde ise elde edilen sonuçlar sunulmuştur.

2. ÖDÜLLÜ ÖĞRENME ALGORİTMASI (REINFORCEMENT LEARNING ALGORITHM)

İzleyen bölümde kısaca ödüllü öğrenme özetlenmiş ve ortalama maliyet yarı Markov karar süreci problemlerinin çözümü için literatürde yer alan SMART ödüllü öğrenme algoritması verilmiştir.

2.1. Ödüllü Öğrenme (Reinforcement Learning)

Ödüllü öğrenmedeki temel düşünce, bir simülatörde adım adım ilerleyerek durumlar için verilen iyi kararların ödüllendirilip kötü kararların cezalandırılması, ve dinamik programlamadaki gibi uzun dönemde her bir durum için olası en küçük maliyeti/en büyük kârı veren değer fonksiyonunun hesaplanmasıdır. Ancak, dinamik programlamada durumun değer fonksiyonu elemanlarıyla ilgilenilirken, ödüllü öğrenmede durum karar ikililerinin değer fonksiyonlarıyla ilgilenilir. Dinamik programlamada durumların değer fonksiyonlarının hesaplanabilmesi için geçiş olasılıklarının, geçiş ödüllerinin ve -yarı Markov karar süreci problemleri için- geçiş sürelerinin hesaplanması gerekir. Bu hesaplamaların yapılması rassal değişkenlerin çok katlı integrallerinin alınmasını gerektirir, ve problemin boyutu ve karmaşıklığı arttıkça

bu hesaplamaların yapılabilirliği imkansızlaşır. Ödüllü öğrenmede ise tüm hesaplamalar bir simülatör vasıtasıyla yapılmakta ve durum-karar ikililerinin değer fonksiyonları Q faktör formlarında arama tablolarında depolanmaktadır. Tüm bu modelin karmaşıklığı ve bir dereceye kadar boyut problemiyle başa çıkabilme beceresine rağmen ödüllü öğrenme algoritmaları, dinamik programlamanın aksine optimal çözüm bulmayı garanti etmez. Buna karşın dinamik programlamayla çözülemeyecek boyut ve karmaşıklıktaki problemlere, ödüllü öğrenmeyle yaklaşık optimal çözüm bulunabilir.

2.2. SMART Algoritması (SMART Algorithm)

Yarı Markov karar süreci problemlerinde zincirdeki geçiş süreleri Markov karar süreci problemlerindeki gibi sabit birim değil, genel dağılımlı bir rassal değişkendir. Bu nedenle yarı Markov karar süreci problemlerindeki geçiş süreleri birlikleştirme (uniformization) süreciyle sabit birime dönüştürülüp, Markov karar süreci problemlerinin çözümü için geliştirilmiş Q öğrenmesi (Q learning), Q-P öğrenmesi (Q-P learning) vb. ödüllü öğrenme algoritmalarıyla çözülebilir. Ancak, yarı Markov karar süreci problemlerinin birlikleştirme süreciyle tam olarak Markov karar sürecine dönüştürülmesi her durumda mümkün değildir. Ayrıca, birlikleştirme süreci için geçiş olasılıklarının da bilinmesi gerekir. Literatürde yer alan SMART algoritması, yarı Markov karar süreci problemlerini Markov karar süreci problemine dönüştürülmesine gerek kalmaksızın yaklaşık optimal olarak çözen bir ödüllü öğrenme algoritmasıdır. Algoritmanın motivasyon kaynağı denklem 1'de verilen Belman [12]'ın optimallik denklemidir:

$$v(\mathbf{x}) = \min_{a \in A(\mathbf{x})} \left[g(\mathbf{x}, a) - \rho^* y(\mathbf{x}, a) + \sum_{\mathbf{x}' \in \mathbf{x}} P(\mathbf{x}'|\mathbf{x}, a) v(\mathbf{x}') \right] \quad (1)$$

Belman'ın optimallik denkleminde $v(\mathbf{x})$, $\mathbf{x}$ durumunda bulunmanın değerini; $g(\mathbf{x}, a)$, $\mathbf{x}$ durumundayken a kararını almanın anlık maliyetini; $P(\mathbf{x}'|\mathbf{x}, a)$, $\mathbf{x}$ durumundayken a kararı seçildiğinde $\mathbf{x}'$ durumuna geçiş olasılığını; $y(\mathbf{x}, a)$, mevcut karar aşamasında $\mathbf{x}$ durumundayken a kararı seçildiğinde sonraki karar aşamasına kadar geçen beklenen süreyi; ρ^*, optimal politikanın ortalama maliyetini göstermektedir. Optimallik denkleminde yer alan optimal politikanın ortalama maliyeti ρ^*, başlangıçta bilinemez. Bu nedenle SMART algoritmasının dinamik programlamadaki optimallik denkleminin Q faktör versiyonu olan güncelleme denkleminde, optimal politikanın ortalama maliyetinin tahmincisi $\hat{\rho}$ kullanılır. SMART algoritmasındaki Q faktör değerlerinin güncelleme denklemi, denklem 2'de verilmiştir.

$$Q(i,a) \leftarrow (1-\alpha)Q(i,a) + \alpha \left[r(i,a,j) - \hat{\rho} t(i,a,j) + \min_{b \in A(j)} Q(j,b) \right] \quad (2)$$

Güncelleme denklemindeki α, öğrenme oranını göstermekte ve (i,a) durum-karar ikililerinin simülatördeki ziyaret sayılarından hesaplanmaktadır. Denklemdeki $r(i,a,j)$, a kararının etkisi altında i durumundan j durumuna geçilmesiyle oluşan anlık maliyeti ve $t(i,a,j)$ ise, aynı geçişte harcanan süreyi gösterir. Algoritmada $\hat{\rho}$, çözüm boyunca güncellenerek gittikçe ρ^*'a yaklaşır, ancak $\hat{\rho}$'nın algoritmada güncellenmesi için, simülatörde seçilen faaliyetin o ana kadar o durumdaki en iyi Q-faktör değerine sahip karar (greedy action) olması gerekir. Aksi durumda $\hat{\rho}$ güncellenmez.

V, durum karar ikililerinin ziyaret sayıları; S, durum uzayı; A, karar uzayı; k, simülasyonda algoritmanın yineleme sayısı; TS, toplam süre; TM, toplam maliyet; α, öğrenme oranı; B, öğrenme oranı sabiti; $(\hat{\rho})$, greedy politikanın maliyeti ve ϕ, seçilen kararın greedy olup olmadığını gösteren gösterge değişkeni olmak üzere SMART ödüllü öğrenme algoritmasının adımları aşağıda verilmiştir:

Adım 1: Tüm Q faktörleri, tüm (V) ziyaret sayılarını, simülasyonda yineleme sayısını (k), toplam süreyi (TS), toplam maliyeti (TM) ve greedy politikanın maliyetini $(\hat{\rho})$ sıfır (0) yap.

$l \in S, u \in A(l)$ için $Q(l,u) \leftarrow 0$, $V(l,u) \leftarrow 0$ ve $k \leftarrow 0, TS \leftarrow 0, TM \leftarrow 0, \hat{\rho} \leftarrow 0$.

1'den küçük öğrenme oranı sabiti B'yi belirle (B<1). Yeterince büyük, algoritmanın çalışacağı yineleme sayısı (k_{enb}) belirle. Sistemin simülasyonunu herhangi keyfi durumdan başlat.

Adım 2: Mevcut durum i olsun. Rastgele bir karar a seç. ϕ'i, denklem 3'e göre belirle.

$$\phi = \begin{cases} 0, & a \in \underset{u \in A(i)}{\operatorname{argmin}}\, Q(i,u) \text{ ise} \\ 1, & \text{Aksi halde} \end{cases} \quad (3)$$

Adım 3: Karar a'yı simüle et. Gidilen durum j olsun. $r(i,a,j)$, a kararı altında i durumundan j durumuna geçildiğinde kazanılan anlık maliyet ve $t(i,a,j)$, aynı geçiş için geçen süreyi göstermek üzere, (i,a) ikilisinin ziyaret sayısı $V(i,a)$ ile yineleme sayısı k'yı 1 arttır. Öğrenme oranı $\alpha = B/V(i,a)$'yı hesapla.

Adım 4a: Denklem 2'de verilen güncelleme denklemini kullanarak $Q(i,a)$'yı güncelle.

Adım 4b: Eğer seçilen a kararı greedy ise ($\phi = 0$), denklem 4'ü kullanarak toplam süre TS'yi ve denklem 5'i kullanarak toplam maliyet TM'yi güncelle.

$$TS \leftarrow TS + t(i,a,j) \quad (4)$$

$$TM \leftarrow TM + r(i,a,j) \quad (5)$$

Adım 4c: Greedy politikanın maliyeti $\hat{\rho}$'yı denklem 6'yı kullanarak güncelle.

$$\hat{\rho} = \frac{TM}{TS} \quad (6)$$

Adım 5: Eğer $k < k_{enb}$ ise adım 2'ye dön. Değilse adım 6'ya geç.

Adım 6: Her bir $l \in S$ için $d(l) \in arg \min_{b \in A(l)} Q(l,b)$ seç. Algoritmayla üretilen politika $\hat{d}$'dır. Dur.

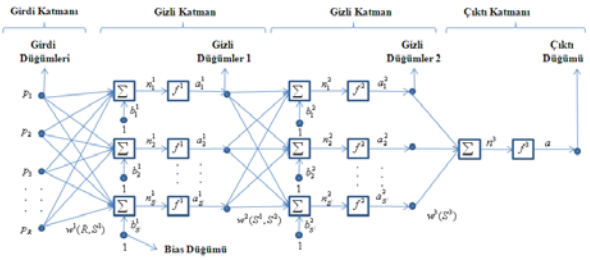
SMART algoritması, zincirdeki geçiş sürelerini sabit birime dönüştürmeksizin ortalama ödül problemlerine yaklaşık optimal çözüm üretir. Ayrıca Markov karar süreci problemleri, algoritmadaki geçiş süreleri sabit birim alınarak SMART algoritmasıyla da çözülebilir.

3. NEUROSMART ALGORİTMASI (NEUROSMART ALGORITHM)

Ödüllü öğrenmede her ne kadar geçiş olasılıkları, geçiş ödülleri ve yarı Markov karar süreci problemleri için geçiş süreleri simülatörde hesaplansa bile, büyük ölçekli problemlerde bu değerlerin bilgisayarda arama tablolarında depolanması ve her algoritma yinelemesinde güncellenmesi problemi hala ortadadır. Örneğin 1000 adet durum ve her bir durum için 3 kararın bulunduğu küçük ölçekli bir problemde bile simülatörden üretilip, depolanması gereken Q faktör değer matrisi $1000^3=10^9$ (bir milyar) elemanlıdır. Her bir durumdaki karar sayısının sadece bir artması durumunda ise depolanması gereken matrisin eleman sayısı 10^{12} (bir trilyon) olacaktır. Günümüz bilgisayarlarında bile hafıza problemi nedeniyle bu büyüklükte matrislerin depolanması mümkün değildir. Durum uzayının kesikli değil de sürekli olduğu problemlerde, depolama problemi daha da dramatik bir hal alır.

Boyut problemiyle başa çıkabilmek için Q faktör değerlerini tek tek arama tablolarına depolamak yerine, fonksiyon yaklaşım metotlarıyla tahmin etmek kullanışlıdır. Böylece az sayıda skaler sayıyla, milyarlarca durum karar ikilisinin Q faktör değerleri tahmin edilebilir. Ayrıca fonksiyon yaklaşımıyla, simülasyon süresince hiç yada çok az ziyaret edilecek durum-karar ikilileri için bile değer üretilebilir. Çok katmanlı yapay sinir ağlarının en önemli özelliği matematiksel olarak modellenemeyen fonksiyonlara yaklaşma yetenekleridir. Girdi katmanı birden fazla düğümden oluşan ve çıktı katmanında tek bir düğümün bulunduğu çok katmanlı yapay sinir ağları, ödüllü öğrenmede durum karar ikililerinin Q faktör fonksiyonuna yaklaşmak için başarıyla kullanılabilir.

Makalede, orta ve büyük ölçekli problemlerde karşılan çok sayıda durum-karar ikililerinin Q faktör değerlerinin depolanması gerekliliği ile baş edebilmek için, Q faktör fonksiyonuna çok katmanlı yapay sinir ağlarıyla yaklaşılmıştır. Şekil 1'de 1 girdi, 2 gizli ve 1 çıktı katmanlı yapay sinir ağı mimarisinde görüldüğü üzere yapay sinir ağları, verilen girdi-çıktı düğümleriyle ağın parametrelerini ayarlama temeline dayanır. Yapay sinir ağları, istenen bir sistem performansını gerçekleştirebilen bir kontrol biçimini öğrenebilmek için uygun ağırlık bağlantılarıyla kendi kendilerini biçimlendirir.



Şekil 1. Dört katmanlı yapay sinir ağı mimarisi (A four-layer neural network architecture)

Geliştirilen NeuroSMART ödüllü öğrenme algoritmasında kullanılan çok katmanlı yapay sinir ağının giriş katmanındaki düğümleri, (i,a) durum-karar ikilileri oluşturmaktadır. Çıktı katmanında ise tek bir düğüm bulunmakta ve bu düğüm girdi katmanında verilen (i,a) ikilisine karşılık üretilen $Q(i,a)$ faktör değeridir. Gizli katmanlar ise, doğrusal olmayan fonksiyonların yapay sinir ağlarıyla tahmin edilebilmesini sağlayan birimlerdir.

w, yapay sinir ağının ağırlıkları; q, algoritmada yapay sinir ağından tahmin edilen durum-karar ikilisi değerleri; S, durum uzayı; A, karar uzayı; k, simülasyonda algoritmanın yineleme sayısı; TS, toplam süre; TM, toplam maliyet; α, öğrenme oranı; $(\hat{\rho})$, greedy politikanın maliyeti ve ϕ, seçilen kararın greedy olup olmadığını gösteren gösterge değişkeni olmak üzere geliştirilen NeuroSMART algoritmasının adımları aşağıda verilmiştir:

Adım 1: Yapay sinir ağının başlangıç ağırlıklarını w(.) rassal olarak belirle.

Adım 2: Tüm q faktör değerlerini, simülasyonda algoritmanın yinelenme sayısını (k), toplam süreyi (TS), toplam maliyeti (TM) ve greedy politikanın maliyetini $\hat{\rho}$ sıfır (0) yap.

$l \in S, u \in A(l)$ için $q(l,u)$ ve $k \leftarrow 0, TS \leftarrow 0, TM \leftarrow 0, \hat{\rho} \leftarrow 0$.

Sabit öğrenme oranı $\alpha, (0 < \alpha < 1)$ belirle. Yeterince büyük yineleme sayısı k_{enb} belirle. Sistemin simülasyonu herhangi keyfi durumdan başlat.

Adım 3: Mevcut durum i olsun. Tüm $u \in A(i)$ kararları için, çok katmanlı yapay sinir ağını kullanarak $q(i,u)$ değerlerini tahmin et.

Adım 4: Rastgele bir karar a seç. ϕ'i, denklem 7'ye göre belirle. Karar a'yı simüle et, geçilen durum j olsun. Simülatörde a kararı altında i durumundan j durumuna geçildiğinde oluşan anlık maliyet $r(i,a,j)$'yi ve aynı geçiş için geçen süre $t(i,a,j)$'yi hesapla.

$$\phi = \begin{cases} 0, & a \in \underset{u \in A(i)}{\operatorname{argmin}}\, q(i,u) \text{ ise} \\ 1, & \text{Aksi halde} \end{cases} \quad (7)$$

Adım 5a: Yapay sinir ağını kullanarak j durumundaki tüm kararlar için $q(j,b), b \in A(j)$ değerlerini belirle. Denklem 8'i kullanarak j durumunun o anki q faktör değeri q_{next}'i hesapla. Denklem 9'u kullanarak (i,a) durum-karar ikilisinin q faktör değeri $q(i,a)$'yı güncelle.

$$q_{next} \leftarrow arg \min_{b \in A(j)} q(j,b) \quad (8)$$

$$q(i,a) \leftarrow (1-\alpha)q(i,a) + \alpha[r(i,a,j) - \hat{\rho}t(i,a,j) + q_{next}] \quad (9)$$

Adım 5b: Eğer seçilen a kararı greedy ($\phi = 0$) ise, denklem 10'u kullanarak toplam süre TS'yi ve denklem 11'i kullanarak toplam maliyet TM'yi güncelle.

$$TS \leftarrow TS + t(i,a,j) \quad (10)$$

$$TM \leftarrow TM + r(i,a,j) \quad (11)$$

Adım 5c: Denklem 12'yi kullanarak greedy politikanın maliyeti $\hat{\rho}$'yı güncelle.

$$\hat{\rho} = \frac{TM}{TS} \quad (12)$$

Adım 6: Eğer $k < k_{enb}$ ise güncellenen $q(i,a)$'yı kullanarak yapay sinir ağını güncelle ve adım 3'e dön. Değilse adım 7'ye geç.

Adım 7: Dur. Öğrenilen politika yapay sinir ağının ağırlıklarında depolanır.

Algoritmanın başlangıcında q faktör değerlerini tahmin etmek için, adım 1'de rassal olarak belirlenen w(.) başlangıç ağırlıklarına sahip yapay sinir ağı kullanılır. Adım 6'da ise algoritma ilerledikçe o ana kadar açığa çıkmış q değerleri yapay sinir ağı girdisi olarak alınır ve artımlı bir şekilde yapay sinir ağının ağırlıkları güncellenerek, istenen durumların q faktör değerlerinin tahmininde kullanılır. Bir durumla ilgili kararı belirlemek için, bu durum için izin verilen kararların yapay sinir ağı çıktıları bulunur. Çıktısı en iyi olan karar, ilgili durum için en iyi politikadır.

4. SAYISAL ÖRNEK (NUMERICAL EXAMPLE)

Geliştirilen çok katmanlı yapay sinir ağı yaklaşımlı NeuroSMART algoritmasının çözüm kalitesinin belirlenebilmesi için, durum karar sayısı nispeten küçük olan çok ürünlü bir stok sisteminde yaklaşık optimal ürün kombinasyonun belirlenmesi problemi tasarlanmıştır. Problem hem SMART hem de NeuroSMART algoritmalarıyla çözülmüş ve daha büyük ölçekli problemler için NeuroSMART algoritmasının genelleştirilip, genelleştirilemeyeceği araştırılmıştır.

Sayısal örnekteki ürün talep süreci, müşteri gelişleri ortalaması $\lambda = 5$ müşteri/saat ve bireysel ürün talepleri Tablo 1'de verilen kesikli dağılıma uyan birleşik Poisson sürecidir. Örnekte, çok ürünlü stok sisteminde 4 çeşit ürün bulunduğu ve ürün çeşitlerinin fiyatlarının sırasıyla 10 TL, 20 TL, 50 TL ve 100 TL olduğu varsayılmıştır. Stok sistemindeki tüm ürün çeşitleri birbirileri ile tam ikamedir, ve ürün ikamesi firma kaynaklı olarak hem aşağı ve hem de yukarı yönlü gerçekleşebilmektedir. Stoklarda yer alan tüm ürün çeşitleri için stokta bulundurma maliyeti h=0,01 TL/TL/saat; birincil sipariş maliyeti K=10 TL/sipariş; ikincil sipariş maliyeti k=5 TL/siparişe eklenen ürün çeşidi sayısı; talebin stoktan karşılanamaması maliyeti w=10 TL/siparişi stoktan karşılanamayan müşteri sayısı ve müşterinin kaçırdığı faydanın ceza maliyeti p=0,5 TL/kayıp müşteri faydası olarak tanımlanmıştır. Belirlenen stok politikasının tüm ürün çeşitleri için en az 0,90 ikinci tip hizmet düzeyini sağlaması gerektiği ve verilen siparişlerin tedarik süresinin sabit 0,5 saat olduğu kabul edilmiştir.

Tablo 1. Müşterilerin bireysel ürün talebi olasılık dağılımı (Probability distribution of individual customer demand)

d	1	2	3	4	5	6	7	8	9	10
$f_D(d)$	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10	0,10

Müşterilerin taleplerine verilecek kararlar ve bu kararlara göre müşterilerin kazanacağı varsayılan faydalar tablo 2'de verilmiştir.

Tablo 2. Müşterilerin kesikli ve özdeş fayda fonksiyonu (The discrete and identical utility function of customers)

	Fayda									
Talep / Karar	10	20	30	40	50	60	70	80	90	100
1	100	80	100	60	50	50	70	50	90	30
2		100	80	100	100	50	80	100	60	50
3				80	90	100	100	90	100	60
4					60	80	70	60	80	100
5						60	60	70	70	60
6							50	50	60	90
7								40	50	60
8									30	40
9										30
10										20
11										10

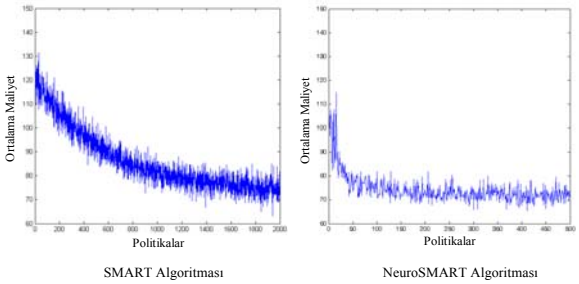
Stok sistemindeki her bir ürün çeşidi için (S,c,s) eşgüdümlü stok kontrol politikası uygulamaktadır. Ürün çeşitlerinin (S,c,s) kontrol parametreleri tablo 3'te yer almaktadır.

Tablo 3. Stok sistemindeki (S,c,s,) stok kontrol politikası parametreleri (Control parameters of the (S,c,s) inventory policy)

Stok Parametreleri / Ürün Çeşitleri	S	c	s
Ürün 1	12	4	0
Ürün 2	11	5	2
Ürün 3	8	3	1
Ürün 4	3	2	0

Olası müşteri taleplerindeki karar sayıları ve uygulanan stok politikasındaki stokların tamamlanacağı üst düzeyler (12, 11, 8, 3) dikkate alındığında, tasarlanan bu çok küçük ölçekli problem için bile durum-karar ikilisi sayısı 155.232'dir.

Sayısal örnek için yineleme sayısı k=200.000 müşteri alınmış ve SMART ödüllü öğrenme algoritması Matlab'de kodlanarak çözülmüştür. Karar seçiminde keşif stratejisi uygulanmış ve SMART algoritmasındaki simülatöre gelen her 100 müşteride bir, o zamana kadar üretilen ürün kombinasyonu politikasının birim zamandaki ortalama maliyeti hesaplanmıştır. Üretilen kombinasyon politikalarının birim zamandaki ortalama maliyet yakınsama grafiği şekil 2'de verilmiştir.



SMART Algoritması NeuroSMART Algoritması

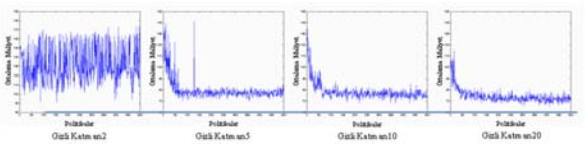
Şekil 2. SMART ve NeuroSMART algoritmaları ortalama maliyet yakınsama grafikleri (Convergence graphics of SMART and NeuroSMART algorithms for average reward)

Sayısal örneğin NeuroSMART algoritmasıyla çözümünde Q faktör değerlerinin tahmini için kullanılan yapay sinir ağı 1 girdi katmanı, 20 gizli katman, 1 çıktı katmanından oluşturulmuş ve gizli katmanlarda tan-sigmoid transfer fonksiyonu, çıkış fonksiyonunda ise purelin doğrusal transfer fonksiyonu kullanılmıştır. Algoritmanın simülatörde yineleme sayısı k=50.000 müşteri olarak alınmış ve her 100 müşteride bir, simülatörden elde edilen verilerle yapay sinir ağı eğitilerek Q faktör fonksiyona yaklaşılıp, o ana kadarki mevcut politikanın birim zamandaki ortalama maliyeti hesaplanmıştır. Çok katmanlı yapay sinir ağı için MATLAB 2008b Neural Fitting Toolbox kullanılmış ve yapay sinir ağı için yazılan matlab kodu, NeuroSMART algoritması kodunun içine gömülmüştür. Yapay sinir ağının eğitimi için Levenberg-Marquardt backpropagation algoritması seçilmiş ve yapay sinir ağının her eğitiminde simülatörden elde edilen verilerin rassal olarak % 60'ı eğitim seti, % 20'si geçerlilik seti ve % 20'si test seti olarak alınmıştır. Q faktör fonksiyonuna 20 gizli katmanlı yapay sinir ağıyla yaklaşıldığı NeuroSMART ödüllü öğrenme algoritması kullanılarak elde edilen ürün kombinasyonu politikalarının birim zamandaki ortalama maliyet yakınsama grafiği şekil 2'deki gibi elde edilmiştir. Algoritmanın durduğu andaki son yapay sinir ağının ortalama hata karesi (mean square error) 0.6687 ve oluşan regresyon denkleminin R korelasyon katsayısı 0.9335 olarak ölçülmüştür. Bu sonuçlara göre algoritma sonlandığında elde edilen yapay sinir ağı, herhangi bir durumla ilgili kararın belirlenmesinde başarıyla kullanılabilir.

Şekil 2'den görüldüğü üzere 20 gizli katmanlı NeuroSMART algoritmasının ürettiği politikaların birim zaman başına ortalama maliyeti ile, SMART algoritmasının ürettiği politikaların birim zaman başına ortalama maliyeti hemen hemen aynıdır. SMART ödüllü öğrenme algoritmasıyla üretilen politikalarının birim zamandaki ortalama maliyeti 1200'ncü politika ya da diğer bir deyişle 120.000'nci müşteriden itibaren yaklaşık olarak 72 değerine yakınsamaktadır.

NeuroSMART algoritması ise SMART algoritmasına göre çok daha hızlı bir şekilde yaklaşık 50'nci politikada yada diğer deyişle simülatördeki 5000'nci müşteriden itibaren 72 değerine yakınsamaktadır.

Sayısal problemin 200.000 müşteri simule edilerek SMART algoritmasıyla çözüm süresi yaklaşık 112 saat sürmüştür. Aynı problemin 50.000 müşteri simule edilerek 20 gizli katmanlı yapay sinir ağı yaklaşımlı NeuroSMART algoritmasıyla çözümü ise, yaklaşık 27 saattir. Yapay sinir ağının yapısındaki gizli katman sayısı, NeuroSMART algoritmasının performansını etkileyen önemli bir etkendir. Gizli katman sayısı arttıkça ağın eğitim süresi uzadığı için algoritmanın performansı süre olarak kötüleşmekte, ancak tahmin olarak iyileşmektedir. Bu nedenle tahmin kalitesi açısından yakın sonuçlar verdiği sürece, yapay sinir ağında mümkün olduğunca az sayıda gizli katman seçilmesi mantıklıdır. Sayısal örnek, gizli katman sayısı 2, 5, 10 verilerek yeniden çözülmüş ve sonuçlar şekil 3'te verilmiştir.



Şekil 3. 2, 5, 10, 20 gizli katmanlı NeuroSMART algoritması yakınsama grafikleri (Convergence graphics of 2, 5, 10, 20-layer NeuroSMART algorithms)

Sayısal örneğin yapay sinir ağında 2 gizli katman kullanılarak NeuroSMART algoritmasıyla çözümü yaklaşık 11 saat sürmüştür. Ancak, şekil 3'ten görüldüğü üzere iki gizli katmana sahip yapay sinir ağıyla Q faktör fonksiyonunu tahmin ederek uygulanan NeuroSMART algoritması sonucunda elde edilen politikaların maliyetleri çok geniş bir aralıkta (100-170) salınmakta ve bir limit maliyete yakınsamamaktadır. Bu nedenle elde edilen çözüm kullanılabilir değildir. Farklı ölçekte problemlerde de yapay sinir ağında iki gizli katman kullanıldığı durumlarda benzer sonuçlar gözlenmiştir. Sonuç olarak, NeuroSMART algoritmasının kullanımında gizli katman sayısının iki olarak belirlenmesi uygun değildir.

Sayısal örneğin yapay sinir ağında 5 gizli katman kullanılarak çözümü yaklaşık 13 saat sürmüştür. Yine şekil 3'te görüldüğü üzere 5 gizli katmana sahip yapay sinir ağıyla Q faktör fonksiyonunu tahmin ederek uygulanan NeuroSMART algoritması sonucunda elde edilen politikaların ortalama maliyetleri, 50'nci yinelemeden sonra 75-80 bandında yakınsamaktadır. Ancak, 47'nci ve 129'ncu politikaların ortalama maliyetleri genel salınımdan çok büyük sapma göstermiştir. Her ne kadar simulasyonun ilerleyen yinelemelerindeki politikaların ortalama maliyet hesaplamalarında yakınsama sağlansa da, 47'nci ve 129'ncu politikaların ortalama maliyetlerindeki büyük sapmalar, Q faktör fonksiyonun yapay sinir ağıyla tahmininde bu iki noktada yakınsamadan uzaklaşılabileceğini göstermektedir. Aynı örneğin, aynı parametrelerle tekrar çözümlerinde genel salınımdan aşırı sapma görülen politikalardan sonra birim zamandaki ortalama maliyetin bir limit değere yakınsamaktan uzaklaştığı çözümlerle de karşılaşılmıştır. Bu nedenle yapay sinir ağında 5 gizli katmanın kullanıldığı NeuroSMART algoritması belirtilen örnekte bazen yakınsama sağlamasına rağmen, gürbüz değildir.

Sayısal örneğin yapay sinir ağında 10 gizli katman kullanılarak çözümü yaklaşık 21 saat sürmüştür. Şekil 3'te görüldüğü üzere 50'nci politika hesaplamasından sonra elde edilen politikaların ortalama maliyetleri 75-80 bantında yakınsamakta ve genel salınımdan büyük sapmalar gözlenmemektedir. Aynı örneğin, aynı parametrelerle tekrar çözümlerinde de 10 gizli katmanlı yapay sinir ağı yaklaşımlı NeuroSMART algoritması benzer sonuçlar üretmiştir. Bu nedenle 10 gizli katmanın kullanıldığı NeuroSMART algoritması oldukça gürbüzdür.

Sonuç olarak, verilen sayısal örnekte NeuroSMART algoritmasındaki yapay sinir ağında 10 ve 20 gizli katman kullanıldığında çok iyi sonuçlar alınmış, 2 gizli katman bulunduğunda birim zamandaki ortalama maliyetlerde yakınsama sağlanamamış ve 5 gizli katman bulunduğunda ise yakınsama sağlanmasına rağmen fonksiyon yaklaşımı güvenilir değildir.

5. SONUÇ (CONCLUSION)

Makalede gerçek yaşam problemlerinin çözümünde karşılaşılan boyut ve karmaşıklık problemiyle başedebilmek için, Q faktöre çok katmanlı yapay sinir ağıyla yaklaşılan NeuroSMART ödüllü öğrenme algoritması geliştirilmiştir. Tasarlanan küçük ölçekli sayısal örnek hem SMART, hem de farklı sayılarda gizli katmanlara sahip yapay sinir ağı yaklaşımlı NeuroSMART algoritmasıyla çözülmüştür. Elde edilen sonuçlara göre, geliştirilen NeuroSMART algoritmasının performansında gizli katman sayısının önemli bir etken olduğu görülmüştür. Ele alınan problemin karmaşıklığına da bağlı olarak yapay sinir ağında az sayıda gizli katman kullanımlarında, geliştirilen algoritmanın performansının gürbüz olmadığı görülmüştür. Nispeten fazla gizli katmanlı yapay sinir ağları kullanımı durumunda ise, NeuroSMART

algoritmasının SMART algoritması kadar iyi sonuçlar ürettiği; ancak, gizli katman sayısı arttıkça hesaplama süresinin giderek uzadığı görülmektedir. Her ne kadar algoritmanın çalışma süresi uzun da olsa, her bir durum için tekrar tekrar hesaplama yapılması gerekmemekte, geliştirilen algoritmayı kullanan benzetim sona erdiğinde elde edilen yapay sinir ağı genelleştirilerek, ağın ağırlıklarından istenen durum-karar ikilisi simüle edilerek uygulanacak politika kolaylıkla belirlenebilmektedir. Bu nedenle dinamik programlama ve ödüllü öğrenmeyle çözülemeyecek büyüklükte durum-karar sayısına sahip Markov ve yarı Markov karar süreci problemlerinde, uygun gizli katman sayısına sahip çok katmanlı yapay sinir ağı yaklaşımlı NeuroSMART ödüllü öğrenme algoritması, yaklaşık optimal çözüm veren iyi bir yöntemdir.

KAYNAKLAR (REFERENCES)

[1] Ocaktan, M.A.B. (2012) İkame ürün dağıtım ağlarında stok optimizasyonu ve optimal dağıtım politikaları, Doktora Tezi, Sakarya Üniversitesi, Endüstri Mühendisliği A.B.D.

[2] Gosavi, A. (2004) 'A reinforcement learning algorithm based on policy iteration for average reward: empirical results with yield management and convergence analysis', Machine Learning, vol. 55, pp. 5-29.

[3] Tadepalli, P. and Ok, D. (1998) 'Model based average reward reinforcement learning algorithms', Artificial Intelligence, vol. 100, pp. 177-224.

[4] Shioyama, T. (1991) 'Optimal control of a queuing network system with two types of customers', European Journal of Operational Research, vol. 52, pp. 361-372.

[5] Dimitri P.B. and Tsitsiklis, J. (1996) Neuro-dynamic programming, Athena Scientific.

[6] Sutton, R. and Barto, A.G. (1998) Reinforcement learning, Cambridge: The MIT Press.

[7] Gosavi,G. (2003) Simulation-based optimization, Kluwer Academic Publishers.

[8] Buşoniu, L., Babuska, R., Schutter, B.D. and Ernst, D. (2010) Reinforcement learning and dynamic programming using function approximators, CRC Press.

[9] Puterman, M.L. (1994) Markov decision processes: discrete stochastic dynamic programming, John Wiley & Sons.

[10] Das, T.K., Gosavi, A., Mahadevan, S. and Marchalleck, N. (1999) 'Solving semi-Markov decisions problems using average reward reinforcement learning', Management Science, vol. 45, no. 4, pp. 560-574.

[11] Gosavi, A. (2004) 'Reinforcement learning for long-run average cost', European Journal of Operational Research, vol.155, no. 3, pp. 654-674.

[12] Bellman, R. (1954) 'The theory of dynamic programming', Bulletin of American Society, vol. 60, pp. 503-516.