T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ÇOKLU ETMEN TAKVİYELİ ÖĞRENMEYE VERİ MADENCİLİĞİ TABANLI YENİ YAKLAŞIMLAR

Mehmet KAYA

DOKTORA TEZİ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ELAZIĞ, 2003

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ÇOKLU ETMEN TAKVİYELİ ÖĞRENMEYE VERİ MADENCİLİĞİ TABANLI YENİ YAKLAŞIMLAR

Mehmet KAYA

DOKTORA TEZİ
ELEKTRİK – ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Bu tez, 25.04.2003 tarihinde aşağıda belirtilen jüri tarafından oybirliği /~~oyçokluğu~~ ile başarılı /~~başarısız~~ olarak değerlendirilmiştir.

Danışman:	Doç. Dr. Ahmet ARSLAN	.....
Üye:	Doç Dr. Saadetdin HERDEM	.....
Üye:	Doç Dr. Erhan AKIN	.....
Üye:	Doç Dr. Z. Hakan AKPOLAT	.....
Üye:	Yrd. Doç. Dr. Yetkin TATAR	.....

Bu tezin kabulü, Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..20/04/2003... tarih ve .....16/4..... sayılı kararıyla onaylanmıştır.

TEŞEKKÜR

Doktora tezi, uzun ve yorucu bir çalışmanın ürünüdür. Her tezde olduğu gibi, bu çalışma boyunca da bir çok kişinin emeği geçmiştir. Bu nedenle, öncelikle akademik hayata atılmamda bana önderlik eden, bilgi ve tecrübelerini benimle paylaşan ve akademik hayatım boyunca hep kullanacağım değerli öğütler veren danışman hocam Sayın Doç. Dr. Ahmet ARSLAN'a şükranlarımı sunarım.

İkinci danışmanım olarak her zaman yanımda olan, tez boyunca beni hep doğruya yönlendiren ve sık sık kendilerine giderek değerli bilgiler aldığım Sayın Prof. Dr. Faruk POLAT'a (ODTÜ Bilgisayar Mühendisliği Bölümü) teşekkür ederim.

Kanada'da kaldığım 1 yıl boyunca benden hiç bir yardımı esirgemeyen, tezin bu aşamaya gelmesinde büyük bir payı olan, akıl hocam Sayın Prof. Dr. Reda ALHAJJ'a (Calgary Üniversitesi, Bilgisayar Bilimleri Bölümü) teşekkür ederim.

Doktora çalışması boyunca bana her zaman destek olan Sayın Prof. Dr. Tamer ÖZSU'ya (Waterloo Üniversitesi, Bilgisayar Bilimleri Bölümü); Kanada'da kaldığım süre boyunca bana akademisyenliği sevdiren ve paha biçilmez öğütlerle beni hep araştırmaya sevk ettiren, Calgary Üniversitesi Bilgisayar Bilimleri Bölüm Başkanı Sayın Prof. Dr. Ken Barker'a ve sürekli olarak araştırma konularımı tartışma imkanı bulduğum ve bu tezin gelişmesinde büyük katkıları olan, gerek Carnegie Mellon Üniversitesi Robotik Enstitü'de iken ve gerekse ayrıldıktan sonra hiçbir zaman bağlantımı kesmediğim Sayın Dr. Cem ÜNSAL'a şükranlarımı sunarım.

Ayrıca, bana her zaman katlanmak zorunda kalan bölümdeki hocalarım ve arkadaşlarıma, sık sık ziyaret ederek çalışmalarını böldüğüm ODTÜ Bilgisayar Mühendisliğindeki asistan arkadaşlarıma ve 1 yıl boyunca bana göstermiş oldukları konuk ve misafirperverlikten dolayı Calgary Üniversitesi Bilgisayar Bilimleri Bölümündeki tüm arkadaşlara en derin teşekkürlerimi gönderiyorum.

Son olarak, beni her zaman destekleyen ve teşvik eden, başarılarımla gururlanan, kısaca her zaman benimle olan aileme sonsuz teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Doktora Tezi

ÇOKLU ETMEN TAKVİYELİ ÖĞRENMEYE VERİ MADENCİLİĞİ TABANLI YENİ YAKLAŞIMLAR

Mehmet KAYA

Fırat Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik ve Elektronik Mühendisliği
2003, Sayfa: 91

Takviye öğrenme, çoklu etmen sistemlerde öğrenme için güçlü bir yöntem olarak önerilmiştir. Buna karşılık, ortamda bulunan diğer etmenleri modelleme ve öğrenme boyunca bazı durumların gereğinden fazla tecrübelenmesi gibi problemleri de içerir. Bir taraftan bazı durumların yeterince tecrübelenmediği halde etmenden uygun bir hareket seçmesi beklenirken, diğer bir taraftan öğrenme süreci tamamlanmadan önce etmenden yeterince tecrübelendiği durumlarda bile kesin bir davranış sergilemesine izin verilmez. Bu da öğrenme süresinin artmasına neden olur. Bu durum, kısmen gözlenebilir ve dinamik çoklu etmen ortamındaki öğrenmenin hala bazı zorluklar içerdiğini ve daha detaylı incelenmesi gereken önemli bir araştırma problemi olduğunu gösterir.

Yukarıda ifade edilen problemleri ele almak için, bu tezde birlikte çalışarak öğrenen bir sistem için yeni çoklu etmen öğrenme yöntemleri sunulmuştur. Yöntemler, etmenler tarafından elde edilen bilgiyi etkili işlemek için bulanıklık kavramını ve çevrim-içi analitik işlem (online analytical processing) madenciliğini birleştirmektedir. Bu maksatla öncelikle, etmenlerin durum bilgilerini depolama ve işlemeye imkan veren bulanık bir veri küpü tanımlanmıştır. Böylece diğer etmenin hareketi, göz önüne alınan etmenin görme alanının dışında olması durumunda bile çevrim-içi bağıntı kuralları yardımıyla tahmin edildi. Daha sonra, yine bağıntı kurallarına dayanan yeni bir hareket seçme yöntemi önerildi. Bu maksatla, önerilen veri küplerinden çok seviyeli bağıntı kuralları çıkarılarak, yeterince tecrübelenmemiş durumlar genelleştirildi. Son olarak, çoklu etmen ortamda, etmenler tarafından karşılaşılan yavaş yakınsama hızı ve yakınsama adım sayısı gibi problemlerin üstesinden gelmek için, yeni ve güçlü bir modüler mimari ve buna uygun öğrenme yaklaşımı sunuldu. Ayrıca, modüler mimari ile daha önce tanımlanan bütün yöntemler başarılı bir şekilde birleştirildi. İyi bilinen, iki farklı av/avcı problemi üzerinde gerçekleştirilen uygulama sonuçları önerilen öğrenme yaklaşımlarının gücünü ve etkisini gösterdi.

Anahtar Kelimeler: Çoklu etmen takviye öğrenme, Veri madenciliği, Bağıntı kuralları, Çevrim-içi analitik işleme, Bulanık kümeler.

ABSTRACT

PhD Thesis

DATA MINING BASED NOVEL APPROACHES TO MULTIAGENT REINFORCEMENT LEARNING

Mehmet KAYA

Fırat University
Graduate School of Natural and Applied Sciences
Department of Electrical and Electronics Engineering
2003, Page: 91

Reinforcement learning is considered as a strong method for learning in multiagent systems environments. However, it still has some drawbacks, including modeling other learning agents present in the domain as part of the state of the environment, and experiencing some states more than needed during the learning phase. On one hand, although some states are not experienced sufficiently, it is expected that an agent has to select an appropriate action in each state. On the other hand, before the learning process is completed, it is not given permission for an agent to exhibit a certain behavior in some states that may be experienced sufficiently. This causes the increment of the learning time. This case shows that learning in a partially observable and dynamic multiagent systems environment still constitutes a difficult and major research problem that worth further investigation.

In order to handle the problems mentioned above, in this thesis, we propose novel multiagent learning approaches for a cooperative learning system. Our approaches incorporate fuzziness and online analytical processing (OLAP) based data mining to effectively process the information reported by the agents. First, we describe a fuzzy data cube OLAP architecture which facilitates effective storage and processing of the state information reported by agents. This way, the action of the other agent, even not in the visual environment of the agent under consideration, can simply be estimated by extracting online association rules, a well-known data mining technique, from the constructed data cube. Second, we present a new action selection model which is also based on association rules mining. Then, we generalize states which are not experienced sufficiently by mining multiple-levels association rules from the proposed data cubes. Finally, we present a new and robust modular architecture and a corresponding learning approach to overcome the various problems encountered by the agents in multiagent environment, such as slow convergence speed and the number of steps to convergence. Then, we successfully combine advantages of the modular approach with all of the features described above. Results obtained for a well-known pursuit domain show the robustness and effectiveness of the proposed mining based learning approaches.

Keywords: Multiagent reinforcement learning, data mining, association rules, online analytical processing, fuzzy sets.

1. GİRİŞ

1.1. Tezin Amacı

Çoklu etmen sistemler, birden fazla etmeni içeren karmaşık sistemleri oluşturma ve bağımsız etmenlerin davranışını koordine etme ile ilgilenen dağıtık yapay zekanın bir alt dalıdır. Çoklu etmen sistemler kullanmanın en önemli nedeni, parçalarının birlikte çalışmaya gerek duyduğu gerçek dünya problemleri için iyi bir modellemeye sahip olmasıdır. Özellikle perspektifleri farklı insanların veya amaçları farklı organizasyonların bulunduğu bir ortamda, çoklu etmen sistemler etkileşimi sağlamak için gereklidir [1]. Bugün çoklu etmen sistemler, mühendislik tasarımı, akıllı arama, hasta teşhisi ve robotik gibi bir çok dalda kendine uygulama alanı bulmuştur [2].

Çoklu etmen sistemler, genel bir kontrolün olmayışı ve global bir bilginin mevcut olmaması anlamında tekli etmen sistemlerden ayrılırlar. Bu yüzden, sadece bir etmenin işlem gücü sınırlamaları, çoklu etmen ortamında kaybolur. Diğer bir değişle, veri ve kontrol dağıtık olduğundan, çoklu etmen sistemler, ölçeklenebilirlik, hata toleransı ve paralelleştirme gibi dağıtık sistemlerin doğal avantajlarını içerirler.

Son zamanlarda çoklu etmen sistemler üzerine büyük çapta çalışmalar başlamıştır. Bunun bir sonucu olarak, çoklu etmen sistemler, göz önüne aldığı ortamın dinamiği hakkında hemen hemen hiçbir bilgiye gerek duymayan takviye öğrenme yöntemini de içeren bir çok disiplinden faydalanmıştır. Bir ortamda mevcut etmen, ortamdan aldığı durumu, amacı içeren bir başka duruma nasıl dönüştürüleceğini öğrenir. Bu işi yaparken minimum kullanıcı müdahalesine gerek duyan böyle bir etmen otonomos olarak adlandırılır [3]. Otonomos etmenler, ortamıyla etkileştikten sonra takviye işaretleri alarak öğrenme işlemini gerçekleştirirler. Bu şekilde etmenler doğrudan ortamın dinamiğiyle etkilendiklerinden, ortamdan öğrenme her zaman için güçlü bir yöntemdir. Literatürde bu maksatla yapılan çalışmaların çoğunda, her bir etmen diğer etmeni ortamın bir parçası olarak görür [4, 5]. Etmenlerin otonomos olarak öğrendiği bir çoklu etmen ortamı göz önüne alındığında, diğer etmenin davranışı da ortama dahil edildiğinden durum-geçiş fonksiyonu zamanla değişir. Yani ortam, bir Markov Karar Süreci olarak modellenemez. Buna karşılık, literatürde tanımlanan çalışmalarının çoğunda, Markov Karar Sürecine dayalı takviye öğrenme yöntemleri çok fazla değişikliğe uğramadan uygulanmıştır.

Çoklu etmen öğrenmeyi modellemek için kullanılan yaklaşımlardan biri , ortamdaki her bir etmenin durumunu diğer etmenlerin bilgisiyle donatmaktır [4-6]. Buna karşılık, bir çoklu

etmen ortamında etmen sayısı artarken, her bir etmen durum uzayı da üstel olarak artacaktır. Bunun sonucu olarak, çok basit çoklu etmen problemleri bile standart takviye öğrenme yaklaşımlarıyla hesapsal olarak ele alınamaz bir hale gelir.

Q-öğrenme çoklu etmen ortamlarında en yaygın şekilde kullanılan bir takviye öğrenme algoritması olmasına rağmen, diğer öğrenene etmenleri modelleme ve öğrenme boyunca bazı durumların gereğinden daha fazla tecrübelenmesi gibi dezavantajlar içerir. Bir taraftan bazı durumlar yeterince tecrübelenmediği halde, etmenin bir durumda uygun bir hareket seçmesi beklenirken; diğer taraftan öğrenme süreci tamamlanmadan, etmenin yeterince tecrübe kazandığı durumlarda bile kesin bir davranış sergilemesine izin verilmez. Bu da öğrenme süresinin artmasına neden olur.

Yukarıda tanımlanan bütün bu problemleri ele almak için, bu tezde öğrenme süreci, veri madenciliği tekniklerinden biri olan bağıntı kural çıkarımıyla birleştirilmiştir. Veri madenciliği, önceden bilinmeyen, muhtemelen faydalı ve saklı bilgiyi geniş veri kümelerinden çıkarma işlemidir. Bağıntı kurallarını çıkarma ise literatürde tanımlanan birkaç veri madenciliği tekniklerinden biridir [7, 8]. Bir bağıntı kuralı farklı öznitelikler veya nesneler arasındaki ilginç korelasyonu tanımlar. Örnek olarak, bir süper market sepet verisinde bir bağıntı kuralı: "Kayıtların %20'sinde, tereyağı alanların %75'i yumurta da satın alır" şeklinde olabilir. Burada %20 ve %75 sırasıyla kuralın destek ve güven değeridir. Bir bağıntı kuralının önemi bu değerlerle ölçülür. Destek, X ve Y nesnelerinin her ikisini içeren kayıtların yüzdesi olarak tanımlanırken; güven, $X \cup Y$ desteğinin X desteğine oranıdır.

Bu tezde, bağıntı kuralları çıkarılırken, çevrim-içi analitik işlem (Online Analytical Processing) tekniklerinden faydalanılır. Çevrim-içi analitik işleme, daha ileri bir analiz için çoklu boyutlu veri kümesindeki önemli bilgiye hızlı erişim sağlayan bir teknolojidir [9].

1.2. Çevrim-içi Analitik İşlem Madenciliği

Çevrim-içi analitik işlem mimarisinde temel yapı veri küpüdür. Bir veri küpü, çok boyutlu değer dizilerine benzer şekilde düzenlenmiş bir veri kümesidir. Bu küp üzerinde soyutlama yapılarak bir çok hiyerarşi elde edilebilir. Böyle bir yapıyı kullanmanın bir çok nedeni vardır. Örnek olarak, bir çok öznitelikli geniş bir veri tabanı verildikten sonra kullanıcı, özniteliklerin küçük bir kümesi ile ilgilenebilir. Ayrıca bu küp üzerinde çevrim-içi olarak bağıntı kuralları çıkarılmak istenebilir.

Çevrim-içi analitik işlem madenciliği, veri madenciliğini, çevrim-içi analitik işlemle birleştirir. Böylece veri madenciliğinin gücünü ve esnekliğini artırır ve madenciliği ilginç bir keşfetme sürecine dönüştürür. Çevrim-içi bağıntı kural çıkarımı, veri madenciliğinde önemli bir

araştırma konusudur. Çünkü böyle bir mimaride veri, "bir defa işle, bir çok defa sorgula" kavramına imkan verecek şekilde düzenlenmiştir. Literatürde bağıntı kurallarının etkili bir şekilde, özellikle de çevrim-içi çıkarılması için bazı çalışmalara rastlanır. Örneğin, Park ve diğ. [10] nesne kümesi verilerini elde edinim süresini azaltmışlardır. Diğer bir yandan, veri tabanındaki faydalı olmayan bilgileri çıkarmak, madencilik sürecini hızlandırır [11]. Hidber [12], yoğun nesne kümelerini çevrim-içi bulmak için bir algoritma önermiş, Relue ve diğ. [13], bağıntı kurallarının çalışma zamanında çıkarılmasını sağlamak için, örüntü deposu olarak adlandırılan özel bir yapı tasarlamışlardır. Han [14], çevrim-içi analitik işlem araçlarını, veri üzerinde analizini yapmak için yeni bir model önermiştir. Daha sonra, önceki mimariye birkaç veri madenciliği bileşeni ekleyerek çalışmasını geliştirmiştir [15]. Han ve Fu [16], çok geniş veri tabanlarından, çoklu seviye bağıntı kurallarını çıkarmak için Apriori'ye [8] dayalı yukarıdan aşağıya işleyen bir yaklaşım tanımlamışlardır. Bu yaklaşımda öncelikle en üst seviyedeki yoğun nesne kümeleri bulunur ve daha sonra madencilik işlemi daha alt seviyelerdeki yoğun nesne kümelerine doğru derinlemesine devam eder. Kamber ve diğ. [17] çok boyutlu bağıntı kurallarının çıkarımı için bir veri küpü önerdi. Onların modelinde küp veri yapısı, çevrim-içi analitik işlem teknikleriyle birleşir. Performans analizi üzerine yaptıkları çalışmada, çevrim-içi analitik işleme dayalı veri madenciliğinin tablo tabanlı Apriori algoritmasına üstünlük sağladığını gösterdiler. Son olarak, Aggarwal ve Yu [18], bağıntı kurallarını çıkarmak için, çevrim-içi analitik işlem benzeri bir algoritma sunmuştur. Yaklaşımlarının arkasındaki genel fikir, verilen bir destek eşiğine göre bütün yoğun nesne kümelerini geleneksel bir algoritma ile bulmak ve daha sonra yine etkileşimli bir şekilde verilen güven eşiğine göre bağıntı kurallarını çevrim-içi üretmektir.

Çevrim-içi analitik işlem madenciliği üzerine şimdiye kadar yapılan çalışmaların hemen hemen tamamı ikili öznitelikli veri küpleri kullanır. Buna karşılık gerçek hayattaki veritabanlarının bir çoğu sürekli değerler alabilen öznitelikler içerir. Ayrıca durum ve hareketleri bulanık kümeler kullanarak modellenen bir çoklu etmen ortamında, etmenler tarafından rapor edilen durum bilgilerinin etkili bir şekilde depolanması ve işlenmesi için sürekli değerli özniteliklerden ibaret veri küplerinin oluşturulması gerekir. Bu nedenlerden dolayı, bu tezde, bulanık bağıntı kurallarının çevrim-içi madenciliği için yeni bir bulanık veri küpü geliştirildi.

1.3. Tezde Geliştirilenler

Bu tez çalışmasının ana katkıları aşağıdaki gibi özetlenebilir:

1. Etmenler tarafından elde edilen ortamla ilgili bütün bilgiyi etkili bir şekilde depolamak ve işlemek için bulanık bir veri küpü tanımlandı.
2. Etmenlerin durum sayısını azaltmak için, durum uzayı, bulanık kümelerle temsil edildi.
3. Normal ve bulanık veri küplerinden çıkarılan iç model bağıntı kurallarını kullanarak, görülmediği durumda bile diğer etmenlerin hareketleri tahmin edildi.
4. Normal ve bulanık veri küplerinden çıkarılan bağıntı kurallarını kullanarak, göz önüne alınan etmen için yeni bir hareket seçme mekanizması sunuldu.
5. Yeterince tecrübelenmemiş durumlar için, normal ve bulanık veri küplerinden çok seviyeli bağıntı kuralları çıkarılarak ilgili durumlar genelleştirildi.
6. Bir çoklu etmen ortamında ikiden fazla etmen olması durumunda oluşacak durum uzayı patlamasına bir çözüm olarak, yukarıda önerilen yaklaşımlar, modüler bir mimari ile gerçekleştirildi. Böylece etmenlerin durum uzay sayısı daha da azaltıldı.

1.4. Tezin İçeriği

Tezin bundan sonraki bölümleri aşağıdaki gibi düzenlenmiştir.

- Bölüm 2'de, çoklu etmen sistemler ve öğrenme üzerine kısa bir bilgi verilmiş ve bu alanda yapılan çalışmalar sistematik olarak sıralanmıştır. Ayrıca, tez çalışması boyunca üzerinde uygulamalar yapılan av/avcı probleminin iki farklı ortamı tanıtılmıştır.
- Bölüm 3'de, bağıntı kurallarının tanımı verilmiş ve bağıntı kurallarının çıkarımı için, çevrim-içi analitik işlem tekniklerinin kullanımı anlatılmıştır. Ayrıca nicel değerli nesneler için bulanık veri küpü mimarisi tanıtılmıştır.
- Bölüm 4'de, önerilen veri küpünden çıkarılan bağıntı kurallarına dayalı olarak yeni bir çoklu etmen takviye öğrenme algoritması geliştirilmiştir.
- Bölüm 5'de, etmenlerin durum-hareket uzayları bulanık kümelerle modellenerek, bu bilgilerin Bölüm 2'de önerilen bulanık veri küpünde tutulması sağlanmıştır. Bölümün son kısımlarında, etmenlerin ortamdan elde ettiği bu bilgiler yardımıyla yeni bir öğrenme yaklaşımı sunulmuştur.
- Bölüm 6'da, ortamdaki etmen sayısına göre üstel olarak artan durum patlamasını ele almak için yeni ve güçlü bir çoklu etmen modüler mimari önerilmiş ve bu mimari üzerinde bağıntı kurallarını kullanan yeni bir öğrenme algoritması geliştirilmiştir.
- Bölüm 7'de, bu tezde elde edilen sonuçlar tartışılmış ve gelecek araştırmalar için bazı öneriler sunulmuştur.

2. ÇOKLU ETMEN SİSTEMLER

Dağıtık ve açık sistemlerdeki gelişmelere paralel olarak, son zamanlarda dağıtık yapay zeka olarak adlandırılan yapay zekanın yeni bir dalı ortaya çıkmıştır. Dağıtık yapay zeka genel olarak bir ortamda birbirleriyle etkileşen çoklu bağımsız varlıklardan oluşan sistemlerle ilgilidir. Bu anlamda dağıtık yapay zeka iki alt disipline ayrılmıştır. Bunlardan biri dağıtık problem çözme, diğeri ise çoklu etmen sistemlerdir. Dağıtık problem çözme ortak bir amaç için birlikte çalışan sistemlerin bilgi yönetimi üzerine odaklanırken; çoklu etmen sistemler bir ortamda bulunan ve etmen olarak isimlendirilen bağımsız varlıkların davranış yönetimiyle ilgilenir. Etmen tabanlı yaklaşımlar, internet gibi dağıtık bir sistem üzerinde çalışan yazılımlardan beklenilen modülerlik ve güvenilirlik unsurlarını içerirler. Alanda yapılan çalışmaların çoğu genel olarak bir tek etmen üzerinedir. Fakat tek etmenli çözümler gerçek hayattaki bir çok problem için artık yetersiz kalmaya başlamıştır. Bu yetersizliğin başlıca iki nedeni vardır: Bunlardan biri, ortamların genellikle dinamik yani herhangi bir kontrol mekanizmasına bağlı kalmaksızın değişime açık olmaları ve belirsiz bir yapıya sahip olmaları; diğeri ise, tek etmene sahip sistemlerin böyle ortamlarda modülerlik ve güvenilirlik etkenlerine yeterince sahip olamamasıdır.

Bir çoklu etmen sistemi, bir ortamı paylaşan birden fazla etmenden oluşur. Böyle bir sistemi oluşturmanın, etmen yapısı, sistemin amacı ve sistemdeki bireyler arasında haberleşme, koordinasyon ve işbirliği gibi etkenlere bağlı olarak bir çok yolu vardır.

Bir çoklu etmen sistemindeki etmenler iki farklı şekilde planlama ve muhakeme yeteneği sergileyebilir. Etmenler bir amaç doğrultusunda kendi planlarını geliştirmeyi tercih edebilirler yada etmenler takım bazında çalışarak, takımın planlama sürecine katılabilirler. Her iki yolda da sadece planlama ve muhakeme yeteneklerine sahip etmenler dinamik ortamlardaki problemi çözmek için yeterince güçlü değillerdir. Çünkü bu etmenler genellikle sabit bir bilgi ve dinamik ve belirsiz ortamlarda mevcut olmayan ortamın bir ön bilgisine sahiplerdir. Bundan dolayı, bu yetenekler bir öğrenme mekanizması tarafından desteklenmelidir.

2.1. Çoklu Etmen Sistemlerin Tanımı

Birden daha fazla etmene sahip olan sistemler çoklu etmen sistem olarak adlandırılır. Bir insan müdahalesi gibi dış bir kaynağa gerek duymadan çalışabilen otonomos yazılımlara olan gereksinim, çoklu etmen sistemlerin önemini daha da artırmaktadır. Bu gibi sistemler daha çok, üst seviyeli bir kontrolün olmadığı veya tutarlı bir bilginin her zaman için mevcut

olmayacağı işlere atanır. Verinin genellikle dağıtık olduğu dinamik sistem ve robotik araştırma gibi alanlarda çoklu etmen sistemler bir gerekliliktir.

Bir çoklu etmen sisteminin sahip olduğu özellikler aşağıdaki gibi sıralanabilir [19]:

- Her bir etmen, kapasitesi sınırlı bir varlık olduğu için problem çözmede yeterli bir bilgi ve yeteneğe sahip değildir.
- Üst seviyeli bir kontrol yoktur.
- Veri uzaysal ve fonksiyonel olarak dağıtıktır.
- Tüm hesaplamalar eşzamansızdır.

Bir sistemde bütün işleri yapmakla görevli etmenleri tasarlamak dinamik ve açık sistemlerde etkili bir çözüm değildir. Modülerliği ve buradan da güvenilirliği sağlamak için dağıtık çözümler gereklidir. Bu yüzden çoklu etmen sistemlerdeki etmenler genellikle sınırlı bir yeteneğe sahip olacak şekilde tasarlanırlar. Bu anlamda tek bir etmen, amacı sağlamada bir şey ifade etmez.

Çoklu etmen sistemlerdeki her bir etmen bir problemi çözmek için gerekli bütün bilgiye sahip değildir. Çünkü bir etmenin bu kadar geniş bir veri tabanına sahip olması mümkün değildir. Ayrıca veriler zamanla değişime uğrarlar. Bu nedenlerden dolayı verileri etmenler arasında dağıtmak; yüksek güvenilirlik için en iyi, hatta bazen tek çözümdür. Oldukça sık değişime uğrayan veri, etmenlerin öğrenme birimlerini tasarlamada ana unsuru teşkil eder.

Bir çoklu etmen sisteminde, üst seviyeli bir kontrol ve başlangıçta etmenlere verilebilecek tutarlı bilgi yoktur. Bununla birlikte, yapılan bazı çalışmalarda etmenlere hareketlerini koordine etmek için dışarıdan sürekli olarak ek bilgiler verilse de bu yöntem, gerçek dünya uygulamalarında başarıya uğramamıştır. Öyle ki, etmenlere sürekli olarak bilgi sağlayan birimin herhangi bir nedenden dolayı kaybolması durumunda bütün sistem çöker. Bir çoklu etmen sisteminden istenen davranış, etmenlerin kendi kendine yetebilmeleri ve gerektiğinde kendi başlarına hareket edebilmeleridir. Ortak bir amacı sağlama isteği ise etmenler arası işbirliği, görüşme ve koalisyon planlarının gelişimine yol açar. Fakat, bazı durumlarda bir etmenin yapacağı hareket diğerinin zararına olabilir. Böyle durumlar yine etmenler tarafından anlaşmazlık giderici planlarla çözülür.

2.2. Çoklu Etmen Sistemlerde Öğrenme

Yapay zeka araştırmasında öğrenme, insan öğrenme modeline benzeyen ve öğrenme algoritmaları olarak adlandırılan hesapsal mekanizmaları içerir. Bugün yapay zekadan bir çok

araştırmacı insan öğrenme mekanizmasının sırlarını çözmeye ve onları bir makine üzerinde gerçekleştirmeyi amaçlar. Diğer bir taraftan, Hesapsal Zeka (Computational Intelligence) bilimcileri ise insana yönelik eğilimleri göz önüne almaksızın tamamen matematiksel modellere dayanan hesapsal mekanizmaları geliştirmeye çalışırlar. 1960'lı yıllardan bu yana Makine Öğrenmesi (Machine Learning) konusunda bir çok araştırma yapılmıştır. Bu araştırmalardan, kavram öğrenme (concept learning), karar-ağacı öğrenme (decision-tree learning), sinir ağları (neural networks), Bayesian öğrenme (Bayesian Learning), duruma dayalı öğrenme (case-based learning), genetik algoritmalar (genetic algorithms) ve takviye öğrenme (reinforcement learning) gibi öğrenme yöntemleri çıkmıştır [20]. Bu yöntemlerin tamamı değişik uygulama alanlarında kendilerine yer bulmuştur. Fakat bunlardan sadece ikisi gerçek dünya uygulamaları için yeterince sağlam ve ölçeklenebilirdir. Bunlar sinir ağları ve takviye öğrenme tabanlı algoritmalardır. Gerçek dünya uygulamaları;

- Tamamen veya kısmen dinamik,
- Doğal olarak belirsiz ve
- Çoğu zaman kısmen gözlemlenebilirdir.

Internet veya robot navigasyon gibi robotik araştırmaları kapsayan dağıtık ve açık sistemlerdeki uygulamalar son derece dinamik ve belirsiz ortamlara sahiptirler. Örneğin; bir internet bağlantısı göz önüne alınırsa, aynı anda internete bağlanan ve internetten ayrılan binlerce kişi olabilir. Bu kişilerin davranışları önceden tahmin edilemez. Kısmen gözlenebilirlik, bir kullanıcının belirli bir anda bilginin sadece belirli bir kısmına erişebilmesidir. Bu sınırlama internet üzerinde çalışan bir etmen için bütün verileri saklayacak bir diske sahip olmamak veya robotlar için yeterli ve keskin sensörlere sahip olmamak şeklinde de tanımlanabilir.

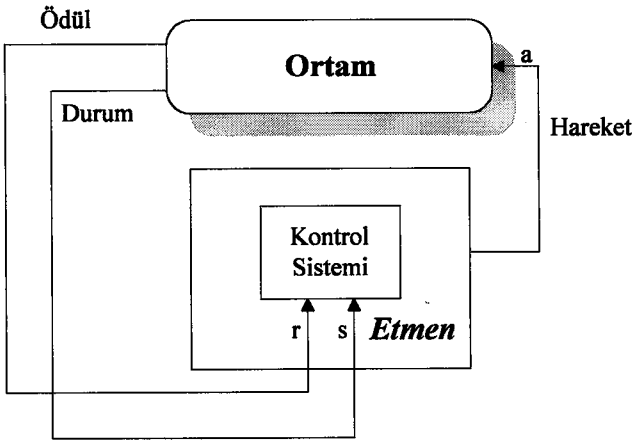
Öğrenme algoritmalarının çoğu sabit kurallara dayalıdır ve ortam hakkında bir ön bilgiye gerek duyarlar. Bu gereksinimlerinden dolayı, hakkında bilgi sahibi olamadıkları dinamik ve belirsiz ortamlara uygulanamazlar. Sinir ağları ve takviye öğrenme metotlarının böyle bir gereksinime ihtiyaçları yoktur. Sinir ağlarının giriş nöronları öğrenme süresince sürekli olarak beslendiğinden, sinir ağları dinamik şekilde değişen veriler üzerinde kolaylıkla çalışabilir. Takviye öğrenme metotları da ortam ve etmen arasındaki etkileşmeye bağlı olarak çalıştığından, dinamik ortamlara güvenli bir şekilde uygulanabilirler. Fakat, sinir ağlarının yakınsaması, takviye öğrenmeden farklı olarak ispatlanamamıştır [21]. Bu nedenle bu yöntem gerçek dünya öğrenme problemlerinde takviye öğrenmeye nazaran daha az tercih edilir.

2.3. Takviye Öğrenme

Takviye Öğrenme (TÖ), bir ortamı algılama ve hareket etme yeteneklerine sahip otonomos bir etmenin, amacını başarmak için en uygun hareketleri nasıl öğreneceği sorusuna cevap verir [22, 23]. Bu problem ortam hakkında ön bilgiye sahip olmayan bir etmenin karşılaştığı tipik durumlardan biridir. Bir makine öğrenme yöntemlerinden biri olan TÖ, bilgisayar bilimi, psikoloji ve istatistik gibi bir çok disiplinden esinlenerek ortaya çıkmıştır.

TÖ ile problemleri çözmek için iki ana yöntem vardır. Bunlardan ilki, ortamı iyiye götüreni bulmak için davranış uzayında bir arama yapmak, ikincisi ise, faydalı hareketi tahmin etmek için istatistik ve dinamik programlama yöntemlerini kullanmaktır [23]. TÖ problemi için önerilen diğer önemli yaklaşımlar ise oyun-teorik ve fonksiyon yaklaştırma yöntemleridir. İlk yöntemde, öğrenme problemi bir stokastik oyun gibi modellenir ve oyun veya oyun takımları öğrenilen bilgiye göre bir dengeyi yakalamaya çalışır [24]. İkinci yöntem ise kullanılan nörona bağlı olarak polinom uzayında problemi çözmek için sinir ağları kullanır [25].

Standart takviye öğrenme modelinde, bir etmen Şekil 2.1'de gösterildiği gibi ortamıyla algı ve hareket mekanizmaları sayesinde etkileşir. Her bir etkileşim anında etmen giriş olarak ortamın şimdiki durumunu (s) gözler ve çıkış olarak bir hareket (a) seçer. Hareket ortamın durumunu değiştirmeyebilir. Yapılan hareketin değeri etmene bir takviye olarak (r) geri döner. Bu takviye pozitif (ödül) veya negatif (ceza) olabilir. Etmen kontrol mekanizması, hareketleri seçerken takviye işaretlerinin toplam değerinin olabildiğince büyük olmasına çalışır. Bunun için ortamı ile sürekli olarak deneme-yanılma etkileşimleri yapması gerekir.



Şekil 2.1. Takviye öğrenme modeli.

Böyle bir model aşağıdaki bileşenleri kapsar:

- Sonlu durumlar kümesi, S

- Sonlu hareketler kümesi, A
- Takviye işaretler kümesi, R

Etmenin amacı takviyelerin toplamını maksimum yapan bir çözüm yolu bulmaktır. Aşağıda TÖ ile ilgili bazı tanımlamalar verilmiştir.

Tanım 2.1. (Politika): Bir π politikası, ortamın algılanan kümesi S'den etmen hareketleri A'ya bir planlamadır. Diğer bir deyişle;

$$\pi : S \rightarrow A \tag{2.1}$$

Tanım 2.2. (Ödül): R ödül fonksiyonu, algılanan durumlardan (veya durum-hareket çiftlerinden), o durumun iyiliğini gösteren nümerik değerlere planıdır. Biçimsel olarak R iki şekilde verilebilir:

$$R : S \rightarrow \Re \text{ veya} \tag{2.2}$$

$$R : SxA \rightarrow \Re \tag{2.3}$$

Burada $\Re$ gerçek sayılar kümesidir. Bu fonksiyonun değeri pozitif veya negatif olabilir. Pozitif değer etmenin ortamda iyi bir hareket yaptığını, negatif ise kötü bir hareket yaptığını gösterir.

Tanım 2.3. (Durum değeri): Değer fonksiyonu, bir ödül fonksiyonunun uzun vadeli göstergesidir. Ödül fonksiyonu R, şimdiki durumun veya durum-hareket çiftinin anlık iyiliğini gösterir. Buna karşılık, değer fonksiyonu mevcut durumdan başlayarak hedefe ulaşıncaya kadar bir ödülün toplam değerini gösterir.

$$V : S \rightarrow \Re \text{ veya} \tag{2.4}$$

$$V : SxA \rightarrow \Re \tag{2.5}$$

TÖ' deki değer fonksiyonu, bir hareket seçilirken "gecikmiş ödülün" göz önünde bulundurulmasını sağlar. Bir etmenin hareketi sadece ortamdan alınan anlık ödüle değil, sonraki duruma da bağlıdır. Sonraki durumun değeri, etmen tarafından gecikmiş ödül olarak alınır. Etmenin uzun vadeli en iyilik modeli, gecikmiş ödülün öğrenme sürecinde nasıl kullanılacağını da açıklar. Gecikmiş ödülden öğrenen bir etmen bir çok hareketten sonra yüksek ödüllü bir duruma (amaç durum) erişebilir.

TÖ' de diğer bir önemli problem, etmenin ortamını keşfetme süresi ile ortamını tanıdıktan sonra yapacağı hareketlerdir. Keşif boyunca etmen, ortamdaki diğer varlıkları

tanımak ve ortam hakkında daha fazla bilgi almak için rasgele hareketler yapabilir. Belirli bir öğrenme sürecinden sonra, etmenin ortamını iyi tanıdığı ve bundan sonraki hareketlerini öğrenme süresince elde ettiği tecrübelere göre yapması beklenir. Bu aşamada etmen daha az öğrenir. Öğrenme algoritması, etmenin ne zaman keşiften öğrendiklerini kullanma aşamasına geçmesi gerektiğini söyleyebilmelidir. Sürekli bir keşif, etmeni yanlış hareketlere sürükler. Buna karşılık ortamını tam olarak tanımdan etmenin bildiği hareketleri yapması, etmeni hiç bir zaman amacına ulaştırmayabilir. Literatürde hareket seçimi yapılırken daha çok Boltzmann dağılımı [23] kullanılmıştır.

Şimdiye kadar bir çok TÖ-tabanlı öğrenme algoritması önerilmiştir [26-34]. Fakat, Q-öğrenme algoritması en yaygın olarak kullanılandır [35]. Bu algoritmanın detaylı bir tanımı sonraki alt bölümde verilecektir.

2.3.1. Q-Öğrenme Modeli

Q-öğrenme, sonlu durumlu Markov Karar Süreci [36] olarak modellenebilen alanlara kolaylıkla uygulanabilecek modelden bağımsız bir TÖ algoritmasıdır. TÖ problemleri matematiksel olarak Markov karar süreçleri gibi modellenebilirler. Markov karar süreci aşağıdaki parametrelere bağlı olarak tanımlanır:

- Sonlu durumlar kümesi, S
- Hareketler kümesi, A
- Bir ödül fonksiyonu, $R : SxA \rightarrow \Re$
- Durum geçiş fonksiyonu $T : SxA \rightarrow \pi(S)$

Yukarıda $\pi(S)$, S kümesi üzerinde bir olasılık dağılımıdır. Örneğin, $T(s,a,s^1)$, a hareketini kullanarak s durumundan s^1 durumuna geçiş olasılığını gösterir [22]. Durum geçiş fonksiyonu, etmenin şimdiki durum ve hareketinin bir fonksiyonu olarak, bir olasılıkla ortamın sonraki durumunu belirler. Ödül fonksiyonu önceden tanımlandığı gibi anlık ödülü verir. Bu özellik "Markov Özelliği" olarak bilinir. Buna göre; geçmiş durum, hareket ve takviye arasında bir bağımlılık yoktur. Bu nedenle, Markov karar süreci ortamın sadece bir adımlık dinamiğini tanımlar.

2.3.2. Q-Öğrenme Algoritması

$Q(s,a)$, $s(\in S)$ durumunda $a(\in A)$ hareketi seçilerek elde edilebilecek maksimum azaltılmış ödül olarak tanımlanır. Q değerleri, her bir durum ve hareket çifti için ayrı girişi olan bir tabloda tutulur. Algoritma bütün tablo girişlerine sıfır veya çok küçük değerler atanarak başlar ve her bir zaman adımında aşağıdaki adımları gerçekleştirir:

1. şimdiki s durumunu gözlemle,
2. a hareketini seç ve yürüt,
3. yeni s^1 durumunu gözlemle,
4. ortamdan r ödülünü al,
5. s durumu ve a hareketi için uygun Q değerini aşağıdaki formüle göre güncelle,

$$Q(s,a) = (1-\beta)Q(s,a) + \beta(r + \gamma \max_{a^1} Q(s^1, a^1)) \tag{2.6}$$

Yukarıdaki formülde, $\beta(0 \le \beta \le 1)$ öğrenme oranı ve $\gamma(0 \le \gamma < 1)$ azalma parametresidir. s^1, s durumunda a hareketi yapıldıktan sonra etmenin gözlemleyeceği durumdur.

β yeni gözlemlerin ve takviyelerin önemini belirler. Ekstremum şartlarda; eğer β sıfır alınırsa, yeni gözlemler dikkate alınmaz ve Q değeri aynı kalır. Eğer β, 1 alınırsa, eski Q değerleri ihmal edilir ve yeni değer sadece en son gözleme bağımlı olur. Öğrenme safhasının başlarında yüksek β değerleri kullanmak ve daha sonra derecesel olarak bu değeri azaltmak faydalı olabilir.

Q-öğrenme algoritmasının, sonlu durumlu Markov karar süreci için optimum Q değerlerine yakınsadığı ispatlanmıştır [35]. Bunun için etmenin her bir durum-hareket çiftini sonsuz sayıda sık ziyaret ettiği kabul edilmiştir. Bu şart gerçek dünyada uygulanabilir değildir ama belirli bir denemeden sonra etmenlerin optimum Q değerlerine yakınsadığı gözlenir.

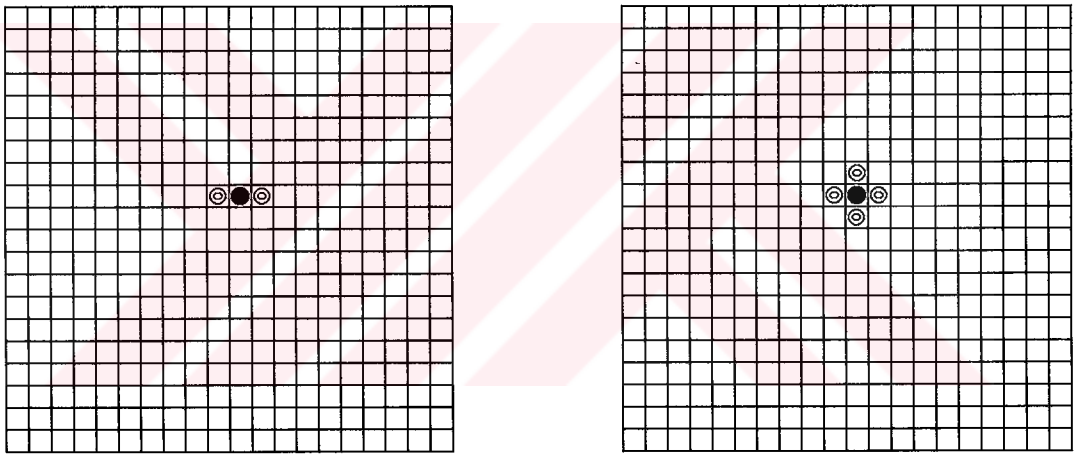
Şimdiki durum için a hareketini seçme yöntemi hala açık bir problemdir. En fazla kullanılan yöntem, etmenin Q değerleri üzerinde Boltzmann dağılımını kullanmaktır. Boltzmann denklemi aşağıdaki formülle verilir:

$$P(a_i \mid s) = \frac{k^{Q(s,a_i)}}{\sum_{j \in A} k^{Q(s,a_j)}} \tag{2.7}$$

Yukarıdaki formülde, $P(a_i \mid s)$, etmen s durumunda iken a_i hareketini seçme olasılığıdır. k değeri daha önce tartışılan keşfetme ve keşfedileni kullanma arasındaki planı belirler. Küçük k değerleri için etmen daha küçük Q değerli hareketleri seçerek farklı durumları keşfetmeye çalışır. Öğrenme ilerlerken k değeri artırılır ve etmen daha büyük Q değerlerini seçer. Böylece etmen öğrendiği bilgiyi kullanmaya başlar.

2.4. Av/Avcı Problemi

Bu tezde önerilen çoklu etmen mimari ve öğrenme yaklaşımlarını değerlendirmek için iyi bilinen av/avcı problemlerinden [37] iki farklı türü tanımlanmıştır. Problem çoklu etmenler için uygun bir öğrenme ortamı sağlar. Ayrıca etmenler aralarında işbirliği yaparak çoklu etmen öğrenme kavramını geliştirirler.



(a) **(b)**

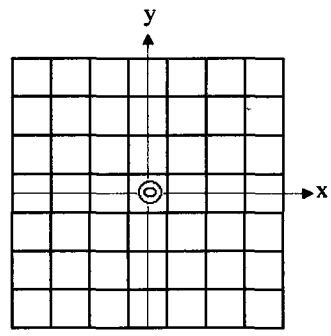
Şekil 2.2. Av/avcı problemi **(a)** İki avcı etmeni ile **(b)** Dört avcı etmeni ile yakalanan versiyonları

Yukarıdaki şekilde gösterilen iki farklı av/avcı problemi aşağıdaki özelliklere sahiptir:

- Her iki ortam tamamen dinamik, kısmen gözlenebilir ve belirsizdir.
- mxn ızgara dünyasında Şekil 2.2. (a) ile gösterilen ilk ortam da iki avcı etmen ve bir av etmeni; Şekil 2.2 (b) ile gösterilen ikinci ortamda dört avcı etmen ve bir av etmeni bulunur. Her bir etmenin başlangıç durumları rasgele belirlenir.
- Her bir zaman adımında etmenler eşzamanlı olarak 5 hareketten birini seçebilirler. Bu hareketler, şu andaki durumdan bir adım kuzeye, doğuya, güneye veya batıya git ile şimdiki durumunda kal'dır. Birden fazla avcı etmen aynı hücreyi paylaşabilir. Buna karşılık, bir avcı etmeni, av etmenle aynı hücrede bulunamaz. Ayrıca bir etmenin

ortamdan dışarı çıkmasına izin verilmez. Bu son iki hareket geçersiz hareketler olarak göz önüne alınır. Eğer bir etmen geçersiz bir hareketi yapmaya kalkışırsa ona izin verilmez ve şimdiki durumunda kalmaya mecbur bırakılır.

- Her etmen belirli uzaklıktaki nesneleri görebilir. Bu uzaklık ve gözlemleyebildiği hücreler sırasıyla etmenin görme derinliği ve görme alanı olarak adlandırılır. Görme derinliği d olan bir etmen etrafındaki $(2d+1)^2$ hücreyi görebilir.



Şekil 2.3. Görme derinliği 3 olan bir avcı etmenin görme alanı.

- Avcılar öğrenen etmenlerdir. Av etmeni ise hareketini rasgele veya Manhattan-uzaklığa göre seçer. Eğer av, Manhattan uzaklığı kullanacaksa görme alanındaki avcıların Manhattan uzaklıkları toplamını maksimum yapmaya çalışır. Manhattan uzaklık aşağıda tanımlanmıştır:

Tanım 2.4. (Manhattan uzaklık): $p_1(x_1, y_1)$ ve $p_2(x_2, y_2)$ ile verilen iki nokta arasındaki Manhattan uzaklık;

$$|x_1 - x_2| + |y_1 - y_2| \tag{2.8}$$

ile bulunur.

- Bir avcı etmenin durumu görme alanındaki diğer av veya avcı etmenlerin pozisyonuna göre belirlenir. Eğer bir avcı etmenin görme derinliği d ise, diğer etmenlerin olabileceği olası yer sayısı $(2d+1)^2+1$'dir. Burada $(2d+1)^2$ hücre sayısını verir iken, 1 diğer etmenin gözlenemediği, yani görme alanın dışında olduğu anlamına gelir.
- Eğer av, birinci ortam için (Şekil 2.2. (a)) her iki taraftan; ikinci ortam için (Şekil 2.2. (b)) dört taraftan avcılar ile kuşatılmışsa, av yakalanmış sayılır. Bu durumda bu deneme biter ve av ve avcılar rasgele bir pozisyona yerleştirilerek sonraki deneme başlar.

2.5. Çoklu Etmen Takviye Öğrenme ve Yapılan Çalışmalara Bir Bakış

Çoklu etmen sistemler son zamanlarda, bilgisayar biliminde büyük ilgiyle çalışılan konulardan biridir. Şimdiye kadar takviye öğrenme de dahil bir çok disiplinden başarılı bir şekilde faydalanılmıştır. Otonomos etmenler, ortamıyla etkileştikten sonra takviye işaretleri alarak ortamından öğrenen varlıklardır. Etmenler ortam dinamiğiyle doğrudan etkileştiklerinden dolayı, ortamdan öğrenme her zaman için güçlü bir yöntemdir.

Çoklu etmen öğrenmeye bir yaklaşım, ortamda var olan diğer etmenlerin bilgilerini her bir etmenin durumuna eklemektir [4-6]. Çoklu etmen takviye öğrenme üzerine yapılan çalışmaların çoğunu da bu öğrenme yöntemi oluşturur. Ortamdaki her bir etmen, bilgi değiş-tokuş mekanizmasıyla desteklenen bireysel öğrenicilerdir. Bir etmen tarafından öğrenilen bilgi sistemdeki diğer etmenlerle paylaşılır. Bu plan ile etmenlerin tamamı sistemin hedefini başaran politika veya politikaları öğrenir. Fakat, ortamdaki etmen sayısı artarken, her bir etmenin durum sayısı da üstel olarak artar. Bu şekilde basit bir çoklu etmen problemi bile standart takviye öğrenme yaklaşımları ile hesapsal olarak üstesinden gelinemeyecek bir hale gelir. Çoklu etmen takviye öğrenmedeki durum sayılarının fazlalığına bir çözüm olarak, Whitehead [38], "modüler Q-öğrenme" olarak adlandırılan bir mimari önerdi. Bu mimariye göre durum uzayı daha küçük alt uzaylara ayrılır ve çoklu modüller arasında dağıtılır. Bu şekilde çoklu-hedef problemlerinin ortak hedefi alt hedeflere ayrıştırılır. Her bir modül sadece kendi hedefini öğrenmeye çalıştığı için, modüllerin alacağı durum sayısı azalır. Whitehead [38], bu şekildeki bir öğrenmenin, öğrenme zamanını geliştirdiğini ve daha az hesapsal kaynaklar harcadığını göstermiştir. Bu çalışmadan sonra, Ono ve Fukumoto [39, 40], Whitehead'in önerisine dayalı yeni bir modüler yaklaşım sundu. Bir av/avcı problemi üzerinde gerçekleştirdikleri tecrübeler ile, başarılı bir şekilde sentezlenmiş modüler Q-öğrenen avcı etmenlerin, rasgele hareket eden av etmenlerini yakalamak için gerekli olduğunu gösterdiler. Bir çoklu etmen ortamı için modüler Q-öğrenme üzerine yapılan diğer bir çalışmada, Park ve diğ. [41], futbol oynayan robotlar için bir hareket seçme mekanizması önerdi ve gerçek robotlar üzerinde gerçekleştirilen tecrübelerle yaklaşımlarının etkili olduğunu gösterdiler.

Bir etmenin bilgiyi nasıl kazanacağı ve besleyeceği takviye öğrenmede önemli bir sorundur. Eğer öğrenme işinin durum uzayı yukarıdaki çalışmalar gibi küçük ve ayrık ise, Q değerleri genellikle bir arama (lookup) tablosunda tutulur. Fakat bu yöntem daha geniş durum-hareket uzaylarında pratik değildir veya sürekli durumlu uzaylarda mümkün değildir. Arama tablolarının ana eksikliği onların ölçeklenebilirliğidir. Geniş durum uzaylı bir öğrenme işinde, sınırlı bir hafızada bütün durumları depolamak ve makul bir zamanda her bir durumu ziyaret etmek mümkün değildir. Bu aşamada çözüm ziyaret edilen durumları, öğreticili öğrenmede

(supervised learning) olduğu gibi ziyaret edilmeyenlere genelleştirmektir. Bu problemi ele almak için, fonksiyon yaklaşım ve genelleştirme yöntemleri daha uygun çözümler olarak görünür. Bununla birlikte, takviye öğrenme algoritmalarının fonksiyon yaklaştırmayı kullanan gerçekleştirimlerinde optimum yakınsamanın henüz ispatlanamamış olduğunu söylemek gerekir [23]. Şimdiye kadar bu alanda iki farklı yaklaşım kullanılmıştır.

İlk yaklaşım, Q değerlerini, ileri-beslemeli sinir ağlarının [42] ve kendinden organize olan eşlemelerin (self-organizing maps) [43] genelleştirme yeteneğini kullanarak depolar. Bazı araştırmacılar bu yöntemler üzerine çalıştılar ve onlara uygun hale getirmek için arama tablosu algoritmalarını değiştirdiler [25, 34, 44]. Fakat bu yaklaşımın dezavantajı sinir ağları ve takviye öğrenme gibi iki yavaş işleyen süreci birleştirmektir.

İkinci yaklaşım, bulanık mantık ve takviye öğrenme tekniklerini birleştirerek geniş ve sürekli durum uzaylı problemleri ele alır. Beom ve Cho [45] hareketli robotların navigasyonu için bulanık mantık ve takviye öğrenmeden faydalanan sensör tabanlı bir navigasyon yöntemi önerdi. Daha sonra, Yung ve Ye [46], EEM-tabanlı bir eğitme yöntemine alternatif bir mimari olan bulanık reaktif navigasyon mimarisini önerdi. Berenji ve Vengerov [47, 48] durum geçişlerinde sabit ve Markovyen olmayan karakterli bir öğrenme problemi ele aldılar. Onlar bu şekilde sürekli durum uzaylı, kısmen gözlenebilir bir ortamda bulanık takviye öğrenen etmenleri kullanmanın etkisini gösterdiler. Son olarak, bir av/avcı ortamında hareket eden etmenlerin durum uzayını farklı bulanık seviyelere gruplandırdık [49]. Bu şekilde, problem uzayının boyutunu azalttık ve optimuma yakın çözüme daha bir hızlı yakınsama sağladık.

Yukarıdaki çalışmaların tamamında her bir etmen diğer etmeni ortamın bir parçası olarak kabul eder. İçinde otonomos olarak öğrenen etmenlerin bulunduğu bir çoklu etmen ortamı göz önüne alındığında, her bir etmenin davranışı değişir. Diğer etmenlerin davranışları da ortamda etkili olduğundan, durum-geçiş fonksiyonu zamanla değişir. Bu şekilde ortam Markov karar süreci olarak modellenemez. Buna karşılık, çoklu etmen takviye öğrenme çalışmalarının çoğunda Markov karar sürecine dayalı takviye öğrenme yöntemleri değiştirilmeden uygulanmıştır.

Literatürde, diğer öğrenen etmenin iç modelinin açık bir şekilde göz önüne alındığı bazı çalışmalara rastlanır. Bu maksat için, Littmann [6], iki oyunculu sıfır-toplam ödüllü bir stokastik oyun tanımlamıştır. Mini-max Q-öğrenme olarak adlandırdığı yönteminde bir etmen diğer etmenin hareketini kendi iç modeline göre tahmin eder ve bu tahmine göre öğrenir. Sıfır-toplam oyununda, bir etmenin kazancı sürekli olarak diğer etmenin kaybıdır. Böylece bir etmenin diğerine göre zıt hedefi vardır. Hu ve Wellman [24] yine iki oyunculu genel-toplam ödüllü farklı bir çoklu etmen takviye öğrenme ortamını önermiştir. Bir etmenin kazancı, genel-toplam oyununda diğer etmenin kaybı olmadığından, mini-max Q-öğrenme yöntemi daima uygun

değildir. Bununla birlikte, her iki yönteme göre, diğer etmenin Q fonksiyonu tahmin edilirken, göz önüne alınan etmen diğer etmenin hareketlerini, gerçek ödüllerini gözlemleyebilmelidir. Ayrıca aynı etmen, diğer etmenin Q-öğrenmede kullandığı parametreleri de bilmelidir. Son olarak, Nagayuki ve diğ. [50], bu problemi ele almak için farklı bir yaklaşım önerdiler. Q-öğrenmeye dayalı yöntemlerinde, bir etmen diğer etmenin Q fonksiyon yerine, politikasını tahmin eder. Böylece ortamdan diğer etmenin gerçek ödüllerini gözlemlemesine ve Q-öğrenme için diğer etmenin kullandığı parametreleri bilmesine gerek yoktur.

3. BAĞINTI KURALLARININ ÇEVRİM-İÇİ ANALİTİK MADENCİLİĞİ

3.1 Bilgi Keşfi ve Veri Madenciliği

Bilgi saklama teknolojilerindeki son gelişmelerle, bir çok organizasyon büyük miktarda verileri toplama ve saklama problemini aşmıştır. Bu veriler içerisine çok faydalı bilgiler de gömülü olduğu halde, bunu kullanıcıya çıkarmak kolay değildir. Bu nedenle, bu saklı bilginin kendiliğinden kullanıcıya çıkarılmasına ve kullanıcının bu bilgiyi analiz etmesine yardım edecek tekniklerin ve araçların gelişmesine sürekli olarak ihtiyaç vardır. İşte Veritabanlarında Bilgi Keşfi (VBK) bu ihtiyacı karşılayacak teknikleri ve araçları içerir.
Fayyad ve diğ. [51] veritabanlarındaki bilgi keşfini aşağıdaki gibi tanımlar:

"VBK, verideki geçerli, yeni, muhtemelen faydalı ve anlaşılabilen örüntüleri tanımlayan önemli bir süreçdir."

VBK gerçekte beklenilmeyen, faydalı ve basit örüntüleri bulmayı amaçlar. Son zamanlarda özellikle, makine öğrenme, örüntü tanıma, veritabanları, istatistik, yapay zeka, uzman sistemler, çizge teorisi ve veri görselleştirme gibi bir çok alandan araştırmacıların ilgisini çekmiştir. VBK sistemleri genellikle bu alanların tamamından yöntemler, algoritmalar ve teknikler kullanır.

VBK bazı özel ölçü ve eşik değerlere göre ilginç bilgiyi çıkartmak için veri madenciliği tekniklerini kullanan interaktif ve çok adımlı iteratif bir süreçdir. Fayyad ve diğ. [51, 52] ile Manila [53, 54] bilgi keşfetme adımlarını aşağıdaki gibi tanımlar:

1. Bilgi alanını, ön bilgiyi ve kullanıcının amaçlarını anlama,
2. Hedef veri kümesi oluşturma,
3. Veri kümesini ön işleme tabi tutma (veri kaynaklarının seçilmesi, verinin hatalardan ve gürültülerden temizlenmesi, bilinmeyen değerleri ele alma gibi...),
4. Veri madenciliği işini ve algoritmasını seçme,
5. İlginç ve yoğun örüntüleri arama (veri madenciliği),
6. Keşfedilen örüntüleri son işlem sürecine sokma (örüntülerin seçimi, eleminasyonu veya sıralanması, sonuçların görselleştirilmesi),
7. Sonuçların kullanıcıya sunumu.

Veri madenciliği VBK'da bir adımdır ve verideki yoğun ve ilginç örüntüleri keşfetmek için kullanılır. Bu örüntüler düzenlilik, istisnalık ve birliktelik formunda olabilir. Veri madenciliği uygulamaya bağımlıdır ve farklı uygulamalar farklı veri madenciliği tekniklerine gerek duyabilir. Genelde veri madenciliği işleri iki sınıfa ayrılır: Tanımlayıcı veri madenciliği ve tahmini veri madenciliği. İlk sınıf veri kümesini kısa ve özet bir biçimde tanımlar ve verinin ilginç özelliklerini sunar. Buna karşılık ikinci sınıf bir veya birkaç veri modeli oluşturur, onlar üzerinde çıkarsamalar yapar ve yeni veri kümelerinin davranışını tahmin etmeye çalışır [55] .

Bir veri madenciliği sistemi aşağıdaki veri madenciliği işlerinden bir veya birkaçını başarabilmelidir:

- Sınıf tanımlama,
- Bağıntı kuralları,
- Sınıflandırma,
- Tahmin,
- Kümeleme,
- Zaman serisi analizi.

Veri madenciliği teknikleri ise aşağıdaki gibi sıralanabilir:

- Tahmini modelleme,
- Kümeleme,
- Özetleme,
- Bağımlı olmayı modelleme,
- Sapma belirleme.

Sınıflandırma ve regresyon, tahmini modellemenin örnekleridir. Bağıntı kuralları, özetlemenin; fonksiyonel bağımlılık, bağımlı olmayı modellemenin; sıralı örüntüler ise sapma belirmenin örnekleri içine girer.

Chen ve diğ. [56] veri madenciliği yöntemlerini 3 kritere göre sınıflandırır:

1. Üzerinde çalışılacak veri tabanının çeşiti (ilişkisel, özniteliğe yönelik, vb.)
2. Çıkartılacak bilginin çeşiti (Bağıntı kuralları, sınıflandırma kuralları, karakteristik kurallar, ayrıştırıcı kurallar, sıralı örüntüler, sapma, benzerlik, kümeleme, regresyon vb.)
3. Faydalanılacak tekniklerin çeşitleri (verim sürümlü madenci, sorgu sürümlü madenci, interaktif madenci vb.)

Veri madenciliği gerçekte zor bir problemdir. Verilen problem için bir çok çözüm mevcut olsa da, sonuçların kalitesini gösteren kesin bir ölçü yoktur. Veri madenciliği uygulamalarının çoğunda, veri tabanı boyutu çok geniştir. Çünkü değerli sonuçların elde edilebilmesi için bu şartdır. Genellikle veri madenciliği sürecinin sonuçları çok fazladır, bu sonuçların, kullanıcın anlayacağı şekilde ayrı bir işleme tabi tutulması kaçınılmazdır. Veri madenciliği keşfe-dayalı bir süreçtir. Yani kullanıcı peşinen ne keşfedileceğini bilmez.

VBK'nin karşılaştığı en önemli problemler aşağıdaki gibi sıralanabilir:

- Çok geniş veri tabanları,
- Çok boyutlu veri tabanları,
- Farklı veri tipleri,
- Veri ve bilginin sürekli değişimi,
- Eksik ve hatalı veriler,
- Öznitelikler arasındaki karmaşık ilişki,
- Sonuçların faydalılığı, kesinliği ve anlamlılığı,
- Sonuçların anlaşılabilirliği,
- Çoklu seviyelerde interaktif madencilik,
- Kullanıcıyla etkileşim ve ön bilginin kullanımı,
- Diğer sistemlerle entegrasyonu,
- Çoklu veri kaynaklarından madencilik, ve
- Güvenlik koruması

3.2. Bağıntı Kuralları

Bağıntı kurallarını (association rules) keşfetme, veri madenciliğinin en önemli ve en yaygın olarak kullanılan tekniklerinden biridir. Bağıntı kuralları, bazı özniteliklerin varlığını diğer özniteliklerin varlığı ile açıklayan bütün olası kuralların çıkarımı ile elde edilir. Diğer bir değişle, bir bağıntı kuralı farklı öznitelikler veya nesneler arasındaki korelasyonu tanımlar. Bu tanıma uygun olarak, $X \Rightarrow Y$, "kayıtların belirli bir kesimi için, X öznitelik kümesi yine belirli güven değeri altında Y öznitelik kümesini tanımlar" ifadesinin biçimsel gösterimidir. Bir bağıntı kuralı örneği olarak, 'bir satış veri tabanında kayıtların %80'i ekmek ve tereyağını birlikte içerir ama kayıtların sadece %5'i bu iki nesneyi kapsar" verilebilir. Burada %80 kuralın "güveni", %5 ise kuralın "desteğidir". Bugün bağıntı kuralları; süper market kayıt analizi, nesneler arasındaki promosyonlar, haberleşme alarm korelasyonu, üniversite kurs kayıt analizi, parekende

alışverişde müşteri davranış analizi, katalog tasarımı, yazı metinleri, kullanıcıların web ziyaretleri ve stok kayıtları gibi bir çok alanda uygulanmıştır.

Bağıntı kurallarını keşfetme problemi ilk olarak Agrawal ve diğ. [7] tarafından, süpermarket sepet verileri üzerinde tanıtılmıştır. Bu çalışmada veri, ikili olarak gözönüne alınmıştır. Yani bir nesne bir kayıtda ya mevcutdur yada değildir. Kayıtdaki nesnelerin miktarı dikkate alınmamıştır. Ayrıca bağıntı kurallarını çıkarma iki alt probleme ayrılmıştır: Bütün yoğun nesne kümelerini bulma ve sonra bu nesne kümelerinden bağıntı kurallarını çıkarma. Bunlardan ikincisi hesapsal olarak fazla maliyet getirmeyen bir süreçtir ve kabul edilebilir bir zamanda etkili bir şekilde bulunabilir. Fakat ilk problem, özellikle çok büyük veri tabanları için hesapsal olarak oldukça yüklü bir iştir. Zaten gerçek hayattaki bir çok veri tabanı da bunlardan oluşur. Örneğin, parekende satış bilgilerini tutan bir veri tabanı genellikle milyonlarca kayıt ve binlerce nesne icerir. Veri N tane nesne içerdiği zaman, olası yoğun nesne küme sayısı 2^N dir. Fakat, veri tabanında var olan yoğun nesne küme sayısı 2^N den çok daha küçüktür. Bu yüzden, bilinen arama teknikleri, yoğun nesne kümelerini elde etmek için üstel bir zamana ihtiyaç duyarlar. Olası yoğun nesne kümelerini azaltmak için, şimdiye kadar bir çok etkili algoritma önerilmiştir [10]. Bu algoritmalar genellikle hash-tablo, hash-ağaçları, kafes yapıları ve çoklu-hiper çizgeleri gibi veri yapıları kullanır. Bu yapılar arama alanlarını daraltıklarından dolayı arama sürecini de hızlandırırlar.

Bağıntı kural algoritmalarının çoğu veri üzerinde çoklu geçiş yapar. Veri tabanındaki sayısını tutmak için her bir nesne kümesine bir sayıcı ilişkilendirilmiştir. Veri tabanının ilk geçişinde, uzunluğu 1 olan yoğun nesne kümeleri belirlenir. Daha sonraki her bir geçişde bu uzunluk artırılır. Her bir geçişden sonra olası yoğun nesne kümeleri oluşturulur, bu kümeye aday nesne kümesi denir. Daha sonra bu kümeden yoğun nesne kümeleri elde edilir.

Genellikle bir bağıntı kural algoritmasının üstünlüğü, çıkardığı aday kümenin boyutu ve veri tabanını tarama sayısı ile ölçülür. Bağıntı kural algoritmalarının çoğu, yoğun nesne kümelerini çıkarmak için aşağıdaki problemleri çözmeye çalışır:

1. Veri tabanı üzerinde tarama sayısını azaltarak giriş/çıkış zamanını düşürmek.
2. Aday nesne küme sayısını azaltmak,
3. Daha düşük zamanda veri tabanı üzerindeki aday nesne kümelerinin desteklerini sayma,
4. Nesne küme üretimini parallelleştirme.

Bu anlamda bağıntı kurallar algoritmaları genellikle bir kaç yönden farklıdır:

1. Adayların üretilmesi,
2. Bir aday nesne kümesinin desteğinin hesaplanması,

3. Veri tabanı üzerinde tarama sayısı,
4. Kullanılan veri yapıları.

3.2.1. Bağıntı Kuralları ile İlgili Tanımlamalar

$I = \{I_1,\ldots,I_m\}$ nesne kümesi ve D bir veri tabanı olsun. Veri tabanındaki her bir kayıt T, $T \subseteq I$ olacak şekilde nesne kümesi içersin ve her bir kayıt KN olarak adlandırılan numaralarla gösterilsin.

Tanım 3.1. (Nesne kümesi): Bir X nesne kümesi I daki nesnelerin kümesidir. Bir nesne kümesi X, I dan k nesne içerirse k-nesne kümesi olarak adlandırılır.

Tanım 3.2. (Destek): Bir T kaydı, eğer $X \subseteq T$ ise bir nesne kümesini sağlar. D'deki bir X nesne kümesinin desteği, $Destek_D(X)$ ile gösterilir ve X'i sağlayan D'deki kayıtların sayısıdır.

Tanım 3.3. (Yoğun nesne kümesi): Eğer D'deki X'in desteği kullanıcı tarafından verilen minimum destek sınırını geçerse, X nesne kümesi yoğun nesne kümesi olarak adlandırılır.

Tanım 3.4. (Bağıntı kuralı): Bir bağıntı kuralı $X \Rightarrow Y$ şeklinde gösterilir. Burada $X \subset I$, $Y \subset I$ ve $X \cap Y = \phi$'dir. $X \Rightarrow Y$ kuralı bir c güvenlik değeriyle D'de geçerlidir. Bir kuralın güvenliği,

$$c = \frac{Destek_D(X \cup Y)}{Destek_D(X)} \tag{3.1}$$

ile verilir. Eğer D'deki kayıtların s kesimi $X \cup Y$'yi içerirse, $X \Rightarrow Y$ kuralı s desteğine sahiptir.

Tablo 3.1. Örnek veri tabanı

KN	Nesneler
100	A, B, C
200	B, D
300	A, C, D
400	A, B, D
500	A, B, D, E

Örnek 3.1. Tablo 3.1 de verilen örnek D veri tabanını gözönüne alınsın. Veri tabanında KN (Kayıt No) ile numaralandırılmış 5 tane kayıt olsun. Bu veri tabanı ayrıca nesne kümesi I={A,

B, C, D, E} 5 tane nesne veya öznitelik içersin. Bu durumda toplam $(2^5 - 1) = 32$ tane boş-olmayan nesne kümesi (*I* nin boş olmayan her bir alt kümesi bir nesne kümesidir). *A* 1-nesne kümesi, *AB* 2-nesne kümesi olacak şekilde devam eder. Veri tabanında 4 tane *A*'yı içeren kayıt bulunduğundan, $Destek_D(A) = 4$ dur. Eğer minimum destek %40 olarak alınırsa, o zaman {*A, B, C, D, AB, AC, AD, BD, ABD*} yoğun nesne kümeleridir. Çünkü bütün bu nesnelerin desteği, toplam kayıt sayısının %40'ından büyük veya eşittir. Geri kalanlar yoğun olmayan nesne kümeleridir. Eğer minimum güven (MinGüven) %60 olarak kabul edilirse, o zaman $A \Rightarrow D$ belirlenen MinDestek ve MinGüven'e göre bir bağıntı kuralıdır. Çünkü bu kuralın desteği %60 ve güveni %75 dir. Diğer bir yandan, $A \Rightarrow C$ güveninin %50 olması nedeniyle geçerli bir bağıntı kuralı değildir.

3.2.2. Bağıntı Kurallarını Çıkarma

Bir *D* veri tabanı verildikten sonra, kullanıcının belirlediği MinDestek ve MinGüven değerlerinden büyük veya eşit bütün bağıntı kuralları çıkarılabilir. Biçimsel olarak ifade edilmek istenirse, bütün $X \Rightarrow Y$ bağıntı kurallarını üretmek için;

$$Destek_D(X \cup Y) \geq MinDestek. \mid D \mid \text{ ve } \frac{Destek_D(X \cup Y)}{Destek_D(X)} \geq MinGüven \qquad (3.2)$$

şartları sağlanmalıdır.Bağıntı kurallarını çıkarma problemi iki kısıma ayrılabilir [7, 8]:

Adım 1: MinDestek olarak verilen belirli bir eşik değerininin üzerindeki desteğe sahip bütün nesne kombinasyonlarını üretme, (bu nesne kümeleri yoğun nesne kümeleri olarak adlandırılır).
Adım 2: Bağıntı kurallarını çıkarmak için yoğun nesne kümelerini kullanma. Öncelikle her yoğun *l* nesne kümesi için, *l*'nin boş-olmayan bütün alt-kümeleri bulunur. Her *a* alt-küme için, eğer $\frac{Destek_D(l)}{Destek_D(a)} \geq MinGüven$ ise $a \Rightarrow (l - a)$ şeklinde bir kural oluşturulur. Bir nesne kümesi ilk adımda yoğun olarak bulunursa, ikinci adımda kuralın güveni minimum güven değerinin üstünde olup olmadığı araştırılır.

L_k, yoğun *k*-nesne kümesini ve C_k, aday k-nesne kümesini göstermek üzere, aşağıda verilen Apriori algoritması bir veritabanında yoğun nesne kümelerini bulmak için kullanılan en yaygın yöntemdir [8].

Algoritma 3.1. (Apriori):

Giriş: Nesne kümeleri, MinDestek

Çıkış: Yoğun nesne kümeleri.

1. L_1 ={Yoğun 1-nesne kümeleri}
2. for(k=2; $L_{k-1} \neq \phi$; k++){

   C_k =aday_nesne(L_{k-1}); Yeni adaylar kümesini oluştur

   for her bir kayıt $t \in D$

   C_t =altküme(C_k , t); t'de içerilen adaylar

   for her bir aday $c \in C_t$ {

   c.sayıcı++;

   }

   L_k ={ $c \in C_k$ | $c.sayıcı \geq MinDestek$ }

   }
3. Cevap= $\bigcup L_k$

   }

Function *aday_nesne(L_{k-1})* {

L_{k-1} den C_k aday k-nesne kümesini oluştur.

$C_k = \phi$;

For her bir nesne kümesi $I_1 \in L_{k-1}$

For her bir nesne kümesi $I_2 \in L_{k-1}$

If (*(I_1-I_2)* ve *(I_2-I_1)*'nin her biri tek bir eleman içerirse) *then* {

c=Birleştir(I_1, I_2);

If c yoğun olmayan (k-1) alt kümeye sahipse *then*

c'yi sil

else *c*'yi C_k ya ekle

}

Return C_k;

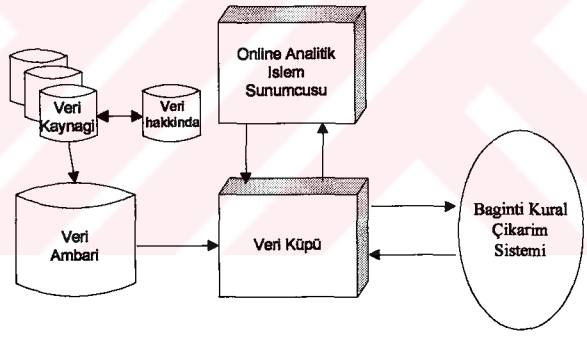
}

3.3. Çevrim-içi Analitik Bağıntı Kural Madenciliği

Veri ambarlarının artan önemi nedeniyle çok geniş miktarlardaki veritabanlarını etkili ve verimli bir şekilde analiz etmek için bir çok teknikler geliştirildi. Bunlar arasında çevrim-içi analitik işleme, çevrim-içi, hızlı ve etkili çok boyutlu veri analizi için geliştirilen önemli araçlardan biridir. Çevrim-içi analitik işleme daha detaylı bir analiz için toplanmış veriyi çok boyutlu olarak ele alan bir teknolojidir. Geleneksel veritabanı tekniklerinden çok daha hızlı büyük verileri sorgulayabilir. Çevrim-içi analitik işleme teknikleri kullanarak, büyük veritabanlarındaki ham veri çoklu boyut içerisinde organize edilir. Bu boyutların her biri de çoklu soyut seviyeleri içerebilir. Bu şekilde bir veri organizasyonu kullanıcılara farklı perspektiften verilere bakma esnekliği sağlar.

Genel olarak çevrim-içi analitik işleme 3 safhaya ayrılır:

1. Veri ambarından verileri seçme,
2. Veri küpünü oluşturma,
3. Küpü kullanarak çevrim-içi analiz yapma.



Şekil 3.1. Çevrim-içi analitik tabanlı bağıntı kural çıkarım sistemi

Çevrim-içi analitik tabanlı bir bağıntı kural çıkarım mimarisi Şekil 3.1'de gösterilmiştir. Mimari 3 ana kısımdan oluşur.

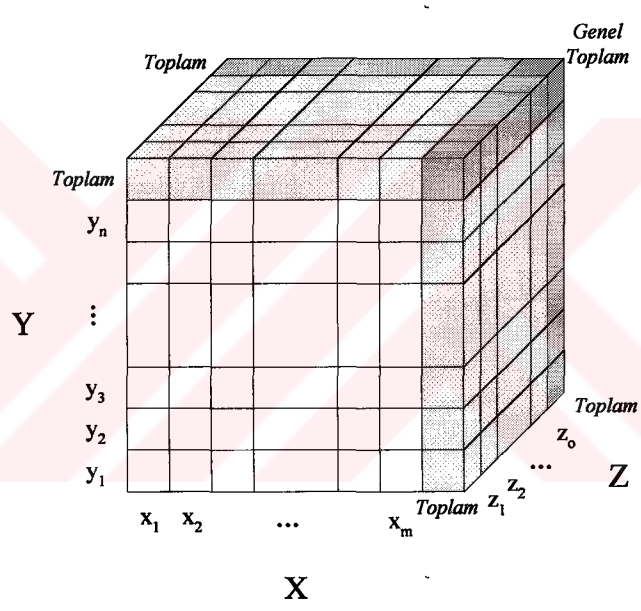
1. Veri ambarı,
2. Veri küpü ve çevrim-içi analitik işlem sunumcusu,
3. Bağıntı kurallarını çıkarım sistemi.

Çevrim-içi analitik işlem sunumcusunun ana görevi veri küpünü oluşturma, toplama, ayrıştırma ve seçme gibi kullanıcı komutlarını gerçekleştirmektir. Bağıntı kuralarını çıkarım sistemi ise oluşturulan veri küpünden uygun kuralları bulur.

Çevrim-içi analitik işlemde temel yapı veri küpüdür. Bir veri küpü aşağıdaki gibi tanımlanabilir:

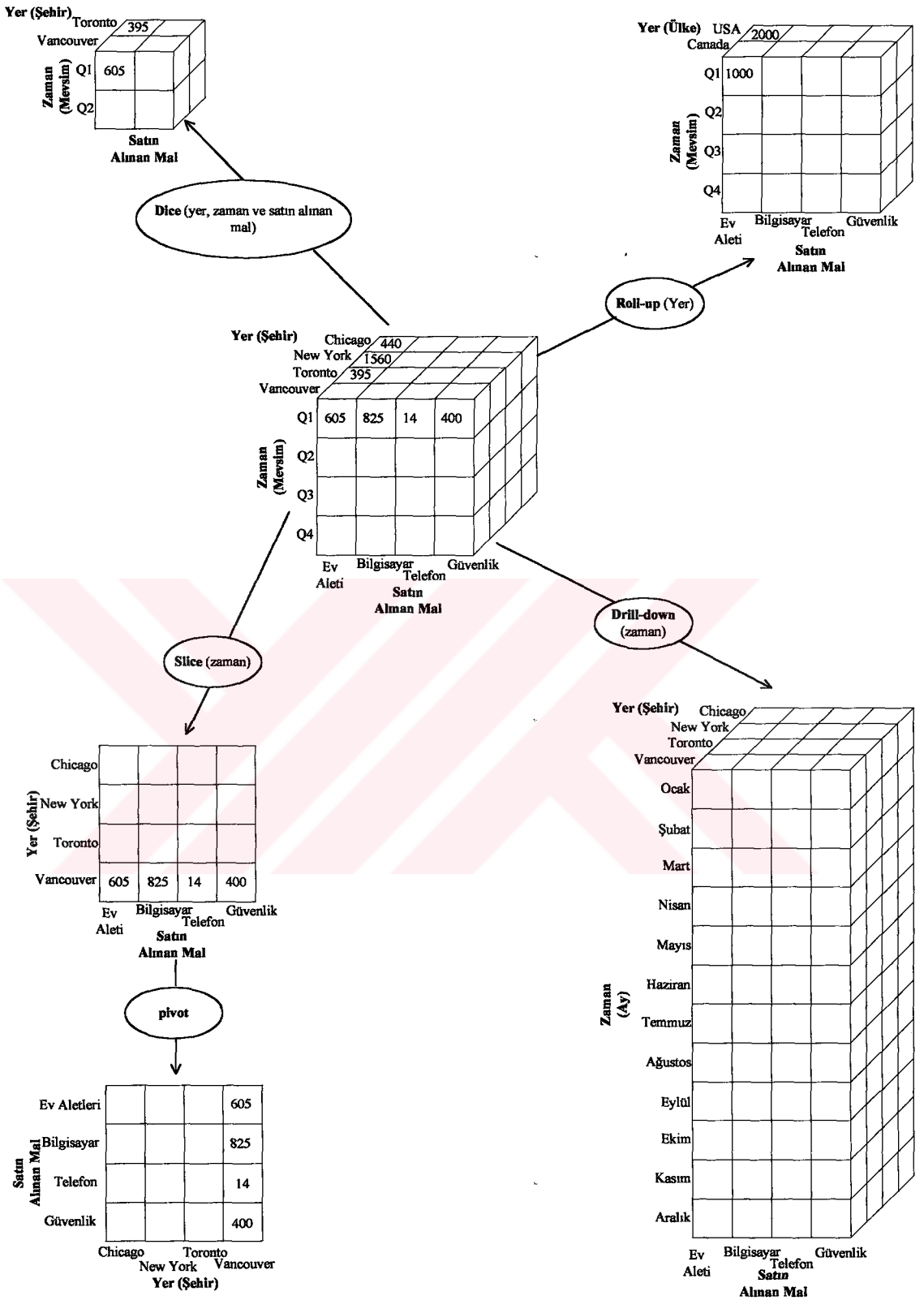
Tanım 3.5. (Veri küpü): Bir küp çok boyutlu değer dizilerine benzer bir veri organizasyonudur. Birkaç boyut üzerindeki ilişkiyi temsil eder. Buna göre, $|A_i|$, A_i boyutundaki farklı değerler sayısı olmak üzere küpün her bir boyutu $|A_i + 1|$ değer veya satır içerir. Her bir hücre boyutlar arasındaki ilişkilerden üretilen sayıcı değerini depolar. Formüldeki +1 değeri özel bir toplam değerini gösterir. Bu hücre önceki satırların toplam değerini verir. Bu toplam değerler veri küpü yapısının ana özelliklerinden birisidir.

Tanıma 3.5'e göre ifade edilen 3 boyutlu bir veri küpü Şekil 3.2'de verilmiştir. Burada her bir boyutun farklı değer sayısı sırasıyla m, n ve o dur.

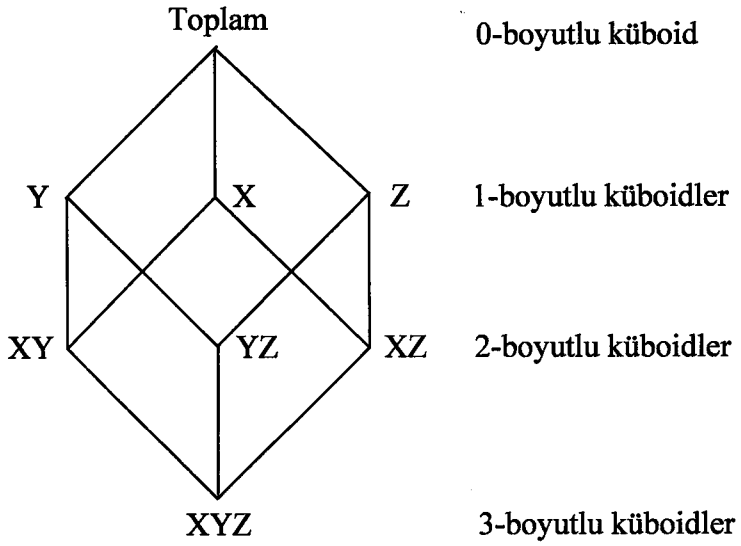


Şekil 3.2. Üç boyutlu veri küpü

Veri küplerini etkili bir şekilde gerçekleştirmek için bir çok hesaplama algoritmaları geliştirilmiştir [57, 58]. Ayrıca bir çok çevrim-içi analitik işlem operasyonları mevcuttur. Bunlar, toplama (roll-up), ayrıştırma (drill-down), verilen küpün bir boyutu üzerinde (slice) veya birden fazla boyutu üzerinde (dice) seçme ve döndürme (pivot)'dir. Şekil 3.3, örnek olarak verilen SATIŞ veri küpü üzerindeki çevrim-içi analitik işlem operasyonlarını gösterir.



Şekil 3.3. SATIŞ veri küpü üzerinde tanımlanan çevrim-içi analitik işlem operasyonları



Şekil 3.4. 3-boyutlu küpün küboid kafes yapısı

Bir kübteki veriler herhangi bir kavram seviyesinde olabilir. Genellikle boyutların ilk kavram seviyesine uygun veri küpü, taban küpoid olarak adlandırılır. Taban küboiddeki veriler yüksek seviyeli bir küboidi elde etmek için daha yüksek hiyerarşiye genelleştirilebilirler. Bir kübteki her boyut için en yüksek seviye *Toplam*'dır. Bir boyutu *Toplam*'a genelleştirmek onu küpten çıkarmak demektir. Bir veri küpünün bütün boyutları *Toplam* seviyesine genelleştirilirse, Tavan küboidi elde edilir. Tavan küboidi sadece bir hücreyi içeren 0-boyutlu bir küboiddir.

Şekil 3.4, boyutları X, Y, Z olan bir veri küpünün, küpoid kafes yapısını gösterir. Her bir boyut *Toplam* hariç sadece bir seviyeye sahiptir. XYZ Taban küpoidi üç boyutun tamamını içerir. Daha yüksek seviyeli XY, YZ ve XZ küboidleri, taban küboidinin bir boyutunun Toplam seviyesine genelleştirilmesi ile elde edilir. A, AB veya AC'den; B, AB veya BC'den; ve son olarak C, BC veya AC'den genelleşir.

Bir küboidin her bir boyutunda en fazla bir seviye görüleceğinden, n-boyutlu küp için küboidlerin toplam sayısı;

$$K = \prod_{i=1}^{n} (m_i + 1) \tag{3.3}$$

ile bulunur. Burada m_i, sanal seviye *Toplam* hariç i'ninci boyutun seviye sayısıdır. Bu formülden Şekil 3.4'deki küp için toplam küboid 2x2x2=8 bulunur.

3.3.1. Veri Küpü Üzerinde Çok Boyutlu Bağıntı Kurallarını Çıkarma

Çok boyutlu bağıntı kurallarını çıkarma yönteminde, korelasyon boyutlar arasındadır. Yani, bir kuralı biçimleyen nesneler farklı boyutlardan gelir. Bu maksat için Apriori'ye dayalı olarak geliştirilen algoritma aşağıda verilmiştir:

Algoritma 3.2. (Verilen n boyutlu bir küp üzerinde, boyutlar arası bağıntı kurallarını oluşturma):

Giriş: n boyutlu veri küpü VK[d_1, d_2, ...d_n], Minimum Destek (MinDestek) ve Minimum Güven (MinGüven)

Çıkış: n boyut arasında bulunan bağıntı kuralları.

1. $k=1$; $L=\phi$;
2. Her bir boyut için aday 1-nesne kümesini oluştur.Yani C_{1,d_i} ={d_i boyutundaki bütün farklı değerler}. $C_1 = \bigcup_{i=1}^{n} C_{1,d_i}$
3. L_1=*yoğun_nesne(1, C_1)*, L_1 yoğun nesne kümesini oluştur.
4. *While* ($L_k \neq \phi$) {

   $L = L \cup L_k$;

   $k=k+1$;

   C_k=*aday_nesne(k, L_{k-1})*; Aday k-nesne kümesini oluştur

   L_k=*yoğun_nesne(k, C_k)*; Yoğun k-nesne kümesini oluştur

   }
5. *bağıntı_kuralları(L)*; Bağıntı Kurallarını bul.

   }

Function *yoğun_nesne(k, C_k)* {

Aday kümesinden L_k k-yoğun nesne kümesini oluştur

$L=\phi$;

For her bir aday $I=\{i_1, i_2, ...i_k\} \in C_k$ {

Hücre_değeri=k boyutlu küpteki (i_1, i_2, ...i_k) hücresinin sayıcı değeri

$$Destek = \frac{Hücre\_Değeri}{GenelToplam}$$

If (Destek > MinDestek) then

$L_k = L_k \cup \{I\}$; I bir yoğun nesne kümesi

}

Return L_k;

}

Function *aday_nesne(k, L_{k-1}) {*

L_{k-1} den C_k aday k-nesne kümesini oluştur.

$C_k = \phi$;

For her bir nesne kümesi $I_1 \in L_{k-1}$

For her bir nesne kümesi $I_2 \in L_{k-1}$

If (*(I_1-I_2)* ve *(I_2-I_1)*'nin her biri tek bir eleman içerirse) *then {*

c=Birleştir(I_1, I_2);

If c yoğun olmayan (k-1) alt kümeye sahipse *then*

c'yi sil

else *c*'yi C_k ya ekle

}

Return C_k;

}

Function *Bağıntı_Kuralları(L){*

$R = \phi$;

For L'deki her bir L_i yoğun nesne kümesi

For L_i'nin boş olmayan her bir *S* alt kümesi için *{*

$$Güven((L_i - S) \Rightarrow S) = \frac{Destek(L_i)}{Destek(L_i - S)}$$

If (*Güven* $\geq$ *MinGüven*){

r="$(L_i - S) \Rightarrow S$" gibi ilginç bir kural oluştur.

$R = R \cup \{r\}$;

}

}

Return R;

}

3.3.2. Bulanık Veri Küpü

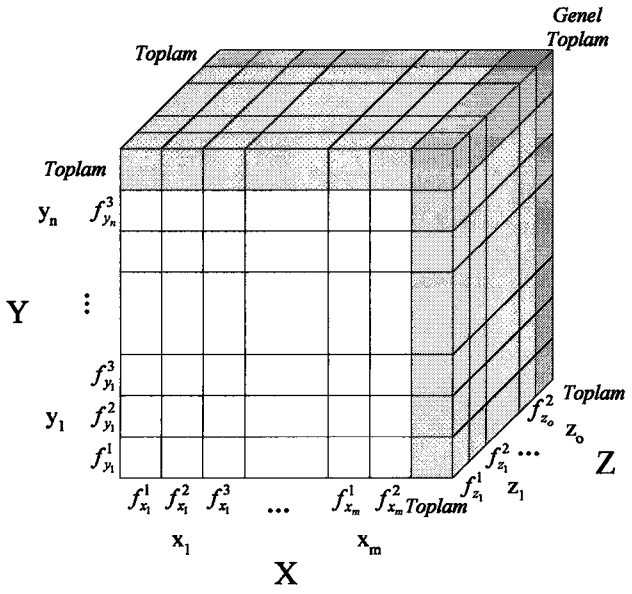
Bir veritabanında belirli aralıkta nicel değerler alabilen bir x özniteliği göz önüne alınsın. Bu durumda x'in her bir değerini bir veya daha fazla üyelik fonksiyonlarıyla niteleyecek şekilde bulanık kümeler tanımlamak mümkündür [59].

Tanım 3.6. (Üyelik derecesi): Bir x özniteliği ve $F_x = \{f_x^1, f_x^2, ...f_x^l\}$ x ile ilişkilendirilmiş l tane bulanık küme olsun. F_x'deki j'inci bulanık kümenin üyelik fonksiyonu, $\mu_{f_x}^j$ x'in alanından [0,1] aralığına bir eşlemeyi ifade eder. Biçimsel olarak; $\mu_{f_x}^j : D_x \rightarrow [0,1]$'dir. Eğer $\mu_{f_x}^j(v) = 1$ ise x'in v değeri bütünüyle f_x^j bulanık kümesine aittir. Diğer bir yandan $\mu_{f_x}^j(v) = 0$, x'in f_x^j bulanık kümesinin bir üyesi olmadığını gösterir. 0 ve 1 arasındaki bütün diğer değerler kısmi bir üyelik derecesini ifade eder.

Tanım 3.6'da tanımlanan kavram aşağıdaki gibi bir bulanık veri küpünü oluşturmada kullanılır.

Tanım 3.7. (Bulanık veri küpü): Boyutları $d_1, d_2, ...d_n$ olan n boyutlu bir veri küpünü ihtiva eden bir bağıntı kuralları çıkarma problemi verildikten sonra probleme özgü veri ambarından bir n boyutlu veri küpü oluşturulabilir. Küpün her bir boyutu $\sum_{i=1}^{k} l_i + 1$ değer içerir. Burada l_i, X boyutundaki öznitelik sayısıdır. Formüldeki son terim +1 bulanık olmayan veri küpünde olduğu gibi önceki satırların toplam değerini gösterir.

Üç boyutlu bir bulanık veri küpü Şekil 3.5'de verilmiştir. Bu veri küpünde her bir boyuttaki özniteliklerin üyelik fonksiyon sayısı 2 ile 3 arasında değişir.



Şekil 3.5. Üç boyutlu bulanık veri küpü

Tanım 3.8. (Paylaşım oranı): Üç boyutlu bulanık bir veri küpü verilsin. Eğer x_i özniteliği f_i^m bulanık kümesinde $\mu^m(x_i)$ üyelik derecesine sahipse, y_j özniteliği f_j^n bulanık kümesinde $\mu^n(y_j)$ üyelik derecesine sahipse ve z_k özniteliği f_k^o bulanık kümesinde $\mu^o(z_k)$ üyelik derecesine sahipse, o zaman ilgili hücredeki bütün bulanık kümelerin paylaşım oranı *po* aşağıdaki şekilde hesaplanır:

$$po = \mu^m(x_i).\mu^n(y_j).\mu^o(z_k) \tag{3.4}$$

Böylece her bir nesnenin yoğunluğu bulanık küpün bir hücresinden doğrudan elde edilebilir. Burada her bir hücre değeri boyut başına bir nesneyi içeren farklı nesnelerin üyelik derecelerinin çarpımıyla elde edilir. Buradan her bir hücrenin bulanık destek değeri;

$$Bul.Destek = \frac{PO}{GenelToplam} \tag{3.5}$$

formülü ile bulunur. Bu formülde *PO* ilgili hücreyle ilgili bütün *po*'ların toplamıdır.

3.3.3. Bulanık Bağıntı Kuralları

Bir bağıntı kuralı veri nesneleri arasındaki korelasyondur ve ikili ve sürekli değerli bağıntı kuralları olmak üzere ikiye ayrılır. Daha önce tanımı verilen ikili bağıntı kurallarından farklı olarak sürekli değerli bağıntı kurallarında nesneler nicel değerler içerir. Ayrıca gerçek hayattaki veritabanlarının çoğu da sürekli değerli özniteliklerden oluşur. Bu yüzden son zamanlarda sürekli değerli bağıntı kuralları üzerine birçok çalışma yapılmıştır. Bu çalışmalardan Srikant ve Agrawal [60] nicel değerlere sahip öznitelikleri öncelikle aralıklara ayırır ve öznitelik değerlerini ardışık tamsayılar kümesine eşler. Daha sonra, ikili bağıntı kurallarının çıkarılması için geliştirilen algoritmalar, dönüştürülmüş bu veri kümesine uygulanır. Fakat bu yöntemin bir dezavantajı vardır: Sürekli değerli bağıntı kuralındaki aralıklar keskin ve kullanıcı için yeterince anlamlı olmayabilir. Bulanık kümeler bir kümeye ait olma veya olmama arasında yumuşak bir geçiş sağlar. Böylece bulanık bağıntı kuralları, bulanık kümelerle ilişkilendirilmiş dilsel terimler nedeniyle kullanıcılar için kolaylıkla anlaşılabilirdir [61-69]. Buna göre bir bulanık bağıntı kuralı aşağıdaki gibi tanımlanır:

Tanım 3.9. (Bulanık bağıntı kuralı): Eğer $X=\{x_1,x_2,...x_p\}, A=\{f_1,f_2,...f_p\}$ ise o zaman $Y=\{y_1,y_2,...y_q\}, B=\{g_1,g_2,...g_q\}$ dur.

Yukarıda X ve Y yoğun nesne kümeleri olarak da adlandırılan özniteliklerin müşterek olmayan bir kümesidir.. Diğer bir değişle, $I=\{i_1,i_2,...i_m\}$, T veritabanındaki bütün öznitelikleri göstermek üzere $X \subset I$, $Y \subset I$ ve $X \cap Y = \phi$ dir. A ve B sırasıyla X ve Y'deki ilgili özniteliklerle ilişkili bulanık kümeler içerir. Yani, f_i, x_i ile ilgili bulanık kümeleri gösterirken g_j, y_j özniteliğiyle ilgili bulanık kümeleri verir. Son olarak "*X, A* ise" kuralın ilk kısmı olarak adlandırılırken, "*Y, B*'dir" kuralın sonuç kısmını ifade eder. Bir kuralın ilginç olabilmesi için, o kuralın kullanıcının belirlediği eşik değerinden daha yüksek destek ve güvene sahip olması gerekir.

Algoritma 3.3. (Verilen n boyutlu bir bulanık küp üzerinde, boyutlar arası bağıntı kurallarını oluşturma):

Giriş: n boyutlu bulanık veri küpü BVK[d_1, d_2, ...d_n], Minimum Destek (MinDestek) ve Minimum Güven (MinGüven)

Çıkış: n boyut arasında bulunan bulanık bağıntı kuralları.

1. $k=1$; $L=\phi$;

Her bir boyut için aday 1-nesne kümesini oluştur.Yani C_{1,d_i}={d_i boyutundaki bütün farklı değerler}. $C_1 = \bigcup_{i=1}^{n} C_{1,d_i}$. Örneğin; Şekil 3.5'de verilen küp için C_1={<(x_1, $f_{x_1}^1$), Toplam,Toplam>, <(x_1, $f_{x_1}^2$),Toplam,Toplam>, <(x_1, $f_{x_1}^3$),Toplam,Toplam>, ... <(x_2, $f_{x_2}^3$),Toplam,Toplam>, <(y_1, $f_{y_1}^1$),Toplam,Toplam>, <(y_1, $f_{y_1}^2$,Toplam,Toplam>, ...<(x_2, $f_{x_2}^2$), (y_2, $f_{y_2}^3$), (z_2, $f_{z_2}^3$)>}

2. L_1=*yoğun_nesne(1, C_1)*, L_1 yoğun nesne kümesini oluştur.
3. *While* ($L_k \neq \phi$) {

   $L = L \cup L_k$;

   k=k+1;

   C_k=*aday_nesne(k, L_{k-1})*; Aday k-nesne kümesini oluştur

   L_k=*yoğun_nesne(k, C_k)*; Yoğun k-nesne kümesini oluştur

   }
4. *bağıntı_kuralları(L)*; Bağıntı Kurallarını bul.

   }

Function *yoğun_nesne(k, C_k)* {

Aday kümesinden L_k k-yoğun nesne kümesini oluştur

$L = \phi$;

For her bir aday $I = (W, F) \in C_k$ { W nesne kümesi ve F, W ile ilişkilendirilmiş bulanık kümeler

Hücre_değeri=k boyutlu küpteki *(W, F)* hücresinin *PO* değeri

$$Bul.Destek = \frac{Hücre\_değeri}{GenelToplam}$$

If (Bul.Destek > MinDestek) then

$L_k = L_k \cup \{I\}$; I bir yoğun nesne kümesi

}

Return L_k;

}

Function *aday_nesne(k, L_{k-1})* {

L_{k-1} den C_k aday k-nesne kümesini oluştur.

$C_k = \phi$;

For her bir nesne kümesi $I_1 \in L_{k-1}$

For her bir nesne kümesi $I_2 \in L_{k-1}$

If ($(I_1\text{-}I_2)$ ve $(I_2\text{-}I_1)$'nin her biri tek bir eleman içerirse) *then* {

c=*Birleştir*(I_1, I_2);

If c yoğun olmayan (k-1) alt kümeye sahipse *then*

c'yi sil

else c'yi C_k ya ekle

}

Return C_k;

}

Function *Bağıntı_Kuralları*(L){

$R = \phi$;

For L'deki her bir L_i yoğun nesne kümesi

For L_i'nin boş olmayan her bir S alt kümesi için {

$$Bul.Güven((L_i - S) \Rightarrow S) = \frac{Bul.Destek(L_i)}{Bul.Destek(L_i - S)}$$

If ($Bul.Güven \geq MinGüven$){

r="$(L_i - S) \Rightarrow S$" gibi ilginç bir kural oluştur.

$R = R \cup \{r\}$;

}

}

Return R;

}

3.4. Sonuçlar

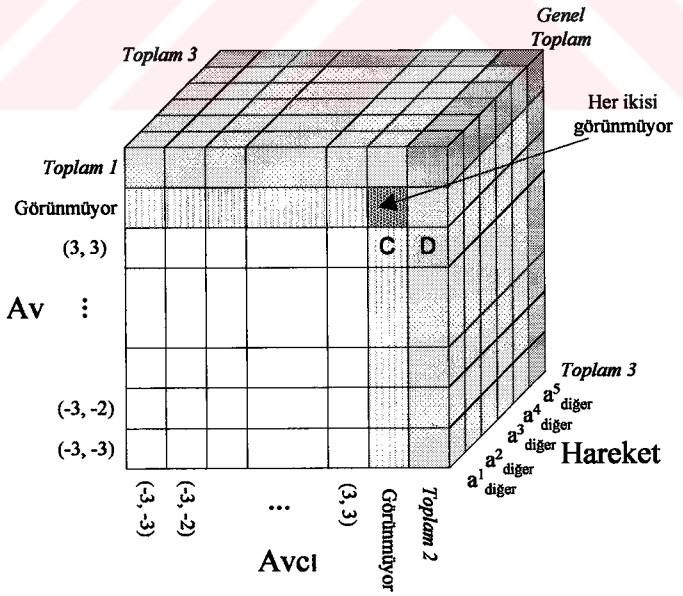
Çevrim-içi analitik işleme; çevrim-içi, hızlı ve etkili çok boyutlu veri analizi için en yaygın olarak kullanılan araçlardan biridir. Bununla birlikte şimdiye kadar çevrim-içi analitik işlem tekniklerini kullanan yöntemler genellikle ikili nesneler üzerine odaklanır. Buna karşın gerçek hayattaki veri tabalarının bir çoğu nicel değerler alabilen nesneler içerir. Ayrıca, böyle nesneler üzerinde bulanık küme teorisinin kullanımı, keşfedilen bilginin anlaşılabilirliğini artırır ve genelleştirilmiş kuralların bulunmasına imkan verir. Bu eksikliklerin üstesinden gelmek için bu bölümde, nicel değerler alan nesnelerdeki bilgi keşfi için bulanık veri küpü kullanan genel bir mimari önerildi. Ayrıca önerilen mimariden bulanık bağıntı kurallarının çevrim-içi madenciliği için bir algoritma sunuldu. Bu mimari ve algoritma daha sonraki bölümlerde etmenlerin durum ve hareket arasındaki ilişkilerini bulmak için kullanılacaktır.

4. ÇOKLU ETMEN ÖĞRENMEDE BAĞINTI KURALLARINI KULLANMA

4.1. Veri Küpünden İç Model Bağıntı Kurallarını Çıkarma

Daha önceki bölümlerde ifade edildiği gibi, çoklu etmen öğrenme üzerine yapılan çalışmaların çoğu ortamı sabit olarak ele alır. Yani, diğer etmenin davranışı ortamda göz önüne alınmaz. Halbuki dinamik ortamlarda etmen sürekli olarak öğrenir. Bu da ortamdaki her bir etmenin zamanla davranışını değiştirmesi demektir. Böyle bir durumda standart Q-öğrenme yaklaşımı uygun bir çözüm değildir. Bu problemi ele almak için bir etmenin iç modeli yöntemi önerildi [6, 24, 70]. Böyle bir yaklaşımda, etmen bir hareket yürütürken diğer etmenin hareketini de göz önüne alır.

Bu tezde, diğer etmenin hareketlerinin tutulduğu iç model veritabanı çevrim-içi analitik işlem mimarisi kullanılarak oluşturuldu. Bu maksat için kullanılan veri küpü Şekil 4.1'de gösterilmiştir. Küpün iki boyutu görsel alandaki av ve avcının durumuyla ilgili iken, üçüncü boyut diğer etmenin hareket uzayını temsil eder. Böyle bir küpte, öğrenme sürecinin sonuna kadar diğer etmenin gözlenen veya tahmin edilen bütün hareketleri güncelleştirilir. Bu şekilde her bir hücre mevcut durum ve gözlenen hareketten çıkarılan sayıcı değerini tutar.



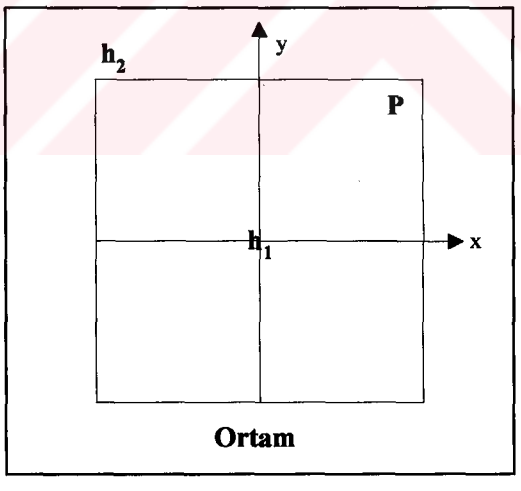
Şekil 4.1. Diğer etmenin iç modelini temsil etmek için önerilen veri küpü

Göz önüne alınan etmenin hareketini seçmede diğer etmenin hareket bağımlılığını açık bir yolla ifade etmek için, mevcut etmenin Q-fonksiyonu aşağıdaki tanıma göre açıklanır:

Tanım 4.1. (Avcı etmenin Q-fonksiyonu): h_2 avcı etmenin hareketini tahmin etmeye çalışan bir h_1 avcı etmeni verilsin. Bu durumda Q-fonksiyonu $Q(s, a_{kendi}, a_{diğer})$ olarak temsil edilir. Buradaki s, h_1 etmenin gözlemlediği durum; $a_{kendi}(\in A_{kendi})$ ve $a_{diğer}(\in A_{diğer})$ sırasıyla h_1 ve h_2 etmenlerinin olası bütün hareket kümesini gösterir.

Yukarıdaki tanıma göre, verilen bir etmenin hareketinin iyi veya kötü olduğuna karar verme diğer etmenin hareketine bağlıdır. Diğer bir değişle; $a_{diğer}$, a_{kendi} hareketini seçmede saklı ve önemli bir değişkendir. Tezin bu bölümünde diğer etmenin hareketi, geliştirilen veri küpünden çıkarılan bağıntı kuralları ile tahmin edilir. Eğer bir avcı, görme alanında diğer avcıyı gözlemlerse, $s \rightarrow a_{diğer}$ bağıntı kuralı, diğer etmenin geçmiş hareketlerinden kolaylıkla çıkarılabilir. Bununla birlikte, eğer avcı diğer avcıyı algılayamazsa, diğer avcının hareket yönü aşağıdaki öneriye göre tahmin edilir:

Öneri 4.1. (Görülmeyen avcının hareketini tahmin etme): Bir çoklu etmen ortamında h_1 ve h_2 avcı etmenleri ve P av etmeni bulunsun. Eğer h_1, h_2'yi gözlemleyemezse ve P, h_1'in görme alanının belirli bir karesinde ise (Şekil 4.2) h_2'nin hareketi; P yine aynı bölgede iken h_1'in görme alanında h_2 tarafından yapılan genel hareket eğilimine göre tahmin edilir.



Şekil 4.2. h_1 avcı etmenin görme alanı

Örneğin, avın [3, 3] karesinde ve diğer avcı etmeninde görme alanının dışında olduğu kabul edilsin. Bu durumda Şekil 4.1'de gösterilen veri küpünün C satırının bir hücresi D satırına göre güncellenir. Buradaki D satırı diğer etmenin pozisyonuna bakmaksızın avın [3, 3] karesinde olma sayısını verir.

Öğrenme sürecinin başlangıcında kullanıcı Şekil 4.1'de *Toplam3* olarak gösterilen hareket sayıcı için bir minimum destek değeri belirler. Eğer bir durumun sayıcı değeri minimum destek değerine erişirse, ilgili durumun yeterli şekilde tecrübelendiği kabul edilir. Böyle bir durumda göz önüne alınan avcı etmeni diğer etmenin hareketini en yüksek güven değerine göre tahmin eder. Buna karşılık eğer bir durum yeterince tecrübelenmemişse, avcı etmeni bu tahmini kullanıcının belirlediği güvenlik değerine göre yapar. Buna göre; bir durum-hareket çiftinin meydana gelme sayısı kullanıcının belirlediği minimum güven değerinden küçükse, ilgili hareket bu durumda seçilmez. Bir durumda minimum güven değerini aşan birden fazla hareket olması durumunda a_i hareketi aşağıdaki olasılığa göre belirlenir:

$$p(a_i \mid s) = \frac{Güven(s \rightarrow a_i)}{\sum_{a_j \in A(MinGüven)} Güven(s \rightarrow a_j)} \tag{4.1}$$

Yukarıdaki formülde, $Güven(s \rightarrow a_i)$, $s \rightarrow a_i$ kuralının güven değeridir. $A(MinGüven)$ ilgili etmen için minimum güven değerini aşan olası hareketler kümesidir. Bunu açıklamak için aşağıdaki tabloda öğrenme sürecinin belirli bir anında veri küpünün bir parçası verilmiştir. Her bir hücre ilgili durum hareket çiftinin meydana gelme sayısını gösterirken, Tablo 4.1'deki *Toplam3* kolonu her bir durumun gözlenme sayısını verir. Buna göre eğer minimum destek 1000 tecrübe olarak verilirse, Tablo 4.1'de verilen s_0 durumunun sayıcısı bu eşik değeri aştığından dolayı göz önüne alınan avcı etmen, diğer etmenin $a^4_{diğer}$ hareketini seçeceğini tahmin eder. Buna karşılık eğer s_1 durumu gözlenirse ve minimum güvenlik değeri %15 olarak belirlenirse, o zaman diğer etmenin hareketi $a^1_{diğer}$, $a^3_{diğer}$ ve $a^5_{diğer}$ hareketlerinden biri olarak tahmin edilir. Bunun yanında, Tablo 4.1'den kolaylıkla görülebileceği gibi $a^1_{diğer}$ hareketinin seçilme olasılığı diğerlerinden daha fazladır.

Tablo 4.1. Veri küpünün belirli bir anında gözlenen durum-hareket çiftlerinin meydana gelme sayıları

Hareket / Durum	$a^1_{diğer}$	$a^2_{diğer}$	$a^3_{diğer}$	$a^4_{diğer}$	$a^5_{diğer}$	*Toplam3*
s_0 ([-3, -3],[-3, -3])	322	154	48	415	128	**1067**
s_1 ([-3, -2],[-3, -3])	211	64	103	21	82	**481**
s_2 ([-3, -1],[-3, -3])	78	145	294	124	462	**1103**
		. . .				
s_n ([3, 3],[3, 3])	56	317	241	18	154	**786**

4.2. Veri Küpünden Bağıntı Kurallarını Çıkararak Çoklu Etmen Öğrenme

Bu bölümde geliştirilen veri küpünden çıkarılan bağıntı kuralları yardımıyla çoklu etmen öğrenme süreci anlatılacaktır. Veri küpünün boyutlarını durum bilgisi ve her iki avcı etmenin hareketleri teşkil eder. Bağıntı kurallarına dayalı çoklu etmen öğrenme algoritması aşağıda verilmiştir.

Algoritma 4.1. (Bağıntı kurallarına dayalı çoklu etmen öğrenme):

1. Göz önüne alınan avcı etmeni, şimdiki durum s'yi gözler ve veri küpünden çıkarılan bağıntı kurallarına dayanarak $a_{diğer}$ diğer avcı etmenin hareketini tahmin eder. Daha önceki bölümde ifade edildiği gibi, eğer s'nin gözlenme sayısı minimum destek sayısından daha büyük ise en yüksek güven değerli $a_{diğer}$ hareketi seçilir. Aksi halde, $a_{diğer}$ hareketi $p(a_i \mid s)$ olasılığına bağlı olarak s durumunda minimum güven değerini aşan hareketlerden seçilir.
2. Etmenin a_{kendi} hareketi, $a_{diğer}$ tahmin edilen harekete göre belirlenir. Örnek olarak; Tablo 4.2 öğrenme sürecinin belirli bir anında veri küpünün bir parçasından elde edilen değerleri gösterir. Burada her bir hücre ilgili durum hareket-çiftinin Q-değerini verirken, *Toplam* değişkeni ilgili durumda $a_{diğer}$'in gerçekleşme sayısını ifade eder. Önceki bölümde tartışılan bağıntı kurallarını çıkarma sürecine benzer şekilde, bir durum-$a_{diğer}$'in toplam değeri belirlenen minimum destek değerinden daha büyük veya eşitse ilgili durum ve $a_{diğer}$'in yeterince tecrübelendiği kabul edilir. Bu durumda göz önüne alınan avcı etmeni en yüksek güven değerli hareketi seçer.
3. Eğer durum-$a_{diğer}$ yeterince tecrübelenmemişse, avcı hareketini $p(a_i \mid s)$ olasılığına bağımlı olarak belirler.
4. Etmen Adım 2 veya Adım 3'de belirlenen hareketi yürütür.
5. Aynı anda diğer etmen $a^*_{diğer}$ hareketini gerçekleştirir.
6. Ortam yeni bir s' durumuna geçer.
7. Etmenler ortamdan r ödülünü alır ve veri küpünü güncelleştirirler.
8. Yeni s' durumu sonlanma şartını sağlarsa, mevcut deneme biter. Aksi halde, s', s'in yerine geçer ($s' \rightarrow s$) ve Adım 1'e geri dönülür.

Yukarıdaki algoritmaya göre minimum destek değeri 3000 olarak belirlenirse, Tablo 4.2'deki değerler aşağıdaki kuralların bulunmasına olanak sağlar.

$s_0 \wedge a^2_{diğer} \rightarrow a^4_{kendi}$, $s_0 \wedge a^5_{diğer} \rightarrow a^5_{kendi}$, $s_n \wedge a^1_{diğer} \rightarrow a^5_{kendi}$ ve $s_n \wedge a^4_{diğer} \rightarrow a^1_{kendi}$

Tablo 4.2. Veri küpünün belirli bir anında gözlenen durum-$a_{diğer}$ çiftlerinin meydana gelme sayıları

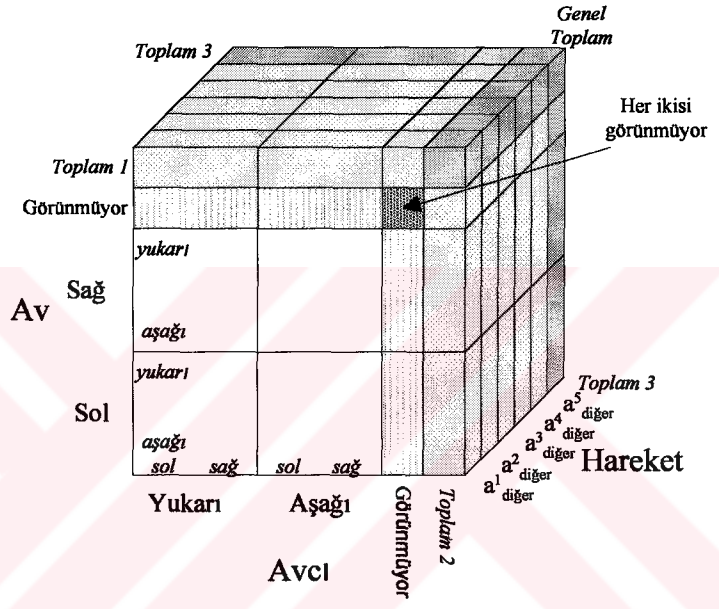
		a^1_{kendi}	a^2_{kendi}	a^3_{kendi}	a^4_{kendi}	a^5_{kendi}	***Toplam***
s_0	$a^1_{diğer}$	52.125	94.657	24.696	83.232	34.763	**1236**
	$a^2_{diğer}$	74.632	19.632	13.698	92.820	69.834	**3234**
	$a^3_{diğer}$	91.852	64.633	25.367	41.140	80.978	**2487**
	$a^4_{diğer}$	69.854	78.012	96.325	53.901	12.658	**2274**
	$a^5_{diğer}$	58.632	23.736	84.132	47.263	97.689	**4256**
. . .							
s_n	$a^1_{diğer}$	23.478	87.412	74.987	54.410	92.103	**5140**
	$a^2_{diğer}$	8.954	91.789	73.587	45.031	76.657	**978**
	$a^3_{diğer}$	72.683	54.312	37.189	87.205	10.002	**2140**
	$a^4_{diğer}$	94.127	51.978	7.014	46.879	34.478	**3106**
	$a^5_{diğer}$	66.798	30.412	87.631	12.163	86.156	**1358**

4.3. Veri Küpünden Çok Seviyeli Bağıntı Kurallarını Çıkarma

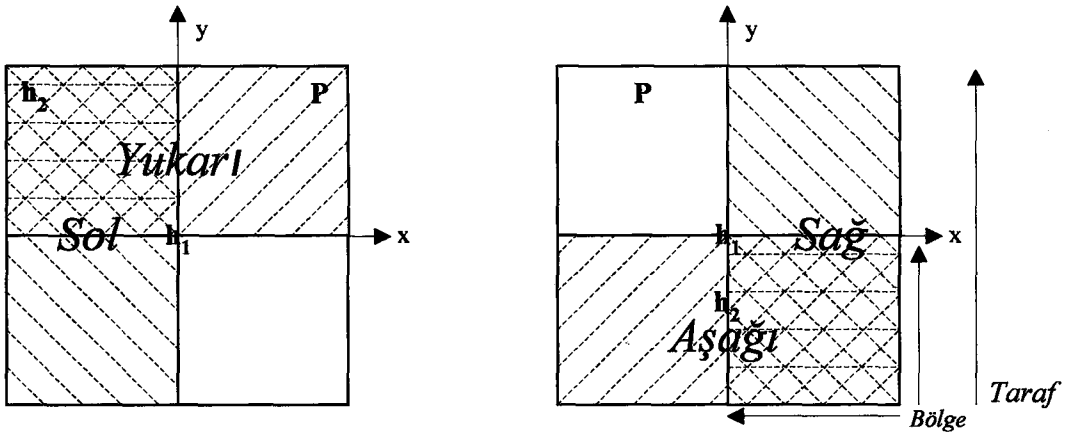
Bir çoklu etmen sisteminde etmen, ortamdan elde ettiği bilgiyi kullanarak kendi hareketini seçer. Bununla birlikte bazı durumlar için bu bilgi etmenlerin hareketlerini belirlemesi için kullandıkları bağıntı kurallarını çıkarmada yeterli değildir. Bu maksatla bilgi düşük seviyelerden daha yüksek seviyelere genelleştirilir. Şekil 4.1'de göz önüne alınan veri küpünün av ve avcının durumlarını gösteren iki boyutu için iki farklı hiyerarşisi Şekil 4.3'de gösterilmiştir. Bu hiyerarşilere göre av ve avcının durumlarını gösteren boyutları genelleştirilerek Şekil 4.4'deki yeni veri küpü elde edilir. Bu hiyerarşinin farklı seviyelerinde minimum destek değerinin belirlenmesi için farklı yöntemler kullanılır [16, 71]. Bu yöntemlerden biri ve de en çok kullanılanı, kural çıkarım sürecinde daha düşük seviyeler için azaltılmış minimum destek değerleri kullanmaktır [16]. Bu nedenle, bu tezde de daha düşük seviyeler için daha küçük minimum destek değerleri atanmıştır. Öğrenme süresi boyunca Şekil 4.3'deki her iki hiyerarşi etmenler tarafından durumları genelleştirmek için kullanılır. Hangi hiyerarşinin kullanılacağına karar verme göz önüne alınan etmenin durumuna bağlıdır. Şekil 4.5 avın iki farklı karede olması durumunda karşılaşılabilecek bazı genelleştirilmiş durumları gösterir.

Şekil 4.5'den kolaylıkla görülebileceği gibi, genelleştirmeler diğer etmenin işgal ettiği bölgeye göre yapılır. Eğer böyle bir bölge, genelleştirilmiş bölgeden daha fazlasıyla (taraf) çakışırsa, o zaman belirlenen tarafların tamamı uygun hareketi bulmada göz önüne alınır.

Şekil 4.3. Durum boyutları için çoklu seviyeler (Hiyerarşiler)



Şekil 4.4. Genelleştirmeden sonra elde edilen yeni veri küpü



Şekil 4.5. İki farklı durum için genelleştirilen bölgeler

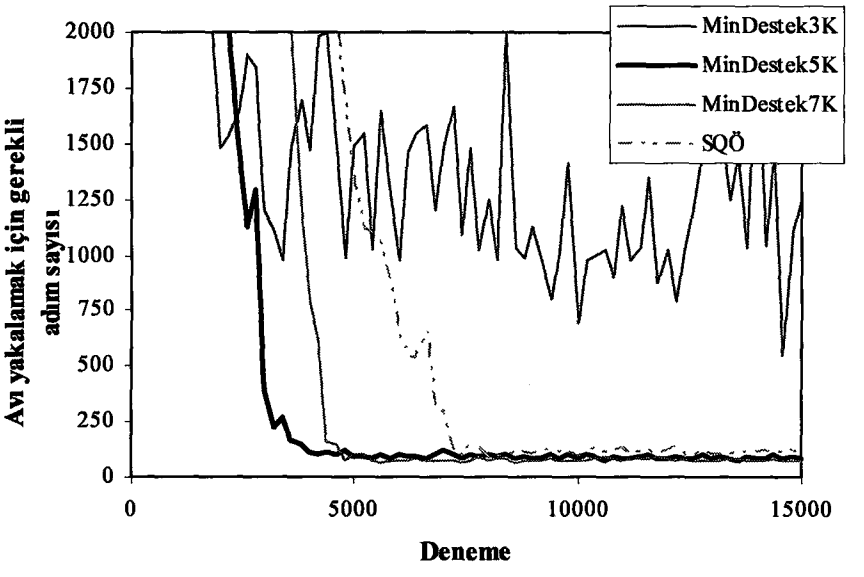
4.4. Uygulama Sonuçları

Bu bölümde önerilen yaklaşımları değerlendirmek, yani durumlar ve hareketler arasındaki ilişkiyi gösteren bağıntı kurallarını çıkararak öğrenmenin etkisini test etmek için bazı uygulamalar gerçekleştirildi. Bu uygulamalarda amaç, önerilen öğrenme yaklaşımını etkileyen ana faktörler minimum destek, minimum güven ve görme derinliğindeki değişimleri değerlendirmektir. Bütün uygulamalar Pentium III 1.4 GHz işlemci 512 RAM ve Windows 2000 işletim sistemli bir bilgisayarda gerçekleştirildi. Uygulamalarda öğrenme süreci denemelerden oluşur ve elde edilen her bir sonuç 10 farklı çalışmanın ortalama değeridir. Bu bölümdeki tecrübelerde ortam olarak 2 avcı etmen ve 1 av etmenli olan seçilmiştir (Şekil 2.2.(a)). Her bir deneme etmenlerin rasgele yerleşimiyle başlar ve avın yakalanması veya 2000 adım sayısına kadar devam eder. Avın yakalanmasıyla avcı etmenleri 100 değerli bir ödül alırlar. Bu ortamda, Q-öğrenme süreci için kullanılan parametreler aşağıdaki gibidir:

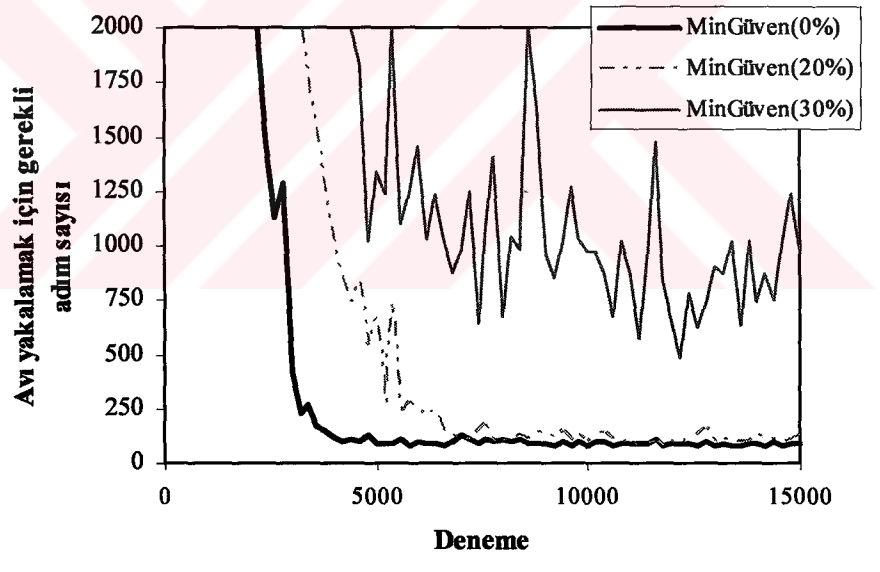
1. Öğrenme oranı $\beta = 0{,}8$
2. Azalma faktörü $\gamma = 0{,}9$
3. Q-fonksiyonun başlangıç değeri: 0,1
4. Etmenlerin görme derinliği: 3

İlk iki uygulamada av rasgele kaçma politikasını kullanır. İlk uygulama farklı minimum destek değerlerindeki öğrenme sürecini araştırır. Bu uygulamada her bir durumun gözlenme sayısı minimum destek değerine erişinceye kadar minimum güven değeri %0 olarak alınmıştır. Farklı minimum destek değerlerine göre avı yakalamak için gerekli adımların öğrenme eğrileri Şekil 4.6'da gösterilmiştir. Şekilden kolaylıkla görülebileceği gibi, minimum destek değeri 5K olarak belirlendiği durumdaki öğrenme eğrisi minimum destek 7K'lı eğriden daha hızlı yakın optimum çözüme yakınsar. Eğer minimum destek değeri 3K olarak kabul edilirse avcı etmenin hemen hemen hiç bir şey öğrenmediği görülür. Bu şekil aynı zamanda iç model yapılı standart Q-öğrenen (SQÖ) etmenlerin verilen ortamda nasıl bir davranış sergilediğini de açıklar. Şekilden açıkça görülür ki, önerilen yaklaşım standart Q-öğrenen etmenlere hem yakınsama hızı hem de adım sayısı bakımından üstünlük sağlar.

İkinci uygulamada minimum destek değeri 5K'ya sabitlendi. Bu durumda amaç, farklı minimum güven değerleri için öğrenme sürecini değerlendirmektir. Minimum desteğin 5K seçilme nedeni ise ilk uygulamada en iyi elde edilen sonuca uygun olmasındandır. Şekil 4.7'den görülebileceği gibi, ortamı keşfetmek, güven değerinin belirli bir seviyesine kadar önemlidir. Yani etmene ortamını keşfetme fırsatı verilmelidir. Buna göre güven değerinin %0 alındığı çözüm sadece daha hızlı yakınsamaz aynı zamanda daha az adım sayısında avı yakalar.



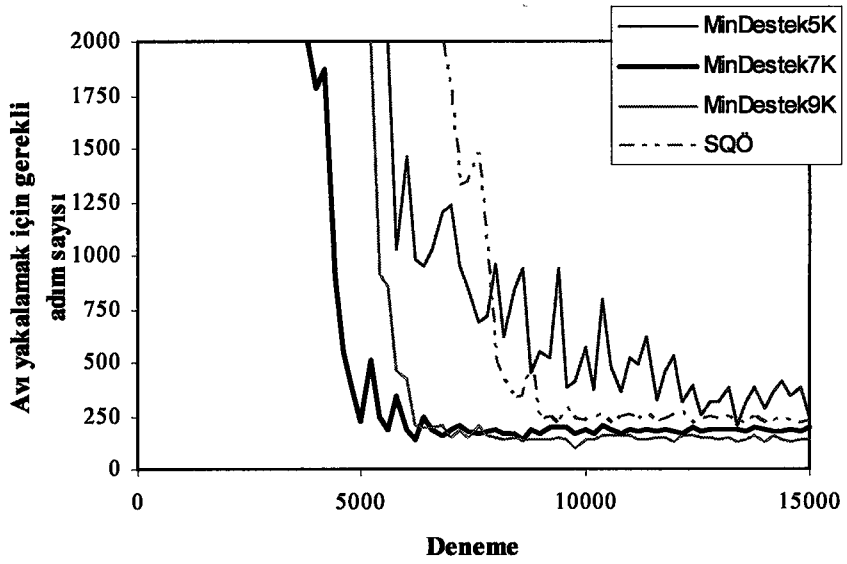
Şekil 4.6. Rasgele kaçan ava göre avcı etmenlerin öğrenme eğrileri



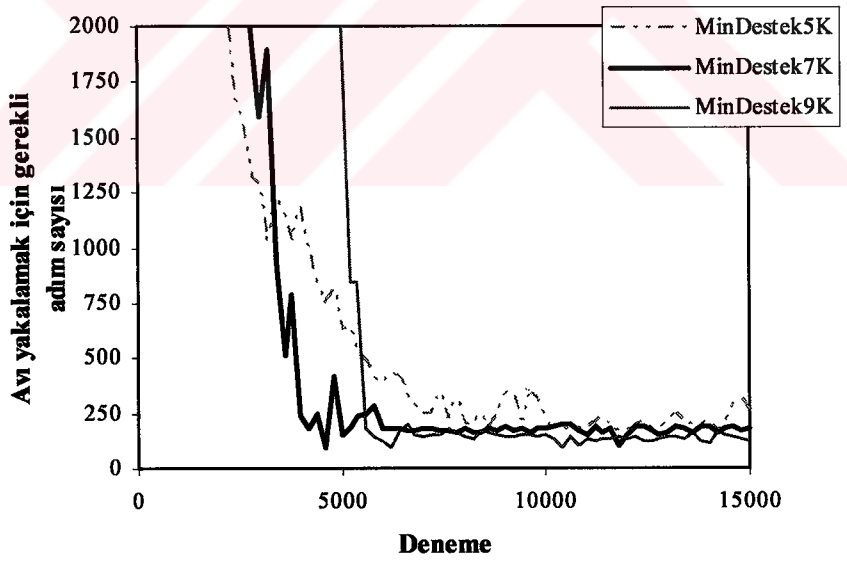
Şekil 4.7. Minimum desteğin 5K'ya sabitlenmesi durumunda avcı etmenlerin öğrenme eğrileri

Üçüncü uygulamada, av etmeni kaçma politikası olarak Manhattan-uzaklığı kullanır. Bu uygulamanın sonuçları Şekil 4.8'de gösterilmiştir. Önceki iki uygulama ile karşılaştırıldığında bu uygulamada her bir durumun daha fazla tecrübelenmesi gerektiği görülür. Bu da daha karmaşık ortamlar için minimum destek değerinin artırılması gerektiği sonucunu çıkarır. Bununla birlikte bu uygulamada minimum destek değerini 7K seçmek yeterlidir. Daha düşük

değerlerde ise avcı etmeni öğrenememektedir. Diğer yandan minimum destek değerini gereğinden fazla artırmak yakınsama hızını azaltır.



Şekil 4.8. Manhattan-uzaklık ile kaçan ava göre avcı etmenlerin öğrenme eğrileri

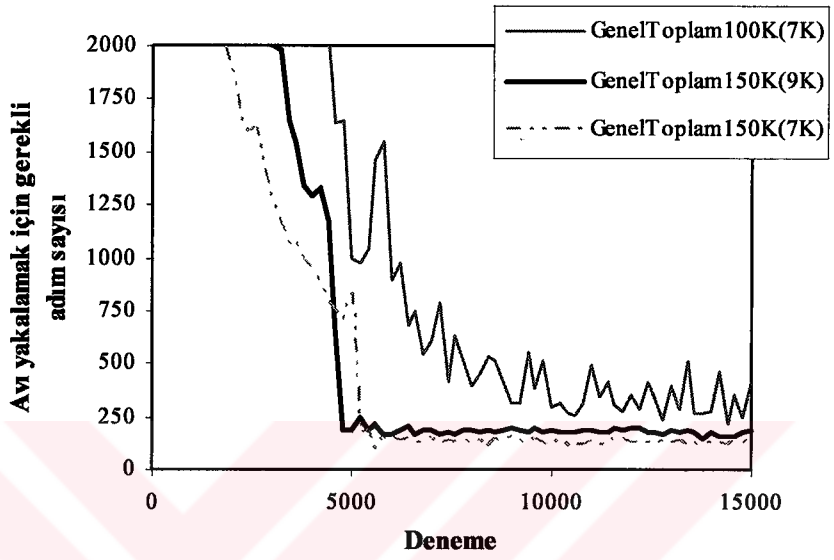


Şekil 4.9. Avcı etmenlerin görme derinliğinin 4'e yükselmesi durumunda öğrenme eğrileri

Dördüncü uygulama, öğrenme ve bağıntı kurallarını çıkarımda üzerine görme derinliğini artırmanın etkisini gözlemlemeye odaklanır. Bu maksatla avın kaçma politikası yine Manhattan-uzaklık iken, avcı etmenin görme derinliği 4'e yükseltildi. Şekil 4.9'da görüldüğü

gibi bu uygulamada avcı daha geniş bir alanı gördüğünden avı daha hızlı yakalar. Yakınsama adım sayısı da bir öncekine göre biraz daha düşüktür.

Bu bölümdeki son uygulama çoklu seviye durumlarıyla ilgilidir. Uygulama sonuçları Şekil 4.10'da verilmiştir. Bu uygulamada amaç, diğer durumlarla karşılaştırıldığında yeterince tecrübelenmemiş bir durumu genelleştirmektir. Böyle bir durumda etmen ortamını daha yüksek



Şekil 4.10. Genelleştirilmiş durumlu avcı etmenlerin öğrenme eğrileri

bir seviyeden gözlemler ve en iyi harekete karar verir. Şekil 4.10'daki eğriler için 3 farklı *GenelToplam* ve minimum destek değerleri belirlenmiştir. Eğer *GenelToplam* önceden belirlenen eşik değerine erişirse ve ziyaret edilen durumun destek değeri, minimum destek değerinden küçükse, etmen ortamdan daha detaylı bilgi almak için daha yüksek bir seviyeye çıkar.

4.5. Sonuçlar

Bu bölümde, bağıntı kurallarına dayalı, yeni bir çoklu etmen takviye öğrenme yöntemi önerildi. Bunun için öncelikle, ortamdan etmenler tarafından elde edilen bütün bilgiyi tutan bir veri küpü tanımlandı. Bu veri küpü kullanılarak, görme alanının dışında olması durumunda bile diğer avcı etmenin hareketi tahmin edildi. Daha sonra, yeterince tecrübelenmemiş durumları genelleştirmek için bir yöntem önerildi. Son olarak, etmenlerin en uygun hareketi yapması için yeni bir hareket seçme yaklaşımı sunuldu. Bir çoklu etmen ortamı üzerinde yapılan uygulama sonuçları, önerilen öğrenme yaklaşımının, yüksek kaliteli optimum çözümleri daha hızlı yakalamak için etkili bir şekilde kullanılabileceğini gösterdi.

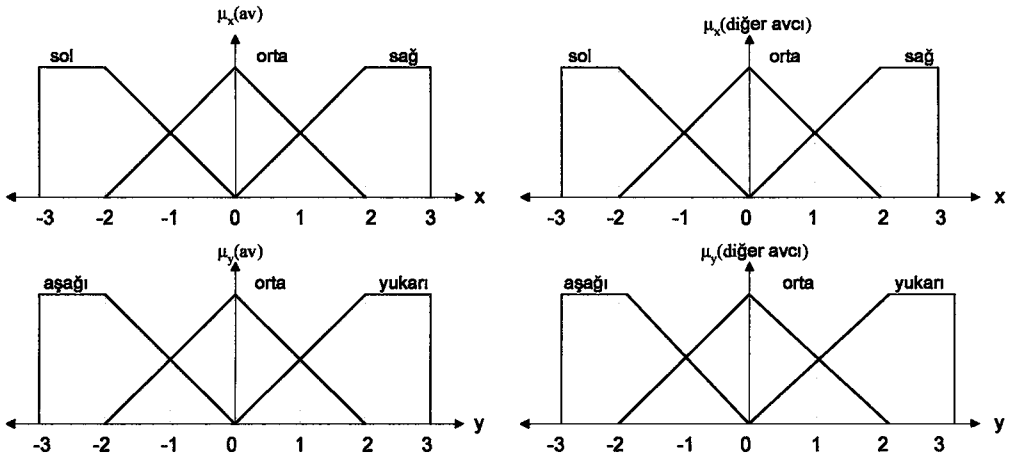
5. BULANIK BAĞINTI KURALLARI KULLANARAK ÇOKLU ETMEN ÖĞRENME

5.1. Çoklu Etmen Sistemlerde Durum ve Hareket Uzayının Bulanık Kümelerle Temsil Edilmesi

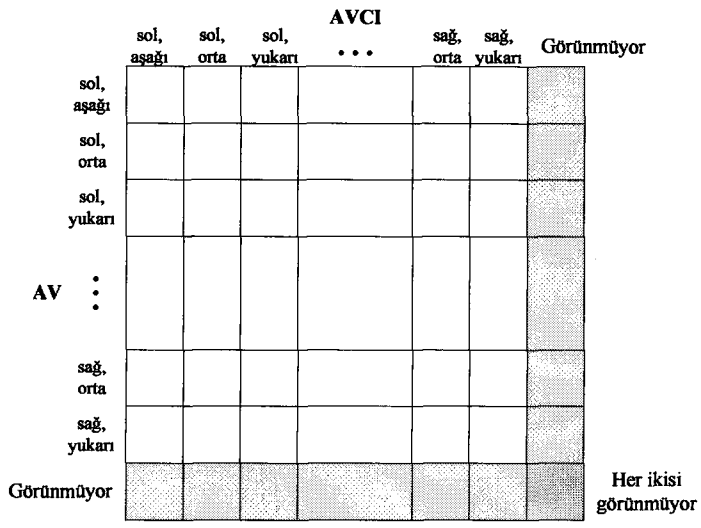
Çok geniş ve sürekli durum uzaylı ortamlardaki durum sayısı problemine çözüm olarak durumları genelleştirme yöntemine gidildiği daha önce anlatılmıştı. Bu yöntemlerden biri olan bulanık kümeler kullanarak problemin durum uzayı azaltılabilir ve buna paralel olarak daha hızlı bir yakınsama gerçekleştirilebilir.

Şekil 5.1 bir etmenin durum uzayının bulanık kümeler kullanarak temsil edilmesi durumunda elde edilecek üyelik fonksiyonlarını gösterir. Dikkat edilirse her bir değişken, düzenli üyelik fonksiyonlarına sahiptir. Değişkenlerin tanım aralıkları [-3, 3] ile sınırlandırılmıştır. Çünkü etmenin görme derinliği şekilde 3 olarak kabul edilmiştir. Ortam bulanık kümler ile modellendiğinden avın veya ortamdaki diğer avcının yeri birden daha fazla durum içerebilir. Örneğin; eğer avcı sadece avı [-1, -1] bölgesinde gözlemlerse eşit ağırlıklı dört farklı durumu elde etmiş olur. Bunlar [(sol, aşağı), görünmüyor], [(sol, orta), görünmüyor], [(orta, aşağı), görünmüyor] ve [(orta, orta), görünmüyor]'dur. Bütün bu durumların ortak ağırlığı 0.25'dir ve diğer avcı, göz önüne alınan avcının görme alanının dışındadır.

Bir etmenin durumu av ve/veya diğer avcının x ve y eksenlerine göre belirlenir. Örneğin; eğer bir av x yönüne göre [-3, 0] aralığında algılanırsa av, etmenin sol tarafında gözlemlenmiştir. Benzer şekilde, eğer y yönüne göre [0, 3] aralığında algılanırsa o zaman av, etmenin üzerindedir. Bu durumda, hareket eden bir avcı etmenin karşılaşabileceği bütün durumlar Şekil 5.2'de verilmiştir.



Şekil 5.1. Bir avcının durumunu temsil eden bulanık üyelik fonksiyonları



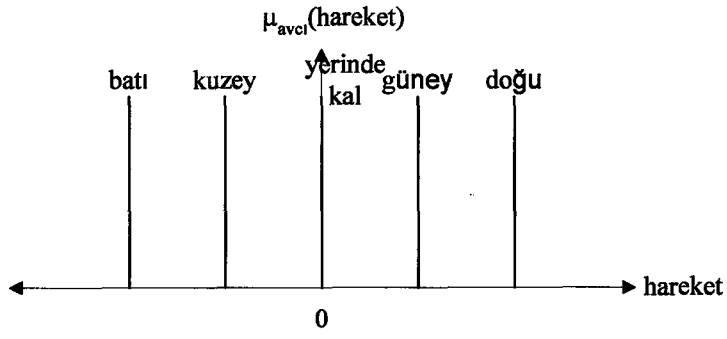
Şekil 5.2. Bir avcı etmenin karşılaşabileceği olası durumlar

Öneri 5.1. (Durum sayısı): Bir çoklu etmen ortamında iki avcı etmeni ve bir av etmeni bulunsun. Bu durumda her bir değişkeni 3 üyelik fonksiyonuyla temsil edilen bir etmenin gözlemleyeceği toplam durum sayısı 100'dür.

İspat: 100=9*9+9+9+1. Burada 9*9 av ve diğer avcı etmenin her ikisinin göz önüne alınan avcı etmenin görme anında iken elde edilen durum sayısı, 9+9 sadece avın veya diğer avcı etmenin gözlemlenmesi anında elde edilen durum sayısıdır. 1 ise hiçbir etmenin gözlemlenmediği durumu temsil eder.

Bir avcı etmenin hareket uzayını temsil eden üyelik fonksiyonları Şekil 5.3'de gösterilmiştir.

Durumların giriş değerleri elde edildikten sonra, bulanık kümeler içerisine gruplandırılır. Diğer bir değişle, giriş değerleri dilsel değerlere dönüştürülür.



Şekil 5.3. Bir avcının hareket uzayını temsil eden bulanık üyelik fonksiyonları

Durum ve hareket uzayının yukarıda tanımlanan üyelik fonksiyonlarına göre, Şekil 5.4 iç model veri tabanını temsil eden üç boyutlu bir bulanık küpü gösterir. Küpün iki boyutu avcı ve avın görme alanındaki durumlarıyla ilgili iken, üçüncü boyut diğer avcının hareket uzayını verir. Durum uzayı ile ilgili boyutlar x ve y eksenlerini [sol, aşağı] veya [sol, orta] şeklinde birlikte ele alır. Burada sol x-eksenindeki bölge iken, aşağı veya orta y-eksenindeki bölgeyi ifade eder.

Küpteki her bir hücre çevrim-içi analitik işlem bölümünde anlatıldığı gibi boyutlar arası paylaşım oranını gösterir.

Öneri 5.2. (Paylaşım oranı hesaplama): 3 boyutlu bir bulanık veri küpü verilsin. Eğer diğer avcının bölgesi x ve y eksenleri için $\mu_x(diğer\ avcı)$ ve $\mu_y(diğer\ avcı)$ üyelik derecelerine sahipse, avın bölgesi x ve y eksenleri için $\mu_x(av)$ ve $\mu_y(av)$ üyelik derecelerine sahipse ve diğer avcının tahmin edilen hareketi $\mu_{hareket}(diğer\ avcı)$ üyelik derecesine sahipse; o zaman ilgili hücre için bütün bulanık kümelerin paylaşım oranı;

$$po = \mu_{durum}(diğer\ avcı).\mu_{durum}(av).\mu_{hareket}(diğer\ avcı) \tag{5.1}$$

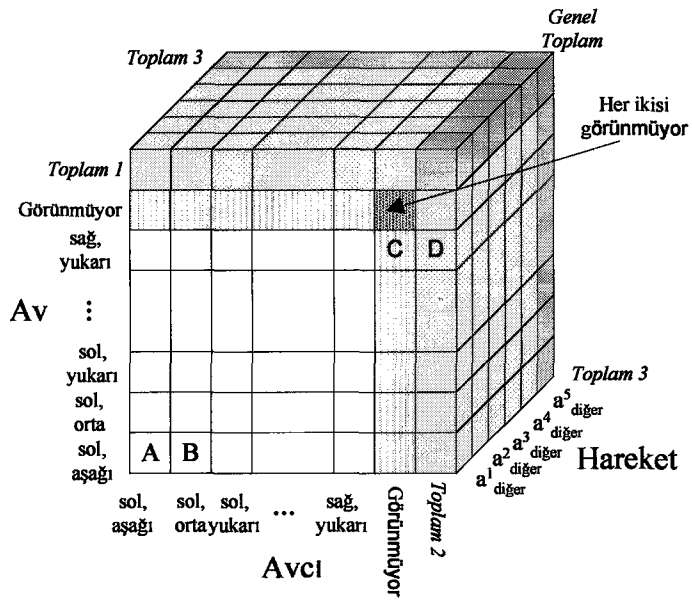
şeklinde hesaplanır. Burada,

$\mu_{durum}(diğer\ avcı) = \mu_x(diğer\ avcı).\mu_y(diğer\ avcı)$ ve $\mu_{durum}(av) = \mu_x(av).\mu_y(av)$ dır.

Örnek olarak, avın ve avcının sırasıyla [-2, -2] ve [-2, -1] karelerinde gözlenmesi ve diğer avcının tahmin edilen hareketinin $a^1_{diğer}$ olması durumunda Şekil 5.4'de gösterilen A ve B hücrelerinin değerleri sırasıyla 1x0.5x1=0.5 ve 1x0.5x1=0.5 olarak hesaplanır.

Daha önceki bölümlerde açıklandığı gibi, çoklu etmen öğrenen sistemlerde durum sayısını azaltmak için kullanılan yöntemlerden biri de durumları genelleştirmektir. Q fonksiyonu $Q(S, a_{kendi}, a_{diğer})$ olarak ifade edilen bir etmenin durumu bulanık kümeler kullanarak modellendiğinden; etmen, üyelik dereceleri farklı birden fazla durumu aynı anda gözlemleyebilir. Biçimsel olarak, $S = (s_1, s_2, ...s_n)$ dır. Burada n, etmenin mevcut konumda gözlemlediği durum sayısıdır.

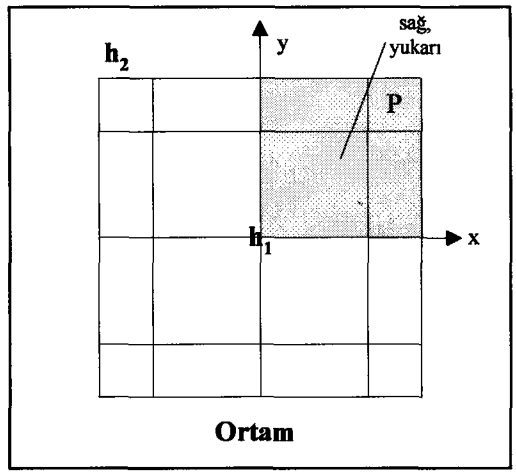
Bir etmen S durumunda iken Q-değerleri iki şekilde güncellenebilir. İlkinde üyelik dereceleri göz önüne alınmaksızın S'nin bütün bileşenleriyle ilgili Q-değerleri güncellenir [49].



Şekil 5.4. Çoklu etmen öğrenme için kullanılan bulanık veri küpü

İkinci yöntemde ise Q-değeri güncellenirken üyelik değerleri de dikkate alınır. Çünkü bu yönteme göre birden fazla durumu gözleyen etmen için her bir durumun ağırlığı diğerinden farklıdır. Bu şekilde durum uzayının bulanık kümeler kullanarak modellenmesi durumunda Bölüm 4'de verilen görünmeyen avcının hareketini tahmin etme önerisi aşağıdaki gibi değişikliğe uğrar.

Öneri 5.3. (Bulanık bir ortamda görülmeyen avcının hareketini tahmin etme): Bulanık bir ortamda iki avcı etmen h_1 ve h_2 ile bir P av etmeni bulunsun. Eğer h_1; P h_1'in görme alanının herhangi bir bölgesinde iken h_2'yi gözlemleyemezse, h_2'nin hareketi, P aynı bölgede iken h_1'in görme alanında yaptığı hareketlerin genel eğilimine göre tahmin edilir.



Şekil 5.5. Bulanık bir ortamda h_1'in görme alanı

Şekil 5.5 avın [3, 3] karesinde, diğer avcının ise h_1'in görme alanının dışında olması durumunu gösterir. Böyle bir durumda Şekil 5.4'deki bulanık veri küpünün C satırının bir hücresi D satırına göre güncellenir. Burada D satırı bulanık olmayan veri küpünde olduğu gibi diğer avcı etmenin durumuna bakmaksızın avın [sağ, yukarı] bölgesinde bulunma sayısını verir. Buna karşılık av etmeni [3, 3]'de değil de [1, 1] karesinde bulunursa bu durumda av etmeninin bulanık veri küpünde [orta, orta], [orta, yukarı], [sağ, orta] ve [sağ, yukarı] durumuna karşılık gelen bütün satırları için görünmeyen avcının hareketi tahmin edilir ve genel S durumunun hareketi her bir $s_k (\in S)$ için tahmin edilen hareketlerin maksimum paylaşım oranlı olanıdır. Biçimsel olarak S durumunda $a_{diğer}$ hareketi $\max_{s_k \in S} PO(s_k, a_{diğer})$'e göre belirlenir.

Bir bulanık veri küpünden $a_{diğer}$ hareketini tahmin etmek için Bölüm 4'de verilen algoritmaya benzer şekilde, kullanıcı öğrenme sürecinin başlangıcında hareket sayıcı için bir minimum destek noktası belirler. Eğer bir s_k durumunun sayıcı değeri bu destek noktasına erişirse, o durumun yeterince tecrübelendiği ve dolayısıyla o durumda bir hareketin kesin olarak belirlendiği kabul edilir. Buna karşılık durum yeterince tecrübelenmemişse a_i hareketi $p(a_i \mid s_k)$ olasılık formülüne göre minimum güven değerini aşan hareketlerden seçilir. Örnek olarak Tablo 5.1 öğrenmenin belirli bir anında bulanık veri küpü parçasını göstersin ve aynı zamanda bir avcı etmeni $S(s_0, s_1, s_2)$ durumunu gözlemlesin. Eğer minimum destek önceden 1000 olarak belirlenmişse avcının gözlemlediği s_0'da diğer etmenin hareketi kesin bir şekilde $a^4_{diğer}$ olarak tahmin edilir. Benzer biçimde s_2 için $a^5_{diğer}$'dir. Buna karşılık s_1 durumu yeterince tecrübelenmemiş olduğundan göz önüne alınan etmen, diğer etmenin hareketini paylaşım oranının büyüklüğü ölçüsündeki bir olasılıkla tahmin eder. Örnek olarak, s_1 için $a^3_{diğer}$ bulunabilir.

Tablo 5.1. Öğrenmenin belirli bir anında bulanık iç model veri küpündeki durum-hareket sayıları

Hareket / *Durum*	$a^1_{diğer}$	$a^2_{diğer}$	$a^3_{diğer}$	$a^4_{diğer}$	$a^5_{diğer}$	***Toplam3***
s_0 (sol, aşağı-sol, aşağı)	322.30	154.25	48.15	415.20	128.10	**1068**
s_1 (sol, aşağı-sol, orta)	211.15	64.20	103.20	21.25	82.20	**482**
s_2 (sol, aşağı-sol, yukarı)	78.10	145.05	294.65	124.10	462.10	**1104**
		. . .				
s_n (sağ, yukarı-sağ, yukarı)	56.15	317.15	241.60	18.05	154.05	**787**

5.2. Bulanık Veri Küpünden Çoklu Etmen Bağıntı Kurallarını Çıkarma

Bu bölümde çevrim-içi analitik işlemli bulanık bağıntı kurallarını çıkararak çoklu etmen öğrenme gerçekleştiren algoritma sunulacaktır.

Algoritma 5.1. (Bulanık bağıntı kuralları tabanlı çoklu etmen öğrenme):

1. Göz önüne alınan avcı, mevcut S durumunu gözlemler ve bulanık veri küpünden bağıntı kurallarını çıkararak diğer avcının $a_{diğer}$ hareketini tahmin eder.

Tablo 5.2. Bulanık veri küpünün belirli bir anında gözlenen durum-$a_{diğer}$ çiftlerinin meydana gelme sayıları

		a^1_{kendi}	a^2_{kendi}	a^3_{kendi}	a^4_{kendi}	a^5_{kendi}	***Toplam***
s_0	$a^1_{diğer}$	52.125	94.657	24.696	83.232	34.763	**1236**
	$a^2_{diğer}$	74.632	19.632	13.698	92.820	69.834	**3234**
	$a^3_{diğer}$	91.852	64.633	25.367	41.140	80.978	**2487**
	$a^4_{diğer}$	69.854	78.012	96.325	53.901	12.658	**2274**
	$a^5_{diğer}$	58.632	23.736	84.132	47.263	97.689	**4256**
				. . .			
s_n	$a^1_{diğer}$	23.478	87.412	74.987	54.410	92.103	**5140**
	$a^2_{diğer}$	8.954	91.789	73.587	45.031	76.657	**978**
	$a^3_{diğer}$	72.683	54.312	37.189	87.205	10.002	**2140**
	$a^4_{diğer}$	94.127	51.978	7.014	46.879	34.478	**3106**
	$a^5_{diğer}$	66.798	30.412	87.631	12.163	86.156	**1358**

Eğer $s_k (\in S)$'in gözlenme sayısı minimum destek değerinden daha büyükse, en yüksek güven değerli $a_{diğer}$ hareketi seçilir. Buna karşılık s_k'nın gözlenme sayısı minimum destek değerinden daha küçükse, $p(a_i \mid s_k)$ olasılığına göre minimum güven değerini aşan hareketlerden biri seçilir.

2. Bundan sonra a_{kendi} hareketi, $a_{diğer}$'e göre belirlenir. Bu işlem bulanık iç model bağıntı kurallarını çıkarmaya benzer. Örnek olarak Tablo 5.2 öğrenme sürecinin belirli bir anında bulanık veri küpünden elde edilen değerleri gösterir. Her bir hücre ilgili durum ve hareketlerin Q-değerini verirken *Toplam* değişkeni ilgili durumun belirli bir $a_{diğer}$

hareketinde bulanık olarak gözlenme sayısını temsil eder. Gözlenen durumda $a_{diğer}$ yeterince tecrübeli ise a_{kendi}, en yüksek güven değerli olandır.

3. Eğer durum-$a_{diğer}$ yeterince tecrübelenmemişse, etmen $p(a_i \mid s_k)$ olasılığını kullanarak mevcut hareketlerden birini seçer. Adım 2 gözlenen S durumunun her bir s_k bileşeni için tekrarlanır. Bütün bileşenlerin hareketi belirlendikten sonra, S durumunun hareketi aşağıdaki şekilde bulunur:

$$\arg \max_{a_{kendi} \in A_{kendi}} Q(s_k, a_{kendi}, a_{diğer})$$

4. Göz önüne alınan avcı etmen Adım 2 veya 3'de seçilen hareketi yürütür.
5. Aynı anda diğer avcı $a^*_{diğer}$'i yürütür.
6. Ortam yeni bir S' durumuna geçer.
7. Avcı etmenler ortamdan r ödülünü alırlar ve bulanık veri küpü güncellenir. $Q(s_k, a_{kendi}, a^*_{diğer})$ hücrelerinin güncellenmesi aşağıdaki biçimde olur.

$$\begin{aligned} Q(s_k, a_{kendi}, a^*_{diğer}) \leftarrow & [1-\beta.\mu_{durum}(av).\mu_{durum}(diğer\ avcı)].Q(s_k, a_{kendi}, a^*_{diğer}) \\ & +\beta.\mu_{durum}(av).\mu_{durum}(diğer\ avcı)[r+\gamma. \max_{a'_{self} \in A_{self}} Q(s'_k, a'_{kendi}, a'_{diğer})] \end{aligned} \quad (5.2)$$

Yukarıdaki formülde $a'_{diğer} = \arg \max_{a'_{diğer} \in A_{diğer}} Güven(s'_k, a'_{diğer})$

$\mu_{durum}(av) = \mu_x(av).\mu_y(av)$ ve $\mu_{durum}(diğer\ avcı) = \mu_x(diğer\ avcı).\mu_y(diğer\ avcı)$'dir.

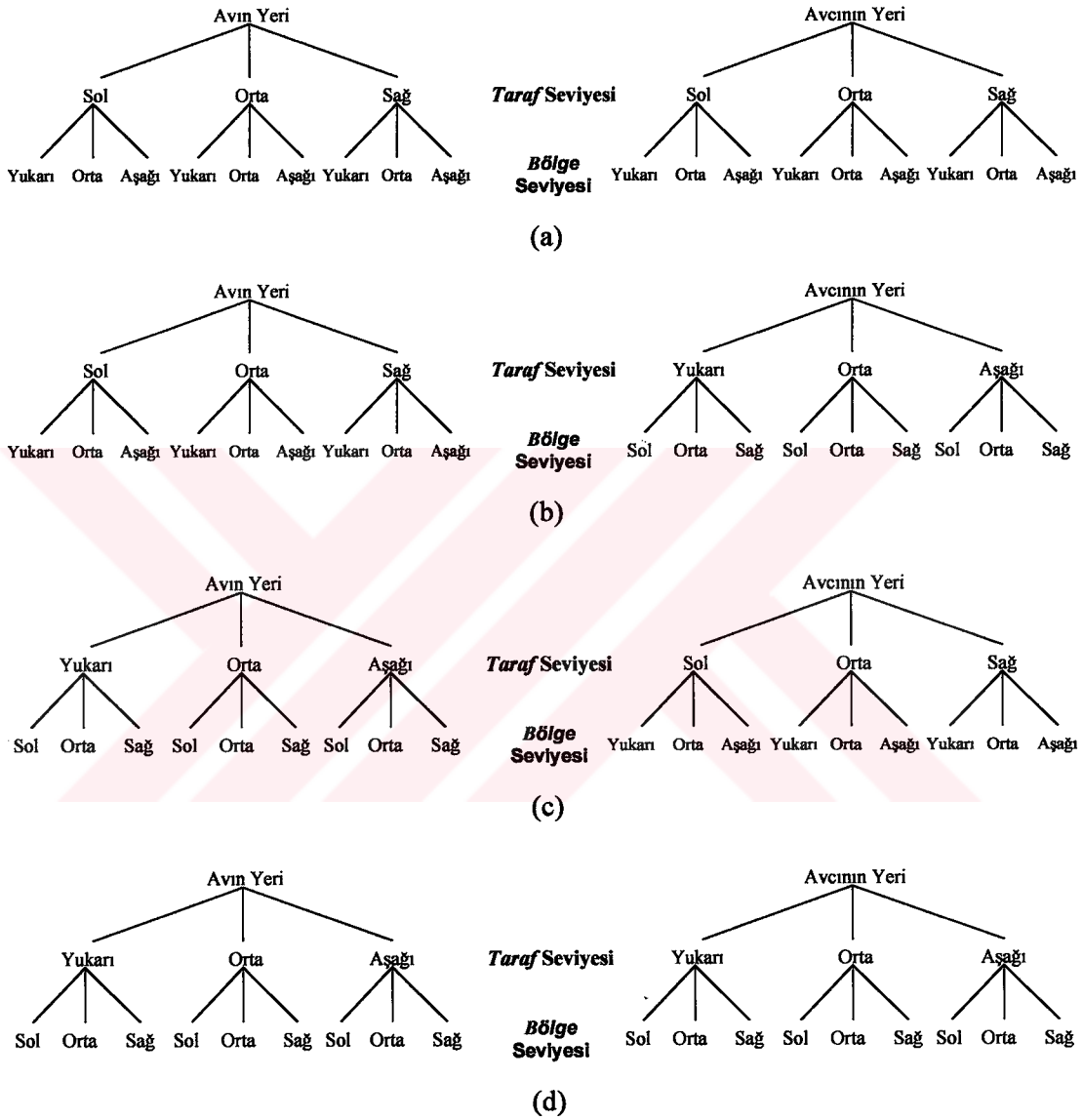
8. Yeni S' durumu sonlanma şartını sağlarsa, mevcut deneme biter. Aksi halde S', S'in yerine geçer ve Adım 1'den başlanarak süreç tekrarlanır.

5.3. Çok Seviyeli Bulanık Bağıntı Kuralları ile Durumları Genelleştirme

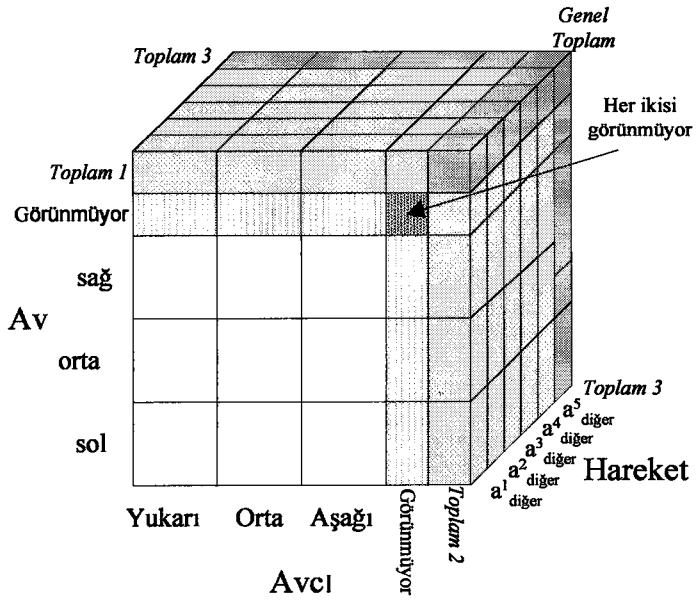
Şekil 5.4'de gösterilen bulanık veri küpü için oluşturulabilecek hiyerarşiler Şekil 5.6'da verilmiştir. Örnek olarak, 1 numaralı hiyerarşinin ilk seviyesinde av veya diğer avcı; ilgili avcının solunda, sağında veya avcı ile aynı bölgede olabilirken ikinci seviyede avın veya diğer avcının yeri yukarı-solda veya sağ-aşağıda gibi tanımlamalarla daha da belirginleştirilir.

Şekil 5.7'deki veri küpü verilen 2 numaralı hiyerarşi için Şekil 5.4'deki bulanık veri küpünün genelleştirilmesi ile elde edilir. Her bir seviyedeki minimum destek değeri, Bölüm 4'de ifade edildiği gibi birbirinden farklı ve daha düşük seviyelerde daha küçük olarak verilir.

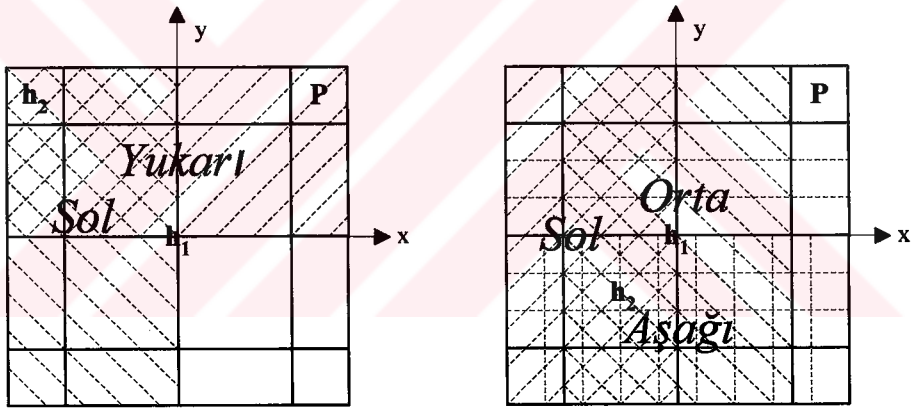
Öğrenme süreci boyunca hangi seviyenin kullanılacağı avcı etmenin durumuna bağlıdır. Buna göre Şekil 5.8, avcının iki farklı durumu için karşılaşılabilecek bazı genelleştirilmiş durumları verir. Şekilden kolaylıkla görülebileceği gibi diğer avcı için genelleştirmeler işgal edilen bölgeye göre yapılır.



Şekil 5.6. Durum boyutu için olası hiyerarşiler (a) 1 numaralı hiyerarşi, (b) 2 numaralı hiyerarşi, (c) 3 numaralı hiyerarşi, (d) 4 numaralı hiyerarşi



Şekil 5.7. 2 numaralı hiyerarşi ile genelleştirmeden sonra elde edilen yeni bulanık veri küpü



Şekil 5.8. Bulanık bir ortamda iki farklı durum için genelleştirilen bölgeler

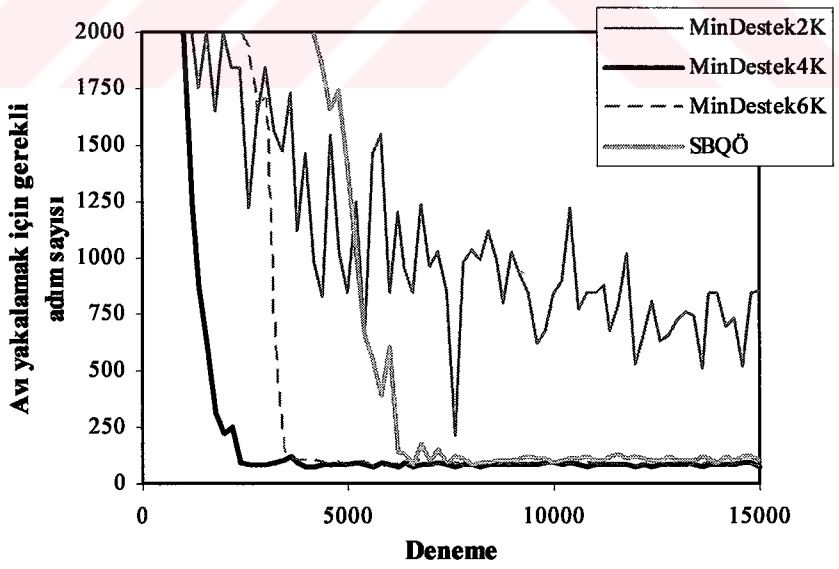
5.4. Uygulama Sonuçları

Bu bölümde bulanık veri küpünü kullanan yeni yaklaşımı değerlendirmek için bazı uygulamalar yapıldı. Ayrıca yaklaşımın standart bulanık Q-öğrenme (SBQÖ) algoritmasına olan üstünlüğü de karşılaştırmalı olarak verildi. Uygulamalarda aksi belirtilmediği müddetçe, durumları modellemek için gerekli bulanık küme sayısı 3 seçildi. Her bir deneme rasgele pozisyonlara yerleştirilmiş iki avcı etmen ve bir av etmeni ile başlar ve avın yakalanması veya 2000 zaman adımına kadar devam eder. Avın yakalanmasıyla bireysel avcılar 100 değerli bir

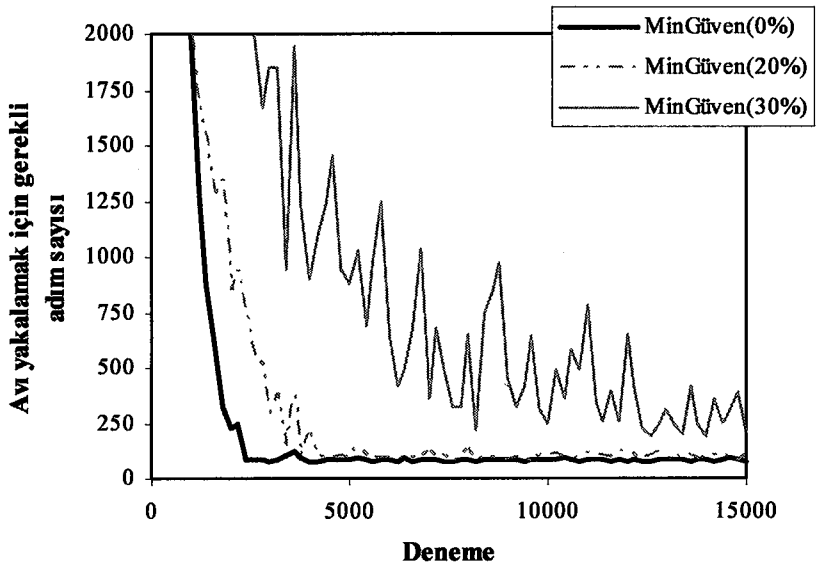
ödül alırlar. Q-öğrenme sürecinde kullanılan parametreler Bölüm 4'de kullanılan parametreler ile aynıdır.

Bulanık veri küpü üzerinde 6 farklı uygulama gerçekleştirildi. İlk uygulamada avın kaçma politikası rasgeledir. Minimum destek değerine ulaşıncaya kadar her bir durumun minimum güven değeri %0 olarak ayarlandı. Şekil 5.9 farklı minimum destek değerlerine göre avı yakalamak için gerekli adımların öğrenme eğrilerini gösterir. Minimum destek değerinin 4K'ya (Şekil 5.9'da MinDestek4K olarak gösterilir) atanması durumundaki öğrenme eğrisi MinDestek6K'dan daha hızlı yakın optimum çözüme yakınsar. Bununla birlikte MinDestek6K için adım sayısı öğrenme anında MinDestek4K'nınkinden biraz daha düşüktür. Bu şekil ayrıca gözlenen durumların sayısını kontrol altına almanın ortalama adım ve özellikle yakınsama hızına göre standart bulanık Q-öğrenmeye üstünlük sergilediğini gösterir. Son olarak, minimum desteğin 2K olarak belirlenmesi durumunda avcı etmenin hemen hemen hiçbir şey öğrenmediği görülür.

Yukarıdaki uygulamada seçilen 3 farklı minimum destek değeri detaylı bir analize dayalı olarak belirlendi. Bu analize göre 4K'dan daha yüksek olarak seçilen minimum destek değerli eğriler 4K'daki eğriye benzer bir eğilim izlemeye başladı. Bu durum MinDestek4K ve MinDestek6K eğrilerinin karşılaştırılmasıyla kolaylıkla görülebilir. Benzer bir analiz bu bölümde verilen diğer şekiller için de geçerlidir.

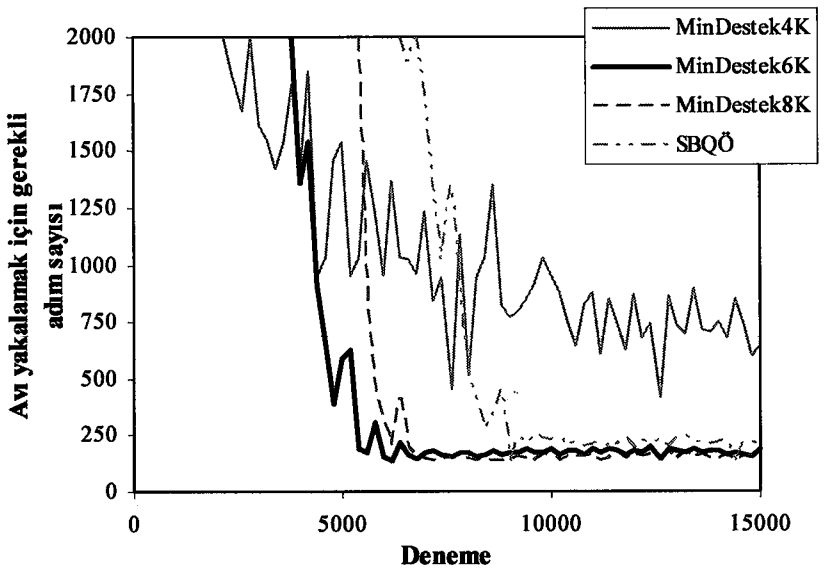


Şekil 5.9. Rasgele kaçan ava göre avcı etmenlerin öğrenme eğrileri

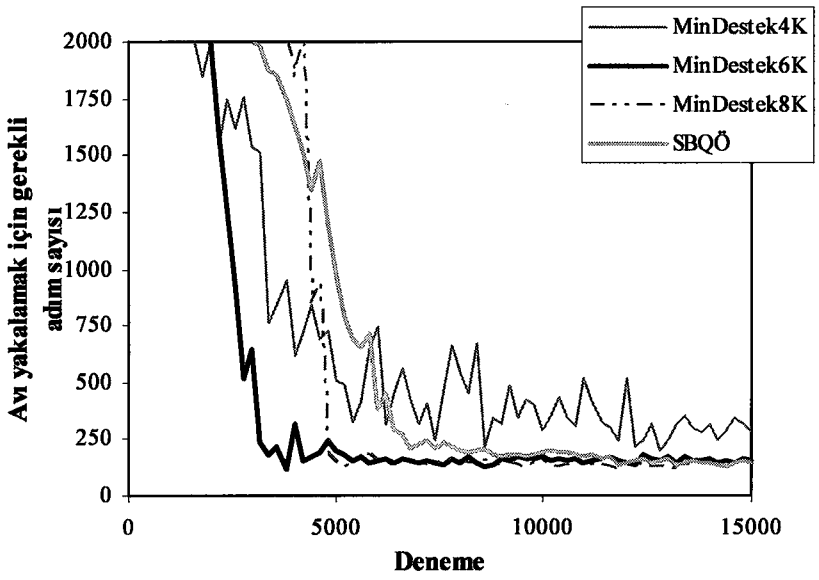


Şekil 5.10. Minimum desteğin 4K'ya sabitlendiği durumda farklı minimum güven değerlerine göre öğrenme eğrileri

İkinci uygulamada minimum destek belirli bir değere sabitlendi ve farklı minimum güven değerleri için öğrenme süreci değerlendirildi. Şekil 5.10, ilk uygulamada en iyi sonuç olarak elde edilen minimum desteğin 4K'ya sabitlenmesi durumunda farklı minimum güven değerlerine göre bulunan öğrenme eğrilerini gösterir. Bu şekilden kolaylıkla görülebileceği gibi ortamın keşfi belirli bir güven değerine kadar önemlidir. Yani etmene ortamını keşfetme fırsatı verilmelidir. Ayrıca güven değerinin %0'a ayarlanması durumundaki çözüm (Şekil 5.10'da MinGüven(%0) olarak gösterilmiştir) daha hızlı yakınsamakla kalmaz avı daha az adım sayısında da yakalar.

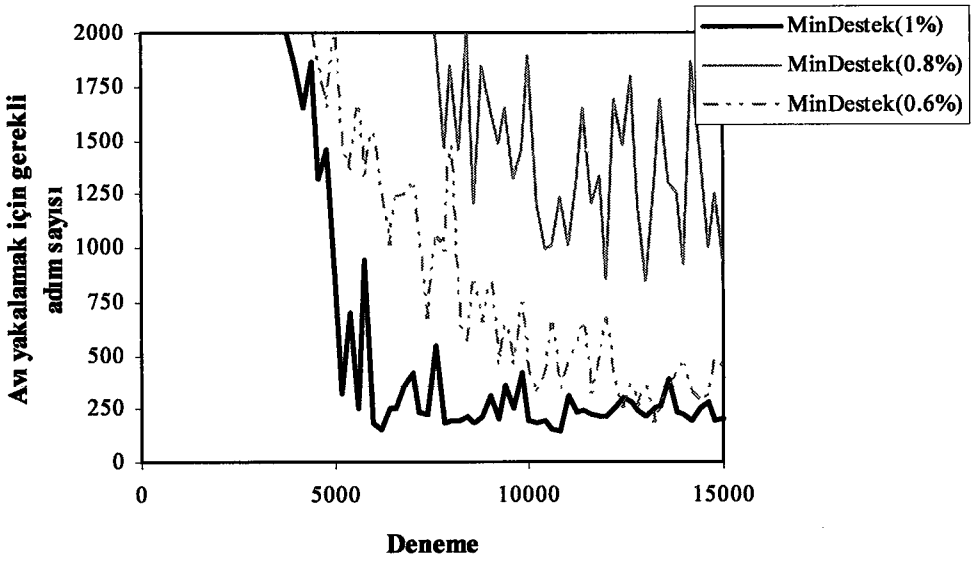


Şekil 5.11. Manhattan-uzaklığı kullanarak kaçan ava göre avcı etmenlerin öğrenme eğrileri

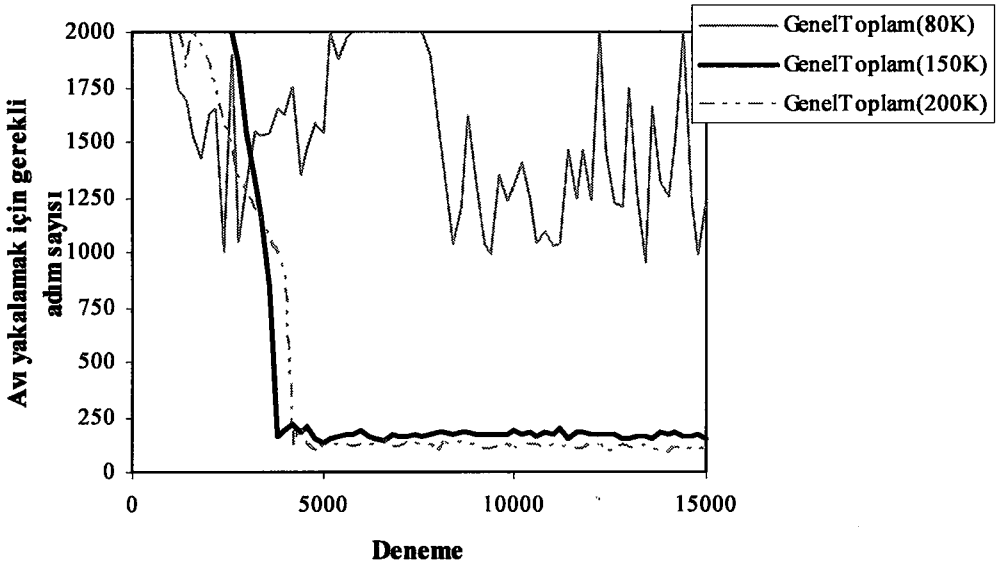


Şekil 5.12. Görme derinliğinin 4'e yükseltilmesi durumda avcı etmenlerin öğrenme eğrileri

Üçüncü uygulamada avın kaçma politikası Manhattan-uzaklık olarak değiştirilmiştir. Bu uygulamanın sonuçları ise Şekil 5.11'de gösterilir. Önceki iki uygulama ile bu sonuçların karşılaştırılması gösterir ki daha karmaşık çoklu etmen ortamlar için her bir durumun daha fazla tecrübelenmesi gerekir. Bu tecrübede minimum desteğin 6K olarak seçilmesi yeterlidir. Çünkü etmen daha düşük destekle öğrenme gerçekleştirememektedir. Buna karşılık minimum desteği 6K'nın üzerine çıkarmak da gereksizdir. Bu bölümde yapılan ilk uygulamaya benzer şekilde bu yeni yaklaşım standart bulanık Q-öğrenmeye üstünlük sağlar.



Şekil 5.13. Minimum desteğin farklı yüzdeleri için avcı etmenlerin öğrenme eğrileri



Şekil 5.14. Genelleştirilmiş durumlu avcı etmenlerin öğrenme eğrileri

Dördüncü uygulama, bağıntı kurallarını çıkarıma dayalı öğrenme süreci üzerinde görme derinliğini artırmanın etkisini araştırır. Bu maksatla avcı etmenlerin görme derinliği 4'e yükseltilmiştir. Avın kaçma politikası ise yine Manhattan-uzaklıktır. Şekil 5.12'deki öğrenme eğrilerinden görüldüğü gibi, avcı daha geniş bir alanı görebildiğinden avı daha hızlı yakalar. Burada bir noktaya dikkat çekilmelidir ki, durumları modellemek için kullanılan üyelik fonksiyon sayısı sabit iken görme derinliğini artırmak etmenlerin karar uzay ayrışımını azaltır. Bununla birlikte önceki uygulama göre bu uygulamada yakınsama hızı yönünde bir iyileştirme vardır.

Beşinci uygulamada minimum destek değeri öncekilerden farklı olarak yüzde değerler ile ifade edildi. Bu yeni yaklaşıma göre, destek eşiğinin üzerindeki yüzde destek değerli durumlar yeterince tecrübelenmiş olarak kabul edilir. Buna göre bir durumun minimum destek değeri aşağıdaki formüle göre bulunur:

$$MinDestek(s_k) = \frac{Toplam3(s_k)}{GenelToplam} \tag{5.3}$$

Yukarıdaki formülde $Toplam3(s_k)$, s_k durumunun gözlem sayısı ve *GenelToplam* ise bulanık küpteki bütün sayıcıların toplamıdır. Bu yaklaşımda, etmenlerin ortamını keşfetmesi için öncelikle *GenelToplam*'a belirli bir değer atanır ve öğrenme süreci başlatılır. *GenelToplam* belirlenen bu değere erişinceye kadar etmenlerin hareketi standart Q-öğrenmedeki gibidir. Fakat bu değere ulaştıktan sonra öğrenme yöntemi, uygun bulanık bağıntı kurallarını çıkarmaya yönelik bu yeni yaklaşıma anahtarlanır. Şekil 5.13 bu tecrübeyle ilgili sonuçları verir. Burada

GenelToplam değeri 100K olarak belirlenmiştir. Şekilden görülebileceği gibi öğrenme eğrileri yeterli kalitede değildir.

Bu bölümün son uygulaması avcı etmenlerin durumlarının genelleştirilmesi, diğer bir değişle çok seviyeli durumlarla ilgilidir. Avın kaçma politikası Manhattan-uzaklık iken, yapılan tecrübeler Şekil 5.14'de gösterilmiştir. Uygulamada 3 farklı *GenelToplam* değeri kullanıldı. Bu şekilde amaç diğer durumlarla karşılaştırıldığında yeterince tecrübelenmemiş olanları genelleştirmektir. Böyle bir durumda etmen, ortamı daha yüksek bir seviyeden gözlemler ve en iyi harekete karar verir. Burada *GenelToplam* değeri, önceden verilen eşik değerine erişirse ve mevcut durumun destek değeri, minimum destek değerinin altında ise avcı etmeni ortamdan daha detaylı bilgi etmek için bir üst seviyeye çıkar. Şekil 5.14'deki sonuçlar sadece diğer avcı etmenin durumlarının genelleştirildiği, av etmeninin durumların genelleştirilmediği eğrileri verir.

5.5. Sonuçlar

Bu bölümde, bulanık çevrim-içi analitik işlemli bağıntı kurallarını çıkararak çoklu etmen takviye öğrenme gerçekleştiren bir yöntem önerildi. Bu maksatla öncelikle, bir etmenin karşılaşabileceği durum sayısını azaltmak için, bulanık küme kavramı, durum uzayına entegre edildi. Daha sonra, etmenler tarafından elde edilen ortamla ilgili bütün bilgiyi tutmak için bir bulanık veri küpü tanımlandı. Bu küp etkili bir şekilde kullanılarak etmenlerin geçmiş hareketlerinden bulanık bağıntı kuralları çıkarıldı. Bu kurallar yardımıyla çoklu etmen öğrenmede sıklıkla karşılaşılan iki problemin üstesinden gelindi. Bunlardan biri bulanık bir ortamda görme alanının dışında olması durumunda bile diğer etmenin hareketini tahmin etmek; diğer ise etmenlerin en uygun hareketini belirlemektir.

Çoklu etmen öğrenme genel olarak çok karmaşık bir problemdir ve elde edilen sonuçlar problemin spesifik özelliklerine bağımlı olabilir. Bununla birlikte, iyi bilinen bir av/avcı ortamı üzerinde yapılan uygulama sonuçları, önerilen bulanık çevrim-içi analitik işlem madenciliğine dayanan öğrenme yönteminin, çoklu etmen sistemlerin adaptif davranışı için iyi bir yaklaşım olduğunu gösterdi.

6. ÇOKLU ETMEN MODÜLER ÖĞRENME SİSTEMİNDE BULANIK BAĞINTI KURALLARINI KULLANMA

Q-öğrenme, etmenlerin ödül veya ceza takviye işaretleri yardımıyla uygun hareket politikalarını öğrenmek için kullandıkları tekniklerden biridir. Bununla birlikte, çoklu etmen sistemler gibi daha karmaşık durumlarla karşılaşıldığı zaman uygun hareket politikalarını öğrenmek için etmenler, çok daha fazla hesapsal kaynaklara gerek duyarlar. Bu şekilde tek parça olarak gerçekleştirilen Q-öğrenme yöntemleri ölçeklenebilirlik özelliklerini kaybeder. Diğer bir değişle, Q-öğrenme, çoklu etmen sistemine uygulandığı zaman bazı güçlüklerle karşılaşır. Çünkü böyle bir ortamda her bir öğrenen etmenin durum uzayı, partnerlerinin sayısına göre üstel olarak artar. Örnek olarak, Şekil 2.2 (b)'deki ortam için bir avcı etmenin durumu, görme alanındaki mevcut diğer etmenlerin pozisyonlarına göre belirlenir.

Tanım 6.1. (Avcının durumu): n etmenli bir ortam verilsin. Bir h avcı etmenin durumu m tane değişken kümesi $(c_1, c_2, ..., c_m)$ ile temsil edilebilir. Burada m=n-1'dir ve h'ın dışındaki etmen sayısını ifade eder. Değişkenler kümesinin her bir elamanı, h'ın dışındaki etmenlerin h'ın bağıl pozisyonunu temsil etmek için kullanılır.

Tanım 6.2. (Durum sayısı): Bir ızgara tipi çoklu etmen ortamında bir h avcı etmenin görme derinliği d olarak verilsin, bu durumda diğer etmenlerin olası bir karede bulunma sayısı $(2d+1)^2+1$ ile verilir. Burada $(2d+1)^2$, h'ın görme alanındaki kare sayısını, 1 ise diğer etmenin görme alanının dışındaki herhangi bir karede olma sayısını verir.

6.1. Modüler Öğrenme

Çoklu etmen sistemler üzerine yapılan çalışmalarda bazı araştırmacılar, bir problem uzayını ayrıştırma yöntemlerinin takviye öğrenme sürecinde öğrenme performansını geliştirdiğini ispatlamışlardır [26, 72]. Bu araştırmacılardan Ono ve Fukomoto [39], çoklu etmen sistemlerde yüksek boyutlu durum uzayı probleminin üstesinden gelebilmek için bir modüler Q-öğrenme mimarisi önerdiler. Mimari birden fazla modülden oluşur ve her bir modül kendine ait bir alt problemle ilgilenir. Böylece Q-tablosunun boyutu küçülür ve çok amaçlı problemler için öğrenme zamanı gelişir.

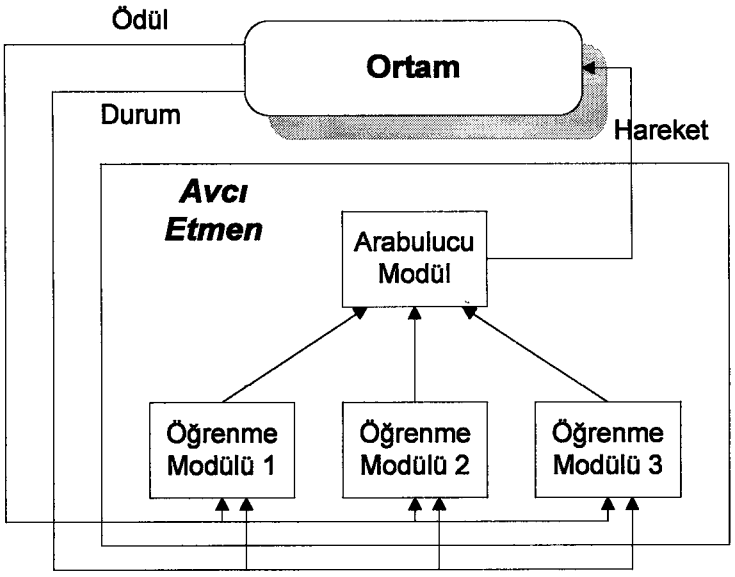
Şekil 6.1, bir av ve dört avcı etmenli bir ortamda (Şekil 2.2 (b)) çoklu etmenlerin davranışını ele almak için Ono ve Fukomoto [40] tarafından önerilen modüler mimariyi gösterir.

Bu mimari 3 öğrenme modülü ve 1 arabulucu modülden oluşur. Her bir öğrenme modülü kendine özgü algısal girişin belirli özellikleri üzerine odaklanır ve Q-öğrenmeyi gerçekleştirir. Mimarinin L_i olarak temsil edilen modülü, i'ninci partnerinin bağıl pozisyonlarını giriş olarak alır. Yani, diğer partnerle ilgili bilgileri ihmal eder. Bu yapılırken her bir avcı etmenin kimliği diğer bütün avcı etmenlerce görünmez. Bir avcı etmen kendisine en yakın kareden başlayarak spesifik bir tarama yapar ve avı ve ilgili modül için göz önünde bulunduracağı partnerini seçer. Diğer bir değişle avcı etmen, gözlenen en yakın diğer avcının durumunu 1 numaralı modüle atar. Bu şekilde her bir modülün durum uzayı bu iki bağıl pozisyonun bir kombinasyonu olarak temsil edilir ve durum uzayı boyutu $(2d+1)^4+1$ olur. Bu durum uzayı boyutu nispeten küçük d değerleri için hesapsal olarak standart Q-öğrenme algoritması tarafından ele alınabilirdir. Öğrenme işlemini gerçekleştirmeyen aracı modül, basit bir sezgisel karar süreci kullanarak öğrenen modüllerin karar politikalarını birleştirir ve ilgili etmen için son hareketi belirler.

Avcı etmenler Whitehead [38] tarafından önerilen "en büyük toplam" stratejisini kullanarak hareketlerine karar verir:

$$\arg\max_{a\in A}\sum_{i=1}^{I=3} Q_i(s_i,a)$$

Yukarıdaki formülde Q_i, L_i öğrenme modülü tarafından beslenen durum-hareket değerini gösterir. Her bir modül diğer modüllerin Q-tablosuna bakmaksızın kendi Q-tablosunu günceller.



Şekil 6.1. Modüler takviye öğrenme mimarisi

Örnek olarak, 4 avcı ve 1 av etmenli çoklu etmen ortamı için her bir avcı etmenin görme alanı d=2 seçilirse, etmenlerin standart tek parçalı Q-öğrenme algoritmasını kullanarak karşılaşabileceği olası durumlar sayısı 459976'dır. Bu demektir ki tek parça Q-tablolu bir öğrenme etmeninin geniş bir hafıza gereksinimi vardır. Buna karşılık modüler yapıda olduğu gibi iki etmen ayrı bir Q-tablosu organize edecek olursa, olası durumlar sayısı 626'ya düşer.

6.2. Bulanık İç Model Yapılı Modüler Yaklaşım

Modüler yaklaşım üzerine daha önce yapılan çalışmaların tamamı ortamı sabit olarak ele alır. Yani, diğer etmenlerin davranışı ortamda göz önüne alınmamıştır. Halbuki daha önceleri ifade edildiği gibi bir etmenin sürekli olarak öğrendiği ve her bir diğer etmenin davranışlarının zamanla değiştiği ortamları dinamik olarak göz önüne almak daha rasyoneldir. Bu maksatla, bu çalışmada her bir bulanık öğrenen modüle, bulanık bir iç model tablosu da eklenmiştir. Bu iç model Nagayuki ve diğ. [50]'nin çalışmasına benzerdir. Fakat tez çalışması boyunca avcı etmenlerin kısmen gözlenebilir bir yapıya sahip oldukları kabul edildiğinden dolayı, bir avcı etmen, görme alanı dışındaki diğer etmenin hareketlerini gözleyemez. Bunun için sezgisel bir yöntemle diğer etmenin hareketi tahmin edilir.

Diğer etmenin hareketi $a_{diğer}$'i tahmin etmede $F(s, a_{diğer})$ bulanık iç model fonksiyonundan faydalanılır. Eğer bir avcı, görme alanında diğer avcıyı algılarsa, o zaman $F(s, a_{diğer})$ diğer etmenin geçmiş hareketlerinden kolaylıkla güncellenir. Diğer bir yandan, eğer bir avcı diğer avcıyı algılayamazsa s durumundaki görünmeyen avcının bütün $F(s, a_{diğer})$ değerlerine 0.2 atanır. Böylece göz önüne alınan avcının hareket etmesinde görünmeyen avcıların etkisi ihmal edilir.

Bir avcı etmenin bulanık iç model fonksiyonu aşağıdaki algoritma ile güncellenir.

Algoritma 6.1. (Bulanık iç model fonksiyonunu güncelleme):
h_1 ve h_2 iki farklı avcı etmeni olsun.

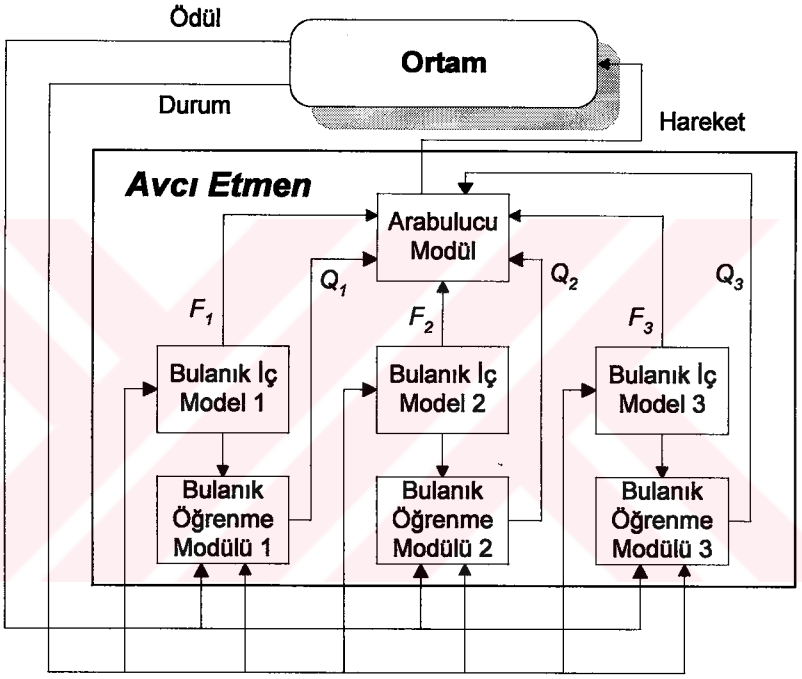
IF h_1, i'ninci modülde $S^i(s_1^i, s_2^i, .. s_n^i)$ (Burada n, i'ninci modülde gözlenen durum sayısı) durumunda h_2'nin $a^*_{diğer}$ hareketini yürüttüğünü gözlemlerse, *THEN*

S^i durumunda yürütülebilir bütün $a_{diğer} (\in A_{diğer})$ hareketleri için bulanık iç model fonksiyonunu aşağıdaki gibi güncellenir.

$$F(s_k^i, a_{diğer}) \leftarrow (1 - \mu_{durum}(av).\mu_{durum}(diğer\ avcı)).F(s_k^i, a_{diğer}) + \mu_{durum}(av).\mu_{durum}(diğer\ avcı).\begin{cases} k & a_{diğer} = a^*_{diğer} \\ 0 & aksi\ halde \end{cases} \quad (6.1)$$

ELSE Bütün hareketler için $F(s_k^i, a_{diğer})$ =0,2.

Yukarıdaki güncelleme formülünde, $\mu_{durum}(diğer\ avcı) = \mu_x(diğer\ avcı).\mu_y(diğer\ avcı)$, $\mu_{durum}(av) = \mu_x(av).\mu_y(av)$ ve $k(0 \le k < 1)$, $a^*_{diğer}$'in etkisini kontrol etmek için kullanılan bir parametredir.



Şekil 6.2. Önerilen iç model yetenekli bulanık-takviye öğrenme mimarisi

Önerilen yaklaşımda durum uzayı bulanık kümeler kullanılarak modellendiğinden ve diğer avcının yeri birden fazla bulanık kümeler ihtiva edebileceğinden, güncelleme sayısı da birden fazla olabilir. Buna karşılık $a_{diğer}$ ve a_{kendi} hareketi en yüksek üyelik dereceli duruma göre tahmin edilir. Eğer bütün durumların üyelik dereceleri birbirine eşitse, tahmin, rasgele seçilen duruma göre yapılır. Bunun yanında geri kalan durumların bulanık iç model değerleri, seçilen $a_{diğer}$ hareketi dikkate alınarak güncellenir. Önerilen mimarinin blok diyagramı son haliyle Şekil 6.2'de gösterilmiştir.

6.3. Önerilen Çoklu Etmen Öğrenme Süreci

Bu bölümde önerilen çoklu etmen mimarisi için geliştirilen öğrenme algoritması verilecektir.

Algoritma 6.2. (İç model yetenekli modüler bulanık-takviye öğrenme algoritması):

1. Her bir bulanık öğrenme modülü L_i ve bulanık iç model bloğu, mevcut S^i durumunu gözlemler. Her bloğun durumu $S^i(s_1^i, s_2^i, ..s_n^i)$ ile temsil edilir.
2. Her bir avcı $F(s_{\max}^i, a_{diğer})$'e göre diğer avcının $a_{diğer}$ hareketini tahmin eder ve Q-değerlerini arabulucu modüle gönderir. Burada $s_{\max}^i$, i'ninci modülde en yüksek üyelik dereceli durumu gösterir.

$$s_{\max}^i = \max_{s_k^i \in S^i} \mu(s_k^i)$$

3. Arabulucu modül aşağıdaki formüle göre en uygun a_{kendi} hareketini seçer:

$$\arg \max_{a_{kendi} \in A_{kendi}} \sum_{i=1}^{l=3} Q_i^{F(s_{\max}^i)}(s_{\max}^i, a_{kendi})$$

Burada $Q_i^{F(s_{\max}^i)}(s_{\max}^i, a_{kendi})$, L_i öğrenme modülünün bulanık iç modeli göz önüne alınarak hesaplanan Q-fonksiyon değerini gösterir. Biçimsel olarak;

$$Q_i^{F(s_{\max}^i)}(s_{\max}^i, a_{kendi}) = \sum_{a_{diğer} \in A_{diğer}} F(s_{\max}^i, a_{diğer}).Q(s_{\max}^i, a_{kendi}, a_{diğer}) \qquad (6.2)$$

4. Avcı etmeni Adım 3'de seçilen a_{kendi} hareketini yürütür.
5. Aynı anda görme alanındaki diğer avcı $a_{diğer}^*$ hareketini yürütür.
6. Ortam yeni bir S' durumuna geçer.
7. Avcı etmeni ortamdan bir r ödülünü alır ve $F(s_k^i, a_{diğer}^*)$ ve $Q(s_k^i, a_{kendi}, a_{diğer}^*)$ değerleri güncellenir.

   7.1. $F(s, a_{diğer}^*)$, Algoritma 1'deki gibi güncellenir.

   7.2.. Q-fonksiyonu ise aşağıdaki formüle göre güncellenir.

$$Q(s_k^i, a_{kendi}, a^*_{diğer}) \leftarrow [1-\beta.\mu_{durum}(av).\mu_{durum}(diğer\ avcı)].Q(s_k^i, a_{kendi}, a^*_{diğer}) + \beta.\mu_{durum}(av).\mu_{durum}(diğer\ avcı)[r+\gamma.\max_{a^{/}_{self}\in A_{self}} Q(s_k^{i/}, a^{/}_{kendi}, a^{/}_{diğer})] \quad (6.3)$$

Burada $a^{/}_{diğer} = \arg\max_{a^{/}_{diğer}\in A_{diğer}} F(s^{i/}_{\max}, a^{/}_{diğer})$ 'dir.

8. Eğer yeni $S^{/}$ durumu sonlanma şartını sağlarsa, o zaman mevcut deneme biter. Aksi halde $S^{/}$, S in yerine geçer ve Adım 1'e gidilerek süreç tekrarlanır.

6.4. Uygulama Sonuçları

Önerilen çoklu etmen mimari ve öğrenme yaklaşımını değerlendirmek için bu bölümde bazı uygulamalar yapıldı. Önceki uygulamalarda olduğu gibi her bir denemede adım sayısı 2000 ile sınırlandırıldı. Avın yakalanması durumunda bireysel avcılar 1,0 değerli ödül alırlar. Böylece bir avcı etmenin bulanık öğrenme modüllerinin tamamı da sahip oldukları karar politikasına bakmaksızın aynı ödülü alır. Buna karşılık yakalanmayan her bir hareket için -0,1 değeri ile cezalandırılır. Bu bölümdeki uygulamalar için aşağıdaki parametreler kullanıldı.

1. Öğrenme oranı $\beta = 0{,}8 \rightarrow 0{,}2$
2. Azalma faktörü $\gamma = 0{,}9$
3. Q-fonksiyonun başlangıç değeri: 0,1
4. Bulanık iç model fonksiyonunun başlangıç değerleri: 0,2

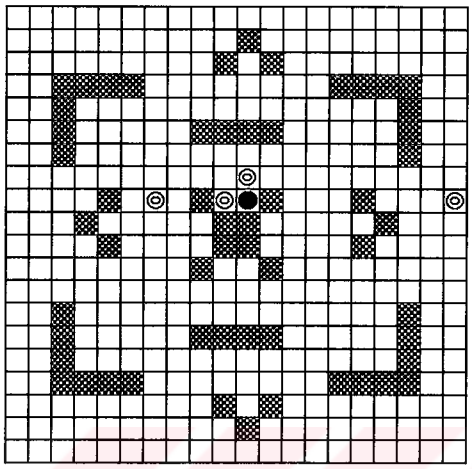
Bu tezdeki bütün uygulamalarda olduğu gibi, sonuçlar 10 farklı çalışmanın ortalamasını alır. Bu bölüm uygulamalarının diğer bir amacı ise farklı bulanık küme sayılarına göre sonuçları değerlendirmektir. Bu maksatla tez çalışması boyunca faydalanılan iki ortam dışında daha karmaşık bir ortam da kullanıldı. Uygulamalar için kullanılan Şekil 2.2. (b) ve Şekil 6.3'deki her iki ortamın boyutu 20x20 olarak belirlendi. Yeni ortam öncekilerden farklı olarak sınırları içerisinde bazı blokları barındırır. Bu bloklar da ortamı heterojen bir yapıya büründürür. Bu yeni ortamın diğer bir özelliği ise av, şekilde görüldüğü gibi sadece avcılar ile değil yeri geldiğinde engeller tarafından da kuşatılabilir. Diğer bir değişle, hangi nedenden olursa olsun dört yanı tutulan av yakalanmış olur.

Her bir ortam için aşağıda 6 farklı durum tanımlandı ve bunlarla ilgili uygulamalar gerçekleştirildi. İlk 4 durumda av ve avcının görme derinliği 3 olarak ayarlandı. Ayrıca ilk iki durum için avcı her bir zaman adımında gözlemlediği bütün Q-fonksiyonlarını günceller. Diğer bir yandan son iki durum için av ve avcının görme derinliği 4'e yükseltildi.

Durum 1: Av rasgele hareket eder.

Durum 2: Av, görünen avcılara Manhattan-uzaklık toplamını maksimum yapacak şekilde hareketini seçer.

Durum 3: Av, Durum 2'deki gibi davranır. Her bir zaman adımında bir avcı sadece en yüksek üyelik dereceli durumun Q-fonksiyonunu günceller.



Şekil 6.3. Bloklu bir yapıya sahip heterojen av/avcı ortamı

Durum 4: Av, Durum 2'deki gibi davranır. Her bir zaman adımında bir avcı yüksek üyelik dereceli bazı durumları aynı anda günceller. Bu durumların sayısı $\left\lceil \frac{|S_t|}{2} \right\rceil$ ile verilir. Burada $|S_t|$, t anında gözlenen durum sayısıdır.

Durum 5: Durum 4'den tek farkı avcı etmenlerin görme derinliğinin 4'e yükseltilmesidir.

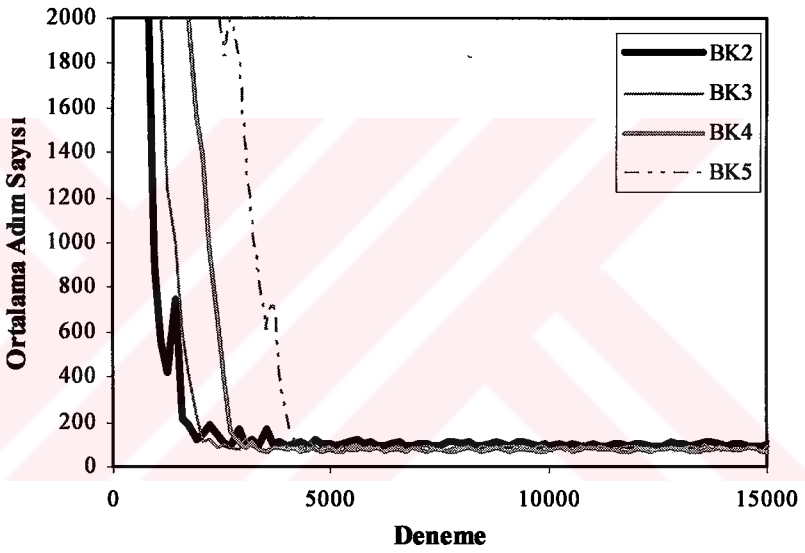
Durum 6: Avcı etmen, Durum 4'de belirlenen durumların seçilmeyen bir hareketini de günceller [73]. Güncelleme aşağıdaki formüle göre yapılır:

$$Q(s_k^i, a_{kendi}, a^*_{diğer}) \leftarrow m.\left[\begin{array}{l}(1-\mu_{durum}(av).\mu_{durum}(diğer\ avcı)).Q(s_k^i, a_{kendi}, a_{diğer}) \\ +\beta.\mu_{durum}(av).\mu_{durum}(diğer\ avcı).(r+\gamma.\max_{a_{kendi}\in A_{kendi}} Q(s_k^{i\prime}, a'_{kendi}, a'_{diğer}\end{array}\right]$$

$$m = \begin{cases} 1 & Eğer\ a_{kendi}\ seçilen\ hareketse \\ (0,1] & a_{kendi} = a'_{kendi} \end{cases}$$

Bu durumlarla ilgili olarak 12 tane şekil elde edilmiştir. Her bir şekilde x-eksenleri deneme sayılarını, y-eksenleri ise 10 farklı çalışma için ortalama adım sayılarını gösterir. Her şeklin yanında ayrıca belirli deneme noktalarındaki adım sayıları da verilmiştir. Böylece şekil açık olmadığı zaman eğriler ararsındaki fark daha rahat gözlenebilir.

Şekil 6.4-6.9 homojen, yani engelsiz ortam için yapılan uygulamalar kümesindeki öğrenme eğrilerini gösterir. Bu şekillerden kolaylıkla görülebileceği gibi bulanık küme sayısı artarken, daha geniş durum uzayından dolayı yakın optimum çözüme yakınsama gecikir. Bununla birlikte, daha fazla üyelik fonksiyonları karar uzayının daha iyi çözümlemesini sağladığından dolayı daha kaliteli çözüm elde edilir. Ayrıca yakınsama avın davranışıyla da etkilenir. Örneğin, av rasgele hareket ettiği zaman yakın optimum çözüme yakınsama daha hızlıdır. Bu önermenin doğruluğu Şekil 6.4'de gösterilen Durum 1'deki uygulama sonuçları ile Şekil 6.5-6.9 arasında gösterilen diğer beş durum ile ilgili sonuçlar karşılaştırılarak ispatlanabilir. Çünkü, son beş durumda av kaçmaya çalışırken ilk durumda av nereye gideceğini bilmeden rasgele hareket eder.



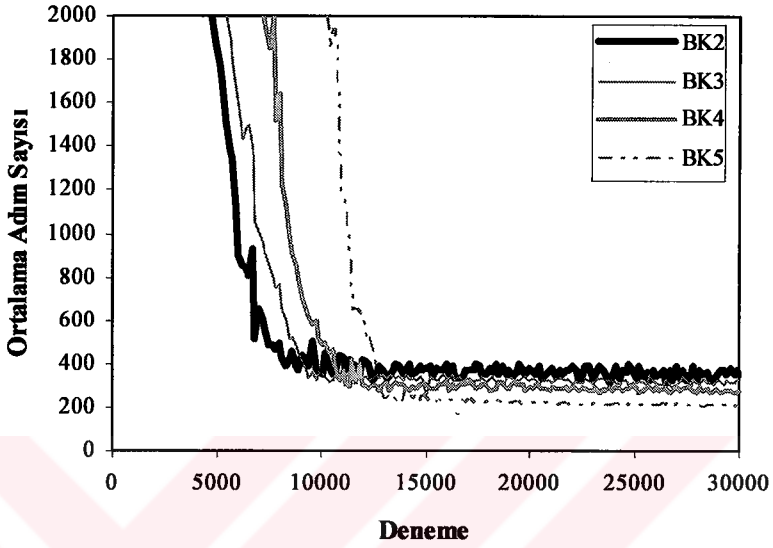
Şekil 6.4. Homojen ortam için Durum 1'in sonuçları

Tablo 6.1. Homojen ortamda Durum 1 için bazı deneme noktalarındaki ortalama adım sayıları

Bulanık Küme Sayısı	Ortalama Adım Sayısı		
	5000 denemeden sonra	10000 denemeden sonra	15000 denemeden sonra
FS2	102	94	92
FS3	92	78	75
FS4	78	72	66
FS5	65	63	61

Durum 2 ve Durum 3 arasındaki tek fark güncellenen Q-fonksiyonu sayısıdır. Bununla ilgili uygulama sonuçları sırasıyla Şekil 6.5 ve Şekil 6.6'da gösterilmiştir. Şekil 6.5'de daha

hızlı bir yakınsama elde edilmiştir. Bunun tek nedeni, bütün durumları göz önüne almaktır. Bu durum paralel güncellemenin önemini gösterir. Ayrıca, eğer aynı anda gözlemlenen daha kötü durumlar, belirli bir kritere göre elenirse daha kaliteli bir çözüm bulunabilir. Şekil 6.7'de gösterilen Durum 4 için yapılan uygulama böyle bir çözümün sonuçlarını verir.



Şekil 6.5. Homojen ortam için Durum 2'in sonuçları

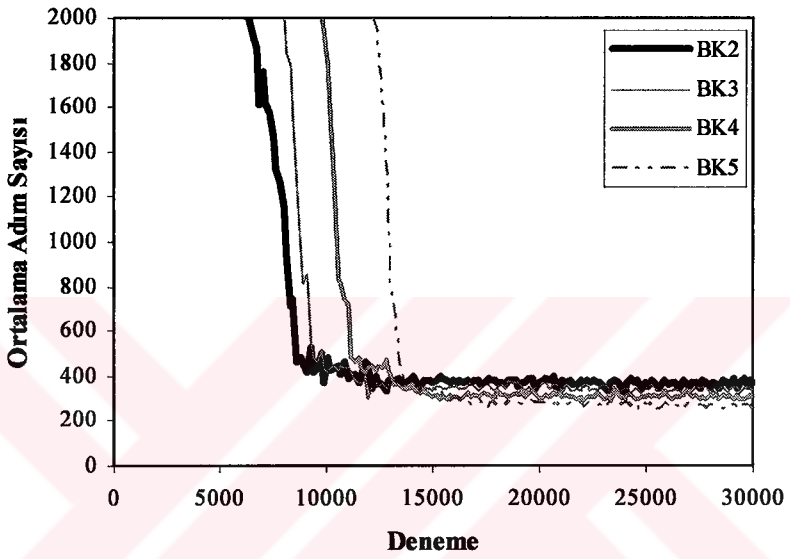
Tablo 6.2. Homojen ortamda Durum 2 için bazı deneme noktalarındaki ortalama adım sayıları

Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra	30000 denemeden sonra
FS2	375	382	369	351
FS3	256	319	315	318
FS4	306	294	273	277
FS5	235	222	211	209

Bir avcı etmenin görme derinliği 3'den 4'e yükseltilirse karar uzay çözünürlüğü azalır. Fakat, avcı daha geniş bir alanı algıladığından avı daha hızlı yakalar. Bu uygulamanın sonuçları ise Şekil 6.8'de gösterilmiştir. Homojen ortamla ilgili son uygulama, Durum 4 tarafından belirlenen durumların seçilmeyen bir hareketini de güncelleyen yaklaşımla ilgilidir. Şekil 6.9'da verilen sonuçlara göre böyle hibrid bir yaklaşım diğer dört durumdan (Durum 2-Durum 5) daha iyi bir çözüm bulur.

Uygulamaların diğer kümesinde bu 6 farklı durum heterojen ortam üzerinde gerçekleştirildi. Daha önce izah edildiği gibi bu yeni ortamın özelliği diğerine göre engeller içererek daha karmaşık olması ve avı yakalamada sadece avcıların değil de yeri geldiğinde

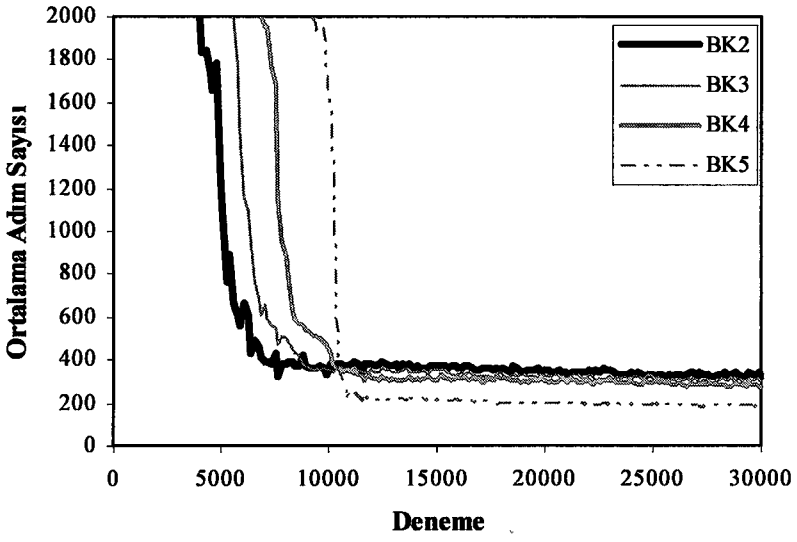
ortamdaki blokların da kullanılmasıdır. Bu bölümdeki uygulama sonuçları Şekil 6.10-6.15'de sırasıyla gösterilmişlerdir. Gözlemler homojen ortam için elde edilenlere benzerdir. Bununla birlikte burada yeni bir sonuca dikkat çekilmelidir ki, daha fazla üyelik fonksiyonları Durum 1 haricinde diğer bütün durumlar için daha hızlı bir yakınsama ve daha az adım sayısı sağlar. Diğer bir yandan, görme derinliğinin yükseltilmesi ve hibrid paralel bir güncellemenin gerçekleşmesi durumunda, çözüm kalitesi, yakınsama hızı ve yakınsama adım sayısı baz alındığında gelişir.



Şekil 6.6. Homojen ortam için Durum 3'ün sonuçları

Tablo 6.3. Homojen ortamda Durum 3 için bazı deneme noktalarındaki ortalama adım sayıları

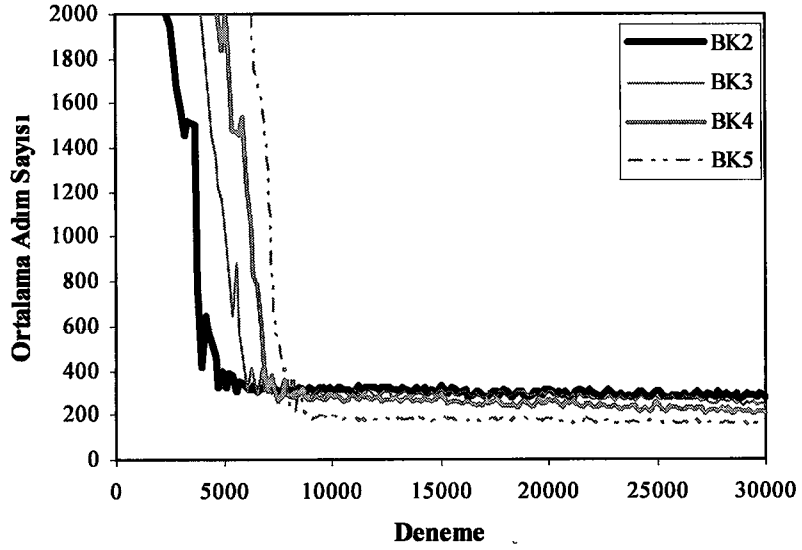
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra	30000 denemeden sonra
FS2	370	369	368	374
FS3	337	357	328	336
FS4	309	294	304	308
FS5	339	281	267	252



Şekil 6.7. Homojen ortam için Durum 4'ün sonuçları

Tablo 6.4. Homojen ortamda Durum 4 için bazı deneme noktalarındaki ortalama adım sayıları

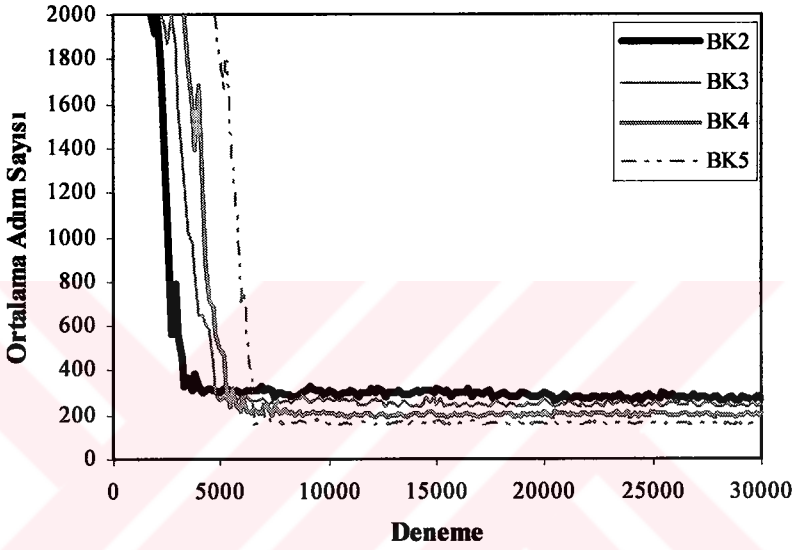
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra	30000 denemeden sonra
FS2	374	355	334	320
FS3	349	323	325	291
FS4	308	307	294	274
FS5	204	199	190	178



Şekil 6.8. Homojen ortam için Durum 5'in sonuçları

Tablo 6.5. Homojen ortamda Durum 5 için bazı deneme noktalarındaki ortalama adım sayıları

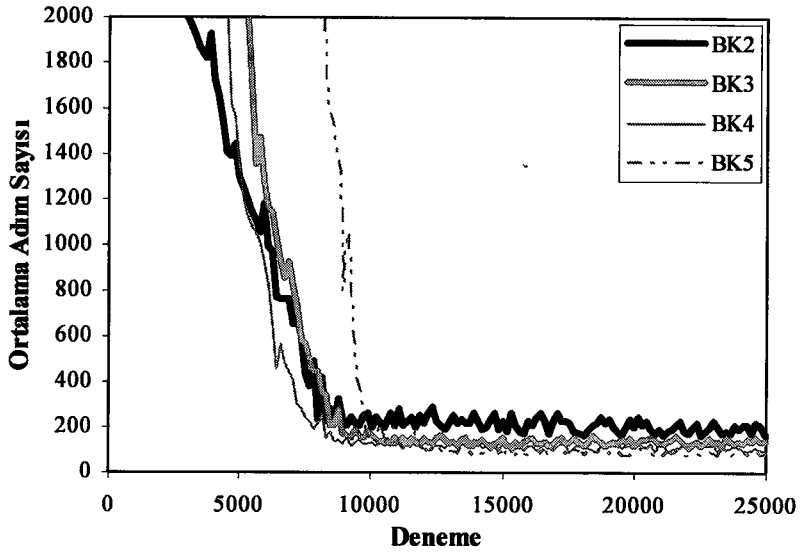
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra	30000 denemeden sonra
FS2	325	312	306	272
FS3	315	303	287	240
FS4	286	271	257	205
FS5	186	180	171	155



Şekil 6.9. Homojen ortam için Durum 6'nın sonuçları

Tablo 6.6. Homojen ortamda Durum 6 için bazı deneme noktalarındaki ortalama adım sayıları

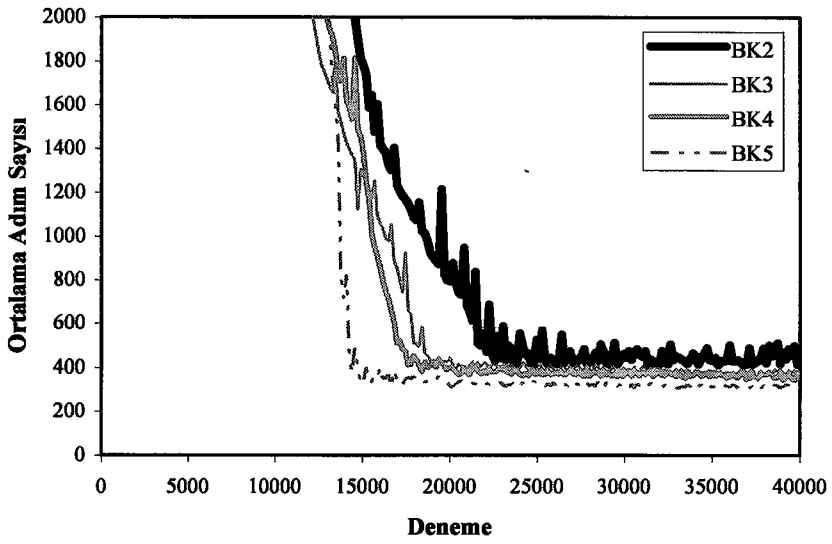
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra	30000 denemeden sonra
FS2	318	301	284	270
FS3	302	269	244	244
FS4	271	218	215	203
FS5	166	163	159	152



Şekil 6.10. Heterojen ortam için Durum 1'in sonuçları

Tablo 6.7. Heterojen ortamda Durum 1 için bazı deneme noktalarındaki ortalama adım sayıları

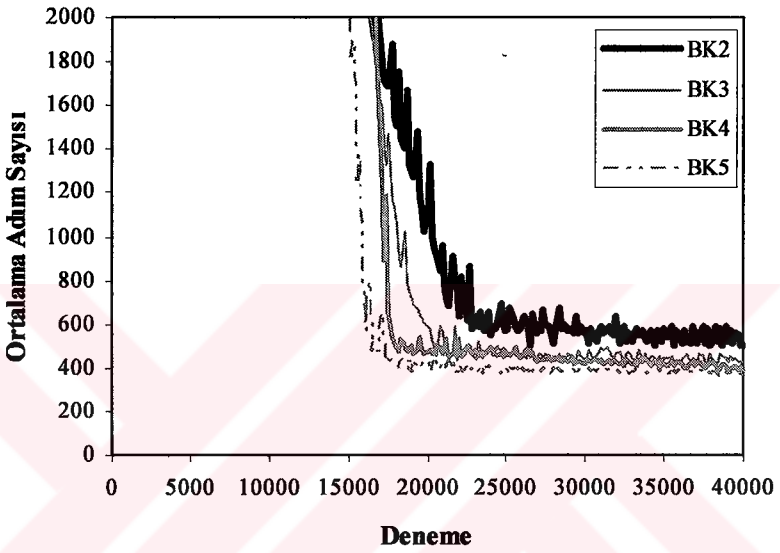
Bulanık Küme Sayısı	Ortalama Adım Sayısı		
	15000 denemeden sonra	20000 denemeden sonra	25000 denemeden sonra
FS2	231	204	166
FS3	155	121	133
FS4	123	138	90
FS5	94	84	84



Şekil 6.11. Heterojen ortam için Durum 2'nin sonuçları

Tablo 6.8. Heterojen ortamda Durum 2 için bazı deneme noktalarındaki ortalama adım sayıları

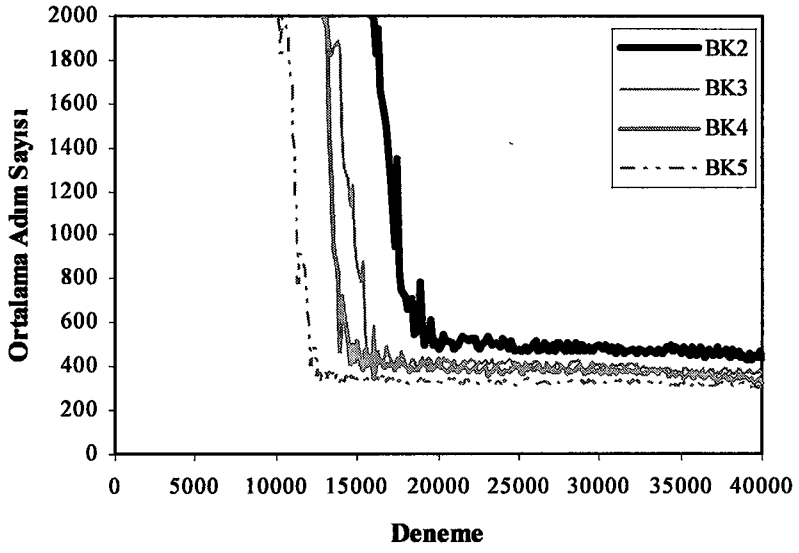
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	25000 denemeden sonra	30000 denemeden sonra	35000 denemeden sonra	40000 denemeden sonra
FS2	525	446	431	461
FS3	401	375	381	382
FS4	371	384	378	359
FS5	330	316	312	310



Şekil 6.12. Heterojen ortam için Durum 3'ün sonuçları

Tablo 6.9. Heterojen ortamda Durum 3 için bazı deneme noktalarındaki ortalama adım sayıları

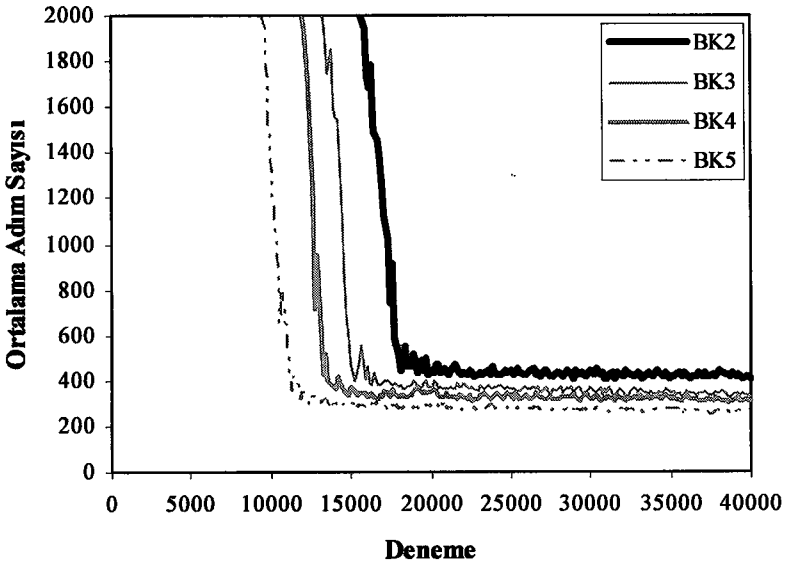
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	25000 denemeden sonra	30000 denemeden sonra	35000 denemeden sonra	40000 denemeden sonra
FS2	576	571	581	504
FS3	486	447	445	425
FS4	459	425	421	385
FS5	397	395	385	355



Şekil 6.13. Heterojen ortam için Durum 4'ün sonuçları

Tablo 6.10. Heterojen ortamda Durum 4 için bazı deneme noktalarındaki ortalama adım sayıları

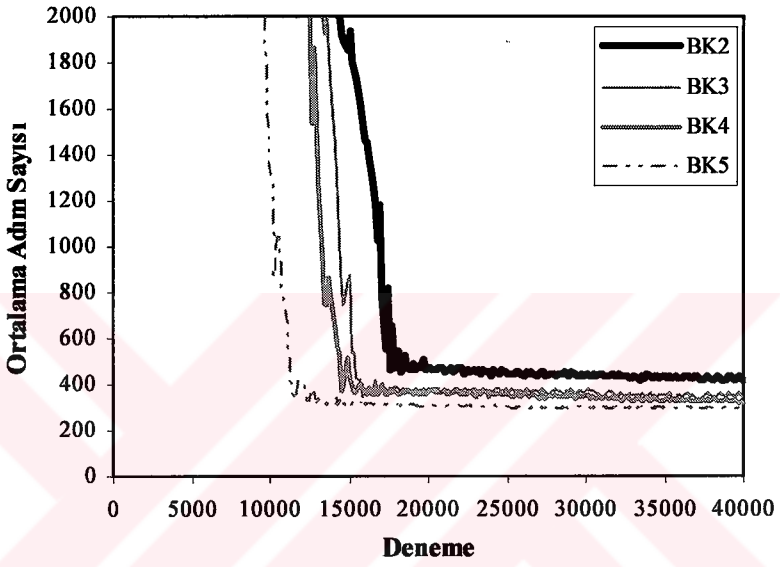
Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	25000 denemeden sonra	30000 denemeden sonra	35000 denemeden sonra	40000 denemeden sonra
FS2	483	486	463	438
FS3	416	396	365	361
FS4	389	367	367	328
FS5	324	328	321	315



Şekil 6.14. Heterojen ortam için Durum 5'in sonuçları

Tablo 6.11. Heterojen ortamda Durum 5 için bazı deneme noktalarındaki ortalama adım sayıları

Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	25000 denemeden sonra	30000 denemeden sonra	35000 denemeden sonra	40000 denemeden sonra
FS2	445	431	420	408
FS3	365	362	346	330
FS4	331	328	318	305
FS5	284	279	271	264



Şekil 6.15. Heterojen ortam için Durum 6'nın sonuçları

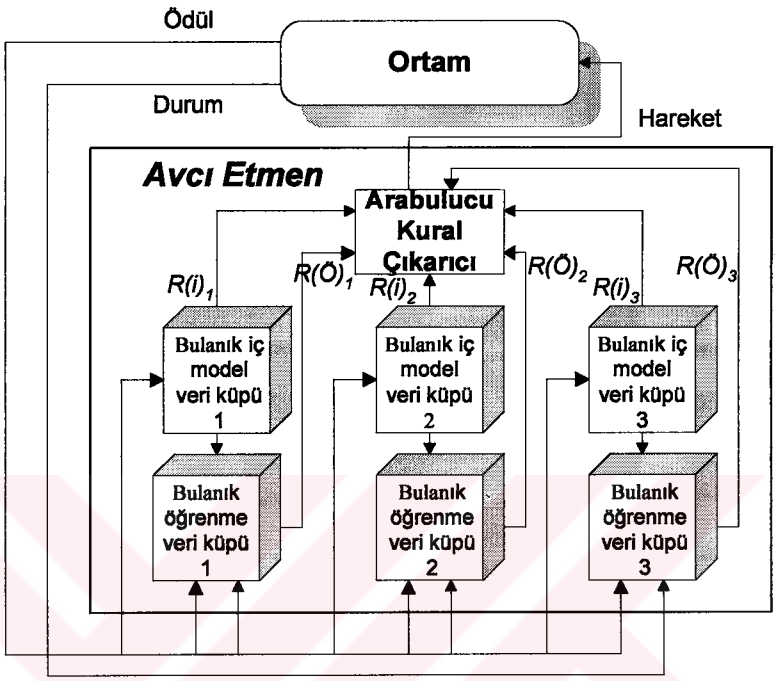
Tablo 6.12. Heterojen ortamda Durum 6 için bazı deneme noktalarındaki ortalama adım sayıları

Bulanık Küme Sayısı	Ortalama Adım Sayısı			
	25000 denemeden sonra	30000 denemeden sonra	35000 denemeden sonra	40000 denemeden sonra
FS2	457	449	436	420
FS3	385	382	352	350
FS4	362	354	336	322
FS5	301	298	296	292

6.5. Bulanık Bağıntı Kurallarını Kullanarak Çoklu Etmen Modüler Öğrenme

Önceki bölümde anlatılan modüler öğrenmeye dayanarak bu bölümde, öncelikle bulanık bağıntı kuralarını çıkaran ve sonra bu kurallar yardımıyla uygun hareketleri seçen yeni bir

modüler öğrenme algoritması tanıtılmıştır. Şekil 6.16 bu maksat için kullanılan modüler mimariyi gösterir. Her bir modül, öğrenme ve iç model bilgilerinin tutulduğu bulanık veri küplerine sahiptir. Veri küplerinden çıkarılan kurallar, arabulucu kural çıkarıcıya iletilir. Bu birim de, etmenin en uygun hareketi yürütebilmesi için gerekli kuralı bulur.



Şekil 6.16. Bulanık veri küplerine dayalı modüler mimari

Algoritma 6.3. (Bulanık bağıntı kurallarına dayalı çoklu etmen modüler öğrenme):

1. Bir avcı etmeni i'ninci modülde $S^i(s_1^i, s_2^i, ...s_n^i)$ durumunu gözlemler (Burada n i'ninci modülde gözlenen durum sayısıdır) ve bulanık veri küpünden çıkarılan bağıntı kurallarına göre $a_{diğer}$ tahmin edilir. $s_k^i (\in S^i)$ durumunun gözlenme sayısı minimum destek değerinden daha büyükse ilgili modülün s_k^i hareketi için en yüksek güven değerli $a_{diğer}$ hareketi seçilir. Aksi durumda, önceki algoritmalarda olduğu gibi hareket seçimi $p(a_i \mid s_k^i)$'ya bağlıdır.
2. Tahmin edilen diğer etmenin hareketine göre, göz önüne alınan etmenin a_{kendi} hareketi seçilir. Eğer $s_k^i - a_{diğer}$ çifti için veri küpünde yeterli tecrübe görünüyorsa, en yüksek güven değerli a_{kendi} kolaylıkla belirlenir. Aksi halde yine olasılık yöntemi uygulanır.

3. Adım 2, i'ninci modülde gözlenen S^i durumunun bütün s_k bileşenleri için tekrarlanır. Her bir bileşenin hareketi seçildikten sonra, S^i durumunun hareketi aşağıdaki formüle göre belirlenir.

$$\arg \max_{a_{kendi} \in A_{kendi}} Q(s_k^i, a_{kendi}, a_{diğer})$$

4. Adım 3'deki gibi bütün modüllerin hareketi belirlendikten sonra arabulucu kural çıkarım birimi avcı etmenin genel a_{kendi} hareketini aşağıdaki kritere göre seçer.

$$\arg \max_{a_{kendi} \in A_{kendi}} \sum_{i=1}^{l=3} Q(s_k^i, a_{kendi}, a_{diğer})$$

5. Aynı anda diğer etmen $a_{diğer}^*$ hareketini yürütür.
6. Ortam yeni bir S' durumuna geçer.
7. Avcı etmen ortamdan r ödülünü alır ve bulanık veri küpleri güncellenir.
8. Eğer yeni S' durumu sonlanma şartını sağlarsa mevcut deneme biter. Aksi halde S', S'in yerine geçerek öğrenme devam eder.

6.6. Uygulama Sonuçları

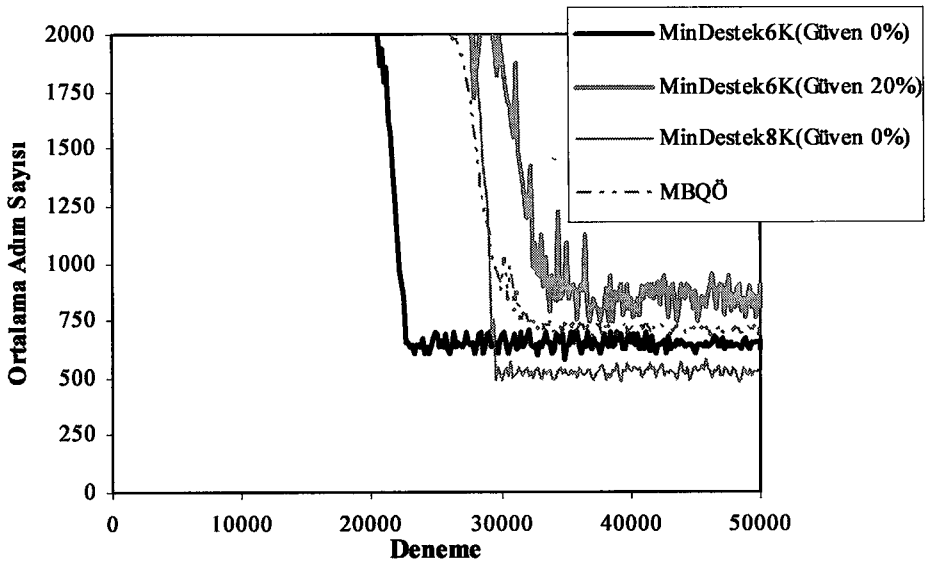
Bu bölümdeki uygulamaların amacı bulanık bağıntı kurallarını kullanarak modüler öğrenen etmenlerin, daha önce verilen standart modüler bulanık öğrenen etmenlere üstünlüğünü göstermektir. Uygulamalardaki öğrenme parametreleri bir öncekiyle aynıdır. Avcı etmenlerin görme derinliği aksi belirtilmediği müddetçe 3 olarak alınmıştır. Uygulamalar, Şekil 2.2.(b)'deki ortamın 25x25 boyutlusunda gerçekleştirilmiştir.

Yeni öğrenme yaklaşımıyla yapılan ilk uygulama farklı minimum destek ve minimum güvendeki öğrenme eğrilerini gözlemlemek içindir. Şekil 6.17 bu uygulamaların sonuçlarını verir. Bu uygulamada öncelikle, her bir durumun gözlenme sayısı belirlenen minimum destek değerine erişinceye kadar minimum güven değeri %0 olarak ayarlanmıştır. Şekilden de görülebileceği gibi minimum destek değerinin 6K'ya atanması durumda elde edilen MinDestek6K(Güven %0) etiketli öğrenme eğrisi MinDestek8K(Güven %0)'ınkinden daha hızlı optimum çözüme yakınsar. Buna karşılık MinDestek8K(Güven %0)'lı öğrenme eğrisinin adım sayısı diğerine göre biraz daha düşüktür. Diğer bir yandan minimum destek değeri 6K'ya sabitlenirse ve minimum güven değeri %20'e yükseltilirse avcı etmenin daha yavaş yakınsar ve daha fazla adımda avı yakalar. Çünkü minimum güven değeri artırılarak, avcı etmene ortamı

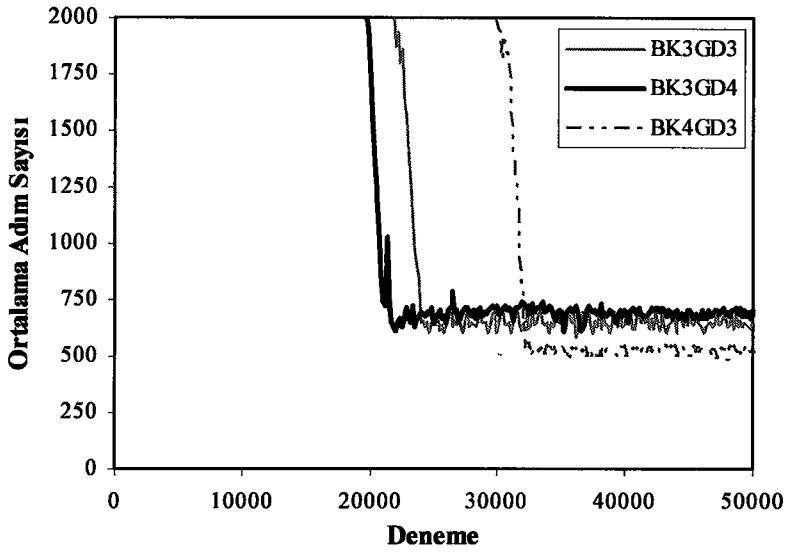
keşfetme fırsatı verilmemiştir. Bu şekil ayrıca durumların tecrübe sayılarını kontrol altına almak olan bu yeni yaklaşımın, ortalama adım ve deneme sayısına göre standart modüler Q-öğrenmeye (MBQÖ) olan üstünlüğünü de gösterir. Önerilen yaklaşımın diğer bir avantajı ise öğrenme eğrilerinin MBQÖ'lü olanlara göre yakınsama noktalarına daha hızlı bir şekilde düşmesidir.

İkinci uygulama farklı bulanık küme ve görme derinliklerinin etkisini araştırmak içindir. Bu maksatla, öncelikle bulanık küme sayısı 3 iken görme derinliği 4'e yükseltildi. Böylece avcı etmenlerin karar uzay çözümlemesi azaltıldı. Bununla birlikte avcı etmenlerin avı daha hızlı fakat daha fazla yakınsama adımında yakaladıkları gözlendi. Bundan sonra, görme derinliği 3 iken bulanık küme sayısı 4'e yükseltilerek bir uygulama yapıldı. Bu uygulamada da, daha yavaş bir yakınsama fakat daha az bir yakınsama adımı elde edildi. Bütün bu tecrübe sonuçları Şekil 6.18'de görülmektedir.

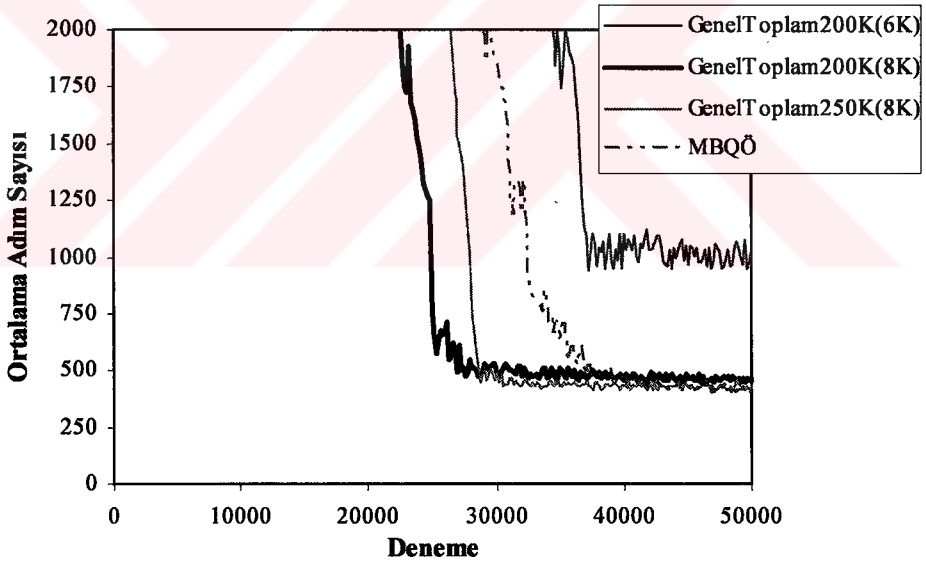
Son iki uygulamada, modüler mimari ile yapılan durum genelleştirme yöntemlerinin etkisi incelendi. Bu uygulamalarda amaç, daha öncekilerde olduğu gibi yeterince tecrübelenmemiş bir durumda etmenin daha yüksek bir seviyeye çıkarak en uygun hareketi belirlemesidir. Şekil 6.19'daki ilk uygulamada iki farklı *Genel Toplam* ve iki farklı minimum destek değerleri belirlendi. Eğer *Genel Toplam* önceden belirlenen eşik değerine erişirse ve bu anda ilgili durumun destek değeri minimum destek değerinin altında ise, etmene durumlarını genelleştirmesi için izin verilir. Bu tecrübelerde etmenlerin görme derinliği 3 iken bulanık küme sayısı 4'e ayarlanmıştır. Şekil 6.19'daki eğrilerde sadece diğer avcı etmenlerin durumları genelleştirilmiş iken Şekil 6.20'de diğer avcı ve avın durumları birlikte genelleştirilmiştir. Böylece Şekil 6.20'den kolaylıkla görülebileceği gibi av ve diğer avcının durumlarını birlikte genelleştirmek iyi bir yaklaşım değildir.



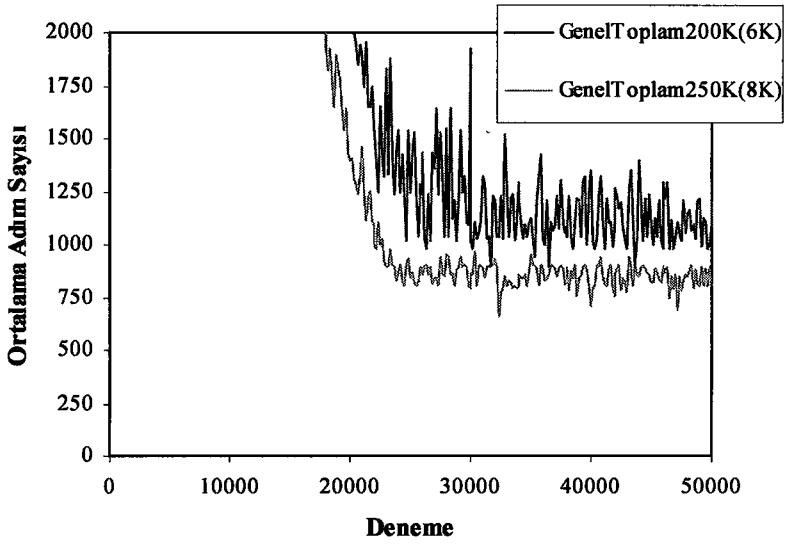
Şekil 6.17. Avcı etmenlerin farklı minimum destek ve güven değerlerine göre öğrenme eğrileri



Şekil 6.18. Görme derinliği ve bulanık küme sayısının etkileri



Şekil 6.19. Diğer avcının durumları genelleştirilerek elde edilen öğrenme eğrileri



Şekil 6.20. Avın ve diğer avcının durumlarının birlikte genelleştirilerek elde edilen öğrenme eğrileri

6.7. Sonuçlar

Bu bölümde, bir çoklu etmen ortamında etmenler tarafından karşılaşılan değişik problemlerin üstesinden gelmek için yeni ve güçlü bir çoklu etmen mimarisi ve öğrenme algoritması sunuldu. Bu problemler başlıca, yavaş yakınsama hızı, yakınsama adım sayısı ve ortam durumlarında diğer etmenleri modellemektir. Bunlara çözüm olarak bu bölümde, modüler mimari, bulanık küme ve bağıntı kurallarının avantajları birleştirildi. Böylece öğrenen etmenlerin durum uzay sayısı daha da azaltıldı. Ayrıca mimariye iç model yapısı entegre edilerek diğer öğrenen etmenleri ele alma problemi çözüldü. Son olarak, yavaş yakınsama problemine bir çare olarak paralel güncelleme yöntemi önerildi. Bu maksatla, av/avcı probleminin homojen ve heterojen olarak adlandırılan iki farklı versiyonunda bazı uygulamalar gerçekleştirilerek önerilen çoklu etmen mimari ve öğrenme yaklaşımının etkisi araştırıldı. Uygulama sonuçları, önerilen yöntemlerin kaliteli çözümler bulduğunu gösterir.

7. SONUÇLAR VE ÖNERİLER

Çoklu etmen sistemler ve veri madenciliği son zamanlarda bilgisayar bilimleri dalında en yaygın olarak çalışılan araştırma konularından ikisidir. Bu tezde, bu iki alan çoklu etmen sistemler üzerinde birleştirilerek yeni öğrenme sistemleri geliştirilmiştir. Veri madenciliğinin öğrenen sistemlerde kullanılması iki farklı bakış açısını da birlikte getirir. Birincisi veri madenciliği yöntemlerinin çoklu etmen sistemlerde kullanılması, çoklu etmen veri madenciliği kavramının oluşmasına neden olur. Çünkü veri madenciliği yöntemlerden biri olan bağıntı kurallarının geleneksel yollarla bulunması, sadece bir etmenin sahip olduğu veritabanı sayesinde gerçekleşir. Buna karşılık, bu tezde birden fazla etmen aynı veritabanını sürekli olarak günceller ve gerektiğinde buradaki verileri kullanır. Bu nedenle, veri tabanında çıkarılan kurallar çoklu etmen çevrim-içi bağıntı kuralları olarak adlandırılır.

Diğer bir bakış açısı ise, çoklu etmen öğrenen sistemlere göredir. Bağıntı kuralları birden fazla etmenin bulunduğu sistemlerde kullanıldığında, bir etmen diğer etmenin hareketini, literatürde yapılan çalışmalardan farklı olarak, Q-değer veya Q-fonksiyon parametrelerinden hiçbirini bilmeden tahmin eder. Tezde, bu maksat için geliştirilen veri küplerinde etmen, sadece diğer etmenin geçmiş hareketlerine bakarak yeni bağıntılar kurmaya çalışır. Önerilen küp aynı zamanda, etmenlerin durum-hareket çiftlerine farklı boyutlardan bakarak etmenlerin en uygun hareketi yürütmesine de yardımcı olur. Dördüncü bölümde gerçekleştirilen uygulama sonuçlarında böyle bir yaklaşımın standart Q-öğrenme algoritmasına olan üstünlüğü kolaylıkla görülebilir.

Çalışmanın bir sonraki adımı olarak, bağıntı kuralları çoklu etmen sistemlerin öğrenmesinde kullanılırken, etmenlerin durum ve hareket uzayları bulanık kümelerle temsil edildi. Böylece veri küplerindeki hücre sayıları bulanıklaştırmanın bir neticesi olarak azaltıldı. Bu küpten elde edilen bulanık bağıntı kuralları ile etmenlerin, kendi hareketlerini, diğer etmenlerin tahmin ettikleri hareketine bağlı olarak belirlendi. Yine bu maksat için av/avcı alanı üzerinde gerçekleştirilen uygulamalarda bu yeni yaklaşımın, hem yakınsama hızı hem de yakınsama adım sayısına göre bulanık Q-öğrenme yaklaşımından daha iyi sonuçlar verdiğini gösterir.

Bir etmenin durum sayısı, ortamdaki diğer etmenlerin sayısına göre üstel olarak artar. Bu nedenle, ikiden fazla etmen barındıran ortamlarda, geleneksel Q-öğrenme yaklaşımları öğrenme problemini çözmede yetersiz kalabilirler. Bu problemin üstesinden gelmek için bu tezde önerilen önceki yaklaşımlarla modüler mimari birleştirilmiştir. Böylece etmen durum sayısı, üstel bir yapıdan kurtulup etmen sayısına göre lineer artan şekle dönüşür. Ortam, bulanık

kümeler ile modellendiğinden, her bir etmen, üyelik dereceleri farklı birden fazla durum gözleyebilir. Bu da, etmen öğrenirken durum-hareket çiftlerinin paralel olarak güncellendiği sonucunu çıkarır. Önerilen farklı bir paralel güncelleme yöntemi ile de, etmenlerin daha hızlı bir yakınsama elde ettiği gözlenmiştir.

Av/avcı probleminin iki farklı ortamı için yapılan uygulamalarda, önerilen çoklu etmen mimari ve bağıntı kurallarına dayalı öğrenme yaklaşımın etkileri görülebilir. Bölüm 6'da verilen uygulamalarda aynı zamanda bulanık küme sayısı ve görme derinliğinin etkileri de karşılaştırmalı olarak verilmiştir. Yine aynı uygulamalar, ortalama adım ve deneme sayısına göre bağıntı kuralları kullanarak öğrenen yöntemin standart modüler bulanık Q-öğrenmeye üstünlüğünü gösterir.

Bu tezde, çoklu etmen öğrenen sistemlere veri madenciliği başarılı bir şekilde dahil edilmişse de, bu alanda yapılması gereken daha bir çok çalışma vardır.

Bu olası çalışmalar aşağıdaki gibi sıralanabilir:

- Önerilen yöntemler sürekli durum uzayı gerektiren daha karmaşık ortamlara uygulanabilir.
- Etmenlerin durum-hareket uzaylarına bazı sınırlamalar veya ağırlıklar verilerek, sınırlı veya ağırlıklı bağıntı kuralları ile öğrenme sağlanabilir.
- Bağıntı kuralları yerine yüne veri madenciliğinin önemli tekniklerinden biri olan sıralı örüntüler (sequential patterns) kullanarak etmenlerin uygun durum-hareket politikaları belirlenebilir.

KAYNAKLAR

1. Stone, P. and Veloso, M., 2000, Multiagent systems: A Survey from a Machine Learning Perspective, Autonomous Robots, Vol.8, no.3.

2. Polat, F. and Guvenir, A., 1994, A Conflict Resolution Based Decentralized Multi-Agent Problem Solving Model, Artificial Social Systems, LNAI 130, Springer-Verlag.

3. Benson, S., 1995, Reacting, planning and learning in an autonomous agent, Ph.D. thesis, Stanford University, Computer Science Department.

4. Sandholm, T .W. and Crites, R. H., 1995, Multi agent reinforcement learning in the Iterated Prisoner's Dilemma, Biosystems, Vol.37, pp.147-166.

5. Tan, M., 1993, Multi-agent reinforcement learning: independent vs. cooperative agents, Proceedings of the International Conference on Machine Learning, pp.330-337.

6. Littman, M. L., 1994, Markov games as a framework for multi agent reinforcement learning, Proceedings of the International Conference on Machine Learning, pp.157-163. San Francisco, CA.

7. Agrawal, R., Imielinski, T. and Swami, A., 1993, Mining association rules between sets of items in large databases, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 207-216.

8. Agrawal, R. and Srikant, R., 1994, Fast algorithms for mining association rules, Proceedings of the International Conference Very Large Databases, pp.487-499.

9. Agrawal, R., Gupta, A., Sarawagi, S., 1997, Modeling Multidimensional Databases, Proceedings of IEEE International Conference on Data Engineering.

10. Park, J. S., Chen, M.S. and Yu, P.S., 1995, An effective hash-based algorithm for mining association rules, Proceedings of ACM SIGMOD International Conference on Management of Data, pp.175-186.

11. Ng, R., Lakshmanan, L. V.S., Han, J. and Pang, A., 1998, Exploratory mining and pruning optimizations of constrained associations rules, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 13-24.

12. Hidber, C., 1999, Online Association Rule Mining, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 145-156.

13. Relue, R., Wu, X. and Huang, H., 2001, Efficient Runtime Generation of Association Rules, Proceedings of ACM International Conference on Information and Knowledge Management, pp.466-473.

14. Han, J., 1997, OLAP Mining: An Integration of OLAP with Data Mining, Proceedings of IFIP International Conference on Data Semantics, pp.1-11.

15. Han, J., 1998, Towards on-line analytical mining in large databases, Proceedings of ACM SIGMOD International Conference on Management of Data.

16. Han, J. and Fu, Y., 1999, Mining multiple-level association rules in large databases, IEEE Transactions on Knowledge and Data Engineering, Vol.11, No.5, pp.798-804.

17. Kamber, M., Han, J. and Chiang, J.Y., 1997, Meta-rule guided mining of multidimensional association rules using data cubes, Proceedings of the International Conference Knowledge Discovery and Data Mining, pp.207-210.

18. Aggarwal, C.C. and Yu, P.S., 2001, A new approach to online generation of association rules, IEEE Transactions on Knowledge and Data Engineering, Vol.13, No.4, pp.527-540.

19. Sycara, K. P., 1998, Multiagent systems, In AAAI Articles, 0738-4602-1998.

20. Mitchell, T., 1997, Machine Learning, McGraw-Hill, NY.

21. Zurada, M. J., 1993, Introduction to Artificial Neural Systems, West Publishing.

22. Kaelbling, L. P., Littman, M. L. and Moore, A. W., 1996, Reinforcement learning: A survey, Artificial Intelligence Research, Vol.4, pp.237-285.

23. Sutton, R.S. and Barto, A. G., 1998, Reinforcement Learning: An Introduction, Cambridge, MA: MIT Press.

24. Hu, J. and Wellman, M. P., 1998, Multiagent reinforcement learning: theoretical framework and an algorithm, Proceedings of the International Conference on Machine Learning, pp.242-250.

25. Abul, O., Polat, F. and Alhajj, R., 2000, Multiagent reinforcement learning using function approximation, IEEE Transaction on Systems, Man and Cybernetics-Part C, Vol.30, No.4.

26. Dayan, P. and Hinton, G.E., 1993, Feudal Reinforcement Learning, Advances in Neural Information Processing Systems 5, pp.271-278.

27. Dayan, P. and Sejnowski, T., 1994, TD(λ) converges with probability 1, Machine Learning, 14:295-301.

28. MacKay, D. J. C., 1999, Introduction to Monte Carlo Methods, in Jordan M.I. (Eds), Learning in Graphical Models, pp.175-204.

29. Mahadevan, S., 1996, Average reward reinforcement learning: Foundations, algorithms and empirical results, Machine Learning, 22:159-196.

30. Rummery, G. A., 1995, Problem Solving with reinforcement learning, PhD Thesis, Cambridge University, Engineering Department.

31. Schwartz, A., 1993, A reinforcement learning method for maximizing undiscounted rewards, In Proceedings of the Tenth International Conference on Machine Learning, CA, pp.298-305.

32. Singh, S. P. and Sutton R. S., 1996, Reinforcement learning with replacing eligibility traces, Machine Learning, 22:123-158.

33. Sutton R. S., 1988, Learning to predict by the method of temporal differences, Machine Learning, 3(1), pp.9-44

34. Sutton, R. S., 1996, Generalization in reinforcement learning: Successful examples using sparse coarse coding, Advances in Neural Information Processing Systems.

35. Watkins, C.J.C.H. and Dayan, P., 1992, Technical note: Q-Learning, Machine Learning, Vol.8, pp.279-292.

36. Bertsekas, D. P., 1995, Dynamic programming and optimal control, Athena Scientific, Belmont, MA.

37. Benda, M., Jagannathan, V., and Dodhiawalla, R., 1985, On optimal cooperation of knowledge source, Technical Report, BCS-G2010-28, Boeing, AI Center.

38. Whitehead, S., Karlsson, J., and Tenenberg, J., 1993, Learning Multiple Goal Behavior via Task Decomposition and Dynamic Policy Merging, Robot Learning, Kluwer Academic Press.

39. Ono, N. and Fukomoto, K., 1996, Multi-agent reinforcement learning: A modular approach, Proceedings of the International Conference on Multi-Agent Systems, pp.252-258.

40. Ono, N. and Fukomoto, K., 1997, A Modular Approach to Multi-Agent Reinforcement Learning, Distributed Artificial Intelligence Meets Machine Learning- Learning in multiagent environments, G. Weiss (Ed), pp.25-39.

41. Park, K. H., Kim, Y.J. and Kim, J. H., 2001, Modular Q-learning based multi-agent cooperation for robot soccer," Robotics and Autonomous Systems, Vol.35, pp.109-122.

42. Au, W.H. and Chan, K.C.C., 1998, An Effective Algorithm for Discovering Fuzzy Rules in Relational Databases, IEEE International Conference on Fuzzy Systems, pp.1314-1319.

43. Touzet, P., 1996, Neural reinforcement learning for behavior synthesis, Proc. CESA'96 IMACS Multiconference, Lille.

44. Baird, L., 1995, Residual algorithms: Reinforcement learning with function approximation, Proceedings of the International Conference on Machine Learning.

45. Beom, H. R. and Cho, H. S., 1995, "A sensor-based navigation for a mobile robot using fuzzy-logic and reinforcement learning," IEEE Transaction on Systems, Man and Cybernetics, Vol.25, No.3, pp.464-477.

46. Yung, N. H. C. and Ye, C., 1999, An intelligent mobile vehicle navigator based on fuzzy logic and reinforcement learning, IEEE Transaction on Systems, Man and Cybernetics, Vol.29, No.2, pp.314-321.

47. Berenji, H. and Vengerov, D., 1999, Cooperation and coordination between fuzzy reinforcement learning agents in continuous state partially observable markov decision processes, Proceedings of the IEEE International Conference on Fuzzy Systems.

48. Berenji, H. and Vengerov, D., 2000, Advantage of cooperation between reinforcement learning agents in difficult stochastic problems, Proc. of IEEE International Conference on Fuzzy Systems.

49. Kaya, M. and Kılıç, A., 2001, Fuzzy-Reinforcement Learning in Cooperative Multi-Agent Systems, Proceedings of the International Symposium on Computer and Information Sciences, Antalya.

50. Nagayuki, Y., Ishii, S. and Doya, K., 2000, Multi-agent reinforcement learning: an approach based on the other agent's internal model, Proceedings of the IEEE International Conference on Multi-agent systems, pp.215-221, Boston.

51. Fayyad U., Piatetsky, G. S. and Smyth, P., 1996, From data mining to knowledge discovery: An overview, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, pp.1-31.

52. Fayyad U., Piatetsky, G. S. and Smyth, P., 1996, Knowledge discovery and data mining: Towards a unifying framework, In Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96), Oregon, USA, pp.82-88.

53. Mannila, H., 1996, Data Mining: Machine learning, statistics and databases, In Proceedings of 8th International Conference on Scientific and Statistical Database Management (SSDBM'96), Sweden, pp.2-9.

54. Mannila, H., 1997, Methods and problems in data mining, In proceedings of 6th International Conference on Database Theory (ICDT'97), Greece, pp.41-55.

55. Zhang, C. and Zhang, S., 2002, Association rule mining, Lecture Notes in Computer Science, Springer Verlag.

56. Chen, M.S., Han, J. and Yu, P., 1996, Data Mining: An overview from database perspective, IEEE Transactions on Knowledge and Data Engineering, Vol.8, No.6, pp.866-883.

57. Chaudhuri, S. and Dayal, U., 1997, An overview of data warehousing and OLAP technology ACM SIGMOD Record, Vol.26, pp.65-74.

58. Zhao, Y., Deshpande, P.M., and Naughton, J.F., 1997, An array-based algorithm for simultaneous multidimensional aggregates, Proceedings of ACM SIGMOD International Conference on Management of Data, pp.159-170.

59. Kaya, M. and Alhajj R., 2003, Integrating Fuzziness with OLAP Association Rules Mining, International Conference on Machine Learning and Data Mining. Leipzig, Germany.

60. Srikant, R. and Agrawal, R., 1996, Mining quantitative association rules in large relational tables, Proceedings of ACM SIGMOD International Conference on Management of Data, pp:1-12

61. Chan, K.C.C. and Au, W.H., 1997, Mining Fuzzy Association Rules, Proceedings of ACM International Conference on Information and Knowledge Management, pp.209-215.

62. Gyenesei, A., 2000, A Fuzzy Approach for Mining Quantitative Association Rules, TUCS Technical Report No.336.

63. Hirota, K. and Pedrycz, 1996, W., Linguistic Data Mining and Fuzzy Modelling, IEEE International Conference on Fuzzy Systems, Vol.2, pp.1448-1496.

64. Hong, T.P., Kuo, C. S. and Chi, S.C., 1999, A fuzzy data mining algorithm for quantitative values, Proceedings of the International Conference on Knowledge-Based Intelligent Information Engineering Systems, pp.480-483.

65. Hong, T.P., Kuo, C.S.and Chi, S.C., 1999, Mining Association Rules from Quantitative Data, Intelligent Data Analysis, Vol.3, pp.363-376.

66. Ishibuchi, H., Nakashima, T. and Yamamoto, T., 2001, Fuzzy Association Rules for Handling Continuous Attributes, Proceedings of IEEE International Symposium on Industrial Electronics, pp.118-121.

67. Kaya, M., Alhajj, R., Polat, F. and Arslan, A., 2002, Efficient Automated Mining of Fuzzy Association Rules, Proceedings of the International Conference on Database and Expert Systems with Applications.

68. Kuok, C. M., Fu, A. W. and Wong, M.H., 1998, Mining fuzzy association rules in databases, SIGMOD Record, Vol.17, No.1, pp.41-46

69. Pedrycz, W., 1998, Fuzzy Sets Technology in Knowledge Discovery, Fuzzy Sets and Systems, 98, pp.279-290.

70. Kaya, M. and Alhajj, R., 2002, Reinforcement learning in multiagent systems: A modular fuzzy approach with internal model capabilities, IEEE International Conference on Tools with Artificial Intelligence, Washington DC.

71. Srikant, R. and Agrawal, R., 1995, Mining generalized association rules, Proceedings of the International Conference Very Large Databases, pp.407-419.

72. Singh, S.P., 1992, Reinforcement learning with a hierarchy of abstract models, Proceedings of the National Conference on Artificial Intelligence, pp.202-207.

73. Kılıç, A., Kaya, M. and Arslan, A., 2002, Finding Sub-optimal Policies Faster in Multi-Agent Systems: FQ-Learning, Proceedings of the International Conference on Intelligent Autonomous Systems, California.

74. Kaelbling, L. P., Littman, M. L. and Cassandra A. R., 1998, Planning and Acting in Partially Observable Stochastic Domains, Artificial Intelligence, Vol.101.

75. Kaya, M., and Arslan, A., 2001, Learning Faster in Cooperative Multi-Agent Systems, Proceeding of the Symposium on Adaptive Agents and Multi-agent Systems, pp.85-89, UK.

76. Kaya, M., Gültekin, İ. and Arslan A., 2002, Modular Fuzzy-Reinforcement Learning in Multi-Agent Systems, Proceedings of the International Conference on Intelligent Autonomous Systems, California.

77. Lin, L. J., 1992, Self-improving reactive agents based on reinforcement learning, planning and teaching, Machine Learning, vol. 8, pp. 293-321.

78. Meuleau, N., Peshkin, L., Kim, K. E. and Kaelbling, L. P., 1999, Learning Finite-State Controllers for Partially Observable Environments," Proceedings of the International Conference on Uncertainty in Artificial Intelligence.

79. Miller, R.J. and Yang, Y., 1997, Association Rules over Interval Data, Proceedings of the ACM SIGMOD, pp.452-461.

80. Singh, S.P., 1994, Learning to Solve Markov Decision Processes, Ph.D. Thesis. University of Massachusetts, Department of Computer Science.

81. Yager, R. R., 1995, Fuzzy Summaries in Database Mining, Proceedings of the Conference on Artificial Intelligence for Application, pp.265-269.

82. Zadeh, L.A., 1965, Fuzzy Sets, Information and Control, Vol.8, pp.338-353.

83. Zhang, W., 1999, Mining Fuzzy Quantitative Association Rules, Proceedings of IEEE International Conference on Tools with Artificial Intelligence, pp.99-102.

ÖZGEÇMİŞ

Mehmet KAYA

Fırat Üniversitesi
Bilgisayar Mühendisliği Bölümü
23119, Elazığ.

e-posta: kaya@firat.edu.tr

1976	Elazığ'da doğdu.
1992-1996	Fırat Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliğini okudu.
1996-1998	Fırat Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliğinde yüksek lisans eğitimini tamamladı.
1997-	Fırat Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliğinde Araştırma görevliliğine atandı.
1999-	Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim dalında Doktora çalışmasına başladı.
2002-2003	Kanada'nın Calgary Üniversitesi Bilgisayar Bilimleri bölümünde aldığı Asistanlık bursu ile 1 yıl boyunca ziyaretçi doktora öğrencisi olarak çalıştı.