İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ARDIŞIK KARAR VERME MODELLERİ VE BİR POMDP UYGULAMASI

YÜKSEK LİSANS TEZİ

Tülay VAROL

Anabilim Dalı : Endüstri Mühendisliği

Programı : Endüstri Mühendisliği

HAZİRAN 2010

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ARDIŞIK KARAR VERME MODELLERİ VE BİR POMDP UYGULAMASI

YÜKSEK LİSANS TEZİ
Tülay VAROL
(507071124)

Tezin Enstitüye Verildiği Tarih : 06 Mayıs 2010
Tezin Savunulduğu Tarih : 08 Haziran 2010

Tez Danışmanı : Doç. Dr. Y. İlker TOPÇU (İTÜ)
Diğer Jüri Üyeleri : Yrd. Doç. Dr. Alp ÜSTÜNDAĞ (İTÜ)
Yrd. Doç. Dr. H. Bersam BOLAT (İTÜ)

HAZİRAN 2010

ÖNSÖZ

Bu tez çalışması kapsamında, ardışık karar verme modelleri ve bu modellerin çözüm yöntemlerinden bahsedilmiş, Endüstri Mühendisliği'nde bu modeller kullanılarak yapılan uygulamalara yer verilmiştir. Tedarik zincirinde yapılan uygulamalar üzerine yoğunlaşılmış, tek ürünlü, kayıp satışlı bir stok kontrol problemi için bir ardışık karar verme modeli kurulmuştur. Bu model, iki farklı veri seti ile bilgisayar ortamında çözdürülerek, elde edilen sonuçlar değerlendirilmiş ve yorumlanmıştır.

Tez çalışmam boyunca, desteği ve ilgisi ile her zaman yanımda olan ve beni yönlendiren tez danışmanım Sayın Doç. Dr. Y. İlker Topçu'ya saygılarımı sunuyor ve teşekkür ediyorum. Ardışık karar verme metotları ve çözümlerine yönelik ayrıntılı bilgi edinmemde bana büyük katkısı olan, başta Prof. Scholomo Zilberstein olmak üzere University of Massachusetts, Amherst, Resource Bounded Reasoning Araştırma Grubu'na teşekkür ediyorum. Özellikle modelin oluşturulmasında büyük katkısı olan, yardımlarını ve desteğini hiçbir zaman esirgemeyen Christopher Amato'ya en içten duygularımla teşekkür ediyorum. Her zaman yanımda olan, beni her zaman destekleyen ve cesaretlendiren sevgili aileme ve arkadaşlarıma gösterdikleri anlayış ve sabırdan dolayı teşekkür ediyorum. Son olarak, yüksek lisans eğitimim boyunca verdiği maddi destekten ötürü TÜBİTAK'a teşekkür ederim.

Haziran 2010 Tülay Varol

İÇİNDEKİLER

Sayfa

KISALTMALAR

ALP : Approximate Linear Programming
DEC-POMDP : Decentralized Partially Observable Markov Decision Process
DEC-ROMDP: Decentralized Markov Decision Process with Restricted Observations
DP : Dinamik Programlama
HSVI : Heuristic Search Value Iteration
KV : Karar Verici
MDP : Markov Decision Process
POMDP : Partially Observable Markov Decision Process
POSG : Partially Observable Stochastic Games
QCLP : Quadratik Constrained Linear Program
YA : Yöneylem Araştırması
YZ : Yapay Zeka

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

ARDIŞIK KARAR VERME MODELLERİ VE BİR POMDP UYGULAMASI

ÖZET

Son yıllarda, teknolojinin hızla gelişmesi, firmaların ürünlerinde çeşitlilik yapmaları, pazarlardaki rekabetin artması...vb. gibi etmenler, talep sürecindeki belirsizliği arttırmıştır. Bu yüzden, birçok stok kontrol problemi, talep sürecinin olasılık dağılımının tam olarak bilinmediği ve zamanla değiştiği çevrelerde meydana gelir. Klasik stok kontrol modelleri, bu tip çevreleri modellemede verimli değildir. Belirsiz çevrelerdeki talep süreci, her durumda, taleplerle ilgili ipuçları içeren gözlemleri kullanarak, maliyetler enküçüklenecek şekilde stoğa eklenecek miktara karar veren kısmi gözlemlenebilir Markov Karar Süreci (Partially Observable Markov Decision Process, POMDP) olarak modellenebilir. Bu modellerin çözümünden elde edilen optimal planlar, stok yöneticilerine stok kontrol kararlarını en uygun şekilde vermede yardımcı olur.

Bu tez kapsamında, ardışık karar verme modellerinden Markov Karar Süreci (Markov Decision Process, MDP), Kısmi Gözlemlenebilir Markov Karar Süreci (Partially Observable Markov Decision Process, POMDP) ve Dağıtık Kısmi Gözlemlenebilir Markov Karar Süreci (Decentralized Partially Observable Markov Decision Process, DEC-POMDP) ve bunların çözüm yöntemlerinden bahsedilmiştir. Ayrıca, tek ürünlü bir stok kontrol problemi için, bekleyen siparişlerin olmadığı, karşılanamayan taleplerin kayıp olarak ele alındığı, talebin sabit olmayan ve kısmi gözlemlenebilir olduğu bir kısmi gözlemlenebilir Markov karar süreci (Partially Observable Markov Decision Process, POMDP) modeli geliştirilmiştir. Bu modelde, tedarik zincirinde tek bir perakendeci karar verici olarak göz önüne alınmıştır. Önerilen bu model, iki veri seti kullanılarak, Trey Smith'in yazmış olduğu ZmdpSolve adlı çözücüde çözdürülmüştür. Çözücünün girdi olarak aldığı dosyaların oluşturulması için C dilinde bir kod oluşturulmuştur. Elde edilen deneysel sonuçlar yorumlanarak, oluşturulan iki ayrı örnek kıyaslanmıştır. Sonuçta bu modellerin bu tip bir stok kontrol problemi için avantajları ve dezavantajlarından bahsedilmiştir. Ayrıca, gelecekte bu modelin nasıl geliştirilebileceği ile ilgili bazı önerilere de yer verilmiştir.

SEQUENTIAL DECISION MAKING METHODS AND A POMDP APPLICATION

SUMMARY

In recent years, quickly developing technology, the increasing variability of production, and growing competition in markets, broaden the uncertainty of the supply/demand process. So many of the inventory control problems occur in an environment where the demand process' probability distribution is not completely known and changes over time. Traditional inventory control models are inefficient for modeling these kinds of environments. The demand process in these uncertain environments can be modeled as a Partially Observable Markov Decision Process, POMDP, which determines the quantity that will be added to the stock such that all costs are minimized by using observations that include clues about the demands in each state. The optimal policies, which are obtained by solving those models, help inventory managers to make decisions in the most suitable way.

The framework of this thesis describes Markov Decision Process (MDP), Partially Observable Markov Decision Process (POMDP) and Decentralized Partially Observable Markov Decision Process (DEC-POMDP) which are sequential decision making methods. Also it describes an improved model for an inventory control problem using a single product in a POMDP where there are no backorders (unsatisfied demands are lost). The demand is non-stationary and partially observable. Here, it is accepted that only one retailer in the supply chain is the decision maker. The proposed model is solved with two different data sets by "ZmdpSolve", written by Trey Smith. The input file for that program was generated using another program written in the "C" programming language. Two different implementations are compared when interpreting the results of the inventory control model. The pros and cons for these kind of inventory control models are mentioned. Also, future suggestions about how the model can be improved are included in this work.

1. GİRİŞ

Günümüz rekabet koşulları içerisinde şirketlerin, amaçlarını gerçekleştirme ve başarı elde edebilmeleri için, her konuda en optimal şekilde karar almaları ve yer aldıkları pazarlarda rakiplerine üstünlük sağlamaları şarttır. Aynı pazarda yer alan şirketler arasındaki rekabette başarı, sadece ürünlerdeki kalite ve farklılıklara değil, aynı zamanda şirketin güvenilirliğine ve müşteri hizmet seviyesine de bağlıdır. Bunun için, pazardaki değişikliklere hızlı bir şekilde uyum sağlanması ve yenilikçi bir yapıya sahip olunması, ayrıca gelen taleplerin zamanında ve tam olarak karşılanması ve müşteri memnuniyetinin arttırılması gerekmektedir.

Bu amaçlardan biri olan, şirketin karlı bir şekilde müşteri talebini karşılayabilme amacı, şirketlerin yer aldığı tedarik zincirinin en uygun şekilde yönetilmesi ile gerçekleştirilebilir. Bu tedarik zincirinde, stok kontrol kararlarının optimal olarak alınması, müşteri memnuniyetini arttırmakla beraber, aynı zamanda, tedarik zinciri maliyetlerini de azaltır.

Günümüzde teknolojinin hızla gelişmesi, bir ürünün piyasaya çıkmasından çok kısa bir süre sonra, bu ürünün daha gelişmiş bir versiyonunun ya da bu ürün ile aynı işlevi gören ve daha kaliteli olan farklı bir ürünün de piyasaya çıkmasını sağlamaktadır. Bu yüzden ürünlerin yaşam çevrimi kısalmıştır. Ürün yaşam çevriminin kısalması, talebin belirsizliğini arttırmaktadır. Aynı zamanda, ürün çeşitliliğinin artması da, daha önceden durgun olan talep sürecini hareketlendirir ve bu süreçteki belirsizliği arttırır. Piyasadaki rekabet de talep sürecinin belirsizliğini arttırmaktadır. Kısacası, pratikte, birçok stok kontrol problemi, talep belirsiz olduğundan, talep süreci ile ilgili bilginin zamanla elde edildiği dinamik çevrelerde meydana gelir.

Tedarik zincirindeki bireyler, bu tip stok kontrol problemlerinin çözümleri için verimli kontrol planları oluşturan uygun karar verme modellerine ihtiyaç duymaktadırlar. Sadece kısa dönemde değil, uzun dönem içerisinde gerçekleşebilecek olası tüm durumlar göz önüne alınarak oluşturulan ardışık karar verme modellerinin kullanılması, tedarik zincirinde verimliliği arttırır.

Bu tip problemler için literatürde 40 yıla yakındır çalışmalar yapılmaktadır. Fakat bu çalışmaların hemen hemen hepsi sabit talep üzerine yoğunlaşmaktadır. Yani, talep sürecinin parametrelerinin bilinen bir dağılımla ifade edildiği durumlar çok nadir olmasına rağmen, birçok stok yöneticisi, gerekli hesaplamaları kolaylaştırmak için, kararlarını, bu dağılımın bilindiği varsayımına dayanarak vermektedir.

Bu tez kapsamında, tek ürünlü bir stok kontrol problemi için, bekleyen siparişlerin olmadığı, karşılanamayan taleplerin kayıp olarak ele alındığı, talebin sabit olmayan ve kısmi gözlemlenebilir olduğu bir kısmi gözlemlenebilir Markov karar süreci (Partially Observable Markov Decision Process, POMDP) modeli geliştirilmiştir. Bu modelde, tedarik zincirinde tek bir perakendeci karar verici olarak göz önüne alınmıştır. Karar verici, taleplerle ilgili tahminlere dayanarak stoğa eklenecek olan miktara, maliyetler enküçüklenecek şekilde karar verecektir. Bu tahminler, gelecek taleplerle ilgili ipuçları taşıyan gözlemlerden elde edilir.

Bu bölümde, karar verme, belirsizlik altında karar verme, ardışık karar verme ve bu alanda şimdiye kadar yapılan çalışmalardan bahsedilecektir. Bölüm 2 ise en çok kullanılan ardışık karar verme modellerini ve bunların bazı çözüm yöntemlerini içermektedir. Bölüm 3'te tedarik zincirinde ardışık karar verme ve bu alanda yapılan çalışmalar anlatılmaktadır. Bölüm 4, yukarıda bahsedilen modelin ayrıntılarını ve deneysel sonuçları içermektedir. Son bölümde ise, genel olarak elde edilen sonuçlardan ve gelecekte yapılabilecek olan çalışmalardan bahsedilecektir.

1.1 Karar Verme

Karar verme, belli bir amaç doğrultusunda, çeşitli alternatifler arasından en uygun olanın seçilmesi eylemidir. Hayatımızın her aşamasında, sık sık karar verme durumuyla karşı karşıya kalırız. Bu kararlar çok küçük, sıradan kararlar olabileceği gibi, hayatımızın akışını etkileyen önemli, büyük kararlar da olabilir. Bu yüzden, karar verme süreci önemli bir süreçtir.

Karar verme sürecini oluşturan bileşenler;

- Karar verici (KV),
- Karar ortamı,
- Amaçlar,

- Alternatifler,
- Kaynaklar

şeklinde ele alınabilir (Evren ve Ülengin, 1992).

Karar verme eylemini, sadece bir bireyin karar vermesi olarak değil, bir takımın, bir ailenin, bir işletmenin ya da herhangi bir örgütün karar vermesi olarak da ele alabiliriz. Özellikle, işletmeler, her gün üretim, satış, finansman gibi konularla ilgili karar verme durumundadırlar. Ayrıca işletme yöneticilerinin vereceği stratejik kararlar da işletme için hayati bir önem taşımaktadır. İşte bu önemli kararların alınması, günlük yaşantımızda karşımıza çıkan basit ve küçük kararların alınması kadar kolay değildir. Bu yüzden de karar verme eylemini daha profesyonel olarak, bilimsel açıdan incelemek gerekmektedir. Yıllardan beri birçok farklı bilim dalı, farklı koşullarda en uygun kararların nasıl alınabileceğini gösteren karar verme modelleri üzerine çalışmışlardır.

Bir karar verme durumuyla karşı karşıya kaldığımızda, göz önüne alacağımız alternatiflerin her birinin, o alternatifi seçtiğimiz takdirde ne gibi bir sonuç sağlayacağını kesin olarak bilemeyebiliriz. Kısacası, karar verme sürecinde, gelecek belirsizlik içerebilir ya da KV bulunduğu çevre ve durum hakkında tam olarak bilgi sahibi olmayabilir. Bu tip karar verme problemleri, *"Belirsizlik Altında Karar Verme Problemleri"* olarak adlandırılır. Bu problemlerin çözümü için geliştirilen özel modeller daha sonraki bölümlerde açıklanacaktır.

1.2 Belirsizlik Altında Karar Verme

Belirsizlik altında karar verme problemlerinin yapısı anlatılmadan önce, bu çalışmada kullanılan bazı temel kavramların açıklanmasında fayda vardır. Yapay Zeka perspektifinden bakılarak açıklanan bu kavramlar, bir KV'yi ve KV'nin etkileşimde bulunduğu ortamı, sadece insan ve çevresi olarak değil, farklı çeşit varlıklar ve ortamlar olarak da ele alabileceğimizi gösterir. Örneğin, bir KV, bir insan olabileceği gibi, bir robot ya da bir makine de olabilir.

Bu kavramlar kısaca şu şekilde açıklanabilir;

Temsilci (Agent):

Bir temsilci, çevresiyle iletişim kurabilen ve bu çevreyi algılayabilen varlıktır. Karar verme problemleri açısından bakıldığında, KV olarak da nitelendirilebilir. Bu çalışmada, karar verme modelleri anlatılırken temsilci kavramı yerine *"karar verici (KV)"* kavramı kullanılacaktır.

Temsilciler, gösterdikleri davranışlara göre çeşitli sınıflara ayrılabilirler. Bunlardan en önemlileri aşağıdaki şekilde açıklanabilir (Russell ve Norvig, 1995);

- *Basit Refleks Temsilci (Simple Reflex Agent):* Basit refleks temsilci, algılara direkt olarak yanıt verebilen temsilcidir.
- *Model Temelli Refleks Temsilci (Model-Based Reflex Agent):* Bu tip bir temsilci, algı geçmişine bağlı olan durumunu takip ederek, kısmi olarak gözlemlenebilir çevrede, göremediği algıların izini sürer.
- *Hedef Temelli Temsilci (Goal-Based Agent):* Bu tip temsilciler bir hedefe sahiptir ve hedeflerini başarma doğrultusunda hareket ederler.
- *Fayda Temelli Temsilci (Utility-Based Agent):* Birçok çevrede, yüksek kalitede davranışlar oluşturmak için, hedefler tek başına yeterli olmaz. Hedefler temsilcinin durumunda "mutlu" ya da "mutsuz" gibi nitelikler oluşturabilirler. Yani, bir durum, diğer bir duruma tercih ediliyorsa, bu durumda temsilci daha mutlu olacaktır. Mutluluktan kasıt, bu durumun temsilciye daha fazla fayda sağlayacak olmasıdır. Fayda temelli bir temsilci, işte bu beklenen mutluluğunu, yani faydayı enbüyüklemeye çalışır.
- *Rasyonel Temsilci (Rational Agent):* Verilen bir algı sırasına göre, performans ölçütünün beklenen değerini enbüyükleyecek şekilde hareket eden temsilcidir.

Temsilci Fonksiyonu (Agent Function):

Bu fonksiyon, temsilcinin herhangi bir algı dizisine cevap olarak bir eylem seçmesini tanımlar.

Performans Kriteri (Performance Measure):

Çevre içerisinde temsilcinin davranışlarını değerlendiren kriterlerdir.

İş Çevresi (Task Environment):

Bu çevre, performans ölçütü, dış çevre, sensörler ve harekete geçiricilerini içerir. Bir temsilci dizaynı yaparken ilk adım, iş çevresini mümkün olduğu kadar tam ve doğru bir şekilde tanımlamaktır. Bu çevre,

- Tam Olarak Gözlemlenebilir Çevre - Kısmi Olarak Gözlemlenebilir Çevre,
- Deterministik Çevre – Stokastik Çevre,
- Bölümlü Çevre – Ardışık Çevre,
- Ayrık Çevre – Sürekli Çevre,
- Statik Çevre – Dinamik Çevre,
- Tek Temsilcili Çevre – Çok Temsilcili Çevre

gibi farklı kategorilerde olabilir.

Çizelge 1.1'de bazı çevre örnekleri ve bunların karakteristikleri görülmektedir.

Çizelge 1.1 : İş çevre örnekleri ve karakteristikleri (Russell ve Norvig, 1995)

Kare Bulmaca	Tam	Deterministik	Ardışık	Statik	Ayrık	Tekli
Saatli Satranç	Tam	Stratejik	Ardışık	Yarım	Ayrık	Çoklu
Poker	Kısmi	Stokastik	Ardışık	Statik	Ayrık	Çoklu
Tavla	Tam	Stokastik	Ardışık	Statik	Ayrık	Çoklu
Taksi Kullanma	Kısmi	Stokastik	Ardışık	Dinamik	Sürekli	Çoklu
Medikal Tanı	Kısmi	Stokastik	Ardışık	Dinamik	Sürekli	Tekli
Görüntü Analizi	Tam	Deterministik	Bölümlü	Yarım	Sürekli	Tekli
Parça Toplama Robotu	Kısmi	Stokastik	Bölümlü	Dinamik	Sürekli	Tekli
Rafineri Kontrol	Kısmi	Stokastik	Ardışık	Dinamik	Sürekli	Tekli
Etkileşimli İngilizce Öğretme	Kısmi	Stokastik	Ardışık	Dinamik	Ayrık	Çoklu

Yukarıda bahsedilen çevre kavramı, her zaman tam olarak gözlemlenebilir, statik ve deterministik olmayabilir. Yani, temsilcinin karar vereceği çevre belirsizlik içerebilir ve temsilciler belirsizlik altında karar verme durumunda olabilirler. Bu şekildeki bir çevrede, temsilci ilk olarak optimal kararı verebileceği uygun bir plan oluşturmalı ve daha sonra da bu planı uygulamalıdır. Belirsiz çevrede, temsilci bu planını uygularken algılarını kullanmalı, çevresini tam olarak algılayamasa bile en azından ipuçları içeren gözlemler elde ederek çevresiyle ilgili bilgi sahibi olmalıdır. Böylece beklenmeyen bir durumda planında değişiklikler yapabilir.

Temsilciler, belirsizlik altındaki çevrelerde tam ve doğru olmayan bilgileri kullanmak zorundadırlar. Bilgilerin tam ve doğru olmama olasılığı, çevredeki belirsizliğin miktarına göre değişir. Belirsizliklerde, eylemlerin gelecek etkileri tam olarak tahmin edilemeyebilir. Fakat mümkün çıktıların olasılıkları önceden bilinebilir. Örneğin bir para atma eyleminin sonucu belirsizlik içerse de, yazı veya tura geleceği olasılıklara bağlı olarak bilinmektedir.

Temsilcinin belirsizlik altında bir seçim yapması için, ilk olarak farklı planların mümkün çıktıları arasında tercihlere (preferences) sahip olması gerekir. Bu tercihlerin temsilciye sağlayacağı faydalar, fayda teorisi (utility theory) kullanılarak bulunur. Fayda teorisi, her bir durumun faydalı olma derecesini verir. Böylece temsilci, daha yüksek faydayı veren durumu seçecektir (Russell ve Norvig, 1995).

Temsilcinin ne istediğinin belirlenmesine yardımcı olan fayda fonksiyonu ile temsilcinin neye inanması gerektiğinin olasılıklarını içeren olasılık fonksiyonu birleştirilerek temsilcinin ne yapması gerektiğini gösteren karar teorisi (decision theory) oluşturulur.

Karar teorisindeki temel düşünceye göre, temsilci kendisine *"En Büyük Beklenen Fayda (Maximum Expected Utility)"* yı sağlayan eylemi seçecektir (Russell ve Norvig, 1995).

Belirsizlik altında karar verme eylemi, tek bir temsilci tarafından yapılabildiği gibi, birden fazla temsilci tarafından da yapılabilir. Bir alt bölümde, çok temsilcili çevrelerde karar verme konusu ayrıntılı olarak ele alınacaktır.

1.2.1 Çok temsilcili çevrelerde belirsizlik altında karar verme

Birden fazla temsilcinin olduğu çevreler, çok temsilcili çevrelerdir. Bu tür çevrelerdeki temsilciler işbirliği içerisinde (cooperative) olabilecekleri gibi, birbirlerinin rakibi (competitive) durumunda da olabilirler;

- *İşbirliği İçerisindeki Temsilciler (Cooperative Agents):* Bu temsilciler, bir takım olarak da düşünülebilir. Takımdaki her bir birey aynı amaç doğrultusunda hareket etmektedir. Yani her bir temsilci, ortak bir faydayı enbüyüklemeye çalışır. En optimal kararların alınabilmesi için, her bir temsilcinin planları aynı anda göz önüne alınarak, ortak bir birleşik plan oluşturulur. Aynı futbol takımında oynayan oyuncular, bu duruma bir örnektir. Takımdaki her bir oyuncu, aynı amaç doğrultusunda hareket eder.
- *Rakip Temsilciler (Competitive Agents):* Bu tip temsilciler birbirleriyle çelişen fayda fonksiyonlarına sahiptirler. Buna örnek olarak satranç gibi oyunlar verilebilir. Bu oyunda, her bir temsilci, karşısındaki rakibinin tüm olası hamlelerini düşünmek zorundadır ve ona karşı üstünlük elde etmeye çalışmaktadır. Bu tip oyunlar ayrıntılı olarak *"Oyunlar Teorisi"* nin inceleme konusudur.

1.3 Ardışık Karar Verme

Ardışık karar verme problemleri, temsilcinin fayda fonksiyonunun, bir kararlar dizisine bağlı olduğu problemlerdir. Temsilci, her adımda algılarının doğrultusunda yeni bir eylem seçer. Sonuçta, çözüm olarak, uygun eylemler dizisinin oluşturduğu bir plan elde eder. Özetle bir ardışık karar verme problemi şu bileşenlerden oluşur;

- *Temsilci (Agent):* Temsilci, ardışık kararı verecek olan KV'dir. Buradan itibaren temsilci yerine *"karar verici (KV)"* kavramı kullanılacaktır.
- *Çevre (Environment):* KV'nin etkileşimde olduğu her şey çevre olarak sayılabilir. KV, çevreyi kısmi ya da tam olarak gözlemleyebilir.
- *Durumlar (States):* KV'nin içerisinde bulunduğu mevcut durumunu niteler. Seçilecek eylem, durum üzerinde bir etki oluşturacak ve mevcut durum değişecektir.
- *Eylemler (Actions):* KV'nin, karar verme sürecinde seçeceği alternatiflerdir.

- *Ödül (Reward):* KV'nin, belli bir durumda, herhangi bir eylemi seçmesi halinde elde edeceği kazançtır. KV, tüm karar aşamaları boyunca elde edeceği beklenen toplam ödülü enbüyüklemeye çalışır.
- *Plan (Policy):* KV'nin, tüm karar aşamaları boyunca elde edeceği beklenen toplam ödülünü enbüyükleyen eylemler dizisinin oluşturduğu plandır.

1.3.1 Ardışık karar verme modellerinin tarihsel gelişimi

Karar verme, belirsizlik altında karar verme, belirsizlik altında ardışık karar verme konuları ve bu konular üzerinde yapılan tüm çalışmalar, başta Yapay Zeka (YZ), Yöneylem Araştırması (YA) ve Oyunlar Teorisi olmak üzere, birçok farklı alanın ortak çalışma konusudur. Bu alanların, bu konular üzerinde yaptığı çalışmaların tarihsel gelişimi şu şekilde açıklanabilir.

1657: Huygens (1657), bu dönemde, rakip durumunda ve işbirliği içerisinde olan karar vericilerin etkileşimleri üzerine bilimsel ve matematiksel bir yaklaşım getirmiştir. Rakip durumundaki karar vericilerin incelenmesi, Oyunlar Teorisi'nin başlangıcını oluşturmuştur.

1928: Von Neumann (1928), iki oyunculu, sıfır toplamlı oyunlarda, en küçüklerin en büyükleri yöntemini geliştirmiştir.

1950: Karar teorisinin kullanımı, bu yıllarda, ekonomi, finans ve yönetim bilimleri gibi birçok alanda bir standart haline gelmiştir.

Nash (1950), Nash denkliğini geliştirmiştir.

1953: Shapley (1953), bu dönemde, Bellman'dan önce bir değer iterasyonu algoritması geliştirmiştir. Fakat tam olarak doğru sonuçlar elde edememiştir.

1957: Bellman (1957) tarafından sıralı karar veme problemlerinin modellenmesinde kullanılan *"Markov Karar Süreci"* yöntemi geliştirilmiştir. Bellman, ardışık karar verme problemlerine modern bir yaklaşım getirmiş, genel olarak dinamik programlama (DP) yaklaşımını, özel olarak da değer iterasyon algoritmasını önermiştir.

1958: YZ'nın kurucusu McCarthy (1958), *"Ortak Algılı Program (Programs with Common Sense*)" makalesinde, pratik sonuç çıkarımı konusunu ele almıştır.

1960-1970: Birçok olasılıklı çıkarım probleminin çözümünde, o dönemlerde yeni keşfedilmiş olan Bayes ağları kullanılmıştır. Bu yaklaşım, tecrübelerin göz önüne alınarak, öğrenmenin gerçekleştirilmesini sağlar.

Howard (1960) doktora tezinde, plan iterasyonu ve sonsuz ufuklu problemlerin çözümü için ortalama ödül fikrini ortaya atmıştır.

1965: Astrom (1965) tarafından, bir POMDP probleminin, sürekli durum uzayına sahip bir MDP şeklinde ifade edilebileceği gösterilmiştir.

1967: Harsanyi (1967) tarafından Bayes-Nash denkliği önerilmiştir.

1971: POMDP'nin çözümüne yönelik ilk algoritma, Sondik (1971) tarafından, doktora tezi çalışmasında geliştirilmiştir.

1974: Birçok YZ araştırmacısı, karar teorik metotlarını, medikal karar verme alanında uygulamışlardır. Ayrıca bu dönemde, Feldman ve Yakimovsky (1974), bu metotları, görme (vision) konusu üzerinde uygulamıştır.

1976: Keeney ve Raiffa (1976) tarafından yazılan *"Kararlar ve Çoklu Amaçlar: Tercihler ve Değer Değiştokuşları"* kitabı çok özellikli fayda teorisine bir giriş niteliği taşımaktadır. Bu kitap, çok amaçlı fayda fonksiyonu için gerekli parametreleri göz önüne alan bilgisayar uygulamalarını ele almıştır.

Van Nunen (1976) tarafından, değiştirilmiş plan iterasyonu algoritması geliştirilmiştir.

1977: Feldman ve Sproull (1977), karar teorik metotlarını, planlama problemlerinin çözümü için kullanmışlardır.

1978: YZ araştırmacısı Simon (1978), *"Tatmin Edici Karar Verme"* üzerine yaptığı çalışması ile ekonomi dalında Nobel ödülü almıştır. Bu çalışmada Simon, karar verme sürecinde optimal kararın hesaplanmasından ziyade, kararların "yeteri kadar iyi" olduğunu göstermenin daha iyi bir yol olacağından bahsetmiştir.

Puterman ve Shin (1978) tarafından, bir diğer değiştirilmiş plan iterasyonu algoritması geliştirilmiştir.

1980: Bu zamana kadar, karar ağaçları, basit karar problemlerinin ifade edilmesinde en önemli araç olarak kullanılmıştır.

1984: Howard ve Matheson (1984) tarafından karar ağları ve etki diyagramları geliştirilmiştir.

1985: Cheeseman'ın (1985) *"Olasılığın Savunmasında (In Defense of Probability)"* adlı makalesi, YZ alanında, olasılık konusunda önemli bir gelişme olmuştur.

1986: Horvitz ve Heckerman (1986), karar teorisi adımlarını izleyerek rasyonel olarak hareket eden ve insan düşünce yapısını taklit etmeyen bir uzman sistem fikrini öne sürmüşlerdir.

Kumar ve Varaiya (1986) tarafından kısmi olarak gözlemlenebilir, stokastik çevreleri ele alan kontrol sistemleri geliştirilmiştir.

Shachter (1986) tarafından, karar ağlarına dayanan bir karar verme metodu geliştirilmiştir.

1987: Papadimitriou ve Tsitsiklis (1987), MDP'nin hesaplama karmaşıklığı ve bunun sonuçlarından bahsetmişlerdir.

1988: Bu konudaki diğer bir önemli gelişme de, Pearl'in (1988) *"Akıllı Sistemlerde Olasılıklı Çıkarım (Probabilistic Reasoning in Intelligent Systems)"* adlı makalesidir. Pearl, YZ'da olasılık ve fayda teorisini derinlemesine inceleyen ilk çalışmayı gerçekleştirmiştir. Bu çalışmada yer verilen belirsizlik altında karar verme ve pratik sonuç çıkarımı metotları, 1990'lı yıllarda fayda temelli KV'lerin kullanılmasına zemin hazırlamıştır.

Bu dönemde, Horvitz *ve diğ.* (1988), YZ'da bir temel teşkil edecek olan, *"Beklenen faydayı enbüyükleyen rasyonelliğin kullanımı"* fikrini öne sürmüşlerdir. Horvitz'in geliştirdiği karar teorik uzman sistemleri, yaygın bir alanda kabul görmüştür.

1988-1989: Sutton (1988) ve Watkins (1989), Markov Karar Süreci (Markov Decision Process, MDP) çözümünde, takviyeli öğrenme (reinforcement learning) metotları üzerinde çalışmışlardır. Bu çalışmalar, YZ dünyasında, MDP'nin tanınmasında önemli bir rol oynamıştır.

Dean ve Kanazawa (1989) tarafından dinamik karar ağlarını kullanan bir karar verici yapısı önerilmiştir.

1991: Koenig (1991), ilk olarak YZ'daki planlama problemleri ve MDP arasındaki ilişkiden bahsetmiştir.

1992: Stewart (1992), *"Çok Amaçlı Karar Verme"* yöntemini önermiştir.

1993: Williams ve Baird (1993) tarafından, eş zamanlı olmayan plan iterasyonu algoritması geliştirilmiştir.

1994: YZ alanına en büyük katkı, Cassandra *ve diğ.* (1994)'nin, kısmi olarak gözlemlenebilir Markov karar süreci (Partially Observable Markov Decision Process, POMDP) değer iterasyonu üzerine geliştirdikleri *"Şahitlik Algoritması (Witness Algorithm)"* dır.

Puterman (1994) tarafından tam ve kısmi gözlemlenebilir çevreler arasındaki ayrımdan bahsedilmiştir.

Oyunlar Teorisi ve MDP, *"Markov Oyunları (Markov Games)"* adı altında biraraya getirilmiştir (Littman, 1994).

1995: Bu dönemde, belirsizlik altında karar verme problemleri üzerine yoğunlaşıldığı için, karar-teorik tekniklerine ilgi artmıştır. Wellman (1995) da bu konu üzerinde çalışan araştırmacılardan biridir.

1998: Henzinger ve Sastry (1998) tarafından, hem ayrık, hem sürekli bileşen içeren çevreleri ele alan hibrid kontrol sistemleri geliştirilmiştir.

1999: Dorf ve Bishop (1999) tarafından tam olarak gözlemlenebilir, deterministik çevreleri ele alan klasik kontrol sistemleri geliştirilmiştir.

2000: Boutilier *ve diğ.* (2000), geçiş ve değer fonksiyonlarının sembolik gösterimleri üzerine bir çalışma yapmışlardır.

2000'li yıllara kadar, ardışık karar verme ile ilgili birçok model ve bu modellerin çözümüne yönelik birçok algoritma geliştirilmiştir. Bu yıllarda, son olarak, çok karar vericili ardışık karar verme durumları göz önüne alınmış ve dağıtık kısmi olarak gözlemlenebilir Markov karar süreci (DEC-POMDP) üzerine yapılan çalışmalar artmıştır.

1.4 Ardışık Karar Verme Modelleri ve Uygulama Alanları Üzerine Yapılan Çalışmalar

1.4.1 Ardışık karar verme modellerinin gelişimi üzerine yapılan çalışmalar

Ardışık karar verme modellerinin çözümü ve optimal planların elde edilmesi, pratikte hiç kolay değildir. Bu yüzden bilgisayar bilimcileri, optimal yöntemlerin geliştirilmesi üzerine çalışmaktadırlar. 2000'li yıllarda bu konuda yapılan çalışmalar hız kazanmıştır. Littman (1996), yazdığı doktora tezinde, MDP, POMDP, Markov oyunları gibi farklı karar verme modellerinin ele aldığı çevrelerde, optimal davranışların bulunmasıyla ilgili birçok sonucu göz önüne almıştır. Burada, algoritmaların yapısı ve karmaşıklığı üzerinde durulmuştur. Yost ve Washburn (2000), bir dual doğrusal programlama modeli kullanarak yeni bir POMDP modeli geliştirmişlerdir. Bu teknik, bir uçağın saldırı yapacağı hedeflerin belirlenmesini amaçlayan, askeri bir probleme uygulanmıştır. POMDP'lerin optimal çözümünün kolaylaştırılmasına yönelik olarak, Rusmevichientong ve Roy (2001), yüksek boyutlu düşünülen durumlar üzerinde bir değer fonksiyonu tanımlamak yerine, durum uzayını daha küçük boyuta indirgeyerek DP tekniklerini daha verimli bir şekilde uygulama yollarını araştırmışlardır.

Daha sonraki yıllarda, DEC-POMDP'de, yerel optimal birleşik bir plan bulunmasında, iletişim hareketlerinin nasıl sunulabileceği (Nair *ve diğ.* 2004) ve iletişimin değerinin nasıl hesaplanabileceği (Carlin ve Zilberstein, 2009) üzerine çalışmalar yapılmıştır.

Wang ve Schmolze (2005), mantık ve karar teorisini birleştirdikleri bir çalışmada, özellikle, POMDP'nin kısa ve etkili gösterimi, ayrıca eylem ve gözlemlerden sonra düşünülen durumları güncellemek için faydalı bir metot ve verilen bir başlangıç durum uzayında, beklenen toplam ödülü enbüyükleyen optimal planın bulunması için, sezgisel bir yöntem geliştirmişlerdir.

DEC-POMDP'nin çözüm yöntemlerinin geliştirilmesi ile ilgili en önemli sayılabilecek çalışma, bu tezde de ayrıntılı olarak yer alan, Bernstein'ın (2005) DEC-POMDP problemlerinin çözümü için geliştirdiği yöntemleri içeren doktora tezidir. Burada, DEC-POMDP çözümünün en kötü karmaşıklık durumu analiz edilmiş ve DEC-POMDP'nin çözümü için optimal plan iterasyonu algoritması geliştirilmiştir.

Stokastik sonlu durum denetçileri ve geliştirilen bir bağıntı aleti kullanılmış ve ayrıca iki değer koruma algoritması geliştirilmiştir.

Porta *ve diğ.* (2006), sürekli uzaya sahip POMDP modellerini ele alarak, bu modellerin optimize edilmesi için yeni bir yöntem önermişlerdir. Ayrıca, bu çalışmada, sürekli POMDP'lerin, ayrık gözlem, eylem ve sürekli duruma sahip oldukları hallerde, değer fonksiyonlarının konveks, parçalı ve doğrusal olduğu gösterilmiştir. *"Önceliklendirilmiş Değer İterasyonu (Prioritized Value Iteration)"* adında yeni bir algoritma içeren diğer bir çalışmada da, önceliklendirilmiş denetçilerin gösterimine yer verilmiş ve düşünülen noktalarda denetçi işlemlerinin sıralanmasının önemi belirtilmiştir (Shani *ve diğ.* 2006). Amato *ve diğ.* (2007), bu sonlu durum denetçilerini kullanarak, problemleri, optimal denetçiyi istenen büyüklükte tanımlayan bir *"Quadratik Kısıtlı Doğrusal Program (Quadratik Constrained Linear Program, QCLP)"* olarak modelleyen bir yaklaşım önermişlerdir. Seuken ve Zilberstein (2008), tüm mevcut ardışık karar verme modellerini karşılaştırılıp kıyaslayarak, bu modellerin birbirleriyle olan ilişkilerini, zayıf ve güçlü yönlerini anlatan bir çalışma yapmışlardır. Burada, işbirliği içerisindeki karar vericilerin olduğu sistemler incelenirken, aynı zamanda oyun-teorik yaklaşımlı, rakip durumundaki ilişkilere de yer verilmiştir.

POMDP ve DEC-POMDP'nin optimal plan bulma problemlerinde, optimal çözümün geliştirilmesi için çevrim içi algoritma (Liu ve Zeng, 2008; Wu *ve diğ.* 2009), çapraz dağınım metodu (Oliehoek *ve diğ.* 2008), nokta temelli algoritmalarda geliştirilen bir önişleme metodu (Bian *ve diğ.* 2008), ağ dağıtık POMDP gibi modeller için ölçeklenebilir nokta temelli DP algoritması (Kumar ve Zilberstein, 2009a), olay saptamalı çok karar vericili Markov karar süreci (Kumar ve Zilberstein, 2009b) algoritmaları ve metotları önerilmiştir. Ross *ve diğ.* (2008), optimal kontrol problemini inceledikleri bir çalışmada, model parametrelerinin tam olarak bilinmediği, sürekli ve kısmi gözlemlenebilir çevreleri ele alarak bir Bayes yaklaşımı ve parçacık filtre algoritması önermişlerdir. Başka bir çalışmada, hiyerarşi keşif probleminin, kısmi gözlemlenebilir ortamlarda, en büyük olasılıklı yaklaşım kullanılarak çözülebileceği gösterilmiştir (Toussaint *ve diğ.* 2008).

Amato ve Zilberstein (2009), sonsuz ufuklu DEC-POMDP için, hedeflerin varlığının, verimli planlama algoritmalarının geliştirilmesine yardımcı olacağını göstermişlerdir. Ayrıca bu alanda, Bernstein *ve diğ.* (2009), DEC-POMDP'nin çözümü için optimal

plan iterasyonu algoritmasının yanında, optimale yakınsamadan fedakarlık eden, plan iterasyonu algoritmasının sezgisel (heuristic) versiyonunu da önermişlerdir.

İşbirliği içerisinde olan karar vericileri inceleyen DEC-POMDP'nin dışında, rakip durumdaki karar vericilerin modellendiği *"Kısmi Gözlemlenebilir Stokastik Oyunlar (Partially Observable Stochastic Games, POSG)"* algoritmaları üzerine de çalışmalar yapılmıştır (Kumar ve Zilberstein, 2009c). Bu modeli ölçeklendirmeye olanak sağlayan sınırlandırılmış yaklaşım teknikleri üzerinde durulmuştur.

2009 yılında, sonlu durum denetçileri kullanılarak ifade edilen POMDP ve DEC-POMDP modellerini bir doğrusal olmayan programlama modeli olarak ifade eden yeni bir yaklaşım (Amato *ve diğ.* 2009a), aynı zamanda DEC-POMDP'de DP algoritmalarının verimliliğini arttırmak için durum uzayı analizinin erişilebilirliliği üzerine kurulu, *"Artımlı Plan Üretimi (Incremental Policy Generation)"* adında yeni bir algoritma geliştirilmiştir (Amato *ve diğ.* 2009b).

Aynı yıl, Petrik ve Zilberstein (2009)'ın yapmış oldukları bir çalışmada, yaklaşımların sanal döngülere neden olduğu problemlerde, *"Yaklaşık Doğrusal Programlama (Approximate Linear Programming, ALP)"* çözümlerinin kalitesinin zayıflığının nedenleri anlatılmıştır.

Srivastava *ve diğ.* (2009), beş doğal boyutu ile birlikte, genelleştirilmiş bir planın basit ve tam tanımını ele almış ve tüm bu beş boyutu içeren plan genelleştirmesi için bir yaklaşım önermişlerdir. Doshi *ve diğ.* (2009), aynı yıl, çok karar vericili, kısmi gözlemlenebilir çevrelerde ardışık karar verme problemleri için yeni bir grafiksel gösterim geliştirmişlerdir.

1.4.2 Ardışık karar verme modellerinin uygulama alanları üzerine yapılan çalışmalar

Belirsizlik altında ardışık karar verme modelleri, gerçek hayatta birçok farklı alandaki problemlere uygulanmıştır. Bu alanlar ve bu alanlarda yapılan uygulamalar, Cassandra (1998) tarafından belli başlıklar altında gruplandırılmıştır.

Bir alt bölümde, Cassandra'nın oluşturduğu alt başlıklar da kullanılarak, bu alanda yapılan uygulamalara yer verilecektir.

1.4.2.1 Bilimsel uygulamalar

Mobil robot kontrolü

Bu alanda, genellikle, uzayın keşfi, okyanus derinliklerinin keşfi ve derin denizlerdeki atıkların temizlenmesi için kullanılan mobil robotların kontrolü üzerine çalışılmıştır. Bir robotun bulunduğu çevre, genellikle tam olarak gözlemlenebilir bir özelliğe sahip olmadığından ve robotun görüşü sınırlı olduğundan, bu tip problemlerin modellenmesinde POMDP yöntemi kullanılmıştır (Simmons ve Koenig, 1995; Nourbakhsh *ve diğ.* 1995; Cassandra *ve diğ.* 1996; Theocharous ve Mahadevan, 2002). Son yıllarda da bu alanda yapılan çalışmalar genişletilmiştir. Ayrıca bazı çalışmalarda robot kullanımı farklı bir amaç için ele alınmış, ev, hastane gibi kapalı çevrelerde, insanlara yardımcı olmak için dizayn edilmiş robotların POMDP modellemesi yapılmıştır (Pineau *ve diğ.* 2003; L*ó*pez *ve diğ.* 2005; Boger *ve diğ.* 2005). Diğer bir çalışmada da, cisimleri tutup kaldıran bir robot kol probleminde, optimal planların elde edilmesi amacıyla, POMDP kullanılarak yeni bir metot geliştirilmiştir (Hsiao *ve diğ.* 2007). Aynı yılda yapılan diğer bir araştırma ise, bir gezici robot problemi için hiyerarşik bir POMDP formülasyonu önermiştir (Foka ve Trahanias, 2007).

Birden fazla robotun bir arada bulunduğu ve işbirliği içerisinde hareket etmek zorunda olduğu çevrelerde ise bu tip problemler, DEC-POMDP kullanılarak modellenmiştir. Özellikle bu çalışmalarda, uzay keşfinde kullanılan robotların karar vermesi ve gezgin robotların işbirliği incelenmiştir (Emery-Montemerlo *ve diğ.* 2004; Emery-Montemerlo *ve diğ.* 2005).

Makine görüşü

Karar verme modelleri makine görüşü problemlerinde de uygulanmaktadır. Makine görüşünden kasıt, kafa ve el hareketlerini izleme, mimik tanıma gibi YZ sistemleridir. Bu alanda özellikle görsel dikkat (Bandera *ve diğ.* 1996) ve mimik tanıma (Darrell ve Pentland, 1996) konularında çalışılmıştır.

Davranışsal ekoloji

Bir organizmanın davranışlarının incelenmesi üzerine yapılan çalışmalarda da karar verme modelleri kullanılmaktadır (Mangel ve Clark, 1988).

Diyalog yönetimi

Diyalog sistemlerinde kontrol işlemi, otomatik konuşma tanıma çok güvenli bir işlem olmadığından ve bu yüzden konuşmanın durumunun bilinemiyor olmasından dolayı zordur. Bu yüzden, diyalog yönetimi konusunda karar verme modellerinin kullanıldığı bir takım çalışmalar yapılmaktadır (Roy *ve diğ.* 2000). 2007 yılında yapılan bir çalışmada, bu tip problemlerin çözümü için yeni bir POMDP optimizasyon tekniği geliştirilmiştir (Williams ve Young, 2007).

Çok erişimli yayın kanallarının dizaynı

Bu tip bir modelde bir mesaj kanalını kontrol eden iki karar verici bulunmaktadır. Karar vericiler mesaja sahip olma ya da olmama durumunda olabilirler. Bu iki karar verici her adımda, mesaj gönderme ya da göndermeme eylemlerinden birini tercih etmektedirler. Her bir adımda yalnızca bir karar verici mesaj gönderebilir. Eğer iki karar verici aynı adımda mesaj gönderirse bir çakışma meydana gelmektedir. Bu yüzden karar vericilerin amacı, kanal üzerinde mesajlarının çarpışmaya uğramadan iletilebilmesidir. İşte bu gibi çok erişimli yayın kanallarının dizaynında DEC-POMDP modeli kullanılmaktadır (Ooi ve Wornell, 1996).

1.4.2.2 Askeri uygulamalar

Hareketli hedef arama

Askeri uygulamalar olarak bakıldığında, bu tarz bir probleme en güzel örnek denizaltı savaşları olarak verilebilir. Ayrıca hareketli füze platformlarının yerleştirilmesi problemi de yine bu alanla ilgili bir problemdir (Eagle, 1984; Pollock, 1970). Bu tip problemlerin modellenmesinde de yine belirsizlik altında ardışık karar verme modelleri kullanılabilir.

Hedef tanımlama

Askeri alanda, yaklaşan bir uçağın zararsız bir uçak mı yoksa düşman uçağı mı olup olmadığını anlamak kolay bir iş değildir. Radar ve radyo yayınları bu konuda bir kısım bilgi sağlıyor olsalar da, her zaman tam olarak doğru bilgi vermeyebilirler. Sensörlerdeki yanlışlıklar ve uçak hareketlerinin belirsizliği, bu alanda POMDP kullanımı ile modellenip çözülmeye çalışılmıştır (D'Ambrosio ve Fung, 1996).

Silah yerleştirme

Savaş uçakları, belirli hedefleri vurmak için füze taşımaktadırlar. Amaçları, bu füzeleri belirlenen hedeflere atarak bu hedeflere zarar vermektir. Her yaptığı atıştan

sonra uçak, bu atışın kaydını tutacak ve performansını değerlendirecektir. Örneğin iki hedefi ve iki füzesi olan bir uçak, ilk füzesini ilk hedefe atmış fakat istediği zararı elde edememişse ve eğer bu ilk hedefin tahrip edilmesi, ikinciye göre daha önemliyse, uçak ikinci füzesini de ilk hedef üzeinde kullanacaktır. İşte bu tip bir problemin modellenmesinde POMDP kullanılmıştır (Yost, 1998).

1.4.2.3 Sosyal uygulamalar

Eğitim

Bu tip problemlerde, kavramların en iyi şekilde öğretilmesini sağlayan yolu bulmak amaçlanmaktadır. n tane kavramın bir bireye öğretilmesi problemi, bu amaçla kurulacak bir POMDP modeli ile çözülebilir (Karush ve Dear, 1967; Smallwood, 1971).

Medikal tanı

Karar verme problemleri tıp alanında da kullanılmaktadır. Bir hastaya teşhis konulması ve hastanın tedavi şeklinin bulunması, ardışık karar verme modelleri kullanılarak çözülebilir (Hu *ve diğ.*1996; Hauskrecht, 1997; Hauskrecht ve Fraser, 2000). Eğer bu medikal tanı, bir kişi için değil de bir grup hasta için yapılacaksa bu tip bir problem de DEC-POMDP şeklinde modellenebilir (Smallwood *ve diğ.* 1971).

1.4.2.4 Ticaret uygulamaları

Ağ sorun giderme

Bir ağ sorun giderme örneği olarak, geniş, bağlantılı, elektrik dağıtma ağı düşünülebilir (Thiebeaux *ve diğ.* 1996). Eğer herhangi bir parça bozulur ya da şalter atarsa, geniş bir alan bundan etkileniyor olabilir. Bu problemdeki amaç, olabildiğince hızlı bir şekilde, hatayı onarmaktır. Ağın bağlantılı yapısı ve uzaktan erişilebilir şalter açıp kapama sensörlü, uzaktan kontrol edilebilir tuşların varlığı, ana kontrol istasyonundan, sistemi yeniden düzenlemeye ve sorun gidermeye olanak sağlayabilir. Aslında, sistem yeniden düzenlenip hangi şalterde problem olduğu izlenirken, kontrol edici, tamir ekibini yollamadan önce bozulan parçanın yerini uzaktan tespit edebilir. Fakat hatanın yeri mümkün olduğu kadar hızlı tespit edilmek istenirken, istenilen sayıda müşteriye servis sürdürülemeyebilir. Buradaki kısmi gözlemlenebilirlik, şalter açıp kapama sensörlerinin sayısının sınırlı olmasından kaynaklanır. Bu sensörler kullanılarak, her zaman tam doğru bilgi de elde edilmeyebilir. Bu tip problemlerin modellenmesinde ardışık karar verme

modellerinden yararlanılır. Diğer farklı çeşit ağların özellikleri de bu örnekteki ile aynıdır.

Dağıtık veritabanı sorguları

Karar süreci modellerinin kullanımı, sorgu dağıtım kontrolünün geliştirilmesi için iyi bir yoldur (Segall, 1976). Ağ örneğinde olduğu gibi, ya çok fazla donanım ya da çok fazla yoğunluk gerektiren global bir sistem durumunun sürdürülebilmesi olası değildir. Fakat ağ yoğunluğu, mevcut ağ durumu hakkında gözlemler sağlayabilir. Bu şekilde kısmi gözlemlenebilir bir model, yani POMDP modeli ile problem çözülebilir.

Pazarlama

POMDP modelinin bu alanda uygulanması ve sonuç olarak optimal planların elde edilmesi, şirketlerin pazarlama kaynaklarının daha verimli bir şekilde kullanılmasını sağlar. Bu alanda, pazarlama kampanyaları üzerine çalışmalar yapılmıştır (Rusmevichientong and Van Roy, 2001).

Anket dizaynı

Pazarlama uygulamalarıyla da ilişkisi bulunan anket dizayn probleminin amacı, her sorunun anket içerisindeki yerini, en doğru cevap elde edilecek şekilde yerleştirecek optimal sorular sırası elde etmektir (White, 1976).

Şirket politikası

Anonim şirketleri ve politikaları, ardışık karar verme modellerinin uygulanabileceği diğer bir alandır. İç kontrollerin yapılmasından (Hughes, 1977) muhasebe kontrollerinin tutulmasına (Kaplan, 1969) kadar olan tüm şirket eylemlerinin çıktıları tam olarak istenilen her şeyi göstermeyebilir veya organizasyonun mevcut durumunu tam olarak yansıtmayabilir. Bir organizasyon için kurulmuş olan MDP, POMDP ya da DEC-POMDP modellleri, tüm yapının analiz edilebilmesine olanak sağlar. Aynı zamanda organizasyonu izleyerek optimal ticaret politikaları üretir.

Tüketici davranışları

Tüketici davranışlarının tespiti ve değerlendirilmesi problemleri de ardışık karar verme modelleri ile ifade edilebilir (Lipstein, 1965). Bu şekilde tüketiciye özel kampanyalar yapılabilir, yeni pazarlama stratejileri geliştirilebilir.

1.4.2.5 Endüstri uygulamaları

Makine bakımı

Fabrikalarda makinelerin bakım, onarım, kontrol işlemleri de POMDP başta olmak üzere, ardışık karar verme yöntemleri kullanılarak modellenebilir. Bu alanda şimdiye kadar birçok çalışma yapılmıştır (Eckles, 1968; Ross, 1971; Smallwood ve Sondik, 1973; Pierskalla ve Voelker, 1976; White, 1977; 1979; Benazera ve Chanthery, 2008). Oluşturulan modellerde, her durumda makinenin performansı hakkında elde edilen gözlemler kullanılarak makinenin bakımının olup olmaması ya da makinenin yenisiyle değiştirilmesi, kontrol edilmesi... vs gibi kararlardan biri seçilmektedir.

Yapısal kontrol

Yapısal kontrol işlemi de makine bakımı problemleriyle benzerlik göstermektedir. Farklı olarak, burada kaldırım, köprü, bina… vb gibi daha uzun ömürlü ve büyük varlıkların bakımı ve kontrolü yapılmaktadır. Zamanla bu varlıkların parçaları, kendi kendilerine bozulmaya uğrayabilir. Önceki yıllarda, yapısal kontrolün POMDP olarak modellenmesine yönelik bir çalışma yapılmıştır (Ellis *ve diğ.* 1995).

Asansör kontrol planları

Yaşam açısından kritik sonuçları olmasa da, asansör kontrolü için iyi ve verimli kontrol planları elde edilmesi önemlidir. Bu da, POMDP'de endüstri alanındaki bir başka uygulama çeşididir (Crites 1996).

Balıkçılık endüstrisi

Deniz yaşamındaki çeşitli canlıların popülasyonları tam olarak bilinmemekte, belirsizlik içermektedir. Denizdeki çeşitli popülasyonların kontrolü ve izlenmesi için ardışık karar verme modelleri kullanılarak bazı çalışmalar yapılmıştır (Lane 1989).

Stok kontrolü

Talebin belirsizlik içerdiği durumlarda, her bir aşamadaki sipariş ya da üretim miktarının belirlenmesi problemleri için ardışık karar verme modelleri kullanılmaktadır. Bu alanda birçok çalışma yapılmıştır. Bu alanda yapılan uygulamalar, daha sonraki bir bölümde ayrıntılı olarak incelenecektir.

1.4.3 Ardışık karar verme problemlerinin endüstri mühendisliğinde kullanımı

Anthony R. Cassandra (1998), özellikle POMDP'de farklı alanlarda yapılan bazı uygulamaların, model içerisindeki kullanımlarından bahsetmiştir. Bu tezde, bu uygulamalar içerisinden, Endüstri Mühendisliği alanında yapılan uygulamaların bir POMDP modelinde ne şekilde ele alındığından bahsedilecektir.

1.4.3.1 Makine bakımı

Cassandra'nın bahsettiği, genel bir makine bakımı üzerine kurulan POMDP modelinin bileşenleri şu şekilde gösterilebilir;

- *Durumlar:* Durum olarak, genellikle, makine parçalarının iç durumları göz önüne alınır.
- *Eylemler:* Makine bakımında seçilebilecek eylemler şu şekilde sıralanabilir;
    1. Bakım yapma,
    2. Parçaları değiştirme,
    3. Makineyi değiştirme,
    4. Makinede işlem yapmaya devam etme,
    5. Kontrol etme (Maliyet ve verimlilik bakımından çeşitlilik gösteren bir çok farklı kontrol tipi olabilir)
- *Gözlemler:* Gözlem olarak; makine performansı ele alınabilir. Bunun yanında, her bir kontrol eylemi sonucunda oluşan çeşitli çıktılar da gözlem olarak düşünülebilir.
- *Amaç:* Amaç, genellikle, işlem maliyetini enküçüklemek ya da üretim kapasitesini enbüyüklemektir.

1.4.3.2 Yapısal kontrol

Bu alanda oluşturulacak bir POMDP modelinin bazı bileşenleri şu şekilde gösterilebilir;

- *Durumlar:* Makine bakımında olduğu gibi, modelin durumu, parçaların ya da materyallerin iç durumu olarak ele alınabilir.

- *Eylemler:* Eylem olarak çeşitli kontrol ya da destek seçenekleri göz önüne alınabilir.
- *Amaç:* Yapısal kontrol problemlerinde, makine bakımının aksine, yapısal parçaların sökülmesi kolay değildir. Bu da, daha az optimal kontrol durumlarının oluşmasına neden olur. Burada verimli kontrol planlarının oluşturulması çok önemlidir. Çünkü amaç, kazançtaki düşüşü enküçüklemekten daha çok, insanların hayatını tehlikeye atmamaktır.

1.4.3.3 Asansör kontrol planları

Bu alanda oluşturulacak bir POMDP modelinin bazı bileşenleri şu şekilde ifade edilebilir;

- *Durumlar:* Durum olarak;
    1. Asansörlerin bulunduğu kat ve hareket yönü,
    2. Bekleyen yolcuların sayısı ve bekledikleri kat

    göz önüne alınabilir.
- *Eylemler:* Eylem olarak;
    1. Her bir asansörün hangi yöne gönderileceği,
    2. Asansörün hangi katta duracağı,
    3. Asansörün hangi kattan geçeceği

    eylemleri göz önüne alınabilir.
- *Gözlemler:* Gözlem olarak;
    1. Her bir katta bulunan, yolcuların hangi yöne çıkmak istediğini belirten yön tuşlarının hangisine basıldığı,
    2. Asansörün içerisinde, yolcuların çıkmak istedikleri katı belirten tuşlardan hangisine basıldığı

    durumları gözlem olarak ele alınabilir.

Bu modeldeki kısmi gözlemlenebilirlik, asansöre daha fazla bilgi verici tuş eklenerek azaltılabilir. Fakat bunun ek bir maliyet getireceği unutulmamalıdır.

1.4.3.4 Balıkçılık endüstrisi

Bu alanda oluşturulacak bir POMDP modelinin bazı bileşenleri şu şekilde gösterilebilir;

- *Eylemler:* Eylem olarak,
    1. Stoğu tamamlama,
    2. Av yasağını uygulama,

    eylemleri göz önüne alınabilir.
- *Amaç:* Bu modeldeki amaç, popülasyondaki ince dengeyi korumaktır.

1.4.3.5 Ağ sorun giderme

Bu modeldeki bileşenler şu şekildedir;

- *Durum:* Burada,
    1. Ağın mümkün ayarları,
    2. Ağdaki tüm parçaların mümkün durumları

    modelin durumları olarak kabul edilebilir.
- *Eylemler:* Bu modelde eylemler, uzaktan kontrol edilen elektrik tuşları kullanılarak alınabilen eylemlerdir.
- *Gözlemler:* Şalter açıp kapama sensörlerinin sağladığı gözlemlerdir.
- *Amaç:* Amaç, müşteri servis maliyetlerini ya da hata tespit süresini enküçüklemek olarak ele alınabilir.

1.4.3.6 Dağıtık veritabanı sorguları

Dağıtık veritabanı sorguları için kurulan bir POMDP modelindeki bazı bileşenler şu şekilde ifade edilebilir;

- *Durumlar:* Durum olarak bilgi kaynaklarının durumları ele alınabilir; Yukarı, aşağı, az yüklenmiş, çok yüklenmiş… vs.
- *Eylemler:* Sorgunun yapıldığı çeşitli bilgi kaynaklarının kullanımı eylem olarak ele alınabilir.

- *Gözlemler:* Gözlem olarak;

    1. Bir önceki sorgunun sonuçları,
    2. Genel ağ trafiği,
    3. Spesifik ağ mesajları

    ele alınabilir.

1.4.3.7 Pazarlama

Pazarlama üzerine kurulan bir POMDP modelindeki bazı bileşenler şu şekilde ifade edilebilir;

- *Durumlar:* Bu modelde durumlar;

    1. Ürünün hitap edilen kitleye uygun olması,
    2. Ürünün hitap edilen kitleye uygun olmaması,

    şeklinde ele alınabilir.

- *Eylemler:* Eylemler;

    1. Potansiyel müşteriye, satıcı tarafından sorulacak sorular,
    2. Satıcı tarafından yapılabilen, müşterinin cevaplamasını gerektirecek her şey

    olarak ele alınabilir.

- *Gözlemler: G*özlemler;

    1. Müşteri tarafından satıcıya gösterilen tepkiler,
    2. Belli bir zaman içerisindeki müşteri davranışları,
    3. Kişinin satın alma geçmişi

    olarak ele alınabilir.

- *Amaç:* Bu modeldeki amaç, ürün satışını enbüyüklemek ya da hedef kitleye uygun olmayan kişiler üzerinde harcanan süreyi enküçüklemek olarak ele alınabilir.

1.4.3.8 Anket dizaynı

Böyle bir POMDP modelindeki bazı bileşenler şu şekilde ifade edilebilir;

- *Durumlar:* Durum olarak, anketin uygulanacağı kişi tipi ele alınabilir.
- *Eylemler:* Eylem olarak, anketi oluşturacak spesifik soruların seçimi ele alınabilir.
- *Gözlemler:* Anket yapan kişinin verdiği cevaplar gözlem olarak ele alınabilir.
- *Amaç:* Amaç, anketten en doğru bilgiyi elde etmektir.

2. ARDIŞIK KARAR VERME MODELLERİ

2.1 Markov Karar Süreci

Bir karar verme durumuyla karşı karşıya kaldığımızda, seçebileceğimiz birçok alternatifle karşılaşırız. Bu durumda, en iyi alternatifi seçmek için, o an elde edeceğimiz etkileri göz önünde bulundurmaktan ziyade, bu alternatiflerin uzun süreli etkilerine de bakmamız gerekir. Fakat uzun dönem etkilerini görmek, anlık etkileri görmek kadar kolay değildir. Bazen anlık etkisi zayıf olan bir eylem, diğer alternatiflere göre, uzun dönemde daha iyi sonuçlar veriyor olabilir. Bu durumda, gelecek kazanımlar ve şimdiki ödül arasında en iyi takası sağlayarak bizi mümkün olan en iyi sonuca ulaştıran eylem seçilmelidir.

Fakat gelecek birçok belirsizlik içerdiğinden bunu yapmak o kadar da basit değildir. Mevcut eylemlerin gelecekteki sonuçları tam olarak kestirilemez ve hatta bazen o eylemin gelecekte önemli olup olmayacağı bile bilinemez. Bu tip problemleri modellemenin bir yolu, 1950'li yıllarda bir sıralı karar verme modeli olarak geliştirilmiş olan Markov Karar Süreci (Markov Decision Process, MDP)' ni kullanmaktır. Böylece, belirsiz çevrelerde karar verme problemleri modellenebilir.

Şekil 2.1'de görüldüğü gibi ve daha önceden de bahsedildiği üzere, karar verme, YZ ve YA alanlarının ortak çalışma konusudur. YA, belirsiz etkileri olan eylemler arasında tercih yaparken, faydayı enbüyükleme amacını göz önüne alarak karar vermede KV'ye yardımcı olur. Bu konuda geliştirilen karar verme yöntemlerinde DP kullanılır. YZ'da ise, bir hedefi gerçekleştirme amacıyla verilecek olan kararlar ele alınır. YZ'da karar verme konusunda, genellikle, çeşitli arama teknikleri kullanılır. Ayrıca YZ;

- Karar-teorik planlama (Decision-theoretic planning)
- Takviyeli öğrenme (Reinforcement learning)

gibi alanları ele almaktadır.

İşte hem YZ hem de YA açısından bakıldığında, karar verme yaklaşımlarının ortak noktası olarak MDP ele alınabilir.

Şekil 2.1 : Yapay Zeka ve Yöneylem Araştırması açısından karar verme yaklaşımları (Zilberstein, 2010a)

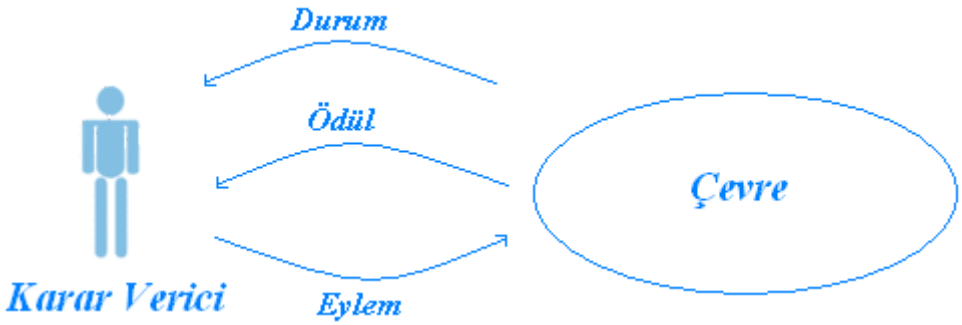
Bir MDP modeli $< S, A, T, R >$ şeklinde gösterilir. Bu bileşenler şu şekilde tanımlanır;

- *Durumlar (States):* KV'nin bulunması muhtemel tüm durumlarının oluşturduğu durumlar kümesidir; $\forall s \in S$
- *Eylemler (Actions):* KV'nin seçebileceği mümkün eylemler (alternatifler) kümesidir; $\forall a \in A$
- *Geçiş Fonksiyonu (Transition Function):* KV'nin $\forall s, s' \in S$ ve $\forall a \in A$ olmak üzere, mevcut durumu s ve seçtiği eylem a bilindiğinde diğer tüm muhtemel s' durumlarına geçiş olasılıklarını veren fonksiyondur;

  T: SxA $\rightarrow$ ΔS, T(s'\s, a)

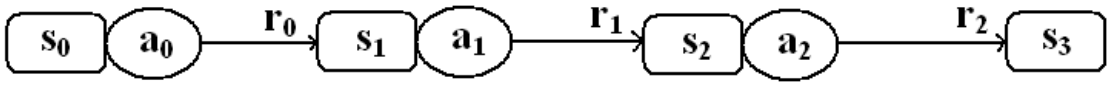
  Markov varsayımına (Markov assumption) göre, KV'nin bir durumdan bir eylem seçerek bir sonraki duruma geçme olasılığı, geçmişinde bulunduğu tüm durumlara değil, yalnızca bir önceki durumuna bağlıdır.

  Yani; $P(s_t \backslash s_{t-1}, s_{t-2}, \ldots., s_1, a) = P(s_t \backslash s_{t-1}, a)$' dir.
- *Anlık Ödüller (Immediate Rewards):* KV'nin $\forall s \in S$ ve $\forall a \in A$ olmak üzere, s durumunda a eylemini seçmesi halinde elde edeceği kazancı veren fonksiyondur; R: SxA $\rightarrow \Re$, R(s, a)



Şekil 2.2 : Bir MDP modelinde karar vericinin çevreyle olan etkileşimi

Stokastik çevrede hareket eden tek bir KV'nin modellenmesinde MDP kullanmak çok faydalı ve doğru bir yoldur. KV, belirli zaman aralıklarında, ardışık olarak çevresiyle iletişim kurar. Şekil 2.2'de görüldüğü gibi, her bir adımda, girdi olarak, çevredeki durumunu gözlemler ve hemen ardından, çıktı olarak, seçilecek olan eylemi belirler (Bernstein, 2005). KV'nin seçmiş olduğu eylem, KV'ye nümerik bir ödül değeri verir ve aynı zamanda, stokastik olarak, KV'nin içinde bulunduğu durumdan yeni bir duruma geçişini sağlar (Bernstein, 2005). MDP'de KV'nin seçeceği eylemin yaratacağı etkiler, belirsizlik altında olduğu halde, KV'nin bulunduğu durum hiçbir şekilde belirsizlik altında değildir (Littman, 1996).



Şekil 2.3 : Bir MDP modelinde, her adımda karar vericinin seçtiği eylemlerle durum değiştirmesi ve bir ödül elde etmesi

KV, Şekil 2.3'te görüldüğü gibi, bir eylem seçtikten sonra, durumunu her zaman tam olarak gözlemleyebilecektir. KV'nin amacı, beklenen uzun dönem ödülünü enbüyüklemektir. Bunun için KV, belirli bir s_0 durumundan başlar ve her bir durumda seçeceği her bir a eylemin r anlık ödülünü göz önüne alarak, uzun dönemde farklı alternatif eylem planlarını karşılaştırır. Bu alternatif eylem planlarının her biri "*plan (policy)*" olarak adlandırılır. Bu planların içerisinden optimal olanı, yani uzun dönem ödüllerini enbüyükleyen plan, MDP'nin çözümüdür.

Planlar "*sabit plan (stationary policy)*" ve "*sabit olmayan plan (non-stationary policy)*" olmak üzere iki çeşittir.

Sabit plan, $\pi: S \rightarrow A$, her bir durum için seçilecek eylemi gösteren plandır. Bu planda, eylemlerin seçimi, zaman ufkundan bağımsız olarak, sadece duruma bağlıdır.

Sabit olmayan plan ise, zamana göre sıralanmış bir durum-eylem plan dizisidir. Sabit olmayan plan, $\delta =< \pi_t, \dots \pi_1 >$'da yer alan π_t, mevcut durumun bir fonksiyonu olarak $t.$ zaman aralığından son zaman aralığına geçerken seçilecek eylemi göstermektedir (Littman, 1996).

2.1.1 Performans kriterleri ve değer fonksiyonları

Bir MDP modelinde, KV beklenen uzun dönem ödülünü enbüyüklemek amacıyla iki farklı performans kriteri kullanabilir;

Sonlu Ufuklu (Finite-Horizon) Performans Kriteri: Bir MDP modelinde ardışık durumlar için belirlenmiş bir zaman ufku uzunluğu varsa, çözüm için sonlu ufuklu performans kriteri kullanılır. Yani böyle bir durumda, KV'nin, önceden belirlenmiş sayıdaki zaman aralıklarında karar vereceği göz önüne alınacaktır. Sonlu ufuklu durumlarda, her zaman optimal, sabit olmayan bir plan vardır.

Değer fonksiyonu (value function), T ufuk uzunluğu ve s durumundaki δ planı için (2.1) denkleminde gösterildiği şekilde hesaplanır (Bernstein, 2005).

$$V^{\delta}(s) = E_{s,\delta}\left[\sum_{t=0}^{T-1} R(s_t, a_t)\right] \quad \textbf{(2.1)}$$

$R(s_t, a_t)$, t zaman aralığında, s durumunda, a eyleminin seçilmesi halinde elde edilecek olan anlık ödülü göstermektedir.

Optimal değer, s durumu için (2.2) denkleminde görüldüğü gibi hesaplanır (Bernstein, 2005).

$$V^{*}(s) = max_{\delta}\left(V^{\delta}(s)\right) \quad \textbf{(2.2)}$$

$V^{\delta}(s)$, s durumu ile başlayan δ planının değerini ifade etmektedir.

Değer fonksiyonu, planla benzerlik gösterir. Planda, her bir durum için bir eylem tanımlanırken, değer fonksiyonunda her bir durum için nümerik bir değer tanımlanır.

MDP'nin amacı, s_0 başlangıç durumu için olan optimal değeri, yani optimal planı bulmaktır.

Sonsuz Ufuklu (Infinite-Horizon Discounted) Performans Kriteri: Bir MDP modelinde ardışık durumlar için belirlenmiş herhangi bir zaman ufku uzunluğu yoksa

ya da zaman ufku uzunluğu sonsuz kabul edilebilecek kadar büyükse, çözüm için sonsuz ufuklu performans kriteri kullanılır. Bu tip problemlerde, $0 < \alpha < 1$ olacak şekilde, önceden belirlenmiş bir α değeri kullanılır. Bu değer, ödüllerin toplamının sonlu olmasını garanti eder. KV'nin, beklenen kalan zaman ufku sayısı, her zaman $1/(1-\alpha)$'dır. Yani, ufka olan beklenen uzaklık hiçbir zaman değişmez. Bu yüzden, eylem stratejileri zamanın bir fonksiyonu değildir. Bundan dolayı, sonsuz ufuklu durumlarda, her zaman optimal sabit bir plan vardır (Littman, 1996).

Değer fonksiyonu, sonsuz ufukta başlangıç durumu s olan bir δ planı için (2.3) denkleminde görüldüğü şekilde hesaplanır (Bernstein, 2005).

$$V^{\delta}(s) = E_{s,\delta}\left[\sum_{t=0}^{\infty} \alpha^{t} R(s_t, a_t)\right] \quad \textbf{(2.3)}$$

$R(s_t, a_t)$, t zaman aralığında, s durumunda, a eyleminin seçilmesi halinde elde edilecek olan anlık ödülü göstermektedir. α değeri ise ödüllerin toplamının sonlu olmasını garantileyen sabittir.

Optimal değer ve optimal plan, bu durum için de, sonlu ufuklu durumda hesaplandığı gibidir.

Başka bir deyişle, sonsuz durumda tanımlanmış, herhangi bir s durumu için, bir δ planının değeri, $V^{\delta}(s)$, hesaplanmak istenildiğinde, bu işlem iteratif olarak (2.4) denkleminde görüldüğü şekildedir.

$$V^{\delta}(s) = R(s,a) + \alpha \sum_{s' \in S} T\left(s' \backslash s, a\right) V^{\delta}(s') \quad \textbf{(2.4)}$$

Denklemdeki $R(s,a)$ anlık ödülü, $T\left(s' \backslash s, a\right)$ s durumunda a eylemi seçilmesi halinde geçilebilecek tüm mümkün durumların olasılıklarını, $V^{\delta}(s')$ ise yeni geçilen s' durumuyla başlayan δ planının değerini göstermektedir.

Elde edilen bu değer fonksiyonları kullanılarak bir plan denklem (2.5)'teki şekilde, buna bağlı olarak, optimal değer fonksiyonu da denklem (2.6)'daki gibi ifade edilir (Bernstein, 2005).

$$\delta(s) = argmax_{a\in A}[R(s,a) + \alpha \sum_{s\prime\in S} T(s'\backslash s,a)V^{\delta}(s')] \quad \textbf{(2.5)}$$

$$V^{*}(s) = max_{a}[R(s,a) + \alpha \sum_{s\prime\in S} T(s'\backslash s,a)V^{\delta}(s')] \quad \textbf{(2.6)}$$

Bu denklem Bellman (1957) tarafından geliştirilmiş olduğundan *"Bellman denklemi"* olarak da bilinir.

2.1.2 Markov karar sürecinin çözüm yöntemleri

MDP'nin çözüm olarak verdiği optimal plan, KV'nin, uzun dönemde, bulunduğu her durumda, seçmesi gereken en iyi alternatifi gösterir. Bir MDP modelinde, bu optimal planların bulunmasının birçok yöntemi vardır. Bu yöntemler içerisinden en çok kullanılan 3 yöntem bu bölümde ayrıntılı olarak açıklanacaktır.

2.1.2.1 Değer iterasyonu

MDP'de amaç, her bir durumda seçilebilecek en iyi eylemleri gösteren, durum-eylem geçişi üretmektir. Eğer her bir durumun değeri biliniyorsa, bu kolaydır. Değer iterasyon (value iteration) algoritması, her biri bir öncekinden elde edilen, bir değer fonksiyonları sırası bularak, bu değer fonksiyonunu hesaplar.

Eğer modelde ufuk uzunluğu sadece 1 ise, yani KV'nin sadece tek bir durum için bir kez karar vermesi gerekiyorsa, bir sonraki adım olmadığından buradaki karara geleceğin herhangi bir etkisi olmayacaktır. Böylece gelecek etkisi göz önüne alınmadan, sadece tüm mümkün eylemlerin anlık ödüllerine bakılarak karar verilebilir.

Fakat modelde ufuk uzunluğu birden fazla ise, seçilebilecek tüm mümkün eylemler değerlendirilirken, eylemin anlık ödülüne, hemen sonrasında seçilecek eylemin değeri de eklenmelidir. Örneğin, ufuk uzunluğu iki ise, birinci adımdaki eylemlerin anlık ödüllerine, ikinci adımdaki, her bir olası durum için, her bir olası eylem göz önüne alınarak hesaplanan, beklenen değer eklenir. Ufuk uzunluğu arttıkça, en alttan başlanarak yukarıya doğru bu işlemler, denklem (2.7)'de de görüldüğü gibi, iteratif olarak sürdürülerek en üst basamaktaki değer hesaplanır. Kısacası burada, DP kullanılır. Bu işlem Bellman denkleminin basit bir iteratif çözümüdür.

$$V^{t+1}(s) = max_a[R(s,a) + \alpha \sum_{s' \in S} T(s'\backslash s, a) V^t(s')] \quad \textbf{(2.7)}$$

Bu denklemin parametreleri, yukarıda yer verilen diğer denklemlerde açıklanmıştır.

Bu değer iterasyonu, limit içerisinde optimale yakınsar. $\|V^{t+1} - V^t\|_\infty$ ifadesi *"Bellman kalanı (Bellman residual) ya da Bellman hata büyüklüğü (Bellman error magnitude) "* olarak adlandırılır. Eğer önceden belirlenmiş bir ε değeri için, $\frac{2\alpha\|V^{t+1}-V^t\|_\infty}{1-\alpha} < \varepsilon$ ise $\|V^* - V^{t+1}\|_\infty \leq \varepsilon$ 'dur. Bu da değer fonksiyonun ε optimali içerisinde olduğunu gösterir. Değer fonksiyonunun optimale yakınsaması, değer fonksiyonunun ait olduğu planın da optimale yakınsaması demek olduğundan optimal plan bulunmuş olur ve algoritma sona erer (Bernstein, 2005). Kısacası, değer iterasyonu algoritması, girdi olarak ε ve başlangıç değeri V^0'ı alır. Çıktı olarak da bir ε-optimal plan verir.

2.1.2.2 Plan iterasyonu

Plan iterasyonu (policy iteration), değer iterasyonundan farklı olarak, bir başlangıç değeri değil, herhangi rastsal bir planı girdi olarak alır. Bu plan üzerinde ilk olarak, değer iterasyonunda yapıldığı gibi, değerler hesaplanarak plan değerlendirmesi yapılır. Fakat plan iterasyonunda bu aşamadan sonra, bir de *"plan geliştirmesi (policy improvement)"* aşaması vardır. Bu aşamada bir DP güncellemesi yapılarak yeni bir plan elde edilir.

Howard (1960)'ın belirttiği üzere, eğer mevcut plan optimal değilse, mutlaka, en az tüm durumlar için mevcut plandaki değere eşit ve bir durum için de, mevcut plandaki değerden daha büyük bir değer elde edilebilir. Plan sayısı sonlu olduğundan bu algoritma da sonlu sayıda iterasyona yakınsar.

Bu aşamadan sonra elde edilen plan, mevcut plan ile karşılaştırılır. Eğer elde edilen plan mevcut plana eşit değilse, tekrar yeni üretilen plan için, plan değerlendirmesi ve geliştirilmesi yapılarak yeni bir plan elde edilir. Fakat elde edilen plan mevcut plana eşit ise, bu, optimal plana ulaşıldığı anlamına gelir ve algoritma sona erer.

Plan iterasyonunda, plan geliştirme aşaması da olduğundan, değer iterasyonuna göre daha fazla işlem gerektirir.

2.1.2.3 Doğrusal programlama

MDP modelinin bir çözümü de, (2.8)'de görüldüğü gibi, bir plan için hesaplanan değerin enbüyüklenmesi amacıyla, bir doğrusal programlama yöntemi kullanılmasıdır (Petrik ve Zilberstein, 2009).

$$min_v \sum_{s \in S} c(s)v(s)$$

$$subject\ to \qquad \textbf{(2.8)}$$

$$v(s) \geq [R(s,a) + \alpha \sum_{s' \in S} T(s'\backslash s, a)V(s')], \quad \forall (s,a) \in (S,A)$$

$$\sum_{s \in S} c(s) = 1$$

Burada ifade edilen $c(s)$ terimi, genellikle düzgün dağılım olarak kabul edilen, durumlar üzerine bir dağılımdır. $v(s)$, s durumu ile başlayan planın değeri olmak üzere, $v(s)c(s)$'nin enküçüklenmesindeki amaç, her bir değer için en küçük üst sınırın seçildiğinden emin olunmak istenmesidir. Bu da aslında bu değeri enbüyüklemek anlamına gelir.

Bu doğrusal program çok büyük olduğundan, yaklaşık olarak yazılabilir. Eğer sol taraf kısıt matrisi için A, sağ taraf için b yazılır ve $v(s) = Mx$ olarak alınırsa, problem (2.9)'daki şekle dönüşür (Petrik ve Zilberstein, 2009).

$$min_x\ c^T Mx$$

$$subject\ to \qquad \textbf{(2.9)}$$

$$AMx \geq b$$

$$Mx \leq \|r\|_\infty/(1-\alpha)\mathbf{1}$$

Buradaki son kısıt $Mx \leq \|r\|_\infty/(1-\alpha)\mathbf{1}$, durumların değerlerinin çok büyük olmamasını garantiler.

2.2 Kısmi Olarak Gözlemlenebilir Markov Karar Süreci

Kısmi olarak gözlemlenebilir Markov karar süreci, (partially observable Markov decision process, POMDP), MDP'nin özel bir şeklidir. Bu modelin MDP'den farkı, KV'nin içinde bulunduğu mevcut durumu tam olarak gözlemleyemiyor olmasıdır. Yani KV, MDP'de olduğu gibi, mevcut durumda seçtiği bir eylemden sonra, geçtiği yeni durumun ne olduğunu tam olarak bilemez. Sadece geçebileceği tüm muhtemel durumların olasılıklarını bilir. Örneğin, 3 durumlu bir modelde, KV, s_1 durumunda iken seçtiği a_1 eyleminden sonra hangi durumda olduğunu kesin olarak bilemez, sadece şu anda bulunduğu durumun s_1, s_2 ve s_3 olma olasılıklarını bilir. İşte bu olasılıkların hepsi KV için *"düşünülen durumlar (belief states)"* kavramını oluşturur. Kısacası, MDP'deki "durum" kavramı yerine POMDP'de "düşünülen durumlar" kavramı kullanılacaktır. POMDP'nin MDP'den bir diğer farkı da, KV'nin, bir eylem seçip diğer bir düşünülen duruma geçtiğinde, bir gözlem elde ediyor olmasıdır. Elde ettiği bu gözlem bir sonraki adımda, KV'nin düşünülen durumlarındaki olasılıkları etkileyecektir. Yani bu gözlem, KV'nin bir sonraki adımda geçeceği durumla ilgili ipuçları verir. Gözlemler olasılıklı olabilir, bir durum için farklı gözlemler elde edilebilir.

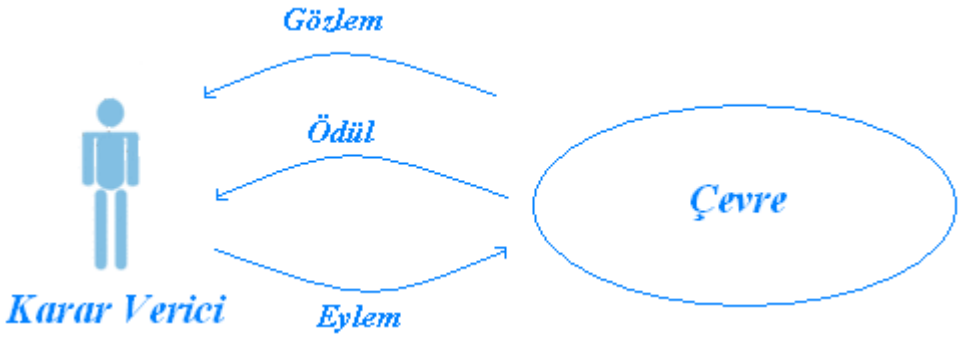
Bir POMDP modeli $< S, A, T, R, \Omega, O >$ şeklinde gösterilir. Bu bileşenler şu şekilde tanımlanır;

- *Durumlar (States):* KV'nin bulunması muhtemel tüm durumlarının oluşturduğu durumlar kümesidir; $\forall s \in S$
- *Eylemler (Actions):* KV'nin seçebileceği mümkün eylemler (alternatifler) kümesidir; $\forall a \in A$
- *Geçiş Fonksiyonu (Transition Function):* KV'nin $\forall s, s' \in S$ ve $\forall a \in A$ olmak üzere, mevcut durumu s ve seçtiği eylem a bilindiğinde, diğer tüm muhtemel durumlara geçiş olasılıklarını veren fonksiyondur; $\mathrm{T: SxA} \rightarrow \Delta \mathrm{S}, \mathrm{T}(\mathrm{s'\backslash s, a})$
- *Gözlemler (Observations):* KV'nin gözlemleyebileceği tüm muhtemel gözlemlerinin oluşturduğu gözlemler kümesidir; $\forall o \in \Omega$

- *Gözlem Fonksiyonu (Observation Function):* KV'nin $\forall s' \in S$ ve $\forall a \in A$ olmak üzere, mevcut durumundan a eylemi seçerek geçtiği s' durumunda, her bir o gözleminin elde edilme olasılığını veren fonksiyondur;

  $O: SxA \rightarrow \Delta\Omega, O(o\backslash s', a)$

- *Anlık Ödüller (Immediate Rewards):* KV'nin$\forall s \in S$ ve $\forall a \in A$ olmak üzere, s durumunda a eylemini seçmesi halinde elde edeceği kazancı veren fonksiyondur; $R: SxA \rightarrow \Re$, $R(s, a)$



Şekil 2.4 : Bir POMDP modelinde karar vericinin çevreyle olan etkileşimi

Şekil 2.4'te görüldüğü gibi, KV, her bir adımda, bir önceki adımda elde ettiği gözlemi ve seçtiği eylemi kullanarak bir düşünülen durum elde eder. Çıktı olarak ise, yeni seçilecek eylemi belirler. KV'nin seçmiş olduğu eylem, nümerik bir ödül değeri ve bir sonraki durum hakkında ipuçları taşıyan yeni bir gözlem verir.

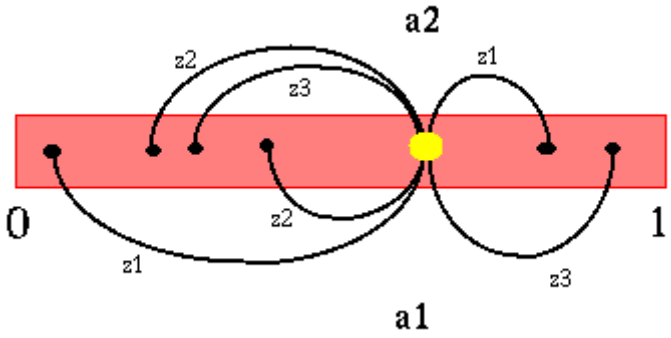
2.2.1 Düşünülen durum güncellenmesi

Bir POMDP modelinde, KV, mevcut düşünülen durumunda elde ettiği gözlem ve seçtiği eyleme göre yeni bir düşünülen duruma geçer. Bunun için, her bir adımda mevcut düşünülen durum b'den geçilen yeni düşünülen durum b', basit bir Bayes kuralı uygulaması olan durum tahmin fonksiyonu kullanılarak (2.10)'daki gibi güncellenir.

$$b'(s') = P(s'o, a, b) = \frac{O(o\backslash s', a)\sum_{s\in S} T(s'\backslash s, a)b(s)}{\sum_{s'\in S} O(o\backslash s', a)\sum_{s\in S} T(s'\backslash s, a)b(s)} \quad \textbf{(2.10)}$$

Buradaki $O(o\backslash s', a)$ ifadesi, gözlem fonksiyonunu, $T(s'\backslash s, a)$ ifadesi geçiş fonksiyonunu, $b(s)$ ifadesi ise, mevcut düşünülen durumu göstermektedir.

Şekil 2.5, iki durumlu, iki eylemli ve üç gözlemli bir POMDP modeli için, mevcut düşünülen durumdan, bir eylem seçildiğinde ve bir gözlem elde edildiğinde geçilen yeni düşünülen durumları, tüm eylem ve gözlemleri göz önünde bulundurarak göstermektedir.



Şekil 2.5 : Mevcut düşünülen durumdan, tüm mümkün yeni düşünülen durumlara geçiş (Cassandra, 2010)

Ayrıca gözlem fonksiyonu, düşünülen durumlar göz önüne alındığında (2.11)'deki gibi yazılabilir.

$$P(o\backslash b,a)=\sum_{s\prime\in S}[O(o\backslash s',a)\sum_{s\in S}T\left(s'\backslash s,a\right)b(s)] \quad \textbf{(2.11)}$$

Denklemin bileşenleri, denklem (2.10) için açıklanmıştır.

2.2.2 Düşünülen durumlu Markov karar süreci

Astrom (1965) tarafından öne sürülen bu düşünceye göre, bir POMDP modelini, *"düşünülen durumlu MDP (belief state MDP)"* olarak tanımlamak mümkündür. Bir düşünülen durumlu MDP, $<\pi,A,T,R>$ şeklinde tanımlanır. Bu bileşenler şu şekilde açıklanabilir;

- π, S kümesi üzerine dağılımların oluşturduğu kümedir.
- A, KV'nin seçebileceği tüm muhtemel eylemlerin oluşturduğu kümedir.
- *Geçiş Fonksiyonu:* Bu fonksiyon (2.12)'de görüldüğü gibidir (Bernstein *ve diğ.* 2009).

$$T(b'\backslash b,a)=\sum_{o\in\Omega}P\left(b'\backslash b,a,o\right)P(o\backslash b,a) \quad \textbf{(2.12)}$$

Burada, $P(b'\backslash b, a, o)$ ifadesi, b düşünülen durumunda a eylemi alınırsa ve o gözlemi elde edilirse b' düşünülen durumuna geçme olasılığıdır. $P(o\backslash b, a)$ ifadesi ise, b düşünülen durumunda, a eylemi alındığında o gözlemini elde etme olasılığıdır. Sonuç olarak bulunacak $T(b'\backslash b, a)$, b düşünülen durumunda a eylemi alındığında, b' düşünülen durumuna geçiş olasılığını verecektir.

- *Anlık Ödüller:* Ödüller (2.13)'te görüldüğü şekilde hesaplanır (Bernstein *ve diğ.* 2009).

$$R(b, a) = \sum_{s \in S} b(s) R(s, a) \qquad \textbf{(2.13)}$$

Burada, $R(s, a)$, s durumunda a eylemi seçildiğinde elde edilecek anlık ödülü, $b(s)$, s durumunun düşünülen durum olasılığını göstermek üzere, $R(b, a)$, b düşünülen durumunda a eylemi seçildiğinde elde edilecek anlık ödülü göstermektedir.

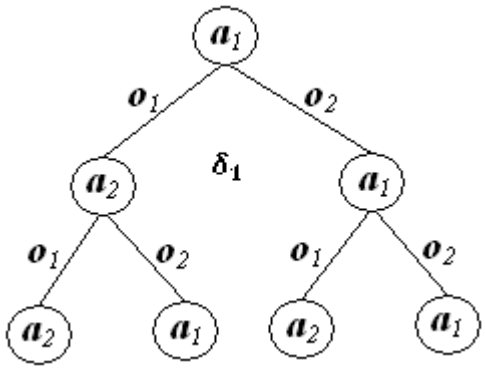
Bu, aslında bir MDP modeli olduğundan, MDP gibi çözülebilse de, düşünülen durumlu MDP, sürekli ve |S| boyutlu durum uzayına sahip olduğundan, klasik MDP yöntemleri, bu model için uygulanabilir değildir. Fakat DP kullanarak çözülebilir (Bernstein *ve diğ.* 2009).

Bir POMDP modeli, MDP'de olduğu gibi çıktı olarak bir plan verir. Burada plan, elde edilen gözlemlerden eylemlere bir fonksiyon olarak, $\Omega^* \rightarrow A$ şeklinde gösterilebilir.

POMDP de MDP'de olduğu gibi sonlu ufuklu ya da sonsuz ufuklu olarak ele alınabilir;

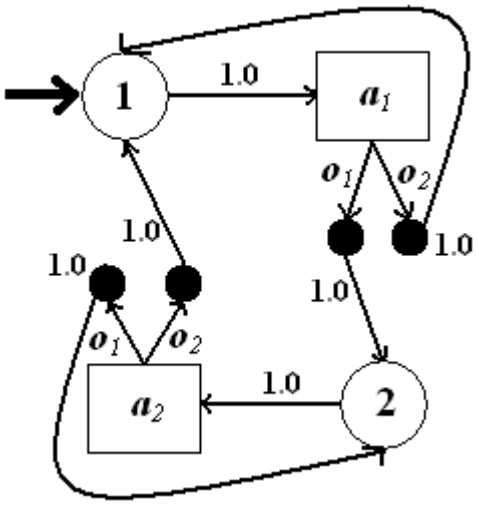
Sonlu Ufuklu Problemler (Finite-Horizon Problems): POMDP'de sonlu ufuklu problemlerde, ardışık durumlar için belirlenmiş bir zaman ufku uzunluğu vardır. Yani böyle bir durumda, KV'nin, önceden belirlenmiş sayıdaki zaman aralıklarında karar vereceği göz önüne alınacaktır. KV'nin amacı, sonlu ufuk boyunca beklenen toplam ödülü enbüyüklemektir. Çözüm, Şekil 2.6'te görüldüğü gibi, düğümlerin (nodes) eylemleri, çizgilerin (edges) de gözlemleri gösterdiği bir *"plan ağacı (policy tree)"* olarak gösterilebilir. Şekildeki plan ağacında, birinci adımda a_1 eylemi seçildikten sonra, ikinci adımda o_1 gözlemi elde edilirse a_2, o_2 gözlemi elde edilirse

a_1 eylemi seçilecektir. o_1 gözlemi elde edildikten sonra seçilen a_2 eylemi tekrar bir gözlem elde etmemizi sağlayacaktır. Bu gözlem eğer o_1 gözlemi ise 3. adımda a_2 eylemi, o_2 gözlemi ise a_1 eylemi seçilecektir. Fakat o_2 gözlemi elde edildikten sonra seçilen $a_1$eylemini kullanırsak, bundan sonraki adımda sağlayacağı gözlem o_1 ise a_2, o_2 ise a_1 eylemi seçilecektir. Kısacası bu plan, tüm zaman ufku boyunca, elde edilen gözlemlerden yola çıkarak hangi eylemin seçilmesi gerektiğini gösterir.



Şekil 2.6 : Örnek bir plan ağacı (Bernstein *ve diğ.* 2009)

Sonsuz Ufuklu Problemler (Infinite-Horizon Problems): POMDP'de sonsuz ufuklu problemlerde, ardışık durumlar için belirlenmiş herhangi bir zaman ufku uzunluğu yoktur ya da zaman ufku uzunluğu sonsuz kabul edilebilecek kadar büyüktür. KV'nin amacı, sonsuz ufuk boyunca beklenen toplam ödülü enbüyüklemektir. Çözüm, Şekil 2.7'da görüldüğü gibi, düğümlerin eylemleri, geçişlerin de gözlemleri gösterdiği bir *"sonlu durum denetçisi (finite state controller)"* olarak gösterilebilir. Şekil 2.7, Şekil 2.6'nın sonlu durum denetçisi şekline dönüştürülmüş halidir. KV, a_1 eylemini seçtikten sonra, o_1 gözlemini elde ederse 2. düğüme geçecektir, bu da a_2 eylemini seçeceği anlamına gelir. Eğer o_2 gözlemini elde ederse, bu sefer de 1. düğüme geri dönerek tekrar a_1 eylemini seçecektir. Durum her zaman buradaki gibi deterministik olmak zorunda değildir. 1.0 yerine değişik olasılıklar da olabilir.



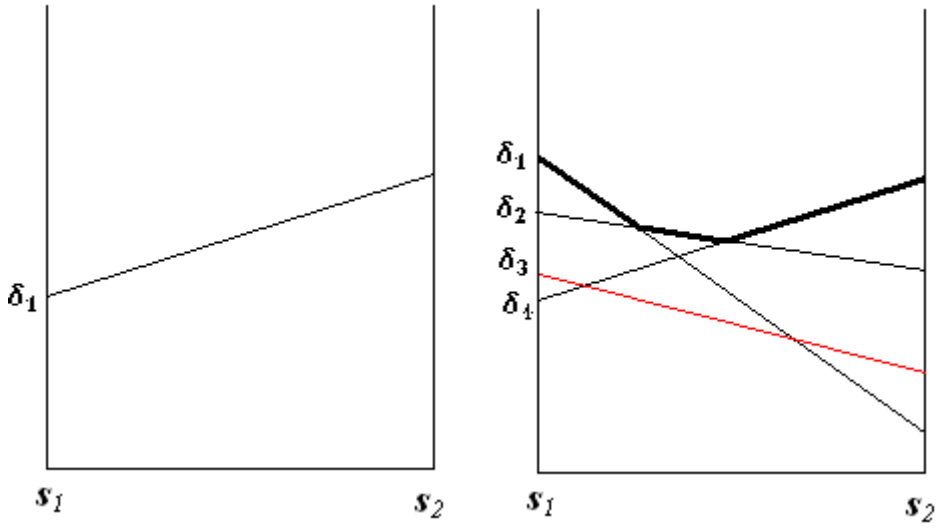
Şekil 2.7 : Örnek bir sonlu durum denetçisi (Bernstein *ve diğ.* 2009)

2.2.3 Kısmi olarak gözlemlenebilir Markov karar sürecinin çözüm yöntemleri

Bir POMDP modelinin çözümünün bulunmasında, MDP'de olduğu gibi birçok yöntem vardır. Bu yöntemler içerisinden sıklıkla kullanılan birkaç yöntem bu bölümde ayrıntılı olarak açıklanacaktır.

2.2.3.1 Değer iterasyonu

POMDP değer iterasyonunda en önemli nokta, hesaplanan değer fonksiyonlarının her zaman parçalı, doğrusal ve konveks olmasıdır. Smallwood ve Sondik (1973), Bellman operatörü kullanarak, sonlu ufuklu problemler için, her adımda değer fonksiyonunun, parçalı, doğrusal ve konveks olma özelliğini koruduğunu ispatlamışlardır. Sonsuz ufuklu problemler için ise, konveks olup parçalı ve doğrusal olmayabileceğini göstermişlerdir. Yine de bu, parçalı, doğrusal konveks bir fonksiyona yaklaştırılabilir. Şekil 2.8(a)'da iki durumlu bir POMDP için, Şekil 2.6'daki gibi bir δ_1 planın değerini ifade eden, bir doğrusal değer fonksiyonu, Şekil 2.8(b)'de ise, bir problem için tüm planlar göz önüne alınarak oluşturulmuş, parçalı, doğrusal konveks değer fonksiyonu gösterilmiştir.



Şekil 2.8 : (a) İki durumlu bir POMDP için bir doğrusal değer fonksiyonu **(b)** İki durumlu bir POMDP için parçalı, doğrusal, konveks, değer fonksiyonu

Şekil 2.8(b)' deki kırmızı ile çizilmiş olan, δ_3 planının değerini gösteren vektör, parçalı, doğrusal, konveks fonksiyonun bir parçası değildir. Çünkü diğer tüm vektörler tarafından bastırılmıştır. Bu tip baskın olmayan vektörlerin göz önüne alınmasına gerek yoktur. Bu tip vektörlerin elenmesinde *"budama (pruning)"* yöntemi kullanılır.

Bu yöntemde, vektörlerin baskın olup olmadığını anlamak için (2.14)'teki gibi bir doğrusal programlama modeli kullanılır (Bernstein *ve diğ.* 2009).

$Max\ \varepsilon$

$subject\ to$

$$\forall \hat{t} \quad \sum_s b(s)\hat{t}(s) + \varepsilon \leq \sum_s b(s)t(s) \qquad \textbf{(2.14)}$$

$$\sum_s b(s) = 1 \ \forall s \quad b(s) \geq 0$$

Burada, $\hat{t}(s)$ ifadesi $t(s)$'ye eşit olmayan bir başka değer vektörü anlamına gelir. Eğer bir ε değerini pozitif yapan değişkenler bulunabiliyorsa, o halde bazı düşünülen durumlar için, değer fonksiyonuna, t durumunu eklemek gereklidir. Eğer değilse, t bastırılmış demektir ve göz önüne almaya gerek yoktur.

Cassandra *ve diğ.* (1997)'nin geliştirdiği bir yaklaşımda ise, ilk olarak tüm vektörler oluşturulup sonra budama işlemi gerçekleştirilmez, vektörler oluşturulurken aynı zamanda budanır.

POMDP'de her bir adımda, (2.15)'te görüldüğü gibi, her bir düşünülen durum için optimal eylem bulunurken, geçiş ve gözlem olasılıkları göz önüne alınarak hesaplanan değere, seçilecek olan eylemin anlık ödülü eklenir ve bu değeri enbüyükleyen eylem seçilir (Bernstein *ve diğ.* 2009).

$$a = argmax_{a\in A}[R(b,a) + \alpha \sum_{o\in\Omega} P(o\backslash b,a)V^t(T(b\backslash a,o))] \quad \textbf{(2.15)}$$

Burada, $R(b,a)$ ifadesi, düşünülen durum için anlık ödülü, $P(o\backslash b,a)$, b düşünülen durumunda, a eylemi alındığında, o gözlemini elde etme olasılığını, $T(b\backslash a,o)$ ifadesi ise, a eylemi alındığında ve sonrasında o gözlemi elde edildiğinde, başlangıçta b düşünülen durumunda olma olasılığını göstermektedir.

Bu hesaplamalarda incelenen her bir stratejinin mutlaka en iyi olduğu düşünülen durumlar vardır. İşte bu durumların bulunması için bazı algoritmalar geliştirilmiştir. Bu algoritmaların başlıcaları; Smallwood ve Sondik (1973)'in geliştirdiği *"Tek Geçiş (One Pass)"* algoritması, Cheng (1988)'in geliştirdiği *"Doğrusal Destek (Linear Support)"* algoritması, Cassandra *ve diğ.* (1994)'nin *"Şahitlik (Witness)"* algoritması olarak sayılabilir. Yukarıda bahsedilen Cassandra *ve diğ.* (1997)'nin budama yöntemi de bu algoritmalar arasında sayılabilir.

POMDP'de değer iterasyonunda, rastsal bir parçalı, doğrusal, konveks fonksiyonla başlanarak DP uygulanır. MDP'de olduğu gibi, bir düşünülen durum için optimal plan (2.16)'da olduğu gibi hesaplanabilir (Bernstein *ve diğ.* 2009). Burada, ele alınan düşünülen durum için hesaplanan değeri enbüyükleyen plan, o düşünülen durum için optimal plandır.

$$\delta(b) = argmax_{a\in A}[\sum_{s\in S} R(s,a)b(s) + \alpha \sum_{o\in\Omega} P(o\backslash b,a)V\big(\tau(b,o,a)\big) \quad \textbf{(2.16)}$$

$\tau(b,o,a)$ ifadesi, b düşünülen durumunda, a eylemi seçilip geçilen ve burada o gözleminin elde edildiği, düşünülen durumdur. Diğer terimler, daha önceki denklemler için açıklanmıştır.

2.2.3.2 Sezgisel aramalı değer iterasyonu

Sezgisel aramalı değer iterasyonu (heuristic search value iteration, HSVI), Smith ve Simmons (2004) tarafından geliştirilmiştir. Bu tezde kullanılan POMDP çözücü bu yöntemden yararlanılarak oluşturulmuştur. Yani, önerilen model örneklerinin çözümünde HSVI kullanılmıştır. Aşağıda HSVI yönteminin açıklanmasında Smith ve Simmons (2004)'un yazmış olduğu makaleden yararlanılmıştır.

Geleneksel değer iterasyonunda, beklenen uzun dönem ödülü bir δ planı için (2.17)'de gösterildiği şekilde hesaplanmaktadır.

$$V^{\delta}(b) = E[\sum_{t=0}^{\infty} \alpha^t R(s_t, a_t) \backslash b, \delta] \quad \textbf{(2.17)}$$

Elde edilecek optimal planın bu değeri enbüyükleyen plan olduğundan daha önce bahsetmiştik. Başlangıç durumu için optimal plan hesaplanması (2.18)'de gösterilmiştir.

$$\delta^* = argmax_{\delta} V^{\delta}(b_0) \quad \textbf{(2.18)}$$

Yaklaşık POMDP planlama problemlerinde genellikle amaç, ele alınan planın değeri, yani beklenen ortalama uzun dönem ödülü ile optimal planın değeri arasındaki farkın enküçüklenmesidir.

Pişmanlık (regret) değeri adı verilen bu farkın hesaplanması, (2.19)'da gösterilmiştir.

$$regret(\delta, b) = V^{\delta^*}(b) - V^{\delta}(b) \quad \textbf{(2.19)}$$

Burada $V^{\delta^*}(b)$ ifadesi optimal planın değerini, yani beklenen ortalama uzun dönem ödülünü göstermektedir.

HSVI, sezgisel aramayı parçalı, doğrusal, konveks değer fonksiyonu gösterimi ile birleştiren bir yaklaşık POMDP çözüm algoritmasıdır. Sezgisellikten kasıt, modelin çözümü kolaylaştırılırken optimallikten fedakarlık yapılmış olmasıdır. Yani sezgisel algoritmalar, optimal çözümler vermek yerine, optimale yakınsayan çözümler sunmaktadır.

HSVI, geleneksel değer iterasyonu algoritmalarındaki gibi değer üzerinde değil, değerin alt ve üst sınırlarında güncellemeler yapar. Alt ve üst sınır fonksiyonları

sırasıyla $V\_$ ve $\bar{V}$ şeklinde gösterilir. Değer aralığı ve bu aralığın uzunluk fonksiyonu (2.20) ve (2.21)'de gösterildiği şekildedir.

$$\hat{V}(b) = [V\_(b), \bar{V}(b)] \quad \textbf{(2.20)}$$

$$width\left(\hat{V}(b)\right) = \bar{V}(b) - V\_(b) \quad \textbf{(2.21)}$$

Birçok değer iterasyonu algoritması alt sınır değerini korumakta ve güncellemektedir. Bunun için de genel olarak "*vektör kümesi*" gösterimi kullanılmaktadır. Farklı planlar için ödül alt sınır değerlerini ifade eden $\propto$ vektörlerinin oluşturduğu $\Gamma_{v\_}$ sonlu kümesi, bu vektör kümesini ifade etmek üzere, belirli bir düşünülen durumdaki alt sınır vektörü (2.22)'deki gibi ifade edilir.

$$V\_(b) = max_{\propto \in \Gamma_{v\_}}(\propto . b) \quad \textbf{(2.22)}$$

Sonlu ufuklu POMDP problemlerinde, sonlu bir vektör kümesi $\Gamma_{v\_}$ ile (2.22)'den tam olarak bir optimal vektör elde edilir. Sonsuz ufuklu problemlerde ise, bu elde edilen vektör optimale yakınsar. Değer fonksiyonunu alt sınır olarak ele almak, vektör kümesine yeni bir vektör ekleyerek, bu vektör kümesinde güncellemeler yapılmasına olanak sağladığından yerel güncelleştirmeyi kolaylaştırır.

Fakat üst sınır için böyle bir vektör kümesi ele almak ve güncelleştirmelerde bu kümeye yeni vektörler eklemek, yerel güncelleştirmenin komşuluğundaki sınırların geliştirilmesinde istenilen etkiyi sağlamamaktadır.

Bu yüzden, üst sınır, bir "*nokta kümesi*" şeklinde ifade edilmiştir. Bu $\gamma_{\bar{V}}$ nokta kümesi, her b düşünülen durumuna karşılık gelen üst sınırları gösteren $(b_i, \bar{v}_i)$ noktalarından oluşmaktadır. Yerel güncellemeler sırasında bu nokta kümesine yeni noktalar eklenerek güncelleme yapılır. Bu üst sınır noktaları konveks bir kabuk oluşturmaktadır. Bu konveks kabuk, bir doğrusal programlama kullanılarak çözülebilir.

HSVI algoritmaları kısaca şu şekilde özetlenebilir;

Algoritma 1: $\delta = HSVI(\varepsilon)$

HSVI, $regret(\delta, b_0) \leq \varepsilon$ olacak şekilde planlar ele alır.

1. İlk sınırlar, yani $\hat{V}$ oluşturulur.
2. $width\left(\hat{V}(b_0)\right) \leq \varepsilon$ olduğu sürece $explore(b_0, \varepsilon, 0)$ algoritması çağırılır.
3. İstenilen kesinlik değerine ulaşıldığında, alt sınır değeri ile ifade edilen mevcut plan δ, optimale yakınsayan plandır.

Algoritma 2: $explore(b, \varepsilon, t)$

Bu algoritma tekrarlı olarak, sınır aralığı uzunluğuna bağlı sonlandırma şartları sağlanana kadar arama ağacında tek bir yolu takip eder. En son olarak da vardığı noktadan geriye doğru, başlangıç düşünülen durumuna kadar bir dizi güncelleştirme yapar.

1. Eğer $width\left(\hat{V}(b)\right) \leq \varepsilon\alpha^{-t}$ ise bu algoritma elde edilen değeri çağrıldığı yere geri gönderecektir.
2. Daha sonra bahsedilecek olan ileri keşif sezgiselliği yöntemine göre bir a^* eylemi ve o^* gözlemi seçilir.
3. $explore(\tau(b, a^*, o^*), \varepsilon, t + 1)$ fonksiyonu çağrılır.
4. b düşünülen durumunda $\hat{V}$ sınırları yerel olarak güncellenir.

Algoritma 1'de ilk sınırlar $\hat{V}$ oluşturulurken, tüm düşünülen durumlar için $V_{0\_} \leq V^* \leq \overline{V_0}$ şartı göz önüne alınmalıdır. Bu ilk sınırlar oluşturulurken ilk olarak bir a eylemi için δ_a planı ele alınır. Bu plan, her bir karar verme aşamasında sadece a eyleminin seçilmesinden oluşan plandır. Bu planda, her bir adımda, en düşük ödülü veren durumda (en kötü durum) olunduğu varsayılarak bu eylem için beklenen ortalama uzun dönem ödülünün alt sınırı $R_{a\_}$ hesaplanır. Bu işlem diğer tüm eylemler için tekrarlanır. Tüm eylemlerin alt sınır ödülleri arasından da en büyük olan, algoritmada kullanılacak ilk ortalama uzun dönem ödülü alt sınırı R_'dir.
Başlangıç alt sınırı $V_{0\_}$, tüm durumlar için $\propto(s) = R\_$ olacak şekilde tek bir $\propto$ vektörü içermektedir.

İlk üst sınırın belirlenmesi için de modelde tam olarak gözlemlenebilirlik olduğu varsayılır ve problem MDP olarak çözülür. Bu çözüm, başlangıç nokta kümesini oluşturan üst sınır değerlerini vermektedir. Kullanılan ilk sınır değerine V_{MDP} adı verilir.

Yerel güncellemeler

Değer iterasyonunda kullanılan Bellman güncellemesi daha önce (2.16)'da ifade edilmiştir. Bunun daha açık bir şekilde ifadesi (2.23) ve (2.24)'te gösterilmektedir.

$$Q^V(b,a) = \sum_s R(s,a)b(s) + \alpha \sum_o P(o\backslash b,a)V(\tau(b,a,o)) \quad \textbf{(2.23)}$$

$$HV(b) = max_a Q^V(b,a) \quad \textbf{(2.24)}$$

$\tau(b,o,a)$ ifadesi, b düşünülen durumunda, a eylemi seçilip geçilen ve burada o gözleminin elde edildiği, düşünülen durumdur. $Q^V(b,a)$ ise b düşünülen durumunda a eyleminin seçilmesi halinde elde edilecek olan beklenen ödül değeridir. Geleneksel değer iterasyonu, tüm düşünülen durumlar için bu güncellemeleri yapar. Fakat HSVI yerel güncelleme operatörleri kullanır. Bu operatörler alt sınır için L_b ve üst sınır için U_b olmak üzere iki tanedir. b düşünülen durumunda yerel güncelleştirmenin yapılması, bu iki operatörün her ikisinin de o düşünülen duruma uygulanması anlamına gelmektedir.

Alt sınır vektörünü güncellerken, eldeki vektör kümesine yeni bir vektör eklenir. Üst sınır için de aynı şekilde eldeki nokta kümesine yeni bir nokta eklenmektedir. Alt ve üst sınırlar için yapılan bu güncellemeler (2.25) ve (2.26)'da gösterilmektedir.

$$\Gamma_{L_b V\_} = \Gamma_{V\_} \cup backup(V\_, b) \quad \textbf{(2.25)}$$

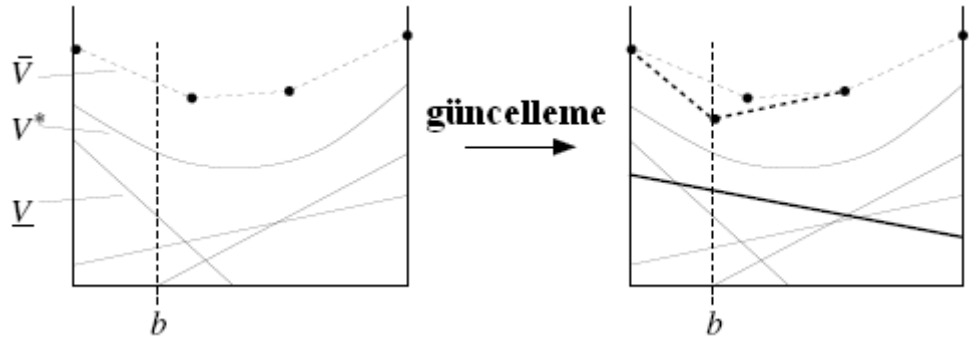
$$\gamma_{U_b \overline{V}} = \gamma_{\overline{V}} \cup (b, H\overline{V}(\mathrm{b})) \quad \textbf{(2.26)}$$

(2.25)'te ifade edilen $backup(V\_, b)$ algoritması aşağıdaki gibi özetlenebilir;

Algoritma: $backup(V\_, b)$

Backup fonksiyonu Bellman güncelleştirmesinin genelleştirilmiş şekli olarak düşünülebilir.

1. $\beta_{a,o} \leftarrow argmax_{\propto \in \Gamma_V}(\propto.\tau(b,a,o))$, yani bir b düşünülen durumunda iken, bir alt adımda a eylemi ve o gözlemi seçildiğinde $(\propto.\tau(b,a,o))$ değerini enbüyükleyen vektör, yeni geçilen düşünülen durumun alt sınır vektörüdür.
2. $\beta_a(s) \leftarrow R(s,a) + \alpha \sum_{o,s'} \beta_{a,o}(s')O(s',a,o)T(s,a,s')$, yani belli bir s durumunda iken bir alt adımda a eylemi seçildiğinde, bu durum için elde edilecek ortalama beklenen alt sınır değeri.
3. $\beta \leftarrow argmax_{\beta_a}(\beta_a.b)$, yani bir b düşünülen durumunda, tüm durumlar için elde edilen alt sınır değerlerinden oluşan vektörlerden, $(\beta_a.b)$ değerini enbüyükleyen vektör, bu b düşünülen durumu için alt sınır vektörüdür.



Şekil 2.9 : b düşünülen durumunda yapılan yerel güncelleme (Smith ve Simmons, 2004)

Şekil 2.9'da alt ve üst sınırların ne şekilde gösterildiği görülmektedir. Sol taraftaki şekilde, noktalar ve kesikli çizgiyle ifade edilen konveks kabuk üst sınırı ifade etmektedir. Düz çizgiyle ifade edilen vektörler ise $\Gamma_{V\_}$'nin elemanı olan vektörlerdir. Güncelleme yapıldıktan sonra, sağ taraftaki şekilden de görülebileceği gibi $\Gamma_{V\_}$'ye yeni bir vektör, $\gamma_{\bar{V}}$'ye ise yeni bir nokta eklenmiştir. Eklenen bu nokta ve bu vektör, b düşünülen durumunda sınırları optimale yani V^*'a yaklaştırmıştır.

HSVI, alt sınır vektör kümesi ve üst sınır nokta kümesi içerisinden periyodik olarak baskın olmayan elemanları budar. Budama işlemi, her bir kümenin eleman sayısındaki artış son budama işleminden sonra %10'a ulaştığında yapılır.

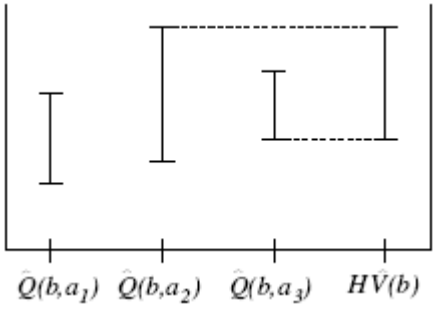
İleri keşif sezgiselliği (forward exploration heuristics)

Bu yöntem, HSVI algoritmasında bir ileri adım araştırılırken hangi eylem ve hangi gözlemin seçileceğine ve sonuç olarak hangi alt düğüme ulaşılacağına karar verir.

HSVI'da başlangıç düşünülen durumdaki belirsizlik, sınırların aralık uzunluğunun ortalaması olarak ifade edilmiştir. Buradaki amaç, bu belirsizliği azaltmak, yani bu sınırlar arasındaki mesafeyi enküçüklemektir.

Bir alt adıma geçildiğinde ilk olarak seçilecek olan eylem düşünülür. Burada aralık fonksiyonu $\hat{Q}(b,a)$, (2.27)'deki gibi ifade edilir.

$$\hat{Q}(b,a) = [Q^{V}\text{-}(b,a), Q^{\overline{V}}(b,a)] \quad \textbf{(2.27)}$$



Şekil 2.10 : $\hat{Q}(b,a_i)$ ve $H\hat{V}(b)$ arasındaki ilişki (Smith ve Simmons, 2004)

Şekil 2.10'de, bir Bellman güncellemesinden sonra, b düşünülen durumunda, her bir mümkün eylem için $\hat{Q}(b,a)$ sınırları ve $H\hat{V}(b)$ sınırları arasındaki ilişki gösterilmiştir. Şekilde görüldüğü gibi $H\hat{V}(b)$ aralığı, biri en büyük üst sınıra, diğeri de en büyük alt sınıra sahip olan iki $\hat{Q}(b,a)$ aralığı kullanılarak belirlenmiştir. Bu ilişki burada kullanılan iki eylemden birinin seçilmesi gerektiğini söylemektedir. Bu iki eylemden ise üst sınırı büyük olan eylem seçilerek yakınsama garanti altına alınır. Bu durum (2.28)'de gösterilmiştir.

$$a^{*} = argmax_{a} Q^{\overline{V}}(b,a) \quad \textbf{(2.28)}$$

Eylem seçildikten sonra bir gözlem belirlenmesi gerekmektedir.

Bulunulan düşünülen durumda seçilen a^* eylemi için değer aralığının uzunluğu (2.29)'daki gibi ifade edilir.

$$width\left(\hat{Q}(b,a^*)\right) = \alpha \sum_o P(o\backslash b,a^*) width(\hat{V}\left(\tau(b,a^*,o)\right)) \quad \textbf{(2.29)}$$

Bu ifade, explore fonksiyonunun sonlandırma kriteri $width\left(\hat{V}(b_0)\right) \leq \varepsilon\alpha^{-t}$'yi açıklamaktadır. Çünkü bir güncellemeden sonra bir b düşünülen durumundaki belirsizlik, en fazla, alt düğümlerin ağırlıklandırılmış ortalamasının α katı kadardır. Daha alt düğümlere inildikçe belirsizlikteki gereksinimler azalmaktadır. Buna göre fazla belirsizlik (2.30)'da görüldüğü gibi ifade edilir.

$$excess(b,t) = width\left(\hat{V}(b)\right) - \varepsilon\alpha^{-t} \quad \textbf{(2.30)}$$

Bir beklenen durum üzerinde, bu ifade negatif oluyorsa, o halde sonlandırma kriteri sağlanmıştır ve bu beklenen durum üzerindeki arama sonlandırılır. Bir b düşünülen durumundaki belirsizlik en fazla bu düğümün alt düğümlerindeki belirsizliklerin ağırlıklandırılmış toplamıdır. Bu durum, (2.31)'de ifade edilmiştir.

$$excess(b,t) \leq argmax_o[\sum_o P(o\backslash b,a^*) excess(\tau(b,a^*,o),t+1)] \quad \textbf{(2.31)}$$

Böylece, gözlem seçilirken (2.31) ifadesine en fazla katkıda bulunan gözlem seçilecektir. (2.32) bu durumu açıklamaktadır.

$$o^* = argmax_o[P(o\backslash b,a^*) excess(\tau(b,a^*,o),t+1)] \quad \textbf{(2.32)}$$

Pratikte her zaman, bu algoritmada kullanılan ε değeri tam olarak bilinmeyebilir. Bu yüzden, bu değerin en uygun bir şekilde belirlenmesi gerekmektedir. Bunun için de HSVI'nin kullanıldığı bir algoritma geliştirilmiştir. Bu algoritmada HSVI, sabit bir ε değeri kullanmaz, bunun yerine "explore" fonksiyonunun çağırıldığı her bir aşamada bu değeri, başlangıç düşünülen durumundaki mevcut belirsizlikten küçük olacak şekilde düzelterek kullanır.

Bu tezde kullanılan çözücü, HSVI algoritmasını bu şekilde ele almaktadır.

2.2.3.3 Plan iterasyonu

POMDP'deki plan iterasyonu, MDP'ye göre daha karmaşıktır ve elde edilmesi zordur. Bernstein (2005), bu planın elde edilmesine yönelik bir yöntem geliştirmiştir.

Bu yöntemde her bir plan, bir sonlu durum denetçisi şeklinde ele alınmıştır. Sonlu durum denetçisinde KV, sonlu sayıda iç duruma sahiptir. Eylemler, sadece mevcut iç duruma ve gözlemler elde edildiği andaki diğer iç durumlara geçişe bağlıdır. Şekil 2.7'de bir sonlu durum denetçisi örneği görülmektedir.

Bir sonlu durum denetçisi $< Q, \Omega, A, \psi, \eta >$ şeklinde ifade edilir (Bernstein, 2005). Bu bileşenler;

- Q, denetçi düğümlerinin sonlu kümesidir.
- Ω, POMDP'nin gözlemleri olarak ele alınan girdilerdir.
- A, POMDP'nin eylemleri olarak ele alınan çıktılardır.
- $\psi: Q \rightarrow \Delta A$, her bir düğümde seçilen eylemlerin dağılımını gösteren eylem seçme fonksiyonudur.
- $\eta: QxAx\Omega \rightarrow \Delta Q$, her bir başlangıç düğümü ve bu düğümde seçilen eylemin dağılımını gösteren geçiş fonksiyonudur.

Denetçinin her bir durum ve başlangıç düğümü için, sonsuz ufukta tanımlanmış bir beklenen toplam ödülü vardır. Bu ödüller göz önüne alınarak, bir q düğümündeki bir s durumunun değeri $V(s,q)$, (2.33)'teki gibi hesaplanır (Bernstein, 2005).

$$V(s,q) = \sum_{a} P(a\backslash q)[R(s,a) + \alpha \sum_{s',o,q'} P\left(o, s'\backslash s, a\right) P\left(q'\backslash q, a, o\right) V(s', q')] \qquad \textbf{(2.33)}$$

$P(a\backslash q)$, q düğümünde a eyleminin seçilmesi olasılığı, $P\left(q'\backslash q, a, o\right)$, q düğümünde a eylemi seçilip o gözlemi elde edildikten sonra q' düğümüne geçme olasılığı, $P\left(o, s'\backslash s, a\right)$ ise, s durumunda a eylemi seçildiğinde, s' durumuna geçip o gözlemini elde etme olasılığıdır. Daha önceden de açıklandığı gibi, $R(s,a)$ anlık ödülü ifade etmektedir. $V(s', q')$ ifadesi ise, bir sonraki düğüm q' de, geçilen s' durumunun değerini göstermektedir.

Denetçinin değeri, b düşünülen durumu için (2.34)'te olduğu gibi ifade edilir.

$$V(b) = max_q \sum_s b(s)V(s,q) \quad \textbf{(2.34)}$$

$b(s)$, s durumunun düşünülen durumu, $V(s,q)$ ise, q düğümündeki s durumunun değerini göstermektedir.

MDP'deki plan iterasyonunda, rastsal bir plan ele alınarak başlandığı gibi, POMDP'de de rastsal bir sonlu durum denetçisi ele alınır.

İlk adım olarak, bu sonlu durum denetçisinin değerlendirmesi yapılır. Her bir durum denetçisi, düşünülen durum uzayı için bir doğrusal değer fonksiyonuna sahiptir. Her bir düşünülen durum için en iyi düğüm seçilerek bir parçalı, doğrusal, konveks fonksiyon elde edilir.

İkinci adımda, DP güncellemesi yapılır. Denetçiye, yeni deterministik düğümler eklenir.

Son adımda, bazı ek işlemler uygulanır. Bu ek işlemler;

- *Denetçi Azaltma (Controller Reductions):* Bu işlemde, budama işleminde olduğu gibi bir doğrusal programlama modeli kullanılarak baskın olmayan düğümler, denetçi içerisinden elenir. Burada dual bir doğrusal programlama modeli kullanılır. Çünkü DP güncellemesinden sonra, bazı eski düğümler bastırılmış olabilir. Fakat bu düğümler, diğer baskın düğümlerin bu düğümle bağlantıları olabileceğinden dolayı kolaylıkla elenemez. İşte bu yüzden, baskınlığın dualliği yöntemi kullanılır.
- *Sınırlandırılmış Denetçi (Bounded Backups):* Bu işlem ile denetçi büyüklüğü aynı kalmak şartıyla, düğüm parametreleri değiştirilir ve denetçinin değerinde iyileştirme yapılır. Bu iyileştirme, yine bir doğrusal programlama modeli oluşturularak gerçekleştirilir.

Sonlu ufukta, sonlu durum denetçisinin kullanımı, optimale yakınsamayı garantiler.

2.2.4 Klasik bir kısmi gözlemlenebilir Markov karar süreci örneği: kaplan problemi

POMDP'nin en yaygın örneklerinden bir tanesi *"Kaplan Problemi"* dir (Kaelbling *ve diğ.* 1998).

Problem şu şekilde özetlenebilir;

- Birinin arkasında aç bir kaplanın saklanmakta olduğu iki kapalı kapı vardır; Sağ kapı ve sol kapı.
- KV'nin amacı, kaplanın olmadığı taraftaki kapıyı açmaktır.
- KV sağ kapıyı açabilir, sol kapıyı açabilir ya da dinleyerek kaplanın yeri hakkında tahminlerde bulunabilir. (Dinleme işleminin doğru sonuç verme olasılığı 1'den küçüktür.)
- KV dinleme eylemini seçerse, bunun bir maliyeti olacaktır.
- Eğer KV'nin açtığı kapıdan kaplan çıkarsa, KV bir ceza, çıkmazsa da bir ödül elde edecektir.
- Her bir adımda, kapı açıldıktan sonra problem tekrar başa döner.

Bu örnekteki POMDP bileşenleri kısaca şu şekilde gösterilir;

- *Durumlar:*
    1. Kaplanın solda olması: Kaplan solda
    2. Kaplanın sağda olması: Kaplan sağda
- *Eylemler:*
    1. Dinleme eylemi: Dinle
    2. Sol kapıyı açma eylemi: SOL-AÇ
    3. Sağ kapıyı açma eylemi: SAĞ-AÇ
- *Gözlemler:*
    1. Kaplanın solda olduğunu düşünme: K-SOL
    2. Kaplanın sağda olduğunu düşünme: K-SAĞ

- *Geçiş Olasılıkları:*

1. Dinleme eylemi seçilirse, bir sonraki adımda kaplanın durumunda bir değişiklik olmaz. Bunu sağlayan olasılıklar Çizelge 2.1'de gösterilmiştir.

Çizelge 2.1 : Dinleme eyleminin geçiş olasılıkları

P(DİNLE)	Kaplan Solda	Kaplan Sağda
Kaplan Solda	1.0	0.0
Kaplan Sağda	0.0	1.0

2. Kapıları açma eylemleri seçilirse, Çizelge 2.2 ve Çizelge 2.3'de görüldüğü gibi, problem başa döner, yani kaplanın her iki kapı arkasında olma olasılığı eşittir.

Çizelge 2.2 : Sol kapıyı açma eyleminin geçiş olasılıkları

P(SOL-AÇ)	Kaplan Solda	Kaplan Sağda
Kaplan Solda	0.5	0.5
Kaplan Sağda	0.5	0.5

Çizelge 2.3 : Sağ kapıyı açma eyleminin geçiş olasılıkları

P(SAĞ-AÇ)	Kaplan Solda	Kaplan Sağda
Kaplan Solda	0.5	0.5
Kaplan Sağda	0.5	0.5

- *Gözlem Olasılıkları:*

1. Çizelge 2.4'te görüldüğü gibi, dinleme eylemi alındıktan sonra, KV'nin kaplanın, gerçekte bulunduğu tarafta olduğunu düşünme olasılığı daha fazladır.

Çizelge 2.4 : Dinleme eyleminin gözlem olasılıkları

P(DİNLE)	K-SOL	K-SAĞ
Kaplan Solda	0.85	0.15
Kaplan Sağda	0.15	0.85

2. Çizelge 2.5 ve Çizelge 2.6'da görüldüğü gibi, kapıları açma eylemleri, kaplanın nerede olduğu hakkında herhangi bir ipucu vermeyeceğinden, bu eylem seçildikten sonra KV'nin kaplanın durumu hakkındaki düşüncelerinin olasılıkları eşittir.

Çizelge 2.5 : Sol kapıyı açma eyleminin gözlem olasılıkları

P(SOL-AÇ)	K-SOL	K-SAĞ
Kaplan Solda	0.5	0.5
Kaplan Sağda	0.5	0.5

Çizelge 2.6 : Sağ kapıyı açma eyleminin gözlem olasılıkları

P(SAĞ-AÇ)	K-SOL	K-SAĞ
Kaplan Solda	0.5	0.5
Kaplan Sağda	0.5	0.5

- *Anlık Ödüller:*

1. Dinleme eyleminin KV için bir maliyeti vardır. Bu maliyetler Çizelge 2.7'de gösterilmiştir.

Çizelge 2.7 : Dinleme eyleminin her bir durumdaki maliyetleri

ÖDÜL(DİNLE)	
Kaplan Solda	-1
Kaplan Sağda	-1

2. Kapıları açma eylemlerinde, kaplanın olduğu kapıyı açmanın bir maliyeti vardır. Kaplanın olmadığı kapı açıldığında da, KV bir ödül kazanacaktır. Bu ödüller, Çizelge 2.8 ve Çizelge 2.9'da gösterilmiştir.

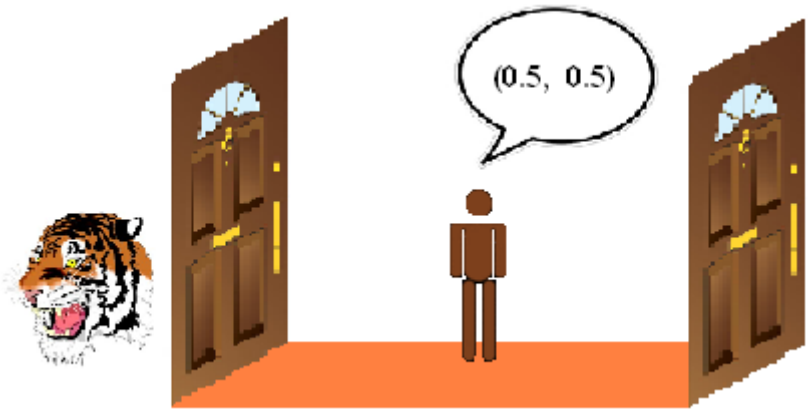
Çizelge 2.8 : Sol kapıyı açma eyleminin getireceği ödül ve maliyetler

ÖDÜL(SOL-AÇ)	
Kaplan Solda	-100
Kaplan Sağda	+10

Çizelge 2.9 : Sağ kapıyı açma eyleminin getireceği ödül ve maliyetler

ÖDÜL(SAĞ-AÇ)	
Kaplan Solda	+10
Kaplan Sağda	-100

Şekil 2.11'de görüldüğü gibi, başlangıç düşünülen durum (0.5, 0.5)'tir. Yani, KV için, kaplanın, iki kapının arkasında bulunma olasılıkları eşittir.



Şekil 2.11 : Kaplan probleminin, (0.5, 0.5) başlangıç düşünülen durumda gösterimi

Bu düşünülen durum için anlık ödülleri (2.35)'deki şekilde hesaplayabiliriz.

$$\begin{bmatrix} -100 & +10 \\ -1 & -1 \\ +10 & -100 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = \begin{bmatrix} -45 \\ -1 \\ -45 \end{bmatrix} \quad \textbf{(2.35)}$$

Yani bu düşünülen durum için eylemlerin değeri; SOL-AÇ:-45, SAĞ-AÇ=-45, DİNLE=-1'dir.

KV, eğer problem sadece tek bir adımdan oluşuyorsa, bu başlangıç düşünülen durumu için, maliyeti enküçükleyen dinleme eylemini seçecektir. Fakat ufuk uzunluğu birden büyükse, o halde her bir eylem için bir sonraki adımın gözönünde bulundurulması gerekmektedir. Bunun için de şu andaki düşünülen durumda, seçilmesi muhtemel eylemlerin ve gözlemlerin oluşturabileceği tüm kombinasyonlar göz önüne alınacak, yeni düşünülen durumlar bulunarak, değerleri hesaplanacaktır. Örnek olarak KV'nin bir sonraki adım için dinleme eylemini seçmesi ve K-SOL gözlemini elde etmesi halinde (2.36) denklemi kullanılarak, değerler (2.37)'deki gibi yerlerine konur ve KaplanSolda durumu için yeni düşünülen durum hesaplanır. Problem iki durumlu olduğundan, KaplanSağda durumunun olasılığı da (2.38)'deki gibi kolayca bulunur.

$$b_1(KaplanSolda) = \frac{P(K-SOL \backslash KaplanSolda, Dinle) \sum_{s_i \in S} P(KaplanSolda \backslash s_i, Dinle) b(s_i)}{\sum_{s_j \in S} P(K-SOL \backslash s_j,\ Dinle) \sum_{s_i \in S} P(s_j \backslash s_i, Dinle) b(s_i)}$$

(2.36)

$$b_1(KaplanSolda) = \frac{[0.85][1 \quad 0]\begin{bmatrix}0.5\\0.5\end{bmatrix}}{[0.85 \quad 0.15]\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}^T\begin{bmatrix}0.5\\0.5\end{bmatrix}} = \frac{0.425}{0.5} \tag{2.37}$$

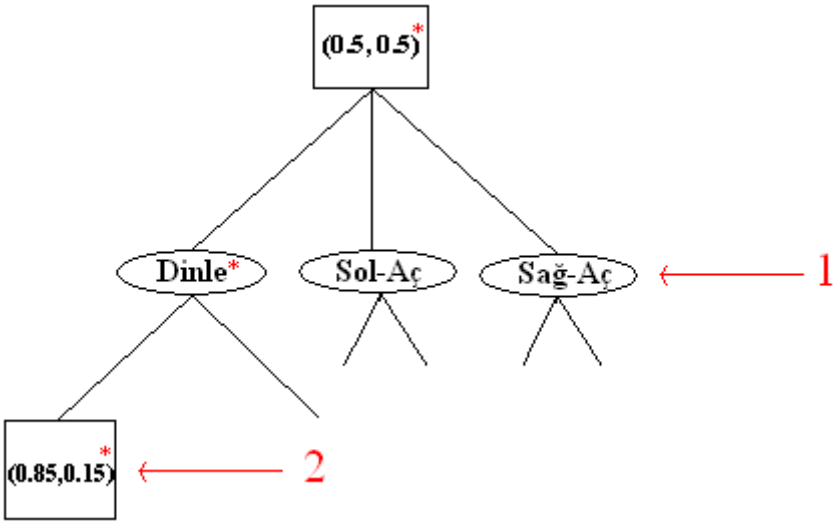
$$= 0.85$$

$$b_1(KaplanSağda) = 1 - 0.85 = 0.15 \tag{2.38}$$

Böylece yeni düşünülen durum Şekil 2.12'de görüldüğü gibi (0.85, 0.15) olarak elde edilir.

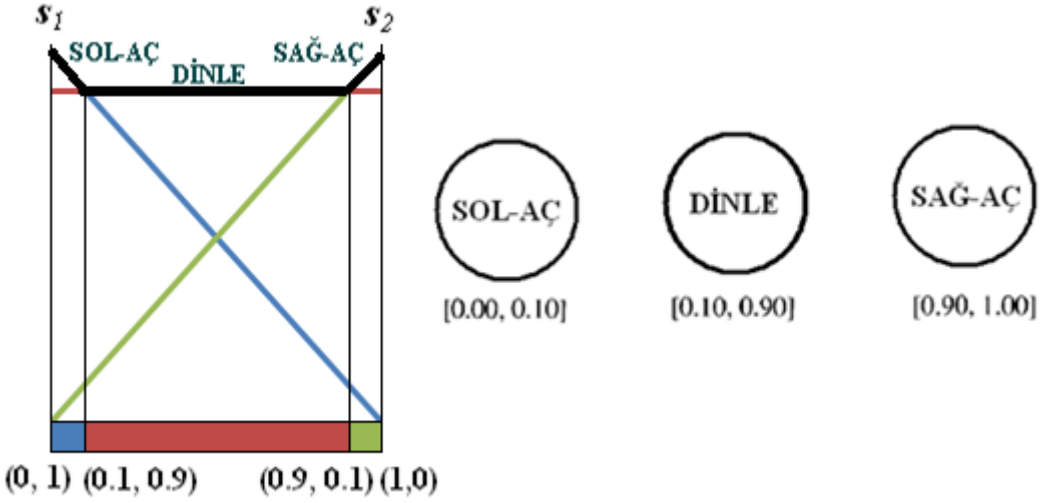
Şekil 2.12 : Kaplan probleminin (0.85, 0.15) düşünülen durumunda gösterimi

Yeni düşünülen durum elde edildikten sonra, ödüller daha önce gösterildiği gibi, yeni düşünülen durum kullanılarak tekrar hesaplanır. Bu işlemler, tüm muhtemel eylemler ve gözlemler göz önüne alınarak, hesaplanan tüm olası yeni düşünülen durumlar için yapılır. Şekil 2.13'te görüldüğü üzere, kırmızı yıldızlı yol, yukarıda hesaplanan yoldur. Bunun gibi diğer tüm kombinasyonlar da hesaplanarak bir ağaç oluşturulur ve bu ağaç üzerine POMDP çözüm yöntemlerinden biri uygulanarak en uygun plan elde edilir.



Şekil 2.13 : POMDP arama ağacı

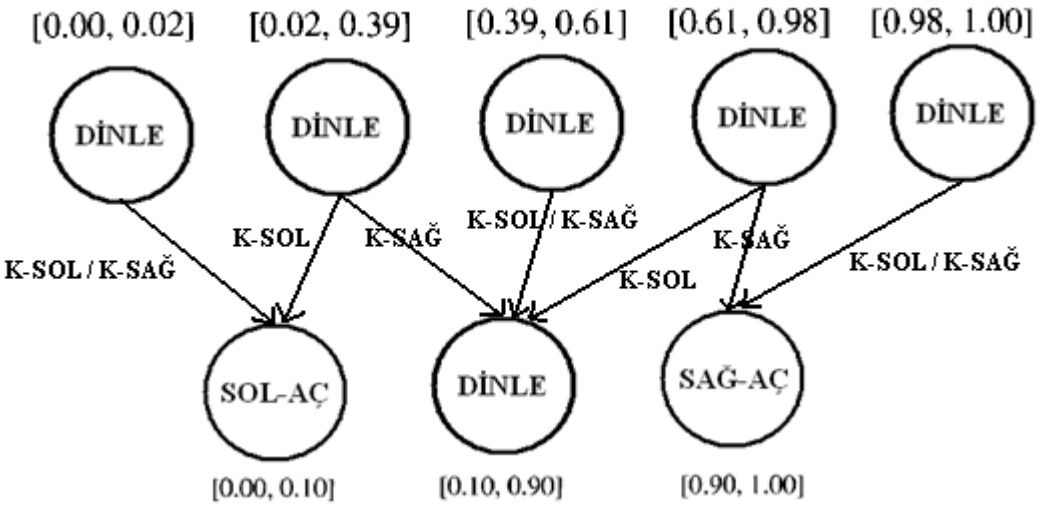
Tüm başlangıç durumları için düşünülerek, problem daha genel olarak ele alındığında, tek bir adımdan oluşan kaplan problemi için, Şekil 2.14'te gösterilen, optimal planlar incelenebilir. Bu optimal planlara göre, eğer birinci durumun başlangıç düşünülen durumu (0, 0.1) arasında ise, ödülü enbüyükleyen eylem SOL-AÇ eylemi olacağından bu eylem seçilecek, aynı şekilde (0.1, 0.9) arasında ise DİNLE eylemi, (0.9, 1) arasında ise SAĞ-AÇ eylemi seçilecektir. Bu örnekte, birinci durumun başlangıç düşünülen durumu 0.5 olduğundan DİNLE eylemi seçilmiştir.



Şekil 2.14 : Tek bir ufuk uzunluklu kaplan problemi için elde edilen optimal planlar (Kaelbling *ve diğ.* 1998)

Eğer problemin ufuk uzunluğu 2 ise, o halde Şekil 2.14'teki grafiğe, daha birçok vektör eklenecek ve aynı şekilde her bir eylem için, baskın olan alanlar bulunacaktır. Şekil 2.15'te, ufuk uzunluğu 2 olan bir problem için sabit olmayan bir plan

gösterilmektedir. Şekilden de görüldüğü gibi, problem uzunluğu 2 olursa, ilk adımda sadece dinleme eylemi alınabilecektir. Çünkü kapıları açma eyleminden sonra problem başa döneceğinden, ilk adımda kapıları açma eylemi seçilirse, düşünülen durum değişmeyecek, yani alınan eylemin KV'ye hiç bir katkısı olmayacaktır.



Şekil 2.15 : Ufuk uzunluğu 2 olan kaplan problemi için elde edilen sabit olmayan optimal plan (Kaelbling *ve diğ.* 1998)

Görüldüğü gibi, ufuk uzunluğunun artması, seçilecek olan eylemi değiştirebilir. Kısacası, ardışık karar verme modelleri, sadece bulunulan anı değil, uzun dönemi göz önüne alarak optimal eylemleri belirler.

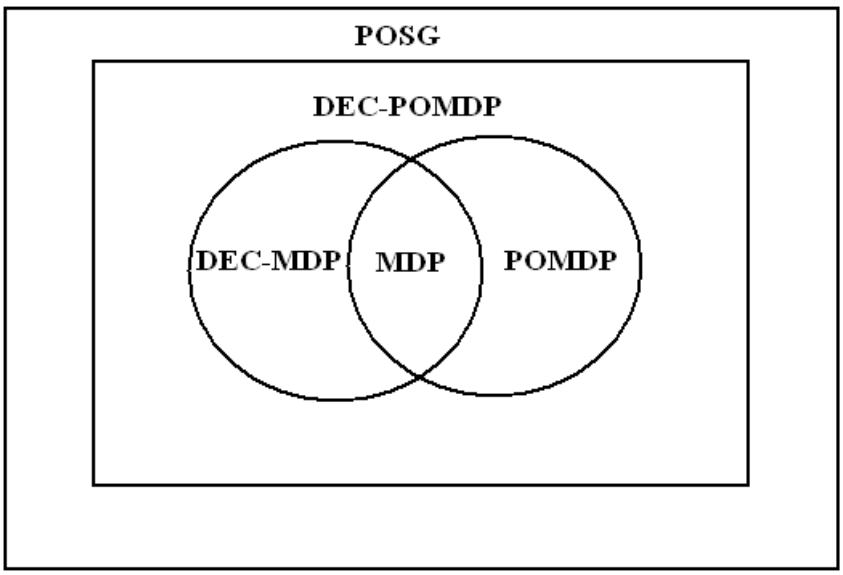
Bu problem için de, ufuk uzunluğu arttıkça sabit olmayan optimal eylem planı değişecektir.

2.3 Dağıtık Kısmi Olarak Gözlemlenebilir Markov Karar Süreci

İlk olarak, Bernstein ve Zilberstein (2000) tarafından geliştirilen, dağıtık kısmi olarak gözlemlenebilir Markov karar süreci, (decentralized partially observable Markov decision process, DEC-POMDP), POMDP'nin çok KV'li hale dönüştürülmüş şeklidir.

Bir DEC-POMDP modeli aynı zamanda, "Kısmi Olarak Gözlemlenebilir Stokastik Oyun (Partially Observable Stochastic Games, POSG)" olarak da nitelendirilebilir. Başka bir deyişle, DEC-POMDP, POSG'un tek ödül fonksiyonlu halidir. Burada KV'ler, aynı amaç için, birlikte hareket ederek ortak bir ödül fonksiyonunu enbüyüklemek ya da enküçüklemek isterler. POSG'ta ise KV'ler birbirinin rakibi olabilirler ve farklı amaç için hareket edebilirler.

Ardışık karar verme modelleri arasındaki ilişki Şekil 2.16'da görülmektedir. Şekilde görülen DEC-MDP, MDP'nin çok KV'li modelidir.



Şekil 2.16 : Belirsizlik altında ardışık karar verme modelleri arasındaki ilişki (Bernstein, 2005)

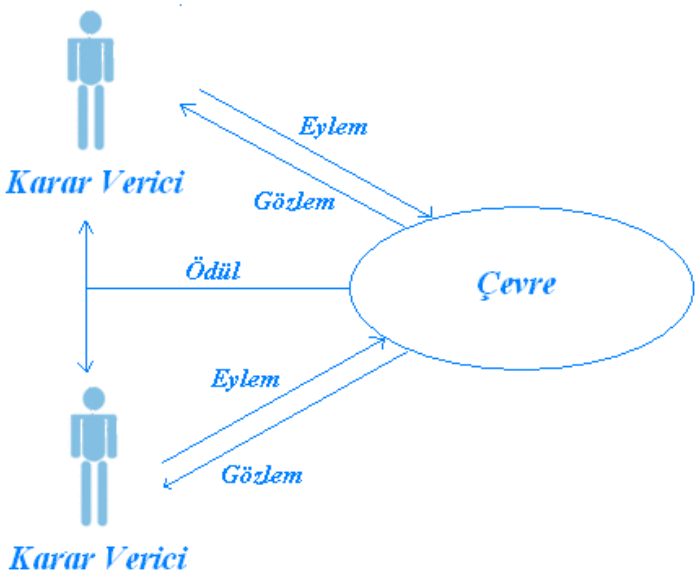
Bir DEC-POMDP modeli $< I, S, \{A_i\}, P, \{\Omega_i\}, O, R, T >$ şeklinde gösterilir (Seuken ve Zilberstein, 2008). Bu bileşenler şu şekilde tanımlanır;

- *Karar Vericiler (Agents):* Sonlu sayıda KV'lerin kümesidir; $I = 1, .., n$
- *Durumlar (States):* KV'lerin birlikte, bulunmaları muhtemel tüm durumlarının oluşturduğu durumlar kümesidir; $\forall s \in S$
- *Birleşik Eylemler (Joint Actions):* A_i, KV i'nin seçebileceği tüm muhtemel eylemlerin kümesidir. $\vec{A} = X_{i \in I} A_i$ ise, $\vec{a} =< a_1, \ldots, a_n >$ birleşik eylemlerinin oluşturduğu birleşik eylemler (joint actions) kümesidir. $\vec{a} \in \vec{A}$
- *Geçiş Fonksiyonu (Transition Function):* KV'lerin, mevcut durum s iken seçtikleri eylemlerle oluşturdukları $\vec{a}$ birleşik eylemi bilindiğinde, diğer tüm muhtemel durumlara geçiş olasılıklarını veren fonksiyondur;

  $T : SX\vec{A} \rightarrow \Delta \mathrm{S}, \mathrm{T}(\mathrm{s}' \backslash \mathrm{s}, \vec{\mathrm{a}})$

- *Birleşik Gözlemler (Joint Observations):* Ω_i, KV i'nin gözlemleyebileceği tüm muhtemel gözlemlerin kümesidir. $\vec{\Omega} = X_{i \in I} \Omega_i$ ise, $\vec{o} =< o_1, \ldots, o_n >$ birleşik gözlemlerinin oluşturduğu birleşik gözlemler (joint observations) kümesidir.

- *Gözlem Fonksiyonu (Observation Function):* KV'lerin, mevcut durumundan seçtikleri eylemlerle oluşturdukları $\vec{a}$ birleşik eylemini alarak geçtikleri s' durumunda, her bir $\vec{o}$ birleşik gözleminin elde edilme olasılığını veren fonksiyondur; $O: S X \vec{A} \rightarrow \Delta\Omega, O(\vec{o} \backslash s', \vec{a})$
- *Anlık Ödüller (Immediate Rewards):* KV'lerin, mevcut durum s iken seçtikleri eylemlerle oluşturdukları $\vec{a}$ birleşik eylemi bilindiğinde elde edecekleri kazancı veren fonksiyondur; $R: \vec{A} X S \rightarrow \Re, R(s, \vec{a})$



Şekil 2.17 : Bir DEC-POMDP modelinde karar vericilerin çevreyle olan etkileşimi

Şekil 2.17'de görüldüğü gibi, her bir adımda, KV'lerin her biri, kendi yerel gözlem geçmişlerine (local observation histories) göre bir eylem seçer. Bunun sonucunda da ortak bir anlık ödül ve her bir KV için ayrı bir gözlem elde edilir. Durum tam olarak gözlemlenemediği için, KV'lerin kendi gözlem geçmişlerini hatırlamaları, KV'lere fayda sağlar.

KV'lerin her biri kendi gözlemleri dışında, diğer KV'lerin elde ettikleri gözlemleri ve seçtikleri eylemleri bilemezler. Sadece, her bir KV, eylemini seçtikten sonra geçilen yeni ortak durumla ilgili ipuçları içeren gözlemler elde ederler.

DEC-POMDP'de plan kavramını iki ayrı şekilde ele almak mümkündür;

- *Yerel Plan (Local Policy):* Bir i KV'si için yerel plan, δ_i, bu KV'nin yerel gözlem geçmişlerinden seçtiği eylemlere giden bir fonksiyondur; $\delta_i: \Omega_i^* \rightarrow A_i$

- *Birleşik Plan (Joint Policy):* Birleşik plan, her bir KV'nin planlarının farklı kombinasyonlarının oluşturduğu kümedir; $\delta =< \delta_1, \dots, \delta_n >$

Bir DEC-POMDP'nin çözümü, beklenen toplam ödülü enbüyükleyen bir birleşik plandır.

2.3.1 Performans kriterleri ve değer fonksiyonları

Bir DEC-POMDP modelinde, çözüm, iki farklı performans kriteri kullanılarak bulunabilir;

Sonlu Ufuklu (Finite-Horizon) Performans Kriteri: Bir DEC-POMDP modelinde ardışık durumlar için belirlenmiş bir zaman ufku uzunluğu varsa, çözüm için sonlu ufuklu performans kriteri kullanılır. Yani böyle bir durumda, KV'lerin, önceden belirlenmiş sayıdaki zaman aralıklarında karar verecekleri göz önüne alınacaktır.

s_0 durumu ile başlayan T ufuk uzunluklu bir δ birleşik planının değeri (2.39)'da görüldüğü şekilde gösterilir (Seuken ve Zilberstein, 2008).

$$V^{\delta}(s_0) = E_{s_0,\delta}\left[\sum_{t=0}^{T-1} R(s_t, \vec{a}_t)\right] \quad \textbf{(2.39)}$$

$R(s_t, \vec{a}_t)$, t zaman aralığında, s durumunda $\vec{a}$ birleşik eyleminin seçilmesi halinde elde edilecek olan anlık ödülü göstermektedir.

Sonsuz Ufuklu (Infinite-Horizon Discounted) Performans Kriteri: Bir DEC-POMDP modelinde ardışık durumlar için belirlenmiş herhangi bir zaman ufku uzunluğu yoksa ya da zaman ufku uzunluğu sonsuz kabul edilebilecek kadar büyükse, çözüm için sonsuz ufuklu performans kriteri kullanılır. Bu tip problemlerde, MDP ve POMDP'de olduğu gibi, $0 < \alpha < 1$ olacak şekilde, önceden belirlenmiş bir α değeri kullanılır.

Değer fonksiyonu, sonsuz ufukta başlangıç durumu s_0 olan bir δ planı için (2.40)'da görüldüğü şekilde hesaplanır (Seuken ve Zilberstein, 2008).

$$V^{\delta}(s_0) = E_{s_0,\delta}\left[\sum_{t=0}^{\infty} \alpha^t R(s_t, \vec{a}_t)\right] \quad \textbf{(2.40)}$$

2.3.2 Dağıtık kısmi olarak gözlemlenebilir Markov karar sürecinin çözüm yöntemleri

Bir DEC-POMDP modelinin çözümünün bulunması, POMDP ve MDP'de olduğu kadar kolay değildir. Bunun için hala yeni yöntemler geliştirilmeye çalışılmaktadır. Bu bölümde bunlardan bazılarına yer verilecektir.

2.3.2.1 DEC-POMDP için genelleştirilmiş dinamik programlama

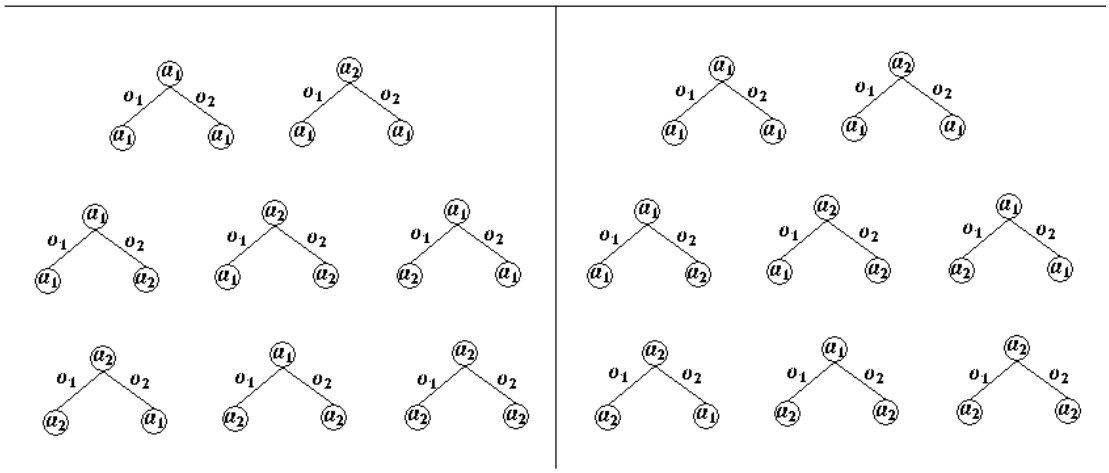
Sonlu ufuklu bir DEC-POMDP'de DP yapılırken ilk olarak tüm KV'lerin planları aynı anda oluşturulur. Temel mantık, bir KV'nin planı sabit tutulup diğer KV'nin tüm planlarıyla ikişerli olarak, tüm plan kombinasyonları göz önüne alınacak şekilde değerlendirilmesidir. Burada her bir plan ikilisinin değeri bulunur. Bu değerlere bakılarak baskın olmayan plan ikilileri elenir. Budama işlemi, genelleştirilmiş düşünülen durum uzayını kullanır. Bu işlem bir doğrusal programlama kullanılarak yaptırılabilir. Fakat burada, POMDP'de olduğu gibi, planlar oluşturulurken aynı anda budama yapılamaz. DP ile, DEC-POMDP için optimal bir birleşik plan elde edilir.

Aşağıda 2 ufuk uzunluğuna sahip, iki KV'li bir DEC-POMDP probleminde DP uygulaması açıklanmaktadır.

Şekil 2.18, problemin ilk adımını göstermektedir. Burada her iki KV için de, seçebilecekleri 2 eylem vardır. KV'lerin bu eylemlerinin oluşturacağı 4 adet plan ikilisi değerlendirilir. Şekilde, KV'lerin eylemlerinden herhangi biri, baskın olmama gibi bir durum sağlamadığından elenen bir eylem yoktur.

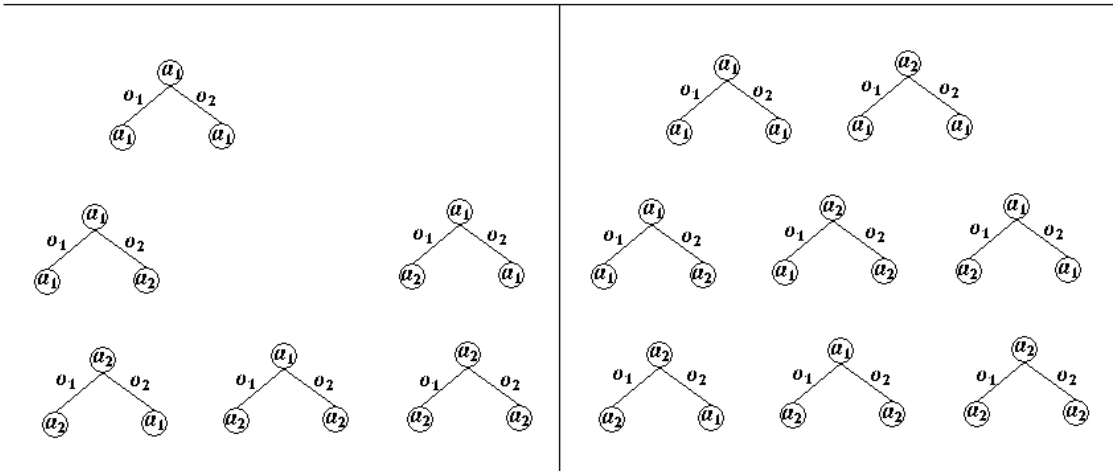
Şekil 2.18 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın ilk adımı (Zilberstein, 2010b)

İkinci adımda, her bir gözlem ve sonrasında seçilecek eylemlerin farklı kombinasyonları göz önüne alınmalıdır. Şekil 2.19, KV 1 ve KV 2 için tüm bu kombinasyonların göz önüne alındığı planları içermektedir.



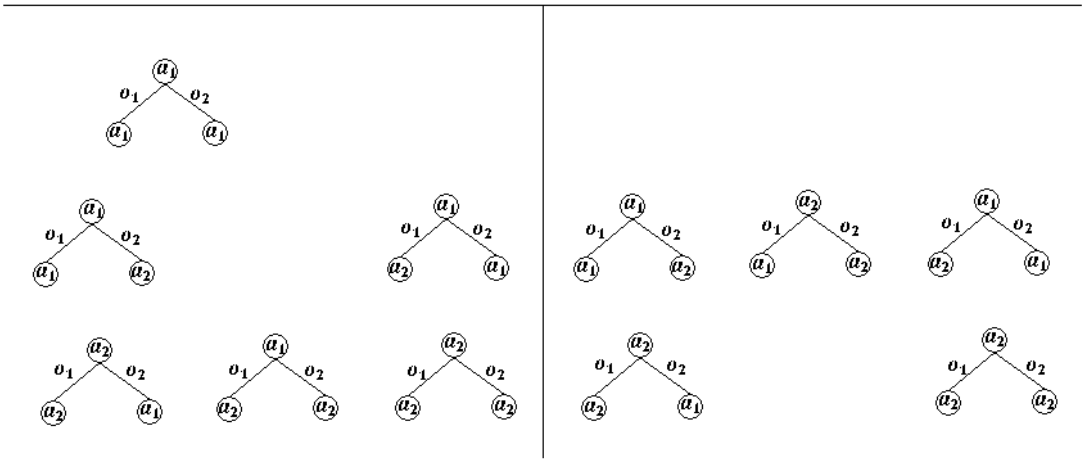
Şekil 2.19 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın ikinci adımı (Zilberstein, 2010b)

Şekil 2.20'de KV 2'nin bir planı sabit tutularak, KV 1'in tüm planlarıyla oluşturacağı farklı birleşik planların değerleri hesaplanmıştır. Tüm bu plan ikililerinde, KV 1'in herhangi bir planı, göz önüne alınan tüm bu plan ikilileri arasında bir baskınlık oluşturmuyorsa, KV 1'in bu planı elenir. Şekil 2.20'de KV 1'in bu tip planlarının elendiği görülmektedir.



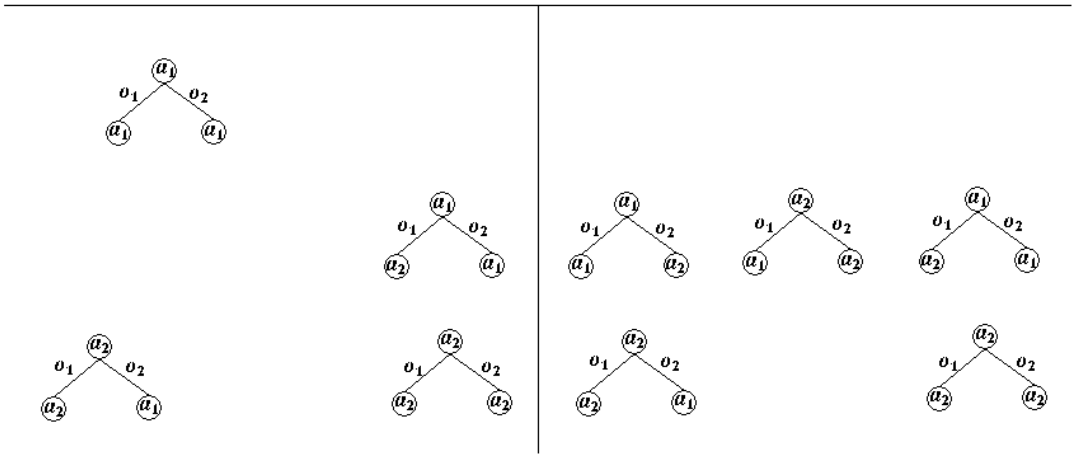
Şekil 2.20 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın üçüncü adımı (Zilberstein, 2010b)

Şekil 2.21'de, KV 1'in bir planı sabit tutularak, aynı şekilde, KV 2'nin tüm planlarıyla oluşturacağı farklı birleşik planların değerleri hesaplanmıştır. Bu sefer de KV 2'nin planları içerisinden baskınlık oluşturmayanlar elenmiştir.

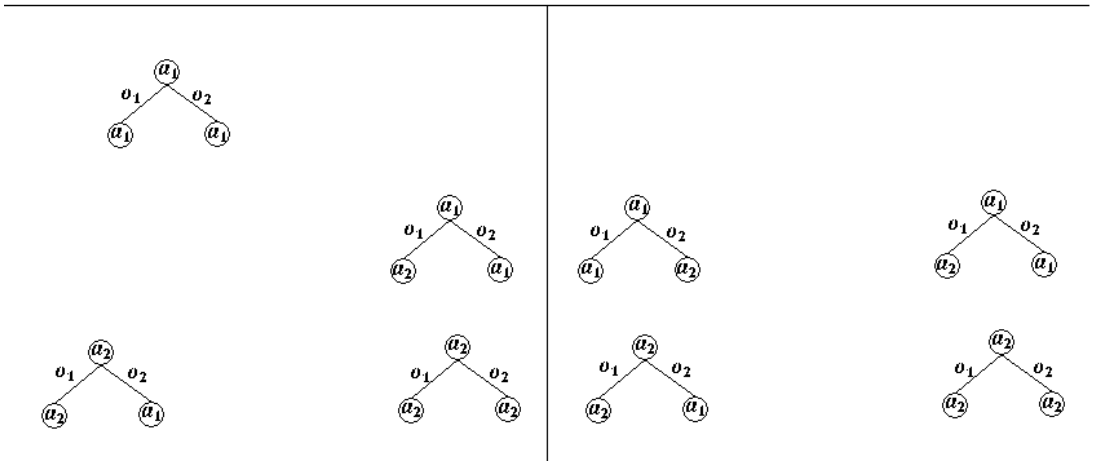


Şekil 2.21 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın dördüncü adımı (Zilberstein, 2010b)

Şekil 2.22 ve Şekil 2.23'te de bu işlemler sırasıyla devam etmektedir.



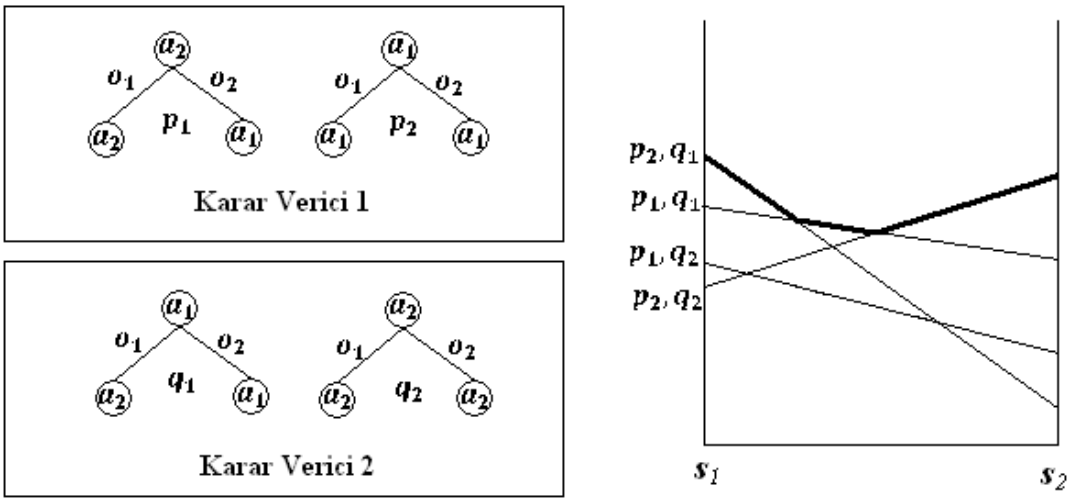
Şekil 2.22 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın beşinci adımı (Zilberstein, 2010b)



Şekil 2.23 : İki ufuk uzunluğuna sahip bir DEC-POMDP modelinin çözümü için uygulanan dinamik programlamanın beşinci adımı (Zilberstein, 2010b)

Eğer bu problem için, üçüncü bir adım daha olsaydı, kalan plan ağaçları kullanılarak, 3 adımlı yeni plan ağaçları oluşturulacak ve aynı işlemler bunlar üzerinde uygulanacaktı.

Şekil 2.24, bu işlemler daha basitleştirilerek anlatılmaktadır. Burada KV 1'in ilk planı ile KV 2'nin ikinci planının oluşturduğu birleşik plan baskın olmadığı için elenecektir. Grafikte de birleşik planların optimal olduğu, düşünülen durum aralıkları, doğrusal, parçalı, konveks fonksiyon ile gösterilmektedir.



Şekil 2.24 : İki karar vericili bir DEC-POMDP modeli için genelleştirilmiş dinamik programlama (Bernstein ve Zilberstein, 2003)

Tüm bu dinamik programlama işlemlerinin, sonsuz ufuklu bir DEC-POMDP problemine uygulanması çok zordur. Bu yüzden POMDP'deki plan iterasyonu yöntemi, sonlu durum denetçileri kullanılarak, DEC-POMDP'ye genişletilmiştir.

2.3.2.2 Bağıntılı sonlu durum denetçileri

Bu yöntemde, her bir KV için bir stokastik sonlu durum denetçisi (finite state controller) kullanılarak birleşik bir plan oluşturulur.

DEC-POMDP'de yerel bir denetçi (local controller) $< Q_i, \Omega_i, A_i, \psi_i, \eta_i >$ şeklinde ifade edilir (Bernstein, 2005). Bu bileşenler;

- Q_i, denetçi düğümlerinin sonlu kümesidir.
- Ω_i, i KV'sinin yerel gözlemleri olarak ele alınan girdilerdir.
- A_i, i KV'sinin eylemleri olarak ele alınan çıktılardır.
- $\psi_i: Q_i \rightarrow \Delta A_i$, i KV'si için, kendi durum denetçisinin her bir düğümünde seçilen eylemlerin dağılımını gösteren eylem seçme fonksiyonudur.
- $\eta_i: Q_i x A_i x \Omega_i \rightarrow \Delta Q_i$, i KV'si için, her bir başlangıç düğümü ve bu düğümde seçilen eylemin dağılımını gösteren geçiş fonksiyonudur.

Burada, POMDP'de kullanılan sonlu durum denetçisi, her KV'nin planlarını ifade etmek için kullanılır. Fakat, farklı olarak bu durum denetçilerine bir *"bağıntı aleti (correlation device)"* eklenir (Bernstein, 2005). Bu bağıntı aleti, tüm KV'ler tarafından paylaşılan bir rastsallık kaynağıdır. Bu, KV'ler arasına herhangi bir iletişim eklenmeden, çözümün kalitesinin arttırılmasını sağlar. Bağıntı aleti, her adımda, her bir KV'ye bir sinyal gönderir. Fakat bağıntı aleti ile KV'ler, diğer KV'lerin gözlemleri hakkında bilgi edinemezler.

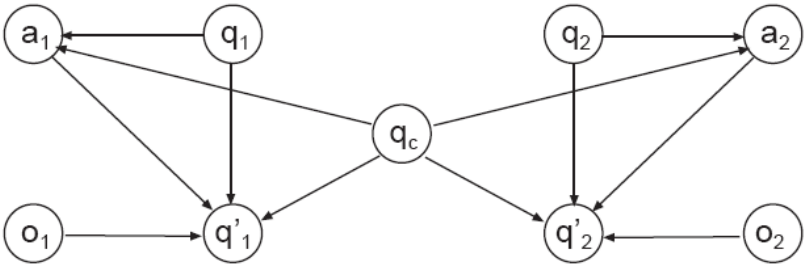
Bağıntı aleti $< Q_c, \psi_c >$ şeklinde ifade edilir. Bu bileşenler;

- Q_c, düğümler kümesidir.
- $\psi_c: Q_c \rightarrow \Delta Q_c$ olmak üzere durum geçiş fonksiyonudur.

Bağıntı aleti, her bir adımda geçiş fonksiyonu ile bir geçiş sağlar ve her KV, bulundukları durumu gözlemler.

Artık her bir adımda, her bir KV'nin bulunduğu düğümlerin oluşturduğu vektöre, bağıntı aleti de eklenecektir. Yani; $\vec{q} =< q_c, q_1, \dots, q_n >$ olacaktır. Tüm KV'lerin yerel durum denetçileri ve bağıntı aleti göz önüne alınarak bir *"Bağıntılı birleşik durum denetçisi (correlated joint controller*)" tanımlanabilir.

Şekil 2.25'te, bir bağıntılı birleşik durum denetçisinde, iki KV için, her KV'nin, her adımda sonlu durum denetçisinde bir eylem alıp başka bir düğüme geçişi ve burada bir gözlem elde edişinin bağıntı aletiyle birleştirilmiş hali gösterilmiştir.



Şekil 2.25 : İki karar vericili durum için bir bağıntılı birleşik denetçisindeki olasılıklı bağlılıkların grafik gösterimi (Bernstein, 2005)

Bir bağıntılı birleşik durum denetçisinin değer fonksiyonu (2.41)'deki gibi hesaplanabilir (Bernstein, 2005).

$$V(s,\vec{q}) = \sum_{\vec{a}} P(\vec{a},\vec{q})[R(s,\vec{a}) + \alpha \sum_{s',o',\vec{q}'} P(s',\vec{o}\backslash s,\vec{a})P(\vec{q}'\backslash\vec{q},\vec{a},\vec{o})V(s',\vec{q}')] \qquad \textbf{(2.41)}$$

Buradaki $P(s',\vec{o}\backslash s,\vec{a})$ ifadesi mevcut durum ve tüm KV'ler tarafından alınan birleşik eylemler bilindiğinde, s' durumuna geçme ve bu durumda $\vec{o}$ birleşik gözlemini elde etme olasılığını verir. $P(\vec{q}'\backslash\vec{q},\vec{a},\vec{o})$ ifadesi ise, KV'lerin bulundukları düğümlerde aldıkları birleşik eylemlerden sonra, elde ettikleri $\vec{o}$ birleşik gözlemini, geçtikleri yeni düğümlerden oluşan $\vec{q}'$ birleşik düğümünde elde etme olasılığını verir. $P(\vec{a},\vec{q})$, $\vec{q}$ birleşik düğümünde, $\vec{a}$ birleşik eylemlerinin olasılıklarını, $R(s,\vec{a})$ ise s durumunda $\vec{a}$ birleşik eylemi alındığında elde edilecek ödülü göstermektedir. Bu işlem, MDP ve POMDP'deki değer iterasyonu mantığı ile aynıdır.

Herhangi bir başlangıç düşünülen durumu için değer hesaplaması (2.42)'de görüldüğü gibidir (Bernstein, 2005).

$$V(b) = max_{\vec{q}} \sum_{s} b(s)V(s,\vec{q}) \qquad \textbf{(2.42)}$$

$b(s)$, s durumunun düşünülen durumunu, $V(s,\vec{q})$ ise $\vec{q}$ birleşik düğümündeki s durumunun değerini göstermektedir.

2.3.2.3 Plan iterasyonu

DEC-POMDP için plan iterasyonunda, DP güncellemesi yerine, "detaylı denetçi *(exhaustive backup)"* kavramı kullanılacaktır (Bernstein, 2005). Bir sonlu durum denetçisinde, belirli bir eylemin alınması halindeki tüm olasılıklar göz önüne alındığında, gözlemlerin ve durumların farklı kombinasyonlarının oluşturduğu küme, detaylı denetçi olarak adlandırılabilir. Detaylı denetçi işlemi, her bir KV'nin yerel denetçilerine düğümler ekler.

DEC-POMDP'de plan iterasyonunda ilk olarak, sistemin doğrusal denklemleri çözülerek, birleşik durum denetçisinin değerlendirmesi yapılır.

İkinci adımda, detaylı denetçi kullanılır ve yerel denetçilere deterministik düğümler eklenir.

Üçüncü adımda, bazı ek işlemler uygulanır. Bu ek işlemler;

- *Denetçi Azaltma (Controller Reductions):* Denetçi azaltma işlemi, bir dual doğrusal programlama modeli kullanarak baskın olmayan düğümleri denetçi içerisinden eler.
- *Sınırlandırılmış DP Güncellemeleri (Bounded DP Updates):* Bu işlem ile, denetçi büyüklüğü aynı kalmak şartıyla, düğüm parametreleri değiştirilir ve denetçinin değerinde iyileştirme yapılır. Bu iyileştirme, yine bir doğrusal programlama modeli oluşturularak gerçekleştirilir.

DEC-POMDP'de, POMDP'de olduğu gibi, ε-optimale yakınsamayı test edecek bir Bellman kalanı yoktur. Bu yüzden son adımda, $|R_{max}|$, anlık ödülün mümkün en büyük değeri olmak üzere, $\frac{\alpha^{t+1}|R_{max}|}{1-\alpha} \leq \varepsilon$ koşulunun sağlanıp sağlanmadığı kontrol edilir. Bu koşul sağlandığında, algoritma sona erer (Bernstein, 2005).

3. TEDARİK ZİNCİRİNDE BELİRSİZLİK ALTINDA ARDIŞIK KARAR VERME

Günümüzde sürekli artan rekabet koşullarında şirketler, bir yandan pazar paylarını arttırmaya çalışırken, diğer yandan da bu hedeflerini gerçekleştirmek için, maliyetlerini düşürmeyi hedeflemektedirler. Bu yüzden potansiyel müşterilere doğru zamanda, doğru ürünleri, doğru şekilde ulaştırırken, bir yandan da bunu gerçekleştirmek için tedarikçilerle etkin bir şekilde çalışarak, doğru hammadde veya malzemeyi, doğru zamanda ve en düşük maliyetle tedarik etmek gerekmektedir. Böyle bir sistemin, günümüzün büyük şirketlerinde oluşturulabilmesi ve rekabet düzeylerinin devamı için tedarik zinciri sisteminin kurulması gerekmektedir (Eymen, 2007).

Tedarik zinciri teknik olarak, malzeme tedariği işlemlerini yerine getiren, bunları yarı mamul ve mamullere dönüştüren ve daha sonra bunları dağıtım kanalıyla müşterilere ulaştıran hizmet ve dağıtım seçeneklerinden oluşan şebekedir. Bu şebeke, malzemelerin sağlanması, bu malzemelerin ara ve tamamlanmış ürünlere dönüşümü ve tamamlanmış ürünlerin müşterilere dağıtımı fonksiyonlarını yerine getirir (Eymen, 2007).

Benzer olarak, Simchi-Levi *ve diğ.* (2000) bu kavramı "Tedarikçileri, üreticileri, depoları, stokları ve ürünleri, doğru zamanda, doğru miktarda, doğru yerde, servis seviyesi gereksinimleri karşılanacak ve aynı zamanda sistem maliyetleri enküçüklenecek şekilde birleştiren yaklaşımlar kümesidir." şeklinde tanımlamışlardır.

Tedarik zinciri yönetiminde, ürünlerin tedarikçiden son kullanıcıya ulaşmasında bir dizi fonksiyon belirli görevleri, temel hedefler doğrultusunda yerine getirir. Bunlar, talep ve sipariş yönetimi, planlama, stok kontrolü, depo yönetimi ve sevkiyat olarak özetlenebilir (Eymen, 2007).

Bu çalışmada, bu fonksiyonlardan stok kontrolü özel olarak ele alınacaktır.

Bir şirketin maliyet yapısı, şirketin bulunduğu endüstrinin tipine bağlı olsa da, birçok endüstride, maliyet yapısının en önemli kısmı Stok Maliyetleri'dir. Elde yeteri kadar stok bulundurma, işlemlerin daha rahat yürümesini ve müşteri servis seviyesini arttırır. Fakat eldeki kazancın tamamen stoklara yatırılması da yanlıştır. Bunun için, verimli ve etkin stok kontrol planlarına ihtiyaç duyulmaktadır.

Bilimsel stok kontrolünün ana amacı; hammadde, malzeme, yedek parça ve diğer gereksinim maddelerinden işletmede ne eksik, ne fazla, ancak yeterli miktarda hazır bulundurarak üretimin aksatılmadan yürütülmesidir (Tanyaş ve Baskak, 2003).

İşte bu amaç doğrultusunda, tedarik zinciri sistemindeki stoklar, belli periyotlarda ardışık olarak kontrol edilir. Tedarik zinciri bireylerinin her biri, elde bulundurma ve bulundurmama gibi maliyetlerini enküçükleyecek şekilde, o dönem içerisindeki sipariş ya da üretim miktarlarına karar vermek zorundadırlar. Bu kararı zorlaştıran, gelecek dönemdeki talebin belirsizliğidir. Bu tip, belirsizlik altında ardışık karar verme modelleri, talep gözlemlerini de kullanarak, gelecek talep hakkında bir çıkarım yapar ve tüm periyotlar boyunca izlenecek en iyi stratejileri içeren bir plan oluşturur. Bu plan, tedarik zincirindeki KV ya da KV'lere, kararlarını oluşturmalarında büyük fayda sağlar.

Bu stok kontrol problemlerinde talep süreci farklılık gösterebilir. Kambhamettu (2000), çalışmasında, ardışık karar verme problemlerindeki talep sürecini 4 grupta incelemiştir.

3.1 Sabit, Tam Olarak Gözlemlenebilir Talep ve Yapılan Çalışmalar

Bu problem sınıflarında, talep süreci sabittir (stationary), yani talep tek bir olasılık dağılımı ile ifade edilir. Tam bilgiden kasıt, dağılımın parametrelerinin tam olarak biliniyor olmasıdır. Bu tip problemler en az karmaşıklığa sahip problemlerdir ve geniş kapsamlı olarak üzerinde çalışılmaktadır.

Lee ve Nahmias (1993), yaptıkları bir çalışmada bu tip bir problem için olan bir klasik teoriyi özetlemişlerdir. Lariviere ve Porteus (1997), talepleri değil, satışları ele almışlar ve satışların tam olarak gözlemlenebilir olduğu bir problemde Bayes tekniklerini kullanarak kayıp satışları incelemişlerdir. Böyle bir durumda, perakendeci, elde fazla stok bulundurarak doğru talep süreci hakkında daha fazla bilgi edinebilir.

3.2 Sabit, Kısmi Olarak Gözlemlenebilir Talep ve Yapılan Çalışmalar

Bu tip problemlerde talep süreci sabittir (stationary), yani tek bir olasılık dağılımına göre ifade edilir. Kısmi gözlemlenebilirlikten kasıt ise, talep dağılımının bilinmeyen parametrelere sahip olmasıdır.

Kısmi bilgili problemlerin çözümü, tam bilgi elde edilebilir durumlara göre daha zordur. Çünkü, tam bilginin azlığı, problemin karmaşıklığını arttırmaktadır. Sabit, kısmi olarak gözlemlenebilir yapıdaki problemler, Bayes metotları kullanılarak çözülebilir. Scarf (1959;1960), stok kontrolünde Bayes metotlarını kullanan öncülerden biridir. Bu metotlar kullanılırken, sipariş ve stok maliyetlerinin doğrusal olduğu varsayımı yapılmıştır. Azoury (1985), Scarf'ın elde ettiği sonuçları genişletmiştir. Kaplan (1988) da bir tedarik zinciri probleminde talep bilgisini güncellemede Bayes yaklaşımını kullanmıştır. Pierskalla (1969), talebin sabit olduğu, sonlandırma tarihinin ise bilinmediği bir stokastik talep problemi üzerinde çalışmıştır. Lovejoy (1990), ilk olarak, miyopik bir parametre tekniği ve basit bir stok kontrol planı kullanmıştır. Özel olarak, parametre tahminleri için Bayes güncellemeleri yapmıştır. Ayrıca, miyopik yaklaşımın kullanıldığı durumlarda, optimal maliyetlere bağlı değer kaybının sınırları belirlenmiştir. Daha sonraki yıllarda yaptığı diğer bir çalışmada Lovejoy (1992), miyopik plan analizini, belli bir noktada miyopik davranışı sonlandıran planları göz önüne alarak genişletmiştir. Diğer yıl ise, sabit talepli, kısmi olarak gözlemlenebilir bilgiye sahip stok kontrol problemlerini genişletmiş, bu problemlere eklemeler yapmıştır (Lovejoy, 1993). Bu eklemeler alt optimal planların sınırlarını ele alır. Bu sınırları iteratif olarak daraltmanın nasıl mümkün olacağını göstermiştir. Ayrıca talep dağılımı sabit iken, bazı parametrelerin sabit olamayabileceğini göstermiştir. Talep dağılımının kısmi olarak gözlemlenebilir, stok seviyesinin ise tam olarak gözlemlenebilir olduğu birleşik durumlu bir POMDP modeli önermiştir. Nahmias (1994), talebin normal dağılımda olduğu ve karşılanamayan taleplerin kayıp satış olarak ele alındığı durumda, talep tahminleri için bir metot önermiştir. Lariviere ve Porteus (1999), müşteri talebinin sabit ve kısmi gözlemlenebilir olduğu problemler için Bayes yaklaşımını kullanmışlardır.

3.3 Sabit Olmayan, Tam Olarak Gözlemlenebilir Talep ve Yapılan Çalışmalar

Bu tip problemlerde talep, her bir adımda farklılık gösteren olasılık dağılımları ile ifade edildiğinden sabit değildir (non-stationary). Fakat talep dağılımlarının tüm parametreleri tam olarak bilinmektedir.

Pratikteki problemler genellikle, sabit olmayan ve kısmi gözlemlenebilir problemler olduğu halde, denetçiler, hesaplama karmaşıklığından dolayı, tahminleri kullanarak dağılımların tam olarak bilindiğini varsaymaktadırlar. Kısacası, bu tip problemler sık sık pratikteki iş senaryolarına uyarlanmaktadır.

Hadley ve Whitin (1961) tarafından yapılan çalışma, bu konu üzerindeki ilk araştırmalardan sayılabilir. Bundan çok kısa bir süre önce ise Karlin (1960), talep dağılımının her periyotta değiştiği, sabit olmayan stokastik bir sistem (dinamik sistem) analiz etmiştir. Bu, Arrow *ve diğ.* (1951)'nin önerdiği dinamik stok kontrol modelinin geliştirilmiş şeklidir. Karlin, talep yoğunluklarının bir fonksiyonu olarak, optimal stok seviyesinin zamanla nasıl değişeceğini göstermiştir. Graves (1997), sabit olmayan problemler için bir stok planı önermiştir.

Bu alanda, Song *ve diğ.* (1993) dalgalı talep çevrelerini incelemiş ve bu çevreyi sürekli zamanlı bir Markov zinciri olarak modellemişlerdir. Burada her bir durum, Poisson oranı ile belirlenmiştir. Ayrıca, bu çalışmada, optimal planların parametreleri için sınırlar geliştirilmiştir. Daha sonraki yıllarda Song *ve diğ.* (1996), benzer bir talep modeli kullanarak, stokların nasıl yönetilebileceğini göstermişlerdir. Modellerinde mevcut talebin tam olarak gözlemlenebildiği varsayımı kullanılmıştır. Elde ettikleri optimal planları iki sezgisel metotla kıyaslamışlardır. Bu tip problemlerin çözümü için literatürde, daha birçok farklı çalışmalar yapılmıştır (Anupindi *ve diğ.* 1996).

3.4 Sabit Olmayan, Kısmi Olarak Gözlemlenebilir Talep ve Yapılan Çalışmalar

Talep süreci, bu durumda, sabit olmayan (non-stationary), yani her bir adımda değişen olasılık dağılımları ile ifade edilir. Bu olasılık dağılımları, bazı bilinmeyen parametrelere sahip oldukları için, tam olarak gözlemlenemezler ve bunlar hakkında kesin bir bilgi elde edilemez. Fakat her bir adımda elde edilen gözlemler kullanılarak, kısmi olarak gözlemlenebilirler.

Ürünlerin yaşam çevriminin gelişen teknolojiyle birlikte kısalmış olması ve rekabet ortamında taleplerdeki dalgalanmalardan dolayı, pratikte karşımıza çıkan stok kontrol problemleri bu çeşit problemlerdendir. Yani talep genellikle kısmi gözlemlenebilir ve sabit olmayan taleptir. Pratikteki stok kontrol problemleri, genellikle bu tarz problemler olduğu halde, bu problemlerin çözümü karmaşık ve zor olduğundan dolayı üzerinde çok çalışılmamaktadır.

Kurawarwala ve Matsuo (1996), bu tip bir stok kontrol probleminde, talep sürecindeki parametrelerin tahmini için bir model önermiştir. Burada, üretim kararları problemin en başında alınmaktadır. Kestirim modelleri parametrelerinin, başlangıçtaki tahmini için yeni bir teknik önerilmiştir.

Treharne ve Sox (1999) çalışmalarında, talep sürecinin sabit olmayan ve kısmi olarak gözlemlenebilir olduğu bir stok kontrol problemini ele almış ve talep sürecinin en az bir parametresinin belirsizlik altında olduğu durumlardaki stok kontrol problemi için, bütünleştirici bir model önermişlerdir. Bu bilinmeyen parametrenin değeri, sabit ya da sabit olmayan şekillerde olabilir. Aynı yıl başlattıkları proje de, yine kısmi gözlemlenebilir, sabit olmayan, rastsal taleplere sahip bir ürünün stok kontrol problemini ele almaktadır (Treharne ve Sox, 1999-2002). Talebin olasılık dağılımı, herhangi bir zaman noktasında kesin olarak bilinememektedir. Ayrıca bu dağılım rastsal olarak bir periyottan diğerine değişmektedir. Talep süreci, bir önceki periyottaki talebe bakılarak kısmi olarak gözlemlenebilmekte ve talep sürecinin sabit olmaması belirsizliği arttırmaktadır.

Treharne ve Sox (2002), daha sonraki yıllarda, bir stok kontrol problemi için farklı planları incelemişlerdir. Her bir periyotta talebin olasılık dağılımı, Markov zinciri kullanılarak belirlenmiştir. Fakat bu süreçte, sadece mevcut şimdiki talep gözlemlenebilmektedir. Bu talep süreci, POMDP olarak modellenmiştir. Ayrıca, *"Açık Döngülü Geri Besleme Kontrolü (Open-Loop Feedback Control)"*, *"Sınırlandırılmış İleri Bakışlı Kontrol (Limited Look-Ahead Control)"* gibi farklı pratik kontrol planları önerilmiş ve elde edilen planlar, denetçilerin pratikte sıklıkla kullandığı *"Belirli Denklik Kontrolü (Certainty Equivalent Control)"*den elde edilen planla kıyaslanmıştır.

Kambhamettu (2000) tezinde, kısmi olarak gözlemlenebilir, sabit olmayan talep içeren çevrelerde bir stok kontrol problemini POMDP olarak modellemiş ve talebin

ayrık parametre olasılık dağılımı ile belirlendiği durumlarda, parametre tahminlerini ele almıştır. Bu çalışmada, POMDP'deki durumların poisson ya da negatif binom dağılımı ile elde edildiği kabul edilmektedir.

Diğer bir araştırmada, gelecek talep, geçmiş satışlara bakılarak tahmin edilmiştir. (Fisher *ve diğ.* 2000) Bu, perakande endüstrisinde ele alınmış ve tedarik zincirinde yol açabileceği, potansiyel negatif etkileri gösterilmiştir (Kamath ve Pakkala, 2002).

Daha sonraki yıllarda, Ortiz *ve diğ.* (2006), yaptıkları bir çalışmada, tek ürünlü, kayıp satışın olduğu tedarik zinciri sistemini POMDP kullanarak modellemişlerdir. Bu modelde, tek bir KV, her bir adımda sipariş miktarına karar vermektedir. Talep süreci, bir periyottaki talep dağılımı bir önceki periyottaki talep seviyesine bağlı olacak şekilde, bir Markov zinciri ile modellenmiştir. Stok seviyeleri tam olarak gözlemlenebilmektedir. Bu çalışmada, optimal plan ve optimal beklenen toplam karın belirlenmesi için bir algoritma geliştirilmiştir. Aynı zamanda, üç sezgisel algoritma önerilmiştir. Modelde, sistem karlılığının, en büyük beklenen artışındaki sınırı belirlemek için, iki uç durum kullanılmıştır. Bunlardan bir tanesi tam olarak gözlemlenebilirlik, diğeri ise sadece satışların tam olarak gözlemlenebildiği durumdur.

3.5 Çok Karar Vericili Tedarik Zinciri

Yıllardan beri, Moyaux *ve diğ.* (2003), yaptıkları çalışmalarda, ardışık karar verme metotlarının tedarik zincirinde kullanımından ziyade, çok KV'li tedarik zinciri sistemlerini incelemişlerdir. İlk olarak yapılan çalışmalarda, bir orman endüstrisinin, bir tedarik zincirindeki dinamiklerin gösterimi için yaygın olarak kullanılan Bira Oyunu'na uyarlanması üzerine çalışılmıştır. Uyarlanan bu oyun, Quebec Tahta Tedarik Oyunu'dur. Bu oyunun, bir çizelge programı üzerinde benzetimi yapılmıştır. Burada, şirketlerin ürünleri nasıl sipariş ettiği, ürettiği ve stokladığı gösterilmiştir. 2004'teki diğer çalışmalarında, aynı oyun üzerinde, her bir şirketin bir KV'yi temsil ettiği ve 3 sıralama şemasından birini kullandığı bir tedarik zinciri simülasyonu yapılmış ve tedarik zincirinin minimum maliyetini bulmak için iki Nash denkliği elde edilmiştir (Moyaux *ve diğ.* 2004). Tedarik zinciri ve çok KV'li sistemler ele alınarak, bu iki alan "Çok KV'li Tedarik Zinciri" adı altında birleştirilmiştir (Moyaux *ve diğ.* 2006a). Tedarik zincirinde karşılaşılan problemler ve bu problemlerin çözümüne yönelik teknikler ele alınmış ve çok KV'li sistemlere geniş olarak yer verilmiştir.

Aynı yıl yayınladıkları diğer iki makalede ise, Moyaux *ve diğ.* (2006b; 2006c) deneysel oyun teorisini kullanmışlardır. Her bir KV'nin faydası iki kısımda ele alınmıştır. İlk kısım, KV'nin direkt faydasını, ikinci kısım ise KV'nin sosyal bilincini yansıtır. Bu yaklaşım, bir tedarik zinciri uygulaması kullanılarak örneklendirilmiştir. Son yıllarda ise, yine aynı oyun üzerine kurulmuş iki simülatör karşılaştırılmıştır (Moyaux *ve diğ.* 2008).

Çok KV'li tedarik zincirinde ardışık karar verme modelleri üzerinde yapılan çalışmalarda ise, tedarik zinciri bireyleri arasındaki işbirliği ele alınmıştır. Genellikle bu, bireyler arasındaki bilgi paylaşımı ile gerçekleşmektedir. Bu da, piyasadaki rekabet ortamında, tedarik zinciri bireylerinin oluşturdukları sinerjiyle, rakiplerine karşı üstünlük elde etmelerini sağlar.

3.6 Tedarik Zincirinde İletişimin Önemi ve Paylaşılan Bilginin Değerini Ölçme İle İlgili Yapılan Çalışmalar

Tedarik zinciri bireyleri arasında bilgi paylaşımı çok önemlidir. Eğer tedarik zincirindeki bilgi, tedarik zinciri bireyleri tarafından paylaşılmıyorsa, tedarik zinciri içerisindeki mevcut talep bilgisinde aksamalar olur, talepteki dalgalanma artar ve bu da gereksiz maliyetlere neden olur. İşte talep çeşitliliğinin bu etkisi "Bullwhip Etkisi" olarak adlandırılır. (Lee *ve diğ.* 1997). Bir çeşitlilik olarak, bullwhip etkisi siparişlerin standart sapması olarak ifade edilir. Bullwhip etkisinin birçok sonucu vardır (Moyaux *ve diğ.* 2006a). Bunlar;

1. *Yüksek stok seviyeleri:* Tedarik zincirindeki tüm bireyler, talepteki dalgalanmanın yüksek olmasından dolayı oluşan belirsizlik yüzünden, elde yüksek miktarlarda stok bulundurmak zorunda kalmaktadırlar.
2. *Tedarik zincirindeki hareketliliğin azalması:* Stok seviyeleri yüksek olduğu için, son müşteri tarafından talep edilen yeni ürünler satılmadan önce, stoktaki ürünler satılmalıdır. Bu da, son müşteri talebinin devamlılığında bir durağanlığa neden olur. Ayrıca, bullwhip etkisi yüzünden ortaya çıkan talep belirsizlikleri, son müşteri tarafından hangi ürünün talep edileceğinin tahminini daha da zorlaştırmaktadır.
3. *Müşteri servis seviyesinde düşüş:* Talep çeşitlilikleri, elde bulundurmama durumuna neden olabilir. Böylece, satılacak ürün elde olmadığından dolayı

müşteri servis hizmeti sağlanamaz. Bullwhip etkisinin bu son iki sonucu belirsizlik altında karar vermeyle alakalıdır ve şunlara neden olur:

4. *Verimsiz taşıma:* Taşıma planlaması, talebin belirsiz olduğu durumda daha zor hale gelmektedir.
5. *Üretim çizelgelerinde aksama:* Taşımaya benzer olarak, üretim planlama da, talep belirsizliklerinden dolayı zorlaşmaktadır.

Bullwhip etkisinin azaltılmasında (Lee *ve diğ.* 1997; 2000; Cachon ve Fisher 2000) ve tedarik zinciri maliyetinin düşürülmesinde (Gavirneni *ve diğ.* 1999; Swaminathan *ve diğ.* 1997) bilgi paylaşımı çok önemlidir. Fakat, kurumun iç bilgi sistemi maliyetleri çok yüksek olduğunda, bu, avantajlı olmayabilir (Swaminathan *ve diğ.* 1997; Cohen, 2000). Bilgi paylaşımı kapsamında önemli olan, paylaşılacak üretim bilgisinin belirlenmesi ve tedarik zincirindeki ortak faydaların enbüyüklenmesindeki paylaşım şeklidir (Huang *ve diğ.* 2003). Fuller *ve diğ.* (1993), yıllık gıda endüstrisinin %25-%33'ünün, bullwhip etkisinden dolayı elde bulundurma maliyetlerine sahip olduğunu göstermiştir. Çalışmada, bilgi paylaşım zincirine uyarlanmış bir planın, maliyetleri azaltacağı ve tedarik zincirindeki bireylerin karlarını arttıracağı gösterilmiştir. Moldova 'da üzerine çalışılan bir vaka, tedarik zincirindeki aksamaların, tedarik zincirindeki bireylere, piyasada başarısızlık olarak geri döndüğünü göstermiştir. (Gorton *ve diğ.* 2006).

Talep ya da kaynaktaki herhangi bir aksamanın daha çabuk onarılmasında da, tedarik zincirindeki bireylerin birbirleriyle olan iletişimi çok önemlidir (Li *ve diğ.* 2006; Russell ve Saldanha, 2003; Sheffi, 2001). Stok seviyesi bilgilerinin paylaşılmasının sadece günlük tedarik zinciri işlemlerinde değil, üretim aksamalarının çabuk onarılmasında da avantaj sağladığı görülmüştür. King *ve diğ.* (2005)'nin yapmış olduğu bir çalışmada da, tedarik zinciri bireyleri arasındaki bilgi paylaşımının değerini belirleyecek analitik bir model geliştirilmiştir.

Wei (2005), doktora tezinde, iki aşamalı ve üç aşamalı tedarik zincirinde bilgi paylaşımı üzerine çalışmıştır. Burada, tedarik zincirindeki müşteri talep dağılımının bilindiği kabul edilmektedir. Paylaşılan bilgi, her bir tedarik zinciri bireyinin stok seviyesidir. Paylaşılan bilginin değerini ölçmek için, tedarik zinciri farklı bilgiler altında ele alınmıştır. Tedarik zincirini modellemede MDP yaklaşımı kullanılmış ve her bir farklı durum için optimal plan elde edilmiştir. Bunlar kıyaslanarak, paylaşılan

bilginin değeri hesaplanmıştır. Denenen farklı bilgi paylaşım modelleri, tam bilgi paylaşımının, kısmi bilgi paylaşımının olduğu ve hiç bilgi paylaşımının olmadığı durumları içerir. Tam bilgi paylaşımı durumunda, tedarik zinciri problemi, Howard (1960)'ın geliştirdiği plan iterasyonu metoduyla kolayca çözülebilen tek KV'li MDP şeklinde modellenmiştir. Kısmi bilgi paylaşımı olan veya hiç bilgi paylaşımı olmayan durumlar için de tedarik zinciri problemi DEC-ROMDP (DEC-POMDP'nin bir çeşididir. Farklı olarak burada, her bir adımda, tek bir gözlem elde edilmektedir) olarak modellenmiştir. Bu çalışmaya benzer bir çalışma da, Mccoy (2007)'un tezinde yer almıştır. Bu tezde, internete dayalı birçok ticaret işleminde paylaşılan bilginin ani olarak kesilmesinin, tek ürünlü, iki aşamalı, bilgi paylaşımlı tedarik zincirinin beklenen kazancı üzerindeki etkileri araştırılmıştır. Tedarikçi (supplier) ve perakendeci (retailer) bileşenlerini içeren bir tedarik zinciri, tam bilgi paylaşımı durumunda MDP olarak modellenmiştir. Modeldeki sistem durumu, her bir tedarik zinciri bireyinin yerel durumlarından (stok seviyeleri) oluşan iki boyutlu bir vektördür. Bilgi tam olarak paylaşılmadığında sistem, Wei'nin çalışmasında olduğu gibi, POMDP olarak modellenmiştir. Sonuçlar, stok bilgisini paylaşan bir tedarikçinin ve perakendecinin, paylaşmayanlara göre daha yüksek sonuçlar elde ettiği yönündedir. Tedarik zincirinde, tedarikçi ve perakendeci arasındaki internet iletişiminin beklenmeyen bir durumdan dolayı kesildiği ve bu yüzden bu süre boyunca birbirlerinin stok seviyelerini göremedikleri durumlarda, tedarikçi ve perakendeci, internet bağlantısı tekrar sağlanana kadar, kararlarını, bilgi paylaşımı olmayan durumdaki acil durum planlarına göre verebilirler.

4. ÖNERİLEN MODEL

Bu tez kapsamında, tek ürünlü bir stok kontrol problemi için, bekleyen siparişlerin olmadığı, karşılanamayan taleplerin kayıp olarak ele alındığı, talebin sabit olmayan ve kısmi gözlemlenebilir olduğu bir kısmi gözlemlenebilir Markov karar süreci (Partially Observable Markov Decision Process, POMDP) modeli geliştirilmiştir. Eylül, 2009 – Mart, 2010 tarihleri arasında, University of Massachusetts, Amherst (UMass), Computer Science bölümünde Prof. Scholomo Zilberstein danışmanlığındaki Resource Bounded Reasoning Araştırma Grubu'nun çalışmaları izlenerek bu modeller ve çözümlerine yönelik ayrıntılı bilgi elde edilmiştir. Model, bu grup içerisinde çalışmalarına devam eden doktora son sınıf öğrencisi Christopher Amato'nun yardımlarıyla, Doç. Dr. Y. İlker Topçu danışmanlığında geliştirilmiştir. Modelin ayrıntıları aşağıdaki bölümlerde anlatılacaktır.

4.1 Model Parametreleri

$\boldsymbol{T}$: Sonlu zaman ufku uzunluğu kullanıldığı durumlardaki periyot sayısı

$\boldsymbol{C}$: Perakendecinin stok kapasitesi

$\boldsymbol{M}$: Maksimum müşteri talebi $0 \leq M \leq C$

$\boldsymbol{h}$: Perakendecinin birim elde bulundurma maliyeti

$\boldsymbol{p}$: Perakendecinin birim elde bulundurmama maliyeti

$\boldsymbol{u}$: Perakendecinin birim satın alma maliyeti

$\boldsymbol{s}$: Perakendecinin birim sipariş maliyeti

$\boldsymbol{I_t}$: Perakendecinin t periyodu sonundaki stok seviyesi

$\boldsymbol{\pi_t}$: t periyodundaki müşteri talep dağılımı, $\pi_t(\lambda_t) = \{\pi_{ti}(\lambda_t)\}$ şeklindedir ve $\lambda_t = k$ ortalama talep ifadesi, $i = 0, \ldots, M$ olacak şekilde $\pi_{ti}(\lambda_t) = f(d_t = i;\ \lambda_t = k)$ poisson dağılımını ifade etmektedir.

$\boldsymbol{\Omega_t}$: $\Omega_t = i$, perakendecinin, $t-1$ periyodunun sonundaki stok seviyesine bakarak, bu t periyodunda gelecek olan talep dağılımının $\lambda_t = i$ olduğunu düşüneceği durumu ifade etmektedir.

$\boldsymbol{d_t}$: t periyodunda, bir π_t poisson dağılımına göre meydana gelen müşteri talebi $0 \leq d_t \leq M$

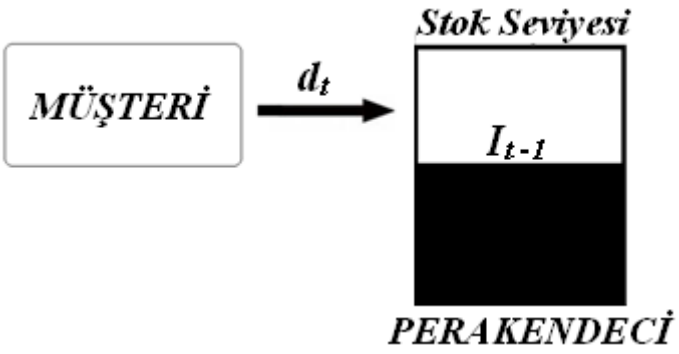
$\boldsymbol{a_t}$**:** Perakendecinin t periyodundaki sipariş miktarı

4.2 Model Varsayımları

- Tedarik zinciri sisteminde, bir tedarikçi ve bir perakendeci bulunmaktadır.
- Sistemde gecikmelerin olmadığı varsayımı yapılmıştır. Yani, tüm siparişler zamanında teslim edilir ya da teslim alınır.
- Sistemde bekleyen sipariş yoktur. Karşılanamayan talepler, kayıp olarak değerlendirilir.
- Şekil 4.1'de görüldüğü gibi, bir t periyodunun başında, perakendeci bir önceki periyodun sonunda elde edilen I_{t-1} stok seviyesine sahiptir. Perakendecinin stok seviyesi bilgisini, müşteri ile paylaştığı varsayılmaktadır.

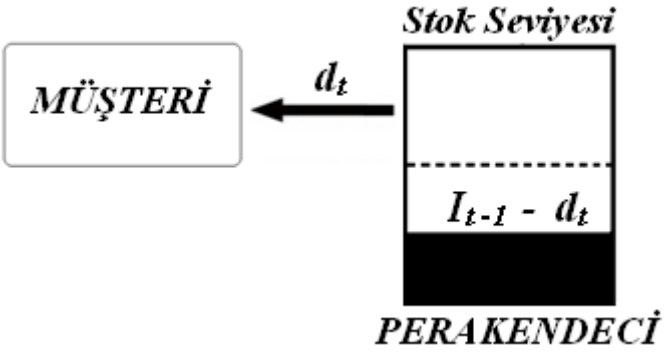
Şekil 4.1 : Perakendecinin t periyodundaki başlangıç stok seviyesi

- İlk olarak, perakendeci, Şekil 4.2'de görüldüğü gibi, mevcut dönem talebi d_t'yi gözlemler. Gözlemlenen bu talep, perakendecinin stoğundan karşılanacaktır. d_t, mevcut dönem ortalama talebi λ_t'den elde edilen π_t dağılımındaki olasılıklara bağlı olarak gözlemlenmektedir.



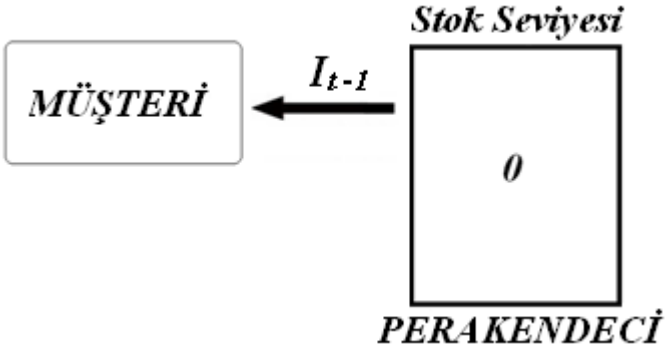
Şekil 4.2 : Perakendecinin mevcut dönem talebini gözlemlemesi

1. Eğer $I_{t-1} \geq d_t$ ise, perakendeci, stoğunda bu talebi karşılayacak miktarda ürün bulundurduğundan dolayı, Şekil 4.3'te görüldüğü gibi, d_t miktarda ürünü müşteriye gönderir.



Şekil 4.3 : Perakendecinin stoğundan müşteri talebini karşılaması

2. Eğer $I_{t-1} < d_t$ ise, perakendeci, stoğunda bu talebi karşılayacak miktarda ürün bulunduramadığından dolayı, Şekil 4.4'te görüldüğü gibi, I_{t-1} miktarda ürünü, yani stoğundaki tüm miktarı müşteriye gönderir.



Şekil 4.4 : Perakendecinin, talebi karşılayacak yeterli stoğu bulunmaması durumu

- Perakendeci, daha sonra, Şekil 4.5'te görüldüğü gibi, tedarikçiden talep edeceği sipariş miktarını belirlemek zorundadır. Fakat bir sonraki periyottaki müşteri talep dağılımını bilmemektedir. Bu modelde müşteri talebinin genellikle yüksek olduğu, bu yüzden perakendecinin stok seviyesinin yüksek olduğu durumlarda, stok seviyesi bilgisi müşteri ile paylaşıldığı için, daha yüksek talebin gelmesinin muhtemel olacağı varsayımı yapılmıştır.
- Perakendecinin stok seviyesi, $0 \leq I_t \leq C, \quad t = 0,1,\dots,T$ olacak şekilde sınırlı olduğundan, belirleyeceği sipariş miktarı, stok kapasitesini aşmayacak şekilde, yani; $0 \leq a_t \leq C - max(0, I_{t-1} - d_t), \quad 0,1,\dots,T$, ifadesi gibi olmalıdır.

Şekil 4.5 : Perakendecinin sipariş miktarına karar vermesi

- Perakendeci t periyodunda, sipariş edeceği miktara karar verirken her bir mümkün $\lambda_{t+1} \in S$ için elde bulundurma ve bulundurmama maliyetlerini göz önüne almak zorundadır.

  Perakendecinin bir λ_{t+1} durumu için elde bulundurma maliyeti (4.1)'deki gibi hesaplanır.

$$\sum_{i=0}^{M} \pi_{(t+1)i}(\lambda_{t+1}) * (h * \max(0, \max(0, I_{t-1} - d_t) + a_t - i)) \qquad \textbf{(4.1)}$$

  Perakendecinin bir λ_{t+1} durumu için elde bulundurmama maliyeti ise (4.2)'deki gibi hesaplanır.

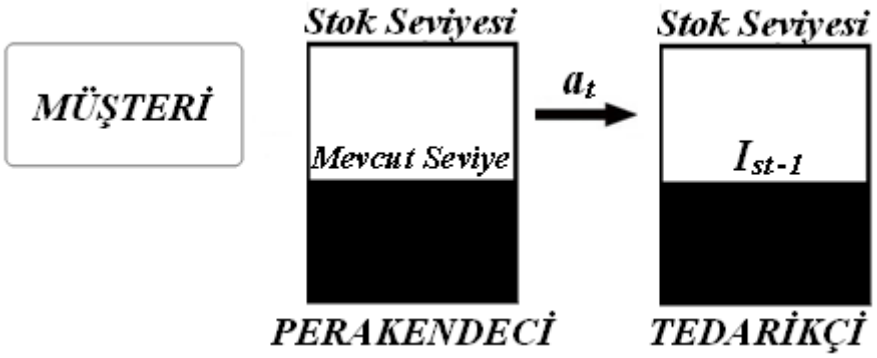
$$\sum_{i=0}^{M} \pi_{(t+1)i}(\lambda_{t+1}) * (-p * \min(0, \max(0, I_{t-1} - d_t) + a_t - i)) \qquad \textbf{(4.2)}$$

- Perakendeci, sipariş miktarına karar verirken, aynı zamanda (4.3)'te hesaplanan satın alma maliyeti ve (4.4)'te hesaplanan sipariş maliyetini de gözönünde bulundurmak zorundadır.

$$u * a_t \quad \textbf{(4.3)}$$

$$s * a_t \quad \textbf{(4.4)}$$

- Bir karara vardıktan sonra perakendeci, Şekil 4.6'daki gibi, bu sipariş miktarını tedarikçiye bildirecektir.



Şekil 4.6 : Perakendecinin siparişini tedarikçiye iletmesi

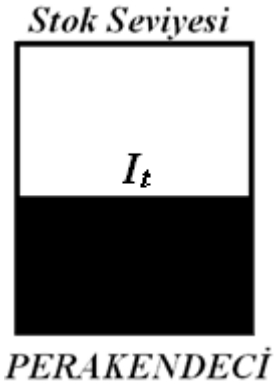
- Tedarikçi perakendecinin talebini, stoğundan karşılamaktadır. Bu modelde, perakendecinin sipariş miktarını her zaman, I_{st-1} tedarikçinin t periyodundaki başlangıç stok seviyesi olmak üzere, $\min(a_t, I_{st-1})$ olacak şekilde vereceği varsayımı yapılmıştır. Yani perakendeci, hiç bir şekilde tedarikçinin stok seviyesinden daha fazla bir sipariş miktarı belirlemeyecektir. Bu yüzden, model hesaplamalarında $\min(a_t, I_{st-1})$ ifadesi yerine yalnızca a_t ifadesi kullanılmıştır. Şekil 4.7'de görüldüğü gibi, mevcut periyodun sonunda, perakendecinin talep ettiği miktar kendisine gönderilmekte ve perakendeci elde ettiği bu miktarı stoğuna eklemektedir.

Şekil 4.7 : Tedarikçinin, perakendecinin talep ettiği miktarı göndermesi

- Böylece, perakendecinin stok seviyesi (4.5)'te görüldüğü gibi güncellenir.

$$I_t = \max(0, I_{t-1} - d_t) + a_t \quad \textbf{(4.5)}$$

- Daha sonra, sistem, Şekil 4.8'de görüldüğü gibi, perakendecinin yeni stok seviyesi ile, bir sonraki periyoda geçecektir.



Şekil 4.8 : Perakendecinin bir sonraki periyodun başındaki yeni stok seviyesi

4.3 Önerilen POMDP Modeli

Bu tezde önerilen POMDP modelinin bileşenleri aşağıdaki şekildedir;

- *Durum:* t periyodu için sistemin durumu, bir sonraki periyodun ortalama talebini, mevcut periyotta oluşan talebi ve perakendecinin başlangıç stok seviyesini (yani bir önceki periyodun sonundaki stok seviyesi) içeren bir vektördür; $s = (\lambda_{t+1}, d_t, I_{t-1})$

  Sistemde, bir durum içerisindeki stok seviyesi I_{t-1} ve mevcut dönemin talebi d_t tam olarak gözlemlenebilirken, bir sonraki dönemin ortalama talebi λ_{t+1} kısmi olarak gözlemlenebilmektedir.

- *Eylem:* Sistem, tek KV'li bir sistem olduğundan KV olarak yalnızca perakendeci ele alınacaktır. Perakendeci eylem olarak, sipariş miktarı a_t'ye karar verecektir.

- *Gözlem:* Perakendeci, gözlem olarak bir sonraki periyotta gelecek ortalama talep hakkındaki düşüncesini, mevcut dönem talebini ve mevcut dönemdeki başlangıç stok seviyesini gözlemler; $o = (\Omega_{t+1}, d_t, I_{t-1})$

- *Geçiş Fonksiyonu:*

  $P(s'\backslash s, a_t),\ s = (\ \lambda_{t+1}, d_t, I_{t-1}),\ s' = (\lambda_{t+2}, d_{t+1}, I_t)\ \ t = 0,1, \dots, T$

  Geçiş matrisi oluşturulurken şu özellikler dikkate alınmalıdır;

  - *Yeni geçilecek olan s' durumunun içerdiği ortalama talep λ_{t+2}'ye geçiş olasılıklarının hesaplanması*: Mevcut bir durumdan bir s' durumuna geçerken, yeni durumda her bir ortalama talebin elde edilme olasılığı birbirine eşittir.

  - *Yeni geçilecek olan s' durumunun içerdiği mevcut dönem talebi d_{t+1}'e geçiş olasılıklarının hesaplanması*: Geçilecek olan s' durumunun içerdiği mevcut dönem talebi d_{t+1}, s durumunun içerdiği ortalama talep λ_{t+1}'in oluşturduğu dağılımdan elde edilmiştir. Bu yüzden, s durumundaki λ_{t+1}'in oluşturduğu dağılımın içerdiği d_{t+1} olasılığı, bu talebi içeren yeni durum s' ne geçiş olasılığını verir.

  - *Yeni geçilecek olan s' durumunun içerdiği stok seviyesi I_t'ye geçiş olasılıklarının hesaplanması*: s durumunun içerdiği bileşenler ve mevcut eylem a_t kullanılarak (4.5)'te görüldüğü gibi stok seviyesi güncellemesi yapılır ve I_t elde edilir. Bir sonraki adımda geçilecek olan s' durumunda, yeni stok seviyesi, mutlaka elde edilen I_t'nin değerini içermelidir. O halde, sadece bu I_t değerini içerecek olan durumlara geçiş olabilecektir (olasılık:1). Bunun dışındaki durumlara geçiş olasılığı her zaman 0'dır.

  - *Yeni geçilecek olan s' durumunun geçiş olasılığının hesaplanması*: Tüm bu elde edilen durum bileşenlerinin olasılıkları çarpılarak yeni duruma geçiş olasılığı elde edilir. Fakat bu olasılıkların satır toplamlarının 1 olması gerektiğinden her bir satırdaki elemanlar, satır toplamına bölünerek matris normalize edilir.

- *Gözlem Fonksiyonu:*

  $O(o\backslash s', a_t), o = (\ \Omega_{t+2}, d_{t+1}, I_t),\ s' = (\ \lambda_{t+2}, d_{t+1}, I_t)\ \ t = 0,1, \dots, T$

  Gözlem matrisi oluşturulurken şu özellikler dikkate alınmalıdır;

  - Yeni geçilen s' durumundaki mevcut dönem talebi d_{t+1} ve stok seviyesi I_t tam olarak gözlemlenebilir olduğundan, bu durumda elde edilecek olan o gözleminde, mevcut dönem talebi d_{t+1} ve stok seviyesi I_t mutlaka bu s' durumunun içerdiği değerlerle aynı olmalıdır. Bunun dışındaki gözlemlerin elde edilme olasılığı 0'dır.

  - Müşteri talebinin genellikle yüksek olduğu ve perakendecinin stok seviyesi bilgisini müşteri ile paylaştığı varsayılmıştır. Bu yüzden, yeni geçilecek olan s' durumundan hesaplanacak stok seviyesi I_t yüksek ise, ortalama talep λ_{t+1}'in büyük olduğunu düşünme olasılığı fazla olacaktır. Yani, büyük Ω_{t+1} içeren gözlemi elde etme olasılığı daha fazla olacaktır. Aynı şekilde, yeni geçilecek olan s' durumundan hesaplanacak stok seviyesi I_t küçük ise, ortalama talep λ_{t+1}'in küçük olduğunu düşünme olasılığı yine fazla olacaktır. Yani, küçük Ω_{t+1} içeren gözlemi elde etme olasılığı daha fazla olacaktır. Bunun dışındaki durumlara geçiş olasılıkları ise daha küçüktür.

- *Stok Seviyesi Güncellemesi :* Her bir periyodun başında, perakendecinin stok seviyesi daha önceden de bahsedildiği gibi, (4.5) denklemi kullanılarak güncellenir.

- *Perakendecinin Tedarik Zincirindeki Toplam Maliyeti:* Perakendecinin toplam maliyeti, (4.6)'da gösterildiği gibi, (4.1), (4.2), (4.3) ve (4.4)'te belirtilen maliyetlerin toplamıdır.

  $$Cost(s, a_t) =$$

  $$\sum_{k=0}^{M} \pi_{(t+1)k}(\lambda_{t+1}) * (h * \max(0, \max(0, I_{t-1} - d_t) + a_t) - k) - p$$
  $$* \min(0, \max(0, I_{t-1} - d_t) + a_t - k) + u * a_t + s * a_t)$$

  (4.6)

- *Ödül Fonksiyonu:* Herhangi bir s durumu için, tedarik zincirinin anlık ödülü (4.7)'de gösterildiği gibidir.

$$R(s, a_t) = \min Cost\ (s, a_t) \quad \textbf{(4.7)}$$

4.4 Deneysel Uygulama

Modelin uygulaması, bilgisayar ortamında, Trey Smith'in geliştirmiş olduğu özel POMDP çözücü ile yapılmıştır (Smith, 2007). Bu çözücü Linux işletim sisteminde çalıştırılmıştır. Burada sonsuz ufuklu zaman uzunluğu kullanılmıştır. Bu çözücü girdi olarak .pomdp uzantılı bir dosya almaktadır. Önerilen modelde girdi ve çıktı dosyaları çok büyük çapta olduğundan, bu dosyaların açıklamaları, kolaylık bakımından örnek kaplan problemi dosyaları üzerinden yapılmıştır. Kaplan problemi için örnek girdi dosyası Şekil 4.9'da görüldüğü gibidir. Discount terimi, sonsuz ufuk uzunluğu kullanıldığı için belirlenmesi gereken değerdir. İkinci satırda girilen "reward" terimi, eylemlerin, verilen ödül değerleri enbüyüklenecek şekilde olduğunu göstermektedir. Durumlar, eylemler ve gözlemler aşağıdaki şekilde tek tek yazılabileceği gibi, çok fazla sayıda durum, eylem ya da gözlem olması halinde sadece kaç adet oldukları da yazılabilir. "start" ile başlayan satır başlangıç düşünülen durumunu ifade etmektedir. T ile başlayan matrisler, her bir eylem için geçiş matrislerini vermektedir. Bu üç matris kaplan problemi örneğinde sırasıyla Çizelge 2.1, Çizelge 2.2 ve Çizelge 2.3'te gösterilmiştir. O ile başlayan matrisler ise her bir eylem için gözlem matrislerini vermektedir. Bu matrisler de kaplan probleminde sırasıyla Çizelge 2.4, Çizelge 2.5 ve Çizelge 2.6'da gösterilmiştir. Ödül değerlerinde kullanılan "*" ifadesi durum, gözlem ya da eylemlerin hangisinin yerine kullanıldıysa bunların tümü için o değerin geçerli olduğunu göstermektedir. Örneğin dinleme eylemi, tüm durumlar, tüm yeni geçilecek durumlar ve tüm gözlemler için 1 puan maliyet getirecektir.

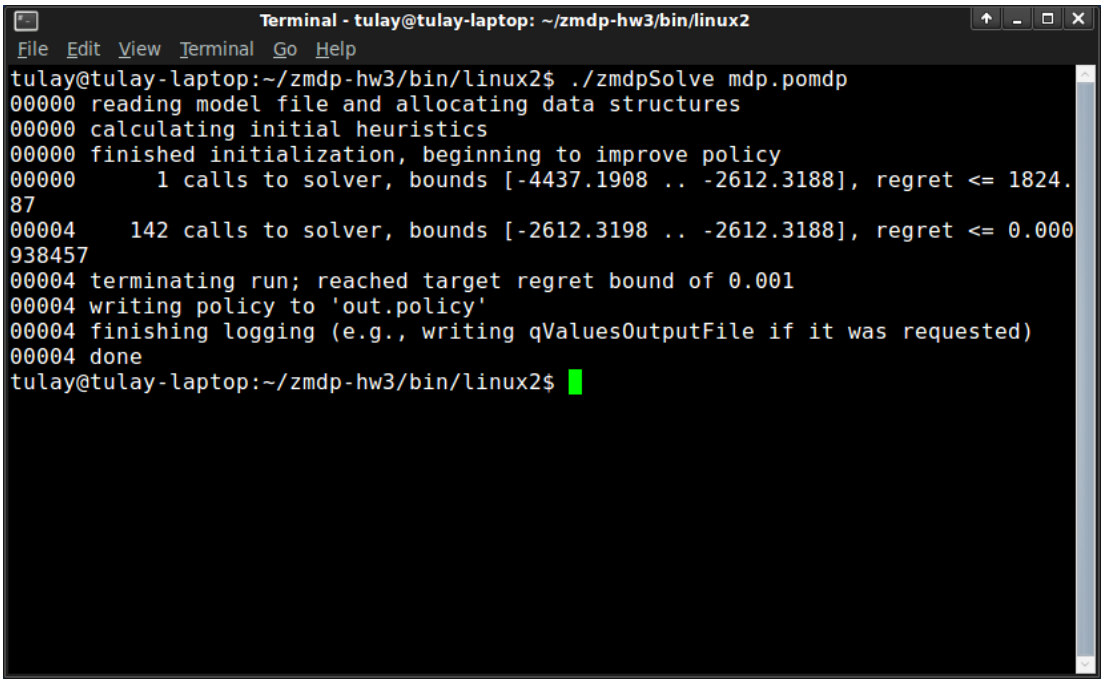
```
discount: 0.75
values: reward
states: tiger-left tiger-right
actions: listen open-left open-right
observations: tiger-left tiger-right

start: 0.5 0.5

T:listen
identity

T:open-left
uniform

T:open-right
uniform

O:listen
0.85 0.15
0.15 0.85

O:open-left
uniform

O:open-right
uniform

R:listen : * : * : * -1

R:open-left : tiger-left : * : * -100

R:open-left : tiger-right : * : * 10

R:open-right : tiger-left : * : * 10

R:open-right : tiger-right : * : * -100
```

Şekil 4.9 : Kaplan problemi için .pomdp uzantılı örnek girdi dosyası (Cassandra, 2010)

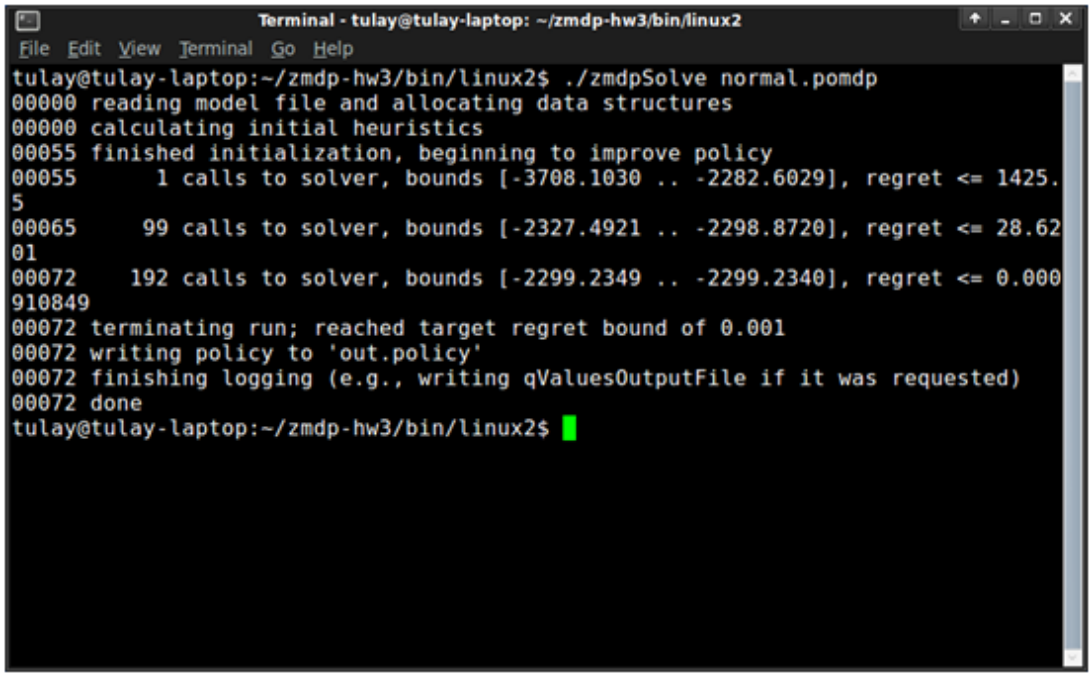
Önerilen model, iki farklı veri seti üzerinde uygulanmıştır. Birinci uygulamada, kısmi olarak gözlemlenebilen ortalama talep değeri tek bir değer olarak ele alınmış ve problem MDP şekline dönüştürülmüştür. İkinci veri setinde ise, bu parametre için iki farklı değer ele alınarak bir POMDP örneği oluşturulmuştur.

Önerilen model için oluşturulan girdi dosyalarındaki geçiş ve gözlem matrisleri, daha önceki bölümde bahsedilen geçiş ve gözlem matrisi oluşturma kurallarına göre yapılmalıdır. Ayrıca (4.6) kullanılarak her bir durum ve eylemin oluşturduğu maliyetlerin tek tek hesaplanarak bu dosyaya yazdırılması gerekmektedir. Bunun yanında, her bir durum için göz önüne alınacak olan eylem sayısı farklıdır. Çünkü

perakendecinin stok seviyesi, $0 \leq I_t \leq C, \quad t = 0,1,\ldots,T$ olacak şekilde sınırlı olduğundan, belirleyeceği sipariş miktarı, stok kapasitesini aşmayacak şekilde, yani; $0 \leq a_t \leq C - max(0, I_t - d_{t-1}), \quad 0,1,\ldots,T$, ifadesi gibi olmalıdır. Her bir aşamada farklı sayıda eylemin göz önüne alınması durumu çözücü tarafından gerçekleştirilemediğinden dolayı, göz önüne alınmaması gereken eylem-durum kombinasyonlarındaki maliyetlere çok yüksek değerler atanarak bunların elenmesi sağlanmıştır. Tüm bunların gerçekleştirilmesi için C programlama dilinde, girdi dosyası oluşturan bir kod yazılmıştır. Her iki veri kümesi için, girdi dosyası oluşturan kodlar, ekler kısmında EK A.1 ve EK A.2'de yer almaktadır. Bu programların oluşturduğu girdi dosyalarının boyutu çok büyük olduğundan, bu tezde bunlara yer verilmemiştir. Elde edilen dosyalar kullanılarak çözücü çalıştırılmıştır. Örnek ekran görüntüsü önerilen model örnekleri için ise kısmi olarak Şekil 4.10 ve Şekil 4.11'de görüldüğü gibidir.



Şekil 4.10 : POMDP çözücünün önerilen modelin MDP örneğini çalıştırması



Şekil 4.11 : POMDP çözücünün önerilen modelin POMDP örneğini çalıştırması

Şekil 4.10 ve Şekil 4.11'de görüldüğü gibi, çözücü HSVI algoritmasını kullanmaktadır. Burada ilk olarak, pişmanlık değeri belli bir ε değerinden küçük bir plan ele alınır. Bu plan, alt ve üst sınırları arasındaki fark, bu ε değerinden küçük olacak şekilde geliştirilir. Daha önceden de bahsedildiği gibi, HSVI, bu ε değerini her adımda düzeltir. Son olarak elde edilen planın alt ve üst sınırları arasındaki fark, o adımdaki ε'dan küçük olduğunda algoritma sona erer ve optimale yakınsayan bir plan elde edilir.

Önerilen modelin deneysel uygulamasında kullanılan parametre ve maliyet değerlerleri Çizelge 4.1'de gösterilmektedir. Burada kullanılan maliyet değerleri, literatürdeki bir örnekten alınmıştır (Wei, 2005).

Çizelge 4.1 : Modelde kullanılan parametre ve maliyet değerleri

	MDP	**POMDP**
T	Sonsuz	Sonsuz
C	9	9
M	9	9
λ_t Değerleri	5	3,8
Ω_t Değerleri	5	3,8
d_t Değerleri	0,1,...,9	0,1,...,9
I_t Değerleri	0,1,...,9	0,1,...,9
a_t Değerleri	0,1,...,9	0,1,...,9
Toplam durum sayısı	100	200
Toplam gözlem sayısı	100	200
h	1	1
p	100	100
u	50	50
s	0	0

Çizelge 4.1'de de görüldüğü gibi, önerilen modelde durum sayısı, λ_t'nin, d_t'nin ve I_t'nin her birinin alabileceği değer sayılarının çarpımları kadar olacaktır. Aynı şekilde, gözlem sayısı da Ω_t'nin, d_t'nin ve I_t'nin her birinin alabileceği değer sayılarının çarpımları kadar olacaktır.

Buna ek olarak, başlangıç düşünülen durumu için, tüm durumların olasılıkları eşit kabul edilmiştir.

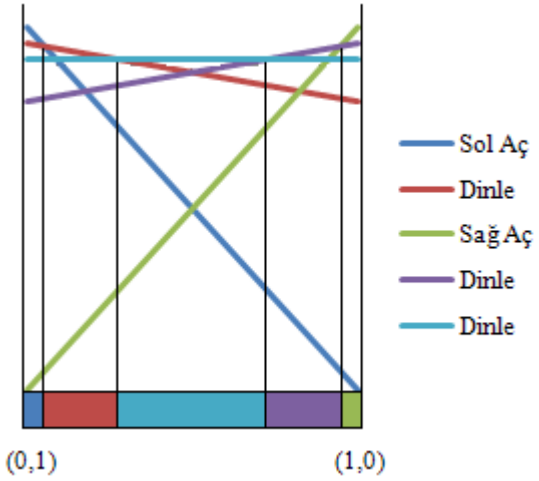
Çözücü dosyayı çözdükten sonra, çıktı olarak "out.policy" adında bir dosya elde edilir. Bu dosya içerisinde, uzun dönemde perakendecinin takip edeceği, optimale yakınsayan stok kontrol planı bulunmaktadır. Şekil 4.12'de kaplan problemi için örnek çıktı dosyası görülmektedir. Önerilen model örnekleri için elde edilen çıktı dosyaları ekler bölümünde EK A.3 ve EK A.4'te yer almaktadır.

```
{
  policyType => "MaxPlanesLowerBound",
  numPlanes => 5,
  planes => [
    {
      action => 1,
      numEntries => 2,
      entries => [
        0, -98.5502,
        1, 11.4498
      ]
    },
    {
      action => 0,
      numEntries => 2,
      entries => [
        0, -10.8547,
        1, 6.51675
      ]
    },
    {
      action => 2,
      numEntries => 2,
      entries => [
        0, 11.45,
        1, -98.55
      ]
    },
    {
      action => 0,
      numEntries => 2,
      entries => [
        0, 6.51684,
        1, -10.8544
      ]
    },
    {
      action => 0,
      numEntries => 2,
      entries => [
        0, 1.93334,
        1, 1.93331
      ]
    }
  ]
}
```

Şekil 4.12 : Kaplan problemi için out.policy adındaki örnek çıktı dosyası

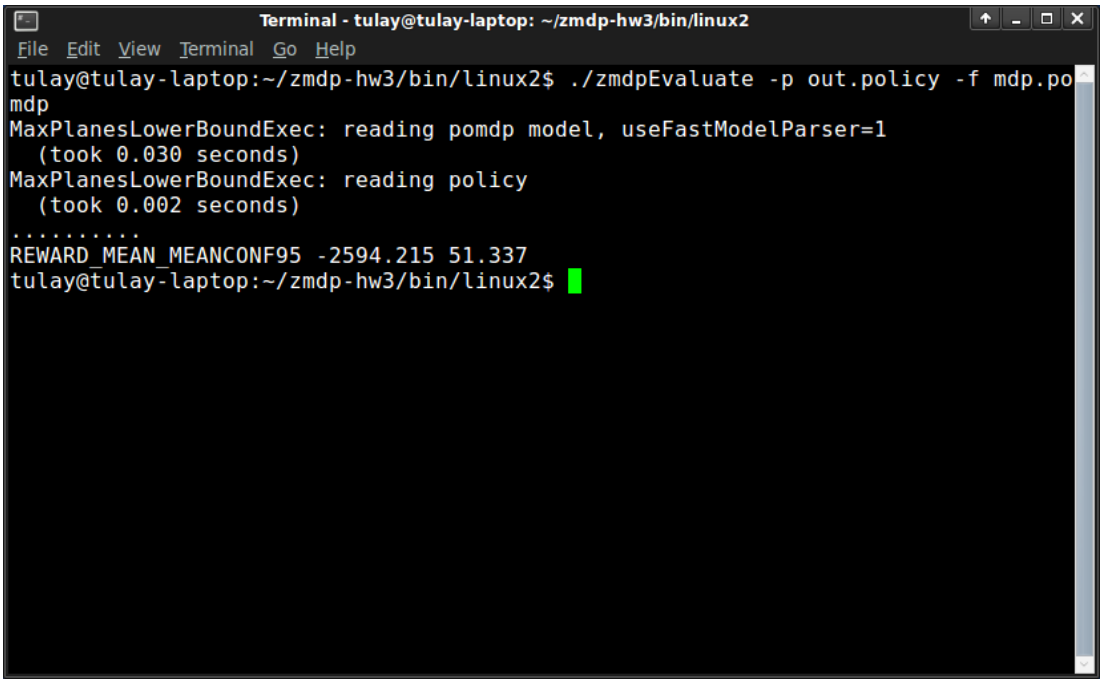
Şekil 4.12, KV'nin her adımda seçeceği eylemleri bulmamızı sağlayan düzlemlerden oluşmuştur. Her bir düşünülen durum b için, b'nin 0 olmayan girdileri, bir düzlemin $\propto$ vektöründeki mevcut girdilerin bir alt kümesi olacak şekilde tanımlanırsa, "o düzlem b'ye uygulanabilirdir" denir. Bulunulan periyottaki düşünülen durum, her

b'ye uygulanabilir düzlem için verilmiş $\propto$ vektörleri ile çarpılır ve elde edilen en büyük değer, beklenen uzun dönem ödülünün alt sınırını verir. Bu alt sınırı sağlayan düzlemin sahip olduğu eylem, bu düşünülen durum için seçilecek olan eylemdir. Şekil 4.13, Şekil 4.12'deki planın grafiksel gösterimidir. Plandaki 5 farklı alan şekilde açıkça görülmektedir.

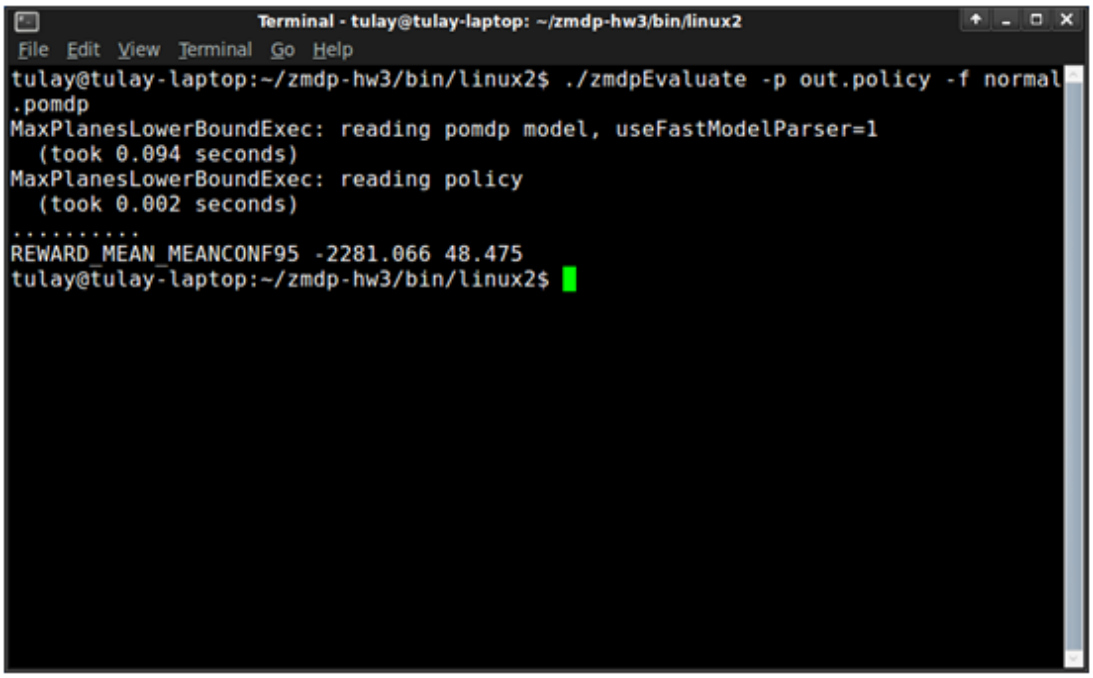


Şekil 4.13 : Kaplan problemi için optimale yakınsayan planın grafiksel gösterimi

Model örneklerinden elde edilen planlar, benzetimde bir dizi deneme yapılarak değerlendirilmiştir. Değerlendirme sonunda, bu planların ortalama ödül değerleri ve her bir denemenin ödülünün normal dağıldığı varsayımı altında, ortalamanın kestirimi için, %95 güven aralığını veren değer, Şekil 4.14 ve 4.15'te görüldüğü gibi elde edilmiştir.

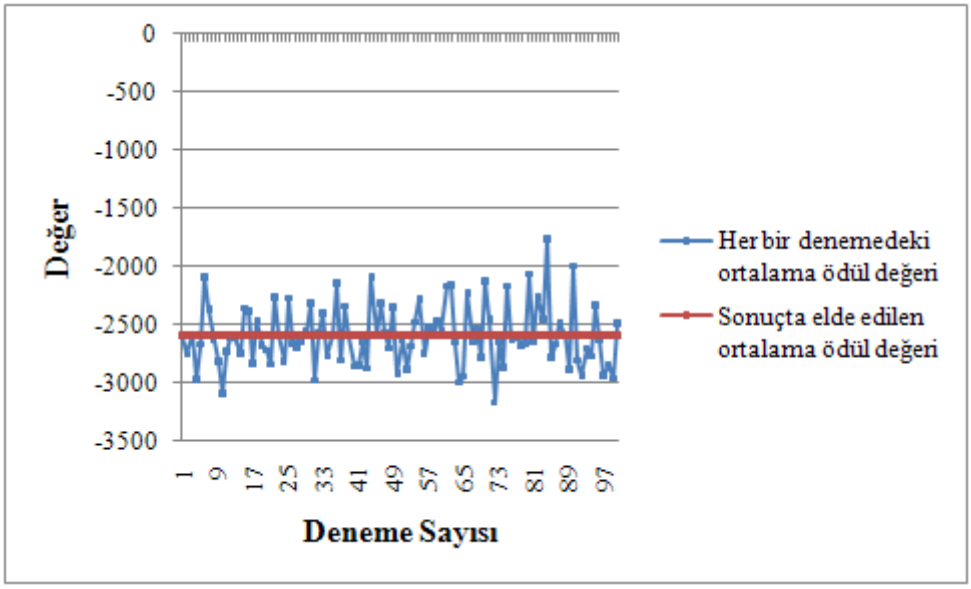


Şekil 4.14 : Önerilen modelin MDP örneği için plan değerlendirmesinin ekran görüntüsü

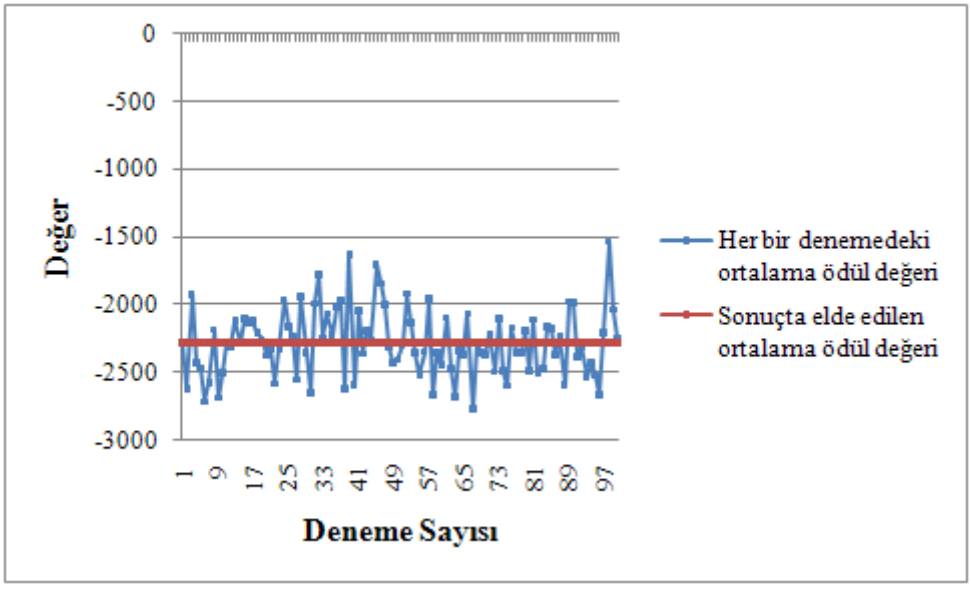


Şekil 4.15 : Önerilen modelin POMDP örneği için plan değerlendirmesinin ekran görüntüsü

Bu değerlendirmede, her bir deneme için elde edilen ödül değerleri ve sonuçta hesaplanan ortalama ödül değeri Şekil 4.16 ve Şekil 4.17'deki grafiklerde görülmektedir.



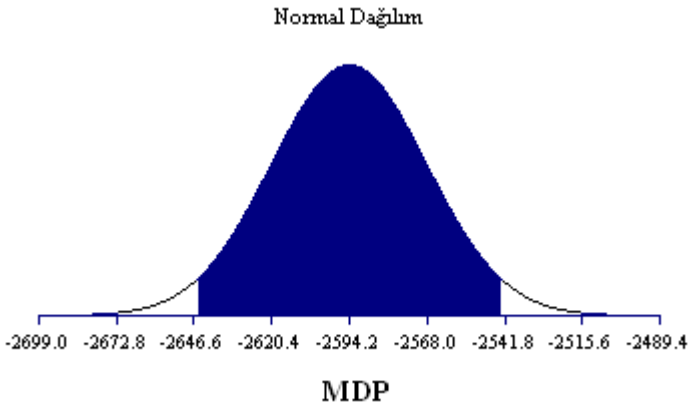
Şekil 4.16 : Önerilen modelin MDP örneği için denemelerin ödül değerleri ve ortalama ödül değeri



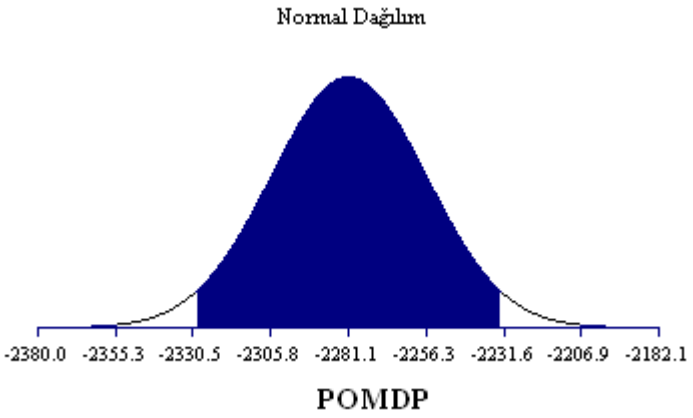
Şekil 4.17 : Önerilen modelin POMDP örneği için denemelerin ödül değerleri ve ortalama ödül değeri

Şekil 4.18 ve Şekil 4.19 önerilen modelin MDP ve POMDP örnekleri için plan değerlendirmesinden elde edilen ortalama ödül değerinin güven aralıklarını göstermektedir.

Plan değerlendirmesinde, MDP örneği için güven aralığı (-2645.5510, -2542.8790), POMDP örneği için güven aralığı ise (-2329.5405, -2232.5915) olarak hesaplanmaktadır. Bu değerler, elde edilen ortalama değere, güven aralığı için verilen değer eklenip çıkarılarak hesaplanır.

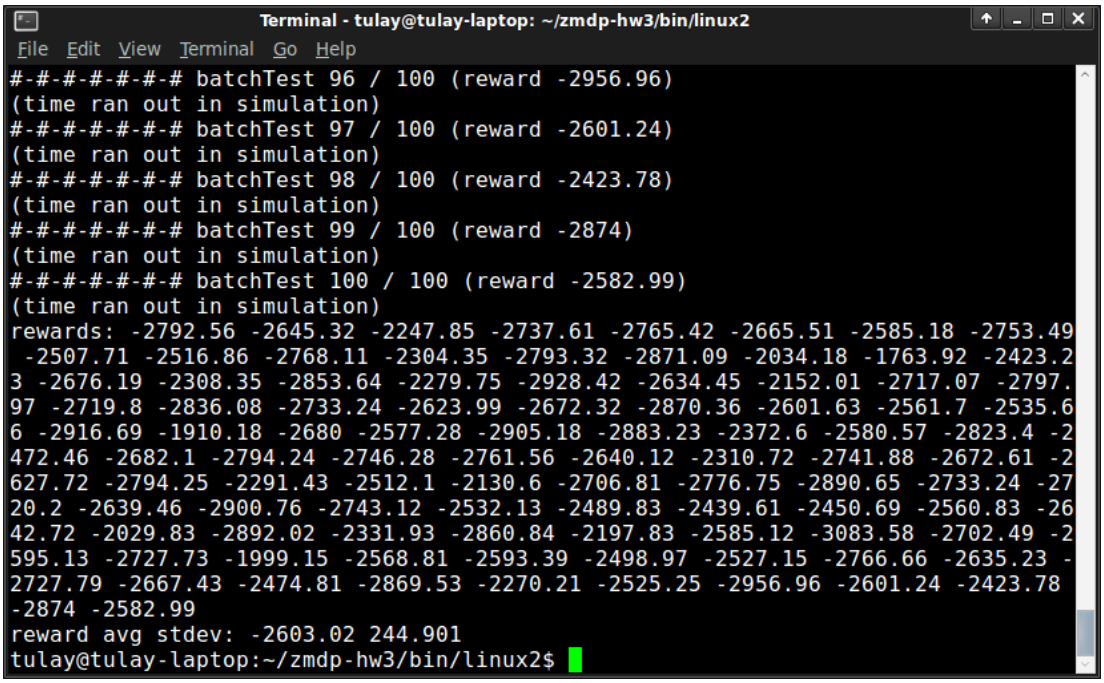


Şekil 4.18 : Önerilen modelin MDP örneği için ortalama ödül değerinin güven aralığı

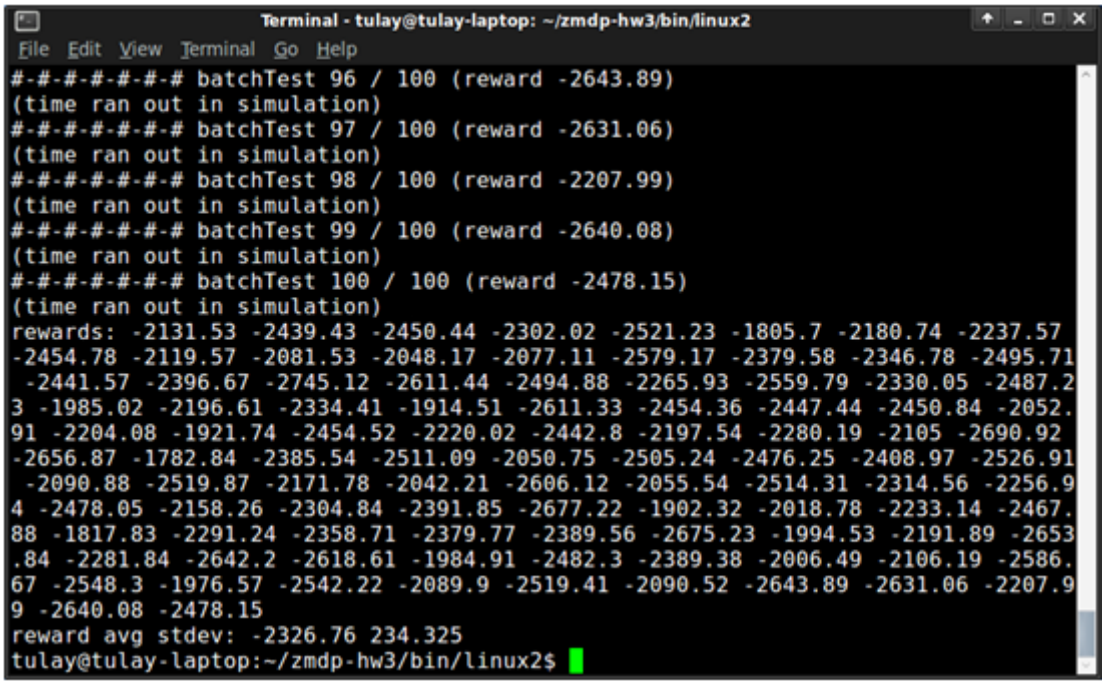


Şekil 4.19 : Önerilen modelin POMDP örneği için ortalama ödül değerinin güven aralığı

Plan değerlendirmesinden sonra, Şekil 4.20 ve 4.21'de görüldüğü gibi, bu model örneklerinin performansı test edilmiştir. Burada her biri 100 denemeden oluşan, bir dizi farklı benzetim yapılmıştır. Bu benzetimlerin her birinde, bir önceki benzetimden elde edilen plan geliştirilmiştir. Son benzetimde elde edilen plan için, her bir denemeden elde edilen ödül değerleri kullanılarak, ortalama ödül değeri ve standart sapma hesaplanmıştır.



Şekil 4.20 : Önerilen modelin MDP örneği için performans değerlendirmesinin ekran çıktısı



Şekil 4.21 : Önerilen modelin POMDP örneği için performans değerlendirmesinin ekran çıktısı

Değerlendirme ve test aşamasında yapılan tüm bu benzetimler, her bir benzetimde kullanılan planın nasıl çalıştığıyla ilgili bilgi elde etmek amacıyla incelenebilir. Örnek kaplan problemi için bulunan optimale yakınsayan planın bir benzetim modeli

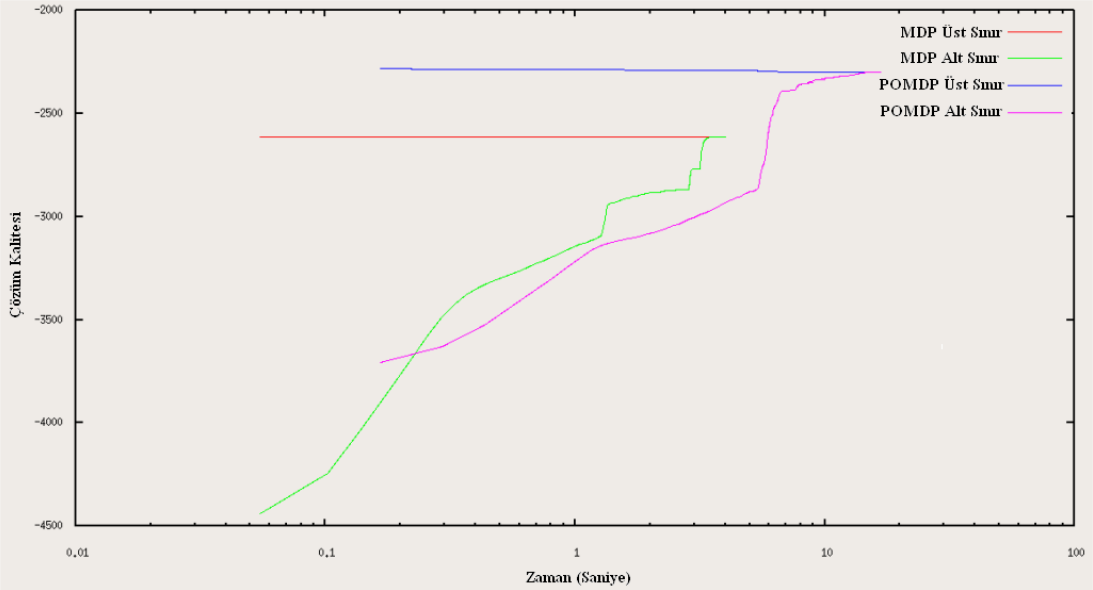
kısmi olarak Şekil 4.22'de görülmektedir. Önerilen model uygulamalarının kısmi benzetim dosyaları ekler bölümünde, EK A.5 ve EK A.6'da yer almaktadır.

```
>>> begin
sim: 1 0 1 1
belief: 1:0.85 0:0.15
sim: 1 0 1 1
belief: 1:0.969799 0:0.0302013
sim: 1 1 1 1
belief: 0:0.5 1:0.5
sim: 1 0 1 1
belief: 1:0.85 0:0.15
sim: 1 0 1 1
belief: 1:0.969799 0:0.0302013
sim: 1 1 0 1
belief: 0:0.5 1:0.5
sim: 0 0 0 1
belief: 1:0.85 0:0.15
sim: 0 0 0 0
belief: 0:0.5 1:0.5
sim: 0 0 0 0
belief: 0:0.85 1:0.15
sim: 0 0 0 0
belief: 0:0.969799 1:0.0302013
sim: 0 2 0 0
belief: 0:0.5 1:0.5
sim: 0 0 0 0
belief: 0:0.85 1:0.15
sim: 0 0 0 1
belief: 0:0.5 1:0.5
sim: 0 0 0 0
belief: 0:0.85 1:0.15
sim: 0 0 0 0
belief: 0:0.969799 1:0.0302013
sim: 0 2 0 1
belief: 0:0.5 1:0.5
sim: 0 0 0 0
belief: 0:0.85 1:0.15
sim: 0 0 0 0
belief: 0:0.969799 1:0.0302013
sim: 0 2 0 0
```

Şekil 4.22 : Örnek kaplan problemi için benzetim çıktısı

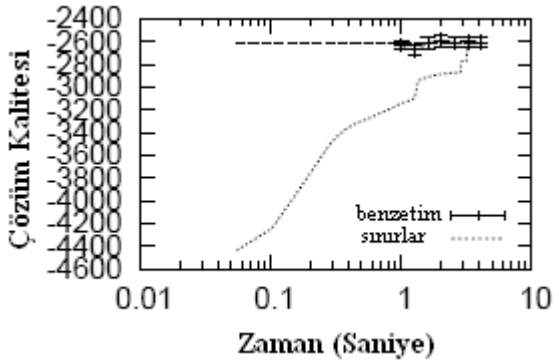
Burada sim satırındaki ilk terim mevcut durumu, ikinci terim seçilen eylemi, üçüncü terim bu eylemden sonra geçilen yeni durumu ve son terim burada elde edilen gözlemi belirtmektedir. Belief satırında ise buna göre hesaplanan yeni düşünülen durum gösterilmektedir. Örneğin ilk aşamada 1011, kaplanın sağda olduğu durumda, dinleme eyleminin seçildiğini ve yeni durumda hala kaplanın sağda olduğunu ve burada kaplan sağda gözleminin elde edildiğini göstermektedir. Bu da bir sonraki düşünülen durumda kaplanın sağda olduğu olasılığını 0.5'ten 0.85'e yükseltmiştir.

Şekil 4.23 model örneklerinin zamana göre çözüm kalitesini, yani, alt ve üst sınır değerlerinin zamana göre nasıl iyileştirildiğini göstermektedir. Bu grafik, HSVI'nın her adımda alt ve üst sınır değerleri arasındaki farkı enküçükleme amacını özetlemektedir. Ayrıca, MDP'nin karmaşıklığının POMDP'ye göre daha az olmasından dolayı, MDP örneğinde, POMDP'ye göre daha kaliteli sonuçlar elde edilmiştir. HSVI, üst sınırı belirlerken, problemi MDP şeklinde ele aldığından dolayı, önerilen modelin MDP örneğinde optimale yakınsama üst sınır üzerinde gerçekleşmiştir. POMDP örneğinde ise, λ_t değeri sadece 2 farklı değer olduğundan, yani kısmi gözlemlenebilirlik çok az olduğundan, optimale yakınsama üst sınırdan çok az bir sapma göstermiştir.

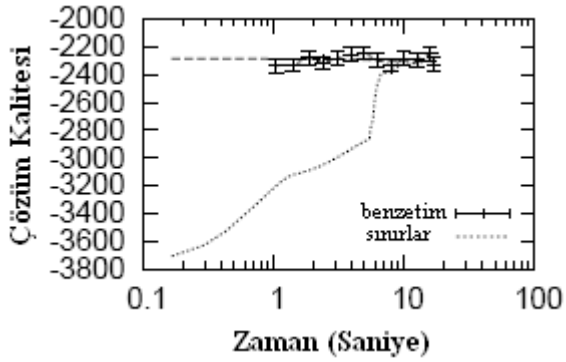


Şekil 4.23 : Önerilen model örneklerinde sınırların optimale yakınsamasının gösterimi

Şekil 4.24 ve Şekil 4.25, bu sınırlarla birlikte, farklı planların benzetimlerinden elde edilen ortalama ödül değerlerini de göstermektedir. Burada, her bir farklı planın benzetiminden elde edilen ortalama ödül değerlerinin, her zaman o durumdaki alt ve üst sınırlar arasında olduğu görülmektedir. Plan geliştirildikçe çözüm optimale yakınsar.



Şekil 4.24 : Önerilen modelin MDP örneğinde ödül ve sınır değerlerinin zamana göre değişimi



Şekil 4.25 : Önerilen modelin POMDP örneğinde ödül ve sınır değerlerinin zamana göre değişimi

5. SONUÇ VE ÖNERİLER

Bu çalışmada, bir tedarik zincirinde tek ürünlü, kayıp satışlı, tek karar vericili bir stok kontrol problemi ele alınarak taleplerin sabit olmayan ve kısmi olarak gözlemlenebilir olduğu bir POMDP modeli kurulmuştur.

POMDP ve diğer ardışık karar verme modelleri, eylemlerin uzun dönemli etkilerini göz önüne aldıkları için gelecekte stok yönetimi ve diğer Endüstri Mühendisliği uygulamalarında çok büyük faydalar sağlayacaklardır.

Ardışık karar verme modellerinin en büyük dezavantajı, karmaşıklığın çok hızlı bir şekilde artmasıdır. DEC-POMDP, POMDP'ye göre, POMDP de MDP'ye göre daha fazla karmaşıklığa sahiptir. Çünkü belirsizliklerin ve karar verici sayılarının artması işlem karmaşıklığını arttırmaktadır. Özellikle DEC-POMDP'de istenilen optimal çözümlerin elde edildiği yöntemler tam olarak geliştirilememiştir.

Ardışık karar verme modellerinde, daha az durum, gözlem ve eyleme sahip modellerin çözümü ve uygulaması daha kolaydır ve optimale daha yakın sonuçlar elde edilir. Daha büyük veri setine sahip uygulamaların yapılabilmesi ve Endüstri alanında da daha gerçekçi problemlerde kullanılabilmesi için mevcut çözücü performanslarının ya da mevcut çözüm algoritmalarının çözümüne yönelik çalışmalar yapılmalıdır. Bu modellerin hesaplama karmaşıklığı yüzünden büyük çapta uygulamalar yapılamadığından dolayı, Endüstri Mühendisliği çerçevesindeki uygulamalar genellikle teoride kalmıştır.

Önerilen modeldeki durum ve gözlemler 3 bileşene sahip vektörler olduklarından dolayı, bu bileşenlerin alacakları en büyük değerlere göre durum ve gözlem sayıları üssel olarak artar. Bunu önlemek için, bu model sadece stok seviyesinden oluşan bir durum ve gözleme sahip MDP modeli olarak çalıştırılabilir. Fakat bu durumda talep dağılımının sabit olduğu ve bilindiği varsayılacaktır.

Ayrıca önerilen modelde tedarikçinin stok seviyesi de göz önüne alınabilir. Tedarikçinin farklı perakendecilere de ürün verdiği ve her dönem belli bir dağılıma

göre üretim yaptığı düşünülerek her dönem tedarikçinin stok seviyesi güncellenir ve hesaplamalarda kullanılır. Daha önce literatürde yapılan bazı çalışmalarda olduğu gibi, bu stok seviyesi bilgisinin paylaşılıp paylaşılmamasına göre oluşan MDP ve POMDP modelleri incelenerek paylaşılan bilginin değeri ölçülebilir.

Tüm tedarik zinciri kapsamında daha gerçekçi sonuçlar elde etmek için de bu problemi çok karar vericili olarak ele almak gerekmektedir. Böyle bir durumda, tedarikçi ve perakendecinin her biri aynı anda karar vermesi gereken iki ayrı karar verici olarak ele alınabilir.

Önerilen modelde, maliyetler hesaplanmış ve bunlar enküçüklenmeye çalışılmıştır. Bunun yanı sıra, yapılan satışlar da bu hesaplamalara eklenerek tedarik zincirinin kazancı enbüyüklenebilir.

Modelde gecikmelerin olmadığı ve karşılanamayan taleplerin kayıp satış olarak ele alındığı varsayılmıştır. Bunun yerine, daha gerçekçi olarak gecikmelerin olduğu ve karşılanamayan taleplerin bekleyen sipariş olarak daha sonradan karşılanacağı düşünülebilir.

Ardışık karar verme modellerinde yeni yöntemlerin geliştirilmesine ve çözümlerine yönelik birçok çalışma sürdürülmektedir. Bu yeni yöntemler geliştirildiği taktirde, yukarıda bahsedilen tüm uygulamalar çok kolay bir şekilde gerçekleştirilecektir. Bu uygulamalar sadece stok yönetiminde değil, makine bakımı, yapısal kontrol, pazarlama ve buna benzer birçok alanda kullanılarak, özellikle zamandan büyük kazanç sağlayacaktır ve uzun dönem etkileri göz önüne alınacağından dolayı sonuçta çok büyük verimlilik elde edilecektir.

KAYNAKLAR

Amato, C., Bernstein, D. S. ve Zilberstein, S., 2007. Solving POMDPs Using Quadratically Constrained Linear Programs. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, 2418-2424.

Amato, C., Bernstein, D. S. ve Zilberstein, S., 2009a. Optimizing Fixed-Size Stochastic Controllers for POMDPs and Decentralized POMDPs. *Journal of Autonomous Agents and Multi-Agent Systems.*

Amato, C., Dibangoye, J. S. ve Zilberstein, S., 2009b. Incremental Policy Generation for Finite-Horizon DEC-POMDPs. In *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling*, 2-9.

Amato, C., Zilberstein, S., 2009. Achieving Goals in Decentralized POMDPs. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, **1**, 593-600.

Anupindi, R., Morton, T. E., ve Pentico, D., 1996. The Non-Stationary Stochastic Lead-Time Inventory Problem: Near-Myopic Bounds, Heuristics, and Testing. *Management Science*, **42**(1), 124–129.

Arrow, K. J., Harris, T., ve Marshak, J., 1951. Optimal Inventory Theory. *Econometrica*, **19**(3), 250-272.

Astrom, K. J., 1965. Optimal Control of Markov Decision Processes with Incomplete State Estimation. *J. Math. Anal. Applic.*, **10**, 174-205.

Azoury, K. S., 1985. Bayes Solution to Dynamic Inventory Models under Unknown Demand Distributions. *Management Science*, **31**(9), 1150–1160.

Bandera, C., Vico, F. J., Bravo, J. M., Harmon, M. E. ve III, L. C. B., 1996. Residual Q-Learning Applied to Visual Attention. *Machine Learning*, 20-27.

Bellman, R. E., 1957. *Dynamic Programming.* Princeton University Press, Princeton, New Jersey.

Benazera, E. ve Chanthery, E., 2008. The Challenge of Solving POMDPs for Control, Monitoring and Repair of Complex Systems. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, 15-22.

Bernstein, D. S. ve Zilberstein S., 2000. The Complexity of Decision-Theoretic Planning for Distributed Agents Source. *Technical Report: UM-CS-2000-004*, University of Massachusetts, Amherst, MA, USA.

Bernstein, D. S. ve Zilberstein S., 2003. Generalized Dynamic Programming for Decentralized POMDPs. *Sunum*, 5 Eylül 2003, University of Massachusetts, Amherst, MA.

Bernstein, D. S., 2005. Complexity Analysis and Optimal Algorithms for Decentralized Decision Making. *Doktora Tezi*, University of Massachusetts, Amherst, MA, USA.

Bernstein, D. S., Amato, C., Hansen, E. A. ve Zilberstein, S., 2009. Policy Iteration for Decentralized Control of Markov Decision Processes. *Journal of Artificial Intelligence Research*, **34**, 89-132.

Bian, A.H., Chong-Jun, W. ve Chen, S. F., 2008. Preprocessing for Point-Based Algorithms of POMDPs. *20th IEEE International Conference on Tools with Artificial Intelligence*, 519-522.

Boger, J., Poupart, P., Hoey, J., Boutilier, C., Fernie, G. ve Mihailidis, A., 2005. A Decision-Theoretic Approach to Task Assistance for Persons with Dementia. In *Proceedings of International Joint Conference on Artificial Intelligence*, 1293-1299.

Boutilier, C., Reiter, R., Soutchanski, M., ve Thrun, S., 2000. Decision-Theoretic, High-Level Agent Programming in the Situation Calculus. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (MI-OO)*, 355-362.

Cachon, G.P., ve Fisher, M., 2000. Supply Chain Inventory Management and the Value of Shared Information. *Management Science*, **45**, 843-856.

Carlin, A. ve Zilberstein, S., 2009. Value of Communication in Decentralized POMDPs. *AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains.*

Cassandra, A. R., 1998. A Survey of POMDP Applications. *Technical Report MCC-INSL*.

Cassandra, A. R., 2010. POMDP Tutorial.

<http://www.pomdp.org/pomdp/tutorial/mdp.shtml> alındığı tarih: 08.02.2010

Cassandra, A. R., Kaelbling, L. P. ve Littman, M. L., 1994. Acting Optimally in Partially Observable Stochastic Domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, 1023-1028.

Cassandra, A. R., Littman, M. L. ve Zhang, N. L., 1997. Incremental Pruning: A Simple, Fast, Exact Method for Partially Observable Markov Decision Processes. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, 54–61.

Cassandra, A., Kaelbling, L. ve Kurien, J., 1996. Acting Under Uncertainty: Discrete Bayesian Models For Mobile-Robot Navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Cheeseman, P., 1985. In Defense of Probability. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1002-1009.

Cheng, H. T., 1988. Algorithms for Partially Observable Markov Decision Processes. *Doktora Tezi*, University of British Columbia, Canada.

Cohen, S. L., 2000. Asymmetric Information in Vendor Managed Inventory Systems. *Doktora Tezi*, Stanford University.

Dean, T. ve Kanazawa, K., 1989. A Model for Reasoning About Persistence and Causation. *Computational Intelligence*, **5**(3), 142-150.

Dorf, R. C. ve Bishop, R. H., 1999. *Modern Control Systems.* Addison-Wesley, Reading, Massachusetts.

Doshi, P., Zeng, Y. ve Chen, Q., 2009. Graphical Models for Interactive POMDPs: Representations and Solutions. *Auton. Agent Multi-Agent Systems*, **18**, 376-416.

Eagle, J. N., 1984. The Optimal Search for A Moving Target When the Search Path is Constrained. *Operations Research*, **32**(5), 1107-1115.

Eckles, J. E., 1968. Optimum Maintenance with Incomplete Information. *Operations Research*, **16**(5), 1058-1067.

Ellis, H., Jiang, M., ve Corotis, R. B., 1995. Inspection, Maintenance and Repair with Partial Observability. *Journal of Infrastructure Systems*, **1**(2), 92-99.

Emery-Montemerlo, R., Gordon, G., Schneider, J., ve Thrun, S., 2005. Game Theoretic Control for Robot Teams. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Emery-Montemerlo, R., Gordon, G., Schneider, J., ve Thrun, S., 2004. Approximate Solutions for Partially Observable Stochastic Games with Common Payoffs. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems*.

Evren, R. ve Ülengin F., 1992. *Yönetimde Çok Amaçlı Karar Verme*. Teknik Üniversite Matbaası, Gümüşsuyu, İstanbul.

Eymen, U. E., 2007. Tedarik Zinciri Yönetimi. *Kaliteofisi Yayınları (e-kitap)*, <*http://www.kaliteofisi.com/dosyalar/tz.pdf*>, alındığı tarih: 01.04.2010

Feldman, J. and Sproull, R. F., 1977. Decision Theory and Artificial Intelligence 11: The Hungry Monkey. *Technical Report*, Computer Science Department, University of Rochester.

Feldman, J. and Yakimovsky, Y., 1974. Decision Theory and Artificial Intelligence I: Semantics-Based Region Analyzer. *Artificial Intelligence*, **5**(4), 349-371.

Fisher, M., Raman, A. ve McClelland, A., 2000. Rocket Science Retailing is Almost Here – Are You Ready? *Harvard Business Review*, **74**, 115–124.

Foka, A. ve Trahanias, P., 2007. Real-Time Hierarchical POMDPs for Autonomous Robot Navigation. *Robotics and Autonomous Systems*, **55**, 561-571.

Fuller, J., O'Conor, J. ve Rawlinson, R., 1993. Tailored Logistics: The Next Advantage. *Harvard Business Review*, 87-98.

Gavirneni, S., Kapuscin, R. ve Tayur, S., 1999. Value of Information in Capacitated Supply Chains. *Management Science*, **46**(1), 16-24.

Gorton, M., Dumitrashko, M. ve White, J., 2006. Overcoming Supply Chain Failure in the Agri-Food Sector: A Case Study from Moldova. *Food Policy*, **31**, 90–103.

Graves, S. C., 1997. A Single-Item Inventory Model for A Non-Stationary Demand Process. *Working Paper*, Massachusetts Institute of Technology.

Hadley, G. ve Whitin, T. M., 1961. An Optimal Final Inventory Model. *Management Science*, **7**, 179–183.

Harsanyi, J., 1967. Games with Incomplete Information Played by Bayesian Players. *Management Science*, **14**, 159-182.

Hauskrecht, M. ve Fraser, H., 2000. Planning Treatment of Ischemic Heart Disease with Partially Observable Markov Decision Processes. *Artificial Intelligence in Medicine*, **18**, 221–244.

Hauskrecht, M., 1997. Planning and Control in Stochastic Domains with Imperfect Information. *Doktora Tezi*, Massachusetts Institute of Technology.

Henzinger, T. A. ve Sastry, S. (Eds.), 1998. *Hybrid Systems: Computation and Control*. Springer-Verlag, Berlin.

Horvitz, E. J. ve Heckerman, D., 1986. The Inconsistent Use of Measures of Certainty in Artificial Intelligence Research. In Kanal, L. N. and Lemmer, J. E (Eds.), *Uncertainty in Artificial Intelligence*, 137-151.

Horvitz, E. J., Breese, J. S. ve Henrion, M., 1988. Decision Theory in Expert Systems and Artificial Intelligence. *International Journal of Approximate Reasoning*, **2**, 247-302.

Howard, R. A. ve Matheson, J. E., 1984. Influence Diagrams. In Howard, R. A. and Matheson, J. E. (Eds.), *Readings on the Principles and Applications of Decision Analysis*, 721-762.

Howard, R. A., 1960. *Dynamic Programming and Markov Processes. MIT* Press, Cambridge, Massachusetts.

Hsiao, K., Kaelbling, L. P. ve Lozano-Pérez, T., 2007. Grasping POMDPs. *IEEE International Conference on Robotics and Automation*, Roma, Italy, 4685-4692.

Hu, C., Lovejoy, W. S. ve Shafer, S. L., 1996. Comparison of Some Suboptimal Control Policies in Medical Drug Therapy. *Operations Research*, **44**(5), 696–709.

Huang, G.Q., Lau, J.S.K. ve Mak, K.L., 2003. The Impacts of Sharing Production Information on Supply Chain Dynamics: A Review of the Literature. *International Journal of Production Research*, **41**(7), 1483-1517.

Hughes, J., 1977. Optimal Internal Audit Timing. *Accounting Review LII*, 56-58.

Huygens, C., 1657. Ratiociniis in Ludo Aleae. In van Schooten, F. (Ed.), *Exercitionum Mathematicorum*, Elsevirii, Amsterdam.

Kaelbling, L. P., Littmann, M. L. ve Cassandra, A. R., 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence,* **101**(2), 99–134.

Kamath, K. R. ve Pakkala, T. P. M., 2002. A Bayesian Approach to A Dynamic Inventory Model under An Unknown Demand Distribution. *Computers and Operations Research*, **29**, 403–422.

Kambhamettu, S. S., 2000. Parameter Estimation for Partially Observable Markov Decision Processes. *Yüksek Lisans Tezi,* Auburn University, Auburn, Alabama.

Kaplan, A. J., 1988. Bayesian Approach to Inventory Control of New Parts. *IIE Transactions*, **20**(2), 151-156.

Kaplan, R., 1969. Optimal Investigation Strategies with Imperfect Information. *Journal of Accounting Research*, **7**, 32-43.

Karlin, S., 1960. Dynamic Inventory Policy with Varying Stochastic Demands. *Management Science*, **6**(3), 231-258.

Karush, W. ve Dear, R., 1967. Optimal Strategy for Item Presentation in A Learning Process. *Management Science*, **13**(11), 773-785.

Keeney, R. L. ve Raiffa, H., 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* Wiley, New York.

King, R. E., Hodgson, T. J., Joines, J. ve Thoney, K. A., 2005. Quantifying the Value of Information in A Supply Chain. *National Textile Center Annual Report*.

Koenig, S., 1991. Optimal Probabilistic and Decision Theoretic Planning Using Markovian Decision Theory. *Master's Report*, Computer Science Division, University of California, Berkeley.

Kumar, A. ve Zilberstein, S., 2009a. **Constraint-Based Dynamic Programming for Decentralized POMDPs with Structured Interactions. In** *Proceedings of the Eighth International Conference on Autonomous Systems and Multi-agent Systems,* 561-568.

Kumar, A. ve Zilberstein, S., 2009b. **Event-Detecting Multi-Agent MDPs: Complexity and Constant-Factor Approximation.** In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence*, 201-207.

Kumar, A. ve Zilberstein, S., 2009c. Dynamic Programming Approximations for Partially Observable Stochastic Games. *Proceedings of the Twenty-Second International FLAIRS Conference.*

Kumar, P. R. ve Varaiya, P., 1986. *Stochastic Systems: Estimation, Identification, and Adaptive Control.* Prentice-Hall, Upper Saddle River, New Jersey.

Kurawarwala, A. A. ve Matsuo, H., 1996. Forecasting and Inventory Management of Short Life-Cycle Products. *Operations Research*, **44**(1), 131–150.

L*ó*pez, M. E., Bergasa, L. M., Barea, R. ve Escudero M. S., 2005. A Navigation System for Assistant Robots Using Visually Augmented POMDPs. *Autonomous Robots*, **19**, 67–87.

Lariviere, M. A. ve Porteus, E. L., 1997. Bayesian Inventory Management with Unobserved Lost Sales. *Working Paper*, Fuqua School of Business, Duke University.

Lariviere, M. A. ve. Porteus, E. L., 1999. Stalking Information: Bayesian Inventory Management with Unobserved Lost Sales. *Management Science*, **45**, 346–363.

Lee, H. L., and Nahmias, S., 1993. Single Product, Single-Location Models. In Graves, S.C., Kan, A.H.G. Rinnooy, and Zipkin, P.H., editors, *Handbooks in Operations Research and Management Science: Logistics of Production and Inventory*, **4**, 1-55.

Lee, H., Padmanabhan, V. ve Whang, S., 1997. Information Distortion in A Supply Chain: The Bullwhip Effect. *Management Science*, **43**(4), 546-558.

Lee, H.L., So, K.C. ve Tang. C.S., 2000. The Value of Information Sharing in A Two-Level Supply Chain. *Management Science*, **46**, 626-643.

Li, G., Lin, Y., Wang, S. ve Yan, H., 2006. Enhancing Agility by Timely Sharing of Supply Information. *Supply Chain Management: An International Journal*, **11**(5), 425– 435.

Lipstein, B., 1965. A Mathematical Model of Consumer Behavior. *Journal of Marketing Research*, **2**(3), 259-265

Littman, M. L., 1994. Markov Games as A Framework for Multi-Agent Reinforcement Learning. In *Proceedings of the 11th International Conference on Machine Learning*, 157-163.

Littman, M. L., 1996. Algorithms for Sequential Decision Making. *Doktora Tezi*, Brown University.

Liu, F. ve Zeng, G., 2008. An Online Multi-Agent Co-operative Learning Algorithm in POMDPs. *Journal of Experimental and Theoretical Artificial Intelligence*, **20**(4), 335-344.

Lovejoy, W. S., 1990. Myopic Policies for Some Inventory Models with Uncertain Demand Distributions. *Management Science*, **36**(6), 724-738.

Lovejoy, W. S., 1992. Stopped Myopic Policies in Some Inventory Models with Generalized Demand Processes. *Management Science*, **38**(5), 688-707.

Lovejoy, W. S., 1993. Suboptimal Policies, with Bounds, for Parameter Adaptive Decision Processes. *Operations Research*, **31**(3), 583-599.

Mangel, M., and Clark, C. W., 1988. *Dynamic Modeling in Behavioral Ecology*. Princeton, New Jersey: Princeton University Press.

McCarthy, J., 1968. Programs with Common Sense. In Minsky, M. L. (Ed.), *Semantic Information Processing*, 403-418.

Mccoy, J., 2007. The Impact of Internet Disruption to An Information-Sharing Supply Chain. *Yüksek Lisans Tezi*, North Carolina State University, Raleigh, North Carolina.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2003. Agent-Based Simulation of the Amplification of Demand Variability in A Supply Chain. In *Proceedings 4th Workshop on Agent-Based Simulation*, 166-172.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2004. Multi-Agent Simulation of Collaborative Strategies in A Supply Chain. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, **1**, 52-59.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2006a. Supply Chain Management and Multi-Agent Systems: An Overview. *Multi-Agent Based Supply Chain Management*, 1-27.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2006b. Design Implementation and Test of Collaborative Strategies in the Supply Chain. *Multi-Agent Based Supply Chain Management*, 247-272.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2006c. Study of Social Consciousness in Stochastic Agent-Based Simulations: Application to Supply Chains. *AAMAS*.

Moyaux, T., Chaib-draa, B. ve D'Amours, S., 2008. Spreadsheet vs. Multiagent-Based Simulations in the Study of Decision Making in Supply Chains. *International Journal of Simulation and Process Modelling*, **4**(2), 89-105.

Nahmias, S., 1994. Demand Estimation in Lost Sales Inventory Systems. *Naval Research Logistics, To Appear*.

Nair, R., Roth, M. ve Yohoo, M., 2004. Communication for Improving Policy Computation in Distributed POMDPs. In Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-agent Systems, **3**, 1098-1105.

Nash, J., 1950. Equilibrium Points in N-Person Games. In *Proceedings of the National Academy of Sciences of the United States of America*, **36**, 48-49.

Nourbakhsh, I., Powers, R. ve Birchfield, S., 1995. Dervish: An Office-Navigating Robot. *AI Magazine*, **16**(2), 253-60.

Oliehoek, F. A., Kooij, J. F. P. ve Vlassis, N., 2008. The Cross-Entropy Method for Policy Search in Decentralized POMDPs. Informatica, **32**, 341-357.

Ooi, J. M., ve Wornell, G. W., 1996. Decentralized Control of A Multiple Access Broadcast Channel: Performance Bounds. In *Proceedings of the Thirty-Fifth Conference on Decision and Control*, 293–298.

Ortiz, O., Erera, A. L. ve White, III, C. C., 2006. Bounds on the Value of Demand Observability for Inventory Control with Markovian Demand and Unobserved Lost Sales. In *Proceedings of the 2006 Manufacturing and Service Operations Management Conference*, Atlanta,GA.

Papadimitriou, C. H. ve Tsitsiklis, J. N., 1987. The Complexity of Markov Decision Processes. *Mathematics of Operations Research*, **12**(3), 441-450.

Pearl, J., 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, California.

Petrik, M. ve Zilberstein, S., 2009. Constraint Relaxation in Approximate Linear Programs. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning*, 809-816.

Pierskalla, W. P., 1969. An Inventory Problem with Obsolescence. *Naval Research Logistics Quarterly*, **16**, 217–228.

Pierskalla, W., ve Voelker, J., 1976. A Survey of Maintenance Models: The Control and Surveillance of Deteriorating Systems. *Naval Research Logistics Quarterly*, **23**, 353-388.

Pineau, J., Montemerlo, M., Pollack, M., Roy, N. ve Thrun., S., 2003. Towards Robotic Assistants in Nursing Homes: Challenges and Results. *Robotics and Autonomous Systems*, **42**(3–4), 271–281.

Pollock, 1970. A Simple Model of Search for A Moving Target. *Operations Research*, **18**, 883-903.

Porta, M. P., Vlassis, N., Spaan, M. T. J. ve Poupart, P., 2006. Point-Based Value Iteration for Continuous POMDPs. *Journal of Machine Learning Research*, **7**, 2329-2367.

Puterman, M. L. ve Shin, M. C., 1978. Modified Policy Iteration Algorithms for Discounted Markov Decision Problems. *Management Science*, **24**, 1127-1137.

Puterman, M. L., 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* Wiley, New York.

Ross, S. M., 1971. Quality Control Under Markovian Deterioration. *Management Science*, **17**(9), 587-596.

Ross, S., Chaib-draa, B. ve Pineau, J., 2008. Bayesian Reinforcement Learning in Continuous POMDPs with Application to Robot Navigation. *IEEE International Conference on Robotics and Automation*, 2845-2851.

Roy, N., Pineau, J., ve Thrun, S., 2000. Spoken Dialog Management for Robots. In *Proceedings of the Association for Computational Linguistics*.

Rusmevichientong., P. ve Van Roy, B., 2001. A Tractable POMDP for A Class of Sequencing Problems. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence.*

Russell, D. ve Saldanha, J., 2003. Five Tenets of Security-Aware Logistics and Supply Chain Operation. *Transportation Journal*, **42**(4), 44-54.

Russell, S. ve Norvig, P., 1995. *Artificial Intelligence: A Modern Approach*. Second Edition (2003), Prentice Hall, New Jersey, U.S.A.

Scarf, H. E., 1959. Bayes Solution of the Statistical Inventory Problem. *Annals of Mathematical Statistics*, **30**, 490–508.

Scarf, H. E., 1960. Some Remarks on Bayes Solution to the Inventory Problem. *Naval Research Logistics Quarterly*, **7**, 591-596.

Segall, A., 1976. Dynamic File Assignment in A Computer Network. *IEEE Transactions on Automatic Control AC*, **21**, 161-173.

Seuken, S. ve Zilberstein, S., 2008. Formal Models and Algorithms for Decentralized Decision Making under Uncertainty. *Journal of Autonomous Agents and Multi-Agent Systems*, **12**(2), 190-250.

Shachter, R. D., 1986. Evaluating Influence Diagrams. *Operations Research,* **34**, 871-882.

Shani, G., Brafman, R. I. ve Shimony, S. E., 2006. Prioritizing Point-Based POMDP Solvers. *Lecture Notes in Computer Science*, 389-400.

Shapley, S., 1953. Stochastic Games. In *Proceedings of the National Academy of Sciences,* **39**, 1095-1100.

Sheffi, Y., 2001. Supply Chain Management under the Threat of International Terrorism. *The International Journal of Logistics Management*, **12**(2), 1-11.

Simchi-Levi, D., Kaminsky, P., Simchi-Levi, E., 2000. *Designing and Managing the Supply Chain*. McGraw-Hill Higher Education.

Simmons, R. ve Koenig, S., 1995. Probabilistic Robot Navigation in Partially Observable Environments. In *Fourteenth International Joint Conference on Artificial Intelligence*, **2**, 1080-1087.

Simon, H., 1978. Rational Decision-Making in Business Organizations. *Nobel Memorial Lecture*, Carnegie-Mellon University, Pittsburgh, Pennsylvania, USA

Smallwood, R. D. ve Sondik, E. J., 1973. The Optimal Control of Partially Observable Markov Processes over A Finite Horizon. *Operations Research,* **21**, 1071-1088.

Smallwood, R., 1971. The Analysis of Economic Teaching Strategies for A Simple Learning Model. *Journal of Math Psych*, **8**, 285-301.

Smallwood, R., Sondik, E. ve Offensend, F., 1971. Toward and Integrated Methodology for the Analysis of Health-Care Systems. *Operations Research*, **19**(6), 1300-1322.

Smith, T., 2007. ZMDP Software for POMDP and MDP Planning. <*http://www.cs.cmu.edu/~trey/zmdp/*>, alındığı tarih: 01.02.2010

Smith, T. ve Simmons, R., 2004. Heuristic Search Value Iteration for POMDPs. *UAI*, 520-527.

Sondik, E. J., 1971. The Optimal Control of Partially Observable Markov Decision Processes. *Doktora Tezi*, Stanford University, Stanford, California.

Song, J. S. ve Zipkin, P., 1993. Inventory Control in a Fluctuating Demand Environment. *Operations Research*, **41**(2), 351-370.

Song, J. S. ve Zipkin, P. H., 1996. Managing Inventory with the Prospect of Obsolescence. *Operations Research*, **44**(1), 215–222.

Srivastava, S., Immerman, N. ve Zilberstein, S., 2009. Challenges in Finding Generalized Plans. *Workshop on Generalized Planning: Macros, Loops, Domain Control*.

Stewart, T. J., 1992. A Critical Survey on the Status of Multiple Criteria Decision Making Theory and Practice. University of Cape Town, South Africa, **20**(5-6), 569-586.

Sutton, R. S., 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning,* **3**, 9-44.

Swaminathan, J. M., Sadeh, N.M. ve Smith, S.F., 1997. *Effect of Sharing Supplier Capacity Information*. Haas School of Business, University of California, Berkeley. The Robotics Institute, Carnegie Mellon University, Pittsburgh.

Tanyaş, M. ve Baskak, M., 2003. *Üretim Planlama ve Kontrol*. İrfan Yayımcılık, İstanbul.

Theocharous, G. ve Mahadevan, S., 2002. Approximate Planning with Hierarchical Partially Observable Markov Decision Processes for Robot Navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation.*

Thiebeaux, S., Cordier, M. O., Jehl, O. ve Krivine, J. P., 1996. Supply Restoration in Power Distribution Systems: A Case Study in Integrating Model-Based Diagnosis and Repair Planning. In *Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence*, 525-532.

Toussaint, M., Charlin, L. ve Poupart, P., 2008. Hierarchical POMDP Controller Optimization by Likelihood Maximization. *Association for the Advancement of Artificial Intelligence*, 55-60.

Treharne, J. T. ve Sox, C. R., 1999. Models and Methods for Adaptive Inventory Control, Under review for *Naval Research Logistics*.

Treharne, J. T. ve Sox, C. R., 1999-2002. Adaptive Inventory Control for Partially Observed, Non-Stationary Demand. *National Science Foundation*.

Treharne, J. T. ve Sox, C. R., 2002. Adaptive Inventory Control for Non-Stationary Demand and Partial Information. *Management Science,* **48**(5), 607–624.

Van Nunen, J. A. E. E., 1976. A Set of Successive Approximation Methods for Discounted Markovian Decision Problems. *Zeitschrift fur Operations Research, Serie A,* **20**(5), 203-208.

Von Neumann, J., 1928. Zur Theorie der Gesellschaftssiele. *Mathematische Annalen,* 295-320.

Wang, C. ve Schmolze, J., 2005. Planning with POMDPs Using A Compact, Logic-Based Representation. In *Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence.*

Watkins, C. J., 1989. Models of Delayed Reinforcement Learning. *Doktora Tezi*, Psychology Department, Cambridge University, Cambridge, UK.

Wei, W., 2005. Quantifying Shared Information Value in A Supply Chain Using Decentralized Markov Decision Processes with Restricted Observations. *Doktora Tezi,* North Carolina State University, Raleigh, North Carolina.

Wellman, M. P., 1995. The Economic Approach to Artificial Intelligence. *ACM Computing Surveys*, **27**(3), 360-362.

White, III, C. C., 1976. Optimal Diagnostic Questionnaires which Allow Less Than Truthful Responses. *Information and Control*, **32**, 61-74.

White, III, C. C., 1977. A Markov Quality Control Process Subject to Partial Observation. *Management Science*, **23**(8), 843-852.

White, III, C. C., 1979. Bounds on Optimal Cost for A Replacement Problem with Partial Observations. *Naval Research Logistics Quarterly*, **26**, 415-422.

Williams, J. D., Young, S., 2007. Scaling POMDPs for Spoken Dialog Management. *IEEE Transactions on Audio Speech and Language Processing*, **15**(7), 2116-2129

Williams, R. J. ve Baird, L. C. I., 1993. Tight Performance Bounds on Greedy Policies Based on Imperfect Value Functions. *Technical Report*, NU-CCS-93-14, College of Computer Science, Northeastern University, Boston.

Wu, F., Zilberstein, S. ve Chen, X., 2009. Multi-Agent Online Planning with Communication. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 321-329.

Yost, K. A. ve Washburn, A. R., 2000. The LP/POMDP Marriage: Optimization with Imperfect Information. *Naval Research Logistics,* **47**(8), 607-619.

Yost, K. A., 1998. Solution of Sequential Weapons Allocation Problems with Imperfect, Costly Information. Presented at *the Sixth INFORMS Computer Science Technical Section Conference*.

Zilberstein, S., 2010a. Lecture 19: Planning Under Uncertainty. *Artificial Intelligence Lecture Notes*, University of Massachusetts, Amherst.

Zilberstein, S., 2010b. Lecture 22: Decentralized POMDPs. *Artificial Intelligence Lecture Notes*, University of Massachusetts, Amherst.

EKLER

EK A.1 : Model uygulamasının birinci veri seti için girdi dosyası oluşturan C kodu
EK A.2 : Model uygulamasının ikinci veri seti için girdi dosyası oluşturan C kodu
EK A.3 : Model uygulamasının birinci veri seti için optimale yakınsayan planı içeren çıktı dosyası
EK A.4 : Model uygulamasının ikinci veri seti için optimale yakınsayan planı içeren çıktı dosyası
EK A.5 : Model uygulamasının birinci veri seti için kısmi plan benzetim çıktısı
EK A.6 : Model uygulamasının ikinci veri seti için kısmi plan benzetim çıktısı

EK A.1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

double Factorial1 (double a)
{
  double answer = 1.0;
  while(a>1)
  answer *= a--;
  return answer;
}

double Poisson(double i,double j)
{
  double  res2;
  double answer = 0.0;
  res2 = 0.0;
  res2 =(pow(i,j) * (exp(-i))) / Factorial1(j);
  return res2;
}

double Min(int min,int b)
{
  if ( min > b) return (b);
  else return (min);
}

double Max(int max, int a)
{
  if ( max < a) return (a);
  else return (max);
}

struct state
{
   double lambda[1];
   int demandp[10];
   int rinventory[10];
};

struct observation
{
   double lambda[1];
   int demandp[10];
   int rinventory[10];
};
```

```
struct olasilik
{
   double lambda[100];
   double demandp[100];
   double rinventory[100];
   double durum[100];
};

main()
{
  double discount=0.9;
  int i,j,a,x,g,d,c,h,p,u,v,oo,kk,yy;
  double m=1/100.;
  double l,y,z,t,aa,n,tt;
  long double cost;
  struct state s;
  struct observation o;
  struct olasilik olasilik;
  int numberOfstates=100;
  int numberOfobservations=100;
  int numberOfactions=10;
  z=0;yy=0;
  FILE *fp;
  fp=fopen("mdp.pomdp","a+");
  if (fp==NULL) {printf("Error!File cannot be opened"); exit(1);}

 for(i=0; i<1; i++)
    s.lambda[i]=5;

 for(i=0; i<10; i++)
    s.demandp[i]=i;

 for(i=0;i<10;i++)
    s.rinventory[i]=i;

 for(i=0; i<1; i++)
    o.lambda[i]=5;

 for(i=0; i<10; i++)
    o.demandp[i]=i;

 for(i=0;i<10;i++)
    o.rinventory[i]=i;

 fprintf(fp,"discount: %.1f\n", discount);
 fprintf(fp,"values: reward\n");
 fprintf(fp,"states: %d\n",numberOfstates);
 fprintf(fp,"actions: %d\n",numberOfactions);
 fprintf(fp,"observations: %d\n",numberOfobservations);
 fprintf(fp,"start: ");
```

```
//Baslangic dusunulen durum yazma
for(i=0; i<100;i++)
  {fprintf(fp,"%.2f ",m);}
fprintf(fp,"\n");
h=-1;tt=0;

//Gecis fonksiyonu yazma
for (a=0; a<10; a++)
{
  yy=0;
  for(i=0; i<1;i++)
    for(j=0; j<10; j++)
      for(x=0; x<10; x++)
        for(g=0; g<1; g++)
          for(d=0; d<10; d++)
            for(c=0; c<10; c++)
            {
              h++;
              l=Max(s.rinventory[x]-s.demandp[j],0);
              olasilik.demandp[h]=Poisson(s.lambda[i],s.demandp[d]);
              if((l+a)==s.rinventory[c]) olasilik.rinventory[h]=1;
              else olasilik.rinventory[h]=0;
              olasilik.durum[h]=olasilik.rinventory[h]*olasilik.demandp[h];
              if ((h+1)==numberOfstates)
              {
                for(kk=0;kk<numberOfstates;kk++)
                tt+=olasilik.durum[kk];
                for(kk=0;kk<numberOfstates;kk++)
                 {
                   if (tt!=0) olasilik.durum[kk]=olasilik.durum[kk]/tt;
                   if (tt==0&&kk==0) (olasilik.durum[kk]=1);
                   else if (tt==0&&kk!=0) (olasilik.durum[kk]=0);
                 }
                for(kk=0;kk<numberOfstates;kk++)
                 {
  if (olasilik.durum[kk]!=0) fprintf(fp, "T: %d : %d : %d
%.2f\n",a,yy,kk,olasilik.durum[kk]);
                 }
h=-1;yy++;tt=0;
              }
            }
}
yy=0;h=-1;tt=0;
```

```
//Gozlem fonksiyonu yazma
for (a=0; a<10; a++)
{
   yy=0;
   for(i=0; i<1; i++)
     for(j=0; j<10; j++)
       for(x=0; x<10; x++)
         for(g=0; g<1; g++)
           for(d=0; d<10; d++)
             for(c=0; c<10; c++)
             {
              h++;
              if((s.demandp[j]==s.demandp[d])&&(s.rinventory[x]==s.rinventory[c]))
olasilik.durum[h]=1;
              else olasilik.durum[h]=0;
              if ((h+1)==numberOfstates)
              {
                for(kk=0;kk<numberOfstates;kk++)
                if (olasilik.durum[kk]!=0) fprintf(fp, "O: %d : %d : %d
%.2f\n",a,yy,kk,olasilik.durum[kk]);
                 h=-1;yy++;tt=0;
              }
             }
}

//Odul fonksiyonu yazma
h=1;
p=100;
u=50;
v=0;
aa=0;
n=0;
y=0;
oo=0;
for(a=0; a<10; a++)
{
  oo=0;
  for(i=0; i<1; i++)
    for(j=0; j<10; j++)
      for(x=0; x<10; x++)
      {
        cost=0;
        aa=Max(0, s.rinventory[x]-s.demandp[j]);
        if (a>(9-aa)) {cost=1.0e30;
        fprintf(fp,"R:%d     : %d    : * : * %Lf\n", a, oo, -cost);oo++;}
        else
            {
              for(z=0; z<10; z++)
               {
                 n=Max(0, aa+a-z);
```

```
      y=Min(0,aa+a-z);
      t=0.0;
      t=Poisson(s.lambda[i],z);
      cost+=t*(h*n-p*y+u*a+v*a);
    }
    fprintf(fp,"R:%d     : %d    : * : * %Lf\n", a, oo, -cost);oo++;
  }
}
}
 fclose(fp);
 return 0;
}
```

EK A.2

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

double Factorial1 (double a)
{
  double answer = 1.0;
  while(a>1)
  answer *= a--;
  return answer;
}

double Poisson(double i,double j)
{
  double  res2;
  double answer = 0.0;
  res2 = 0.0;
  res2 =(pow(i,j) * (exp(-i))) / Factorial1(j);
  return res2;
}

double Min(int min,int b)
{
  if ( min > b) return (b);
 else return (min);
}

double Max(int max, int a)
{
  if ( max < a) return (a);
 else return (max);
}

struct state
{
  double lambda[2];
  int demandp[10];
  int rinventory[10];
};

struct observation
{
  double lambda[2];
  int demandp[10];
  int rinventory[10];
};
```

```
struct olasilik
{
  double lambda[200];
  double demandp[200];
  double rinventory[200];
  double durum[200];
};

main()
{
  double discount=0.9;
  int i,j,a,x,g,d,c,h,p,u,v,oo,kk,yy;
  double m=1/200.;
  double l,y,z,t,aa,n,tt;
  long double cost;
  struct state s;
  struct observation o;
  struct olasilik olasilik;
  int numberOfstates=200;
  int numberOfobservations=200;
  int numberOfactions=10;
  z=0;yy=0;
  FILE *fp;
  fp=fopen("normal.pomdp","a+");
  if (fp==NULL) {printf("Error!File cannot be opened"); exit(1);}
  for(i=0; i<2; i++)
     s.lambda[i]=3+(5*i);

  for(i=0; i<10; i++)
     s.demandp[i]=i;

  for(i=0;i<10;i++)
     s.rinventory[i]=i;

  for(i=0; i<2; i++)
     o.lambda[i]=3+(5*i);

 for(i=0; i<10; i++)
    o.demandp[i]=i;

for(i=0;i<10;i++)
   o.rinventory[i]=i;

fprintf(fp,"discount: %.1f\n", discount);
fprintf(fp,"values: reward\n");
fprintf(fp,"states: %d\n",numberOfstates);
fprintf(fp,"actions: %d\n",numberOfactions);
fprintf(fp,"observations: %d\n",numberOfobservations);
fprintf(fp,"start: ");
```

```
//Baslangic dusunulen durum yazma
for(i=0; i<200;i++)
  {fprintf(fp,"%f ",m);}
fprintf(fp,"\n");
h=-1;tt=0;

//Gecis fonksiyonu yazma
for (a=0; a<10; a++)
{
  yy=0;
  for(i=0; i<2;i++)
    for(j=0; j<10; j++)
      for(x=0; x<10; x++)
        for(g=0; g<2; g++)
          for(d=0; d<10; d++)
            for(c=0; c<10; c++)
            {
              h++;
              l=Max(s.rinventory[x]-s.demandp[j],0);
              olasilik.demandp[h]=Poisson(s.lambda[i],s.demandp[d]);
              if((l+a)==s.rinventory[c]) olasilik.rinventory[h]=1;
              else olasilik.rinventory[h]=0;
              olasilik.lambda[h]=1;
   olasilik.durum[h]=olasilik.rinventory[h]*olasilik.demandp[h]*olasilik.lambda[h];
              if ((h+1)==numberOfstates)
              {
               for(kk=0;kk<numberOfstates;kk++)
               tt+=olasilik.durum[kk];
               for(kk=0;kk<numberOfstates;kk++)
                 {
                  if (tt!=0) olasilik.durum[kk]=olasilik.durum[kk]/tt;
                  if (tt==0&&kk==0) (olasilik.durum[kk]=1);
                  else if (tt==0&&kk!=0) (olasilik.durum[kk]=0);
                 }
              for(kk=0;kk<numberOfstates;kk++)
                 {
                  if (olasilik.durum[kk]!=0) fprintf(fp, "T: %d : %d : %d
%.4f\n",a,yy,kk,olasilik.durum[kk]);
                 }
h=-1;yy++;tt=0;
              }
            }
}
yy=0;h=-1;tt=0;
```

```
//Gozlem fonksiyonu yazma
for (a=0; a<10; a++)
{
  yy=0;
  for(i=0; i<2; i++)
     for(j=0; j<10; j++)
        for(x=0; x<10; x++)
           for(g=0; g<2; g++)
              for(d=0; d<10; d++)
                 for(c=0; c<10; c++)
                 {
                   h++;
                   if((s.demandp[j]==s.demandp[d])&&(s.rinventory[x]==s.rinventory[c]))
olasilik.rinventory[h]=1;
                  else olasilik.rinventory[h]=0;
                  l=Max(s.rinventory[x]-s.demandp[j],0);
                  if ((l+a)<5&&s.lambda[g]==3) olasilik.lambda[h]=0.75;
                 else if ((l+a)>=5&&s.lambda[g]==3) olasilik.lambda[h]=0.25;
                 else if ((l+a)<5&&s.lambda[g]==8) olasilik.lambda[h]=0.25;
                 else if ((l+a)>=5&&s.lambda[g]==8) olasilik.lambda[h]=0.75;
                 olasilik.durum[h]=olasilik.rinventory[h]*olasilik.lambda[h];
                  if ((h+1)==numberOfstates)
                  {
                     for(kk=0;kk<numberOfstates;kk++)
                     if (olasilik.durum[kk]!=0) fprintf(fp, "O: %d : %d : %d
%.2f\n",a,yy,kk,olasilik.durum[kk]);
                     h=-1;yy++;tt=0;
                  }
                }
}

//Odul fonksiyonu yazma
h=1;
p=100;
u=50;
v=0;
aa=0;
n=0;
y=0;
oo=0;
for(a=0; a<10; a++)
{
   oo=0;
   for(i=0; i<2; i++)
      for(j=0; j<10; j++)
         for(x=0; x<10; x++)
         {
          cost=0;
          aa=Max(0, s.rinventory[x]-s.demandp[j]);
          if (a>(9-aa)) {cost=1.0e30;
```

```
        fprintf(fp,"R:%d     : %d    : * : * %Lf\n", a, oo, -cost);oo++;}
         else
             {
              for(z=0; z<10; z++)
               {
                 n=Max(0, aa+a-z);
                 y=Min(0,aa+a-z);
                  t=0.0;
                  t=Poisson(s.lambda[i],z);
                  cost+=t*(h*n-p*y+u*a+v*a);
                }
              fprintf(fp,"R:%d     : %d    : * : * %Lf\n", a, oo, -cost);oo++;
             }
         }
}
fclose(fp);
return 0;
}
```

EK A.3

```
{
 policyType => "MaxPlanesLowerBound",
 numPlanes => 101,
 planes => [
  {
   action => 3,
   numEntries => 1,
   entries => [
    26, -2348.95
   ]
  },
  {
   action => 3,
   numEntries => 1,
   entries => [
    48, -2348.95
   ]
  },
  {
   action => 7,
   numEntries => 1,
   entries => [
    92, -2542.57
   ]
  },
  {
   action => 5,
   numEntries => 1,
   entries => [
    68, -2445.75
   ]
  },
  {
   action => 7,
   numEntries => 1,
   entries => [
    86, -2542.57
   ]
  },
  {
   action => 7,
   numEntries => 1,
   entries => [
    93, -2542.56
   ]
  },
```

```
    {
      action => 7,
      numEntries => 1,
      entries => [
        11, -2542.56
      ]
    },
    {
      action => 2,
      numEntries => 1,
      entries => [
        38, -2300.52
      ]
    },
    {
      action => 7,
      numEntries => 1,
      entries => [
        94, -2542.56
      ]
    },
    {
      action => 6,
      numEntries => 1,
      entries => [
        12, -2494.15
      ]
    },
    {
      action => 6,
      numEntries => 1,
      entries => [
        78, -2494.15
      ]
    },
    {
      action => 5,
      numEntries => 1,
      entries => [
        13, -2445.74
      ]
    },
{
      action => 7,
      numEntries => 1,
      entries => [
        95, -2542.55
      ]
    },
```

```
{
    action => 4,
    numEntries => 1,
    entries => [
      14, -2397.33
    ]
  },
  {
    action => 3,
    numEntries => 1,
    entries => [
      59, -2348.92
    ]
  },
  {
    action => 2,
    numEntries => 1,
    entries => [
      49, -2300.51
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      96, -2542.55
    ]
  },
  {
    action => 1,
    numEntries => 1,
    entries => [
      28, -2252.1
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      88, -2542.55
    ]
  },
  {
    action => 4,
    numEntries => 1,
    entries => [
      69, -2397.33
    ]
  },
```

```
{
    action => 1,
    numEntries => 1,
    entries => [
      17, -2252.1
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      97, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      87, -2542.55
    ]
  },
  {
    action => 2,
    numEntries => 1,
    entries => [
      27, -2300.51
    ]
  },
  {
    action => 3,
    numEntries => 1,
    entries => [
      15, -2348.92
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      67, -2494.14
    ]
  },
  {
    action => 0,
    numEntries => 1,
    entries => [
      7, -2203.69
    ]
  },
```

```
{
    action => 4,
    numEntries => 1,
    entries => [
      47, -2397.33
    ]
  },
  {
    action => 5,
    numEntries => 1,
    entries => [
      57, -2445.73
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      77, -2542.55
    ]
  },
  {
    action => 1,
    numEntries => 1,
    entries => [
      39, -2252.1
    ]
  },
  {
    action => 3,
    numEntries => 1,
    entries => [
      37, -2348.92
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      50, -2542.55
    ]
  },
  {
    action => 2,
    numEntries => 1,
    entries => [
      16, -2300.51
    ]
  },
```

```
{
    action => 5,
    numEntries => 1,
    entries => [
      79, -2445.73
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      1, -2494.14
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      98, -2542.55
    ]
  },
  {
    action => 0,
    numEntries => 1,
    entries => [
      29, -2203.69
    ]
  },
  {
    action => 5,
    numEntries => 1,
    entries => [
      2, -2445.73
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      89, -2494.14
    ]
  },
  {
    action => 4,
    numEntries => 1,
    entries => [
      3, -2397.33
    ]
  },
```

```
{
    action => 3,
    numEntries => 1,
    entries => [
      4, -2348.92
    ]
  },
  {
    action => 0,
    numEntries => 1,
    entries => [
      18, -2203.69
    ]
  },
  {
    action => 2,
    numEntries => 1,
    entries => [
      5, -2300.51
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      99, -2542.55
    ]
  },
  {
    action => 1,
    numEntries => 1,
    entries => [
      6, -2252.1
    ]
  },
  {
    action => 0,
    numEntries => 1,
    entries => [
      19, -2155.65
    ]
  },
  {
    action => 0,
    numEntries => 1,
    entries => [
      9, -2111.16
    ]
  },
```

```
{
    action => 7,
    numEntries => 1,
    entries => [
      60, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      30, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      70, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      20, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      80, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      90, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      10, -2542.55
    ]
  },
```

```
{
    action => 7,
    numEntries => 1,
    entries => [
      41, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      51, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      42, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      52, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      61, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      31, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      43, -2542.55
    ]
  },
```

```
{
  action => 7,
  numEntries => 1,
  entries => [
    53, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    62, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    32, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    44, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    54, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    63, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    71, -2542.55
  ]
 },
```

```
{
    action => 7,
    numEntries => 1,
    entries => [
      33, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      55, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      64, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      72, -2542.55
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      45, -2494.14
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      34, -2494.14
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      21, -2542.55
    ]
  },
```

```
{
  action => 7,
  numEntries => 1,
  entries => [
    73, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    65, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    40, -2542.55
  ]
 },
 {
  action => 7,
  numEntries => 1,
  entries => [
    22, -2542.55
  ]
 },
 {
  action => 6,
  numEntries => 1,
  entries => [
    56, -2494.14
  ]
 },
 {
  action => 5,
  numEntries => 1,
  entries => [
    46, -2445.73
  ]
 },
 {
  action => 5,
  numEntries => 1,
  entries => [
    35, -2445.73
  ]
 },
```

```
{
    action => 7,
    numEntries => 1,
    entries => [
      74, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      81, -2542.55
    ]
  },
  {
    action => 6,
    numEntries => 1,
    entries => [
      23, -2494.14
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      66, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      82, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      0, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      75, -2542.55
    ]
  },
```

```
{
    action => 4,
    numEntries => 1,
    entries => [
      36, -2397.33
    ]
  },
  {
    action => 5,
    numEntries => 1,
    entries => [
      24, -2445.73
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      83, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      76, -2542.55
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      84, -2542.55
    ]
  },
  {
    action => 4,
    numEntries => 1,
    entries => [
      25, -2397.33
    ]
  },
  {
    action => 7,
    numEntries => 1,
    entries => [
      91, -2542.55
    ]
  },
```

```
{
  action => 7,
  numEntries => 1,
  entries => [
    85, -2542.55
  ]
},
{
  action => 4,
  numEntries => 1,
  entries => [
    58, -2397.33
  ]
},
{
  action => 0,
  numEntries => 1,
  entries => [
    8, -2155.64
  ]
},
{
  action => 0,
  numEntries => 100,
  entries => [
    0, -2754.25,
    1, -2657.68,
    2, -2563.2,
    3, -2473.31,
    4, -2391.5,
    5, -2319.56,
    6, -2257.51,
    7, -2203.69,
    8, -2155.64,
    9, -2111.16,
    10, -2754.25,
    11, -2754.25,
    12, -2657.68,
    13, -2563.2,
    14, -2473.31,
    15, -2391.5,
    16, -2319.56,
    17, -2257.51,
    18, -2203.69,
    19, -2155.64,
    20, -2754.25,
    21, -2754.25,
    22, -2754.25,
    23, -2657.68,
    24, -2563.2,
```

25, -2473.31,
26, -2391.5,
27, -2319.56,
28, -2257.51,
29, -2203.69,
30, -2754.25,
31, -2754.25,
32, -2754.25,
33, -2754.25,
34, -2657.68,
35, -2563.2,
36, -2473.31,
37, -2391.5,
38, -2319.56,
39, -2257.51,
40, -2754.25,
41, -2754.25,
42, -2754.25,
43, -2754.25,
44, -2754.25,
45, -2657.68,
46, -2563.2,
47, -2473.31,
48, -2391.5,
49, -2319.56,
50, -2754.25,
51, -2754.25,
52, -2754.25,
53, -2754.25,
54, -2754.25,
55, -2754.25,
56, -2657.68,
57, -2563.2,
58, -2473.31,
59, -2391.5,
60, -2754.25,
61, -2754.25,
62, -2754.25,
63, -2754.25,
64, -2754.25,
65, -2754.25,
66, -2754.25,
67, -2657.68,
68, -2563.2,
69, -2473.31,
70, -2754.25,
71, -2754.25,
72, -2754.25,
73, -2754.25,
74, -2754.25,

```
        75, -2754.25,
        76, -2754.25,
        77, -2754.25,
        78, -2657.68,
        79, -2563.2,
        80, -2754.25,
        81, -2754.25,
        82, -2754.25,
        83, -2754.25,
        84, -2754.25,
        85, -2754.25,
        86, -2754.25,
        87, -2754.25,
        88, -2754.25,
        89, -2657.68,
        90, -2754.25,
        91, -2754.25,
        92, -2754.25,
        93, -2754.25,
        94, -2754.25,
        95, -2754.25,
        96, -2754.25,
        97, -2754.25,
        98, -2754.25,
        99, -2754.25
      ]
    }
  ]
}
```

EK A.4

```
{
 policyType => "MaxPlanesLowerBound",
 numPlanes => 101,
 planes => [
  {
   action => 7,
   numEntries => 2,
   entries => [
    23, -2193.23,
    123, -2235.92
   ]
  },
  {
   action => 1,
   numEntries => 2,
   entries => [
    29, -1893.55,
    129, -2020.93
   ]
  },
  {
   action => 7,
   numEntries => 2,
   entries => [
    67, -2193.22,
    167, -2235.92
   ]
  },
  {
   action => 8,
   numEntries => 2,
   entries => [
    64, -2243.16,
    164, -2271.75
   ]
  },
  {
   action => 5,
   numEntries => 2,
   entries => [
    47, -2093.33,
    147, -2164.25
   ]
  },
```

```
{
 action => 7,
 numEntries => 2,
 entries => [
  89, -2193.21,
  189, -2235.91
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  53, -2243.16,
  153, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  91, -2243.16,
  191, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  87, -2243.16,
  187, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  95, -2243.15,
  195, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  97, -2243.15,
  197, -2271.74
 ]
},
```

```
{
  action => 5,
  numEntries => 2,
  entries => [
    69, -2093.32,
    169, -2164.24
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    73, -2243.15,
    173, -2271.74
  ]
},
{
  action => 3,
  numEntries => 2,
  entries => [
    27, -1993.43,
    127, -2092.58
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    59, -2043.37,
    159, -2128.41
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    50, -2243.15,
    150, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    55, -2243.15,
    155, -2271.74
  ]
},
```

```
{
  action => 3,
  numEntries => 2,
  entries => [
    5, -1993.43,
    105, -2092.58
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    98, -2243.15,
    198, -2271.74
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    78, -2193.21,
    178, -2235.9
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    44, -2243.15,
    144, -2271.74
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    48, -2043.37,
    148, -2128.41
  ]
},
{
  action => 3,
  numEntries => 2,
  entries => [
    38, -1993.43,
    138, -2092.58
  ]
},
```

```
{
  action => 0,
  numEntries => 2,
  entries => [
    8, -1843.59,
    108, -1985.09
  ]
},
{
  action => 1,
  numEntries => 2,
  entries => [
    18, -1893.54,
    118, -2020.92
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    88, -2243.15,
    188, -2271.74
  ]
},
{
  action => 6,
  numEntries => 2,
  entries => [
    68, -2143.26,
    168, -2200.07
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    74, -2243.15,
    174, -2271.74
  ]
},
{
  action => 2,
  numEntries => 2,
  entries => [
    28, -1943.48,
    128, -2056.75
  ]
},
```

```
{
  action => 5,
  numEntries => 2,
  entries => [
    58, -2093.32,
    158, -2164.24
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    83, -2243.15,
    183, -2271.74
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    37, -2043.37,
    137, -2128.41
  ]
},
{
  action => 2,
  numEntries => 2,
  entries => [
    39, -1943.48,
    139, -2056.75
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    92, -2243.15,
    192, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    11, -2243.15,
    111, -2271.74
  ]
},
```

```
{
  action => 6,
  numEntries => 2,
  entries => [
    35, -2143.26,
    135, -2200.07
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    31, -2243.15,
    131, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    54, -2243.15,
    154, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    96, -2243.15,
    196, -2271.74
  ]
},
{
  action => 2,
  numEntries => 2,
  entries => [
    17, -1943.48,
    117, -2056.75
  ]
},
{
  action => 0,
  numEntries => 2,
  entries => [
    19, -1843.59,
    119, -1985.09
  ]
},
```

```
{
 action => 8,
 numEntries => 2,
 entries => [
  93, -2243.15,
  193, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  84, -2243.15,
  184, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  99, -2243.15,
  199, -2271.74
 ]
},
{
 action => 6,
 numEntries => 2,
 entries => [
  57, -2143.26,
  157, -2200.07
 ]
},
{
 action => 5,
 numEntries => 2,
 entries => [
  25, -2093.32,
  125, -2164.24
 ]
},
{
 action => 1,
 numEntries => 2,
 entries => [
  7, -1893.54,
  107, -2020.92
 ]
},
```

```
{
  action => 5,
  numEntries => 2,
  entries => [
    14, -2093.32,
    114, -2164.24
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    45, -2193.21,
    145, -2235.9
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    75, -2243.15,
    175, -2271.74
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    12, -2193.21,
    112, -2235.9
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    90, -2243.15,
    190, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    30, -2243.15,
    130, -2271.74
  ]
},
```

```
{
  action => 8,
  numEntries => 2,
  entries => [
    65, -2243.15,
    165, -2271.74
  ]
},
{
  action => 2,
  numEntries => 2,
  entries => [
    6, -1943.48,
    106, -2056.75
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    94, -2243.15,
    194, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    77, -2243.15,
    177, -2271.74
  ]
},
{
  action => 6,
  numEntries => 2,
  entries => [
    13, -2143.26,
    113, -2200.07
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    32, -2243.15,
    132, -2271.74
  ]
},
```

```
{
  action => 8,
  numEntries => 2,
  entries => [
    40, -2243.15,
    140, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    85, -2243.15,
    185, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    20, -2243.15,
    120, -2271.74
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    15, -2043.37,
    115, -2128.41
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    60, -2243.15,
    160, -2271.74
  ]
},
{
  action => 5,
  numEntries => 2,
  entries => [
    36, -2093.32,
    136, -2164.24
  ]
},
```

```
{
  action => 3,
  numEntries => 2,
  entries => [
    16, -1993.43,
    116, -2092.58
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    80, -2243.15,
    180, -2271.74
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    1, -2193.21,
    101, -2235.9
  ]
},
{
  action => 6,
  numEntries => 2,
  entries => [
    2, -2143.26,
    102, -2200.07
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    10, -2243.15,
    110, -2271.74
  ]
},
{
  action => 5,
  numEntries => 2,
  entries => [
    3, -2093.32,
    103, -2164.24
  ]
},
```

```
{
  action => 8,
  numEntries => 2,
  entries => [
    51, -2243.15,
    151, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    61, -2243.15,
    161, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    43, -2243.15,
    143, -2271.74
  ]
},
{
  action => 6,
  numEntries => 2,
  entries => [
    46, -2143.26,
    146, -2200.07
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    26, -2043.37,
    126, -2128.41
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    56, -2193.21,
    156, -2235.9
  ]
},
```

```
{
 action => 8,
 numEntries => 2,
 entries => [
  0, -2243.15,
  100, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  41, -2243.15,
  141, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  66, -2243.15,
  166, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  21, -2243.15,
  121, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  52, -2243.15,
  152, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  81, -2243.15,
  181, -2271.74
 ]
},
```

```
{
  action => 6,
  numEntries => 2,
  entries => [
    24, -2143.26,
    124, -2200.07
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    42, -2243.15,
    142, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    72, -2243.15,
    172, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    71, -2243.15,
    171, -2271.74
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    63, -2243.15,
    163, -2271.74
  ]
},
{
  action => 7,
  numEntries => 2,
  entries => [
    34, -2193.21,
    134, -2235.9
  ]
},
```

```
{
 action => 8,
 numEntries => 2,
 entries => [
  22, -2243.15,
  122, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  33, -2243.15,
  133, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  86, -2243.15,
  186, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  76, -2243.15,
  176, -2271.74
 ]
},
{
 action => 8,
 numEntries => 2,
 entries => [
  70, -2243.15,
  170, -2271.74
 ]
},
{
 action => 6,
 numEntries => 2,
 entries => [
  79, -2143.26,
  179, -2200.07
 ]
},
```

```
{
  action => 3,
  numEntries => 2,
  entries => [
    49, -1993.43,
    149, -2092.58
  ]
},
{
  action => 0,
  numEntries => 2,
  entries => [
    9, -1805.93,
    109, -1941.36
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    82, -2243.15,
    182, -2271.74
  ]
},
{
  action => 4,
  numEntries => 2,
  entries => [
    4, -2043.37,
    104, -2128.41
  ]
},
{
  action => 8,
  numEntries => 2,
  entries => [
    62, -2243.15,
    162, -2271.74
  ]
},
{
  action => 0,
  numEntries => 200,
  entries => [
    0, -2330.56,
    1, -2233.77,
    2, -2146.3,
    3, -2072.81,
    4, -2013.28,
    5, -1964.24,
```

6, -1921.47,
7, -1881.86,
8, -1843.59,
9, -1805.93,
10, -2330.56,
11, -2330.56,
12, -2233.77,
13, -2146.3,
14, -2072.81,
15, -2013.28,
16, -1964.24,
17, -1921.47,
18, -1881.86,
19, -1843.59,
20, -2330.56,
21, -2330.56,
22, -2330.56,
23, -2233.77,
24, -2146.3,
25, -2072.81,
26, -2013.28,
27, -1964.24,
28, -1921.47,
29, -1881.86,
30, -2330.56,
31, -2330.56,
32, -2330.56,
33, -2330.56,
34, -2233.77,
35, -2146.3,
36, -2072.81,
37, -2013.28,
38, -1964.24,
39, -1921.47,
40, -2330.56,
41, -2330.56,
42, -2330.56,
43, -2330.56,
44, -2330.56,
45, -2233.77,
46, -2146.3,
47, -2072.81,
48, -2013.28,
49, -1964.24,
50, -2330.56,
51, -2330.56,
52, -2330.56,
53, -2330.56,
54, -2330.56,
55, -2330.56,

56, -2233.77,
57, -2146.3,
58, -2072.81,
59, -2013.28,
60, -2330.56,
61, -2330.56,
62, -2330.56,
63, -2330.56,
64, -2330.56,
65, -2330.56,
66, -2330.56,
67, -2233.77,
68, -2146.3,
69, -2072.81,
70, -2330.56,
71, -2330.56,
72, -2330.56,
73, -2330.56,
74, -2330.56,
75, -2330.56,
76, -2330.56,
77, -2330.56,
78, -2233.77,
79, -2146.3,
80, -2330.56,
81, -2330.56,
82, -2330.56,
83, -2330.56,
84, -2330.56,
85, -2330.56,
86, -2330.56,
87, -2330.56,
88, -2330.56,
89, -2233.77,
90, -2330.56,
91, -2330.56,
92, -2330.56,
93, -2330.56,
94, -2330.56,
95, -2330.56,
96, -2330.56,
97, -2330.56,
98, -2330.56,
99, -2330.56,
100, -2505.74,
101, -2434.09,
102, -2362.57,
103, -2291.56,
104, -2221.89,
105, -2154.93,

106, -2092.28,
107, -2035.4,
108, -1985.09,
109, -1941.36,
110, -2505.74,
111, -2505.74,
112, -2434.09,
113, -2362.57,
114, -2291.56,
115, -2221.89,
116, -2154.93,
117, -2092.28,
118, -2035.4,
119, -1985.09,
120, -2505.74,
121, -2505.74,
122, -2505.74,
123, -2434.09,
124, -2362.57,
125, -2291.56,
126, -2221.89,
127, -2154.93,
128, -2092.28,
129, -2035.4,
130, -2505.74,
131, -2505.74,
132, -2505.74,
133, -2505.74,
134, -2434.09,
135, -2362.57,
136, -2291.56,
137, -2221.89,
138, -2154.93,
139, -2092.28,
140, -2505.74,
141, -2505.74,
142, -2505.74,
143, -2505.74,
144, -2505.74,
145, -2434.09,
146, -2362.57,
147, -2291.56,
148, -2221.89,
149, -2154.93,
150, -2505.74,
151, -2505.74,
152, -2505.74,
153, -2505.74,
154, -2505.74,
155, -2505.74,

```
        156, -2434.09,
        157, -2362.57,
        158, -2291.56,
        159, -2221.89,
        160, -2505.74,
        161, -2505.74,
        162, -2505.74,
        163, -2505.74,
        164, -2505.74,
        165, -2505.74,
        166, -2505.74,
        167, -2434.09,
        168, -2362.57,
        169, -2291.56,
        170, -2505.74,
        171, -2505.74,
        172, -2505.74,
        173, -2505.74,
        174, -2505.74,
        175, -2505.74,
        176, -2505.74,
        177, -2505.74,
        178, -2434.09,
        179, -2362.57,
        180, -2505.74,
        181, -2505.74,
        182, -2505.74,
        183, -2505.74,
        184, -2505.74,
        185, -2505.74,
        186, -2505.74,
        187, -2505.74,
        188, -2505.74,
        189, -2434.09,
        190, -2505.74,
        191, -2505.74,
        192, -2505.74,
        193, -2505.74,
        194, -2505.74,
        195, -2505.74,
        196, -2505.74,
        197, -2505.74,
        198, -2505.74,
        199, -2505.74
      ]
    }
  ]
}
```

EK A.5

>>> begin
sim: 92 0 70 70
belief: 70:1
sim: 70 7 27 27
belief: 27:1
sim: 27 2 97 97
belief: 97:1
sim: 97 7 57 57
belief: 57:1
sim: 57 5 77 77
belief: 77:1
sim: 77 7 37 37
belief: 37:1
sim: 37 3 47 47
belief: 47:1
sim: 47 4 37 37
belief: 37:1
sim: 37 3 57 57
belief: 57:1
sim: 57 5 47 47
belief: 47:1
sim: 47 4 37 37
belief: 37:1
sim: 37 3 47 47
belief: 47:1
sim: 47 4 57 57
belief: 57:1
sim: 57 5 47 47
belief: 47:1
sim: 47 4 37 37
belief: 37:1
sim: 37 3 27 27
belief: 27:1
sim: 27 2 57 57
belief: 57:1
sim: 57 5 27 27
belief: 27:1
sim: 27 2 37 37
belief: 37:1
sim: 37 3 57 57
belief: 57:1
sim: 57 5 97 97
belief: 97:1
sim: 97 7 77 77
belief: 77:1
sim: 77 7 27 27
belief: 27:1
sim: 27 2 77 77
belief: 77:1

sim: 77 7 67 67
belief: 67:1
sim: 67 6 27 27
belief: 27:1
sim: 27 2 47 47
belief: 47:1
sim: 47 4 97 97
belief: 97:1
sim: 97 7 97 97
belief: 97:1
sim: 97 7 17 17
belief: 17:1
sim: 17 1 37 37
belief: 37:1
sim: 37 3 37 37
belief: 37:1
sim: 37 3 37 37
belief: 37:1
sim: 37 3 37 37
belief: 37:1
sim: 37 3 77 77
belief: 77:1
sim: 77 7 47 47
belief: 47:1
sim: 47 4 57 57
belief: 57:1
sim: 57 5 27 27
belief: 27:1
sim: 27 2 47 47
belief: 47:1
sim: 47 4 37 37
belief: 37:1
sim: 37 3 27 27
belief: 27:1
sim: 27 2 67 67
belief: 67:1
sim: 67 6 67 67
belief: 67:1
sim: 67 6 47 47
belief: 47:1
sim: 47 4 37 37
belief: 37:1
sim: 37 3 87 87
belief: 87:1
sim: 87 7 47 47
belief: 47:1
sim: 47 4 17 17
belief: 17:1
sim: 17 1 47 47
belief: 47:1

EK A.6

>>> begin
sim: 75 0 30 130
belief: 30:0.5 130:0.5
sim: 30 8 128 128
belief: 28:0.5 128:0.5
sim: 128 2 198 198
belief: 98:0.5 198:0.5
sim: 198 8 78 78
belief: 78:0.5 178:0.5
sim: 78 7 58 158
belief: 58:0.5 158:0.5
sim: 58 5 28 128
belief: 28:0.5 128:0.5
sim: 28 2 118 118
belief: 18:0.5 118:0.5
sim: 118 1 188 188
belief: 88:0.5 188:0.5
sim: 188 8 178 78
belief: 78:0.5 178:0.5
sim: 178 7 38 138
belief: 38:0.5 138:0.5
sim: 38 3 28 128
belief: 28:0.5 128:0.5
sim: 28 2 128 128
belief: 28:0.5 128:0.5
sim: 128 2 68 168
belief: 68:0.5 168:0.5
sim: 68 6 128 28
belief: 28:0.5 128:0.5
sim: 128 2 58 158
belief: 58:0.5 158:0.5
sim: 58 5 118 118
belief: 18:0.5 118:0.5
sim: 118 1 78 78
belief: 78:0.5 178:0.5
sim: 78 7 148 48
belief: 48:0.5 148:0.5
sim: 148 4 38 38
belief: 38:0.5 138:0.5
sim: 38 3 38 138
belief: 38:0.5 138:0.5
sim: 38 3 18 118
belief: 18:0.5 118:0.5
sim: 18 1 48 148
belief: 48:0.5 148:0.5
sim: 48 4 48 48
belief: 48:0.5 148:0.5
sim: 48 4 118 18
belief: 18:0.5 118:0.5

sim: 118 1 98 98
belief: 98:0.5 198:0.5
sim: 98 8 78 178
belief: 78:0.5 178:0.5
sim: 78 7 148 48
belief: 48:0.5 148:0.5
sim: 148 4 88 188
belief: 88:0.5 188:0.5
sim: 88 8 148 48
belief: 48:0.5 148:0.5
sim: 148 4 68 168
belief: 68:0.5 168:0.5
sim: 68 6 58 158
belief: 58:0.5 158:0.5
sim: 58 5 38 138
belief: 38:0.5 138:0.5
sim: 38 3 148 48
belief: 48:0.5 148:0.5
sim: 148 4 188 188
belief: 88:0.5 188:0.5
sim: 188 8 78 78
belief: 78:0.5 178:0.5
sim: 78 7 48 148
belief: 48:0.5 148:0.5
sim: 48 4 88 88
belief: 88:0.5 188:0.5
sim: 88 8 148 148
belief: 48:0.5 148:0.5
sim: 148 4 188 88
belief: 88:0.5 188:0.5
sim: 188 8 198 198
belief: 98:0.5 198:0.5
sim: 198 8 38 138
belief: 38:0.5 138:0.5
sim: 38 3 148 148
belief: 48:0.5 148:0.5
sim: 148 4 148 48
belief: 48:0.5 148:0.5
sim: 148 4 188 88
belief: 88:0.5 188:0.5
sim: 188 8 78 78
belief: 78:0.5 178:0.5
sim: 78 7 68 168
belief: 68:0.5 168:0.5
sim: 68 6 128 128
belief: 28:0.5 128:0.5
sim: 128 2 98 98
belief: 98:0.5 198:0.5
sim: 98 8 148 148
belief: 48:0.5 148:0.5

ÖZGEÇMİŞ

Tülay Varol 1985 yılında İzmir'de doğdu. Lise öğrenimini İzmir 60. Yıl Anadolu Lisesi'nde tamamladı. 2003 yılında liseden mezun olup, aynı yıl Yıldız Teknik Üniversitesi Matematik Mühendisliği'ni kazandı ve 2007 yılında mezun oldu. 2007'de İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Programında yüksek lisans eğitimine başlayan Tülay Varol, bu bölümde eğitimine devam etmektedir.