T.C.

İNÖNÜ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

TAKVİYELİ ÖĞRENME İÇİN YAPAY ATOM ALGORİTMASI (A^3) KULLANIMI

Ahmet KARADOĞAN

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

EYLÜL 2014

MALATYA

Tezin Başlığı : Takviyeli Öğrenme İçin Yapay Atom Algoritması (A^3) Kullanımı

Tezi Hazırlayan : Ahmet KARADOĞAN

Sınav Tarihi : 30.09.2014

Yukarıda adı geçen tez jürimizce değerlendirilerek Bilgisayar Mühendisliği Ana Bilim Dalında Yüksek Lisans Tezi olarak kabul edilmiştir.

Sınav Jüri Üyeleri

Tez Danışmanı : Doç.Dr. Ali KARCI .....
İnönü Üniversitesi

Doç.Dr. Asım KAYGUSUZ .....
İnönü Üniversitesi

Yrd.Doç.Dr. A.Fatih KOCAMAZ .....
İnönü Üniversitesi

Prof.Dr. Mehmet ALPASLAN
Enstitü Müdürü

ONUR SÖZÜ

Yüksek Lisans Tezi olarak sunduğum "Takviyeli Öğrenme İçin Yapay Atom Algoritması (A^3) Kullanımı" başlıklı bu çalışmanın bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurmaksızın tarafımdan yazıldığını ve yararlandığım bütün kaynakların, hem metin içinde hem de kaynakça yöntemine uygun biçimde gösterilenlerden oluştuğunu belirtir, bunu onurumla doğrularım.

Ahmet KARADOĞAN

ÖZET

Yüksek Lisans Tezi

TAKVİYELİ ÖĞRENME İÇİN YAPAY ATOM ALGORİTMASI (A^3) KULLANIMI

Ahmet KARADOĞAN

İnönü Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

59+viii sayfa

2014

Danışman: Doç.Dr. Ali KARCI

Bu tezde, metasezgisel bir algoritma olan Yapay Atom Algoritmasının, Takviyeli Öğrenme Algoritması ile kullanılması ve bu algoritmanın Av Avcı problemi üzerinde uygulanarak bu tür problemlere uygun olup olmayacağının görülmesi amaçlanmıştır.

Bu çalışmada takviyeli öğrenme algoritmalarından olan Q öğrenme algoritması kullanılarak av avcı probleminin çözümü elde edilmiştir. Diğer metasezgisel algoritmaların birçoğunda olduğu gibi doğadan esinlenen ve kimyasal bileşiklerin oluşumuna dayanan bir algoritma olan Yapay atom algoritması, av avcı problemine uyarlanarak bir uygulama geliştirilmiştir.

Çalışmanın sonucunda Yapay Atom Algoritmasının problemin çözümünde başarılı olduğu ve bu tür optimizasyon problemlerine kolaylıkla uygulanabildiği görülmüştür.

ANAHTAR KELİMELER: Takviyeli Öğrenme, Q Öğrenme, Av Avcı Problemi, Yapay Atom Algoritması

ABSTRACT

Master Thesis

ARTIFICIAL ATOM ALGORITHM(A^3) FOR REINFORCEMENT LEARNING

Ahmet KARADOĞAN

İnönü University

Graduate School of Natural and Applied Sciences

Department of Computer Engineering

59+viii pages

2014

Supervisor: Assoc.Prof.Dr. Ali KARCI

In this thesis, it is intended to be seen the using Artificial Atom Algorithm which is a metaheuristics algorithm, with Reinforcement Learning Algorithm and whether it is appropriate for such problems by applying this algorithm on Prey Hunter problem.

In this study, it is obtained the solution of Prey Hunter problem using the Q Learning algorithm which is a Reinforcement learning algorithm. An application has been developed by applying Artificial Atom Algorithm to prey hunter problem which is an algorithm that based on naturel process as in many of metaheuristics algorithms and it is based on chemical process of compound formation.

In the end of the study, it has been observed that Algorithm Artificial Atoms is successful in solving the problem and it can be easily applied to such optimization problem.

KEYWORDS: Reinforcement Learning, Q Learning, Prey Hunter Problem, Artificial Atom Algorithm

TEŞEKKÜR

Bu tez çalışmasının her aşamasında yardım, öneri ve desteğini esirgemeden beni her konuda yönlendiren, sürekli motivasyonumu sağlayan ve bana değerli zamanını ayıran danışman hocam Sayın Doç. Dr. Ali KARCI'ya;

Ayrıca bu uzun tez yolculuğunda kendilerine yeterince zaman ayıramadığım halde benden desteklerini esirgemeyen değerli AİLEM'e

Teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR

s	Etmenin bulunduğu durum
S	Etmenin bulunduğu durumlar kümesi
a	Etmenin yapabileceği hareket
A	Q öğrenmede etmenin yapabileceği hareketler kümesi
$\delta(s,a)$	s durumundaki a hareketi ile davranıldığındaki durum
r(s,a)	s durumunda yapılan a hareketi sonrasındaki ödül
$Q(s,a)$	s durumundaki a hareketi sonrası etmenin Q değeri
β	Öğrenme katsayısı
γ	Azaltma katsayısı
V	Amaç fonksiyonu
R	Ödül fonksiyonu
T	Durum geçiş fonksiyonu
$\prod$	Davranış politikası
$P(a_i \mid s)$	Etmen s durumunda iken a_i hareketini seçme olasılığı
k	Etmenin deney politikasının katsayısı
A^3	Yapay Atom Algoritması (Artificial Atom Algorithm)
KA	Kovalent Alan
İA	İyonik Alan
A_i	i.Atom
E	Atomdaki her bir elektron
H	Avcı etmeni
P	Av etmeni

1. GİRİŞ

Makine öğrenmesi, makine/bilgisayarların daha önceki bilgilerden elde ettiği tecrübelere dayanarak, ilerde karşılaşacağı durumlar hakkında tahmini bilgilere sahip olmasına ve bu durumu modelleyebilmesine imkân veren bir yapay zekâ alanıdır (Yardımcı, 2011).

Makine öğrenmesinde amaç, edinilen tecrübelerden yararlanarak genelleştirme yapabilmektir. Bu şekilde makine karşılaştığı yeni durumlar karşısında karar verebilme yeteneğine sahip olabilecektir.

Öğrenme işlemi gerçekleştirilirken, etmen denilen varlıklar görev yapmaktadır. Etmen (agent) kavramı genel olarak, bulunduğu ortamı çeşitli algılayıcılar vasıtasıyla algılayıp ortamda çeşitli hareketlerde bulunabilen, etkileşim içinde olabilen bir varlık olarak tanımlanabilir.

Etmenlerde bulunması gereken özelliklerden bazıları öğrenme yeteneği, özerklik, işbirliği yapabilme ve tepkisellik olarak sıralanabilir.

Etmen vasıtasıyla öğrenme işlemi gerçekleştirilen gerçek hayattaki uygulamalara bakıldığında çoğu durumda tek etmenin yetersiz olduğu görülmektedir. Bu nedenle problemlerin çözümünde tek etmen yerine birden çok etmen kullanılarak problemin daha hızlı ve kolay bir şekilde çözülmesi sağlanmaktadır.

Çoklu etmen sistemleri, aynı ortam içerisinde birden fazla etmenin bir arada hareket ettiği sistemlerdir.

Öğrenme metotları genel olarak, Denetimli Öğrenme, Denetimsiz Öğrenme ve Takviyeli Öğrenme olmak üzere 3 ana gruba ayrılmaktadır. Bu tezde üzerinde durulan öğrenme yöntemi Takviyeli Öğrenme yöntemidir.

Takviyeli Öğrenme, bir etmenin bulunduğu ortamda algılama ve hareket yaparak, hedefine ulaşmasını sağlayan en uygun hareketleri nasıl öğrenip uygulayacağını gösterir.

Takviyeli öğrenmede etmene hangi hareketleri yapması gerektiğini söylemek yerine, ödül kazandıran hareketleri keşfetmesi sağlanır. Takviyeli öğrenmede ödül ve ceza işlemleri mevcut olup, etmenin her bir hareketine karşılık ortaya çıkan durum için ödül veya ceza sağlanır. Etmenin amacı, en büyük ödülü üretecek olan hareketlerin sırasını öğrenmektir. Burada etmen için kullanabileceği eğitim verileri bulunmaz. Bunun yerine sadece anlık ödüller kullanılarak öğrenmesi sağlanır.

Takviyeli öğrenme yaklaşımları içinde en çok bilinen ve kullanılan yöntem Q Öğrenme yöntemi olup, labirent ve arama problemlerinde sıklıkla kullanılmaktadır. Q öğrenmede, etmen için bulunduğu durumları ve yapılabilen hareketleri tutan bir tablo kullanılır. Öğrenme işleminin her bir adımında Q değerleri güncellenir ve etmenin bu tabloyu kullanarak en uygun hareketi seçmesi sağlanır.

Bu tez kapsamında, Q Öğrenme yöntemi ile av avcı probleminin çözülmesi uygulandıktan sonra, bir metasezgisel algoritma olan Yapay Atom Algoritması kullanılarak uygulama yapılmıştır.

Yapay Atom Algoritması (Artificial Atom Algorithm-A^3), diğer birçok metasezgisel algoritmalarda olduğu gibi doğadan esinlenerek ortaya çıkarılmış olan, kimyasal bileşiklere dayanan bir metasezgisel algoritmadır. Kimyasal bileşiklerin oluşumundaki İyonik Bağ ve Kovalent Bağ kimyasal işlemleri algoritmanın temel iki işlemini teşkil etmektedir.

Yapay atom algoritmasında problem bir atom kümesi olarak düşünülebilir. Problemin özelliklerini barındıran her bir parametre ise bir elektron olarak düşünülecektir. Başlangıç aşamasında her atom, problemin rassal bir çözümüdür. Birden fazla atom ile işleme başlanır ve bu çözümler kümesi Atom Kümesi olarak adlandırılmaktadır. Atom kümesindeki atomlarda bulunan elektronların amaç fonksiyonu üzerindeki etkilerin tutulduğu küme ise Elektron Etkileri Kümesi olarak adlandırılmaktadır.

Bu çalışma kapsamında üzerinde çalışılan av avcı problemi, çoklu etmen sistemleri için uygun bir öğrenme ortamı sunmaktadır. Av avcı probleminde amaç belirtilen sayıdaki avcıların avı yakalaması ve bunun için de ava yani hedefe giden en kısa yolu bulmasıdır. Bu açıdan bakıldığında Q Öğrenme yönteminin av avcı problemine uygulanabilir olduğu görülmektedir.

Yapay Atom Algoritmasının da bu tür problemlere uygulanabilir olup olmadığı, bu işlem için istenilen parametreleri optimize edip etmeyeceği konuları bu tezin başlıca amaçlarını oluşturmaktadır.

Bu tez çalışması şu şekilde düzenlenmiştir:

- Bölüm 2, makine öğrenmesi ve öğrenme metotları hakkında genel bilgiler içermektedir. Ayrıca öğrenme işleminde kullanılan etmenler hakkında bilgiler verilerek, etmenlerin özellikleri ve etmen çeşitleri, çoklu etmen sistemleri konuları üzerinde durulmuştur.
- Bölüm 3, takviyeli öğrenme ve Q Öğrenme konularını içermektedir. Q öğrenme algoritmasının nasıl çalıştığı bir örnek üzerinde anlatılarak öğrenme işleminin sonuçları gösterilmiştir.
- Bölüm 4, Yapay Atom Algoritması hakkında bilgilerden oluşmakta olup bu bölümde algoritmadaki işlemler üzerinde durulmuştur.
- Bölüm 5, tez çalışması kapsamında yapılan uygulamanın olduğu bölümdür. Bu bölümde Av avcı problemi, Q Öğrenme algoritması ile problemin çözümü, Yapay Atom Algoritmasının Av Avcı problemine uygulanması anlatılmıştır.

2. MAKİNE ÖĞRENMESİ VE ETMENLER

Makine öğrenmesi, geçmiş bilgilerden veya örnek verilerden yararlanarak herhangi bir performans kriterini optimize etmek için bilgisayarın programlanmasıdır (Alpaydın, 2010).

Makine öğrenmesini Arthur Samuel 1959 yılında şöyle tanımlanmıştır: "Bilgisayarlara tamamen programlama yapmadan öğrenme yeteneği veren çalışma alanıdır" (Simon, 2013).

Tom Mitchell ise şöyle tanımlamıştır : "Eğer bir bilgisayar programı, bir T görevini, P performansında yaparak, deneyiminde E artış sağlıyorsa, o bilgisayar programı, T görevinde, P performansıyla, E deneyimlerinden öğreniyordur denilir" (Mitchell, 1997).

Öğrenme işlemindeki temel amaç, edinilen tecrübelerden genelleştirme yapabilmektir. Böylece öğrenme işlemini yapan makine, bilinmeyen yeni durumlarla karşılaşıldığında ne yapılacağına karar verebilecek duruma gelecektir.

Makine Öğrenmesi, verilen bir problemi probleme ait ortamdan edinilen veriye göre modelleyen bilgisayar algoritmalarının genel adıdır. Yoğun çalışılan bir konu olduğu için önerilmiş birçok yaklaşım ve algoritma mevcuttur. Bu yaklaşımların bir kısmı tahmin (prediction) ve kestirim (estimation) bir kısmı da sınıflandırma (classification) yapabilme yeteneğine sahiptir.

Makine öğrenmesi, bilgisayarların geçmiş bilgilerden elde edilen tecrübelerden yararlanarak, gelecekteki olayları tahmin etmesine ve modelleme yapmasına imkan veren bir yapay zeka alanıdır. Bilgisayarın bir olay ile ilgili bilgileri ve tecrübeleri öğrenerek gelecekte oluşacak benzeri olaylar hakkında kararlar verebilmesi ve problemlere çözümler üretebilmesi olarak da tanımlanabilir. Makine öğrenimi araştırmalarının odaklandığı konu bilgisayarlara karmaşık örüntüleri algılama ve veriye dayalı akılcı kararlar verebilme becerisi kazandırmaktır. Bu, makine öğreniminin istatistik, olasılık kuramı, veri madenciliği, örüntü tanıma, yapay zekâ, uyarlamalı denetim ve kuramsal bilgisayar bilimi gibi alanlarla yakından ilintili olduğunu göstermektedir (Yardımcı, 2011).

Makine öğrenmesi ile bilgisayar yazılımlarının, olayları daha önceki örneklerden edinilmiş tecrübelerin kullanılarak öğrenmesi sağlanır. Öğrenecek olan bilgisayar sistemleri, önce bir örnek almakta ve bu örnekten bazı bilgileri öğrenmektedir. Daha sonra, ikinci örneğe bakarak biraz daha bilgi edinmektedir. Bu işlemi öğrenilecek olay ile ilgili genellemeler yapılmaktadır. Bu olaya tecrübelerden öğrenmenin bir yolu olarak bakmak mümkündür (Öztemel, 2003).

Makine öğrenmesinde, tümevarım kullanılarak çıkarımlar yapılmaktadır. Yapılan çıkarımlar, geleceğe yönelik tahminde bulunmak ya da bir tanım yapmak için kullanılır. Veri, geleceğe yönelik bilgi tahmini için kullanılacaksa, yani oluşturulacak model tahmin edici model ise, iki aşama gereklidir: eğitim aşaması ve test aşaması. Eğitim aşamasında, belirli miktarda veri kullanılarak bir model oluşturulur. Kullanılan veri, eğitim verisi olarak adlandırılır. Eğitim verisinin ne kadar ve nasıl seçileceği ayrı bir konudur. Oluşturulan model, sadece örnek veriyi değil tüm veriyi temsil eder. Test aşamasında ise, eğitim aşaması sonucunda oluşan modele, test aşaması için ayrılan ya da ileride toplanacak olan veriler sunulur. Ortaya çıkan bilgi ise tahmin etmek için kullanılır (Çentik, 2013).

Makine öğreniminin başlıca uygulamaları, makine algılaması, bilgisayarlı görme, doğal dil işleme, sözdizimsel örüntü tanıma, arama motorları, tıbbi tanı, biyoinformatik, beyin-makine ara yüzleri, kredi kartı dolandırıcılığı denetimi, borsa çözümlemesi, DNA dizilerinin sınıflandırılması, konuşma ve el yazısı tanıma, bilgisayarlı görmede nesne tanıma, oyun oynama, yazılım mühendisliği, uyarlamalı web siteleri ve robot gezisi gibi alanlardır (Anonymous, 2013).

Öğrenme metotları genel olarak 3 ana grupta toplanmaktadır:

1. Denetimli Öğrenme (Supervised Learning) : Öğrenme işlemini gerçekleştirecek olan etmene yapması gerekenleri gösteren eğitim örnekleri verilerek kendini eğitmesi sağlanır. Yapay sinir ağları algoritması bu sınıfa girmektedir.

2. Denetimsiz Öğrenme (Unsupervised Learning) : Öğrenme işlemi daha önceden belirlenmiş bir algoritmaya göre hareket ederek amacına ulaşır. Hangi durumda ne işlem yapılacağı bellidir.

3. Takviyeli Öğrenme (Reinforcement Learning) : Takviyeli Öğrenme; kendi ortamında algılama ve hareket yapan bağımsız bir etmenin, hedefine ulaşması için optimal hareketleri nasıl öğrenip uygulayacağını gösteren bir öğrenme türüdür.

2.1. Etmen Kavramı

Etmen kavramını şu şekilde tanımlayabiliriz; Belli bir çevre içinde konumlanmış, hedefi, hareketi ve çevre bilgisi olan bir varlıktır (Örneğin robot gibi) (Stone ve Veloso, 2000).

Etmen (agent) basit bir ifadeyle, bulunduğu ortamda (environment) algılayıcıları aracılığıyla algılayabilen ve bir eylemde bulunabilen bir varlıktır. Her bir etmen yaptığı işe göre değiştiğinden genel bir etmen tanımı yapmak her zaman için geçerli olmayabilir.

Etmen kavramının evrensel olarak kabul edilen bir tanımı yoktur ve bu konuda tartışmalar devam etmektedir. Bir etmene ait çeşitli özniteliklerin farklı ortamlarda farklı öneme sahip olması da zorluğun bir parçasıdır. Bazı uygulamalar için etmenin tecrübelerinden öğrenme yeteneği büyük önem taşırken, diğer bazı uygulamalarda öğrenme işlemi sadece önemsiz değil, istenmeyen bir durum da olabilir. Bununla birlikte, tanım yapmak önemlidir aksi halde kavramın tüm anlamını kaybetme tehlikesi bulunmaktadır. Burada şöyle bir tanım yapabiliriz; Bir etmen, belli bir ortamda konumlanmış ve hedefe ulaşmak için özerk hareket etme yeteneğine sahip bir bilgisayar sistemidir (Weiss, 1999).

Etmen bulunduğu ortamdaki durumları algılayabilen ve çevresindeki diğer etmenlerle etkileşim içinde bulunabilen bir varlıktır. Etmen mimarisi günümüzde ve gelecekteki bilgisayar uygulamalarında önemli görevler almaktadır. Etmenlerin diğer öğrenme algoritmalarına göre en önemli avantajlarından biri de birçok ortamla uyumlu çalışabilmeleri ve öğrenme için bir ön bilgiye ihtiyaç duymamalarıdır. Modern sistemleri sıkıntıya sokan geniş çevre uzayı, büyük veriler ve işlem yoğunluğundan dolayı oluşan karmaşık yapıların çözümünde etmen mimarisi önemli rol oynamaktadır (Hacıbeyoğlu, 2006).

2.2. Etmenin Özellikleri

Etmenlerin taşıması gereken özellikler, Öğrenme Yeteneği, Zeka, Özerklik, İşbirliği ve Tepkisellik şeklindedir.

- **Öğrenme Yeteneği**

Birçok etmen öğrenme yeteneğine sahiptir. Kendilerine yeni bir bilgi verildiği zaman bu verileri kullanışlı bir şekilde tutabilir. Örneğin etmenler bir kullanıcıdan hareketleri gözlemleyerek veya eğitim verilerek öğrenebilirler. Ayrıca çok etmenli sistemlerde etmenlerin haberleştirilmesi yoluyla diğer etmenlerden de öğrenebilirler.

Öğrenme işlemi etmenin zamanla tecrübelerini geliştirmelerini sağlar. Eğer kullanıcı etmene başarısız olduğunu bildirirse, etmenin sonraki süreçte aynı hataya düşmemesini öğrenmesi sağlanmış olur (Coppin, 2004).

Öğrenme işlemi problemin türüne göre farklılık gösterebilir. Bazı problemlerde öğrenme işleminin yapılabilmesi için etmenin eğitilmesi gerekmekte iken, bazı problemlerde etmen kendi kendine öğrenme işlemini yapabilmektedir.

- **Zekâ**

Etmen bir insan yerine görev yapabilen bir varlıktır. Örneğin, bir etmen bir hisse senedinin değeri istenen seviyenin altına düştüğünde onu satın alabilir.

Zeki etmenler, görev parametreleri değiştiği zaman veya beklenmedik bir durum meydana geldiği zaman görevlerini gerçekleştirmelerini sağlayan ek çevre bilgisine sahiptirler (Coppin, 2004).

Etmenler öğrenme yetenekleri sayesinde zamanla tecrübe kazanmakta ve yeni durumlarda ne yapmaları gerektiği konusunda bu tecrübelere başvurmaktadır. Öğrenen ve tecrübe edinen etmenler bu şekilde daha zeki hale gelmektedirler.

- **Özerklik**

Özerklik, etmenin kullanıcısından veya programcısından bağımsız olarak hareket etme ve karar verme yeteneğidir. Birçok etmenin zekâya ek olarak sahip oldukları önemli bir özelliktir. Örneğin, zeki bir alışveriş etmeni, bir kullanıcının adına kullanıcının neleri satın alabileceği hakkında kendisine sormadan karar verebilir. Bu özerklik, zeki etmenleri diğer birçok Yapay Zekâ tekniklerinden ayırmaktadır (Coppin, 2004).

Etmen öğrenme yeteneği sayesinde kendini geliştirip öğrendikleri ile tek başına karar verebilir duruma gelmektedir.

- **İşbirliği**

Çok etmenli sistemlerde, etmenler birbirleriyle işbirliği yapmaktadırlar. Bu işbirliği etmenler arasında bir sosyal etkileşim biçimi olduğunu gösterir. Etmenler aynı zamanda birbirlerinden öğrenebilirler. Örneğin bir satın alma etmeni diğer bir satın alma etmeninden yeni bir alışveriş ortamını öğrenip kullanabilmektedir. Aynı zamanda etmenler kendi kullanıcıları ile de işbirliği yapabilirler (Coppin, 2004).

Etmenlerin işbirliği yapmalarına en güzel örnek olarak av avcı problemleri verilebilir. Birden fazla avcı etmeni birbirleriyle işbirliği yaparak ve haberleşerek daha çabuk bulup yakalayabilmektedirler.

- **Tepkisellik**

Etmenler bulundukları ortamları, kullanıcıları veya diğer etmenleri anlayabilir ve bu ortamlarda meydana gelecek herhangi bir değişime karşı tepki verebilirler.

2.3. Etmen Çeşitleri

Etmenler genel olarak şu şekilde sınıflandırılabilir:

- Akıllı etmenler
- İşbirlikçi Etmenler
- Arabirim Etmenleri
- Hareketli Etmenler
- Bilgilendirme Etmenleri
- Tepkisel Etmenler
- Melez Etmenler

➢ Akıllı Etmenler

Akıllı yazılım etmenleri, kullanıcıların davranış biçimlerini öğrenerek, bir kimse tarafından belirlenen bilgiyi arayan ve tasnif eden veya bir kimse adına belirli işleri gerçekleştiren yazılım parçaları olarak da tanımlanabilir (Kaya, 1998).

➢ İşbirlikçi Etmenler

İşbirlikçi etmenler, bulundukları ortamda bulunan diğer etmenlerle beraber çalışan ve haberleşme yoluyla görevlerini yerine getiren etmenlerdir.

İşbirliği yapan etmenlerin görevlerinden bazıları şunlardır (Kaya, 1998) :

- Tek bir etmen için çok büyük olan problemlerin çözümünü sağlamak
- Çoklu uzman sistem ve karar destek sistemlerin birlikte çalışmasına izin vermek
- Dağıtık problemlerin çözümünü sağlamak
- Karmaşıklığı azaltmak

➢ Arabirim Etmenleri

Arabirim etmenleri, işbirlikçi etmenlerden farklı olarak diğer etmenler yerine kendi kullanıcısı ile haberleşen etmenlerdir. Arabirim etmenleri görevlerini yerine getirmek için gerekli bilgileri kullanıcısından öğrenir. Diğer etmenlerle işbirliği yapmaya ihtiyaç duymazlar.

Arabirim etmenleri kullanıcısı adına öğrenme işlemini gerçekleştirirken şu yolları izlemektedir (Hacıbeyoğlu, 2006) :

- Kullanıcıyı izleyip takip ederek
- Kullanıcıdan olumlu veya olumsuz geri bildirim alarak
- Kullanıcıdan direk talimat alarak
- Tavsiye amacıyla diğer etmenlere sorarak

➢ Hareketli Etmenler

Hareketli etmenler, bilgisayar veya internet gibi geniş alanlı ağlar üzerinde sürekli hareket eden ve bu şekilde kullanıcısı adına gerekli bilgileri toplayan ve yapması gereken görevleri yerine getirdikten sonra tekrar kullanıcısına dönen yazılım etmenleridir (Kaya, 1998).

Hareketli etmenlerin bir ağda gezebilmeleri ve işlem yapabilmeleri için bir program aracılığıyla kodlanmaları gerekir. Bunun için etmen tasarımcıları, bazı script dilleri kullanarak haberleşme işlemini gerçekleştirirler (Hacıbeyoğlu, 2006).

- **Bilgilendirme Etmenleri**

Bilgilendirme etmenleri internet üzerindeki çok geniş kaynaklar arasında kullanıcının aradığı bilgileri bulmayı sağlayan, genellikle internet üzerinde çalışan etmenlerdir.

İnternet etmenleri temel olarak üç grupta incelenebilir (Hacıbeyoğlu, 2006):

Web Arama Etmenleri: Arama motorları tarafından kullanılan etmenlerdir. Kullanıcı tarafından arama motoruna girilen anahtar kelimeleri internet üzerinden sorgulayarak buna uygun sonuçları listeler.

Bilgi Filtreleme Etmenleri: Kullanıcı tarafından verilen anahtar kelimeleri, web sayfası veya haber kaynakları gibi farklı kaynaklardan toplayıp filtreledikten sonra kullanıcıya sunan etmenlerdir.

Bildirim etmenleri: Kullanıcı için önemli olan olayları haber veren etmenlerdir. Örneğin belirli bir web sayfasında meydana gelen değişiklik veya kullanıcıya ait özel hatırlatmalar gibi.

- **Tepkisel Etmenler**

Tepkisel etmenler, kendilerine bir bilgi geldiği zaman buna karşılık olarak kendisine daha önce verilen kurallara göre bir eylem gerçekleştirir. Örneğin e-posta sistemlerinde kullanılan filtreleme sistemleri, gelen e-postaları sınıflandırırken daha önceden belirlenen kurallara göre hareket eder (Coppin, 2004).

- **Melez Etmenler**

Melez etmenler, yukarda belirtilen etmen çeşitlerinin birkaçının karışım yapılarak bir arada kullanılmasıyla oluşan etmenlerdir. Her bir etmen çeşidinin avantajlı ya da dezavantajlı olduğu problemler bulunmaktadır. Bazı problemlerde

birkaç farklı etmenin olumlu özellikleri birleştirilerek kullanılırsa daha etkili bir etmenin ortaya çıkması sağlanabilir.

2.4. Çoklu Etmen Sistemleri

Etmen sistemlerinin gerçek hayattaki uygulamalarına bakıldığında çoğu zaman tek bir etmenin yetersiz olduğu görülmektedir. Bu nedenle problemin çözümünde tek etmen yerine birden fazla etmen bir arada kullanılarak çoklu etmen sistemleri oluşturulmakta ve problemler daha hızlı ve daha kolay bir şekilde çözülebilmektedir.

Çoklu etmen sistemi, aynı ortamda bulunan birden fazla etmenden oluşan bir sistemdir. Çoklu etmen sistemindeki etmenler iki farklı şekilde davranabilirler. Birincisi etmenler amaçları doğrultusunda kendi planlarını geliştirmeyi tercih edebilirler, ikincisi ise etmenler beraber çalışarak ortak plan geliştirebilirler. Her iki durumda da etmenler dinamik ortamlardaki problemleri çözmek için yeterli olmayabilirler. Bu durumda etmenlerin bir öğrenme mekanizması tarafından desteklenmesi gerekmektedir (Kaya, 2003).

Çoklu etmen sistemlerinde işlemler ve veriler tek bir etmen üzerine yüklenmeyip birden fazla etmen arasında dağıtıldığı için hem modülerlik hem de güvenilirlik sağlanmış olmaktadır. Çoklu etmen sistemindeki her bir etmen tek başına problemi çözmek için gerekli tüm bilgiye sahip değildir.

Birden fazla etmene ihtiyaç duyulan bazı durumlar şu şekilde verilebilir (Bilgin, 2013) :

- İşbirliği gerektiren durumlar
- Rekabet veya taraf bulunduran durumlar
- Tek etmenle yapılması çok uzun süren veya tek etmenle başarılı olunmayan durumlarda birden fazla etmen kullanılarak görevleri bölmek ve kaynak dağılımını sağlamak gereken durumlar

3. TAKVİYELİ ÖĞRENME

Takviyeli Öğrenme (Reinforcement Learning), kendi ortamında algılama ve hareket yapan bağımsız bir etmenin, hedefine ulaşması için en uygun hareketleri nasıl öğrenip uygulayacağını gösteren bir öğrenme türüdür.

Takviyeli öğrenme, sayısal bir ödül sinyalini en üst düzeye çıkarmak için ne yapmak- eylemlerle durumları nasıl eşleştirmek- gerektiğini gösteren bir öğrenmedir. Öğrenen etmene, makine öğrenmesinin birçok yöntemindeki gibi hangi hareketleri yapması gerektiği söylenmez, bunun yerine hangi hareketlerin en çok ödülü kazandıracağını, onları deneyerek keşfetmesi söylenir. Birçok farklı ve değişken durumlarda hareketler sadece anlık ödülleri değil, aynı zamanda bir sonraki durumları ve böylece sonraki tüm ödülleri etkileyebilir. Bu iki özellik-deneme ve yanılma-gecikmiş ödül- takviyeli öğrenmenin en önemli ayırt edici özellikleridir. (Sutton ve Barto, 2012).

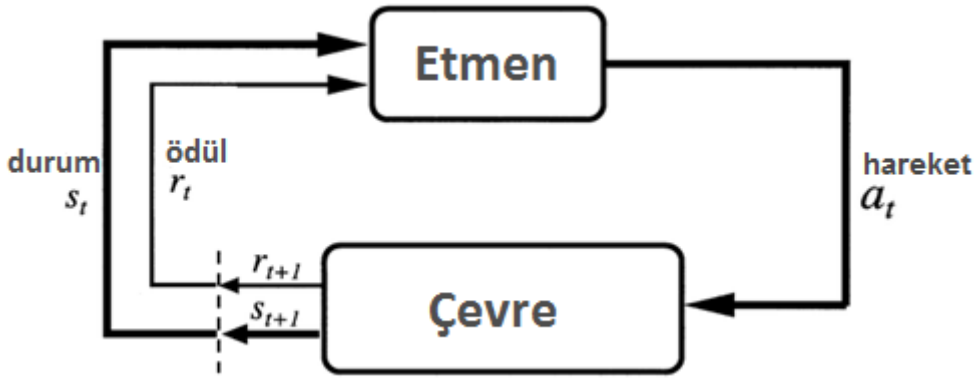
Takviyeli öğrenmeyi eğiticili öğrenmeden (supervised learning) ayıran en belirgin özelliği, etmenin tahminleri için etmene sadece bir kısım geribildirim verilir. Ayrıca buradaki tahminler kontrol edilen sistemin gelecekteki durumu üzerinde uzun vadeli etkili olabilir. Böylece zaman önemli bir faktör olmaktadır (Szepesvari, 2009).

Takviyeli öğrenmede ödül ve ceza işlemleri mevcuttur. Bir ortamda bulunan etmenin ortamdaki her bir hareketine karşılık ortaya çıkan durum için bir ödül veya ceza sağlanır. Etmenin buradaki amacı en büyük ödülü üreten hareketlerin sırasını öğrenmektir. Örneğin bir oyun içindeki etmen için, oyun kazanıldığı durumlarda pozitif bir ödül, kaybedildiği durumlarda negatif bir ödül(ceza), diğer durumlarda sıfır ödül alabilir.

Takviyeli öğrenme algoritmaları robotik uygulamaları, fabrikadaki işlemlerin optimizasyonu, oyun uygulamaları gibi birçok alandaki problemleri çözmede kullanılabilmektedir.

Takviyeli öğrenme metotları, hakkında bilgi sahibi olamadıkları belirsiz ortamlarda uygulanabilirler. Takviyeli öğrenme metotları çevresiyle etkileşim içinde bulunarak tecrübesini arttırır. Sürekli ödül alan hareketleri seçerek başarısını maksimum değere çıkarmaya çalışır.

Takviyeli öğrenme sistemi genel olarak Şekil 3.1'de gösterildiği gibi bir modele sahiptir.



Şekil 3.1 Takviyeli öğrenme sisteminin genel modeli

Şekil 3.1 'den de görüldüğü gibi, etmen(agent) yaptığı her bir hareket (action) sonrası yeni bir duruma(state) geçer ve bulunduğu ortamdan(environment) buna karşılık bir ödül alarak hareketine devam eder. Burada yapılan işlemler genel olarak şu şekilde belirtilebilir (Hacıbeyoğlu, 2006) :

- Etmen ile çevre birçok zaman aralığında etkileşim içinde bulunurlar.

$$t = 0, 1, 2, \dots$$

- Etmenin her bir zaman aralığında bulunduğu durumlar

$$s_t \in S$$

şeklinde gösterilebilir. S, etmenin bulunduğu durumları içeren bir durum kümesidir.

$$S = \{s_1, s_2, s_3, \dots\}$$

- Her bir s durumunda, etmen yeni bir hareket seçerek işlemine devam eder.

$$a_t \in A(s_t)$$

- Etmenin s_t durumunda yapabileceği hareketler kümesi şu şekilde gösterilir:

$$A(s_t) = \{a_1, a_2, a_3, \dots\}$$

- Seçilen yeni hareket sonrası etmen bir ödül alarak yeni durumuna geçer.

Takviyeli öğrenme ile problemleri çözmek için iki temel yöntem vardır. İlk yöntem, ortamı iyiye götüren hareketi bulmak için davranış uzayında bir arama yapmak, ikinci yöntem ise, faydalı hareketi tahmin etmek için istatistik ve dinamik programlamayı kullanmaktır (Sutton ve Barto, 2012).

Takviyeli öğrenmeyi diğer yöntemlerden ayıran bazı özellikler şunlardır (Mitchell, 1997):

- Gecikmiş Ödül: Etmenin amacı, şu andaki s durumundan, optimal hareket $a = \pi(s)$'i planlayan bir hedef fonksiyon öğrenmektir. Diğer birçok öğrenme yöntemlerinde π gibi bir hedef fonksiyon öğrenildiği zaman her bir eğitme örneği $(s, \pi(s))$ şeklindedir. Takviyeli öğrenmede ise eğitme bilgisi bu formda değildir. Bunun yerine eğitici; etmen hareketlerini yürütürken sadece anlık ödül değerlerin sırasını sağlar.

- Kısmen Gözlenebilir Durumlar: Normalde teorik olarak herhangi bir zaman adımında etmen sensörlerinin ortamın bütün durumunu algılayabildiğini kabul etmek uygun olsa da, aslında birçok uygulama durumunda sensörler yeterince bilgi sağlayamaz. Örneğin, sadece önünü görmeye yarayan kameraya sahip bir robot arkasında ne olduğunu idrak edemez. Böyle durumlarda hareketler seçildiği zaman

şu andaki gözlemlerle birlikte öncekileri de dikkate almak faydalı olabilir.

- Yaşam Boyu Öğrenme: Robot öğrenme aynı sensörler kullanarak aynı ortamda birden fazla ilgili işi öğrenmeyi içerebilir. Örneğin, hareketli bir robotun aynı anda, şarj edicisine yaklaşma, dar koridorlar boyunca gezinme ve lazer yazıcıdan çıktıları toplama gibi öğrenme amaçları olabilir.

Takviyeli öğrenmede etmenin optimal politikayı doğrudan öğrenmesi zordur. Çünkü etmen için kullanabileceği eğitim verileri bulunmamaktadır. Etmen için kullanılabilir eğitim verisi sadece anlık ödüllerdir.

Takviyeli öğrenmede bir eğitmen bulunur fakat denetimli öğrenmedeki gibi sisteme çok detay vermez veya veremez. Bunun yerine öğrenen sistem bir karar verdiğinde bu kararın doğru olduğu durumlar için sistemi ödüllendirir ve yanlışlar için de cezalandırır. Amaç, öğrenen sistemin denediği olası durumların hedef olup olmadığının kontrolü ve denenen doğru veya yanlış tüm durumların hatırlanmasıdır.

Karar verilen durumlar artarda gelen diziler şeklinde hatırlanırsa sonunda başarıya ulaşan duruma bağlı olarak hatırlanan ardışık durumlar dizisindeki her karara büyük ödülden hisseler dağıtılır.

Ödül veya cezayı belirleyen genellikle bir değer (amaç veya hedef de denilir) fonksiyonu (V) vardır. Davranış politikası ($\prod$) ile t anındaki durumda iken (s_t) yapılabilecek hareketlerden (a_t) optimumu seçilebilir. Takviyeli öğrenme, değer fonksiyonunun ürettiği en büyük ödüle sahip davranış politikasını tercih eder. Optimum davranış politikasının tercihi şöyle ifade edilir.

$$\prod = \arg\max(V(s_t, a_t))$$

3.1. Q Öğrenme

Takviyeli öğrenme yaklaşımları içinde en çok bilinen ve kullanılan yöntem Q- Öğrenme (Q- Learning) yöntemidir. Bu algoritma labirent ve arama gibi problemlere sıklıkla uygulanmaktadır.

Q öğrenme, sonlu durumlu Markov Karar Süreci olarak modellenebilen alanlara kolaylıkla uygulanabilecek modelden bağımsız bir Takviyeli Öğrenme algoritmasıdır. Takviyeli öğrenme problemleri matematiksel olarak Markov karar süreçleri gibi modellenebilirler. Markov karar süreci aşağıda belirtilen parametrelere bağlı olarak tanımlanır (Kaya, 2003) :

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

Burada $\pi(S)$, S kümesi üzerinde bir olasılık dağılımı olarak tanımlanabilir. Durum geçiş fonksiyonu, etmenin o andaki durum ve hareketinin bir fonksiyonu olarak, bir olasılıkla ortamın sonraki durumunu belirler. Ödül fonksiyonu ise anlık ödülleri verir. Bu özellik Markov Özelliği olarak bilinir. Buna göre, geçmiş durum, hareket ve takviye arasında bir bağımlılık söz konusu değildir. Bu yüzden, Markov karar süreci ortamın sadece bir adım için dinamiğini tanımlar (Kaya, 2003).

$T: SxA \rightarrow \pi(S)$ ifadesi S kümesinde geçerli durumda, A kümesinden uygun hareket a'nın seçilmesi anlamına gelmektedir. Burada etmenin amacı toplam ödülü maksimum yapmak olduğuna göre π politikası aşağıdaki gibi ifade edilir (Mitchell, 1997):

$$\pi * (s) = \arg\max[\mathrm{r(s,a)} + \gamma V * \big(\delta(s,a)\big)]$$

Buradaki parametrelerin anlamı şu şekilde belirtilir:

- $\delta(s,a)$: s durumundaki a hareketi ile davranıldığındaki durum
- $V*$: toplam ödül
- $r(s,a)$: s durumunda yapılan a hareketi sonrasındaki ödül
- $\pi*(s)$: etmenin optimal hareketi
- γ : azaltma katsayısı ($0 \leq \gamma \leq 1$)

V* eğer $\delta(s,a)$ ve $r(s,a)$ tam olarak biliniyorsa bir anlam ifade edebilir.

Bu denklem incelenirse;

$$Q(s,a) = r(s,a) + \gamma V*\big(\delta(s,a)\big)$$

Bu denklemde V*(s) yerine;

$$V*(s) = maxQ(s,a')$$

değeri yazılırsa;

$$Q(s,a) = r(s,a) + \gamma\, maxQ(\delta(s,a),a')$$

Bu denklemde $\delta(s,a)$ ifadesi için s durumundaki a hareketi ile davranıldığındaki durum olduğu belirtilmişti. Bu durumda bir sonraki durum şu şekilde ifade edilebilir;

$$s' = \delta(s,a)$$

Ayrıca denklemdeki $r(s,a)$ ifadesi için de s durumunda a hareketini seçmesi sonrası aldığı ödül denildiği için denklem şu şekilde yazılabilir;

$$Q(s,a) = r + \gamma\, maxQ(s',a')$$

Denklemi elde edilir. Bu denklemin yinelemeli (recursive) bir yapıda olduğu görülmektedir. Q tablo girişleri bu şekilde her adımda yenilenmiş olmaktadır.

Q öğrenmede yeni gözlemlerin ve takviyelerin önemi dikkate alınacaksa öğrenme katsayısı bu formüle dâhil edilir. Bu durumda formül şu şekilde yazılabilir:

$$Q(s,a) = (1-\beta)Q(s,a) + \beta(r + \gamma\, maxQ(s',a'))$$

Bu formülde β öğrenme katsayısı olup $0 \leq \beta \leq 1$ değerlerini alır. β yeni gözlemlerin ve takviyelerin önemini belirler. Eğer β sıfır alınırsa, yeni gözlemler dikkate alınmaz ve Q değeri aynı kalır. Eğer β, 1 alınırsa eski Q değerleri ihmal edilmiş olur ve sadece en son verilere bağımlı olur. Öğrenme işleminin ilk aşamalarınsa β için yüksek değerler kullanılıp daha sonraki aşamalarda kademeli olarak azaltmak öğrenme işlemi için daha faydalı olabilir.

Q öğrenmede karşılaşılan bir diğer durum a hareketinin nasıl seçileceğidir. Bazı problemlerde hareket seçimi mevcut hareket kümesi A içinden rastgele yapılabilir. Hareket seçimi için en fazla kullanılan yöntem, etmenin Q değerleri üzerinde Boltzman dağılımı uygulamaktır.

Boltzman dağılımının denklemi şu şekildedir:

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

Bu denklemde, $P(a_i \mid s)$ etmen s durumunda iken a_i hareketini seçme olasılığıdır. k değeri daha önce gidilen ve gidilecek yerleri kullanma arasındaki planı belirler. Küçük k değerleri için etmen daha küçük Q değerli hareketleri seçerek yeni durumları bulmaya çalışır. Öğrenme ilerlerken k değeri arttırılır ve etmen daha büyük Q değerlerini seçer. Böylece etmen öğrendiği bilgiyi kullanmaya başlar (Kaya, 2003).

Q öğrenme işleminde başlangıç değerleri için Q tablosu rastgele değerler ile ya da sıfır değerleri ile doldurulur. Algoritmanın daha iyi anlaşılabilmesi için sıfır değerleri ile başlatılmalıdır. Q tablosunda her bir s durumu ve bu durumda yapılabilecek her bir a hareketi için alınan değerler tutulur.

Q değerlerinin tutulduğu Q tablosunun yapısı Tablo 3.1'de örnek olarak gösterilmiştir.

Tablo 3.1. Q değerlerinin tutulduğu örnek bir tablo

	s_1	s_2	$\cdots$	S_m
a_1	$Q(s_1, a_1)$	$\cdots$		
a_2	$\vdots$			
$\vdots$				
A_n				$Q(s_m, a_n)$

Q öğrenme algoritmasının adımları şu şekilde belirtilebilir:

- Her bir durum ve hareket ikilisi için $Q'(s, a)$ değerlerini sıfır değerleri ile başlat
- Etmenin bulunduğu mevcut s durumunu gözle
- Öğrenme işlemi boyunca tekrarla:
  - Bir a hareketini seç ve çalıştır
  - Anlık r ödülünü al
  - Yeni s' durumunu gözle
  - s durumu ve a hareketi için uygun Q değerini aşağıdaki formüle göre güncelle:

$$Q(s, a) = r + \gamma \, max Q(s', a')$$

  - Yeni durumu güncelle:

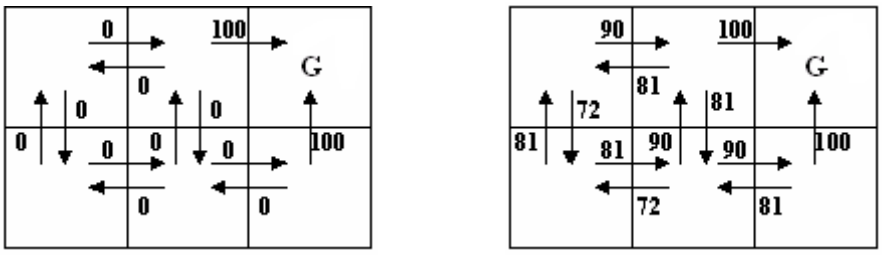
$$s \leftarrow s'$$

3.1.1. Q Öğrenme İçin Bir Örnek

Q öğrenmeye bir grid(ızgara) üzerinden örnek verilerek daha iyi anlaşılması sağlanabilir. Şekil 3.2'de görüldüğü gibi 6 bölmeden oluşan bir grid üzerinde etmenin amacı G (Goal state) ile belirtilen hedef hücreye ulaşmaktır. Şekil üzerinde belirtilen oklar, etmenin gidebileceği yönleri göstermektedir (Hacıbeyoğlu, 2006).

Ödüllerin gösterildiği soldaki gridde etmenin her bir hareketine karşılık alacağı anlık ödüller gösterilmiştir. Bu ortamda G durumuna geçiş dışındaki bütün durum ve hareket geçişleri için anlık ödül sıfırdır. G hedef durumu için ise anlık ödül 100 olarak belirlenmiştir. Etmen G durumuna geçtikten sonra, artık hedefe ulaştığı için bu durumda kalacaktır.

Sağdaki grid Q tablosunun değerlerini göstermektedir. Şekilde de görüldüğü gibi G hedef durumuna ulaşıldığında alınan ödül değeri 100 olduğundan, etmeni bu duruma götüren hücrelerin de değeri formülün uygulanması sonucu 100 olmaktadır. Aynı şekilde bundan önceki hücrelerin değeri de buna bağlı olarak değişim göstermektedir. Etmeni hedefe yakınlaştıran hücrelerin değerinin, hedefe uzaklaştıran hücrelerin değerinden daha yüksek çıktığı görülmektedir.



r(s,a) anlık ödül değerleri Q(s,a) değerleri

Şekil 3.2 Q öğrenme için bir grid ortamı örneği

Q öğrenme işleminde etmenin amacının alınan toplam ödülü en üst düzeye çıkarmak olduğu belirtilmişti. Buna göre Q tablosundaki her bir durum-hareket için alınan değerler bu şekilde belirlendikten sonra, etmenin izleyeceği optimal politika da ortaya çıkmış olmaktadır.

4. YAPAY ATOM ALGORİTMASI (A^3)

Metasezgisel, birçok farklı problemlere uygulanabilen sezgisel yöntemlerden oluşan bir algoritma grubu olarak tanımlanabilir. Metasezgisel, arama uzayının yüksek kaliteli çözümlerini kapsayan bölgelerinde aramayı gerçekleştirmek için probleme özgü sezgisellere rehberlik etmek amacıyla tasarlanan genel amaçlı sezgisel yöntemdir (Dorigo ve Stützle, 2004).

Metasezgisel algoritmaların birçoğu doğadan esinlenerek ortaya çıkarılmıştır. Bunlar genelde fiziksel işlemler, kimyasal işlemler, biyolojik işlemler, sosyal işlemler, coğrafik işlemler, müzik ve bunların karışımından oluşan işlemlerdir. Örneğin, Genetik algoritmalar (Goldberg, 1989), Arı kolonisi algoritması (Karaboğa ve Baştürk, 2007), Karınca kolonisi algoritması (Dorigo ve Stützle, 2004), Parçacık sürü optimizasyonu (Kennedy ve Eberhart, 1995), Harmoni arama algoritması (Lee ve Geem, 2005), Elektromanyetizma benzeri algoritmalar (Birbil ve Fang, 2003), Yapay bağışıklık sistemi (De Castro ve Von Zuben, 2002), Ateşböceği algoritması (Yang, 2009) gibi birçok algoritma doğadan yararlanılarak ortaya çıkarılan metasezgisel algoritmalardır.

Metasezgisel algoritmalarda amaç, en iyi çözümü ya da en iyi çözüme yakın çözümleri bulmak için arama uzayını hızlı bir şekilde araştırmaktır. Arama uzayındaki yerel en iyi noktalardan kurtulup genel çözüme ulaşmak için çeşitli mekanizmalar kullanırlar.

Yapay atom algoritması (Karcı, 2012; Karadoğan ve Karcı, 2013; Yıldırım ve Karcı, 2013), kimyasal bileşiklere dayanan bir metasezgisel algoritmadır. Kimyasal bileşiklerin oluşum işlemi, bilimsel ve mühendislik problemlerinin optimizasyonu için taklit edilebilir. İyonik Bağ ve Kovalent Bağ kimyasal işlemleri birbirinden ayrı olarak taklit edilmektedir. Bu iki işlem algoritmanın temel iki işlemini teşkil eder.

Problem bir atom olarak düşünülebilir ve problemin özelliklerini barındıran her parametre elektron olarak düşünülecektir. Her atom başlangıçta problemin rassal bir çözümüdür. Başlangıçta birden fazla atom ile işleme başlanır ve bu çözümler kümesine Atom Kümesi denilmektir. Atom Kümesi, genetik algoritmalarda Popülâsyona, Atom, kromozoma ve elektron ise gene benzetilebilir.

Atom Kümesi A={A_1, A_2, …., A_p} şeklinde olsun. n tane elektronu olan bir atom Şekil-4.1' de görüldüğü gibi olmaktadır.

E_1	E_2	.....	E_{n-1}	E_n

Şekil 4.1 Yapay atom algoritmasında bir atomun temsili

Atomlar elektron alıp-verirken genel olarak iki prensipten söz edilebilir.

Kovalent Bağ: Kimyasal olaylardan biri olan kovalent bağ, iki atomun en az bir elektronu ortaklaşa paylaşması sonucu oluşur. İki atomun sonuç üzerindeki etkisi çok olan elektronlar arasında en az birer tanesinin etkisi karşılaştırılır. Hangisi daha etkili ise, onun değeri diğerin üzerine kopyalanır. Bu şekilde iki atom da aynı parametre için aynı değere sahip olur. Bu şekilde oluşturulan işleme Kovalent Bağ işlemi denilir.

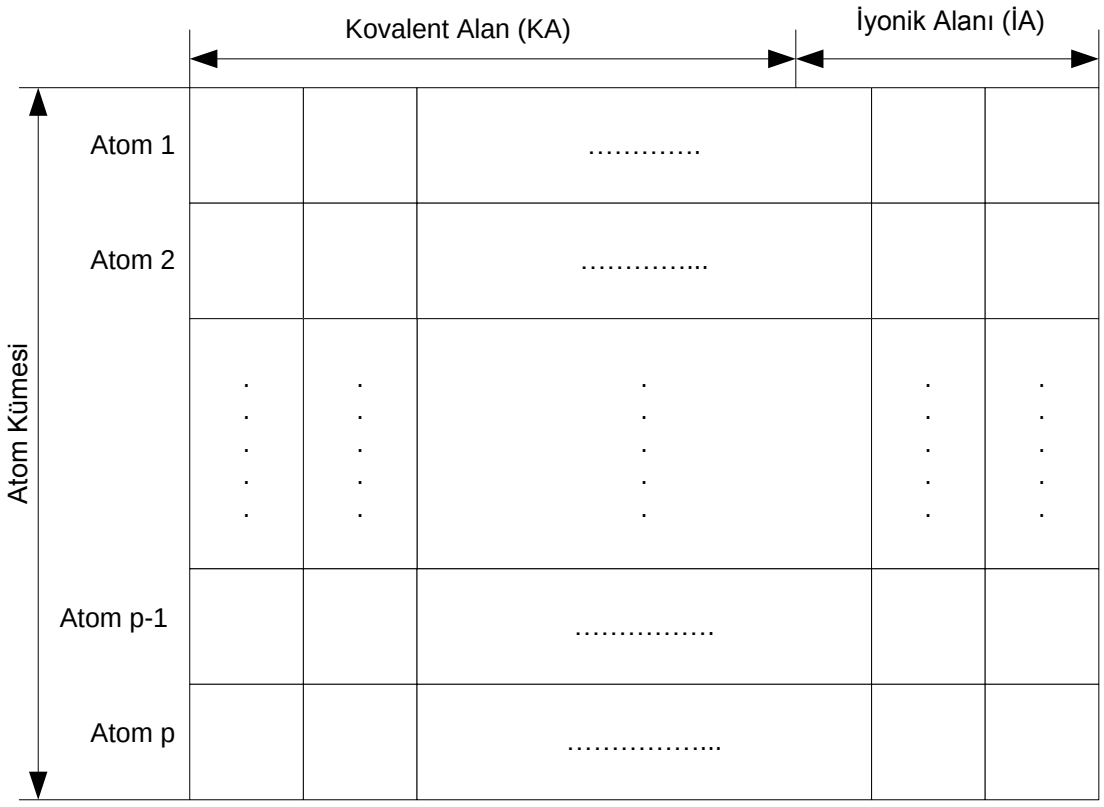
İyonik Bağ: Diğer bir kimyasal olay ise atomların elektron vererek veya elektron alarak oluşturdukları iyonik bağ işlemidir. Bu yöntemde sonuç üzerinde etkisi az olan elektronların değerleri atılacaktır ve yerine rassal (random) olarak yeni değer oluşturulur. Bu işleme İyonik Bağ işlemi denilir.

Bu tez kapsamında uygulanmakta olan sezgisel hesapsal yöntemde ilk olarak bir Atom Kümesi oluşturulur ve her atomun amaç fonksiyonu değeri hesaplanır ve her elektronun amaç fonksiyonu üzerindeki etkisi hesaplanır. Her atomun elektron etkilerine göre sıralama işlemi yapılır ve ondan sonra Kovalent Bağ ve İyonik Bağ işlemi uygulanır. Atomların yeni halinin amaç fonksiyon değeri ve elektron etkileri hesaplanır. Bu işleme bitim şartı sağlanıncaya kadar devam edilir.

Her elektronun amaç fonksiyonu üzerindeki etkisi hesaplanır. Daha sonra her atomun elektronları amaç fonksiyonu üzerindeki etkilerine göre büyükten küçüğe doğru sıralanır. Bu işlemden sonra bir atomdaki elektron sayısı kovalent oranı ile çarpılır. Bu şekilde en iyi kaç tane elektronun kovalent bağ için kullanılacağı ortaya çıkmış olur. Bu elektronların oluşturduğu alana ***Kovalent Alanı*** denir. Geriye kalanlar ise iyonik bağ için kullanılır. Bu alana ***İyonik Alanı*** denir. İyonik alanında

yer alan elektronların eski değerleri iptal edilir ve bunların yerine yeni rassal değerler üretilir. Kovalent alanındaki elektronlar çözümün birer parçasıdırlar.

Yapay atom algoritması iki işlem içermektedir. İyonik ve Kovalent Bağlar bu iki işlemi temsil etmektedirler. Kovalent bağ hangi elektronların en az iki atom arasında ortaklaşa paylaşılacağını belirler ve iyonik bağ ise, hangi elektronların değerinin değiştirilmesi gerektiğini belirler. İyonik bağ, etkisi dikkate alınmayacak kadar kötü olan elektronlar için yeni değer üretme işlemidir. Eğer bu elektronlardan Kovalent Alanındaki elektronlardan daha iyi bir değere sahip yeni elektron üretilirse, bu elektron kendisini kovalent alanında bulacaktır. Şekil-4.2' de atom kümesi üzerinde kovalent alanı ile iyonik alanı görülmektedir.



Şekil 4.2 Bir Atom Kümesinin Kovalent Alanı ve İyonik Alanı.

4.1. Kovalent Bağ

Kovalent Bağ işlemi iyi elektronların değerlerinin en az iki atom arasından paylaştırılmasıdır. Her elektronun amaç fonksiyonu üzerindeki etkileri hesaplanır ve her atomun elektronları kendi içinde iyiden kötüye doğru sıralanır. Aynı atom içinde, elektron etkileri diğer elektron etkilerinden daha iyi olan elektronlar ilk konumda yer alırken, elektron etkileri diğer elektron etkilerinden daha kötü olan elektronlar son konumda yer alır. A_1 ve A_2 birer atom olsun ve her atom n tane elektrona sahip olsun. β kovalent oranı ve α iyonik oranı olmak üzere βn elektron KA alanında yer alır ve $(1-\beta)n = \alpha n$ elektron ise İA yer alır.

$A1(k)$ elektronu A_1 atomunun k. elektronunu temsil eder ve $E[A1(k)]$ ise A_1 atomunun k. elektronunun amaç fonksiyonu üzerindeki etkisini temsil etsin. Algoritma 1, kovalent bağ algoritmasıdır.

$j \leftarrow 1,2,\dots,\left\lfloor \frac{m}{2} \right\rfloor$ ve $r \leftarrow \left\lfloor \frac{m}{2} \right\rfloor + 1, \left\lfloor \frac{m}{2} \right\rfloor + 2, \dots, m$ (m tane atom vardır.)

$k \leftarrow 1,2,\dots,\beta n$

Eğer $E[A_j(k)]$ değeri $E[A_r(k)]$ değerinde daha iyi ise // $k \leq \beta n$

$A_r(k) \leftarrow A_j(k)$

Değilse

$A_j(k) \leftarrow A_r(k)$

Algoritma 1: Kovalent Bağ

KovalentBağ(AtomKümesi,ElektronEtkisi,|AtomKümesi|,βn)

j←1…|AtomKümesi|/2

k←1… βn

eğer ($E[A_j(k)] < E[A_{|AtomKümesi|/2+j}[k]$)

$E[A_{|AtomKümesi|/2+j}[k] \leftarrow E[A_j(k)]$

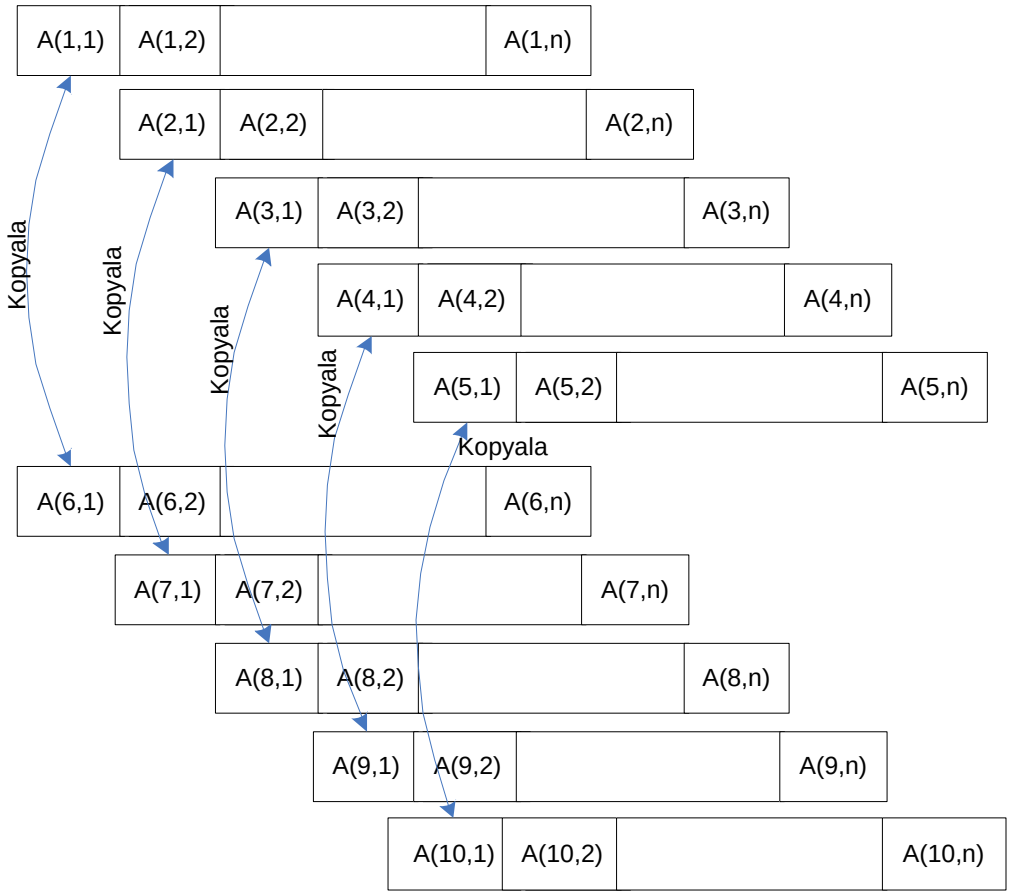
$A_{|AtomKümesi|/2+j}[k] \leftarrow A_j[k]$

değilse

$E[A_j(k)] \leftarrow E[A_{|AtomKümesi|/2+j}[k]$

$A_j[k] \leftarrow A_{|AtomKümesi|/2+j}[k]$

Şekil-4.3’ te kovalent bağ işlemi gösterilmektedir. Bu işlem olasılıksal bir işlem değildir. Bundan dolayı uygulaması çok kolaydır.



Şekil 4.3 Kovalent Bağ işleminin temsili.

4.2. İyonik Bağ

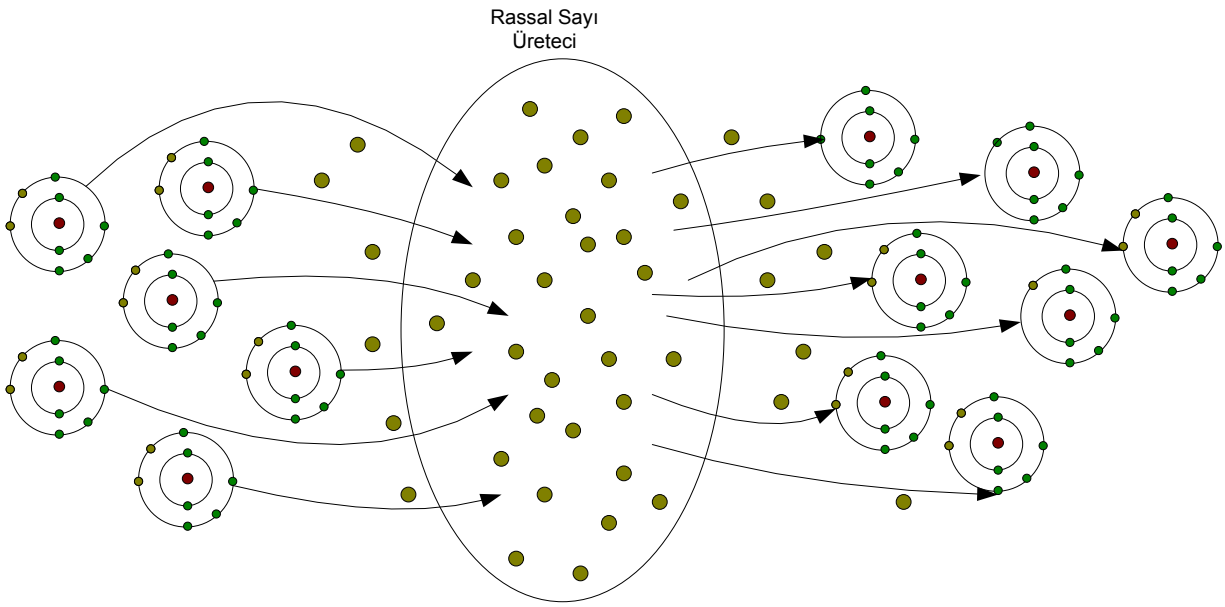
İyonik Bağ işlemi ile elektron etkisi iyi olmayan bir elektronun değeri iptal edilir ve yerine yeni değer rassal olarak üretilir. İyonik bağ işleminden sonra her atomdaki elektronların amaç fonksiyonu üzerindeki etkileri tekrar hesaplanır ve ondan sonra her atomun elektronları elektron etkilerine göre iyiden kötüye tekrar sıralanır. İyi etkiye sahip olan elektronlar Kovalent Alanında yer alırlar ve değişime uğramazlar ve İyonik Alanında yer alan elektronlar tekrar yeni değer alırlar. A_1, Atom Setindeki bir atom olsun ve $E[A1(j)]$, $\beta n + 1 \leq j \leq n$ için İyonik Alanında yer alsın. Bu durumda $A1(j)$ elektronu amaç fonksiyonun parametre tanım kümesinden rassal olarak değer alır. $A1(j)$ ifadesi A_1 atomunun j. elektronunu temsil eder ve $E[A1(j)]$ ise A_1 atomunun j. elektronunun bireysel elektron etkisini temsil etmektedir. Şekil 4.4' te atomlardan elektron koparma ve elektron bağlama işleminin

temsili görülmektedir. Her atomun n tane elektronu varsa, bunun anlamı temsil ettikleri problem n tane parametreye sahiptir. Her elektronun tanım kümesi T_i şeklinde olsun ($1 \leq i \leq n$).

Eğer $E[A_1(j)]$ İA içinde ise // $\beta n+1 \leq j \leq n$

$A_1(j) \leftarrow RANDOM(T_j)$

Şekil 4.5' te iyonik bağ işleminin akış diyagramı görülmektedir ve bu işlem her iterasyonda her atomun İA kısmındaki her elektrona uygulanır. Bu işlem rassal bir işlem olup uygulaması oldukça kolaydır. Algoritma 2 ise iyonik bağ işleminin algoritmasını göstermektedir.



Şekil 4.4 Atomların elektron alış-veriş işleminin temsili.

Algoritma 2: İyonik Bağ

İyonikBağ(AtomKümesi, n, βn, T)

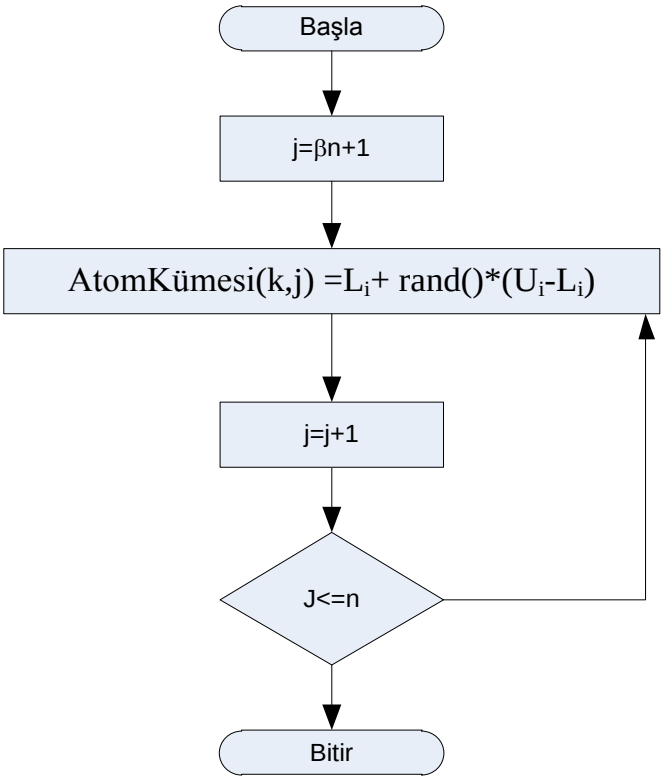
i←1,…,r //r=|AtomKümesi|

j←βn+1,…,n

AtomKümesi(i,j) =L_j+ random()*(U_j-L_j)

//L_i : T_i kümesinin en küçük elemanı

//U_j : T_i kümesinin en büyük elemanı



Şekil 4.5 İyonik bağ işleminin akış diyagramı.

İyonik bağ operatörü, her atoma yeni değerler eklenmesi ve eskilerinin giderilmesi işlemidir. Bundan dolayı iyonik bağa benzetilmiştir, çünkü iyonik bağ olayı kabaca bir atomun en az bir elektronu vermesi (bırakması) başka bir atomunda elektron alması işlemidir. Bu algoritmada ise, oluşturulan aday çözümler kümesinde bir aday çözümün bir iyonik bağ bölgesi vardır. Bu bölgedeki her değişkenin var olan değeri atılır ve yerine yeni değer alınır. Bu yeni değerin sonuç üzerinde etkisi kovalent bağ bölgesindeki elektronlardan en az birinden daha iyiyse, bu elektronun değeri kovalent bağ bölgesindeki elektron ile değiştirilir. Bu şekilde iyonik bağ populasyona çeşitliliği üzerinde de etkili bir operatördür.

4.3. Elektron Etkileri

KA ve İA bölgelerini belirlemek için, her bir elektronun amaç fonksiyonu üzerindeki etkisinin hesaplanması gerekir. Eğer amaç fonksiyonu bağımsız değişkenlere göre ayrılabilirse, her bir elektronun amaç fonksiyonu üzerindeki etkisi tam olarak hesaplanabilir. Aksi durumda yaklaşık olarak hesaplanabilir.

Amaç fonksiyonu $f(\vec{x})$ şeklinde olduğu kabul edilsin. Genellikle amaç fonksiyonların hesabı denklem (1) verilen şeklindedir.

$$f(\vec{x}) = \sum_{i=1}^{n} k(x_i) \tag{1}$$

Burada k bağıntısı i. elektronun etkisini hesaplayan bir bağıntıdır. Böylece iki durum söz konusudur:

1) Amaç fonksiyonu değişkenlerine ayrışabilen bir fonksiyon olsun: Her elektronun etkisi kesin olarak hesaplanabilir. Denklem (2)' de bir atom içerisindeki elektronların etkilerinin nasıl hesaplanacağı verilmiştir.

$$E_{i,j} = k(x_j), \qquad j = 1,2,...,n, \quad i = 1,2,...,r \tag{2}$$

Burada $E_{i,j}$ değeri i. atomun j. elektronunun etkisini temsil etmektedir.

2) Amaç fonksiyonu değişkenlerine ayrışamayan bir fonksiyon olsun. Her elektronun etkisi yaklaşık olarak hesaplanır. Aynı atom içerisindeki elektronların etkileri denklem (3)' te verilen yöntemle hesaplanır.

$$Amaç\quad fonksiyonu = \begin{cases} F_1(\vec{x}) = \sum_{i=2}^{n} G(x_i) \\ F_2(\vec{x}) = \sum_{i=1, i\neq 2}^{n} G(x_i) \\ F_3(\vec{x}) = \sum_{i=1, i\neq 3}^{n} G(x_i) \\ \vdots \\ F_n(\vec{x}) = \sum_{i=1}^{n-1} G(x_i) \\ f(\vec{x}) = \sum_{i=1}^{N} G(x_i) \end{cases} \tag{3}$$

Burada $F_i(\vec{x})$ aynı atom içerisinde i. elektron hariç diğer elektronların toplam etkisini temsil etmektedir. Bu durumda i. elektronun etkisi $f(\vec{x}) - F_i(\vec{x})$ şeklinde hesaplanabilir.

4.4. Yapay Atom Algoritma Adımları

Bu kimyasal işlemler bir hesapsal zekâ algoritması tarafından gösterilmiştir. Yapay Atom Algoritması, Kovalent Bağ ve İyonik Bağ olmak üzere iki operatörden oluşur. Bu algoritma iki adımlık bir algoritma olarak görülebilir:

Adım 1: Atomda bulunan tüm elektronlar için elektron değerleri oluşturulur. Bu işlem Atom Kümesindeki tüm atomlar için elde edilir. Ardından tüm atomlar için amaç fonksiyonu değerleri hesaplanır.

Adım 2: Kovalent bağ işlemi mevcut Atom Kümesine uygulanır. Kovalent bağ uygulandıktan sonra, aynı Atom Kümesine iyonik bağ işlemi uygulanır. İyonik bağ işleminden sonra İA bölgesinde yeni elektron değerleri oluşur. İA bölgesindeki tüm elektronlar için elektron değeri tekrar hesaplanmalıdır. Her atomun elektronları

elektron değerlerine bakılarak tekrar iyiden kötüye doğru sıralanır. Bu adım bitirme kriteri sağlanıncaya kadar devam eder.

Algoritma 3: Yapay Atom Algoritması – A^3

1- Atom Kümesi (A_0) rassal olarak oluştur.

2- Bütün atomlar için her elektronun amaç fonksiyonu üzerindeki etkilerini hesapla.

3- i←0

4- Aşağıdaki işlemleri durdurma kriteri sağlanıncaya kadar devam et

   a. A_i kümesine Kovalent Bağ işlemini uygula // B←KovalentBağ(A_{i-1})

   b. A_i kümesine İyonik Bağ işlemini uygula

      // A_{i+1}←İyonikBağ(B)

   c. Her atom için İA kısmındaki elektronların amaç fonksiyonu üzerindeki etkilerini hesapla.

   d. Her atomun amaç fonksiyonu değerini hesapla.

   e. i←i+1

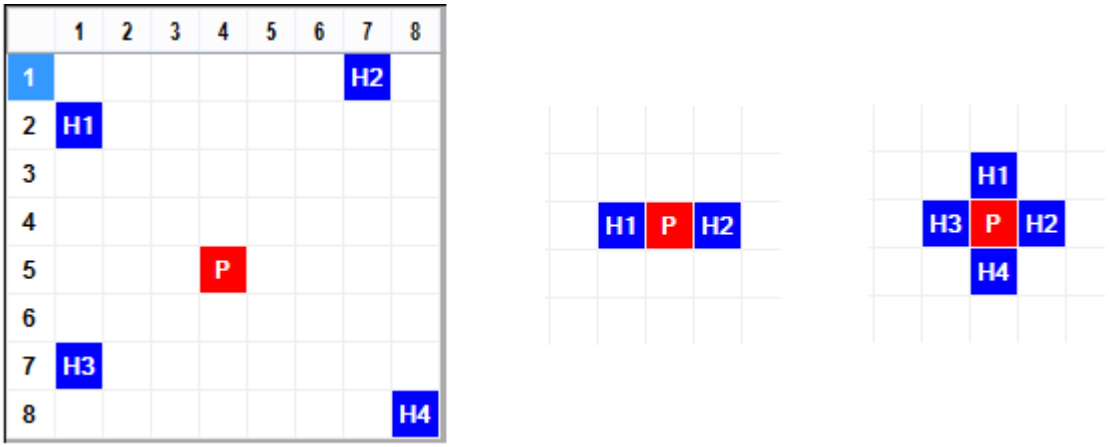
5. UYGULAMA

Bu tezde önerilen yöntemi değerlendirmek için av-avcı problemi kullanılmıştır.

5.1. Av-Avcı Problemi

Av-avcı problemi literatürde "prey-hunter" ya da "pursuit domain" olarak geçmekte olup çoklu etmen sistemleri için uygun bir öğrenme ortamı sunmaktadır.

Av-avcı probleminde amaç avcıların avı yakalaması olup probleme göre yakalama durumu değiştirilebilir. İki avcı etmeni ile yakalama ya da dört avcı etmeni ile yakalama gibi farklı versiyonlarda uygulanabilir (Şekil 5.1).

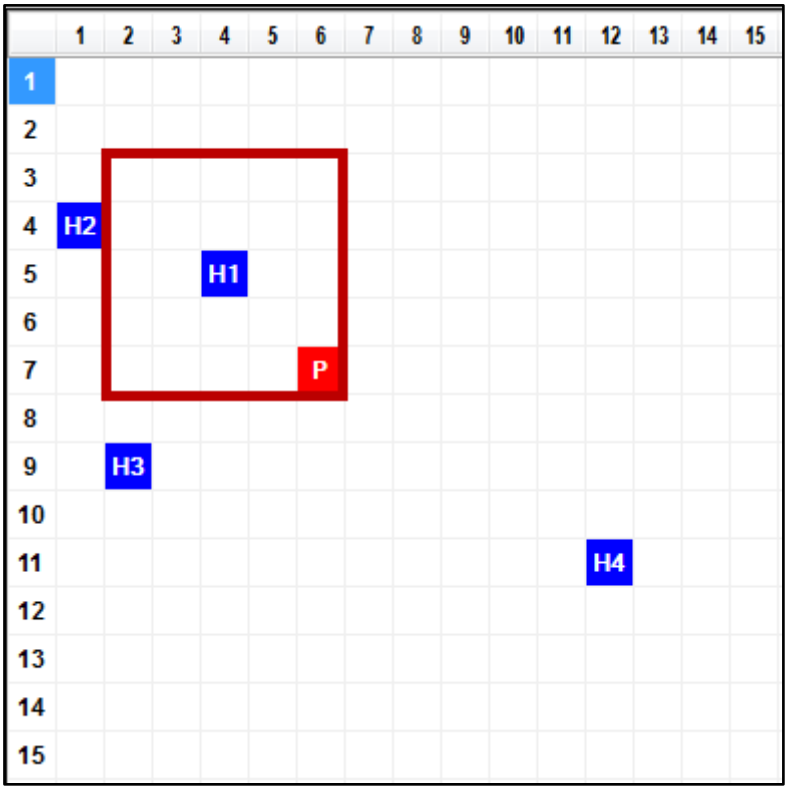


Şekil 5.1 Av-avcı probleminde iki avcı ve dört avcı ile yakalama durumları

Av-avcı problemi genel olarak aşağıdaki özelliklere sahiptir (Kaya, 2003) :

- Bulunulan ortam dinamik ve belirsiz bir ortamdır.
- Etmenlerin başlangıç durumları rastgele belirlenmektedir.
- Her bir zaman adımı için etmenler sağa, sola, yukarıya ve aşağıya olmak üzere dört farklı yönde hareket edebilir. Etmenlerin hareket sırasında konumları kontrol edilerek ortam dışına çıkmalarına izin verilmez.

- Her bir etmen belli bir mesafedeki etmenleri görebilir. Bu mesafe görme derinliği olarak ifade edilir. Görme derinliği d olan bir etmen, etrafındaki $(2d + 1)^2$ hücreyi görebilir (Şekil 5.2).



Şekil 5.2 Görme derinliği 2 olan bir etmen için görüş alanının gösterimi

- Avcılar hareketlerini öğrenerek yapabilirken, av etmeni ise rasgele veya avcı etmenleri ile arasındaki mesafeyi maksimum yapacak şekilde hareket eder. Bunun için av etmeninin bütün avcı etmenleri ile arasındaki mesafe Denklem 5.1'de verilen öklit uzaklığı veya Denklem 5.2'de verilen Manhattan uzaklığı yöntemleri ile hesaplanabilir.

$$d = \sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2} \quad (5.1)$$

$$d = |x_1 - x_2| + |y_1 - y_2| \quad (5.2)$$

Av etmeninin bir sonraki durumu tespit edilirken kullanılan formül aşağıdaki gibi ifade edilebilir:

$$D = \sum_{i=1}^{N} d(h_i)$$

Formül 5.1 Av etmeni için mesafe hesaplama formülü

Burada N değeri, görüş alanındaki avcıların sayısını ifade eder. $d(h_i)$ ise av ve avcı etmenleri arasındaki en kısa yoldur. Av etmeninin amacı D mesafesini maksimum yapmaktır.

Avcı etmenleri için bir sonraki hareketi seçerken kullanılan bir yöntem de Q değerleri üzerinde Boltzman Dağılımı uygulanmasıdır. Boltzman Dağılımı Formül 5.1' de gösterilmektedir.

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

Formül 5.2 Boltzman Dağılımı

Bu formüldeki parametreler;

A, hareket kümesi

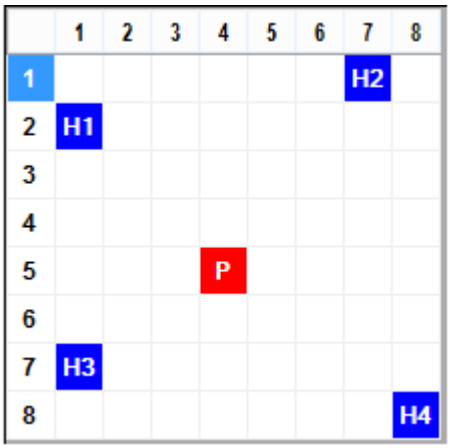
$P(a_i \mid s)$, a hareketini seçme olasılığı

k= Etmenin deney politikasının katsayısı (0<k<1). k değerleri büyük seçilirse büyük Q değerlerinin seçilme olasılığı daha yüksek olur. Bu nedenle etmenler öğrenme aşamasında iken küçük k değeri seçilip, öğrenme işlemi devam ettikçe k değeri arttırılabilir.

5.2. Av-Avcı Problemi için Q Öğrenme Algoritması

Av-avcı probleminin çözümü için öncelikle Q Öğrenme algoritması kullanılmıştır.

Q Öğrenme algoritması kullanılarak yapılan av-avcı uygulamasında bir av ve dört avcı kullanılmıştır. Uygulama ortamı için 8x8 boyutunda bir grid kullanılarak av avcılar grid üzerinde rastgele yerleştirilmektedir. Av ve avcıların konumu uygulama üzerinden elle değiştirilebilmektedir. Bu uygulama için av ve avcıların başlangıç konumları Şekil 5.3'te gösterildiği gibi seçilmiştir.



Şekil 5.3 Av ve Avcıların uygulama üzerindeki başlangıç konumları

Şekil 5.3'te avcılar H1, H2, H3, H4 olarak, av ise P olarak simgelenmiştir.

Uygulama 8x8 boyutlu bir ortam üzerinde gerçekleştirildiğinden her bir avcı için 64 durum bulunmaktadır. Avcıların gidebileceği yönler olarak da dört yön bilgisi bulunmaktadır. Tüm bu durumlar göz önüne alındığında tüm avcıların durum ve yön bilgilerinin tutulacağı Q tablosu için 4x64x4 büyüklüğünde üç boyutlu bir dizinin kullanılması gerektiği görülmüştür. Ayrıca avcıların yaptıkları her bir hareket sonrası bulundukları güncel konumlarına erişmek için de bir güncel konum dizisi kullanılmıştır.

Av avcı uygulaması için Q öğrenme algoritması adımları şu şekilde verilebilir:

1. Av ve avcıların bulundukları konumları belirle.
2. Q tablosunu başlangıç değeri olarak 0 değerleri ile doldur.
3. İterasyon sayısına ulaşıncaya kadar devam et;
   - Av yakalanmadığı sürece her bir avcı etmeni için aşağıdaki adımları tekrarla;
     - Avcının yapacağı hareketi seç
     - Seçilen a hareketine ve avcının bulunduğu s konumuna göre avcının hedefe ulaşıp ulaşmadığını kontrol et
     - Eğer avcı hedefe ulaşmışsa ödül bilgisi olarak 100 değerini ver aksi takdirde ödül 0 olacaktır.
     - $Q(s,a) = (1-\beta)Q(s,a) + \beta(r + \gamma\, maxQ(s',a'))$ formülünü kullanarak Q tablo değerlerini güncelle
     - Seçilen a hareketine göre avcının yeni konumunu güncelle
     - Eğer avcı hedefe ulaşmışsa avcıyı ilk konumuna alarak bir sonraki iterasyona geç
     - Eğer av yakalanmamışsa adım sayısını bir arttırarak bir sonraki iterasyona geç.

Burada avcının yapacağı hareket seçilirken Takviyeli Öğrenme bölümünde anlatılan iki farklı yöntem kullanılmıştır. İlk yöntem Boltzman dağılımı yöntemidir:

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

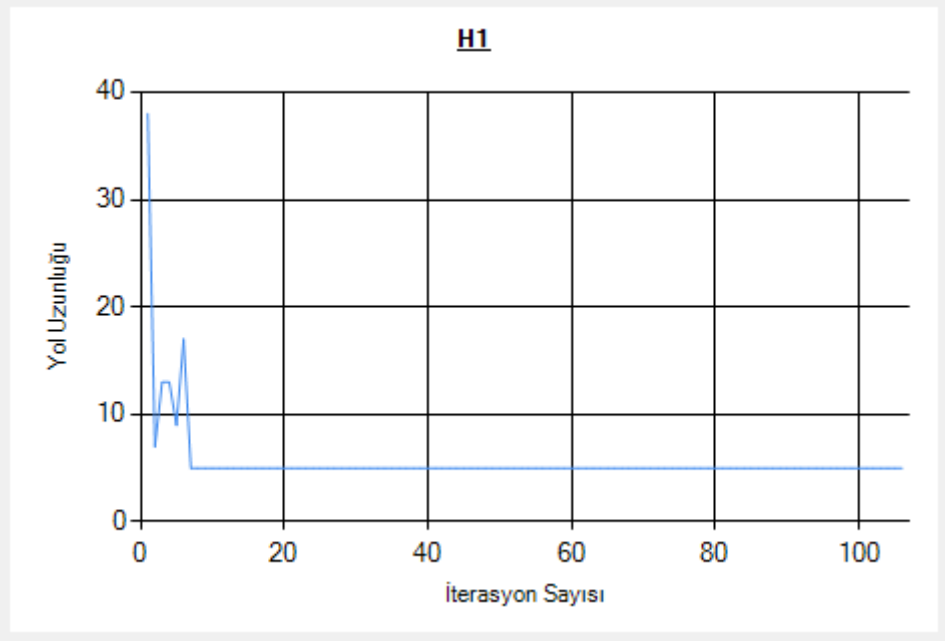
Bu denklem kullanılarak Q değeri en yüksek olan hareket seçilerek devam edilir. Hareketlerin Q değerlerinin aynı olması durumunda ise rastgele bir hareket seçilmektedir.

Boltzman dağılımı denklemi kullanılmadan yapılan hareket seçme yönteminde ise tüm hareketler için Q değerleri karşılaştırılarak Q değeri yüksek olan hareketin seçilmesi sağlanmıştır. Burada da aynı şekilde Q değerlerinin aynı olması durumunda rastgele bir hareket seçilmektedir.

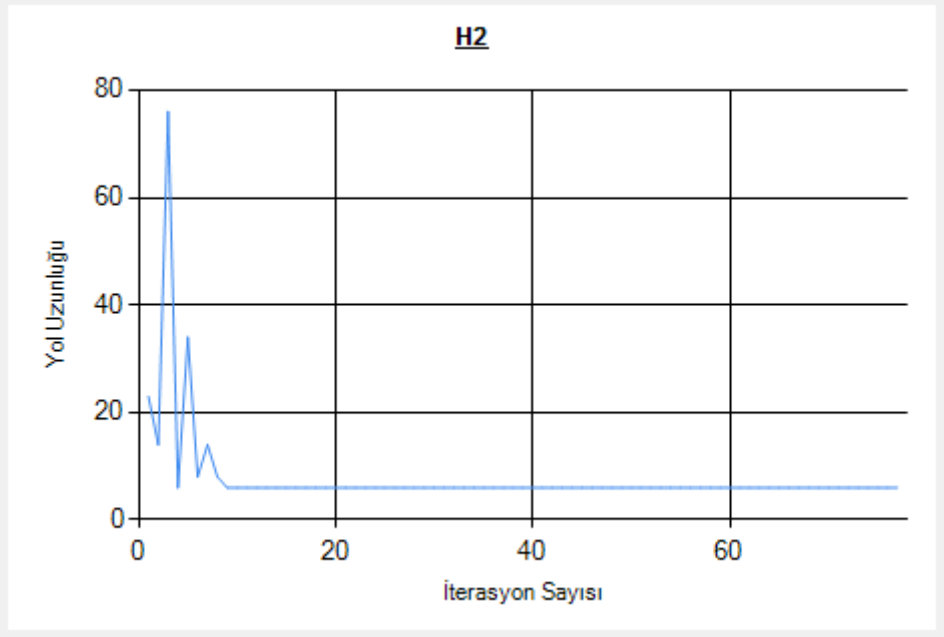
Q Öğrenme algoritması kullanılarak yapılan uygulama sonucunda ortamın büyüklüğü ve avcıların av etmenine olan uzaklığı gibi faktörlerin sonuç üzerinde etkili olduğu görülmüştür.

Farklı büyüklükteki ortamlarda yapılan uygulamadan alınan sonuçlar aşağıda gösterilmektedir.

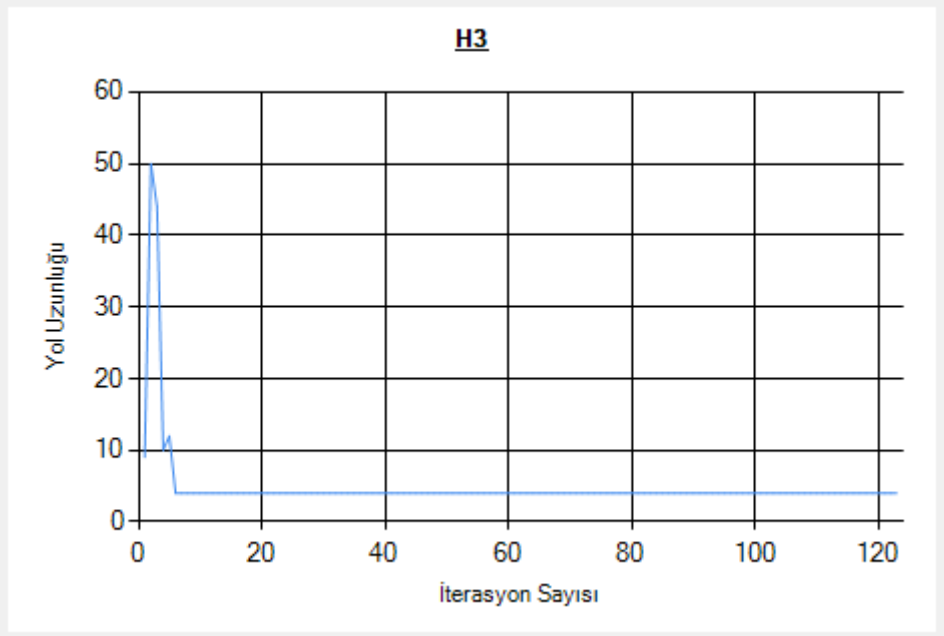
İlk olarak uygulamanın 8x8 boyutlu bir ortamda tüm avcılar için öğrenme adım sayıları gösterilmektedir.



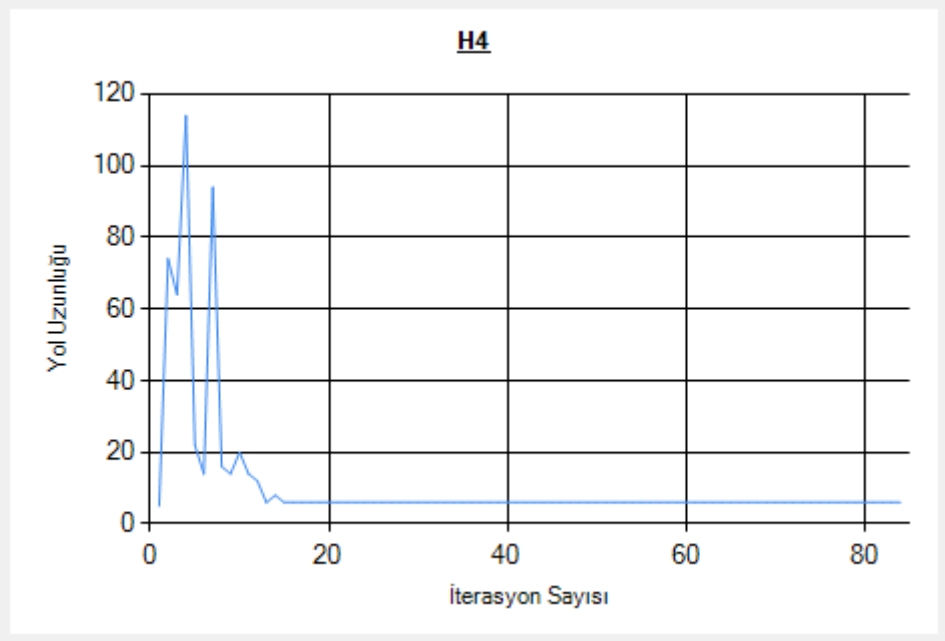
Şekil 5.4 1.Avcı için 8x8 boyutlu ortamda öğrenme adım sayısı



Şekil 5.5 2.Avcı için 8x8 boyutlu ortamda öğrenme adım sayısı

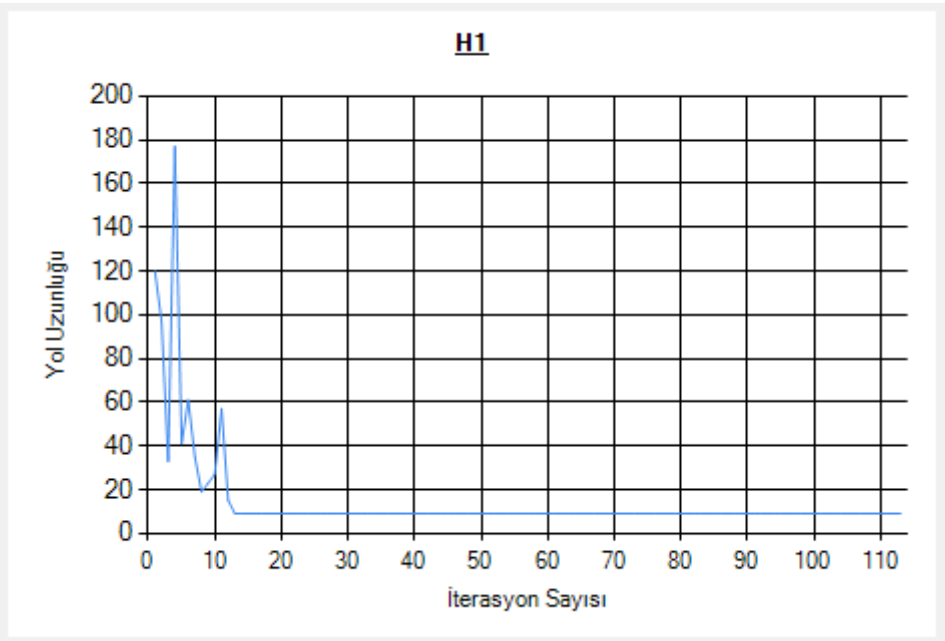


Şekil 5.6 3.Avcı için 8x8 boyutlu ortamda öğrenme adım sayısı

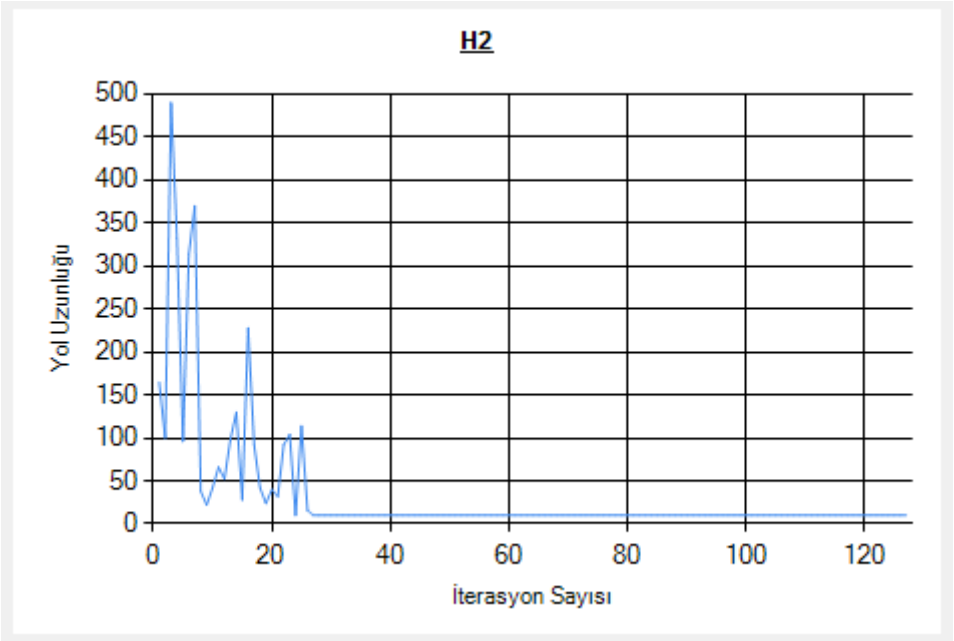


Şekil 5.7 4.Avcı için 8x8 boyutlu ortamda öğrenme adım sayısı

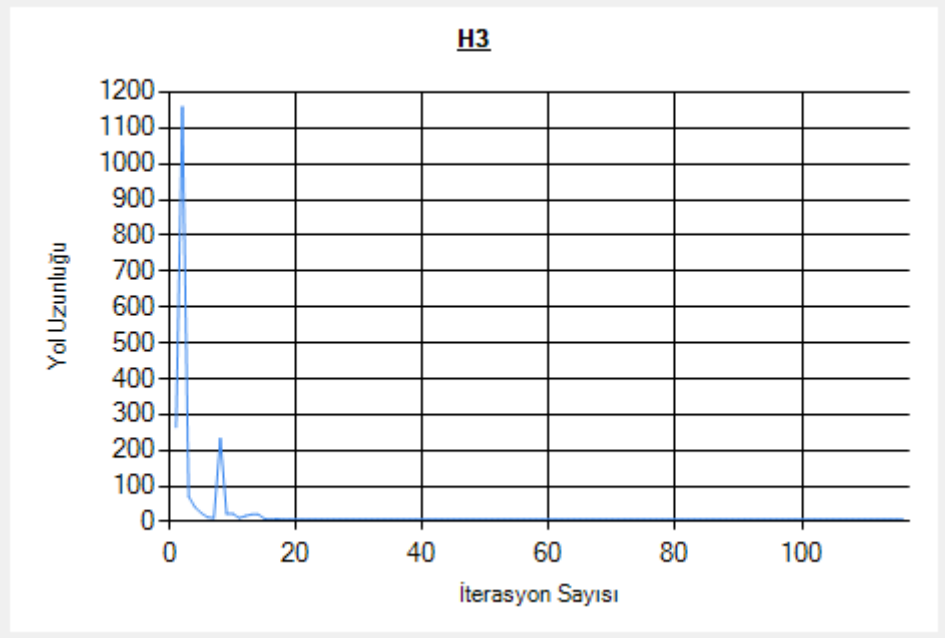
Uygulama ortamı büyütülerek 16x16 boyutuna getirilmiş ve aşağıdaki sonuçlar elde edilmiştir:



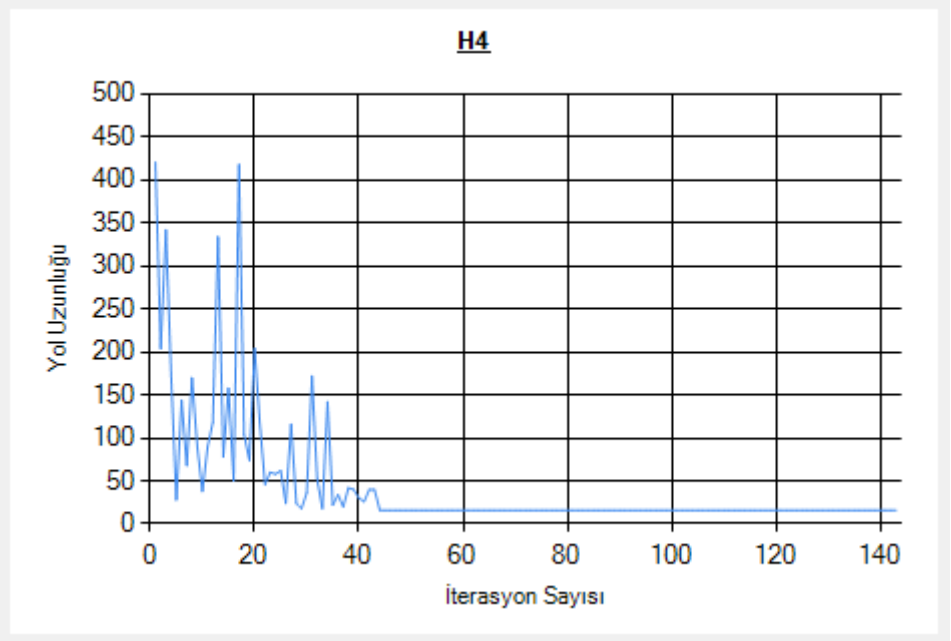
Şekil 5.8 1.Avcı için 16x16 boyutlu ortamda öğrenme adım sayısı



Şekil 5.9 2.Avcı için 16x16 boyutlu ortamda öğrenme adım sayısı



Şekil 5.10 3.Avcı için 16x16 boyutlu ortamda öğrenme adım sayısı



Şekil 5.11 4.Avcı için 16x16 boyutlu ortamda öğrenme adım sayısı

Q Öğrenme ile yapılan av avcı uygulama sonuçlarına bakıldığında ortamın büyüklüğünün ve avcıların av ile aralarındaki mesafenin öğrenme üzerinde direk etkili olduğu görülmüştür. Uygulamada ava en yakın olarak konumlandırılmış bulunan 3.avcının her seferinde avı yakalamadaki adım sayısının en az olduğu Şekil 5.6 ve Şekil 5.10'da görülmektedir. Aynı şekilde ava en uzak olarak konumlandırılmış olan 4.avcının ise her seferinde avı yakalamadaki adım sayısının en fazla olduğu Şekil 5.7 ve Şekil 5.11'de görülmektedir.

Q öğrenme ile avcı etmenlerinin belli bir iterasyon sonrasında av etmenine giden en kısa yolu buldukları ve bu yol üzerinde sabit kaldıkları görülmüştür. Uygulamanın başlangıcında farklı yollar deneyerek hedefe ulaşmaya çalışan etmenler, öğrenme işlemi ilerledikçe hedefe giden yolu da minimize etmektedirler. Ortamın büyüklüğüne bağlı olarak en kısa yolu bulma iterasyon sayısının da arttığı grafikler üzerinde de görülebilmektedir.

5.3. Yapay Atom Algoritması ile Av Avcı Uygulaması

Yapay atom algoritması kullanılarak yapılan uygulamada yine dört avcı ve bir av etmeni kullanılmıştır.

Uygulamada her bir avcı etmeni için birer Atom Seti oluşturulmaktadır. Atom Setinde bulunacak atom sayısı uygulama istenildiği gibi değiştirilebilmekte olup genel olarak 6 atom üzerinden çalıştırılmıştır. Her bir atomda bulunacak elektron sayısı olarak da üzerinde çalışılan grid ortamının boyutu kadar olmasına karar verilmiştir. Bu örnek için 8x8 boyutlu grid ortamında atoma ait elektron sayısı 16 olarak belirlenmişken, 16x16 boyutlu grid ortamında ise atoma ait elektron sayısı 32 olarak belirlenmiştir.

Her bir avcının Atom Setinde kullanılacak elektron bilgisi için avcının yön bilgisi tutulmaktadır. Etmenlerin gidebileceği dört yön olduğu göz önüne alınarak elektron değerleri olarak 0-3 arası rakamlar tutulmaktadır. Elektron değerleri ve yön bilgisi karşılıkları Tablo 5.1'de gösterilmektedir.

Yön Bilgisi	**Elektron Değeri**
Sol	0
Sağ	1
Yukarı	2
Aşağı	3

Tablo 5.1 Elektron değerleri ve yön bilgisi karşılıkları

Yapay atom algoritmasında atomdaki elektron değerlerinin amaç fonksiyonu üzerindeki etkisi için Elektron Etkisi seti oluşturulur. Elektron Etkisi setinde her bir atomdaki her bir elektronun amaç fonksiyonu üzerindeki etkisi tutulur. Burada amaç fonksiyonu olarak avcı etmenlerinin av etmenine olan uzaklığına bakılmaktadır. Uzaklık hesaplaması için Denklem 5.2'de verilen Manhattan uzaklığı yöntemi kullanılmıştır.

Gerçekleştirilen uygulama üzerinde altı atom ve on altı elektrona sahip örnek bir Atom Seti şu şekilde gösterilebilir:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A1	3	3	3	2	3	2	0	3	1	2	0	0	1	2	3	2
A2	3	2	3	0	3	2	2	2	2	3	2	2	0	1	0	1
A3	1	0	1	2	3	1	3	3	0	2	1	0	1	3	2	1
A4	3	1	3	3	2	0	0	1	1	1	3	-1	-1	-1	-1	-1
A5	0	0	1	2	3	1	2	2	1	1	1	2	2	2	3	3
A6	0	0	2	0	3	1	0	3	3	1	1	3	1	0	0	1

Şekil 5.12 Bir avcı etmeni için oluşturulan örnek Atom Seti

Bu Atom Setine karşılık gelen Elektron Etkileri Seti ise şu şekilde olmaktadır:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A1	5	4	3	4	3	4	5	4	3	4	5	6	5	6	5	6
A2	5	6	5	6	5	6	7	8	9	8	9	10	11	10	11	10
A3	5	6	5	6	5	4	3	2	3	4	3	4	3	2	3	2
A4	5	4	3	2	3	4	5	4	3	2	1	-1	-1	-1	-1	-1
A5	7	8	7	8	7	6	7	8	7	6	5	6	7	8	7	6
A6	7	8	9	10	9	8	9	8	7	6	5	4	3	4	5	4

Şekil 5.13 Bir avcı etmeni için Elektron Etkileri Seti

Şekil 5.12 ve Şekil 5.13'ten görüldüğü üzere Atom Setinde 0-3 arası olacak şekilde yön bilgisi, Elektron Etkileri Setinde ise her bir elektron için yön bilgisine bağlı olarak gideceği konumdaki ava olan uzaklık bilgisi tutulmaktadır. Bu örnekte A4 atomunun 12.elektrondan itibaren -1 değerini aldığı görülmektedir. Bu durum A4 atomunun hedefe ulaştığını, diğer atomların ise bu aşamada istenen sonucu elde edemediğini göstermektedir. Ayrıca avcının başlangıç konumundan hedefe ulaştığı adım sayısının 11 olduğu anlaşılmaktadır.

Av avcı uygulaması için Yapay Atom Algoritması genel olarak şu adımlardan oluşur:

1. Hedefe ulaşıncaya veya İterasyon sayısı kadar aşağıdaki adımları tekrarla:
    - Her bir avcı etmeni için;
        1. Atom Setini rastgele değerlerle doldur
        2. Avcı etmenine ait atomlardan hedefe ulaşan atomlar var mı kontrol et
        3. Hedefe ulaşan atom yoksa Kovalent Bağ ve İyonik Bağ işlemlerini belirtilen adım sayısı kadar veya finale ulaşan atom bulununcaya kadar sırasıyla uygula
        4. İterasyon değerini bir arttırıp bir sonraki iterasyona geç

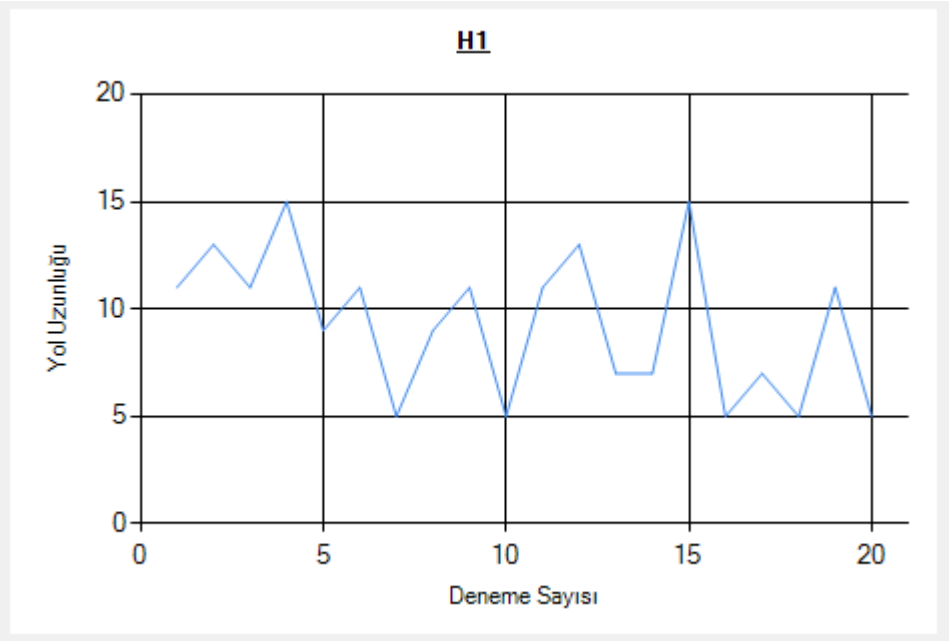
Burada avcı etmeni için hedefe ulaşan atomun olup olmadığının belirlenebilmesi için atomdaki her bir elektronun etki değerleri kontrol edilir. Etki değeri 1'den küçük veya eşit olan elektron varsa hedefe ulaştıran yol bulunmuş demektir. Bu durumda elektron dizisindeki geri kalan elektronlar için hem Atom Setindeki hem de Elektron Etkileri setindeki elektronlara -1 değeri verilerek hedefe ulaşıldığının anlaşılabilmesi sağlanmıştır.

Kovalent Bağ: Burada Kovalent Bağ işlemi olarak her bir avcı etmenine ait atomlar iki gruba bölünerek atomlar birbirleriyle elektron etkilerine göre karşılaştırılmaktadır. Elektron etki değeri küçük olan atoma ait elektronlar diğer atoma kopyalanarak kovalent bağ işlemi uygulanmaktadır. Elektron etki değeri küçük olan elektronların alınması sonucu hedefe daha yakın olan elektronlar atom içerisine yerleşmiş olmaktadır. Böylece avcı etmenlerini hedefe götüren yol mesafesi kısalmış ve av etmenine daha kısa sürede ulaşmaları sağlanmış olacaktır.

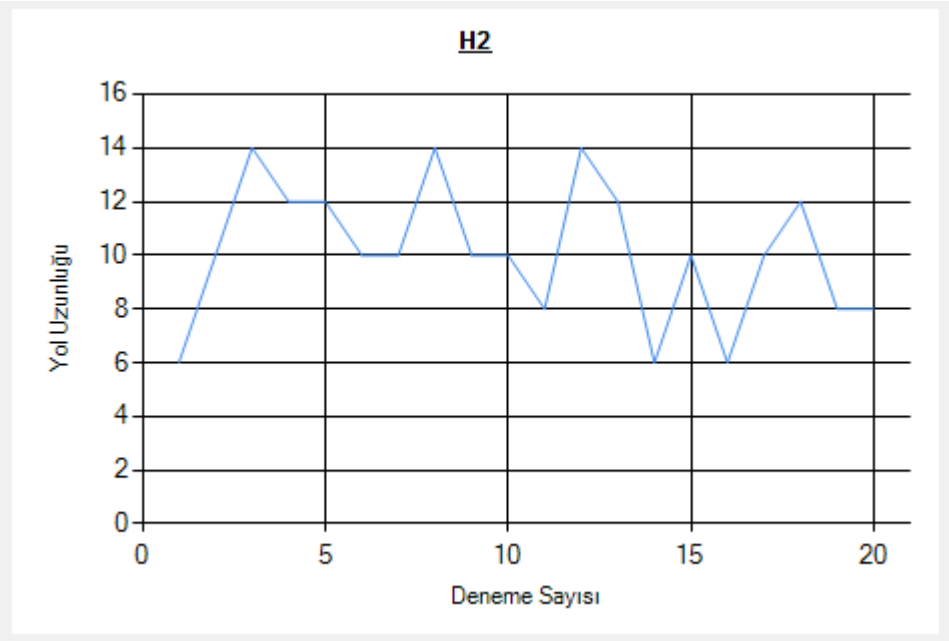
İyonik Bağ: Algoritmanın her bir adımında Her bir avcı etmeni için etmene ait tüm atomlara bakılır. Atomdaki elektronların etki değerleri toplanıp her bir

atomun toplam elektron etki değeri hesaplanır. Elektron etki değerleri toplamı en büyük olan yani hedefe olan uzaklık mesafesi en çok olan atom bulunarak, atomdaki elektronlar atılarak yerine yeni elektronlar rastgele bir şekilde oluşturularak atomun yeni şekliyle işleme devam edilir.

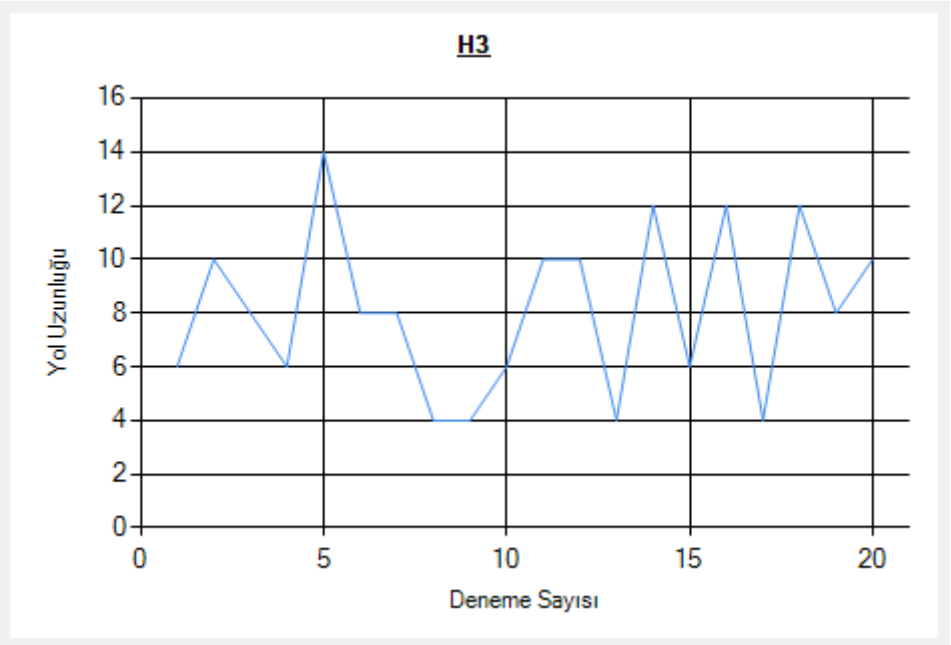
Yapay atom algoritması ile yapılan bu uygulamada her bir deneme için avcı etmenlerini hedefe ulaştıran yol uzunluğu aşağıdaki şekillerde elde edilmiştir.



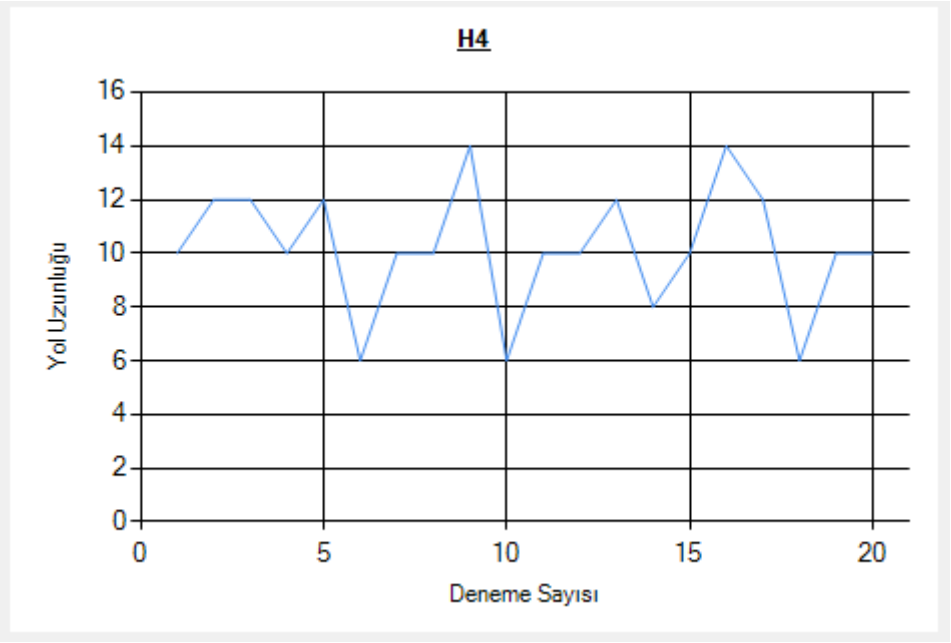
Şekil 5.14 A^3 ile 1.Avcı etmeni için 8x8 boyutlu ortamda yol uzunluğu



Şekil 5.15 A^3 ile 2.Avcı etmeni için 8x8 boyutlu ortamda yol uzunluğu

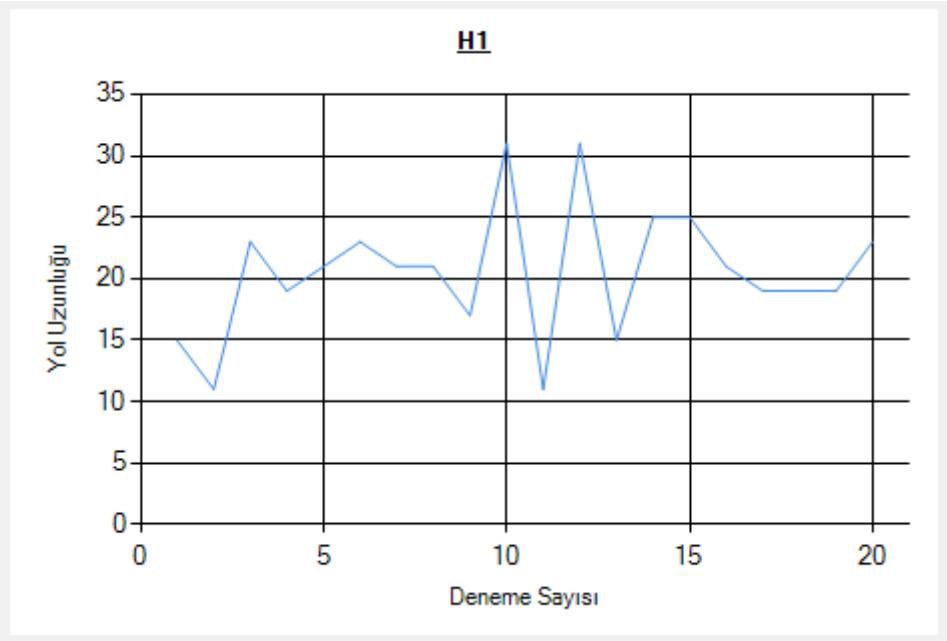


Şekil 5.16 A^3 ile 3.Avcı etmeni için 8x8 boyutlu ortamda yol uzunluğu

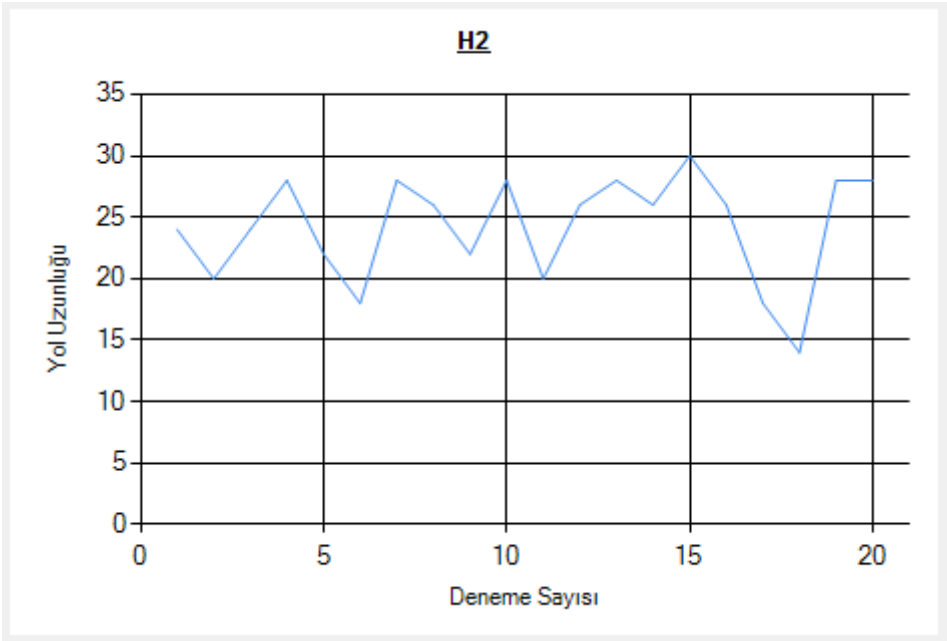


Şekil 5.17 A^3 ile 4.Avcı etmeni için 8x8 boyutlu ortamda yol uzunluğu

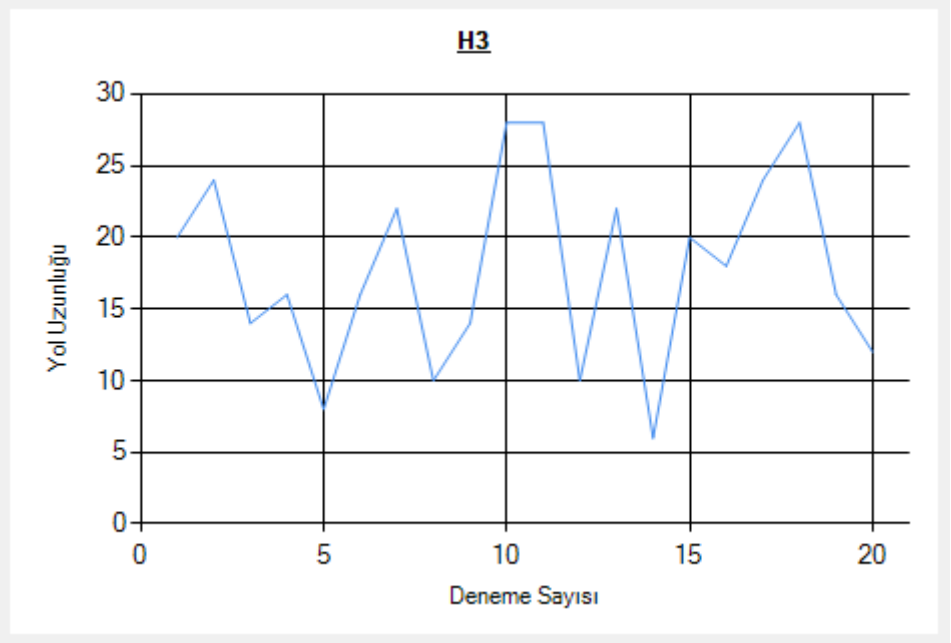
Uygulama ortamı iki kat büyütülerek yapılan çalıştırmada ise yapay atom algoritması ile aşağıdaki sonuçlar elde edilmiştir.



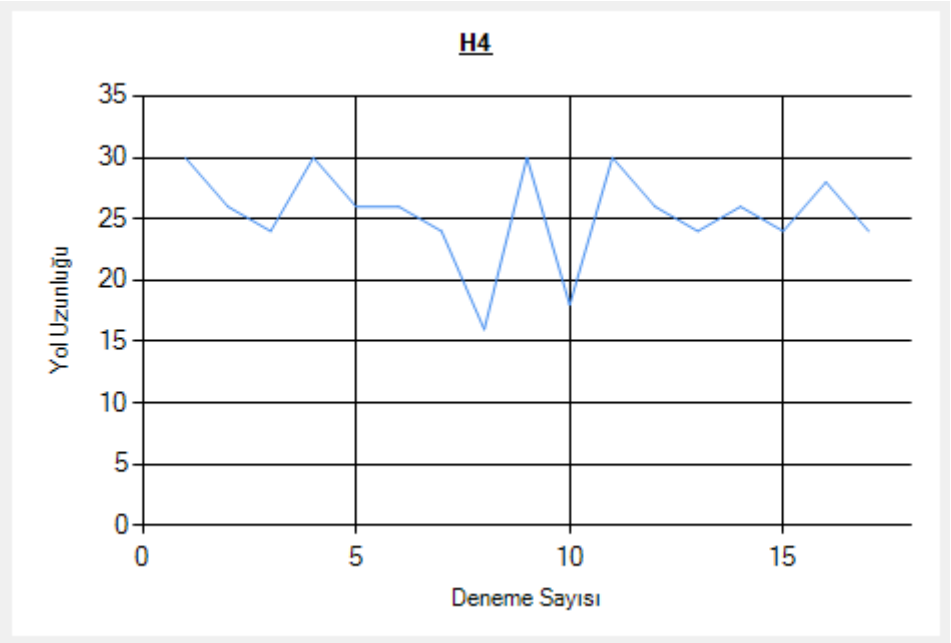
Şekil 5.18 A^3 ile 1.Avcı etmeni için 16x16 boyutlu ortamda yol uzunluğu



Şekil 5.19 A^3 ile 2.Avcı etmeni için 16x16 boyutlu ortamda yol uzunluğu

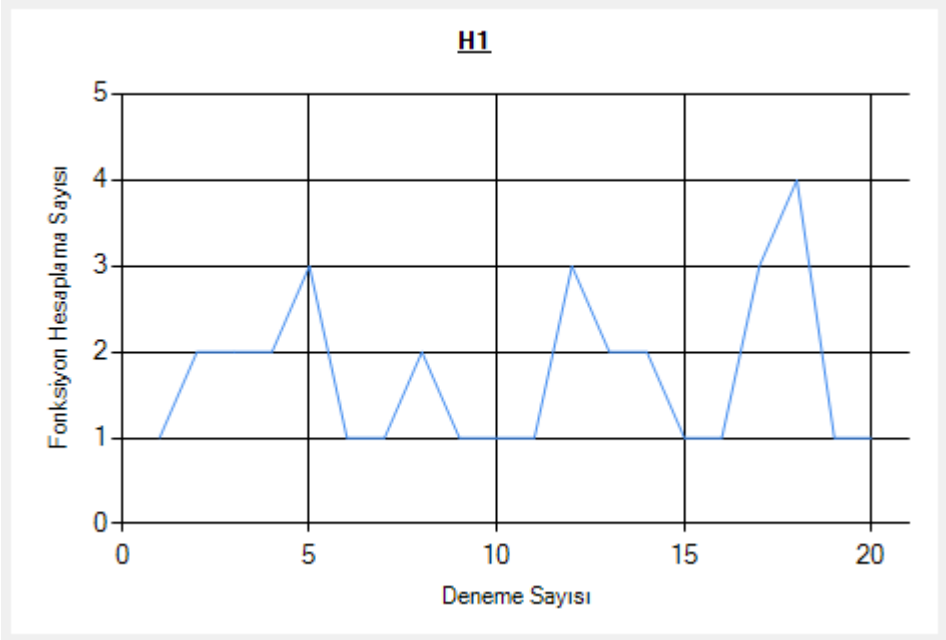


Şekil 5.20 A^3 ile 3.Avcı etmeni için 16x16 boyutlu ortamda yol uzunluğu

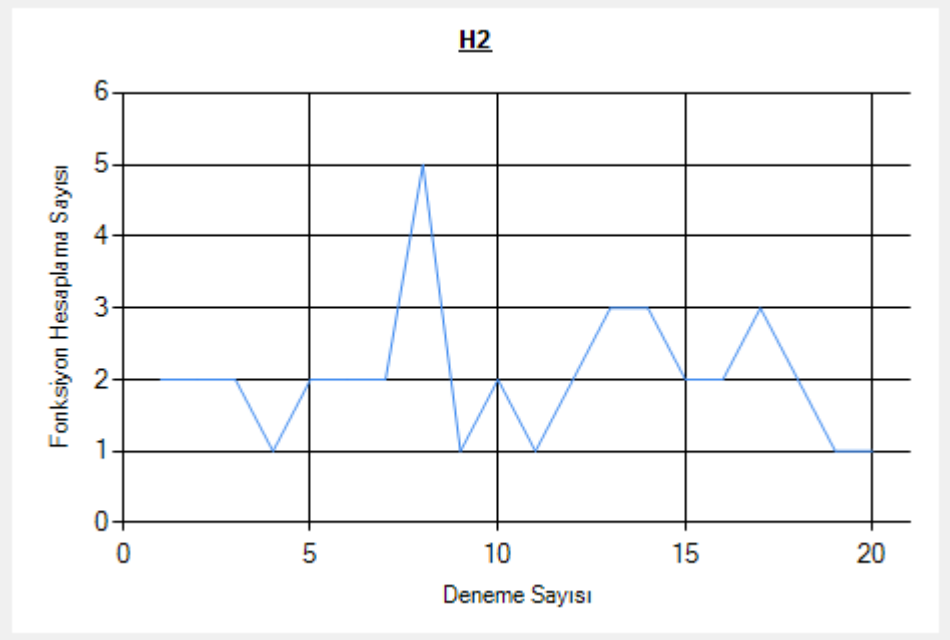


Şekil 5.21 A^3 ile 4.Avcı etmeni için 16x16 boyutlu ortamda yol uzunluğu

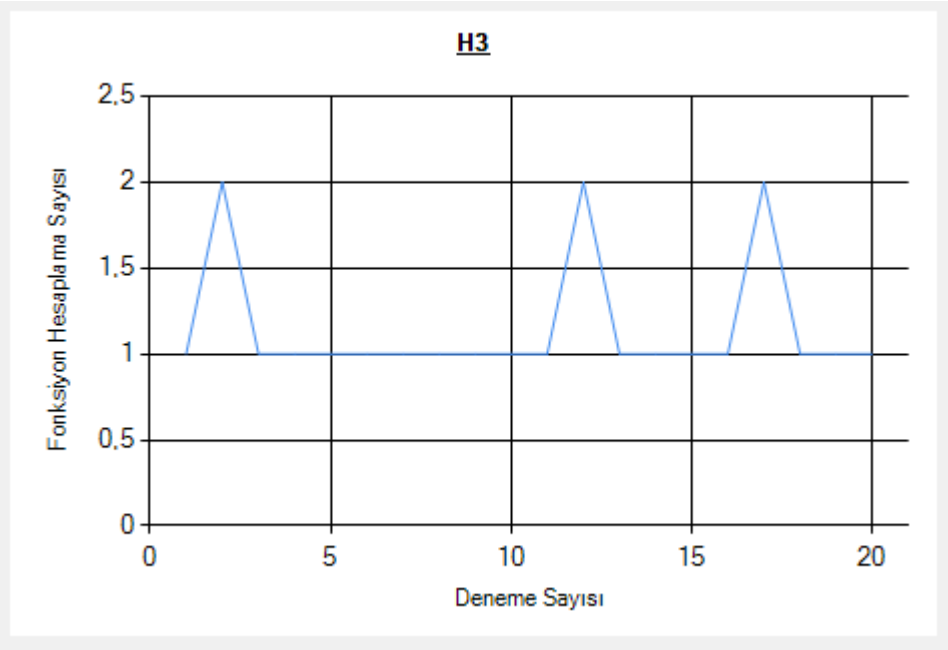
Yapay atom algoritması ile yapılan uygulamada her bir deneme için hedefe ulaşana kadarki yapılan işlem sayısı, yani her adımdaki Kovalent Bağ ve İyonik Bağ fonksiyon hesaplama sayıları grafiksel olarak her bir avcı için şu şekilde gösterilmektedir:



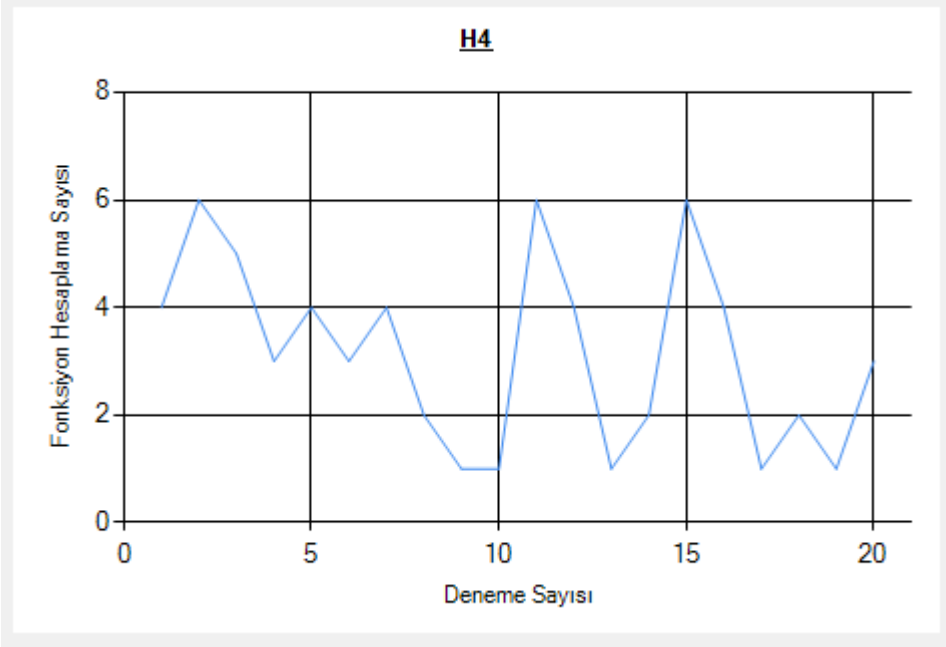
Şekil 5.22 A^3 ile 1.Avcı etmeni için 8x8 boyutlu ortamda fonksiyon hesaplama sayısı



Şekil 5.23 A^3 ile 2.Avcı etmeni için 8x8 boyutlu ortamda fonksiyon hesaplama sayısı

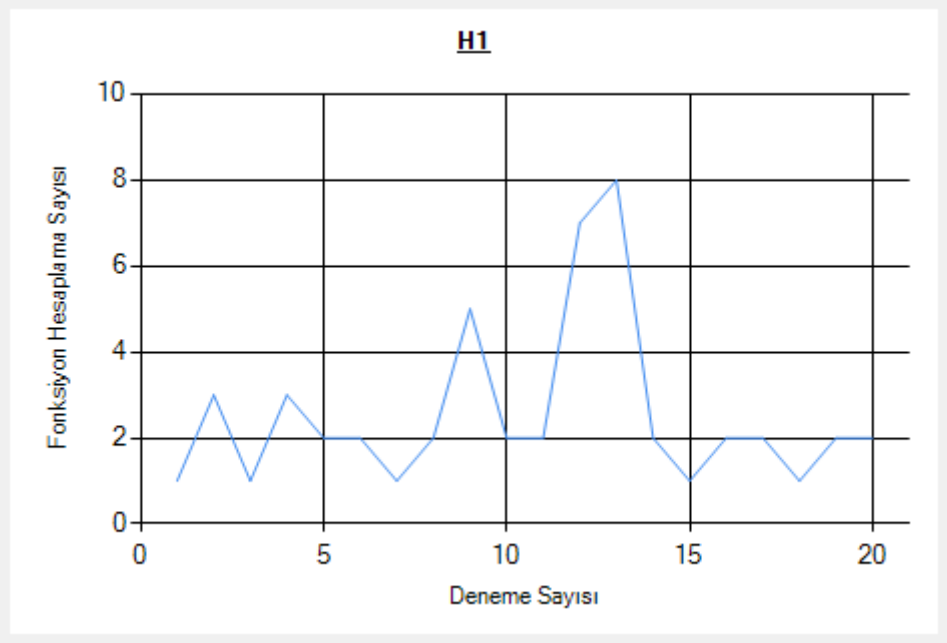


Şekil 5.24 A^3 ile 3.Avcı etmeni için 8x8 boyutlu ortamda fonksiyon hesaplama sayısı

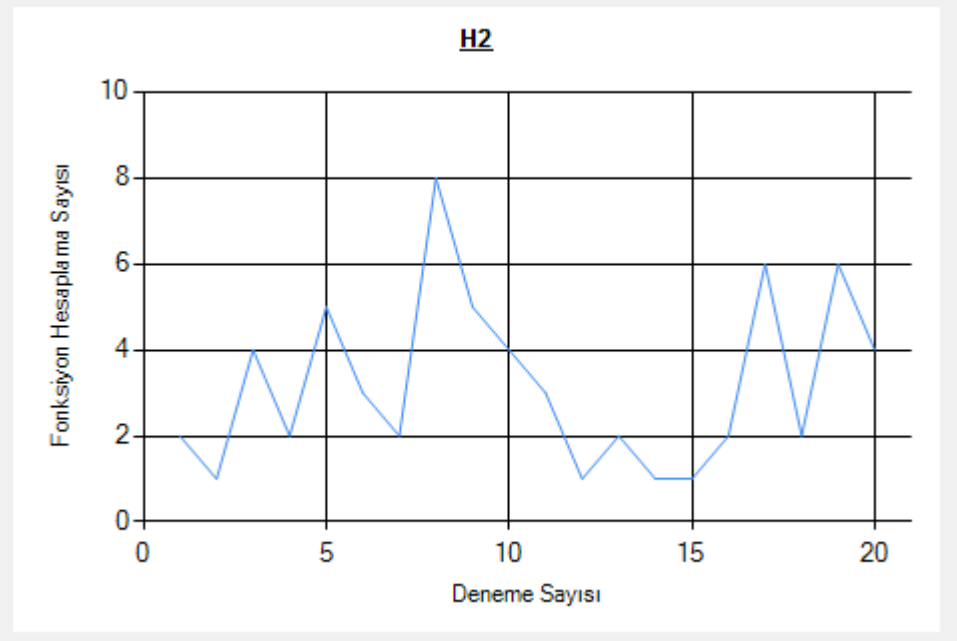


Şekil 5.25 A^3 ile 4.Avcı etmeni için 8x8 boyutlu ortamda fonksiyon hesaplama sayısı

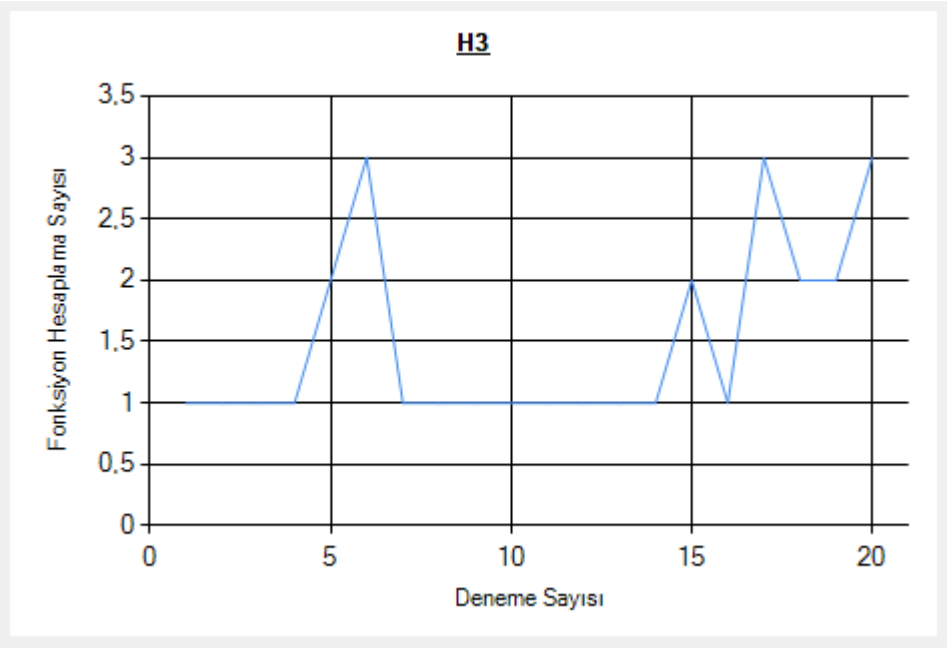
Fonksiyon hesaplama sayısının daha büyük ortamdaki (16x16) sonuçları ise şu şekildedir:



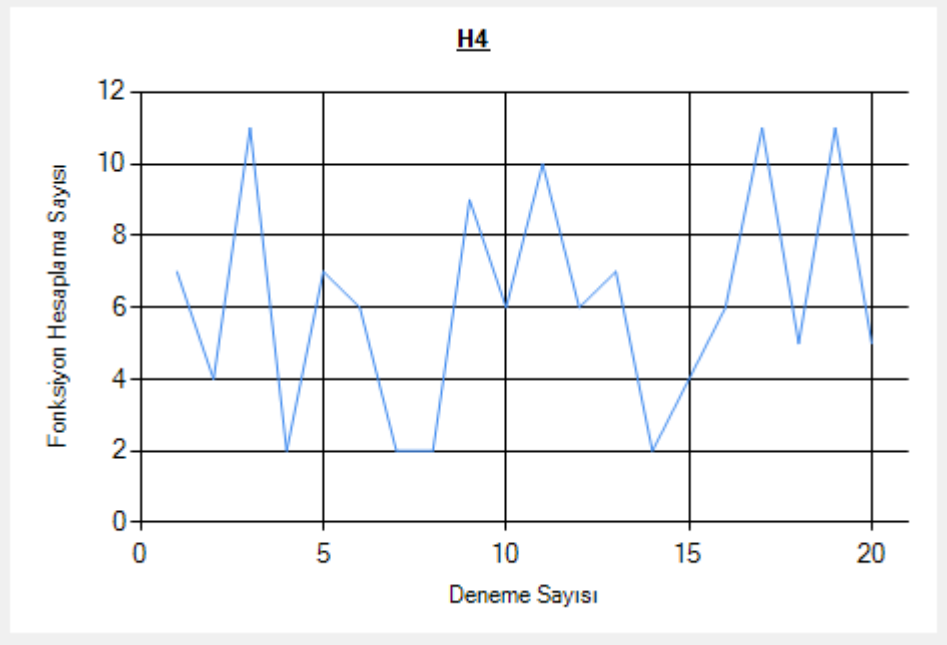
Şekil 5.26 A^3 ile 1.Avcı etmeni için 16x16 boyutlu ortamda fonksiyon hesaplama sayısı



Şekil 5.27 A^3 ile 2.Avcı etmeni için 16x16 boyutlu ortamda fonksiyon hesaplama sayısı



Şekil 5.28 A^3 ile 3.Avcı etmeni için 16x16 boyutlu ortamda fonksiyon hesaplama sayısı



Şekil 5.29 A^3 ile 4.Avcı etmeni için 16x16 boyutlu ortamda fonksiyon hesaplama sayısı

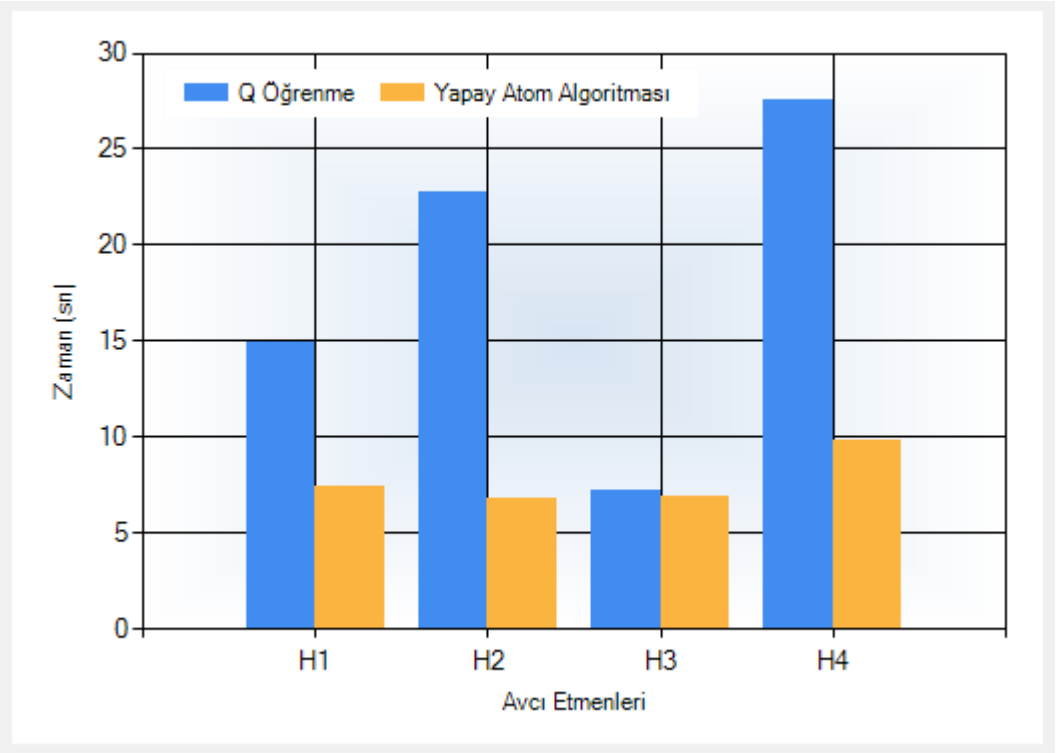
Yapay atom algoritması ile yapılan uygulamanın sonuçlarına bakıldığında tüm avcı etmenlerinin de hedefe ulaşabildiği görülmüştür. Yapay atom algoritmasında da Q öğrenmede olduğu gibi avcı etmenlerinin hedefe ulaşan en kısa yolu bulabildikleri gözlenmiştir. Ortamın büyüklüğüne bağlı olarak hedefe giden yolların uzunluğu, yapılan işlemler ve fonksiyon hesaplama sayıları da değişmektedir.

Yapay atom algoritmasında atomlardaki elektron değerleri rassal veriler ile başladığından dolayı her denemede farklı sonuçlar elde edilebilmektedir. Denemelerdeki iterasyon sayısı ve fonksiyon hesaplama sayısına bağlı olarak bazı denemelerde hedefe ulaşan yol bulunamazken, bazı denemelerde ise ilk işlemde hedefe ulaşıldığı ve en kısa yolun bulunabildiği görülmüştür.

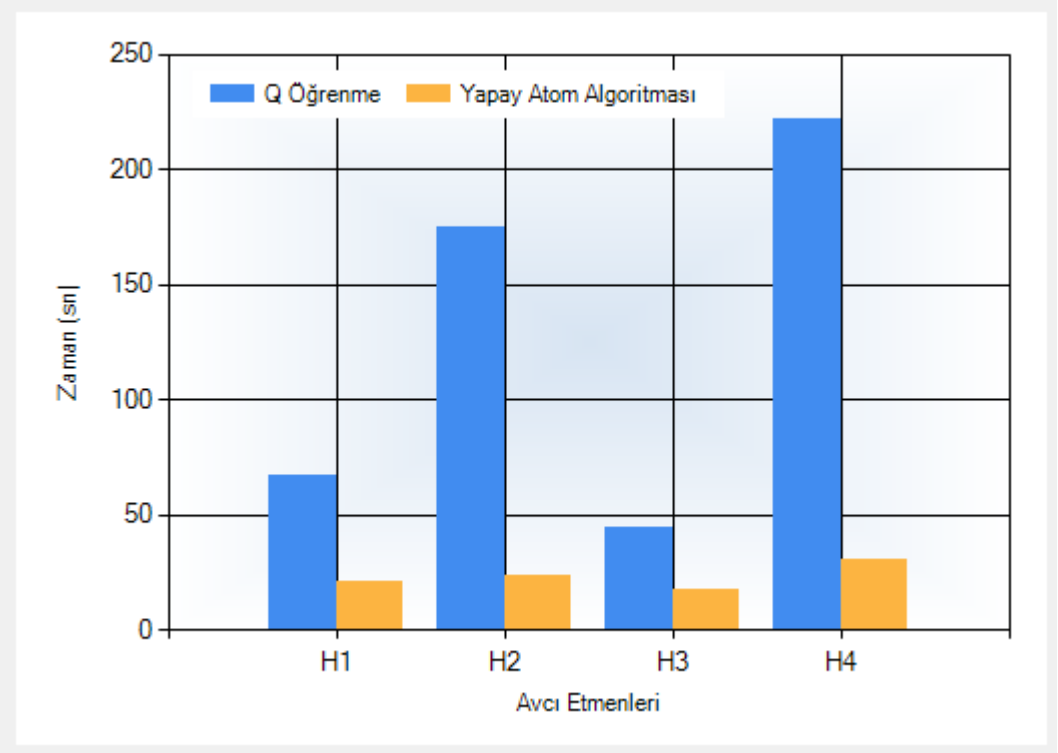
Avcı etmenlerine ait grafikler kıyaslandığında hedefe daha yakın olan etmenlerin hem yol uzunluğu hem de fonksiyon hesaplama sayısı bakımından, hedefe uzak olan etmenlere göre daha kısa sürede ve daha az hesaplama ile sonuca ulaştıkları görülmektedir.

Bu uygulamada ayrıca hedefe ulaşma süreleri için de sonuçlar elde edilmiştir. Hedefe ulaşma sürelerine bakıldığında Yapay atom algoritmasının daha kısa sürede hedefe ulaştığı görülmüştür. Ayrıca uygulama ortamı büyüdüğünde Q öğrenmenin hedefe ulaşma süresinin yapay atom algoritmasına göre daha fazla arttığı tespit edilmiştir. Avcı etmenlerinin av etmenine olan uzaklığı arttıkça Yapay atom algoritması ile Q öğrenme algoritması arasındaki hedefe ulaşma süresi farkı da belirginleşmektedir. Örneğin hedefe en yakın olan 3.avcı etmeni için bakıldığında hedefe ulaşma süresi her iki algoritmada da birbirine yakın değerler gösterirken, hedefe en uzak etmen olan 4.avcı etmeninde bu farkın daha çok belirginleştiği görülmüştür.

Q öğrenme ve Yapay atom algoritması için 8x8 ve 16x16 boyutlu ortamlarda hedefe ulaştıkları ortalama süreler karşılaştırılmış ve aşağıdaki grafikler elde edilmiştir.



Şekil 5.30 Q öğrenme ve Yapay atom algoritmasının 8x8 boyutlu ortamda hedefe ulaşma ortalama süreleri



Şekil 5.31 Q öğrenme ve Yapay atom algoritmasının 16x16 boyutlu ortamda hedefe ulaşma ortalama süreleri

6. SONUÇ

Bu çalışmada öncelikle takviyeli öğrenme algoritmalarından olan Q öğrenme algoritması kullanılarak av avcı probleminin çözümü elde edilmiştir. Daha sonrasında, kimyasal bileşiklerin oluşumuna dayanan metasezgisel bir algoritma olan Yapay Atom Algoritması(A^3), av avcı problemine uyarlanarak problemin çözümü elde edilmiştir.

Yapay atom algoritmasının av avcı probleminin çözümünde başarılı olduğu ve bu tür optimizasyon problemlerine uygulanabilirliği görülmüştür. Avcı etmenlerinin av etmenine ulaşmak için izlemesi gereken en kısa yolun hem Q öğrenme algoritması hem de Yapay atom algoritması tarafından bulunduğu tespit edilmiştir. Ayrıca Yapay atom algoritmasının hedefe ulaşma süreleri açısından daha performanslı çalıştığı görülmüştür. Algoritmanın rassal değerler ile başlamasına rağmen, her adımda yapılan Kovalent Bağ ve İyonik Bağ işlemleri ile sonuca kısa sürede ulaştığı görülmüştür. Uygulama ortamı büyütüldüğünde ise Q öğrenme algoritmasının sonuca çok daha geç ulaştığı, Yapay atom algoritmasının ise daha kısa sürede sonuca ulaştığı tespit edilmiştir.

Yapay atom algoritmasının arama ve labirent problemlerine uygulanabilir olduğu ve bu problemlerde başarı sağladığı görülmüştür.

KAYNAKLAR

Alpaydın, E. (2010), *Introduction to Machine Learning,*The MIT Press, London, p 3.

Anonymous. (2013), http://en.wikipedia.org/wiki/Machine_learning

Bilgin A.T. (2013), *Çok Ajanlı Kaçma Kovalama Problemlerine Takviyeli Öğrenme Yaklaşımı,* Yüksek Lisans Tezi, TOBB Ekonomi ve Teknoloji Üniversitesi, Ankara.

Birbil S.I., Fang S.C. (2003), *An electromagnetism-like mechanism for global optimization*, Journal of Global Optimization, 25.

Coppin, B. (2004), *Artificial Intelligence Illuminated*, Jones and Barlett Publishers, USA. ISBN 0-7637-3230-3

Çentik G. (2013), *Makine Öğrenmesi Yöntemlerinin Polisomnografik Verilere Uygulanması*. Doktora Tezi, Trakya Üniversitesi, Edirne.

De Castro, L.N., Von Zuben F.J. (2002), *Learning and optimization using the clonal selection principle*, IEEE Transactions on Evolutionary Computation, 6.

Dorigo M., Stützle T. (2004), *Ant Colony Optimization*, The MIT Press, London.

Goldberg D.E., *Genetic algorithm in search: Optimization and machine learning*, Boston, MA, Kluwer Academic Publishers, 1989.

Hacıbeyoğlu M. (2006), *Çoklu Etmen Mimarisi ve Takviyeli Öğrenme*, Yüksek Lisans Tezi, Selçuk Üniversitesi, Konya.

Karaboğa D, Baştürk B. (2007), *A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm*, Journal of Global Optimization, 39.

Karadoğan A., Karcı A. (2013), *Artificial atom algorithm for reinforcement learning*. In: Proceedings of 2nd International Eurasian Conference on Mathematical Sciences and Applications, Saraybosna, Bosna Hersek.

Karcı A. (2012), *A new meta-heuristic algorithm based on chemical process: Atom algorithm*. In:Proceedings of 1st International Eurasian Conference on Mathematical Sciences and Applications, Priştine, Kosova.

Kaya M. (1998), *Akıllı Yazılım Etmenleri*, Yüksek Lisans Tezi, Fırat Üniversitesi, Elazığ.

Kaya M. (2003), *Çoklu Etmen Takviyeli Öğrenmeye Veri Madenciliği Tabanlı Yeni Yaklaşımlar*, Doktora Tezi, Fırat Üniversitesi, Elazığ.

Kennedy J., Eberhart R.C. (1995), *Particle swarm optimization*, In proceedings of IEEE international conference on neural Networks, Australia, 4.

Lee K.S., Geem Z.W. (2005), *A new metaheuristics algorithm for continues engineering optimization: Harmony search theory and practice*, Computer Methods in Applied Mechanics and Engineering, 194.

Mitchell, T. (1997). *"Machine Learning"*, McGrawHill. ISBN 0-07-042807-7, p2.

Öztemel, E. (2003), *Yapay Sinir Ağları*, Papatya Yayıncılık Egitim, İstanbul.

Simon, P. (2013), *TooBigtoIgnore: The Business Case forBig Data*. Wiley, ISBN 978-1118638170, p. 89

Stone, P., Veloso M. (2000), "*Multiagent Systems: A Survey from a Machine Learning Perspective*", To Appear in *Autonomous Robotics* volume 8, number 3.

Sutton, R.S., Barto A.G. (2012), *Reinforcement Learning: An Introduction*, Second Edition, The MIT Press, London.

Szepesvari, C. (2009), *Algorithms for Reinforcement Learning*, Morgan & Claypool Publishers, USA.

Weiss, G., (1999), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, The MIT Press, ISBN 0-262-23203-0, p28-29.

Yang X.S. (2009), *Firefly algorithms for multimodal optimization*, In stochastic algorithms: Foundations and applications, SAGA, 2009. LNCS, 5792.

Yardımcı T. (2011), *Makine Öğrenmesi Teknikleri İle Rss Besleme Yönetimi*. Yüksek Lisans Tezi, Gazi Üniversitesi, Ankara.

Yıldırım A.E., Karcı A. (2013), *Solutions of travelling salesman problem using genetic algorithm and atom algorithm*, In: Proceedings of 2nd International Eurasian Conference on Mathematical Sciences and Applications, Saraybosna, Bosna Hersek.

ÖZGEÇMİŞ

Ad Soyad : Ahmet KARADOĞAN

Doğum Yeri ve Tarihi : MALATYA-1988

Adres : İnönü Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü, MERKEZ/MALATYA

E-Posta : ahmet.karadogan@inonu.edu.tr

Lisans : Selçuk Üniversitesi Bilgisayar Mühendisliği Bölümü (2005-2010)

Mesleki Deneyim : 3A Bilişim, Konya (2010-2010)

Datatime Yazılım, Ankara (2010-2012)

TEZDEN TÜRETİLEN YAYINLAR/SUNUMLAR

- **Karadoğan A.,** Karcı A. (2013), *Artificial atom algorithm for reinforcement learning*. In: Proceedings of 2nd International Eurasian Conference on Mathematical Sciences and Applications, Saraybosna, Bosna Hersek.