İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

a_i, b_i, c_i	*i*.nci generatörün yakıt maliyeti katsayıları
FC_i	Yakıt Maliyeti
P_i	*i*-biriminin çıkış gücü
SC_i	Birim devreye alma (başlangıç) maliyeti
σ_i, δ_i ve τ_i	*i*-biriminin başlangıç maliyet katsayıları
SD_i	Birim durdurma maliyeti
P_G	Bütün birimlerin ürettiği toplam güç
D	Talep gücü
R	Yedek Güç
P^{min}	Birimin sağlayabileceği minimum güç
P^{max}	Birimin sağlayabileceği maksimum güç
MUT	Birimin minimum devrede kalma süresi
MDT	Birimin minimum devre dışı kalma süresi
UR	Birimin artan güç eğim oranı
DR	Birimin azalan güç eğim oranı
T^{OFF}	Birimin devre dışı kaldığı süre
T^{ON}	Birimin devrede kaldığı süre
N_{pop}	Popülasyondaki birey sayısı
X_{rate}	Seçim oran
N_{keep}	Elit birey sayısı
μ	Mutasyon Oranı
r	Automaton için aksiyon sayısı
a, b	Lineer pekiştirme düzeni için öğrenme parametreleri
$\underline{\alpha}$	Automaton için çıkış (aksiyon) kümesi
α_i	Automatonun *i*.nci aksiyonu
$\alpha(n)$	*n*-zaman adımındaki automaton aksiyonu
$\beta(n)$	*n*-zaman adımında çevrenin cevabı
c_i	*i*.nci aksiyon için penaltı olasılığı
$p_i(n)$	*n*-zaman adımında *i*.nci aksiyonun seçilme olasılığı
$\underline{p}(n)$	Aksiyon olasılıkları vektörü
M_0	Saf şans automaton için ortalama penaltı
$M(n)$	*n*-zaman adımında automatonun ortalama penaltı değeri
$G(\cdot)$	Durum-çıkış automatonu için çıkış fonksiyonu
$H(\cdot,\cdot)$	Automaton için çıkış fonksiyonu
P_{kritik}	Kritik güç
V_{kritik}	Kritik gerilim
δ_{kritik}	Kritik yük açısı
A, B, C, D	İletim hattının devre sabitleri
P	Aktif güç
Q	Reaktif güç
S	Kompleks güç

KISALTMA LİSTESİ

EP	Evrimsel Programlama
GA	Genetik Algoritma
LA	Öğrenen Automata
SA	Simulated Annealing (Benzetimli Tavlama)
IEE	Institue of electrical Engineers
IEEE	Institute of Electrical and Electronics Engineers
UCP	Unit Commitment Problem
ED	Ekonomik Yük Dağıtımı

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Bu tez çalışmasında, elektrik güç sistemlerinin optimum birim yüklenme problemi, gerilim kararlılığı kriterleri dikkate alınarak sistemin en uygun çalışma noktası evrimsel programlama, genetik algoritma ve öğrenen automata tekniklerinin birlikte kullanılması ile belirlenmektedir. Bu şekilde sistem için mümkün olan en düşük maliyet değeri ve iyi bir kararlı işletim noktası elde edilmiş olmaktadır.

Çalışmalarım sırasında yakın ilgilerini gördüğüm, danışman hocam sayın Prof.Dr. Celal KOCATEPE'ye, beni yetiştiren fedakar anne ve babama, çalışmalarım sırasında sabır ve hoşgörü ile bana destek olan sevgili eşime ve çalışma arkadaşlarıma teşekkürü bir borç bilirim.

ÖZET

Elektrik enerji sistemlerinin optimum işletimi, üretim birimlerinin farklı işletme maliyet karakteristiklerinden dolayı özellikle büyük ölçekli güç sistemlerinde çok büyük öneme sahiptir. Ancak bir takım teknik kısıtlardan dolayı en ekonomik işletim noktası her zaman sistem ve/veya üretim birimlerinin işletilmesi açısından arzu edilen çalışma bölgesinde olmayabilir. Güç sistemlerinin işletilmesinde ekonomik kriterlerin yanında kararlılık ve güvenilirlik gibi birtakım diğer kriterler de oldukça önemlidir. Genellikle, sözkonusu bu işletim kriterleri aynı çalışma noktalarından oluşmaz.

Bu çalışmada, güç sisteminin en ekonomik çalışma noktası birim yüklenme problemi dikkate alınarak elde edilmiş olup, sistem gerilim kararlılığı da her bir yük profili için nihai çözüme dahil edilmiştir. Bu tez çalışmasında optimum ekonomik işletim maliyeti, evrimsel programlama ve genetik algoritma kullanılarak elde edilmiş olup, daha sonra sistem gerilim kararlılığı farklı baralar için hesaplanmıştır ve bu iki kriter öğrenen automata (learning automata) tekniği kullanılarak hem iyi bir kararlılık bölgesi hem de ekonomik bir işletim noktası elde edilmiştir.

Anahtar kelimeler: Birim yüklenme problemi, ekonomik yük paylaşımı, gerilim kararlılığı, evrimsel programlama, genetik algoritma, öğrenen automata.

ABSTRACT

The optimum operation of the electrical energy systems is of significant importance in terms of the different cost characteristics of the generation units especially in large scale systems. But due to the technical limitations, the most economical operation point, is not always the desired operating point for the system and/or the generation units. Some other criteria such as stability and reliability are also important in the operation of power systems. Generally, these operation criteria do not occur at same points.

In this study, the most economical operating point of the power system is obtained considering the unit commitment problem, and the system voltage stability is taken into account in the consequent solution for each load profile. In this thesis, the optimum economic operation cost is obtained using the evolutionary programming and the genetic algorithms and then the voltage stability is calculated for different system buses. Finally, these two criteria are used within the "Learning Automata" technique to get not only an excellent stability region but also an economical operation point.

Keywords: Unit commitment problem, economic dispatch, voltage stability, evolutionary programming, genetic algorithm, learning automata.

1. GİRİŞ

Elektrik enerjisinin üretimi, iletimi ve dağıtımı ile yükümlü bulunan kuruluşların başlıca amacı, üretilen enerjinin tüketicilere sürekli, ucuz ve kaliteli bir şekilde sunulmasını sağlamaktır. Bu amaç; kalite, ekonomi ve güvenilirliğin uygun bir birleşimi olarak belirlenmektedir. Elektrik enerji sisteminden beslenecek çeşitli tüketici merkezlerinin gereksinim duyacakları elektrik enerjisinin bu niteliklerle sağlanması gereği, enerji sağlayanlara bir takım zorluklar getirmekte ve bütün ülkelerde ulusal yatırımların önemli bir bölümünün elektrik enerji sektörüne ayrılmasını zorunlu kılmaktadır. Bundan dolayı elektrik enerji sistemi planlama problemi, ekonomik, güvenilir ve geleceğe yöneltilmiş, esnek üretim, iletim ve dağıtım sistemi planlarının üretilmesi bakımından büyük önem taşımaktadır.

Elektrik enerji sistemi dediğimizde ilk aklımıza gelen eleman, enerji üretimini gerçekleştiren enerji üretim birimleri yani generatörlerdir. Enterkonnekte sistemin birer parçası olan generatör grupları tarafından üretilen elektrik enerjisinin tüketicilere ulaştırılması enerji iletim ve dağıtım hatları sayesinde sağlanmaktadır. Enerji üretimini gerçekleştiren generatör birimlerinin her birinin fiziksel karakteristiği ve bu birimlerde kullanılan yakıtın cinsi ve maliyeti farklı olabilmektedir. Bu etkenler nedeniyle, üretilen enerjinin maliyeti de değişmektedir. Bundan dolayı elektrik enerjisi üretiminin toplam maliyetini en ekonomik olarak elde etmek için, enerjiyi üretecek generatör birimlerinden üretim maliyeti en düşük olanlar sistemde devrede olmalıdır.

Günlük, aylık ve yıllık olarak sistemden talep edilen güç miktarı değişkenlik göstermektedir. Generatör birimlerinin devreye alınması veya devre dışına çıkarılması belli bir plana göre yapılmalıdır. Genellikle bu planlamalar minimum bir saatlik periyotlar içerisinde günlük veya iki günlük olarak yapılmaktadır. Generatörlerin ürettiği elektrik enerjisinin toplam maliyetini en ekonomik seviyede tutmak için, bu birimlerin devrede olup, olmamaları durumunu belli bir işletim dilimi içerisinde gerçekleştirme problemi "Birim Yüklenme Problemi" olarak ifade edilmektedir.

Bu çalışmada, birim yüklenme problemi ile ilgili yeni bir yöntem geliştirilmiştir. Birim yüklenme problemi, evrimsel algoritma ve genetik algoritmanın en ekonomik maliyetin hesaplanmasında kullanılması, V-Q duyarlılık analizi ile kararlılık şartlarının kıyaslanarak en uygun kararlılık şartının elde edilmesi ve öğrenen automata yöntemi ile bu iki problemi belli kriterler içerisinde sağlayan bir optimum işletim noktasının elde edilmesi incelenmiş ve

analizlerle ortaya konmuştur.

Elektrik enerjisinin üretimini sağlayan generatör birimlerinin işletim planlamasında, bu birimlerin işletimde olup, olmamalarına ek olarak devrede bulunan birimlerin üretecekleri güç miktarının da belirlenmesi gerekmektedir. "Ekonomik Yük Dağıtımı" olarak isimlendirilen bu problemin birim yüklenme problemi içerisinde çözümü gerekmektedir. Birim yüklenme problemi yük tahmininden farklı olarak 24 veya 48 saatlik işletim dilimleri için sürekli bir şekilde yapılmaktadır.

Birim yüklenme problemi ile ekonomik yük dağıtımı arasındaki ilişkiyi şu şekilde açıklayabiliriz. Bir elektrik güç sisteminde enerji üretimini gerçekleştirecek birimlerin hangilerinin işletimde olacağı birim yüklenme problemi ile planlanmaktadır. Bu planlama yapılırken ekonomik yük dağıtımının da bir alt problem olarak elde edilmesi gerekmektedir. Birim yüklenme probleminin bir özel durumu olarak, sistemdeki bütün birimlerin devrede olması veya işletmede olan birimlerin belli olması durumunda, 1 saatlik bir periyot için generatörlerin üretecekleri güç miktarlarının belirlenmesi sadece ekonomik yük dağıtım probleminin çözümü ile elde edilmektedir.

Birim yüklenme problemi bir minimizasyon problemidir, fakat kısıtları ve analitik çözüm yöntemi bulunmaması nedeniyle zor bir çözüme sahiptir. Problemin çözümü için amaçlanan, minimum toplam yakıt maliyetinin elde edilmesidir. Birim yüklenme probleminde yakıt maliyeti; üretim sırasında oluşan yakıt maliyetinin yanı sıra generatör birimlerin başlatma ve durdurma maliyetlerini de içermektedir.

Birim yüklenme probleminin kısıtları, generatör birimlerinin fiziksel sınırlarından dolayı oluşmaktadır. Bir elektrik güç sisteminde generatörlerin ihtiyaç duyulduğu an hemen devreye alınması veya belirli bir süre geçmeden devre dışına çıkarılması mümkün olmamaktadır. Bunun yanında her bir generatör biriminden çekilecek güç de generatörün üretebileceği maksimum ve minimum güç sınırları içerisinde kalmalı ve bu değer aşılmamalıdır. Bu sayılanlara ek olarak sistemden üretilmesi istenen toplam talep gücünün yanında, generatör birimlerinin karşılaşabileceği herhangi bir arıza durumu için önceden belirlenmiş bir oranda yedek gücün sistem tarafından karşılanması da sağlanmalıdır [Sapmaz, 2004].

Birim yüklenme problemi birçok kısıtlamalara sahip bir optimizasyon problemidir. Generatörlerin üretecekleri güç limitleri, birimlerin minimum devrede veya devre dışında kalma süreleri, sistemin karşılaması gereken talep gücü ve yedek gücü bu problemin içerdiği en önemli kısıtlardandır. Birim yüklenme problemi, sahip olduğu kısıtları, karmaşık çözüm

yapısı ve analitik çözümünün olmamasından dolayı, çözümü basit olmayan, geniş ölçekli, nonlineer yapıya sahip karmaşık bir optimizasyon problemidir.

Optimizasyon problemlerinin çözümünde kullanılan bütün çözüm yöntemleri birim yüklenme probleminin çözümünde de kullanılmıştır. Bunlar arasında öncelikli liste yöntemi, dinamik programlama, Lagrange indirgeme yöntemi, karışık tamsayı programlaması, doğrusal programlama gibi klasik optimizasyon yöntemleri ve bunlara ek olarak yapay sinir ağları, genetik algoritmalar, tabu arama tekniği, evrimsel programlama ve benzetimli tavlama gibi modern yöntemler de sayılabilir [Padhy, 2004].

Birim yüklenme probleminin çözüm uzayı çok geniş olup analitik bir çözüm yolu yoktur. Bu da problemin çözümünde, olasılıksal olarak çözüm uzayında gezinen yöntemlerin çok daha iyi sonuçlar vermesini sağlamaktadır. Evrimsel hesaplama tekniklerinin yapısının olasılıksal bir yapıya dayanması ve nonlineer programlamadaki etkinliklerinin yüksek olması sonucu bu türdeki optimizasyon problemlerinin çözümündeki başarıları kanıtlanmıştır. Kullandığımız modern bilgisayar yapılarının, işlem hızlarının ve bellek miktarlarının yüksek oluşu, evrimsel programlama ve genetik algoritma yöntemlerinin daha kısa sürede en iyi sonuçları üretmelerine olanak sağlamaktadır.

Bu çalışmada, evrimsel programlama birim yüklenme probleminin çözümünde kullanılmıştır. Genetik algoritma ise, birim yüklenme probleminin bir parçası olarak, işletimde bulunan (ON konumunda olan) generatörlerin her birinin ne kadar yük üreteceğini belirleyen ekonomik yük dağıtımı probleminin çözümünün elde edilmesinde uygulanmıştır.

“Akıllı” sistemler içinde önemli bir yer alan evrimsel algoritmalar, güç sistemlerinde zor optimizasyon problemlerinin çözümünde kullanılmaya başlanmıştır. Evrimsel hesaplama tekniklerinin en çok kullanılanı, evrimsel programlama ve genetik algoritmalardır.

John Holland ve öğrencileri tarafından 1960’larda geliştirilen genetik algoritmalar (GA), doğal seçim ve genetik mekaniğine dayalı arama algoritmalarıdır [Goldberg, 1989]. Bu algoritmalar, çözüm kümesini kodlayarak, topluluk kümesi içinde, olasılığa bağlı operatörler ile sadece amaç fonksiyonu değerlerini kullanarak arama yaparlar. Bu algoritmaların, özellikle lineer olmayan, çok değişkenli, çok amaçlı optimizasyon problemlerinin çözümünde güçlü bir arama yöntemi oldukları kanıtlanmıştır [Goldberg, 1989]. Güç sistemlerinde genetik algoritma uygulamaları, ekonomik yük dağıtımı, arıza yerinin bulunması hidrotermal elektrik üretim santrallerinin optimal işletimi, reaktif güç kompanzasyonu, güç sistemlerinde faz kaydırıcıların optimal yerleşimi, büyük dağıtım sistemlerinin planlanması, güvenilirlik

indislerinin hesaplanması ve güç sistemi modernizasyonu gibi konular üzerinde yaygınlaşmıştır [Aygen, 2002].

Genetik algoritmalar, genel anlamda doğadaki canlıların geçirmiş oldukları değişimin incelenip problem çözümüne aktarılmasıdır. Rasgele oluşmuş bir popülasyonun üyeleri içinden seçilen ebeveynler tarafından türetilen yavru bireyler ile yeni bir popülasyon meydana getirilir. Her yeni popülasyon çözümün kalitesini arttırmaktadır. Belirlenen bir nesil sayısına ulaşıldığında yeni popülasyon üretimi sona erdirilir. En iyi çözüme sahip birey belirlenmiş olur [Goldberg, 1989].

GA ile çözüm arama işlemi canlıların biyolojik gelişimine benzemektedir. Canlılar doğarlar ve üreyerek yeni nesilleri yetiştirirler. Doğal seçim mekaniğine bağlı olarak, canlı grubu arasından yapısı bakımından en uygun olan birey hayatta kalmaktadır. Her yeni nesil rasgele bilgi değişimi ile oluşturulan diziler içinden hayatta kalanların birleşmesi ile elde edilir. Bu algoritmalar, özellikle lineer olmayan çok değişkenli, çok amaçlı optimizasyon problemlerinin çözümünde güçlü bir arama yöntemidir.

Genetik algoritmalar, son yıllarda kompleks optimizasyon problemlerinin büyük bir kısmını çözmede yeterliliği ve basitliğinden dolayı başarıyla kullanılmıştır [Man vd., 1996]. Evrimsel temelli tekniklerin başarısının ana sebebi, teknikler arasındaki etkili bilgi alışverişi ve özel bir problem için aday çözümlerin bir popülasyon ile paralel olarak çok yönlü araştırma yapabilmeleridir.

Evrimsel hesaplama tekniklerinin bir diğeri de evrimsel programlamadır. J. Fogel tarafından 1960 yılında geliştirilen bu yöntem genetik algoritmaya yapı olarak çok benzemektedir. Genetik algoritmadan farkı, ebeveynler ve yavruları arasındaki davranışsal bağa dikkat etmektedir. Genetik algoritmada bulunan seçme ve çaprazlama operatörleri bu yöntemde bulunmamaktadır. Ebeveyn bireylerden yavru bireylerin oluşumu mutasyon sayesinde sağlanmaktadır [Fogel vd., 1966].

Elektrik enerjisine olan talep, teknolojideki gelişmelere ve nüfus artışına bağlı olarak her geçen gün artış göstermektedir. Enerji miktarının artması ile birlikte elektrik enerjisinin kalitesinin de arttırılması, düşünülmesi gereken bir durumdur. Enerji üretim merkezleri ile tüketim merkezleri arasındaki uzaklıkların büyük olması, enerji iletiminde, yüksek gerilim hatlarının kullanılmasını zorunlu kılmaktadır. Dolayısıyla güç kalitesi ile ilgili problemler de günümüzde daha etkin bir biçimde ortaya çıkmaktadır [Uzunoglu, 2000].

Daha güvenilir ve kaliteli bir enerji için elektrik güç sistemlerinin planlanması, işletimi ve kontrolünde, enerji sisteminin kararlılığının da dikkate alınması gerekir. Enerji sistemlerinde kararlılık, bir bozucu etkiye maruz kalan sistemin bozucu etki sonrası, tekrar bozucu etki öncesi şartlarına dönme yeteneğidir. Kararlılık temel olarak, geçici, dinamik ve sürekli hal kararlılığı olarak sınıflandırılabilir. Senkron çalışmanın ani olarak kaybının söz konusu olduğu ani yük değişimleri ve enerji iletim hatlarındaki kısa devreler gibi bozucu etkilere sistemin cevabına geçici hal kararlılığı, birkaç saniyelik geçici olay süresinden sonra mekanik regülatörlerin de devrede olduğu birkaç dakikalık sürede sistemin bozucu etkiye cevabına dinamik hal kararlılığı ve beklenen yük değişimleri ya da küçük bozucu etkilere sistemin cevabına ise sürekli hal kararlılığı adı verilir [Tacer, 1990; Anderson, 1994].

Ekonomik ve çevre ile ilgili nedenler yüzünden elektrik enerjisinin uzun mesafelere iletimi söz konusu olduğu ve gerilim kararsızlığı birçok şebeke çökmelerine neden olduğu için, gerilim kararlılığı son yıllarda büyük önem kazanmıştır.

Bu çalışmada, sistemin kararlılığı ile ilgili olarak V-Q duyarlılık endeksi kullanılmış ve sistemlerin kararlı çalışma şartları altında analizler gerçekleştirilmiştir.

Mühendislik alanında ödün vermeye ait karar, birçok kalite kriterinin eşzamanlı olarak ele alınması ile karakterize edilebilir. Güç sistemi alanında, genellikle birden fazla güç sistem özelliğini eşzamanlı olarak optimize etmek gerekmektedir. Hem ekonomik olarak güç üretiminin sağlanması ve hem de üretilen gücün kararlı olması, elektrik enerji sektöründe önemli iki faktördür, fakat bunlar genelde birbirleriyle çelişkili olaylardır. Bu tür çok nesneli problemlerin çözümü, en iyi uzlaşmayı sağlayan çözümü gerektirmektedir [Lee vd., 1998; 2005].

Son 30 yıl içerisinde önemli bir ilgi alanı bulan öğrenen automata (learning automata, LA) [Narendra ve Thathachar, 1989], içeriği bilinmeyen (tanımsız) ortamlarda (çevrelerde) işlem yapabilen ve bir öğrenme işlemi aracılığıyla performanslarını geliştiren adaptif karar verici aygıttır. İlk olarak psikolojik ve biyolojik bakış açısıyla insan davranışını tanımlamak amacıyla kullanılmıştır. LA, tüm mühendislik alanları üzerinde de önemli bir etki oluşturarak, yüksek derecede belirsizlik ve nonlineerlik ile karakterize edilen modelleme ve kontrol problemlerine uygulanabilmektedir. Düşük hesaplama kompleksliği ile hızlı ve akıcı yakınsama özelliğine sahip olmaları dolayısıyla geniş bir uygulama alanı bulmuştur.

Bu çalışmada öğrenen automata, sistemin işletimi için ayrı ayrı belirlenen en ekonomik işletim noktası ve en uygun kararlı çalışma noktası arasında belirli kriterler dahilinde bir

işletim noktası belirleyerek sistemin optimum çalışmasının elde edilmesinde kullanılmıştır.

Öğrenme, geçmiş deneyimin bir neticesi olarak davranışta meydana gelen herhangi bir kalıcı değişme olarak tanımlanmaktadır. Bu yüzden bir öğrenme sistemi, zaman içerisinde davranışını kesin bir hedefe doğru geliştirme yeteneğine sahip olmalıdır. Matematiksel anlamda öğrenme sisteminin hedefi, açık şekilde tanımlanamayan fonksiyonun optimizasyonudur [Narendra vd., 1974].

LA, matematiksel psikoloji alanında geliştirilmiştir. Bu alandaki ilk çalışmalar Bush, Mosteller [Bush ve Mosteller, 1958] ve Atkinson vd. [Atkinson vd., 1965] tarafından yapılmıştır. Tsetlin [Tsetlin, 1973], deterministik (belirleyici) automatonu rasgele ortamlarda çalışan bir öğrenme modeli olarak tanıtmıştır. LA teorisi üzerinde yapılan çalışmaların çoğu, Tsetlin, Varshavski ve Vorontsova [Varshavski vd., 1963] tarafından kurulan akımı takip etmektedir. Bu akım, deterministik automata ile kıyaslandığında, durum sayısındaki azalmaya neden olan aksiyon olasılıklarının güncellendiği stokastik automata olarak tanımlanmaktadır.

Yukarıda belirtildiği üzere bu tez çalışmasında, birim yüklenme probleminin çözümünde yeni bir yöntem hedeflenmiştir. Bu çözümün gerçekleştirilmesinde orijinal olarak, evrimsel programlama yöntemi generatör birimlerinin sistemde devrede olup olmadıklarını belirlemek için, genetik algoritma devredeki generatörlerin ne kadar güç üreteceklerini belirlemek için, V-Q duyarlılık endeksi sistemin farklı işletim noktalarındaki kararlılığını karşılaştırmak için, öğrenen automata ise sistemin ekonomik ve kararlı işletim noktalarının kıyaslanarak optimum bir işletim noktası elde etmek için kullanılmıştır.

Çalışmanın bölümleri aşağıda verilmiştir:

Çalışmanın birinci bölümünde, birim yüklenme problemi ve çözüm yöntemleri ile ilgili genel bir literatür araştırılması yapılmıştır. Tezin konusu ve tez çalışmasında kullanılacak yöntemler hakkında bilgi verilmiştir.

Çalışmanın ikinci bölümünde, ayrıntılı olarak birim yüklenme problemi açıklanmıştır. Probleme ait olan kısıtlar hakkında bilgi verilmiştir. Problemin çözümünde kullanılan yöntemler maddeler halinde incelenmiştir.

Çalışmanın üçüncü bölümünde, sistemin kararlılığı hakkında genel bir bilgi verilerek, V-Q duyarlılık endeksi ile ilgili bilgi verilmiştir.

Çalışmanın dördüncü bölümünde, evrimsel hesaplama tekniklerinden genetik algoritma ve

evrimsel programlama anlatılmıştır. Genetik algoritmanın işlem operatörleri açık olarak incelenerek bu yöntemin kullanılmasına basit bir örnek verilmiştir. Evrimsel algoritmanın işlem basamakları benzer olarak anlatılmıştır.

Çalışmanın beşinci bölümünde, öğrenen automata anlatılarak, automata yapıları ve davranış biçimlerine ilişkin bilgi verilmiştir.

Çalışmanın altıncı bölümünde, tez çalışmasında ortaya konan yöntem anlatılmıştır. Ayrıca bu bölümde tezde kullanılan yöntemlerin birbirleri ile olan ilişkileri verilmiştir.

Çalışmanın yedinci bölümünde, önerilen yöntem iki adet örnek sistem üzerinde ayrıntılı bir şekilde uygulanmıştır. Son bölümde ise sonuçlar ve önerilere yer verilmiştir.

Bu tez çalışmasında birim yüklenme problemi için yeni bir yöntem önerilmiş olup, yenilik olarak evrimsel programlama ve genetik algoritma kombinasyonu kullanılarak birim yüklenme probleminin analizi gerçekleştirilmiş ve sonuçlar analiz sürecinde kullanılmıştır. Yine bu tez çalışmasında yenilik olarak öğrenen automata sistemin hem ekonomik hem de kararlılık anlamında optimum bir işletim noktasını elde etmek için kullanılmıştır ve sonuçların elde edilmesi adımında değerlendirilmiştir. Bu tez çalışmasında diğer bir yenilik olarak birim yüklenme probleminin çözümünde V-Q kararlılık endeksinin kullanımı, sistem kararlılığının elde edilmesi işleminde gerçekleştirilmiştir.

2. BİRİM YÜKLENME PROBLEMİ (UNIT COMMITMENT PROBLEM)

2.1 Giriş

Bu çalışmada birim yüklenme probleminin çözümü çeşitli metotların kullanılmasıyla orijinal olarak gerçekleştirildiğinden birim yüklenme probleminin açıklanması ve çözüm yöntemleri bu bölüm içerisinde verilmiştir.

Elektrik enerji sistemlerinde, günlük yük eğrisinde meydana gelen değişimler enerji üretiminin, bir veya iki gün için saatlik değerler temel alınarak planlanmasını gerektirmektedir. Enerji üretiminin planlaması yapılırken, sistemin üreteceği enerjinin maliyetinin minimum olması istenir. Elektrik enerji sisteminin ekonomik olarak enerji üretiminin gerçekleştirmesinde, sahip olduğu günlük yük talep eğrisine bağlı olarak, sistemdeki hangi üretim biriminin devreye alınacağı veya çıkarılacağı ve devredeki birimlerin üretim kapasitesinin ne olacağı belirlenir. Bu işlemin yanında, sistemdeki enerji üretim birimlerinin dinamik değişim kısıtlamaları, enerji sisteminin fiziksel kısıtlamaları gibi birçok kısıtlamaların da göz önüne alınması gerekir.

Bir elektrik şebekesinde çoğunlukla karşılaşılan ve çözülmesi gereken bu temel problem, kısa bir işletme periyodu için şebekeye ait toplam talep eğrisinde belirtilen tahmini yük miktarlarının işletme kısıtlarını da dikkate alarak hangi birimler tarafından ekonomik bir şekilde karşılanacağının tespit edilmesidir. Bu problem genellikle, güç sistem birim üretim planlaması veya birim yüklenme problemi (unit commitment problem) olarak adlandırılır.

Birim yüklenme problemi genel olarak, bir enerji sisteminde 24 veya 48 saatlik dönemler için, sistemin yük değişimi ve elde edilen maliyet göz önüne alınarak en uygun generatör birimlerinin devreye alınıp çıkarılması olarak ifade edilmektedir. Birim yüklenme problemindeki parametreler, üretim maliyetlerinden ve sistem kısıtlamalarından meydana gelmektedir.

Birim yüklenme problemi, potansiyel maliyet kazançlarından dolayı yaklaşık son 30 yıldır aktif bir araştırma alanı olarak literatürdeki yerini korumaktadır. Bazı çalışmalarda, işletme maliyetlerindeki %1'lik bir iyileştirme ile kurulu gücü 10 GW olan bir elektrik şebekesi için yaklaşık 10-30 milyon USD'lik bir yıllık kazanç sağlandığı tahmin edilmektedir [Tseng, 1996].

Birim yüklenme probleminin çözümü için birçok optimizasyon yöntemi önerilmiştir. Lagrange indirgeme (relaxation) metotları, bu tür problemlerin çözümünde en yaygın olarak kullanılan yöntemler arasındadır [Bertsekas vd., 1983; Guan vd., 1992]. Problemin modellenmesinde, geleneksel birim yüklenme eşitlikleri gerçek uygulama şartlarını tamamen formülize edemez. Generatör birimlerini veya elektrik şebeke karakteristiklerini ya da üretim ve tüketim arasındaki etkileşimi tanımlayan kısıtlar, güç sistemleri işletiminde artan bir önem kazanmaktadır. Bu çeşit kısıtlar örnek olarak, üretim birimine ait eğim oranı (ramp rate), şebeke iletimi ve üretim-tüketim dengesi ile ilgili kısıtları içermektedir.

Üretim maliyetleri; generatörler devrede iken oluşan yakıt maliyetleri, birimlerin devreye girmesi sırasında meydana gelen başlatma maliyetleri ve generatörlerin devre dışı kalmaları sırasında göz önüne alınan durdurma maliyetleridir [Momoh, 2001].

Birim yüklenme probleminin amacı, planlanan zaman dilimi boyunca güç sisteminin toplam maliyetini verilen kısıtlar çerçevesinde minimize etmektir.

Bu tez çalışmasında, sadece termik santral birimlerinin olduğu bir güç sistemi dikkate alınmıştır. Bir buhar ünitesinin işletimi esnasında ortaya çıkan dinamik davranış nispeten karmaşık bir yapıdadır. Bir termik generatör birimi için, devreye almak için gerekli süre ve bununla ilgili başlatma (start up) maliyetleri birimin ne zamandan beri devre dışı kaldığına bağlıdır. Bunun nedeni, generatör biriminin güç üretebilmesi için yüksek sıcaklıkta buhara ihtiyaç duymasıdır.

Generatör ne kadar süre devre dışında kalmış ise kazandan o kadar çok ısı kaybolmuştur. Bu nedenle, kazandaki suyun yeniden ısıtılmasında daha uzun bir zamana ve dolayısıyla daha yüksek bir maliyete ihtiyaç duyulmaktadır. Neticede, birim yüklenme probleminin formülasyonu iki farklı maliyet teriminin minimizasyonunu gerektirmektedir. Birinci terim, güç üretimi ile ilgili olup, doğrudan tüketilen yakıt miktarına bağlıdır. İkinci terim ise, generatörü devreye almak için gerekli olan başlatma maliyeti olup, bu maliyet kazan sıcaklığının bir fonksiyonudur [Tseng, 1996].

Üretim planlaması sonucuna göre devrede olması gereken birimler, aynı zamanda yükte meydana gelen dalgalanmaları veya eleman arızalarını içeren acil durumları karşılamak için yedek güç değerlerini de temin etmelidir. Başka bir ifade ile beklenmedik bir durum ortaya çıktığında, mevcut o anki devrede olan birimlerden ilave güç temin edilebilmelidir. Bu güç yedek kapasite gücü (spinning reserve) olarak bilinmektedir. Yedek güç, devrede olan birimlerin toplam kapasitesi ile talep edilen güç arasındaki farkı ifade eden bir terimdir.

Yedek güç miktarı genellikle, bölgesel güvenilirlik miktarları dikkate alınarak tahmin edilen gücün belirli bir oranı cinsinden verilir. Örnek olarak, her bir saat için yedek güç, tahmin edilen güç miktarının en az %5'i olmalıdır [Tseng, 1996].

Birim yüklenme probleminde göz önüne alınan kısıtlar, sistemin fiziksel özelliklerinden kaynaklanmaktadır. Generatör birimlerine güç sağlayan türbinlerin ısıtılması ihtiyacı, enerji üretimi sırasında türbinlerin dönmeye başlamasının ve durdurulmasının belirli bir süre alması, sistem tarafından karşılanması gereken talep güç ve yedek güç kısıtların nedenlerini oluşturur.

Bu bölümde birim yüklenme problemi hakkında genel bir bilgi verilerek matematiksel ifadesi tanımlanacaktır. Daha sonra bu problemin uygulanması sırasında göz önüne alınması gereken kısıtlamalar detaylı olarak incelenerek problemin genel formülasyonu verilecektir. Yine bu bölüm içerisinde birim yüklenme probleminin optimizasyonu için literatürde kullanılmış yöntemler ve çalışmalar özet olarak açıklanmıştır.

2.2 Birim Yüklenme Probleminin Maliyetleri

2.2.1 Amaç Formülasyonu

Birim yüklenme problemi, bir optimizasyon problemidir ve amacı toplam üretim maliyetinin (TC) değerini belli kısıtları da göz önüne alarak minimum yapmaktır. Buna göre Eşitlik (2.1) ile verilen toplam üretim maliyetinin değeri minimum olmalıdır [Padhy, 2004].

$$\min \mathrm{TC} = \sum_{t=1}^{T} \sum_{i=1}^{N} \{\mathrm{FC}_i(t) \cdot u_i(t) + \mathrm{SC}_i(t) \cdot u_i(t) \cdot (1 - u_i(t)) + \mathrm{SD}_i(t) \cdot u_i(t-1) \cdot (1 - u_i(t))\} \qquad (2.1)$$

burada $u_i(t)$ i-biriminin t-saatinde devrede olması durumunda "1", olmaması durumunda ise "0" değerini almaktadır. N, sistemdeki toplam üretim birimlerinin sayısını gösterirken, T=1,2,…t ise yük eğrisine denk düşen saatlik çalışma periyotlarını ifade etmektedir.

2.2.2 Yakıt Maliyeti

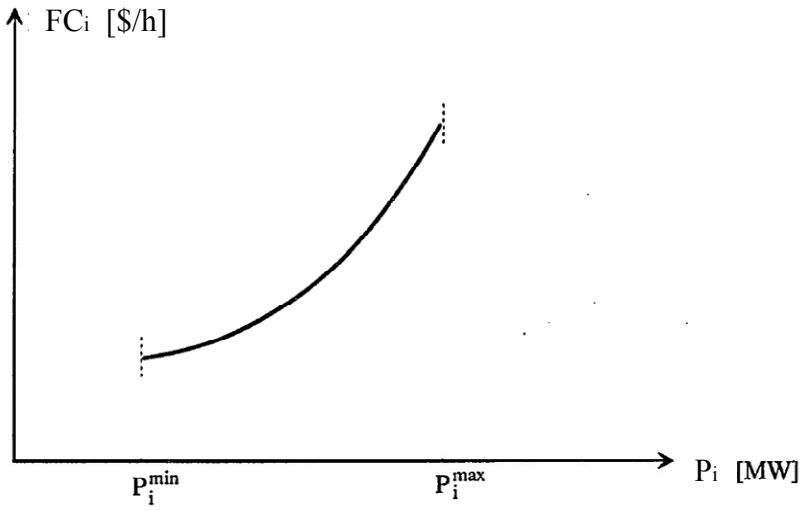
Yakıt maliyeti (FC), güç sistemindeki generatör birimlerinin devrede olması durumunda, generatörden çekilen güce bağlı olarak değişen bir maliyet fonksiyonudur. Güç sisteminde, termik santraller genel olarak iki farklı türbin yapısına sahip olmaktadır. Bu türbin tipleri buharlı ve yanmalı türbinler olarak ifade edilebilir. Fosil yakıtların yanması veya nükleer

enerjinin işlenmesi sonucu elde edilen buhar ile buhar türbinleri döndürülür. Yanmalı türbinlerde ise doğal gaz veya damıtılmış petrolün yanması ile türbinin hareketi sağlanır.

Bu iki farklı yapıya sahip üretim birimi, yapısal farklılıklarına rağmen matematiksel olarak benzer şekilde modellenebilir. Genel olarak, termik üretim birimlerinin yakıt maliyeti Eşitlik (2.2)'de görüldüğü gibi ifade edilebilir. Üretim birimlerinin çıkış güçlerine bağlı maliyetleri genel olarak Şekil 2.1 ile gösterilebilir [Swarup ve Yamashiro, 2002].

$$\mathrm{FC}_i(t) = a_i + b_i P_i(t) + c_i P_i^2(t) \tag{2.2}$$

burada a_i, b_i ve c_i i.generatör biriminin yakıt maliyet katsayıları olup P_i, i.generatörün t saatindeki çıkış gücüdür.



Şekil 2.1 Termik bir birimin yakıt maliyet fonksiyonu

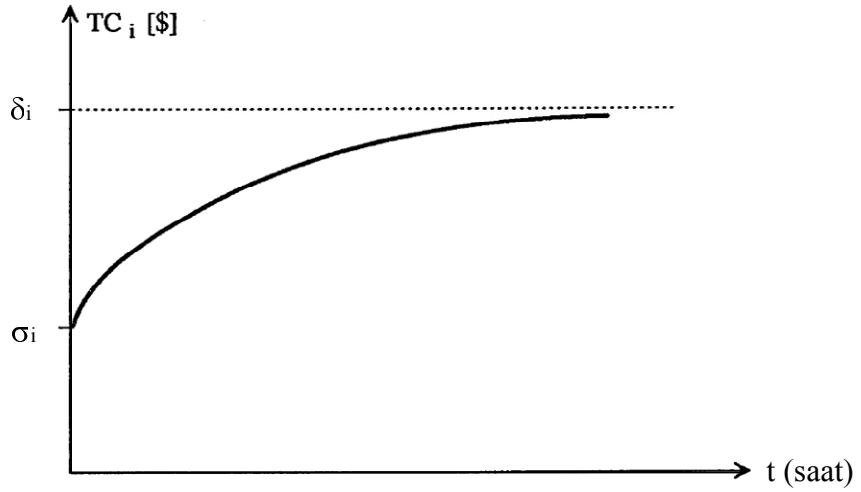
2.2.3 Başlangıç Maliyeti

Başlangıç maliyeti (SC), sistemdeki bir generatör biriminin devre dışı iken, üretim yapmak üzere devreye alınması sırasında oluşan maliyettir. Termik üretim birimleri genelde iki şekilde devre dışı bırakılabilir, bunlardan birincisi generatör biriminin şebeke ile bağlantısı kesildikten sonra türbinlerin soğumaya bırakılmasıdır. Bu durumda, eğer birim ortam sıcaklığına kadar soğuduktan sonra türbinler tekrar ısıtılıp birim devreye alınırsa bu duruma soğuk başlatma adı verilir. Soğuk başlatma maliyeti Eşitlik (2.3) ile verilmiştir [Swarup ve

Yamashiro, 2002; Wood ve Wollenberg, 1996].

$$SC_i(t) = \sigma_i + \delta_i \left[1 - e^{\left(\frac{-T_{i,t}^{Off}}{\tau_i} \right)} \right] \tag{2.3}$$

burada σ_i, δ_i ve τ_i i-biriminin başlangıç maliyet katsayıları olup, $T_{i,t}^{Off}$ t-saatinde i-biriminin işletmede olmadığı (OFF) birim zaman değeridir. Şekil 2.2'de başlangıç maliyetinin zaman ile değişimi verilmiştir.



Şekil 2.2 Termik bir birimin başlangıç maliyet fonksiyonu

Termik birimleri devre dışı bırakmanın bir diğer yolu türbinleri yatık (banked) konumda bırakmaktır. Yatık konumda iken generatörün şebeke ile bağlantısı kesilerek üretim yapmaması sağlanır, fakat türbinlerin soğumasına da izin verilmez, bu sayede generatörün tekrar devreye alınması kolaylaşır. Üretim birimleri kısa süreli olarak devre dışı kalacakları zaman yatık konumda bırakılır. Yatık konumdaki üretim biriminin başlatma maliyeti Eşitlik (2.4) ile verilmiştir.

$$SC_b = \rho \cdot T^{Off} \tag{2.4}$$

burada ρ, yatık konum maliyet sabiti ve T^{OFF}, yatık konumda kalma süresidir [Momoh, 2001].

2.2.4 Durdurma Maliyeti

Durdurma maliyeti (SD), enerji üretiminde bulunan bir generatör biriminin devre dışına alınması sırasında oluşan maliyettir. Herhangi bir yakıt maliyeti veya sınırlamadan ilişkisi olmayıp, genellikle durdurmak için gerekli çalışanlardan kaynaklanan maliyet olarak düşünülür. Sistemdeki bir birimi devre dışına almak için, mevcut birim çalışanları haricinde ek bir işgücüne gerek duyulmadığından, genellikle durdurma maliyeti sıfır kabul edilir [Swarup ve Yamashiro, 2002; Padhy, 2004].

$$SD_i(t)=0 \tag{2.5}$$

2.3 Birim Yüklenme Problemine Ait Kısıtlar

Birim yüklenme problemini çözümü zor bir optimizasyon problemi haline getiren yapı, kısıtlar olarak ifade edilebilir. Birim yüklenme problemi kısıtlarını, sistem kısıtları ve birim kısıtları olmak üzere iki ana başlık altında toplayabiliriz. Sisteme ait kısıtlar, sistemin yük dengesinin sağlanması yani talep gücünü karşılaması ve yedek güç ihtiyacıdır. Birim kısıtları ise generatör birimlerinin fiziksel ve karakteristik özelliklerinden kaynaklanmaktadır [Padhy, 2004].

2.3.1 Yük Dengesi

Bir güç sisteminde, enerji üreten birimlerin her t anında üretmiş olduğu toplam güç, sistemden karşılanması istenen talep güce eşit veya daha fazla olmalıdır. Burada sistemden talep edilen güç dendiğinde, sadece tüketim birimleri tarafından harcanan güç değil, sistemde bulunan bütün kayıp güçlerin de göz önüne alındığı dikkate alınmalıdır. Yük dengesi Eşitlik (2.6)'daki gibi ifade edilmektedir [Swarup ve Yamashiro, 2002; Padhy, 2004].

$$P_G = \sum_{i=1}^{N} u_i(t) P_i(t) \geq D(t) \tag{2.6}$$

burada P_G, bütün generatör birimlerinin ürettiği toplam gücü, $D(t)$ ise t-saatinde sistemden talep edilen toplam gücü göstermektedir.

2.3.2 Yedek Güç

Elektrik güç sistemlerinde, generatör birimleri tarafından üretilen güç miktarının talep edilen yükü karşıladığı gibi, bazı durumlarda sistemin güvenilirliğini sağlamak için ek bir güç miktarını da karşılaması istenir. Bunun nedeni, herhangi bir nedenle üretim birimlerinden bir veya birkaçının üretim dışı kalması durumunda bu farkın ek güç tarafından karşılanmasıdır. Bu ek güç "yedek güç (spinning reserve)" olarak tanımlanmaktadır. Elektrik enerji sistemlerinde yedek güç genellikle, üretim birimlerinden bir kısmının maksimum güç limiti değerinin altında çalıştırılarak sağlanır. Yedek güç ifadesi Eşitlik (2.7)'da verilmiştir [Swarup ve Yamashiro, 2002; Sun vd., 2006].

$$\sum_{i=1}^{N} u_i(t) P_i^{\max} \geq D(t) + R(t) \tag{2.7}$$

burada $R(t)$, t-saatinde sistemin karşılaması gereken yedek gücü göstermektedir.

2.3.3 Birimlerin Üretim Limitleri

Güç sistemindeki bütün generatör birimlerinin üreteceği güç değerlerinin Eşitlik (2.8)'den de görüldüğü üzere minimum ve maksimum değerleri bulunmaktadır [Padhy, 2004; Sun vd., 2006; Momoh, 2001].

$$P_i^{\min} \leq P_i(t) \leq P_i^{\max} \tag{2.8}$$

burada P_i i-generatör biriminin t-saatinde üreteceği gücü, min ve max indisleri ise bu generatörün sağlayabileceği en düşük ve en yüksek güç miktarını belirtmektedir.

2.3.4 En Düşük Devrede ve Devre Dışında Kalma Süresi

Generatörleri tahrik eden türbinlerin ani sıcaklık değişimlerinden zarar görmeleri, ayrıca her birimin enerji üretimi için devreye alınması ve devre dışı edilmesi sırasında oluşan maliyetler gibi nedenlerden dolayı, birimlerin üretime başlaması ve durdurulması işlemlerinden sonra belirli bir saatin geçmesi istenmektedir. Bu da birim yüklenme problemi için kısıtlama oluşturur.

Generatörlerin minimum devrede kalma süresi Eşitlik (2.9) ile gösterilmiştir [Swarup ve

Yamashiro, 2002].

$$T_{i,t}^{On} > \mathrm{MUT}_i \tag{2.9}$$

burada MUT_i, ilgili generatör biriminin minimum devrede kalma süresidir. Benzer olarak generatörlerin minimum devre dışı kalma süresi de Eşitlik (2.10)'daki gibi verilebilir [Swarup ve Yamashiro, 2002].

$$T_{i,t}^{Off} > \mathrm{MDT}_i \tag{2.10}$$

eşitlikte MDT, ilgili generatörün biriminin minimum devre dışı kalma süresini göstermektedir.

2.3.5 Eğim Oranı Limitleri

Generatör birimlerinin yapısındaki termik ve mekanik limitler, generatörden çekilecek yükte oluşacak ani değişimlere izin vermez. Bundan dolayı, bir generatör biriminden çekilecek güçte saatlik olarak meydana gelebilecek değişim belirli bir oranı geçmemelidir. Eğim oranı olarak ifade edilen generatör birimlerine ait bu limitler Eşitlik (2.11)'daki gibi tanımlanabilir.

$$\begin{aligned} P_i(t) - P_i(t-1) &\le \mathrm{UR}_i \\ P_i(t-1) - P_i(t) &\le \mathrm{DR}_i \end{aligned} \tag{2.11}$$

burada UR_i, generatör biriminin bir saatlik periyot içerisinde gücünde oluşabilecek maksimum artış miktarını, DR_i ise benzer olarak bir saat içerisinde birimin gücünde meydana gelebilecek maksimum azalma miktarını ifade etmektedir [Sun vd., 2006].

2.4 Birim Yüklenme Probleminin Çözüm Metotları

Birim yüklenme problemi birbirine bağlı iki optimizasyon problemi olarak ele alınabilir. Bunlardan birincisi birleşimsel problem olarak ifade edebileceğimiz üretim yapacak birimlerin seçilmesi, ikincisi ise nonlineer problem olarak ele alınabilecek seçilmiş birimlerin üreteceği güç değerlerinin elde edilmesidir. Olası çözüm uzayının yüksek boyutlu olması problemin çözümü için ciddi bir problem oluşturmaktadır. Birim yüklenme problemini çözmek için kullanılan yöntemler dört gruba ayrılabilir:

- Klasik optimizasyon metotları; (dinamik programlama, Lagrange indirgemesi vb.)
- Bulgusal (Heuristic) metotlar; (öncelikli sıralama metodu, uzman sistemler vb.)
- Yapay zeka metotları; (Genetik algoritma, Tabu arama tekniği, sinir ağları vb.)
- Hibrit algoritmalar; (iki veya daha fazla metodun birleşimi).

2.4.1 Yoğun Listeleme Metodu (Exhaustive Enumeration)

İlk zamanlarda birim yüklenme probleminin çözümü, üretim birimlerinin bütün muhtemel kombinasyonlarının listelenmesi ve bunlar arasından en düşük işletim maliyetini veren kombinasyonun optimal çözüm olarak belirlenmesi yöntemine dayanmaktaydı. Kerr vd. ve Hara vd., tarafından birim yüklenme problemi, Florida Güç Şirketi için yoğun listeleme metodu kullanılarak başarılı bir şekilde gerçekleştirilmiştir [Kerr vd., 1966; Hara vd., 1966]. Bu metot her ne kadar büyük ölçekli elektrik şebekeleri için uygun bir yöntem olmasa da, oldukça doğru sonuçlar verebilmektedir.

2.4.2 Öncelikli Sıralama Metodu (Priority Listing)

Öncelikli sıralama metodu, ilk olarak üretim birimlerinin en düşük işletim maliyeti karakteristiklerini dikkate alarak bir öncelik sıralaması yapar. Bu şekilde önceden belirlenmiş sıralama, sistem yükleri karşılanacak şekilde birim yüklenme problemine uygulanır. Burns vd. ve Lee tarafından öncelikli sıralama yöntemi birim yüklenme problemine başarıyla uygulanmıştır [Burns vd., 1975; Lee, 1988]. Shoults vd, enerji transfer kısıtlarını dahil ederek öncelikli sıralama yöntemini kullanarak, doğrudan ve nümerik açıdan oldukça verimli bir algoritma ortaya koymuştur [Shoults vd., 1980]. Tek ve çok bölgeli birim yüklenme problemi için, Lee tarafından klasik bir indislemeyi baz alan öncelikli sıralama yöntemi başarıyla kullanılmıştır [Lee, 1991].

2.4.3 Dinamik Programlama (Dynamic Programming)

Dinamik programlama, birim yüklenme problemine ilk olarak uygulanan optimizasyon tabanlı bir yöntemdir. Çok farklı ölçekteki problemlerin çözümünde kullanılabilen ve özel şebeke karakteristiklerine de kolay bir şekilde uyarlanabildiği için oldukça etkin bir metottur

[Snyder vd., 1987]. Bu yöntem ile bir saatlik işletimi etkileyen kısıtları dikkate almak nispeten kolay olup, bu kısıtlar temel olarak ekonomik yük paylaşımını ve çözüm metodolojisini etkiler. Dinamik programlama yönteminin dezavantajlarını aşağıda belirtildiği gibi sıralamak mümkündür [Li vd., 1998; Padhy, 2004]:

- İşletim esnasında göz önünde bulundurulan herhangi bir saat dilimi için birimlerin seçimi ve devreye alınması ile ilgili bir sınırlamaya ihtiyaç duyar,
- Minimum devrede/devre dışı kalma kısıtlarının tam olarak optimize edilememesi,
- Zamanın bir fonksiyonu olan devreye alma maliyet kısıtlarının probleme tam olarak adapte edilememesi.

Lowery, birim yüklenme probleminin çözümü için, dinamik programlama yönteminin pratik uygulanabilirliğini ortaya koymuştur [Lowery, 1966]. Happ kişisel bilgisayarlarla yapılan çözüm ile manüel çözüm sonuçları karşılaştırmıştır ve 100-makineli bir güç sistemi için toplam maliyette %1'in üzerinde, yaklaşık 7000 USD'ye karşılık gelen bir kazanç elde ettiğini ifade etmiştir [Happ vd., 1971].

2.4.4 Lagrange İndirgeme Metodu (Lagrangian Relaxation)

Lagrange indirgeme metodu, sistem kısıtları (güç dengesi ve yedek güç) için Lagrange çarpanlarını kullanır ve Lagrange fonksiyonunu biçimlendirmek için amaç fonksiyonundaki ilişkili penaltı terimlerini ekler [Kazarlis vd., 1996]. Lagrange indirgeme yöntemi düzenli olarak bir çok elektrik şirketi tarafından kullanılmaktadır [Cohen ve Wan, 1987; Nieva vd., 1987; Tong ve Shahidehpour, 1989]. Bu yöntemin birim yüklenme probleminde kullanımı dinamik programlamaya göre çok daha yeni bir yöntemdir. Bu yöntem büyük ölçekli güç sistemlerinde daha etkili sonuçlar üretebilmektedir. Ayrıca belirli şebekelerin karakteristik-lerini modellemek için kolay bir şekilde düzenlenebilmektedir. Birimlere ilişkin kısıtları probleme eklemek nispeten basittir. Lagrange indirgeme yönteminin başlıca dezavantajı ise bölgesel optimum noktalara takılabilme özelliğidir [Padhy, 2004].

Merlin vd., yeni bir Lagrange indirgeme yöntemini kullanarak birim yükleme problemine uygulamıştır ve sonuçların doğruluğunu Electricite De France şebekesi verilerini kullanarak göstermiştir [Merlin ve Sandrin, 1983]. Aoki vd., Lagrange indirgeme yöntemini üç farklı

ünite çeşidini (termik ünite, yakıt-kısıtlı termik ünite ve pompalı hidrolik ünite) içeren büyük ölçekli güç sistemine başarılı bir şekilde uygulamıştır [Aoki vd., 1987]. Wang vd., eğim oranı kısıtlarını ve rotor yorulma etkisini dikkate alarak ekonomik bir üretim planlaması için detaylı bir matematiksel model vermiştir [Wang vd., 1994]. Ma vd., optimal güç akışını birim yüklenme formülasyonuna dahil etmiştir [Ma ve Shahidehpour, 1999].

2.4.5 Tabu Arama Tekniği (Tabu Search)

Tabu arama tekniği, birçok farklı kombinasyonlardaki optimizasyon problemlerine başarılı bir şekilde uygulanabilen güçlü bir optimizasyon yöntemidir. Bu yöntem, esnek hafıza sistemi sayesinde bölgesel minimum noktalara takılmadan işlem yapabilme kabiliyetine sahiptir [Mantawy vd., 1998]. Mori vd., öncelik sıralama metodunu tabu arama tekniğine adapte ederek birim yüklenme problemi için bir algoritma ortaya koymuştur [Mori vd, 1996; 2000]. Rajan vd., yapay sinir ağları tabanlı tabu arama tekniğini kullanarak birim yüklenme problemini çözmüştür [Rajan vd., 2002]. Lin vd., ekonomik yük paylaşımı problemine geliştirilmiş bir tabu arama tekniği ile çözüm bulmuştur [Lin vd., 2002].

2.4.6 Tavlama Benzetimi (Simulated Annealing)

Tavlama benzetim tekniği, ilk olarak 1982 yılında Kirkpatrick, Gela ve Vecchi ile ardından 1985 yılında Cerny tarafından birbirlerinden habersiz olarak ortaya atılmıştır [Padhy, 2004]. Birim yüklenme probleminin çözümü için iki farklı parametreye ihtiyaç duyulmaktadır. Bu parametreler, üretim biriminin durumunu (binary olarak) temsil eden U ve V değişkenleri ile birim çıkış gücü P'dir. Problem iki alt probleme ayrılabilir: ***i)*** U ve V'nin farklı kombinasyonlarındaki optimizasyon problemi, ***ii)*** Nonlineer P çıkış gücünün optimizasyonu. Bu nedenle, tavlama benzetim yöntemi birim yüklenme probleminin çözümüne uygun bir metottur. Mantawy vd., birim yüklenme problemine bu metodu uygulamıştır, olumlu ve olumsuz özelliklerini şu şekilde sıralamıştır: bu yöntem uzun CPU zamanına ihtiyaç duymasına karşın, başlangıç çözümlerinden ve kompleks matematik yapıdan bağımsızdır [Mantawy vd., 1998].

2.4.7 Uzman Sistemler (Expert Systems)

Uzman sistemler yapay zekanın en önemli uygulama alanlarından biridir. Bu tür bir sistem belli bir alanda uzmanlık kazanmış kişilerin bilgisine dayanarak çözüm aramaktadır [Castillo ve Alvarez, 1991]. Uzman sistemler hem makine hem de insan müdahalesine ihtiyaç duyulan uygulamalarda kullanılmaktadır. Bir uzman sistem üç ana yapıdan oluşur. Bunlar, kural tabanı, veri tabanı ve kural çözümleyici olarak sıralanabilir. Uzman sistemler, çözümü kesin olarak belli olmayan problem türlerinde algoritmaya sahip olmayan uzmanlıkları problemin çözümüne dahil eden bilgi uygulamalarıdır. Mokhtari vd., generatör birimlerinin işletiminin düzenlenmesi için güç sistem operatörlerine yardımcı olan uzman sistem tabanlı bir danışmanlık geliştirmiştir [Mukhtari vd.,1988]. 1991 yılında Salam vd. tarafından bir uzman sistemi geliştirilmiş ve işletimsel olarak uygun bir çözüm elde etmek için dinamik programlama tabanlı birim yüklenme programına bir önişlemci olarak kullanılmıştır [Salam vd., 1991].

2.4.8 Bulanık Sistemler (Fuzzy Systems)

İlk olarak 1965 yılında Zadeh tarafından ortaya atılan bulanık küme mantığı ile, belirsizlik içeren sözel ve sayısal bilgiler insan aklına en uygun bir biçimde matematiksel olarak modellenebilmektedir. Bu fikir, klasik küme teoreminin genelleştirilmiş bir hali olarak kabul edilebilir. Birim yüklenme problemi esasında kompleks bir karar verme işlemi olup her bir saat için tahmini talep yükü ile devreye alınacak olan birimler arasında işletim maliyetini minimum yapacak şekilde bir denge kurma işlemidir. Talepteki belirsizlikler ve üretim birimlerindeki devre dışı kalmalar nedeniyle birim yüklenme problemini uygun bir şekilde çözüme ulaştırmak zor olmaktadır [Chowdhury vd., 1990]. Araştırmacılar belirsizliğin mevcut olduğu şartlarda stokastik yöntemlerin deterministik yöntemlere göre daha iyi sonuç verdiğini ortaya koymuştur, fakat bu yöntemlerin de kendine özgü kısıtları vardır [Takriti vd., 1996]. Saneifard vd., bulanık mantık yönteminin birim yüklenme problemine uygulanmasını göstermiştir [Saneifard vd., 1997]. Bu yöntem ile sistem davranışının ve sistem karakteristiğinin iyi bir tanımlaması yapılabilmekte, hatta nihai matematiksel formulasyonlara ihtiyaç duymadan sistem cevabını elde edilebilmektedir.

2.4.9 Yapay Sinir Ağları

Yapay sinir ağları (YSA), insan beyninin nöronlardan oluşan yapısını ve öğrenme yöntemlerini inceleyerek, biyolojik sinir ağlarının davranışını modellemek için tasarlanmıştır. YSA'lar özellikle öğrenme üzerinde yoğunlaşmış, lineer olmayan sistemlerde ve sisteme ait bilginin kesinlik içermediği veya hatalı olduğu durumlarda çözüme ulaşmak için uygundur. Sasaki vd., birleşimsel optimizasyon probleminin çözümünü, özellikle birim yüklenme problemine Hopfield sinir ağını uygulayarak incelemiştir [Sasaki vd., 1992]. Önerilen sinir ağı 24 saatlik bir periyot üzerinden 30 birimlik bir sistem için birim yüklenme probleminin çözümünde kullanılmış ve oldukça iyi sonuçlar elde edilmiştir. C. Wang vd., eğim-oranı kısıtlarının da dahil edildiği birim yüklenme problemi için bir yapay sinir ağı modeli önermiştir [Wang vd., 1993]. Birim yüklenme probleminin geleneksel Hopfield ağının yapısı içerisinde kesin bir şekilde işlem göremediği bulunmuştur. Bu yüzden Walsh vd., ayrık ve sürekli terimleri içeren daha genel bir enerji fonksiyonu veren, sinirler arasında yeni bir bağlantı şeklinin olduğu arttırılmış bir ağ yapısı sunmuştur [Walsh vd., 1997].

2.4.10 Evrimsel Programlama

Evrimsel programlama, genetik algoritma benzeri mutasyon tabanlı bir evrimsel hesaplama tekniğidir. Yang vd. ve Juste vd. birim yüklenme problemine, rasgele değişimler, çekişmeler ve seçimler sonucu evrim geçiren iddialı bir çözüm popülasyonunun kullanıldığı bir evrimsel programlama yaklaşımı sunmuştur [Yang vd., 1996; Juste vd., 1999]. Rajan vd., birim yüklenme probleminin çözümü için, evrimsel programlama tabanlı çözüm metodu önermiştir [Rajan vd.,2004; 2006]. Contreras vd., kar tabanlı birim yüklenme problemine bir evrimsel programlama yaklaşımı getirmiştir [Contreras vd., 2006]. Yine Rajan ve Christober vd., yatık konumda kalma kısıtları ile birim yüklenme problemini evrimsel programlama tabanlı yöntemler ile çözmüştür [Rajan, 2006; Christober vd., 2004].

2.4.11 Karınca Kolonisi Arama Algoritması (Ant Colony Search Algorithm)

Karınca kolonisi algoritması birleşimsel optimizasyon problemlerinde iyi sonuçlar veren bir tekniktir. Bu algoritmanın geliştirilmesinde tabiattaki karınca kolonilerinin yiyecek arama yapılarından faydalanılmıştır. Karıncalar öncelikle yuvalarının etrafında rasgele dolaşarak

yiyecek aramak için keşif yaparlar. Bir yiyecek kaynağı bulduklarında ise yiyeceğin kalitesini ve miktarını inceleyip bir miktar örneği yuvalarına taşırlar. Yuvaya dönüş sırasında diğer karıncaların da yiyecek kaynağını bulabilmeleri için feromen (pheromone) adı verilen bir kimyasal maddeyi geçtikleri yerlere bırakırlar. Bırakılan bu izler diğer karıncalara da yiyecek kaynağını bulmada rehberlik eder. Feromen maddesi aracılığıyla yapılan dolaylı iletişim, yuva ile yiyecek kaynağı arasındaki en kısa yolun bulunmasını sağlar. Gerçek karıncaların bu davranışı ile karınca sistemi esinlenilmiştir. İlk karınca kolonisi algoritması M. Dorigo tarafından 1991 yılında gerçekleştirilmiştir [Dorigo vd., 2004]. Bu algoritmaya "Karınca Sistemi" adı verilmiştir. Karınca sistemi birim yüklenme problemi gibi birleşimsel optimizasyon problemlerine uygulanmaktadır [Padhy, 2004]. Sisworahardjo ve El-Kalib, karınca kolonisi arama yöntemini kullanarak birim yüklenme problemini çözmüştür [Sisworahardjo ve El-Kalib, 2002]. Benzer olarak Huang, karınca kolonisi sistemi tabanlı optimizasyon yaklaşımlarını kullanarak hidroelektrik üretim planlamasını çözmüştür [Huang, 2001].

2.4.12 Genetik Algoritmalar

Son 30 yıl içerisinde evrimsel prensipleri ve makine öğrenmesini baz alan problem çözme sistemlerine ilgide büyük bir artış olmuştur [Sood vd., 2003; Orero ve Irving, 1997]. Bu tür sistemler, potansiyel bir çözüm popülasyonunu içermektedir. Bu popülasyonda, bireylerin uygunluk değerini ve bazı "genetik" operatörleri temel alarak seçim işlemi yapılmaktadır. Scheble vd., genetik algoritmayı birim yüklenme problemine birden yedi güne kadar uygulamıştır [Scheble vd., 1996]. Birim yüklenme problemleri için genetik algoritma uygulamalarının yapılabilirliği hem küçük hem de büyük boyutlu problemler için incelenmiştir [Shanti vd., 2004]. Maifeld vd. yaptıkları çalışmalarla, belirli mutasyon operatörleri ile genetik algoritma yöntemini kullanarak yeni bir birim yüklenme planı sunmuştur. Önerilen algoritmanın doğruluğu, farklı şebekeler üzerinde birim yüklenme problemi için Lagrange indirgeme yöntemi kullanılarak elde edilen değerler ile kıyaslanarak gösterilmiştir. Yang vd. 1997'de birim yüklenme problemini çözmek için paralel bir genetik algoritma yaklaşımını önermiştir. Swarup vd. tarafından geniş ölçekli birim yüklenme problemi için kromozomların gösteriminde ve çözüm uzayının kodlanmasında yeni bir strateji geliştirmiştir. Xing vd. 2002 yılında enerji sözleşmelerini içeren birim yüklenme problemini genetik algoritmaya dayalı olarak gerçekleştirmiştir. Cheng vd. yapılan çalışmalarla, Lagrange indirgemesi ve genetik algoritma ile birim yüklenme problemini çözmüştür.

Bu tez çalışmasında birim yüklenme probleminin çözümünde evrimsel algoritma, genetik algoritma, V-Q duyarlılık kriteri ve öğrenen automata yöntemleri karma bir biçimde orijinal olarak kullanılmış ve çözüm gerçekleştirilmiştir.

3. GÜÇ SİSTEM GERİLİM KARARLILIĞI

Bu çalışmada güç sistem kararlılığının ortaya konması bakımından V-Q duyarlılık kriteri kullanılmıştır. Güç sistem kararlılığı ile ilgili genel açıklamalar bu bölümde anlatılmıştır.

Normal işletim şartlarındaki bir elektrik güç sisteminin, bozucu bir etkiye maruz kalmasından sonra sistemdeki bütün bara gerilimlerinin sürekli olarak kabul edilebilir değerlerde kalması gerilim kararlılığı olarak ifade edilebilir. Sistem üzerinde meydana gelebilecek herhangi bir değişiklik, bazen gerilimde kontrol dışı bir düşmeye neden olabilir ki, bu durumda sistem için gerilim kararsızlığı söz konusu olmaktadır [Indulkar, 1989].

Genellikle, güç sisteminde meydana gelen aşırı bir yüklenme gerilim kararlılığı problemlerini ortaya çıkarır. Gerilim çökmesi olarak tarif edilen bu durumu oluşturan başlıca etken, güç iletim sınırlarıdır. Generatör reaktif güç kontrol sınırları, yük karakteristikleri, reaktif güç kompanzasyon cihazlarının karakteristikleri ve yük altında kademe değiştiren transformatörler gibi gerilim kontrol cihazlarının davranışları da bu etkenlere ilave edilebilir.

Gerilim kararlılığı, sistemin yük kararlılığının bir parçasıdır. Bir sistemde işletim gerilimleri, gerilim kararlılığı limitlerine bağlıdır. Düşük gerilimlerde enerji üretmek için, büyük akımlar çekilmektedir. Güç katsayısı birden küçük olan yüklerde oluşacak küçük bir değişim, büyük gerilim düşümlerine neden olabilmektedir ve bunun sonucunda da sistemde gerilim kararsızlığı oluşabilir.

Gerilim kararlılığı açısından yük karakteristikleri oldukça önemlidir. Meydana gelebilecek gerilim değişimlerine göre davranışları bakımından yükler 3 ana gruba ayrılarak incelenebilir [Sekine ve Ohtsuki, 1990].

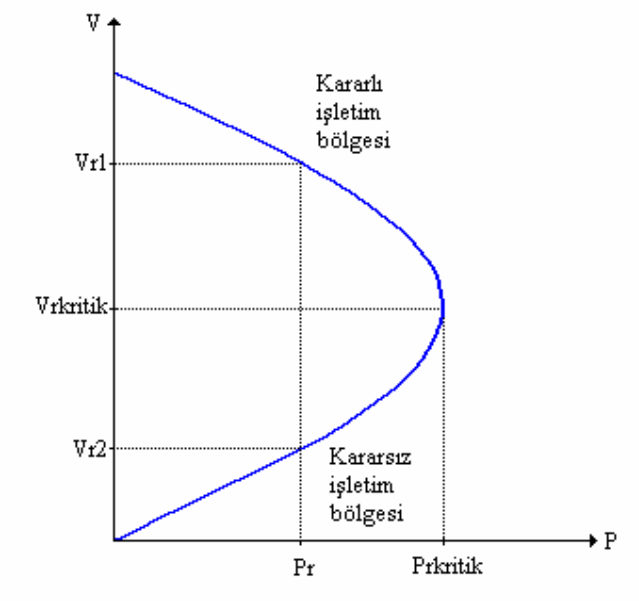
i) Sabit empedans yükleri: Genellikle omik karakterli olan bu yükler, aydınlatma, ısıtma ve ark ocakları olarak kabul edilebilir [Elgerd, 1985].

ii) Sabit akım yükleri: Genellikle elektroliz, metal parlatma gibi uygulamalar, sabit akım tekniğine bağlı yükleri oluşturur [Geçkinli, 1989].

iii) Sabit güç yükleri: Bu gruba asenkron motorlar ve kontrollü empedans yükleri dahil edilebilir. [Elgerd, 1983].

Gerilim kararlılığı açısından iletim hatları büyük önem teşkil etmektedir. Enerji iletim hatları, hattın direnç, endüktans ve kapasitesine bağlı olarak kısa, orta ve uzun iletim hatları olarak

tanımlanmaktadır [Stevenson, 1988]. Orta uzunluktaki iletim hatları ile uzun iletim hatlarının kararlılığı mutlaka incelenmelidir ve özelikle gerilim kararlılığı bakımından uzun iletim hatları büyük öneme sahiptir [Yalçın, 1995].



Şekil 3.1 Radyal iletim hattına ilişkin P-V eğrisi

Gerilim kararlılığı, aktif güç, reaktif güç ve gerilim değerleri arasındaki ilişkilere bağlıdır. Bu ilişkileri gösteren karakteristikler iki baralı bir güç sistemi için analitik olarak elde edilebilirken, geniş ölçekli güç sistemleri için güç akış analizi sonunda belirlenmektedir.

Şekil 3.1'de radyal bir iletim hattına ilişkin P-V eğrisi görülmektedir. Şekilden görüldüğü üzere her bir güç değeri (P_r) için iki adet farklı gerilim (V_{r1} ve V_{r2}) değeri elde edilmektedir. Sadece bir adet tekil nokta vardır ki, bu işletim noktasında bir güç değeri için bir adet gerilim değeri bulunmaktadır. Bu işletim noktası, gerilim kararlılığı için o barada taşınabilecek maksimum güç değerini ve buna karşılık gelen kritik gerilim değerini göstermektedir. Bu işletim noktasındaki güç değeri "kritik güç" (P_{kritik}) ve gerilim değeri de "kritik gerilim" (V_{kritik}) olarak ifade edilmektedir.

P-V eğrisinin üst kısmı kararlı işletim bölgesini, alt kısmı ise kararsız işletim bölgesini ifade etmektedir. Kritik güçten daha büyük bir güç değerine sahip yük, güç sistemine bağlanacak olduğunda, yük gerilimi kararsız hale gelecek ve güç sistemin işletiminin kararsız işletme bölgesine kaymasına neden olacaktır [Taylor, 1992; Kundur, 1994].

Güç sisteminin karmaşık yapısından dolayı kritik güç ve gerilim değerlerinin P-V eğrilerinden

elde edilmesi her zaman kolay bir şekilde sağlanamaz. Bunun için, kritik değerlerin doğrudan ve hızlı bir şekilde elde edilmesi, güç sistemine ait jakobiyen matrisinin ıraksadığı noktalar bulunarak gerçekleştirilebilir.

3.1 V-Q Duyarlılık Analizi

Bir baranın V-Q duyarlılığı, verilen bir çalışma noktası için Q-V eğrisinin eğimini ifade eder. V-Q duyarlılık analizi reaktif güç enjeksiyonlarının etkisini ve bunun gerilim üzerine etkisini aşağıda maddeler halinde verilen ilkeler üzerine inceler [Castellanos vd., 2003]:

- Pozitif V-Q duyarlılık değeri, kararlı çalışmayı göstermektedir.
- Negatif V-Q duyarlılık değeri, kararsız çalışmayı ifade etmektedir.
- Duyarlılık değerinin küçük olması daha kararlı bir sistem demektir. Duyarlılığın sıfır değerinde olması sistem karalılığının mükemmel olduğunu ifade etmektedir.
- Eğer duyarlılık değeri sonsuz değerinde ise, sistem kararlılık limitinin eşik değerinde çalışmaktadır.

Bir enerji sisteminde, bu analizi gerçekleştirmek için, sisteme ait Jakobiyen matrisini kullanmak gereklidir. Arzu edilen baraya reaktif güç enjekte edilir ve riskli durum uygulanır. Bu durumda işletmede olan sistemin Jakobiyen matrisine bakılır. Bu matrisin dört adet alt matristen veya "alt-Jakobiyenler"den oluştuğu görülebilir:

$$\mathrm{J} = \begin{bmatrix} J_{P\theta} & J_{PV} \\ J_{Q\theta} & J_{QV} \end{bmatrix} \tag{3.1}$$

burada,

$J_{P\theta}$= P'nin θ'ya göre kısmi türevlerinin Jakobiyeni,

J_{PV}= P'nin V'ye göre kısmi türevlerinin Jakobiyeni,

$J_{Q\theta}$= Q'nun θ'ya göre kısmi türevlerinin Jakobiyeni,

J_{QV}= Q'nun V'ye göre kısmi türevlerinin Jakobiyenidir.

Bara numaraları i=(2,3,4,...,n) ve k=(2,3,4,...,n) olmak üzere, kısmi türev ifadeleri,

$$\left.\begin{aligned}\frac{\partial \Delta P_i}{\partial \theta_i} &= -\sum_{\substack{j=1 \\ k \neq i}}^{n} Y_{ik} V_i V_k \, sin(\theta_i - \theta_k - \gamma_{ik}) \\ \frac{\partial \Delta P_i}{\partial \theta_k} &= Y_{ik} V_i V_k \, sin(\theta_i - \theta_k - \gamma_{ik}) \qquad (k \neq i)\end{aligned}\right\} \tag{3.2}$$

$$\left.\begin{aligned}\frac{\partial \Delta P_i}{\partial V_i} &= \sum_{\substack{j=1 \\ k \neq i}}^{n} Y_{ik} V_k \, cos(\theta_i - \theta_k - \gamma_{ik}) - 2 V_i Y_{ii} \, cos(\gamma_{ii}) \\ \frac{\partial \Delta P_i}{\partial V_k} &= Y_{ik} V_i \, cos(\theta_i - \theta_k - \gamma_{ik}) \qquad (k \neq i)\end{aligned}\right\} \tag{3.3}$$

$$\left.\begin{aligned}\frac{\partial \Delta Q_i}{\partial \theta_i} &= \sum_{\substack{j=1 \\ k \neq i}}^{n} Y_{ik} V_i V_k \, cos(\theta_i - \theta_k - \gamma_{ik}) \\ \frac{\partial \Delta Q_i}{\partial \theta_k} &= -Y_{ik} V_i V_k \, cos(\theta_i - \theta_k - \gamma_{ik}) \qquad (k \neq i)\end{aligned}\right\} \tag{3.4}$$

$$\left.\begin{aligned}\frac{\partial \Delta Q_i}{\partial V_i} &= \sum_{\substack{j=1 \\ k \neq i}}^{n} Y_{ik} V_k \, sin(\theta_i - \theta_k - \gamma_{ik}) - 2 V_i Y_{ii} \, sin(\gamma_{ii}) \\ \frac{\partial \Delta Q_i}{\partial V_k} &= Y_{ik} V_i \, sin(\theta_i - \theta_k - \gamma_{ik}) \qquad (k \neq i)\end{aligned}\right\} \tag{3.5}$$

Eşitlik 3.1'deki terimler kullanılarak bir duyarlılık değeri hesaplanabilmektedir:

$$J_R = \left(J_{QV} - J_{PV} \cdot J_{P\theta}^{-1} \cdot J_{Q\theta}\right) \tag{3.6}$$

burada sistemin indirgenmiş Jakobiyen matrisidir. J_R^{-1}değeri, duyarlılık değeri olarak ifade edilmektedir [Kundur, 1994]. Bu değer kullanılarak, herhangi bir baradaki gerilimin duyarlılık değerini veya sistem için bir kararlılık değerini karşılaştırmak mümkündür [Lof vd. 1992; Hong vd., 1997].

Tez çalışmasında, V-Q duyarlılık kriteri birim yüklenme problemine dahil edilerek çözüm araştırılmış ve sistemin daha kararlı çalışması yönünde değerlendirme imkanı elde edilmiştir.

4. EVRİMSEL HESAPLAMA TEKNİKLERİ

Evrimsel hesaplama teknikleri arama, öğrenme ve optimizasyon gibi problem çeşitlerine biyolojik popülasyon genetiği yapısına dayalı çözüm üreten hesaplama yöntemlerini içermektedir. Evrimsel hesaplama teknikleri genetik algoritma (GA), evrimsel programlama (EP) ve evrim stratejileri (ES) olarak sayılabilir.

Klasik metotlar, problem çözümü için bir çözüm adayı belirler ve bu adayı değiştirerek daha iyi çözüm adayları elde etmeye çalışır. Evrimsel hesaplama teknikleri ise, problemin çözümü için çözüm adaylarını içeren rasgele bir popülasyon oluşturur. Bu popülasyon nesiller boyunca değişimlere uğrar ve yeni bireyler elde edilir. Bu bireylerin çözüme ne kadar yakın oldukları problemin yapısına bağlıdır. Her çözüm adayı bir değişkenler topluluğunu veya kurallar grubunu temsil edebilir.

Algoritmalar, her bireyin uygunluk değerine bağlı olarak bir sonraki nesile seçilip seçilmeyeceklerini belirler. Yeni popülasyona seçilen bu bireylerden ebeveynler belirleyerek, genetik işlem operatörlerini (yeniden yapılandırma, mutasyon) uygular. Bir sonraki nesil daha güçlü yada daha iyi çözüm üreten bireyleri içerecek şekilde bu işlemler tekrarlanır.

Bu çalışmada, evrimsel hesaplama tekniklerinden genetik algoritma ve evrimsel programlama yöntemleri problem çözümleri için kullanılmıştır. Şimdi bu yöntemleri inceleyelim.

4.1 Genetik Algoritmalar

Çalışmamızda, birim yüklenme probleminin bir parçası olan ekonomik yük dağıtımının çözümünde genetik algoritma kullanılarak en uygun maliyet değerine sahip çözümün elde edilmesi sağlanmıştır. Genetik algoritmanın açıklanması bu bölümde yapılmış ve diğer yöntemlerle kıyaslanması verilmiştir.

Günümüzde teknolojinin gelişmesiyle birlikte mühendislik alanlarında karşılaşılan birçok optimizasyon probleminin çözümünde, problemlerin karmaşıklığı, boyutlarının büyüklüğü (bilgisayarlar kullanılsa da çözüm sürelerinin uzun olması) nedeniyle mevcut klasik optimizasyon yöntemleri ile bu tür problemlerin çözümünde zorluklarla karşılaşılmaktadır. 20.yy.'ın son yarısı içerisinde araştırmacılar doğa içerisindeki karmaşık optimizasyon problemlerinin çözümünü incelemeye başlamışlardır. Bu incelemeler ışığında doğal evrim sürecini baz alan evrimsel algoritmalar geliştirilmiştir. Evrimsel algoritmaların en genel

kullanılanı “Genetik Algoritmaları”dır. Genetik algoritmalar (GA), 1975 yılında Michigan Üniversitesi’nden John Holland ve arkadaşları ile öğrencileri tarafından geliştirilmiştir [Holland, 1975]. Algoritmaya genel olarak baktığımızda, uygun bir çözüm kümesi içerisinden keyfi olarak seçilen bir ilk neslin oluşturulduğunu ve bu toplum ile araştırmaya başlandığını görürüz. İlk nesil içerisindeki her üye, optimizasyon problemi için bir çözüm üreterek birey adını alır. Nesil içerisindeki her birey esasında, genel olarak ikili sayı düzeninden meydana gelen kodlamadır. Her nesil, en seçkin elemanlarını kullanarak bir sonraki nesli oluşturur.

Genel anlamda GA, doğal seçimi temel alan ve doğal genetiğe bağlı birer araştırma algoritmasıdır. Buradaki araştırma işlemi ile ifade edilen yapı, aynen canlıların biyolojik gelişimi olarak alınabilir. Bilindiği üzere canlılar doğal yaşam içerisinde doğarlar ve bir önceki neslin kendilerini meydana getirdikleri gibi kendileri de bir sonraki nesli meydana getirirler. Doğal seçim yapısına göre, nesillerin devamını en uygun bireyler sağlamaktadır.

Genetik algoritma ile bir optimizasyon probleminin çözümünün adımları şu şekildedir [Momoh, 2001]:

i. İlk neslin oluşturulup kodlanması,
ii. Nesil içerisindeki her bireye ait uygunluk fonksiyonu değerlerinin belirlenmesi,
iii. Uygun değerlere sahip bireyler ile yeni nesillerin oluşturulması.

Genetik algoritmaların geliştirilmesinde, doğal seçme ve doğal genetik kurallarından ilham alınmıştır. Doğal seçme kavramı, doğal yaşam koşullarına uyum derecesi en yüksek olan canlı popülasyonunun yaşamına devam etmesi, uyum derecesi düşük olan veya uyum sağlayamamış olan canlı neslinin ise elenmesi görüşünü ifade etmektedir. Bu görüşe göre, bir canlının genleri nesiller boyu süregelirken, aynı zamanda doğal genetik kurallar uygun olarak da farklı genler ile eşleşirler, mutasyona (değişime) uğrarlar ve yeni genleri meydana getirirler.

Genetik algoritma yöntemi de, doğada görülen canlıların genlerini yeni nesillere nasıl ilettiklerini, uğradıkları değişimleri ve yeni bireylerin, popülasyonların oluşturulması olaylarını birleştirerek bir çözüm uzayı içerisine en iyi çözümü arama işlemini yapar. Bir önceki popülasyona ait bireylerin parçalarını, genlerini ve genleri oluşturan bitleri kullanarak yeni popülasyonu meydana getirir. Üremeler sonucu elde edilen bireylerin uygunluk derecelerini inceleyerek, en yüksek uygunluk derecesine sahip bireyleri (ebeveynleri) kullanarak yeni bireyleri (yavruları) oluşturur. Genetik algoritmaların bu işlemleri yerine

getirmedeki rasgeleliğin önemi büyük olmasına karşın olasılık tabanlı arama yöntemleridir.

Michigan Üniversitesi'nden John Holland ve arkadaşları ile öğrencileri tarafından [Holland, 1975] genetik algoritmaların geliştirilmesi sırasında iki amaç hedeflenmiştir. Birincisi, canlıların uyum yapısını anlayarak, basitçe ifade edebilmek; ikincisi, bu modelden hareketle insan yapısına benzer sistemlere uygulanabilecek bir algoritma geliştirmek. GA araştırmalarındaki temel nokta, değişen ortam şartlarına adapte olabilecek bir model oluşturabilmektir. Geniş ölçekli adaptif sistemler bu tür bir yapı ile oluşturulabilirse, yüksek maliyetli yeniden tasarım problemleri optimize edilebileceği gibi, bu maliyetler ortadan da kaldırılabilir. Adaptif bir kontrol ile eldeki mevcut sistemlerin değişen ortam şartları içerisinde tekrar kullanılabilir hale getirilebilmesi, güncellenmesi ile görmüş oldukları görevleri daha verimli yapmaları ve bu görevleri daha uzun süre yerine getirmeleri sağlanabilir.

Günümüzde en sağlam uyumun doğada var olan uyum olduğu rahatlıkla söylenebilir. Bu yapıya sahip olan genetik algoritmalar da ortaya atıldıkları ve geliştirildikleri tarihten itibaren literatürde yerini almış ve etkinliklerini ispatlamıştır. Genetik algoritmalar karmaşık hesaplamalar gerektirmemektedir. Diğer en iyi çözümü arama yöntemlerinde görülen süreklilik, türev vb. gibi matematiksel etkenler bu yöntem için bir zorluk teşkil etmemektedir.

Genetik algoritmalar, evrimsel hesaplama yöntemlerine ait bir metottur. Evrimsel hesaplama tekniği temel anlamda, ebeveynler ve onların eşleştirilip değişime uğratılması ile üretilen yavru bireyleri birer çözüm adayı olarak ele almaktadır. Bu tekniğin Fogel, Ovens ve Walsh tarafından geliştirilmesi sonucunda genetik algoritmalara benzeyen evrimsel programlama (evolutionary programming) metodu oluşturulmuştur. Bu metoda ek olarak, yeni üretilen yavru bireyin belirli bir uygunluk fonksiyonu göre değerlendirilmesi de eklenmiştir [Goldberg, 1989].

Genetik algoritma kavramını ilk kullanan ve çalışmalarını yayınlayan 1967 yılında Bagel olmuştur [Goldberg, 1989]. John Holland 1975 yılında "Doğal ve Yapay Sistemlerde Adaptasyon" isimli kitabını yayınlamıştır [Holland, 1975]. Holland araştırmaları sonucunda, tek bir yapının öğrenme becerisini geliştirmesi yerine, bu yapıların oluşturduğu bir popülasyonun çoğalma, eşleşme, mutasyona uğrama gibi genetik aşamalardan geçerek daha yetenekli, başarılı bireyler oluşturduğunu gözleyerek, doğal koşullara benzer bir yapı oluşturmuştur. Holland'ın oluşturmuş olduğu bu yapı "genetik algoritmalar" olarak literatürde yerini almış ve bilimin birçok alanındaki çalışmalara farklı bir bakış açısı getirmiştir.

4.1.1 Genetik Algoritmaların Geleneksel Yöntemler İle Kıyaslanması

Bilindiği üzere geleneksel çözüm arama yöntemleri üç ana başlık altında toplanabilir. Bunlar, hesaba dayalı, nümerik ve olasılıksal yöntemlerdir. Hesaba dayalı yöntemler kendi aralarında doğrudan ve dolaylı olmak üzere iki gruba ayrılırlar. Doğrudan arama metodunda, optimize edilecek amaç fonksiyonunun maksimum noktasının yeri hakkında bir varsayımda bulunarak, bu noktaya yakın herhangi bir noktanın türevi (gradyanı) alınarak arama sürdürülür. Bu metot, literatürde tepe tırmanma (hill-climbing) olarak ifade edilir. Dolaylı arama metodunda, amaç fonksiyonunun türevi alınarak sıfıra eşitlenmesi ile elde edilen nonlineer denklem takımlarının çözümü yoluna gidilir. Çözümün aranması her yöndeki türevin sıfıra eşit olduğu olası tepe veya çukurların civarında yapılır.

Optimum yerel noktayı bulmak için eğimi en dik olan yönde ilerlemek gerekir. Bu yöntemlerin en büyük dezavantajı, yerel bakış açısı ile hareket etmeleridir. Aranılan en iyi çözüm noktası, başlangıç noktasının yakınlarındadır. Amaç fonksiyonun her ne kadar bir adet minimum noktası bulunsa da, birçok yerel minimumu bulunabilir. Eğer yerel bir minimum etrafında arama yapılmışsa çözüm yerel minimuma takılabilir, aslında arzu edilen minimum bu olamayacaktır. Tek tepe veya çukur noktasına sahip fonksiyonlar için hesaba dayalı arama metotları için çözüm kolay olurken, birden çok minimum veya maksimuma sahip fonksiyonlar da arama yapılırken zorluklar ile karşılaşılabilecektir. Bunu ortadan kaldırmak için de rasgele başka bir başlangıç noktası seçip tekrar arama yapmak gibi farklı iyileştirmelere gitmek bir çözüm olabilir.

Hesaba dayalı metotlarda istenmeyen bir diğer durum ise, optimize edilecek amaç fonksiyonlarının türevlerinin mevcut olması zorunluluğudur. Gerçek hayatta karşılaşılan problemlere ait amaç fonksiyonlarının süreksizlikleri ve daha karmaşık olmaları yüzünden bu metotların bütün problemler için elverişli olmaları mümkün değildir.

Nümerik arama yöntemlerini ele aldığımızda amaç, sınırlı bir çözüm uzayı kümesindeki noktalar için optimize edilecek fonksiyona ait değerlerin elde edilip, buna göre yeni noktaların bulunmasıdır. Bu yöntemin dezavantajı ise, uygulamada çözüm uzayının çok geniş olmasından dolayı arama uzayındaki her noktanın uygunluk değerine bakılmasının yöntemin etkinliğini kısıtlamasıdır.

Rasgele veya olasılıksal arama yöntemleri, diğer arama yöntemlerinin bahsettiğimiz eksikliklerinden dolayı günümüzde giderek artan bir ilgiyle araştırılmaya ve kullanılmaya

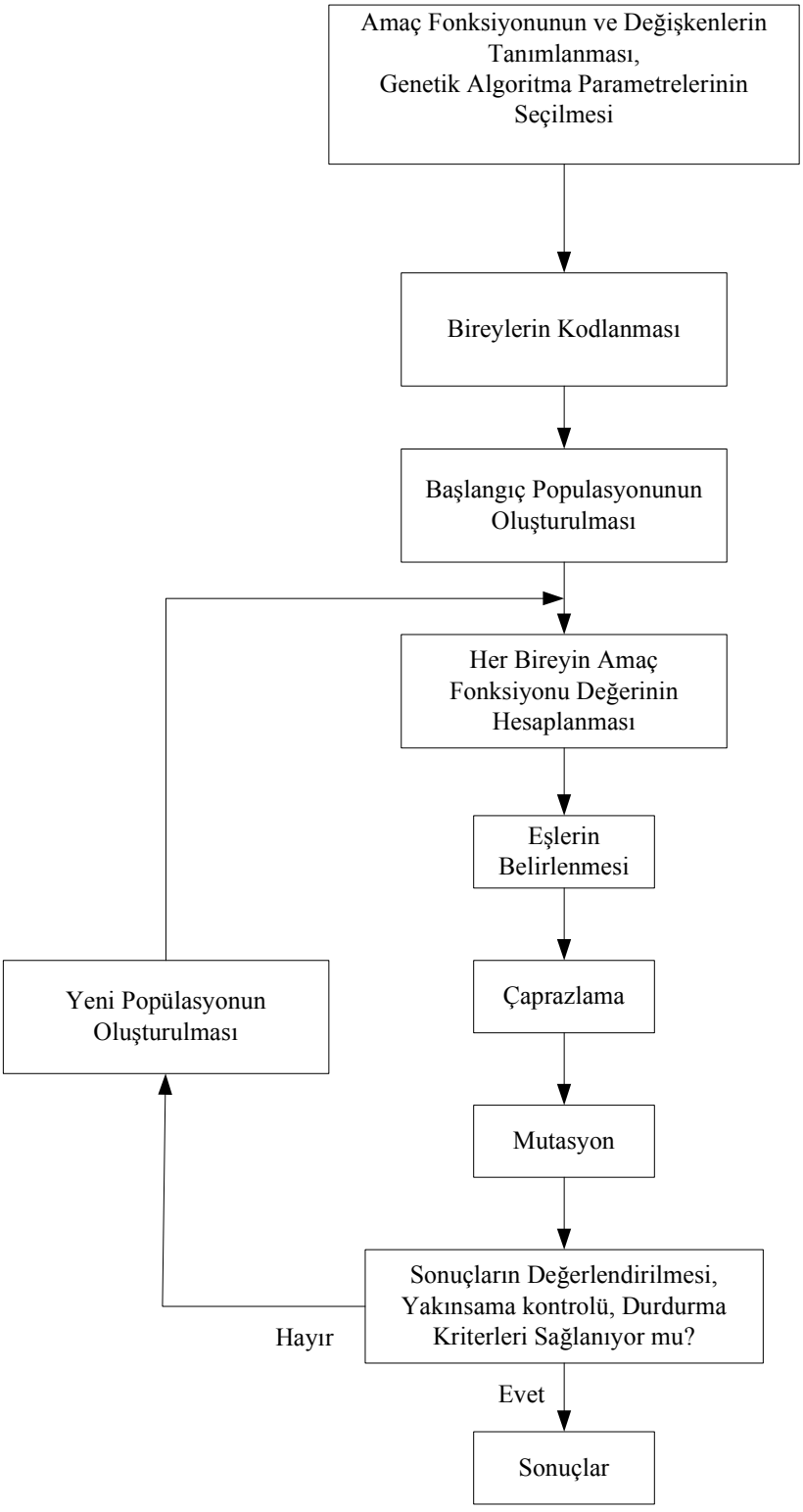
başlanmıştır [Özdeş, 2003].

Klasik optimizasyon yöntemleri ile genetik algoritmayı kıyaslarsak, belli başlı üstünlükleri ve farklılıkları da şunlardır [Haupt, 2004]:

i. Genetik algoritmalar kodlanmış parametreler ile işlem yapmaktadırlar. Bundan dolayı optimize edilecek fonksiyon süreksizliğe sahip olsa da arama işlemi yapılırken kodlanmış parametre kullanılacağından herhangi bir sorun yaşanmaz.
ii. Genetik algoritmanın problemin en uygun çözümü bulmasındaki en önemli nokta, tek bir nokta üzerinden değil de, noktalar kümesi ile arama işlemi yapmasından kaynaklanmaktadır.
iii. Genetik algoritmalar optimize edilecek amaç fonksiyonunu temel aldıklarından dolayı, türev ve benzeri matematiksel etkenler sonucu karşılaşılabilecek süreksiz veya nonlineer fonksiyonlar üzerinde de işlem yapabilirler.
iv. Genetik algoritmalar daha esnek işlem görmelerini sağlayacak stokastik geçiş elemanları kullanmaktadır.

Bunun yanında her yöntemde karşılaşıldığı üzere GA'nın dezavantajları da bulunmaktadır. Bunlar şöyle sıralanabilir:

i. Arzu edilmeyen çözüme ulaşabilirler.
ii. İşlem süreleri uzun olabilir.
iii. Algoritmanın uygunluğu için, parametrelerin belirlenmesinde yapılacak deneme sayısı fazladır ve deneyim de önemlidir.



Şekil 4.1 Temel Genetik Algoritma Döngüsü [Haupt, 2004]

4.1.2 Genetik Algoritmanın İşleyişi

Genetik algoritmanın işleyişi sırasındaki temel adımlar Şekil 4.1.'de verilmiştir. Yöntem, optimizasyon işlemine bir başlangıç neslinin oluşturulması ile başlar, belirli bir dizi rastlantısal süreç izleyerek çözüme ulaşmaya çalışır.

GA'da yönteminde bir problemde çözüm arama yapmak için kullanılan 4 adet ana operatör kullanılmaktadır. Bunlar [Haupt, 2004]:

- Bireylerin kodlanması,
- Popülasyondaki bireylerin seçiminin yapılması,
- Çaprazlama,
- Mutasyon.

Şimdi bu operatörleri inceleyelim.

4.1.3 Başlangıç Neslinin Oluşturulması

Bütün canlı organizmalar hücre adı verilen temel taşlardan meydana gelmektedir. Hücreleri ise organizmaya ait bütün bilgileri depolayan ve DNA dizilerinden oluşan kromozomlar oluşturmaktadır. Kromozomlar ise gen adı verilen yapılar ile kodlanmışlardır. Kromozom içindeki yer alacak her bir genin yeri bellidir. Genetik yapıyı oluşturan bütün kromozomların hepsine genome adı verilir [Haupt, 2004].

Optimize edilecek amaç fonksiyonunun çözüm kümesinde yapılacak bir arama işleme için, birden fazla kromozomdan oluşan bir topluluk kullanılmaktadır. Bu topluluğa literatürde, "nesil", "popülasyon", "toplum" gibi isimler verilmektedir. Genetik algoritma kullanılarak yapılan optimizasyon işleminin ilk adımı belirli sayıda bireyin dahil olduğu bir başlangıç neslini oluşturmaktır. Nesli (popülasyonu) oluşturan bireylerin sayısı (N_{pop}), GA için önemli bir kontrol parametresidir. Bir popülasyon içerisindeki birey sayısı veya bir diğer anlamda kromozom sayısı (N_{pop}), sabit tutulmaktadır. Genetik algoritmada, popülasyon içerisindeki birey sayısı ile ilgili herhangi bir kısıt veya genel bir kural bulunmamaktadır. Bu sayı yapılan denemeler veya daha önceki deneyimler sayesinde probleme uygun olarak belirlenmektedir. N_{pop} değerinin büyük olarak belirlenmesi geniş ölçekte bir uzay kümesinde arama yapılması avantajını verecektir, fakat bu değerin çok büyük olması da çözüm süresinin uzamasına neden olacaktır [Haupt, 2004].

Başlangıç neslini oluşturan bireylerin sayısı N_{pop}, denemeler sonucunda elde edilecektir. Popülasyonu oluşturacak bireyler hakkında herhangi bir bilgi yoksa, amaç fonksiyonunun kısıtlarını da göz önünde bulundurarak bireylerin keyfi olarak seçilmesi en kolay yoldur. Bununla birlikte, daha önceki deneyimlerden elde edilen bilgiler ışığında bireylerin alabileceği değerlerin seçiminde belirli tahminler yapılabiliyorsa, bu da çözüm süresini kısaltacaktır.

4.1.4 Bireylerin Kodlanması

Optimize edilecek amaç fonksiyonunun çözüm kümesini oluşturacak her bireyin uygun bir şekilde kodlanması gerekmektedir. Mühendislik alanında ilk genetik uygulamaları için bireylerin kodlanmasında binary sayı tabanı temel alınmıştır. Daha sonraları bu ikili kodlamaya ek olarak reel sayı ve tam sayı kodlamaları da geliştirilip kullanılmaya başlanmıştır.

Popülasyonu oluşturan her birey binary (ikili) tabanında 1 ve 0'ların birleşiminden oluşan kodlar şeklindedir ve "kromozom" adını alır. Kromozomlar ise "gen" adı verilen değişken kodların birleşiminden oluşmaktadır. Popülasyonu bir matris olarak ifade edersek;

$$N_{pop} = \begin{bmatrix} birey_1 \\ birey_2 \\ .... \\ birey_n \end{bmatrix} = \begin{bmatrix} gen_{11} & gen_{12} & .... & gen_{1m} \\ gen_{21} & gen_{22} & .... & gen_{2m} \\ .... & .... & .... & .... \\ gen_{n1} & gen_{n2} & .... & gen_{nm} \end{bmatrix} \tag{4.1}$$

burada matrisin satırları bireyleri (kromozomları), sütunlar ise bireyleri meydana getiren genleri gösterecektir. Burada bireyler, optimize edilecek amaç fonksiyonunun çözüm kümesini oluştururken, bireyleri meydana getiren genler ise amaç fonksiyonunun sahip olduğu değişkenleri, kısıtları ve probleme ait diğer bilgileri ifade edecektir.

Genetik algoritmada, bireylerin kodlanması genel anlamda ikili sayı tabanında (binary olarak) yapılmaktadır. J. Holland tarafından yapılan kodlama işlemi genelde ikili tabanda bit dizileri şeklinde olmuştur. İkili sayı tabanı ile yapılan bu yapı, basit olması, bilgisayarlar tarafından daha kolay ve hızlı bir şekilde işlenebildiğinden tercih edilmektedir. Bu yapı ile çok farklı bireyler oluşturulabilir fakat uygulamada ortaya çıkan birçok problem için bu ikili kodlama

sistemi yeterli olamayabilir. Tamsayıların veya ondalık sayıların binary olarak gösterimi gerektiğinde bazı düzeltmelere ihtiyaç duyulmaktadır. Çaprazlama ve mutasyon sonrası bazı düzeltmeler yapmak gerekebilir.

İkili sayı tabanını baz alan kodlama şeklinden farklı olarak en çok kullanılan ikinci bir yapı ise değer tabanlı yapılan kodlama şeklidir. Değer tabanlı yapılan bir kodlama sisteminde bireyleri oluşturan genler uygulanacak probleme uygun olarak tamsayı, ondalık sayı, kesirli sayı veya harf şeklinde olabilir. Bu yapının dezavantajı ise bireye ait genlerin alacağı değer sayısının bilgisayar belleğinde kaplayacağı alanın büyük olması bunun neticesinde de algoritmanın yavaş işlemesi olacaktır.

4.1.5 Bireylerin Seçimi

Bu işlem, mevcut popülasyondan yeni nesildeki yavruları oluşturacak ebeveyn bireylerinin seçilmesi olarak ifade edilebilir. Popülasyondaki her birey amaç fonksiyonuna göre belirlenen uygunluk değerlerine göre bir sıralamaya tabi tutulur. Bireylerin bu uygunluk değerlerine göre belirli seçim yöntemleri uygulanır. Bu yöntemlerin belli başlıları şunlardır:

- Rulet Seçim,
- Turnuva Seçimi,
- Sıralı Seçim,
- Boltzman Seçimi.

Çaprazlama veya mutasyona uğrayacak bireylerin mevcut popülasyon içerisinden seçilmesi için farklı yöntemler kullanılmaktadır. Seçim operatörü adı verilen bu yöntemler genetik algoritmanın işlem hızını etkileyeceği gibi algoritmanın en uygun çözümü bulması için gereken zamanı da etkileyecektir. Genetik algoritmanın üreme, çaprazlama ve mutasyon süreçleri sonrasında, mevcut popülasyon içerisindeki en iyi uygunluk değerine sahip bireylerin korunması için bu bireylerin yeni nesile doğrudan aktarılması gerekmektedir. Bu işleme genetik algoritmada elitizm ve işleme katılan bireylere de elit bireyler adı verilmektedir. Elit bireylerin sayısı da popülasyona göre uygun bir değerde olmalıdır. Elit birey sayısı seçim oranı (X_{rate}) ile belirlenir. Bir sonraki nesil için saklanılacak en iyi değerlere sahip elit birey sayısı (N_{keep}), Eşitlik (4.2) ile bulunur:

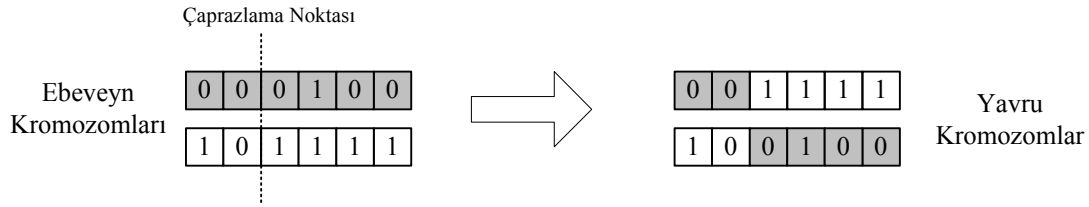
$$N_{keep} = X_{rate} \cdot N_{pop} \tag{4.2}$$

Popülasyon içerisindeki her bireyin çözüme olan uygunluk değerinin belirlenmesi için kullanılan fonksiyon da uygunluk fonksiyonu (fitness function) adını almaktadır. Seçim sürecinde bireylerin belirlenmesinde bu uygunluk değerinden yararlanılmaktadır [Haupt, 2004].

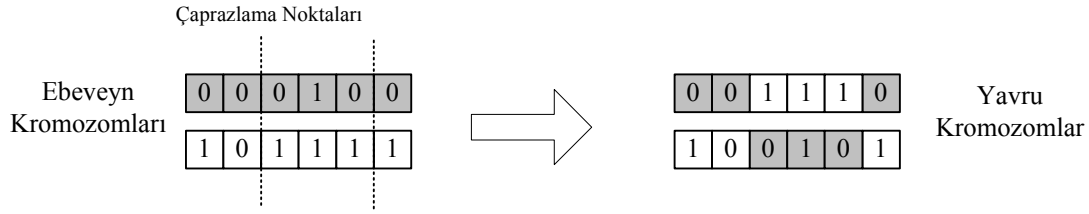
Bireylerin seçiminde, düzenli seçim uzayı ve geliştirilmiş seçim uzayı olmak üzere iki tip seçim uzayı kullanılır. Düzenli seçim uzayında, popülasyon bir önceki nesildeki tüm yavru ve ebeveynleri içermemektedir. Yeni nesili oluşturan bireyler (yavrular), bir önceki nesil bireyleri (ebeveynler) ile yer değiştirir. Popülasyon sayısı N_{pop} sabit tutulur. Bu yöntemin arzu edilmeyen yönü yüksek uygunluk değerine sahip olan bir ebeveynin yavrusu ile yer değiştirerek kaybedilebilmesidir [Haupt, 2004]. Geliştirilmiş seçim uzayında ise, tüm ebeveynler ve yavrular bir sonraki popülasyona aktarılırlar.

4.1.6 Çaprazlama

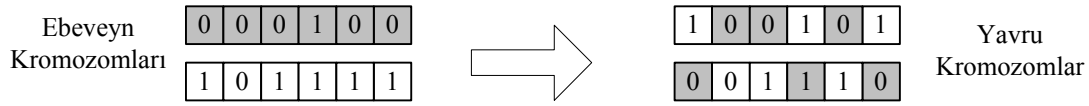
Genetik algoritmanın en önemli bir operatörü olan çaprazlama, doğal yaşamdaki canlıların birbirleri arasındaki döllenme olarak ifade edilebilir. Aynı popülasyona ait bireyler arasındaki bilgi alışverişini sağlayarak, ebeveyn bireylerden alınan parçalar ile yeni yavru bireylerin gen yapısı oluşturulur. Bireylerin kodlanma şekline göre çaprazlama yöntemi de değişecektir. İki bireyin bir araya getirilerek sahip oldukları genetik bilgilerin alış-verişinin sağlanmasına "çaprazlama" adı verilmektedir. Çaprazlama işlemini gerçekleştirmek için bireylerin seçilmesi olayına da "eşleştirme (mating)" adı verilir. Çaprazlama için eşleştirilen bireyler "ebeveyn", bu işlem sonucunda üretilen bireyler ise "yavru" ismini almaktadır. Çaprazlama operatörü, genetik algoritmanın bel kemiğini oluşturmaktadır. Eğer çaprazlama oranı %100 olarak seçilirse, popülasyon içerisindeki tüm bireylerin çaprazlanması sağlanarak yeni nesil oluşturulur. Bu operatör sayesinde, çözüm uzayı içerisindeki farklı çözüm noktalarına ulaşılması sağlanır. Genetik algoritmadaki kodlama şekilleri farklı olduğundan, çaprazlama yöntemlerinde de farklılıklar bulunmaktadır.



A) Tek Noktalı Çaprazlama



B) İki Noktalı Çaprazlama



C) Düzgün (uniform) Çaprazlama

Şekil 4.2 Çaprazlama Metotları

Şekil 4.2'den görüldüğü üzere, binary tabanlı kromozomlar için 3 farklı çaprazlama tekniği bulunmaktadır. Bunlardan tek noktalı çaprazlama olarak ifade edilen birinci çaprazlama yöntemi, m adet "bit"e sahip bir birey için, $1< c <$m-1 aralığında bir c tam sayısı seçilerek, bu değer çaprazlama noktası olarak belirlenmesidir. Bir diğer çaprazlama yöntemi ise, bireylerin belirlenen iki noktadan çaprazlamaya uğrayarak, gen bilgilerini paylaşmalarıdır. Bu yöntem de iki noktalı çaprazlama olarak ifade edilir. Son yöntem, bireylerin rasgele olarak belirlenen noktalardan çaprazlanmasıdır ki bu da düzgün (uniform) çaprazlama adını alır.

Eğer bireylerin kodlanması sürekli tamsayılar şeklinde yapılmışsa bu durumda çaprazlama işlemi şu şekilde olacaktır;

Eşleşecek ebeveyn bireyleri,

$$birey_a = [gen_{a1}, gen_{a2}, ..., gen_{ac}, ..., gen_{an}]$$
$$birey_b = [gen_{b1}, gen_{b2}, ..., gen_{bc}, ..., gen_{bn}] \quad (4.3)$$

burada *a* indisi anne ve *b* indisi baba bireyini temsil etmektedir. *c* indisi ile çaprazlama noktası gösterilmektedir. Bu çaprazlama sonucunda yavrularda gözükecek yeni gen değişkeni;

$$gen_{yeni1} = gen_{ac} - \beta(gen_{ac} - gen_{bc})$$
$$gen_{yeni2} = gen_{bc} + \beta(gen_{ac} - gen_{bc}) \quad (4.4)$$

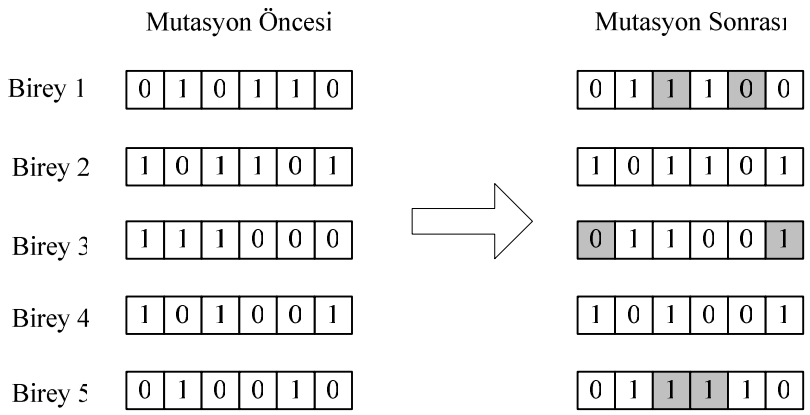
ile hesaplanır. Burada β, 0 ile 1 arasında rasgele bir değer olarak seçilir. Bu işlem sonucunda, Eşitlik (4.3)'teki anne ve baba bireylerinin çaprazlaması yapılıp Eşitlik (4.4)'te hesaplanan yeni gen değeri de eklendiğinde yavru bireylerinin kromozom yapısı aşağıdaki şekilde olacaktır [Haupt, 2004].

$$yavru_1 = [gen_{a1}, gen_{a2}, ..., gen_{yeni1}, ..., gen_{bn}]$$
$$yavru_2 = [gen_{b1}, gen_{b2}, ..., gen_{yeni2}, ..., gen_{an}] \quad (4.5)$$

4.1.7 Mutasyon

Genetik algoritmanın bir diğer operatörü de "mutasyon"dur. Mutasyon, bir bireyin genlerinde herhangi bir etkene bağlı olmaksızın keyfi olarak meydana gelen değişim olarak ifade edilir. Mutasyon işlemi rasgele şekilde yapılan bir süreç olup çözüm uzayında bulunup da popülasyonda yer almayan yeni bireyleri de popülasyona dahil eden bir operatördür. İlk olarak mutasyon oranına bağlı olarak popülasyon içerisinde yapılacak mutasyon sayısı elde edilir. Burada mutasyon oranı (μ), popülasyonda yapılacak mutasyon sayısını belirlemektedir. Yapılacak mutasyon sayesinde, çözüm uzayında daha fazla çözüm noktasında arama yapılabilecektir [Haupt, 2004].

Daha sonra bu mutasyonların hangi bireylere ve bu bireylerin hangi elemanlarına yapılacağı belirlenir. Bireylerin kodlama şekillerine göre farklı mutasyon yöntemleri mevcuttur. Temel anlamda mutasyon işlemi ikili kodlama sistemi için bireylerin 1 olan bit değerlerini "0", 0 olan bit değerlerini ise "1" haline getirmektir. Şekil 4.3'te ikili tabanda bir mutasyon örneği görülmektedir. 1, 3 ve 5 nolu bireyler mutasyona uğramışlardır.



Şekil 4.3 Mutasyon Örneği

Mutasyon işlemi de genetik algoritma için çok önemlidir. Eğer popülasyon içerisinde yapılacak olan mutasyon oranı %0 olarak belirlenirse; çözüm, amaç fonksiyonunun sahip olduğu yerel minimumlarda takılabilir. Eğer mutasyon sayısı çok küçük seçilirse, bu durumda çözüm uzayı içerisinde bulunan optimizasyon problemi için en uygun olabilecek bireylere ulaşılamadan arama yapılmış olabilir. Diğer taraftan çok büyük bir mutasyon oranı seçilmiş ise, yeni oluşturulan popülasyonun bir önceki popülasyona olan benzerliklerinden büyük oranda sapma olasılığı çıkabilir ki yeni nesillere bir önceki nesilden aktarılması gereken genetik bilginin kaybolmasına veya minimuma inmesine neden olacağından, bu durumda elde edilmiş mevcut en uygun bireylerin de kaybolmasına neden olabilir. Bu da çözüm süresinin çok uzamasına neden olacaktır.

4.1.8 Algoritmanın Durdurma Kriterinin Belirlenmesi

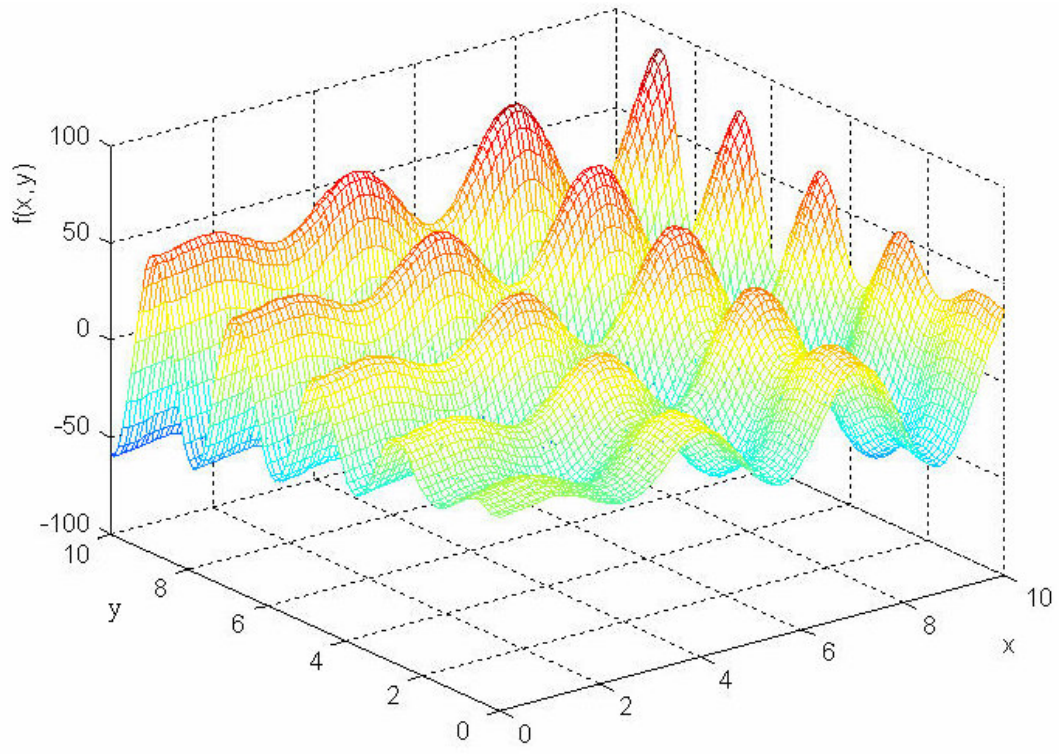
Genetik algoritma yönteminde iterasyon sayısı, amaç fonksiyonunun optimum çözümünü kabul edilebilir bir değerde elde edecek şekilde belirlenmelidir. Genetik algoritmayı sonlandırmada kullanılabilecek en basit yol, popülasyon sayısının belli bir değer seçilmesi ile sağlanabilir. Daha farklı durdurma kriterleri de probleme uygun olarak kullanılabilir. Populasyondaki en iyi birey belirli bir iterasyon sayısı boyunca değişmiyorsa, bu da bir durdurma kriteri olarak belirlenebilir. Yine popülasyonun ortalama uygunluk değeri ile en iyi uygunluk değerinin kıyaslanması sonucu da bir diğer durdurma kriteri olarak kabul edilebilir.

4.1.9 Basit Bir Genetik Algoritmasının Örnek Uygulaması

Bu bölümde basit bir örnek üzerinde genetik algoritma operatörlerini görelim. Bunun için $f(x,y)= 4\cdot x\cdot\sin(2\cdot x)+6\cdot y\cdot\sin(3\cdot y)$ ifadesi ile verilen amaç fonksiyonumuzun minimum değerini bulalım. Fonksiyonumuzun değişkenlerinin $0 \le x \le 10$ ve $0 \le y \le 10$ aralıklarında değerler aldığını kabul edelim. Böylece fonksiyonumuz için belli kısıtlar da oluşturmuş oluruz. Şekil 4.4'te de görüldüğü üzere fonksiyonumuzun bir adet genel fakat birçok da yerel minimum noktası bulunmaktadır. Biz genetik algoritmayı kullanarak bu fonksiyonun genel minimum noktasını bulmak istiyoruz.

Fonksiyondaki değişken sayımız N_{var} = 2'dir. Bundan dolayı genetik algoritmamızda da her bir kromozomumuz iki adet genden oluşacaktır.

birey (kromozom) = [x , y]



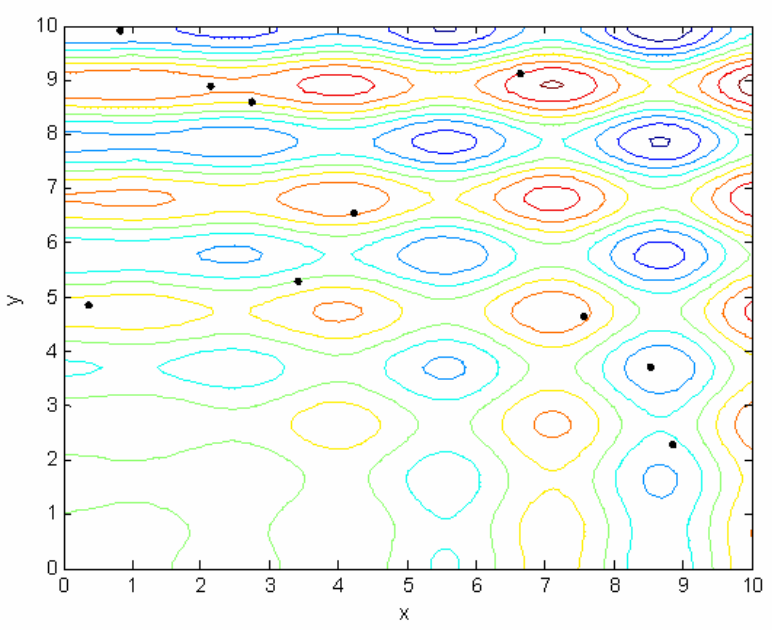
Şekil 4.4 Örnek olarak verilen amaç fonksiyonunun grafiği

İlk olarak başlangıç popülasyonumuzu oluşturmamız gerekir. Fonksiyon kısıtlarından göründüğü üzere rasgele belirleyeceğimiz bireyler 0 ile 10 arasında değerler alacaktır. Popülasyonumuzun sayısını da N_{pop}=10 birey olarak alalım. Tablo 4.1'de ilk popülasyon

değerleri ve bu değerlerin fonksiyona verdikleri sonuçlar bulunmaktadır. Şekil 4.5'de fonksiyonun eş yükselti eğrisi görülmektedir. Başlangıç popülasyonuna ait bireyler de bu eğri üzerinde gösterilmiştir. Eğri üzerinden de görüldüğü üzere başlangıç toplumuna ait bireyler geniş bir alana yayıldıkları gibi hiç biri de genel minimum içerisinde değiller.

Tablo 4.1 Birey sayısı 10 olan başlangıç popülasyonu ve ilişkili fonksiyon değerleri

x	y	f(x, y)
7,5402	4,6615	45,3530
6,6316	9,1376	58,6259
8,8349	2,2858	-25,2228
2,7216	8,6204	26,3319
4,1943	6,5662	44,0081
2,1299	8,9118	45,7815
0,3560	4,8814	26,5326
0,8116	9,9265	-56,1879
8,5057	3,7333	-54,7475
3,4020	5,3138	-0,6016



Şekil 4.5 Eşyükselti eğrisi üzerindeki başlangıç popülasyonuna ait bireylerin gösterimi.

Şimdi başlangıç popülasyonundan hangi bireylerin yeni nesile aktarılacağını ve yavruları üreteceğini kararlaştırmamız gerekmektedir. Bunun için en iyi fonksiyon değerini uygun olarak minimum değerden maksimum değere doğru bireyler sıralanır. Hayatta kalacak elit birey sayısı (N_{keep}) belirlenir. En iyi fonksiyon değerlerine göre sıralanmış popülasyon içerisindeki N_{pop} adet bireyden en üstteki N_{keep} adet birey eşleşme için saklanır ve diğer bireyler yeni popülasyonda yer alacak yavrulara yer açmak için yok edilir. Bu işlem genetik algoritmanın her iterasyonu için tekrarlanır. Bu örnek için yeni nesil için saklanacak birey sayısı N_{keep}= 4 (seçim oranı X_{keep}= %40) olarak belirlenmiştir.

Tablo 4.2 Seçim oranı %40 seçildiğinde hayatta kalan bireyler ve fonksiyon değerleri

Birey	x	y	f(x, y)
1	0,8116	9,9265	-56,1879
2	8,5057	3,7333	-54,7475
3	8,8349	2,2858	-25,2228
4	3,4020	5,3138	-0,6016

Örneğimiz için 10 bireylik popülasyonun ortalama fonksiyon değeri 10,9873 olup en iyi fonksiyon değeri −56,1879'dır. Tablo 4.2'den görüldüğü üzere popülasyondan kötü uygunluk değerine sahip bireyler iptal edildiğinde ortalama değer -34,1899 değerini alır ki bu da doğal seçimin bir sonucudur.

Tablo 4.3 Seçim oranı %40 seçilmesi durumu için rulet çarkı

Birey	P_n	$\sum_{i=1}^{n} P_n$
1	0,4	0,4
2	0,3	0,7
3	0,2	0,9
4	0,1	1,0

Bilindiği üzere, popülasyonun birey sayısı sabit kalmalıdır. Bunun için N_{keep}= 4 adet en uygun birey kullanılarak yeni yavrular oluşturulacaktır. Bunun için 4 bireylik eşleşme havuzundan 3

adet anne (A) ve baba (B) çifti rasgele şekilde seçilir. Her çift aile özelliklerini içeren iki adet yavru bireyi üretir. Bu örnekte eşlerin seçimi Tablo 4.3'deki değerlerden hareketle rulet çarkına göre belirlenmiştir.

Anne (A) ve Baba (B) ebeveyn bireyleri için, belirlenen rasgele sayılar sırasıyla;

(0,9501 , 0,2311 , 0,6068) ve (0,4860 , 0,8913 , 0,7621)

olarak belirlenmiştir. Bu değerler Tablo 4.3'ün 3.sütununa karşılık gelen değerler ile düzenlendiğinde çaprazlama için seçilecek ebeveynler belirlenmiş olacaktır.

A = [4 1 2] , B = [2 3 3]

A ve B vektörleri çaprazlama yapılacak bireylerin indislerini içermektedir. Böylece eşleşecek çiftler (4,2), (1,3) ve (2,3) olarak gösterilebilir. Çaprazlama noktası olarak c=1 olarak seçilmiştir. β değeri rasgele olarak (<1) belirlenmektedir. Bu durumda çaprazlama sonucunda elde edilecek yeni yavrular şu şekilde bulunacaktır;

Birinci eşleşme,

$Birey_4$= (3,4020 , 5,3138)

$Birey_2$= (8,5057 , 3,7333)

β = 0,4565

$Yavru_1$= [3,4020 − 0,4565 × (3,4020 − 8,5057) , 3,7333] = **[5,7318 , 3,7333]**

$Yavru_2$= [8,5057 + 0,4565 × (3,4020 − 8,5057) , 5,3138] = **[6,1759 , 5,3138]**

İkinci eşleşme,

$Birey_1$= (0,8116 , 9,9265)

$Birey_3$= (8.8349 , 2.2858)

β = 0,0185

$Yavru_3$= [0,8116 − 0,0185 × (0,8116 − 8,8349) , 2,2858] = **[0,6789 , 2,2858]**

$Yavru_4$= [8,8349 + 0,0185 × (0,8116 − 8,8349) , 9,9265] = **[8,6865 , 9,9265]**

Üçüncü eşleşme,

$Birey_2$= (8,5057 , 3,7333)

$Birey_3$= (8,8349 , 2,2858)

β = 0,8214

$Yavru_5$= [8,5057 − 0,8214× (8,5057 − 8,8349) , 2,2858] = **[8,7761 , 2,2858]**

$Yavru_6$= [8,8349 + 0,8214× (8,5057 − 8,8349) , 3,7333] = **[8,5645 , 3,7333]**

Böylece eşleşme sonucu üretilen 6 adet yavru birey ile birlikte popülasyon sayımız (N_{pop}) yine 10'a yükseldi. Şimdi oluşturduğumuz bu popülasyona mutasyon operatörünü uygulayalım. Bu örnek için mutasyon oranı (μ) %20 olsun. Bu durum için uygulanacak mutasyon sayısı ise 0,20×9×2 ≅ 4 olacaktır. Şimdi hangi bireylerin ve bu bireylerin hangi geninin mutasyona tabi tutacağımızı rasgele olarak tespit edelim:

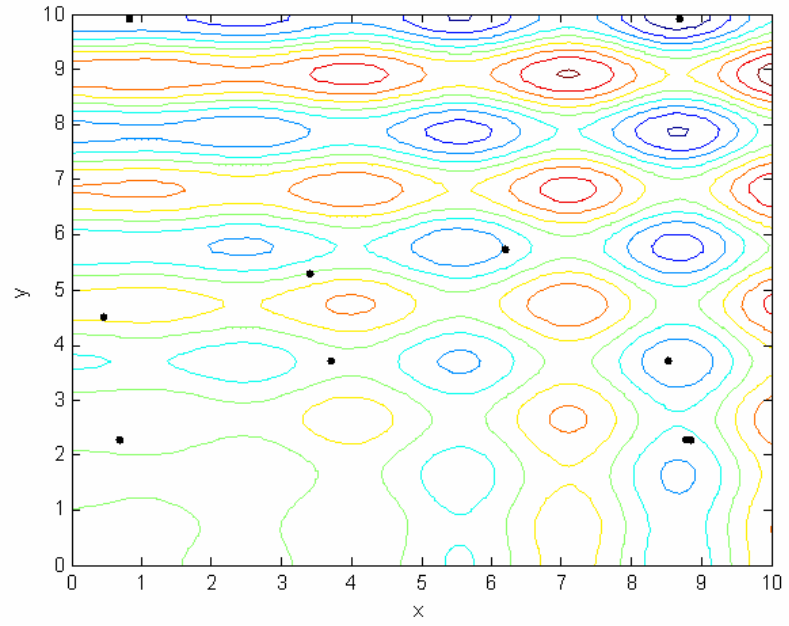
Bireyler: [5 6 10 10]

Genler : [1 2 2 1]

Mutasyon ise seçilen bireyin mutasyona uğrayacak geninin bir ile on arasındaki rasgele bir sayı ile değiştirilmesi olacaktır. Tablo 4.4'de çaprazlama yapıldıktan sonra oluşturulan popülasyon ile mutasyon sonucu elde edilen birinci nesili ve buna bağlı fonksiyon değerleri görülmektedir. Mutasyona uğramış genler bold-italik olarak gösterilmiştir.

Tablo 4.4 Popülasyonun Mutasyon Sonraki Değerleri

Birey	Çaprazlamadan Sonraki Popülasyon		Mutasyondan Sonraki Popülasyon		
	x	*y*	*x*	*y*	*f(x,y)*
1	0,8116	9,9265	0,8116	9,9265	-56,1888
2	8,5057	3,7333	8,5057	3,7333	-54,7479
3	8,8349	2,2858	8,8349	2,2858	-25,2224
4	3,4020	5,3138	3,4020	5,3138	-0,6040
5	5,7318	3,7333	***3,7048***	3,7333	-8,5539
6	6,1759	5,3138	6,1759	***5,7515***	-39,7589
7	0,6789	2,2858	0,6789	2,2858	10,1038
8	8,6865	9,9265	8,6865	9,9265	-94,0226
9	8,7761	2,2858	8,7761	2,2858	-26,3506
10	8,5645	3,7333	***0,4390***	***4,5142***	23,7883



Şekil 4.6 Birinci popülasyona ait bireylerin eşyükselti eğrisi üzerinde gösterilimi

Popülasyon için ortalama değer -27,1557 olarak hesaplanmıştır. 8 nolu birey en iyi değere sahiptir. Birinci nesile ait bireylerin dağılımı Şekil 4.6'da gösterilmiştir. Bu işlemlere kabul edilebilir bir çözüm bulunana kadar tekrar edilir.

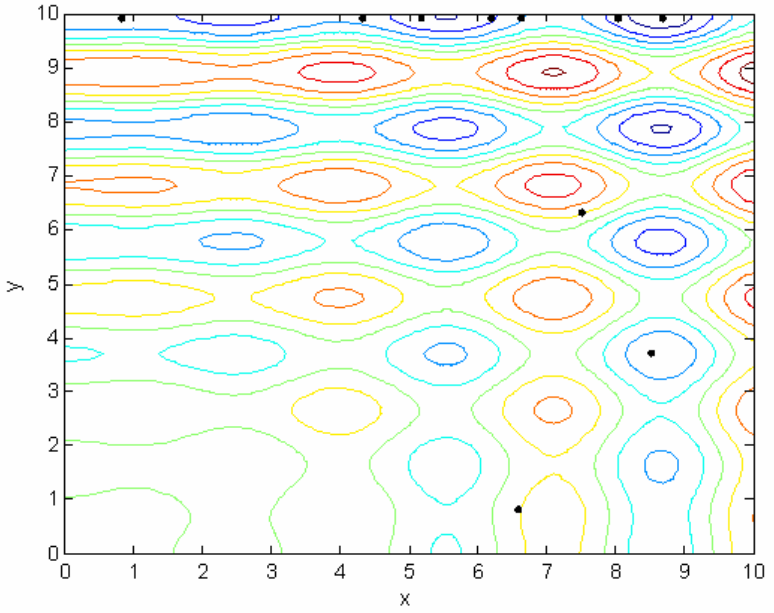
Tablo 4.5 İkinci nesile ait bireylerin değerleri ve ilişkili fonksiyon değerleri

x	*y*	f(*x*, *y*)
8,6865	9,9265	-94,0226
5,1809	9,9265	-76,1298
0,8116	9,9265	-56,1888
8,5057	3,7333	-54,7479
6,4798	9,9265	-49,4992
4,3017	9,9265	-46,8339
6,1759	5,7515	-39,7589
8,3826	7,3491	-31,6797
4,8250	8,1876	-30,8174
4,4923	8,9032	61,0749

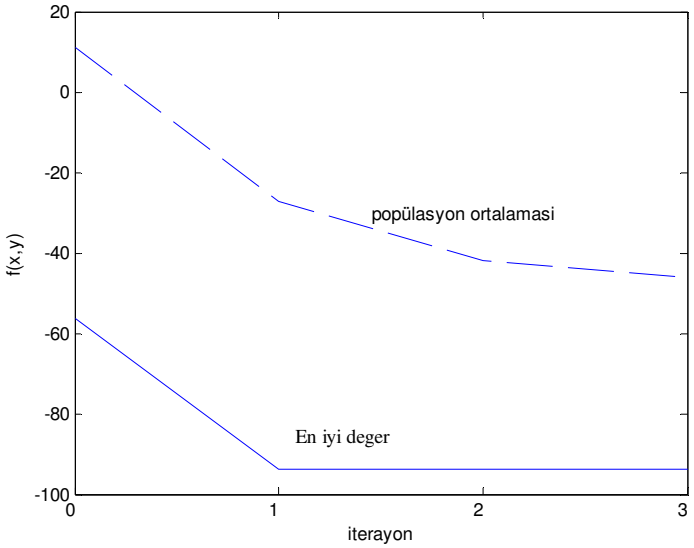
Tablo 4.6 Üçüncü nesile ait bireylerin değerleri ve ilişkili fonksiyon değerleri

x	y	f(x, y)
8,6865	9,9265	-94,0226
5,1809	9,9265	-76,1298
8,0303	9,9265	-70,5246
6,1814	9,9265	-64,4295
0,8116	9,9265	-56,1888
8,5057	3,7333	-54,7479
4,3187	9,9265	-47,1915
6,6309	9,9265	-42,4367
6,5733	0,8388	17,3598
7,5052	6,3427	26,0426

Tablo 4.5 ve 4.6'dan görüldüğü üzere daha üçüncü iterasyon sonucunda bireylerin vermiş oldukları çözüm değerleri hızlı bir şekilde iyiye gitmektedir. Şekil 4.7'de üçüncü iterasyon sonucunda bireylerin global minimuma yaklaştıkları görülmektedir. Şekil 4.8'de ise her popülasyona ait en iyi değerler ile popülasyonun ortalama değeri de üretimin (iterasyonun) bir fonksiyonu olarak gösterilmiştir.



Şekil 4.7 Üçüncü popülasyona ait bireylerin eşyükselti eğrisi üzerinde gösterilimi

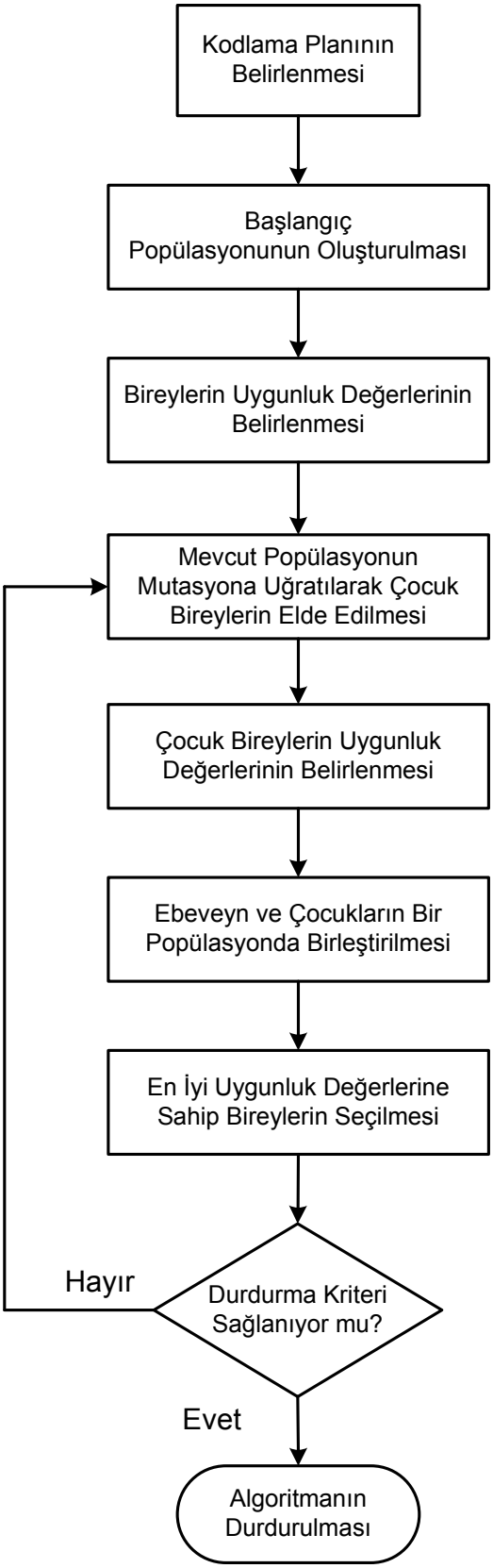


Şekil 4.8 Verilen örnek fonksiyon için GA ile elde edilen minimum değer ve popülasyonun ortalama değerinin iterasyon ile değişimi

4.2 Evrimsel Programlama

Çalışmamızda, birim yüklenme problemi için sistemdeki hangi generatörlerin devrede olup, hangilerinin devre dışında kalacaklarını belirleyen işletme planı evrimsel programlama yöntemi kullanarak belirlenmiştir.

Evrimsel programlama, ilk olarak Lawrence J. Fogel tarafından 1960 yılında tasarlanmıştır [Fogel vd., 1966]. Genetik algoritma yapısına benzeyen stokastik bir optimizasyon stratejisidir. Fakat genetik algoritmadan farklı olarak, doğada gözlenen özel genetik operatörler kullanarak bir diğer kişiden daha iyisini yapmak yerine, belli bir tür seviyesinde ebeveynler ile onların çocukları arasında kuvvetli bir davranışsal bağ aramayı vurgulamaktadır. Genetik algoritmada olduğu gibi bireylerin seçilip çaprazlamaya tabi tutulduğu seçim ve çaprazlama operatörleri evrimsel programlamada bulunmamaktadır. Evrimsel programlama genel anlamda mutasyon tabanlı bir evrimsel hesaplama tekniğidir.



Şekil 4.9 Evrimsel Programlama Akış Diyagramı

Şekil 4.9'dan da görüldüğü üzere evrimsel programlamanın genel bir planı şu sırayı takip etmektedir:

1) Rasgele olarak *N* adet bireye sahip bir başlangıç popülasyonu belirlenir. Popülasyonun boyutu problemin çözümü için önemlidir. Yüksek boyutlu bir popülasyon çözüm süresini uzamasına neden olacağı gibi, düşük sayıdaki popülasyon da çözüm uzayını tam olarak tarayamayabilmektedir. Problemin yapısına bağlı olarak yapılan denemeler neticesinde popülasyon sayısı belirlenmektedir.
2) Popülasyondaki her birey için bir uygunluk değeri belirlenir. Bu uygunluk değeri optimize edilecek problemin değeri de olabilir.
3) Popülasyondaki her ebeveyn birey mutasyona tabi tutularak *N* adet çocuk birey elde edilir. Elde edilen her çocuk birey için de uygunluk değeri belirlenir.
4) Bütün ebeveynler ve çocuklar bir bütün olarak popülasyona dahil edilip uygunluk değerlerine göre sıralanırlar. Popülasyonun birey sayısı $2\times N$ olmuştur. Şimdi en iyi uygunluk değerine sahip *N* adet birey seçilerek bir sonraki nesile aktarılırlar, diğerleri ise elenirler. Böylece popülasyonun boyutu yine *N* olmuş olur.
5) Bu işlem istenen çözüm elde edilene kadar veya belli bir nesil sayısına kadar devam ettirilir.

5. ÖĞRENEN AUTOMATA (LEARNING AUTOMATA)

5.1 Giriş

Tez çalışmamızda birim yüklenme probleminin çözümünde orijinal olarak öğrenen automata dahil edilmiştir. Bu yöntem, sistem için en ekonomik maliyet değeri ile en iyi kararlı çalışma değeri arasında istenen kriterleri sağlayan optimum bir işletim noktasının belirlenmesinde en iyi sonucun elde edilmesi aşamasında kullanılmış ve analiz gerçekleştirilmiştir. Bu bölümde öğrenen automatayı açıklayalım.

Bir automaton, daha önceden belirlenmiş işlemler dizisini veya kodlanmış komutlara ait bir cevabı otomatik olarak izlemek için tasarlanmış bir kontrol mekanizmasıdır. Klasik kontrol teorisinde bir işlemin kontrolü, sisteme ait bütün bilgilerin bilinmesini temel almaktadır. Bu durumda matematiksel modelin bilindiği varsayılmaktadır ve işlemin girişleri de deterministik zaman fonksiyonlarıdır. Kontrol teorisinde daha sonra gerçekleştirilen gelişmeler sistemdeki belirsizlik durumunu göz önüne almıştır. Stokastik kontrol teorisi bazı belirsizliklerin karakteristiğinin bilindiğini kabul eder. Bununla birlikte, belirsizlikler ve/veya giriş fonksiyonları üzerindeki bütün varsayımlar, başarılı bir şekilde sistemin kontrolünün yapılmasında yeterli olmayabilir. Bundan dolayı işlem süreçlerinin izlenmesi ve sistem hakkında daha fazla bilginin elde edilmesi gereklidir [Ünsal, 1997].

Öğrenme, geçmiş deneyimlerin bir sonucu olarak davranışta meydana gelen herhangi bir kalıcı değişim olarak tanımlanmaktadır. Bu yüzden bir öğrenme sistemi, zamana bağlı olarak davranışını daha iyi bir hedefe doğru düzeltme yeteneğine sahip olmalıdır. Matematiksel anlamda, öğrenme sisteminin hedefi, içeriği açıkça bilinmeyen bir fonksiyonun optimizasyonudur [Narendra vd., 1974]. Stokastik automaton, optimum bir aksiyon üzerinde herhangi bir bilgisi olmaksızın problemin çözümüne çalışır. Rasgele bir aksiyon seçildiğinde, çevreden gelecek cevap izlenir, aksiyon olasılıkları cevaba bağlı olarak güncellenir ve işlem tekrarlanır.

Öğrenen automata (Learning automata) denildiğinde, çıkış sonuçları için çevrenin (environment) verdiği cevap ile performansını arttırmak için sistematik olarak stratejisini güncelleyen bir automaton anlaşılmaktadır. Buna bağlı olarak çeşitli yapılarda öğrenen automata düzenekleri bulunmaktadır. Hem ekonomik işletim için üretim maliyetlerini belirleyen birim yüklenme problemi, hem de güç sisteminin kararlı işletimi için performans ölçümü eşzamanlı olarak optimizasyon anlamında birer performans indisi olarak

kullanılabilir. Değişken yapıdaki öğrenen automata (variable-structure learning automata), yüklerin rasgele değiştiği bir durum altında güç sisteminin optimum işletimi ile kararlı işletimi arasında bir uzlaşma sağlamak için uygulanabilir. Öğrenen automata, olasılık teorilerini ve Markov süreçlerini temel almaktadır.

İlk olarak, Tsetlin rasgele çevrelerde işlem gören deterministik automata davranışı konusunda çalışmıştır [Tsetlin, 1961]. Sabit yapılı öğrenen automata (fixed-structure learning automata), durum geçiş olasılıklarının sabit olduğu durumda, yüksek olasılığa sahip en iyi aksiyonları asimptotik olarak seçmeyi öğrenmektedir. Değişken yapılı stokastik automata, girişi temel alarak her adımda hem geçiş olasılıklarını hem de aksiyon olasılıklarını güncellemektedir.

Cover ve Hellman, sonlu hafızaya sahip iki-aksiyonlu sabit yapı üzerine çalışmış [Cover vd., 1970], Aso ve Kimura hangi tür mantıksal yapının çok aksiyonlu elverişli (expedient) automataya götürdüğünü göstermişlerdir [Aso ve Kimura, 1976]. Viswanathan ve Narendra, değişken yapıdaki öğrenen automata için pekiştirme düzeneği (reinforcement scheme) ve asimptotik karakteristikleri üzerine çalışma yapmıştır [Viswanathan vd., 1972]. Narendra ve Thathachar, öğrenen automatanın öğrenme algoritması, asimptotik davranışı ve hiyerarşik yapısını içine alan geniş bir saha üzerinde göze çarpan bir çalışmayı tatbik etmiştir [Narendra, 1989].

Kural-tabanlı (rule-based) sistemler, birçok kontrol probleminde iyi bir performans gösterse de, problem uzayındaki en ufak bir değişiklik için bile modifikasyon ihtiyacına bağlı bir dezavantaja sahiptirler. Buna ek olarak kural-tabanlı bir yaklaşım, özellikle uzman sistemler, beklenilmedik durumlar karşısında başarılı olamayabilir. Bir öğrenme sisteminin ardında yatan ana fikir, kontrol edilecek sisteme/çevreye ait tam bir bilgi olmaksızın davranışı sağlam bir şekilde sağlayabilmektir. Diğer öğrenme yaklaşımlarıyla kıyaslandığında, pekiştirmeli öğrenmenin en önemli avantajı pekiştirme işareti haricinde çevre hakkında herhangi bir bilgiye gereksinim duyulmamasıdır [Narendra vd., 1989; Marsh vd., 1993].

Pekiştirmeli bir öğrenme sistemi, memnun edici bir performans değeri elde edebilmek için her aksiyonunun belli sayıda test edilmesi ihtiyacından dolayı çoğu uygulama için diğer öğrenme yaklaşımlarından daha yavaştır. Bundan dolayı ya öğrenme işlemi çevre değişikliklerinden daha hızlı olmalı, ya da pekiştirmeli öğrenme çevre değişikliklerini daha önceden tahmin eden gelişmiş bir adaptif model ile birleştirilmelidir [Peng vd., 1993].

1970 yılından itibaren büyük ölçekli sistemler ile ilgili yapılan çalışmalarda, nonlineerlikler, belirsizlikler, hiyerarşik yapılar, eksik (tamamlanmamış) bilgilere sahip yapılar, gerçek-

zamanlı hesaplamalar, farklı sınıftaki kontrol yapıları gibi problemler üzerinde odaklanılmıştır. Bununla beraber bu tür problemlerin çözümünde her ne kadar geleneksel modelleme ve matematiksel analiz yöntemleri kullanılmış olsa da tatmin edici sonuçlar alınamamaktadır. Örnek olarak, güç sistem kontrolü ve kararlılığı alanında, sistemin kararlılığı, güvenilirliği ve ekonomik işletiminin potansiyelini araştırmak için uzun bir süre bunların birlikte kontrolünün yapılması düşünülmüştür. Bilindiği üzere bir güç sistemi, generatör uyarma sistemleri, türbin regülatörleri, kazanlar, statik gerilim kompanzatörleri, koruma röleleri, yükler gibi birçok elemanın kontrolörleri tarafından kontrol edilmektedir. Farklı kontrol yapılarına sahip bu tür bileşenler için tümleşik kontrol modeli oluşturacak bir kontrol stratejisini elde etmek zordur. Mevcut kontrol felsefesinde, çoğu kontrol algoritması lineer gösterimlere dayalıdır, sistem modelleri veya tahmini modellere ihtiyaç duyar ve güç sisteminin nonlineerliği, belirsizliği ve karmaşık işletim şartlarını ele almaz [Wu, 1995].

Mühendislik alanında ödün vermeye (trade-off) ait karar, bir çok kalite kriterinin eşzamanlı olarak ele alınması ile karakterize edilebilir. Güç sistemi alanında, genellikle birden fazla güç sistem özelliğini eşzamanlı olarak optimize etmek gerekmektedir. Hem ekonomik olarak güç üretimin sağlanması ve hem de üretilen gücün kararlı olması, elektrik enerji sektöründe önemli iki faktördür, fakat bunlar genelde birbirleriyle çelişkili olaylardır. Bu tür çok nesneli problemlerin çözümü, en iyi uzlaşmayı (veya ödünü) sağlayan çözümü gerektirmektedir [Lee vd., 1998].

Son yıllarda öğrenme teknikleri güç sistem problemlerine de uygulanmaktadır [Garcia vd., 1991; Wu vd., 1992]. Geleneksel kontrol felsefesinin aksine, öğrenen kontrol yapısı doğrudan bir modelleme işlemi gerektirmeden sistem belirsizlikleri, nonlineerlikler ve karmaşık yapılara sahip yapılarda çalışabilmektedir. Öğrenme tekniklerinden biri olarak “Learning Automata”, yapısı bilinmeyen turbo generatörlerin kontrolünde kullanılmaktadır [Wu, 1992; Wu vd., 1993].

Fu, stokastik automatayı kontrol mühendisliği alanına tanıtmıştır [Fu, 1967]. McLaren tarafından lineer güncelleme planları sunulmuştur [McLaren, 1966]. Chandrasekaran ve Shen nonlineer güncelleme planları, sabit olmayan ortamlar (nonstationary enviroments) ve automata oyunları üzerinde çalışmışlardır [Chandrasekaran ve Shen, 1968, 1969]. Narendra ve Thathachar LA’in teori ve uygulamalarında araştırmalar yaparak bu alana simülasyon çalışmalarını uygulamışlardır. “Learning Automata” isimli kitapları 1980 yılının sonuna kadar bu konuda yapılan çalışmaların tetkik edildiği learning automata teorisinin bir tanıtımıdır [Narendra ve Thathachar, 1989].

Learning automata'nın uygulandığı gerçek yaşam problemlerinin bir kısmı şu şekilde sıralanabilir; bioreaktörler [Gilbert vd., 1992], imalat işletmelerinin kontrolü [Sequeria vd., 1992], patern tanımlama [Oommen vd., 1994], graf bölümleme [Oommen vd., 1994], aktif taşıt süspansiyonu [Marsh vd., 1993].

5.2 Çevre (Environment) ve Automaton

Öğrenen automaton, rasgele bir çevre (environment) içerisinde sonlu sayıda aksiyonun rol alması olarak ifade edilebilir. Belirli bir aksiyonun görev alması durumunda, çevre buna karşın elverişli veya elverişsiz bir cevap üretecektir. Automatonun tasarımındaki amaç, daha önceki aksiyon ve cevapların da kılavuzluk ettiği herhangi bir evredeki aksiyonun nasıl seçileceğine karar vermektir. Buradaki önemli nokta, kararlar verilirken çevrenin "doğası" ile ilgili çok az bilgiye sahip olunmasıdır. Karar verici, hiyerarşik yapının bir parçası olabilir fakat bu hiyerarşi içerisindeki rolünden haberdar olmayabilir. Bunun yanında, karar verici tarafından bilinmeyen diğer etkenlerin aksiyonlarının, çevrenin vereceği yanıtı etkilemiş olabileceği de belirsizliği ortaya çıkarabilmektedir [Ünsal, 1997; Narendra vd., 1989].

Automatonun "yaşadığı" çevrenin, automaton aksiyonu için üreteceği cevap, kabul edilebilir cevaplar kümesine ait olup, automaton aksiyonu ile olasılıksal olarak ilişkilidir. Çevre terimi, learning automata bağlamında kolayca tanımlanamamaktadır. Tanım, bir automatonun işlem yaptığı, geniş ölçekli içeriği bilinmeyen bir rasgele ortam sınıfını kapsar. Matematiksel olarak, $\{\underline{\alpha}, \underline{c}, \underline{\beta}\}$ üçlüsü ile gösterilir. Burada $\underline{\alpha}$ sonlu aksiyon/çıkış kümesini, $\underline{\beta}$ (binary) giriş/cevap kümesini, $\underline{c}$ ise penaltı olasılıklarına ait kümeyi gösterir. Her c_i elemanı $\underline{\alpha}$ kümesinin bir aksiyonu α_i ile ilişkilidir.

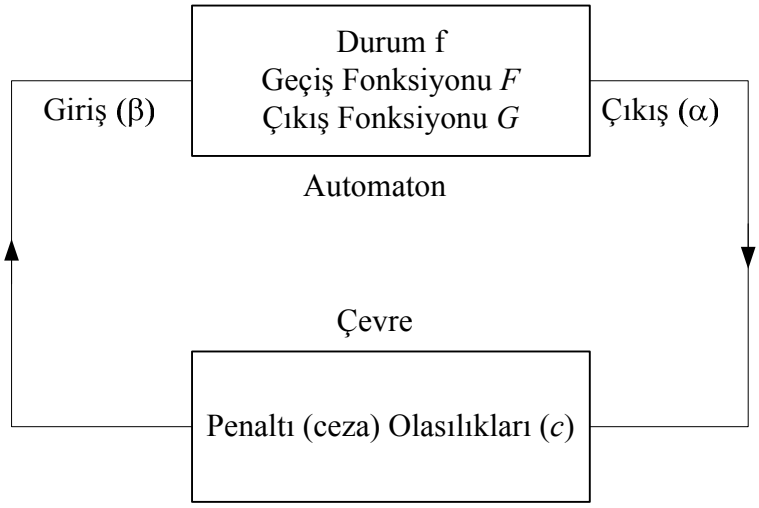
Automatonun çıkışı (aksiyon) $\alpha(n)$, $\underline{\alpha}$ kümesine ait olup, t=n anında çevreye uygulanır. Giriş $\beta(n)$, çevrenin yanıtı olup $\underline{\beta}$ kümesinin bir elemanıdır, β_1 ve β_2 değerlerinden birini alır. En basit durumda β_i değerleri 0 ve 1 olarak seçilir. Burada 1 başarısızlık/penaltı yanıtı ile ilişkilidir. $\underline{c}$ elemanlarının tanımı;

$$P\{\beta(n) = 1 \mid \alpha(n) = \alpha_i\} = c_i \quad (i = 1, 2, \ldots) \tag{5.1}$$

Bu nedenle c_i, α_i aksiyonunun çevreye girişinin penaltı ile sonuçlandığı olasılık değeridir. Penaltı olasılıkları c_i sabit olduğunda, çevre "sabit (durağan) çevre" olarak ifade edilir.

Çevrenin yanıt kümesine bağlı olarak tanımlanan çeşitli modeller bulunmaktadır. Çevrenin

cevabının sadece 0 veya 1 değerlerinden birini aldığı modeller, "P-modelleri" olarak ifade edilmektedir. Bu durumda, 1 değerini alan yanıt "elverişsiz" (başarısız, penaltı) cevap olarak ele alınırken, 0 değerinin anlamı aksiyonun "elverişli" olduğudur. Çevrenin daha ileri bir genelleştirmesi, [a,b] aralığındaki sonlu sayı değerlerine sahip, cevap kümelerine izin verebilir. Bu tür modellere "Q-modelleri" adı verilir. Çevrenin yanıtının [a,b] aralığındaki muhtemel değerler ile sürekli rasgele değişen olması durumunda ise model, "S-modeli" olarak adlandırılır [Narendra ve Thathachar, 1989].



Şekil 5.1 Automaton ve Çevre

Automaton $\{\underline{\Phi}, \underline{\alpha}, \underline{\beta}, F(\bullet,\bullet), H(\bullet,\bullet)\}$ beşlisi ile ifade edilebilir. Burada,

- $\underline{\Phi}$, dahili durumlar kümesidir. Herhangi bir n anında, $\phi(n)$ durumu $\underline{\Phi}=\{\phi_1, \phi_2, ..., \phi_s\}$ sonlu kümesinin bir elemanıdır.
- $\underline{\alpha}$, aksiyonların (veya automatonun çıkışları) kümesidir. n anında, $\alpha(n)$ ile gösterilen automatonun çıkışı veya aksiyonu, $\underline{\alpha}=\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ sonlu kümesinin bir elemanıdır.
- $\underline{\beta}$, cevapların (veya çevreden gelen girişler) kümesidir. Çevreden gelen $\beta(n)$ girişleri, sonlu veya sonsuz sayıda eleman içeren $\underline{\beta}$ kümesinin bir elemanıdır;

$$\underline{\beta}=\{\beta_1, \beta_2, \ldots, \beta_m\} \text{ veya } \underline{\beta}=\{(a,b)\} \tag{5.2}$$

- $F(\bullet,\bullet)$: $\underline{\Phi}\times\underline{\beta}\rightarrow\underline{\mathbf{\Phi}}$ mevcut durum ve girişten bir sonraki durumu tasarlayan fonksiyondur. F deterministik (belirleyici) veya stokastik (olasılıksal) olabilir:

$$\phi(n+1)=F[\phi(n),\beta(n)] \tag{5.3}$$

- $H(\bullet,\bullet)$: $\underline{\Phi}\times\underline{\beta}\rightarrow\underline{\alpha}$ mevcut durum ve girişten mevcut çıkışı oluşturan fonksiyondur. Eğer mevcut çıkış sadece mevcut duruma bağlıysa, automaton durum-çıkış automatonu olarak ifade edilir. Bu durumda, $H(\bullet,\bullet)$ fonksiyonu yine deterministik veya stokastik bir çıkış fonksiyonu olan $G(\bullet):\underline{\Phi}\rightarrow\underline{\alpha}$, ile yer değiştirir:

$$\alpha(n)=G[\phi(n)] \tag{5.4}$$

5.3 Stokastik Automaton

Stokastik automaton, F ve G fonksiyonlarından en az birinin stokastik olması durumunda tanımlanır. Eğer geçiş fonksiyonu F stokastik ise F'nin f_{ij}^{β} elemanı, automatonun takip eden bir β girişi için ϕ_i durumundan ϕ_j durumuna geçişinin olasılığını göstermektedir:

$$f_{ij}^{\beta} = \Pr\{\phi(n+1)=\phi_j \mid \phi(n)=\phi_i\ ,\ \beta(n)=\beta\} \qquad i,j=1,\ 2,\ \ldots,\ s \qquad \beta=\beta_1,\ \beta_2,\ \ldots,\ \beta_m \tag{5.5}$$

G fonksiyonu için de tanımlama benzer olarak:

$$g_{ij}=\Pr\{\alpha(n)=\alpha_j \mid \phi(n)=\phi_i\} \quad i,j=1,\ 2,\ \ldots,\ r \tag{5.6}$$

f_{ij}^{β} olasılıkları [a,b] aralığında olup, olasılık derecesini korumak zorundadır:

$$\sum_{j=1}^{s} f_{ij}^{\beta} = 1 \ \text{, her } \beta \text{ ve } i \text{ değeri için.} \tag{5.7}$$

Örnek:

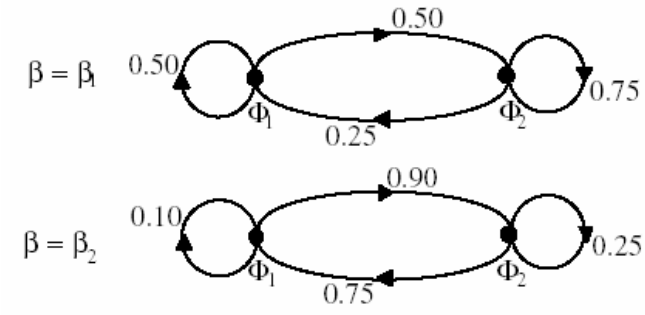
Durumlar: ϕ_1, ϕ_2

Girişler: β_1, β_2

Geçiş Matrisleri: $F(\beta_1)=\begin{bmatrix}0.50 & 0.50\\ 0.25 & 0.75\end{bmatrix}$

$$F(\beta_2)=\begin{bmatrix}0.10 & 0.90\\ 0.75 & 0.25\end{bmatrix}$$

Geçiş grafı (graph):



Bu örnekte f_{ij}^{β} koşullu olasılıklarının sabit, zaman adımı *n*'den ve giriş sıralamasından bağımsız olduğu varsayılmıştır. Bu tür bir stokastik automaton, "sabit-yapılı automaton (fixed-structure automaton)" olarak ifade edilmektedir. Her *n* adımında, o andaki çevre yanıtını temel alarak geçiş olasılıklarının güncellenmesi yararlı olmaktadır. Geçiş olasılıklarının güncellendiği bu tür automaton da "değişken-yapılı automaton (variable-structure automaton)" olarak anılmaktadır.

Bunun yanında, değişken-yapılı automaton durumu ele alındığında, yukarıda ifade edilen *F* ve *G* geçiş fonksiyonlarının tanımlamaları belirgin olarak kullanılmaz. Geçiş matrislerinin yerine, pekiştirme düzeneklerini (reinforcement schemes) belirtmek için aksiyon olasılıkları vektörü *p*(*n*) tanımlanır.

Eğer değişken-yapılı automaton bir durum-çıkış automatonu ise, ve bir durumdan diğerine geçiş olasılıkları f_{ij}^{β} mevcut durum ve çevre girişine bağlı değilse, bu durumda aksiyon olasılık vektörü ile geçiş matrisleri arasındaki ilişki şu şekildedir:

- Automaton bir durum-çıkış automatonu ise, *G* geçiş matrisini ihmal ederiz ve sadece *F* geçiş matrisini göz önüne alırız.
- Geçiş (durum değiştirme) çevrenin yanıtına bağlı olmadığından, elemanlarının f_{ij} olduğu sadece bir adet durum değiştirme matrisi *F* bulunmaktadır:

$$F(\beta_1) = F(\beta_2) \equiv F = \begin{bmatrix} f_{11} & f_{12} \\ f_{21} & f_{22} \end{bmatrix} \tag{5.8}$$

- Bunun yanında, bir durumda olma olasılığı (veya bir aksiyonu üretme), automatonun başlangıç veya bir önceki durumuna bağlı değildir. Bu durumda, geçiş matrisi sadeleştirilirse:

$$F = [f_{11} = f_{21} \equiv f_1 \quad f_{12} = f_{22} \equiv f_2] \equiv p \tag{5.9}$$

burada p vektörü, automatonun i.nci durumda olma (veya i.nci çıkışı/aksiyonu seçme) olasılıkları p_i'leri içermektedir.

5.4 Değişken-Yapılı Automaton ve Performans Değerlendirmesi

Değişken-yapılı automatayı tanıtmadan önce, kendi kendine öğrenen bir değişken-yapılı automatanın performans değerlendirmesi için gerekli tanımlamaları yapacağız. Kendi kendine öğrenen bir automaton, çevre ile etkileşimini temel alan bir dizi aksiyonu üretir. Eğer automaton işlem esnasında öğreniyorsa, performansı "sezgisel" metotlara göre daha üstün olacaktır. Automatonun performansını değerlendirmek için, nicel (quantitative) davranış biçimlerini (norms of behavior) incelememiz gerekir. Öğrenme davranışını değerlendirecek nicel temeller, en basit P-modelleri ve sabit rastlantısal çevrelerde bile oldukça karmaşıktır. "Davranış biçimleri" tanımlamalarını yapmak için en basit durumu göz önüne alacağız.

5.4.1 Davranış Biçimleri

Eğer önceye ait bir bilgi mevcut değilse, α_i farklı aksiyonların ayırt edilebileceği esaslar yoktur. Bu tür bir durumda, bütün aksiyon olasılıklarının eşit olacağı bir "saf-şans (pure chance)" durumu bulunmaktadır. r-aksiyonlu bir automaton için, aksiyon olasılık vektörü $p(n)=\Pr\{\alpha(n)=\alpha_i\}$ şu şekilde verilir;

$$p_i(n) = \frac{1}{r} \qquad i=1, 2, \ldots, r \tag{5.10}$$

Bu tür bir automaton, "saf-şans automatonu" olarak adlandırılır ve karşılaştırma için standart olarak kullanılır. Herhangi bir öğrenen automaton en azından saf-şans automatonundan daha iyi işlem yapacaktır.

Penaltı olasılıkları $\{c_1, c_2, \ldots, c_{1r}\}$'nın $c_i=\Pr\{\beta(n)=1|\alpha(n)=\alpha_i\}$olduğu bir sabit rastlantısal çevreyi göz önüne alalım. Verilen bir aksiyon olasılık vektörü için, "ortalama penaltı" $M(n)$

ile belirtilmektedir:

$M(n)=E[\beta(n) \mid p(n)]=\Pr\{\beta(n)=1 \mid p(n)\}$

$$= \sum_{i=1}^{r} \Pr\{\beta(n)=1 \mid \alpha(n)=\alpha_i\} \cdot \Pr\{\alpha(n)=\alpha_i\} \tag{5.11}$$

$$= \sum_{i=1}^{r} c_i p_i(n)$$

Saf-şans automatonu için $M(n)$, sabit bir değer olan M_0 ile gösterilir:

$$M_0 = \frac{1}{r} \sum_{i=1}^{r} c_i \tag{5.12}$$

Göz önüne alınacak bir diğer eşitlik de,

$E[M(n)]=E\{E[\beta(n) \mid p(n)]\}$

$$=E[\beta(n)] \tag{5.13}$$

burada $E[M(n)]$ automatonun ortalama girişidir. Yukarıdaki tanımlamalar kullanılarak automaton davranış biçimlerine ait tanımlamalar yapılabilir:

Tanım 5.1: Eğer $\lim_{n\to\infty} E[M(n)] < M_0$ ise öğrenen automaton "elverişli (expedient)"dir.

$\sum_{i=1}^{r} p_i(n) = 1$ olduğundan, $\inf M(n) = inf_{p(n)} \left\{ \sum_{i=1}^{r} c_i p_i(n) \right\} = \min_i \{c_i\} \equiv c_l$ yazılabilir.

Tanım 5.2: Eğer $\lim_{n\to\infty} E[M(n)] = c_l$ (5.14) ise öğrenen automaton için "en iyi (optimal)" olarak ifade edilir.

En iyilik, minimum penaltı olasılığı c_l'ile ilişkili α_l aksiyonunun asimptotik olarak 1 olasılığı ile seçilmesini gerektirmektedir. Birçok araştırmacının göstermiş olduğu gayretlere rağmen, en iyiliği sağlayan genel bir algoritma bulunamamıştır [Baba, 1983; Kushner vd., 1972; Narendra vd., 1989]. Verilen her durum için en iyiliğin sağlanabilmesi mümkün olmayabilir.

Tanım 5.3: Eğer $\lim_{n\to\infty} E[M(n)] = c_l + \varepsilon$ (5.15) ise öğrenen automaton "*e*-en iyi" olarak ifade edilir. Burada *e*, keyfi olarak en küçük pozitif sayıdır.

Açıkça belirtilmiş başlangıç koşulları ve belirli penaltı olasılık kümeleri için bazı automatalar

tanımlamalarda ifade edilen koşulları sağlar. Bununla birlikte, biz automatanın keyfi çevreler ve keyfi başlangıç koşullarında arzu edilir bir davranış sergilemesini isteriz. Bu gereksinimler, tamamıyla elverişli automaton tarafından kısmen karşılanır.

Tanım 5.4: Eğer $E[M(n+1) \mid p(n)] < M(n)$ (5.16) ise öğrenen automaton "tamamen elverişli"dir.

bütün n değerleri için, $p_i(n) \in (0,1)$ arasında ve olası kümeler $\{c_1, c_2, \ldots, c_r\}$'dir. İki tarafın ortalamaları alındığında, aşağıdaki eşitsizlik elde edilir.

$$E[M(n+1)] < E[M(n)] \tag{5.17}$$

Akıllı kontrol uygulamalarına learning automata tekniklerinin uygulanmasında, biz en çok c_j=0 penaltı olasılığına sahip bir adet "en iyi" α_j bulunması durumu ile ilgilenmekteyiz. Bu tür bir çevrede, öğrenen automatonun "saf en iyiliğe (pure optimal)" ulaşmaya çalıştığı beklenir:

$$p(n) \rightarrow e_j^r \tag{5.18}$$

burada e_j^r, j.nci bileşeni 1'e eşit olan birim vektördür.

Diğer bir anlamda, $E[M(n)] = \sum_{i=1}^{r} c_i p_i(n) \rightarrow c_l$ durumunda automaton en iyi olacaktır.

5.4.2 Değişken-Yapılı Automata

Aksiyon olasılıklarının (veya durum geçişlerinin) bir pekiştirme düzeni (reinforcement scheme) kullanılarak her adımda güncellendiği daha genel stokastik sistemlerin dikkate alınmasıyla, çok daha esnek bir öğrenen (learning) automaton elde edilebilir. Lineer ve nonlineer pekiştirme düzeninin genel bir tarifini yapalım. Kolaylık olması için, her durumun bir aksiyona tekabül ettiğini varsayalım. Bu durumda automatonumuz bir durum-çıkış automatonu olacaktır.

Genel olarak, pekiştirme düzeni şu şekilde gösterilebilir:

$$p(n+1)=T_1[p(n),\alpha(n),\beta(n)] \tag{5.19a}$$

veya

$$f_{ij}^{\beta}(n+1) = T_2\left[f_{ij}^{\beta}(n), \phi(n), \phi(n+1), \beta(n)\right] \tag{5.19b}$$

burada T_1 ve T_2 eşlemelerdir (mappings). α aksiyonu, β çevreden gelen girişi ve ϕ automatonun durumunu göstermektedir. Şu andan itibaren, pekiştirme düzeni için (5.19a)'daki matematiksel ifadeyi kullanacağız. Eğer $p(n+1)$, $p(n)$'nin lineer bir fonksiyonu ise pekiştirme düzeni lineerdir, diğer durumda ise nonlineerdir.

En Basit Durum:

$\underline{\beta} = \{0,1\}$ olan sabit bir çevrede, r aksiyonlu değişken-yapılı bir automatonu ele alalım. Aksiyon olasılıklarının güncellenmesi için kullanılan genel düzen şu şekildedir:

Eğer $\alpha(n)=\alpha_i$ ise,

$$\left.\begin{array}{ll} & p_j(n+1) = p_j(n) - g_i(p(n)) \\ \beta=0 \text{ için,} & \text{tüm } j \neq i \text{ için} \\ & p_i(n+1) = p_i(n) + \sum_{\substack{k=1 \\ k\neq i}}^{r} g_k(p(n)) \\ & \\ & p_j(n+1) = p_j(n) + h_i(p(n)) \\ \beta=1 \text{ için,} & \text{tüm } j \neq i \text{ için} \\ & p_i(n+1) = p_i(n) - \sum_{\substack{k=1 \\ k\neq i}}^{r} h_k(p(n)) \end{array}\right\} \qquad (5.20)$$

burada g_k ve h_k (k=1, 2, …, r) sürekli, negatif olmayan fonksiyonlar olup aşağıdaki eşitliklere sahiptirler:

$$0 < g_k(p(n)) < p_k(n)$$

$$0 < \sum_{\substack{k=1 \\ k\neq i}}^{r} [p_k(n) + h_k(p(n))] < 1 \qquad (5.21)$$

yukarıdaki kısıtlar tüm i=1, 2, …, r ve (0,1) aralığındaki p_k olasılıkları için sağlanır. Güncelleme fonksiyonlarındaki bu iki kısıt, bütün zaman adımlarında aksiyon olasılıklarının toplamının 1'e eşit olmasını garanti eder.

5.5 Pekiştirme Düzenleri

Pekiştirme düzeni, öğrenen automata için öğrenme işleminin temelidir. Bu düzenekler lineerlikleri baz alınarak kategorize edilirler. Değişken yapıdaki öğrenen automata için lineer, nonlineer ve hibrit (karma) düzenekleri mevcut olarak bulunmaktadır [Narendra vd., 1989]. Bir pekiştirme düzeneğinin lineerlik karakteristiği, g_k ve h_k güncelleme fonksiyonlarının lineerliği ile tanımlanmaktadır. Bunun yanında, stokastik öğrenen automata için mevcut bütün algoritmaları iki sınıfa ayırmak da mümkündür [Najim ve Poznyak, 1994]:

- Öngörüsel olmayan (nonprojectional) algoritmalar; burada bireysel olasılıklar bir önceki değerleri temel alınarak güncellenir:

$$p_i(n+1) = p_i(n) + \delta \cdot k_i(p_i(n), \alpha(n), \beta(n)) \tag{5.22}$$

denklemdeki δ küçük bir reel sayı, k_i ise güncelleme için eşleme (mapping) fonksiyonudur.

- Öngörüsel (projectional) algoritmalar; burada olasılıklar, her eleman için olasılık vektörünü belirli bir değere eşleyen bir fonksiyon tarafından güncellenir:

$$p_i(n+1) = k_i(p(n), \alpha(n), \beta(n)) \tag{5.23}$$

eşitlikteki k_i eşleme fonksiyonudur.

Bu iki alt sınıf arasındaki ana fark, öngörüsel olmayan algoritmaların sadece çevre cevabının binary olması durumunda (yani P-çevre modelinde) kullanılabilir olmasıdır. Öngörüsel algoritmalar bütün çevre modellerinde ve daha karmaşık yapılarda kullanılabilir. Bunun yanında, öngörüsel algoritmaların gerçeklenmesi hesaplama anlamında daha fazla işlem gerektirir.

Pekiştirme düzenekleri alanında yapılan ilk çalışmalar, çözümün basit olması nedeniyle lineer düzenekleri merkez almaktaydı. Daha karmaşık ve verimli pekiştirme düzeneklerine olan ihtiyaç, en sonunda araştırmacıları nonlineer (ve hibrit) düzeneklere götürmüştür.

5.5.1 Lineer Pekiştirme Düzenekleri

r-aksiyonlu bir öğrenen automaton için, lineer pekiştirme düzeneğinin genel ifadesi aşağıdaki eşitliklerde verilen fonksiyonların Eşitlik 5.20'de yerine konulmasıyla elde edilebilir:

$$g_k\left(p(n)\right) = a \cdot p_k(n)$$

$$h_k\left(p(n)\right) = \frac{b}{r-1} - b \cdot p_k(n) \tag{5.24}$$

$$0 < a, b < 1$$

Bu durumda, genel lineer düzeneğine ait ifade şu şekilde verilebilir:

Eğer $\alpha(n)=\alpha_i$ ise,

$$\left.\begin{array}{ll} & p_j(n+1) = (1-a)\cdot p_j(n) \\ \beta=0 \text{ için,} & \text{tüm } j \neq i \text{ için} \\ & p_i(n+1) = p_i(n) + a\cdot\left[1 - p_i(n)\right] \\ & \\ & p_j(n+1) = \dfrac{b}{r-1} + (1-b)\cdot p_j(n) \\ \beta=1 \text{ için,} & \text{tüm } j \neq i \text{ için} \\ & p_i(n+1) = (1-b)\cdot p_i(n) \end{array}\right\} \tag{5.25}$$

Tanımdan da görüldüğü üzere, *a*- parametresi ödül (reward) cevabı ile *b*-parametresi ise ceza (penaltı) cevabı ile ilişkilidir. Eğer öğrenme parametreleri *a* ve *b* birbirine eşit ise, düzenek lineer ödül-ceza düzeneği L_{R-P} olarak adlandırılır [Bush ve Mosteller, 1958]. Bu durumda, olasılık vektörünün güncelleme oranı, çevrenin cevabı ne olursa olsun her adımda aynıdır. Bu düzenek, matematiksel psikolojide göz önüne alınan ilk düzenektir.

Lineer ödül-ceza düzeneği L_{R-P} için, n-zaman adımında olasılık vektörü için beklenen değer $E[p(n)]$, çok basit bir şekilde hesaplanabilir. Diferansiyel denklem (difference equation) sonucunun özdeğerlerinin analizi ile, diferansiyel denklem takımının asimptotik çözümünün bize aşağıdaki sonucu verdiği görülebilir:

$$\lim_{n\to\infty} E\left[M(n)\right] = \frac{r}{\sum_{k=1}^{r} 1/c_k} < \frac{\sum_{k=1}^{r} c_k}{r} = M_0 \tag{5.26}$$

Bu durumda, Tanım 5.1'de belirtildiği üzere, L_{R-P} düzeneğini kullanan çok-aksiyonlu automaton, tüm başlangıç aksiyon olasılıkları ve sabit rasgele çevreler için elverişlidir.

Değişken-yapıdaki bir automatonun öğrenme davranışı göz önüne alındığında elverişlilik, diğerlerine göre zayıf bir durumdur. Elverişli bir automaton, saf-şanslı bir automatondan daha iyi işlem yapar, fakat optimum çözüme ulaşacağı garanti edilemez. Daha iyi bir öğrenme mekanizması elde etmek için, lineer pekiştirme düzeneğinin parametrelerinin değişimi şu şekilde olmalıdır: eğer öğrenme parametresi *b* sıfır olarak alınırsa, bu durumda düzenek lineer ödül-hareketsizlik (reward-inaction) düzeneği L_{R-I} ismini alır. Bunun anlamı, çevreden gelen bir ödül cevabında aksiyon olasılıklarının güncellendiği, ceza durumlarının değerlendirmeye alınmadığıdır.

Bu düzenek için, minimum ceza olasılığı c_l'ye sahip olan α_l aksiyonunun olasılığı $p_l(n)$'nin tekdüze olarak (monotonically) 1'e yaklaşacağını göstermek mümkündür. *a* parametresi keyfi küçük bir değer olarak seçildiğinde, arzu edildiği üzere $\Pr\left\{\lim_{n\to\infty} p_l(n) = 1\right\}$ birim değere yaklaşacaktır. Bu öğrenen automatayı "ε-en iyi (optimal)" yapacaktır [Narendra vd., 1989].

6. METODOLOJİ

Tez çalışmasında geliştirilen yönteme ait açıklamalar bu bölümde verilmiştir ve çalışmanın gerçekleştirilmesinde uygulanan metodoloji aşamaları açıklanmıştır. Tez çalışmasında geliştirilen yöntemin uygulanması üç ana aşamadan oluşmaktadır. Birinci aşamada, sistemin enerji üretimi, maliyet açısından en ekonomik olacak şekilde elde edilmektedir. İkinci aşamada sistemin kararlılığı göz önüne alınmakta ve kararlılık açısından sistem için en uygun işletim noktası belirlenmektedir. Üçüncü aşamada ise, ilk iki aşamada elde edilen, en ekonomik maliyet ve en iyi kararlılık noktaları arasında belirli eşik değerlerin de dikkate alınması sonucunda sistem için bu iki kriteri de birlikte sağlayan optimum bir çalışma noktasına ulaşmak hedeflenmektedir. Tez çalışmasında uygulamanın gerçekleştirilmesinde belirtilen bu aşamalar aşağıda ayrıntılı bir biçimde verilmiştir.

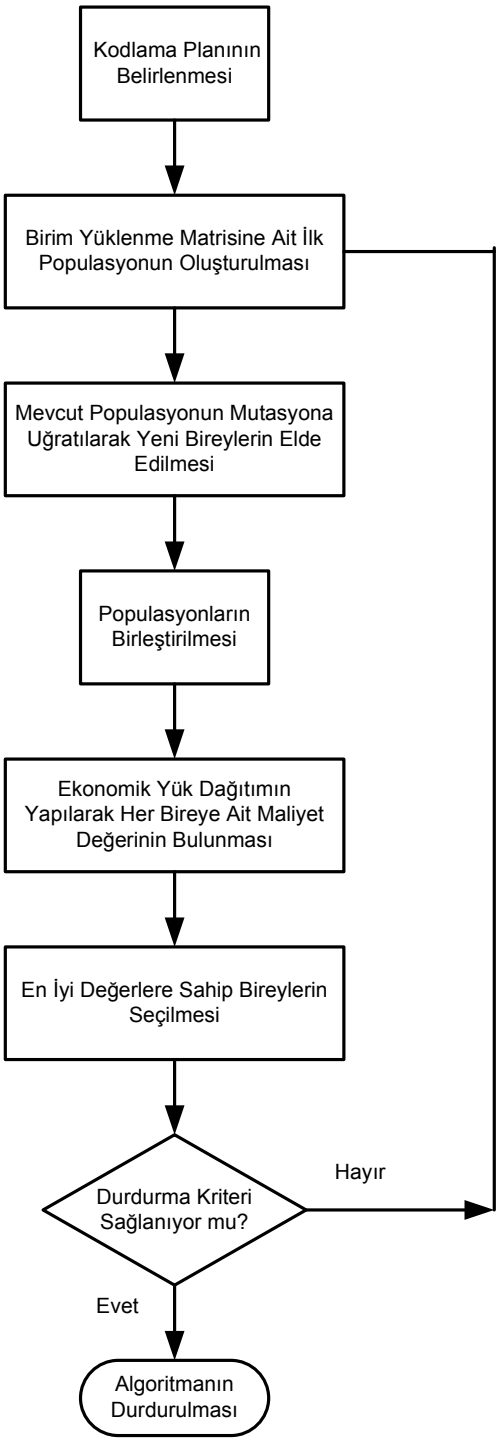
6.1 Ekonomik Kriter

Sistemde ekonomik olarak enerji üretimi söz konusu olduğunda aklımıza ilk olarak birim yüklenme problemi gelmektedir. Birim yüklenme problemi basitçe iki ana kısımdan oluşmaktadır. Birincisi birimlerin devrede veya devre dışında olmaları durumunun belirlenmesi, ikincisi ise devrede bulunan birimlerin üreteceği enerjinin en ekonomik şekilde belirlenmesidir. Pek çok sayısal veya bulgusal metotlar bu problemin çözümü için kullanılmaktadır. Probleme ait olan kısıtların çokluğu ve uygulamadaki zorlukları nedeniyle kullanılan metotlar farklılık göstermekte ve birbirlerine karşı belli üstünlüklere sahip olabilmektedir. Bu çalışmada en ekonomik maliyetin bulunmasında, evrimsel hesaplamalar yöntemlerinden olan evrimsel programlama ve genetik algoritma yöntemleri kullanılmıştır.

Birim yüklenme problemi daha önce de bahsedildiği üzere sahip olduğu kısıtlar yüzünden çözümü zor bir problem haline gelmektedir. Özellikle generatör birimlerinin minimum devrede kalma ve minimum devre dışı kalma süreleri problemin çözümünde büyük zorluklar meydana getirmektedir. Tez çalışmasında hangi birimlerin devrede olup olmayacağını evrimsel programlama yöntemi ile belirlemekteyiz. Bu yöntemin kullanılması ile çözüme ait akış diyagramı Şekil 6.1'de verilmiştir.

Şekil 6.1'den de görüldüğü üzere problemin çözümünde ilk olarak probleme ait değişkenlerin ve kısıtların program içerisinde yer almaları için kodlanmaları gerekmektedir. Her birey (ebeveyn ve çocuklar), N×T boyutunda matris yapısına sahiptir. Bu matrise birim yüklenme

matrisi adı verilmektedir. Matris elemanları birimlerin devrede olup olmadığını ifade eden sırasıyla 1 ve 0 değerlerini alarak birimlerin çalışma durumunu göstermektedir. Her birey esasında birim yüklenme planlamasına ait bir çözümü ifade etmektedir. Şekil 6.2'de bu yapı görülmektedir.



Şekil 6.1 Birim yüklenme probleminin evrimsel programlama ile çözümü

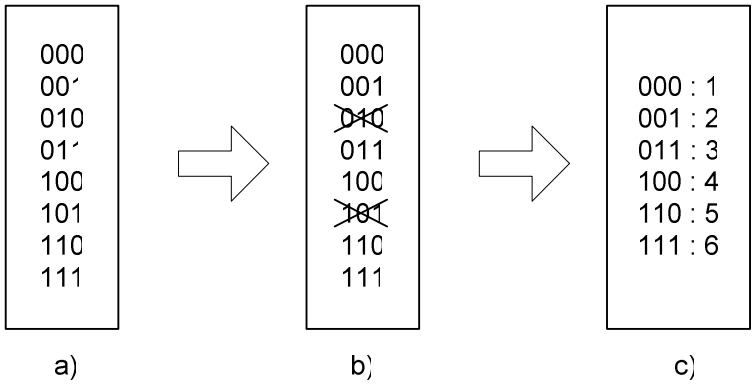
Birimler (N)	Saat									
	1	2	3	4	5	6	·	*t*	·	*T*
1	1	1	1	1	1	1	·	1	·	1
2	0	0	1	1	1	0	·	0	·	0
.	.	.	.	.	.	.	.	.	.	.
i	0	0	0	1	1	1	·	1	·	1
.	.	.	.	.	.	.	.	.	.	.
N	1	1	1	1	0	0		0	·	0

Şekil 6.2 Birim Yüklenme Matrisi

Şekil 6.2'de yapısı görülen birim yüklenme matrisinin oluşturulması, enerji sisteminin fiziksel kısıtları ve enerji üretim birimlerinin dinamik kısıtları yüzünden zorluklar içermektedir. Özellikle minimum devrede veya devre dışında kalma süreleri, bu matrisin oluşturulmasındaki zorlukta en büyük rolü oynamaktadır. Evrimsel programlama ile bu zaman kısıtlarının kodlanması ve matrisin oluşturulmasında büyük kolaylık sağlanmıştır.

Şimdi basit bir örnek üzerinde birim matrisin bu zaman kriterlerini göz önüne alarak nasıl oluşturulduğunu inceleyelim. Sistemdeki *i*-generatörünün minimum devrede kalma süresinin (MUT) 3 saat ve minimum devre dışında kalma süresinin (MDT) 2 saat olduğunu kabul edelim. $p \geq \max(MUT_i, MDT_i)$ olan bir *p* değişkeni seçelim. Bu *p* değişkeni *T* işletim dilimini p-eşit parçaya bölsün. *T*=12 periyotluk işletim dilimi için *p,* generatör biriminin minimum devrede kalma süresinin değeri olan 3 olarak belirlenmiştir.

Şimdi, Şekil 6.3a'da görülen p-bit uzunluğuna sahip binary sayılardan oluşan bir binary sistem üretelim. Bu sistemden MUT ve MDT kısıt değerlerini aşan binary sayılarını eleyelim (Şekil 6.3b). Sonuç olarak, geri kalan uygun binary sayılarını kodlayalım (Şekil 6.3c). Bu kodlar, *i*-generatörünün *p*-saat kadar işletim durumunu içeren kümelerden oluşmaktadır.



Şekil 6.3 Generatör çalışma durumlarını içeren kümelerin oluşturulması

i-generatörünün çalışma durumlarını gösteren kümeleri elde ettikten sonra, bu generatörün *T* periyot boyunca işletim durumunu gerçekleştirmek için Şekil 6.3c'deki kümelerin uygun bir sırayla kombinasyonlarının oluşturulması gerekmektedir. Tablo 6.1'de mevcut durumdaki kümeden sonra gelecek uygun kümelerin hangileri olacağı gösterilmiştir.

Tablo 6.1 *i*-generatör birimine ait durum kümelerinin uygun kombinasyonları

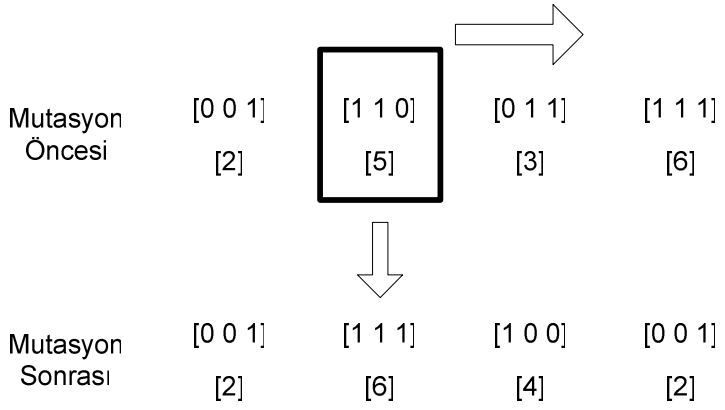
Mevcut Durum Kümesinin Kodu	Bir Sonra Gelebilecek Uygun Kümenin Kodu
1	1, 2, 3, 6
2	5, 6
3	4, 5, 6
4	1, 2, 3, 6
5	1, 2, 3
6	1, 2, 4, 5, 6

Tablo 6.1'in yardımı ile *i*-generatörüne ait *T* işletim dilimine ait bir çalışma durumunun belirlenmesi Tablo 6.2'de verilmiştir. Bu işletim durumu *i*-generatörüne ait MUT_i ve MDT_i kısıtlarını kapsayan bir çözümdür. Bu işlem sistemdeki bütün generatör birimleri için uygulanarak birim yüklenme matrisi elde edilmektedir. Her birim yüklenme matrisi de evrimsel programlama tekniği için popülasyondaki bir bireyi oluşturmaktadır.

Tablo 6.2 *i*-generatör birimine ait *T* işletim dilimine ait çalışma durumu

Birim , Saat	1 2 3	4 5 6	7 8 9	10 11 12
i	[0 0 1]	[1 1 0]	[0 1 1]	[1 1 1]
Küme Kodu	[2]	[5]	[3]	[6]

Şekil 6.1'deki akış diyagramından da görüldüğü üzere, evrimsel programlamada mevcut popülasyona mutasyon işlemi uygulanarak yeni popülasyonlar elde edilmektedir. Mutasyon işlemine şu şekilde basit bir örnek verebiliriz. Tablo 6.2'de *i*-generatörü için belirlemiş olduğumuz işletme durumunu ele alalım. Rasgele olarak bir durum kümesini belirleyelim. Bu örnek için ikinci sıradaki durum kümesi mutasyon işlemi için başlangıç noktamız olsun. Yeni bireyimiz Şekil 6.4'teki gibi belirlenecektir. Seçilen küme bir önceki durum da göz önüne alınarak uygun bir durum kümesi ile değiştirilerek bu işlem takip eden diğer durum kümelerine de uygulanmaktadır.

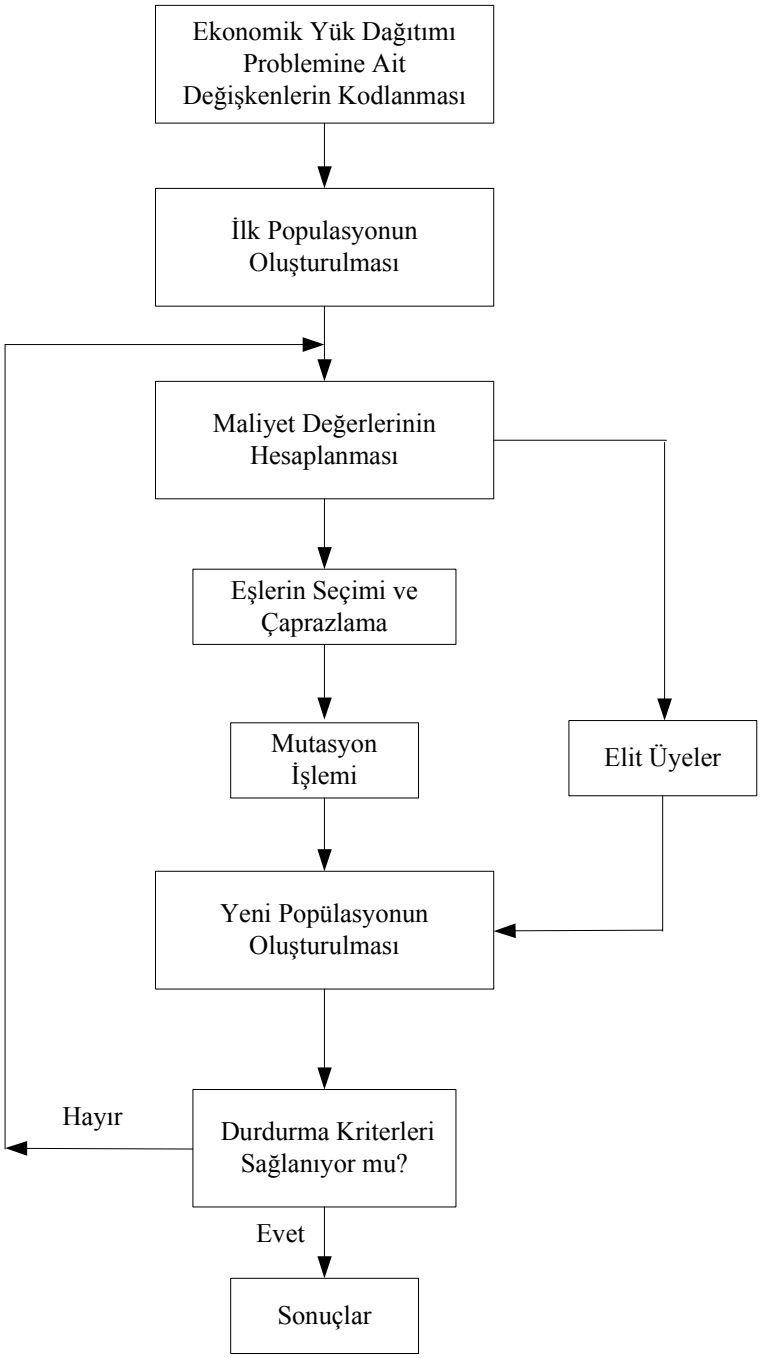


Şekil 6.4 Evrimsel programlama için mutasyon işleminin yapılması

Mutasyon işlemi sonucunda elimizde $2\times N_{pop}$ adedince bireye sahip oluruz. Elde edilen her bireyin uygunluk değeri maliyet değerine bağlı olarak belirlenir. Minimum değerden maksimum değere doğru maliyetler sıralanıp en iyi N_{pop} adet birey yeni popülasyon olarak alınır. Diğer bireyler ise elenir. Bu işlem durdurma kriteri sağlanana kadar devam etmektedir. Bu çalışmada durdurma kriteri iterasyon sayısı olarak belirlenmiştir. Farklı problemler için bu popülasyonun ortalama maliyet değerinin belli bir değere yakınsaması olarak da alınabilir veya problemin yapısına göre belirlenebilir.

Birim yüklenme probleminin iki ana kısımdan oluştuğu ifade edilmiş idi. İlk kısım sistemdeki

generatör birimlerinin hangilerinin devrede (ON), hangilerinin de devre dışında (OFF) olacağı durumlarının belirlenmesi idi. Evrimsel programla yöntemi ile bu durumları belirlemiş olmaktayız. İkinci kısım ise devrede olan generatörlerinin üretecekleri enerji miktarlarını optimum şekilde belirlemektir. Bu problemin çözümünde de genetik algoritma yönteminden yararlanılmıştır. Çalışmada kullanılan problemin çözümüne ait akış diyagramı Şekil 6.5'te verilmiştir.



Şekil 6.5 Ekonomik yük dağıtımı için uygulanan genetik algoritmanın akış şeması

Yapılan analizler ve çözümler sonucunda, böylece sistemin ekonomik işletimi için optimum maliyet değeri ile optimum değerleri elde edilen yük dağıtımı elde edilmiş olmaktadır. Çalışmada hedeflenen çözümün ilk aşaması bu şekilde gerçekleştirilmiştir.

6.2 Kararlılık Kriteri

Çalışmanın bu aşamasında, sistemin kararlılık açısından incelenmesinde V-Q duyarlılık endeksi kullanılmıştır. V-Q duyarlılık endeksi, farklı işletim noktaları için sistemin kararlılığı hakkında bize karşılaştırma yapma imkanı sağlamaktadır.

Böylece daha kararlı sistemlerin elde edilmesi mümkün olmaktadır. Bir işletim noktasında sistemin kararlılığını incelemek için öncelikle Bölüm 3'te tarif edilen, sisteme ait mevcut işletme durumundaki indirgenmiş Jakobiyen matrisini (J_R) elde ederiz. Bu matrisin tersi bize V-Q duyarlılık endeksinin değerini vermektedir [Kundur, 1994].

Kararlılık analizi yapmak için sistemdeki generatör birimlerinin ürettikleri güç miktarlarını, generatör güç limitleri içerisinde kalmak ve sistemden talep edilen güç miktarını da sağlamak koşulu ile değiştirerek farklı işletim noktaları elde ederiz. Her işletim noktası için yük akış analizini yaparak o noktaya ait V-Q duyarlılık endeksi değerini hesaplarız. V-Q duyarlılık endeksinin değerleri sistem hakkında bilgi vermektedir. Bir enerji sisteminde V-Q değerinin sıfır olması mükemmel bir kararlılık yapısını ifade etmektedir. Yapılan inceleme ile her işletim noktası için elde ettiğimiz V-Q duyarlılık değerlerini küçükten büyüğe doğru sıralarsak, minimum V-Q duyarlılık endeksine sahip nokta, sistem için en iyi kararlı işletim noktası olarak elde edilecektir.

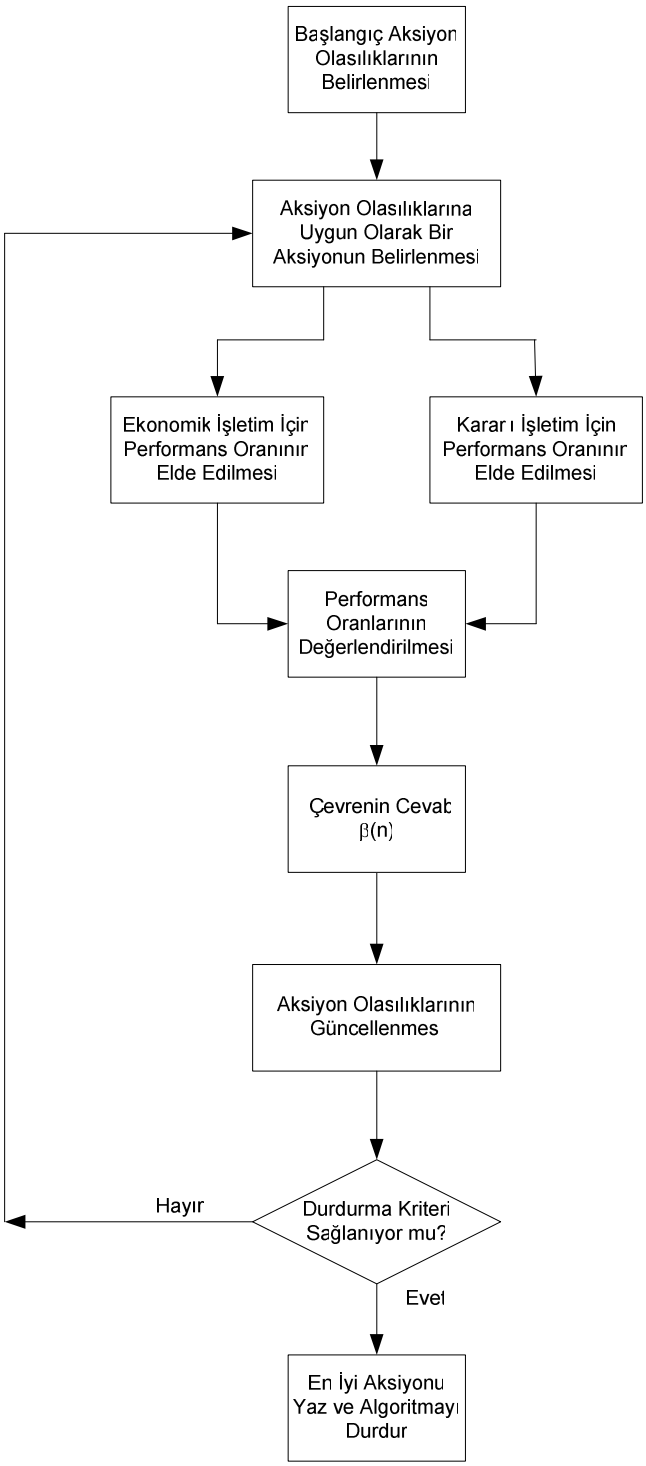
6.3 Optimum İşletim Noktasının Elde Edilmesi

Sistem için, Bölüm 6.1'den elde ettiğimiz en ekonomik işletim maliyet değerini $J_{\min}^{M}$ olarak ve Bölüm 6.2'de belirlediğimiz en uygun kararlılık kriterini de $J_{\min}^{K}$ olarak belirleyelim. Bu iki kriter arasında sistemin işletiminde optimum bir işletme noktası elde etmek için uygun eşik değerler kullanarak bir çözüm alanı belirleyebiliriz. Bu bölge içerisinde belirleyeceğimiz işletim noktalarının da ekonomik ve kararlılık anlamında kriter değerlerini elde ederiz.

Hem üretim maliyeti hem de kararlılık için iki adet performans oranı seçeriz ki;

- Birincisi, minimum üretim maiyetinin, herhangi bir *i*-işletim noktasındaki üretim maliyetine oranı ($J_{\min}^{M} / J_{i}^{M}$) olup bu değer, ekonomik işletim için %90 olarak belirlenmiştir.
- İkincisi, en uygun kararlılık indisinin, herhangi bir *i*-işletim anındaki kararlılık indisine oranı ($J_{\min}^{K} / J_{i}^{K}$) olup bu değer, kararlı bir işletim için %50 olarak belirlenmiştir.

Bu iki kriter için belirlemiş olduğumuz eşik değerler, karar verici tarafından belirlenmektedir. Bu iki performans oranı eşik değerler de göz önüne alınarak sıfır ile bir arasında sigmoid bir çıkış verecek şekilde normalize edilir. Daha sonra bu iki performans oranının toplamı değişken yapılı öğrenen automatanın çevresine bir giriş değeri olarak girilir ve çevreden gelen $\beta(n)$ cevabı, automatanın aksiyonlarının bu cevaba göre güncellenmesini ve en iyi aksiyonun bulunmasını sağlar. Performans oranlarının değeri verilen eşik değerlerden büyük olduğunda bu durum elverişli olacak ve $\beta(n)$ cevabı 0 olacaktır. Benzer olarak, performans oranlarının değeri eşik değerlerden küçük ise bu durum elverişli olmayıp $\beta(n)$ cevabı 1 olacaktır. Değişken yapılı öğrenen automatanın işleyişini ifade eden akış şeması Şekil 6.5'te verilmiştir.



Şekil 6.5 Çalışmada kullanılan öğrenen automatanın akış şeması

7. SAYISAL UYGULAMA

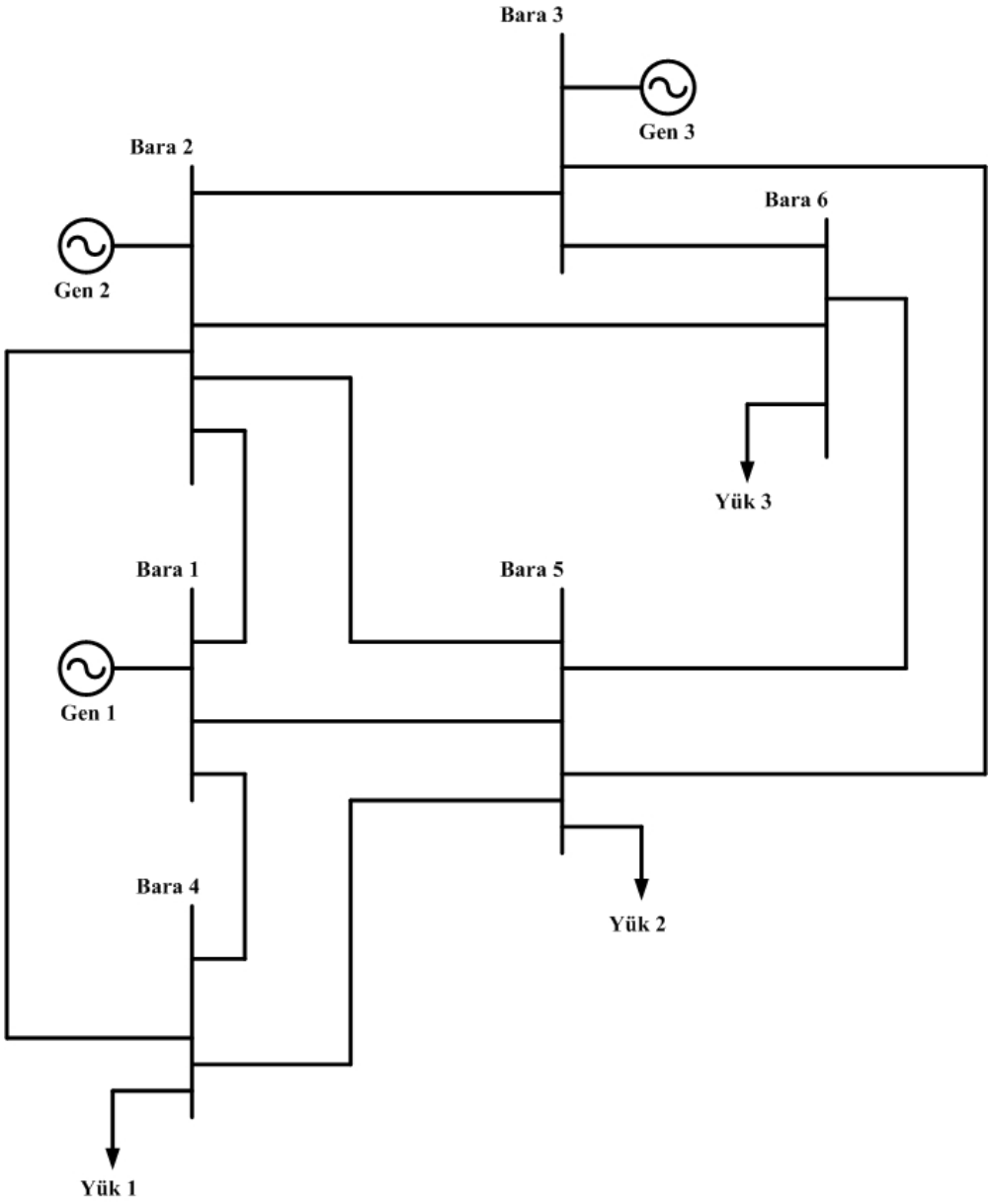
Tez çalışmasında önceki üçüncü, dördüncü ve beşinci bölümlerde anlatılan evrimsel programlama, genetik algoritma, öğrenen automata ve V-Q duyarlılık kriterinin kullanıldığı iki adet örnek uygulama gerçekleştirilmiştir. Örnek uygulamalarda, 6-baralı 3-generatör birimine sahip bir test sisteminin analizi için geliştirilen metot uygulanmıştır. Birinci uygulama bir saatlik ekonomik yük dağıtımı göz önüne alınarak yapılmıştır. İkinci uygulama ise 12 saatlik bir periyot için birim yüklenme problemi ele alınarak gerçeklenmiştir.

7.1 Sayısal Uygulama 1

Sayısal uygulamanın gerçekleştirilmesi için 6-bara ve 3-generatörden meydana gelen bir güç sistemi ele alınmıştır [Wood ve Wollenberg, 1996]. Elektrik enerji sistemine ait tek hat şeması Şekil 7.1'de verilmiştir. Sistemde, 100 MVA baz alınmış olup hattın karakteristik değerleri Tablo 7.1'de ve baralardaki yüklerin aktif ve reaktif güç değerleri de Tablo 7.2'de verilmiştir.

Tablo 7.1 Örnek Sisteme ait hat parametreleri

Hat No	No'lu Baradan	No'lu Baraya	Hat Parametreleri		
			R (pu)	X (pu)	B(pu)
1	1	2	0,10	0,20	0,020
2	1	4	0,05	0,20	0,020
3	1	5	0,08	0,30	0,030
4	2	3	0,05	0,25	0,030
5	2	4	0,05	0,10	0,010
6	2	5	0,10	0,30	0,020
7	2	6	0,07	0,20	0,025
8	3	5	0,12	0,26	0,025
9	3	6	0,02	0,10	0,010
10	4	5	0,20	0,40	0,040
11	5	6	0,10	0,30	0,030



Şekil 7.1 Sayısal uygulama için göz önüne alınan örnek enerji sistemi

Tablo 7.2 Baralardaki yüklerin aktif ve reaktif güçleri

Bara No	$P_{yük}$ (MW)	$Q_{yük}$ (MVAr)
4	70	70
5	70	70
6	70	70

Sistemde 3 adet generatör bulunduğundan her bir üretim birimine ait maliyet fonksiyonları,

$$F_1(P_1) = 400{,}0 + 7{,}5 \times P_1 + 0{,}004 \times P_1^2$$

$$F_2(P_2) = 200{,}0 + 4{,}0 \times P_2 + 0{,}009 \times P_2^2$$

$$F_3(P_3) = 350{,}0 + 6{,}5 \times P_3 + 0{,}007 \times P_3^2$$

ve üretim birimlerinin güç limitleri

$$50{,}0\,\text{MW} \le P_1 \le 200\,\text{MW}$$

$$37{,}5\,\text{MW} \le P_2 \le 150\,\text{MW}$$

$$45{,}0\,\text{MW} \le P_3 \le 180\,\text{MW}$$

olarak verilmiştir.

Bu uygulamada, öncelikle sistemdeki bütün generatörlerin devrede olması durumu için ekonomik yük dağıtım problemi, 210 MW'lık talep yükü ve buna ek olarak 10 MW'lık yedek güce bağlı olarak çözülmüştür. Problemin çözümünde genetik algoritma kullanılmıştır. Yapılan bir dizi denemeler sonucunda elde edilen maliyet değerleri, Tablo 7.3'de görüldüğü üzere farklı popülasyon ve mutasyon değerleri için verilmiştir.

Tablo 7.3 Maliyet değerinin farklı popülasyon ve mutasyon değerleri için değişimi

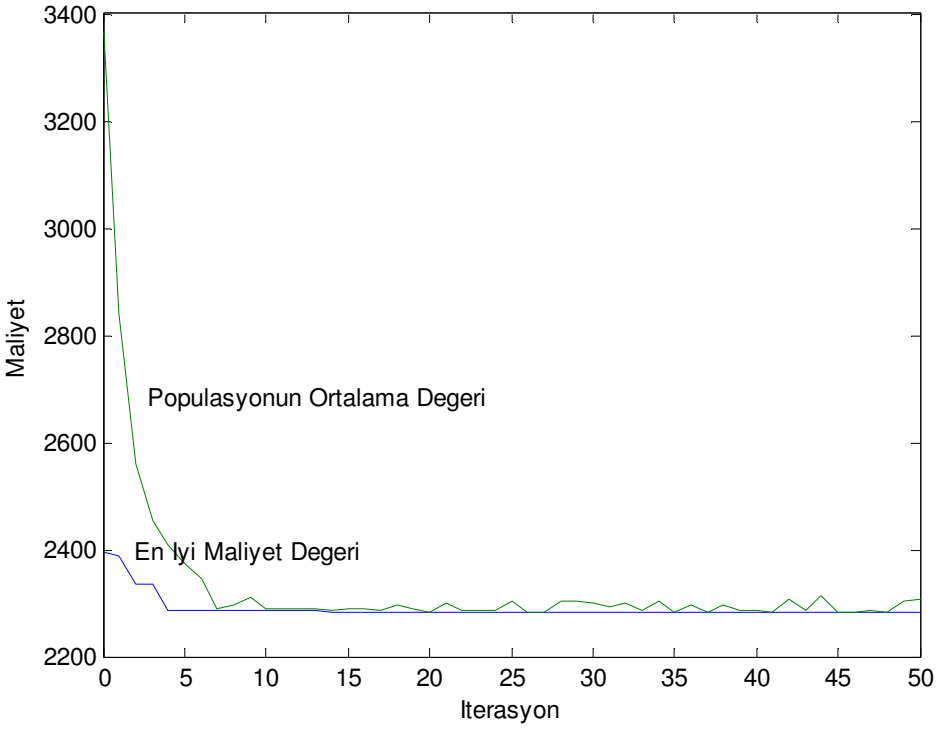
Popülasyon Değeri	Mutasyon Değeri		
	0,1	0,01	0,001
20	2285,53	2284,57	2282,83
30	2282,87	2282,37	2283,97
40	2282,64	2282,3	2282,81

Bu uygulama için popülasyon değeri 40 ve mutasyon değeri de 0,01 olarak belirlenmiştir. Şekil 7.2'de en iyi maliyet değeri ile popülasyona ait ortalama maliyet değerinin iterasyona

bağlı olarak değişimi verilmiştir. En iyi maliyet değerini veren güç değerleri Tablo 7.4'de görülmektedir.

Tablo 7.4 Genetik algoritma kullanılarak elde edilen optimum maliyet ve optimum güç dağılımları

Üretim Maliyeti ($)	Güç Değerleri		
	P_1 (MW)	P_2 (MW)	P_3 (MW)
2282,3	50	125	45



Şekil 7.2 Genetik algoritma ile elde edilen maliyet değerlerinin iterasyona bağlı değişimi

Ekonomik işletim için en uygun maliyet değerini ve bu değeri veren güç değerlerini bulduktan sonra, sıra sistemin kararlı işletimi için en uygun değerin bulunmasına gelmektedir. Bunun için birimlerin ürettiği güç değerlerini değiştirerek yük akışı analizi yaparız ve sistemin Jakobiyen matrisini elde edip V-Q duyarlılık kriteri değerlerine bakarız. Burada birimlerin güç değerlerini değiştirirken, minimum ve maksimum güç limitlerini aşmadan, talep yük ile yedek gücün karşılanmasına da dikkat ederiz. Yük akışı yaparken bir birimi salınım barası

olarak belirleriz, geri kalan birimlerden birinin ürettiği güç değerlerini değiştirirken diğerlerinin güç miktarlarını ekonomik yük dağılımında elde edilen değerlerinde sabit tutarız. Elde ettiğimiz her farklı durum için V-Q duyarlılık kriterini inceleyerek sistem için en kararlı cevabı veren durumu elde ederiz. Böylece ekonomik işletim ve kararlı işletim anlamında, sistem için ayrı ayrı en optimum sonuç veren iki adet nokta elde etmiş oluruz. Bu iki nokta arasında farklı işletim durumları elde ederek, sistem için bu iki farklı optimizasyon probleminin çözümünü belli eşik değerleri içerisinde sağlayan optimum bir nokta elde ederiz. Bu nokta sistemin hem ekonomik hem de kararlı olarak işletimde olacağı durumdur. Öğrenen automata yöntemi bu noktanın bulunmasında kullanılacaktır. Bunun için öncelikle aksiyonların belirlenmesi gerekmektedir.

Tablo 7.5'te öğrenen automata için farklı aksiyon durumlarını göstermektedir. 1 nolu aksiyon sistemin kararlılık anlamında verdiği en iyi cevaptır. 10 nolu aksiyon ise genetik algoritma yöntemi kullanılarak elde edilen en ekonomik maliyete sahip işletme noktasıdır. Bu iki nokta arasındaki işletim bölgelerinden farklı durumlar elde edilmiştir. Bunun için ilk olarak 1 nolu generatör birimi salınım barası olarak seçilmiştir ve değeri yük akışı analizi sonucunda belirlenecektir. İlk olarak 3 nolu generatörün güç değeri sabit tutularak 2 nolu generatörün güç değeri değiştirilmiş, ardından 2 nolu generatör değeri sabit tutularak 3 nolu generatörün değeri değiştirilmiştir. Güç değişimleri yapılırken generatörlerin alabileceği minimum ve maksimum değerlerin arası eşit parçalara ayrılarak farklı durumlar elde edilmiştir. En uygun değerler aksiyonlar olarak belirlenmiştir.

Tablo 7.5 Hem ekonomik hem de kararlı çalışma için kullanılan öğrenen automatada aksiyonların belirlenmesi

Generatör Güçleri (MW)	Aksiyonlar									
	1	2	3	4	5	6	7	8	9	10
P_2	45	55	65	70	85	100	105	115	120	125
P_3	45	45	45	45	45	45	45	45	45	45

Bu uygulama için 3 nolu generatörün değeri 45 MW olarak belirlenmiş, bu generatör için diğer farklı güç değerlerinin istenen optimum kriterleri sağlamadığı görülmüştür. 2 nolu

generatörün değerleri de Tablo 7.5'te görüldüğü üzere belirlenmiştir. Öğrenen automata yöntemi için 10 adet aksiyon seçilmiştir. Aksiyonlar belirlendikten sonra öğrenen automata analizleri üç farklı öğrenme metodu için tekrarlanmıştır. Bunlar sırası ile L_{R-I}, L_{R-P} ve $L_{R-\varepsilon P}$ pekiştirme düzenleridir. Şimdi bu öğrenme durumları için tek tek analizleri elde edelim.

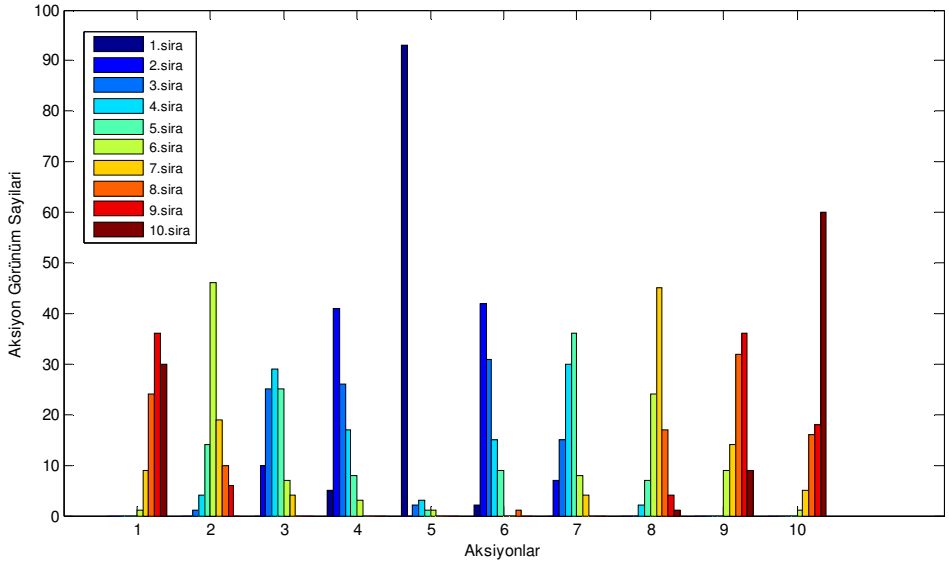
1.Durum

Öğrenme Metodu: L_{R-I}

L_{R-I} pekiştirme yapısına sahip öğrenme planı, çevreden gelen bir ödül cevabında aksiyon olasılıklarının *a* ödül katsayısına bağlı olarak güncellenmesi, bir ceza cevabında ise herhangi bir aksiyon olasılık değerinin güncellenmemesi durumudur. Bu duruma ait *a* ve *b* parametrelerinin değeri aşağıdaki gibidir.

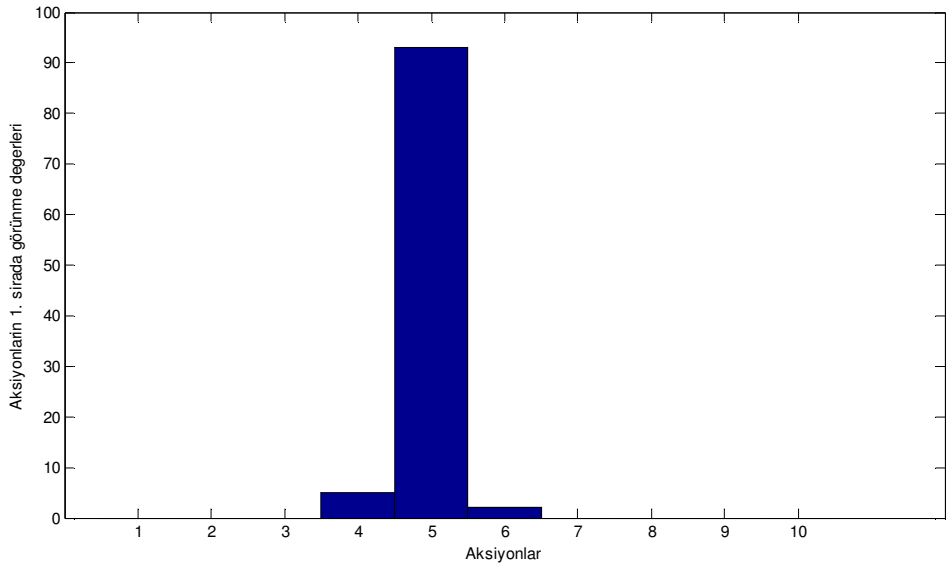
Öğrenme Parametreleri: $a = 0.015$, $b = 0$

İlk olarak 100 adet öğrenme işlemi (deney) yapılarak aksiyonların çevreye bağlı olarak almış oldukları sıralamalar elde edilmiştir. Her öğrenme işleminin iterasyon adedi 5000 olarak belirlenmiştir. Şekil 7.3'te aksiyonların yapılan deneyler sonucunda en iyiden en kötüye doğru sıralanmaları sonucu elde ettikleri yerlerin değerleri görülmektedir.



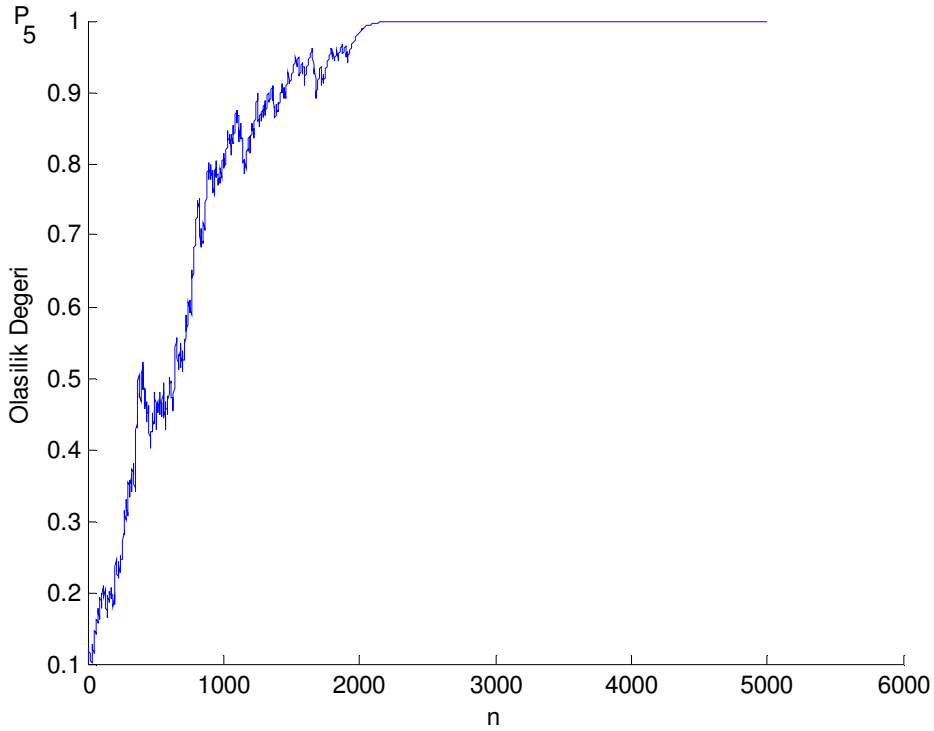
Şekil 7.3 Öğrenen automata ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri

Şekil 7.4'te ise Şekil 7.3'ün özel bir görünümü verilmiştir. Burada yapılan her öğrenme işlemi için en iyi gelen birinci aksiyon ve kaç defa en iyi sırada geldiği görülmektedir.

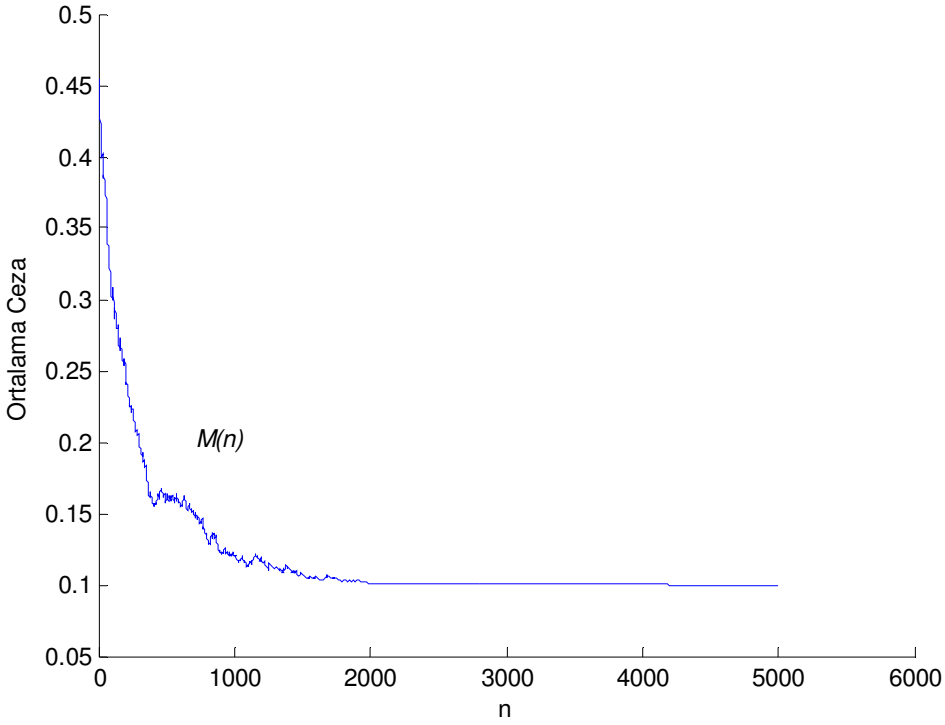


Şekil 7.4 Öğrenen automata kullanımında, aksiyonların kaç defa birinci sırada göründüklerinin değeri

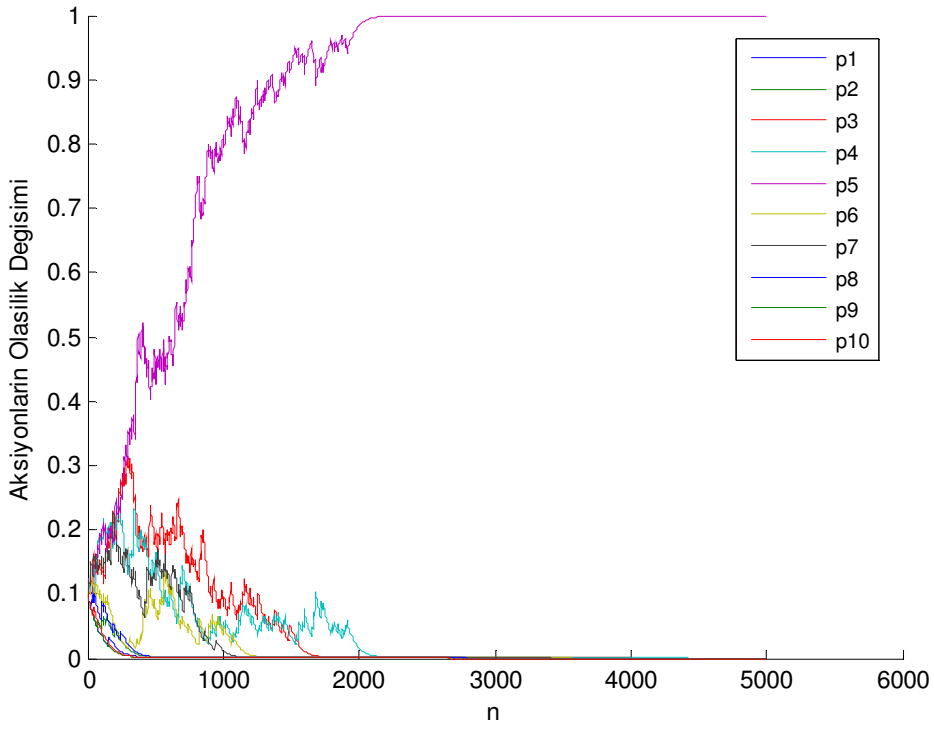
Şekil 7.5'te aksiyonlar içerisinden en iyi aksiyonun, 5 nolu aksiyon olduğu ve aksiyon olasılığının iterasyona bağlı olarak 1'e yakınsadığı görülmektedir.



Şekil 7.5 Öğrenen automata kullanımında, 5. aksiyona ait olasılık değerinin iterasyona bağlı değişimi



Şekil 7.6 LA için ortalama ceza değerinin değişimi



Şekil 7.7 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması

Şekil 7.6'da ortalama ceza değerinin iterasyona bağlı değişimi görülmektedir. *M(n)*, en iyi

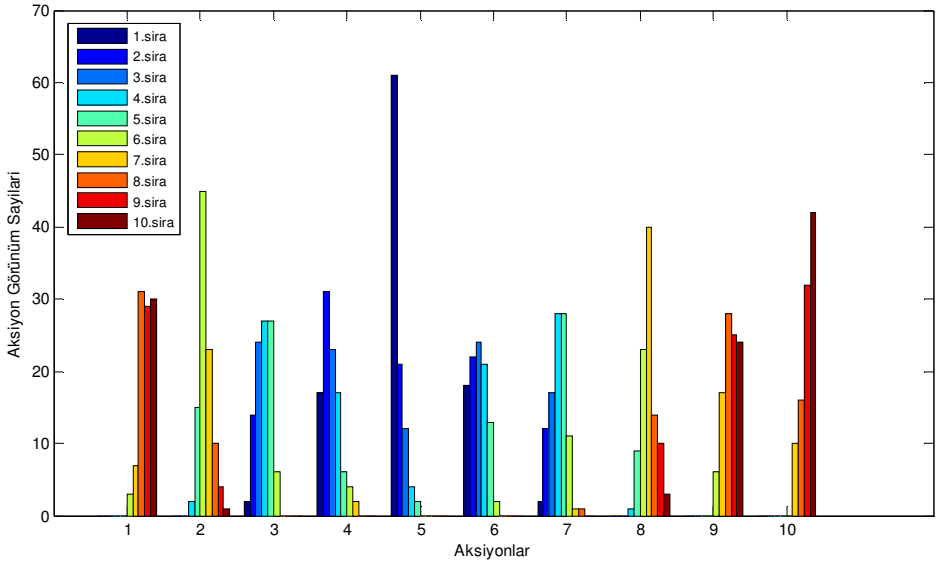
aksiyonun ceza olasılığı olan 0.1 değerine gelmektedir. Şekil 7.7'de ise bütün aksiyonların aksiyon olasılıkları değişimi görülmektedir. 5 nolu aksiyonun olasılığı 1 değerine yakınsarken diğer aksiyon olasılıkları 0 değerine düşmektedir. Dolayısıyla en iyi sonucun, sistem için optimum işletim noktasının bu aksiyon ile elde edildiği görülmektedir

2.Durum

Öğrenme Metodu: L_{R-P}

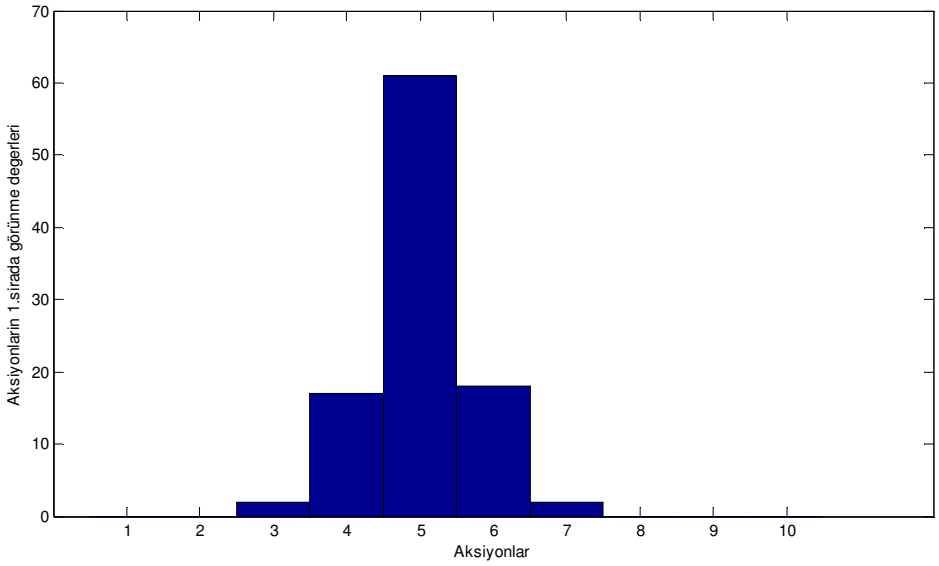
L_{R-P} öğrenme planı, ödül (a) ve ceza (b) parametrelerinin birbirine eşit olduğu pekiştirme düzeneğidir. Bu durumda aksiyon olasılıklarının güncellenme oranı, çevrenin vereceği cevap ne olursa olsun her adımda aynı kalmaktadır. Bu duruma ait a ve b parametrelerinin değeri aşağıdaki gibidir.

Öğrenme Parametreleri: a = 0.015 , b = 0.015

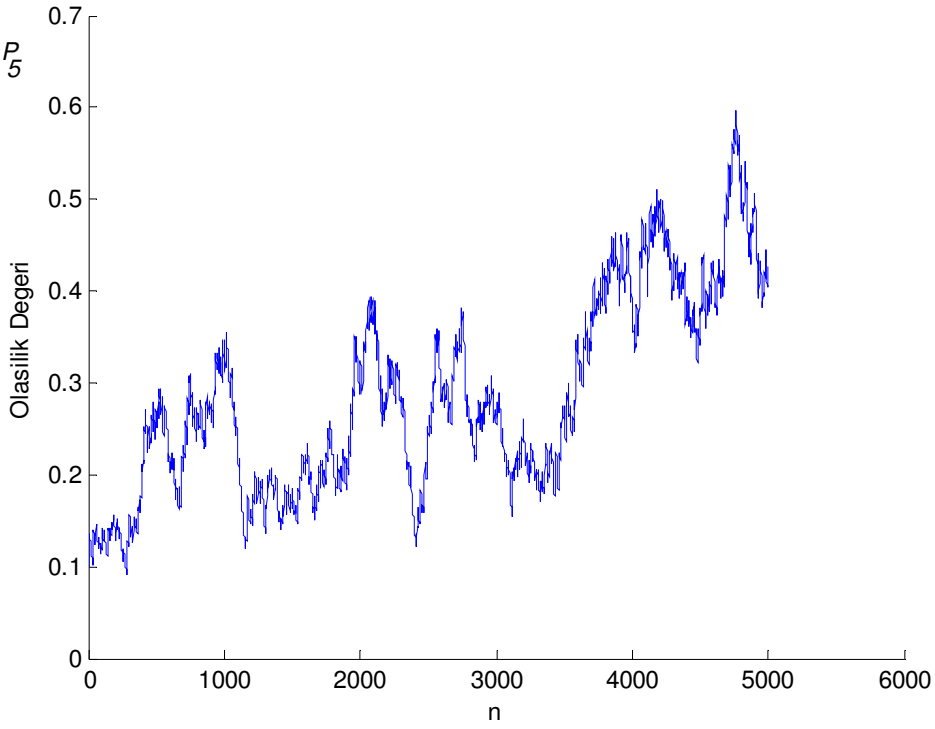


Şekil 7.8 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri

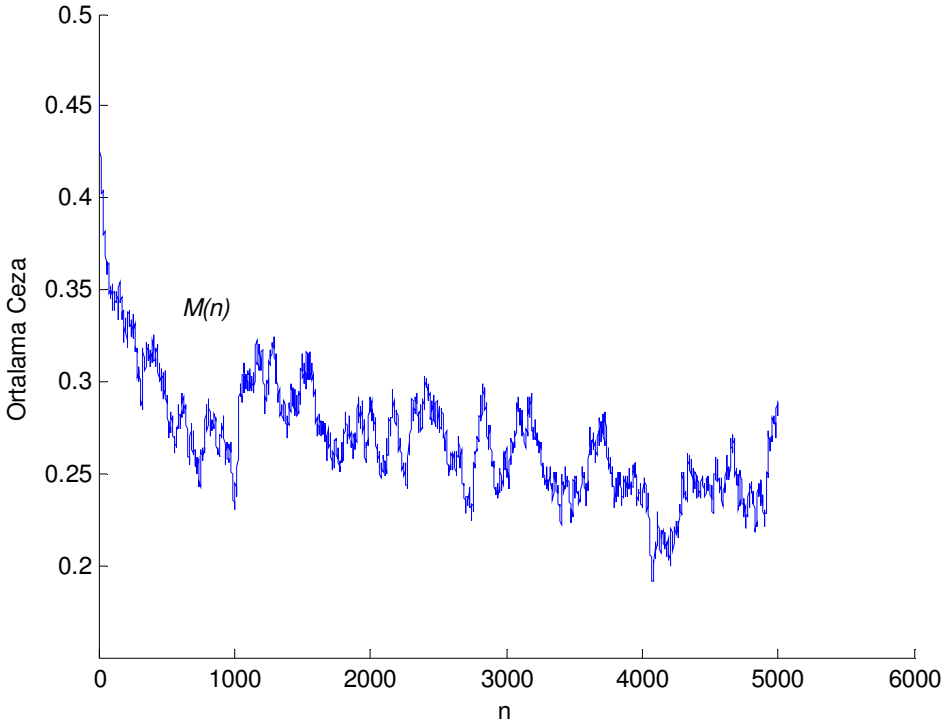
Şekil 7.8'de L_{R-P} pekiştirme düzeneğinin kullanıldığı öğrenen automata için yapılan öğrenme işlemleri sonucunda, aksiyonların en iyiden en kötüye doğru sıralanmaları sonucu elde ettikleri yerlerin değerleri görülmektedir. Şekil 7.9'da ise her öğrenme işlemi için en iyi gelen birinci aksiyon ve kaç defa en iyi sırada geldiği görülmektedir. Şekil 7.10'da en iyi aksiyon olan 5 numaralı aksiyonun olasılık değişimi verilmiştir.



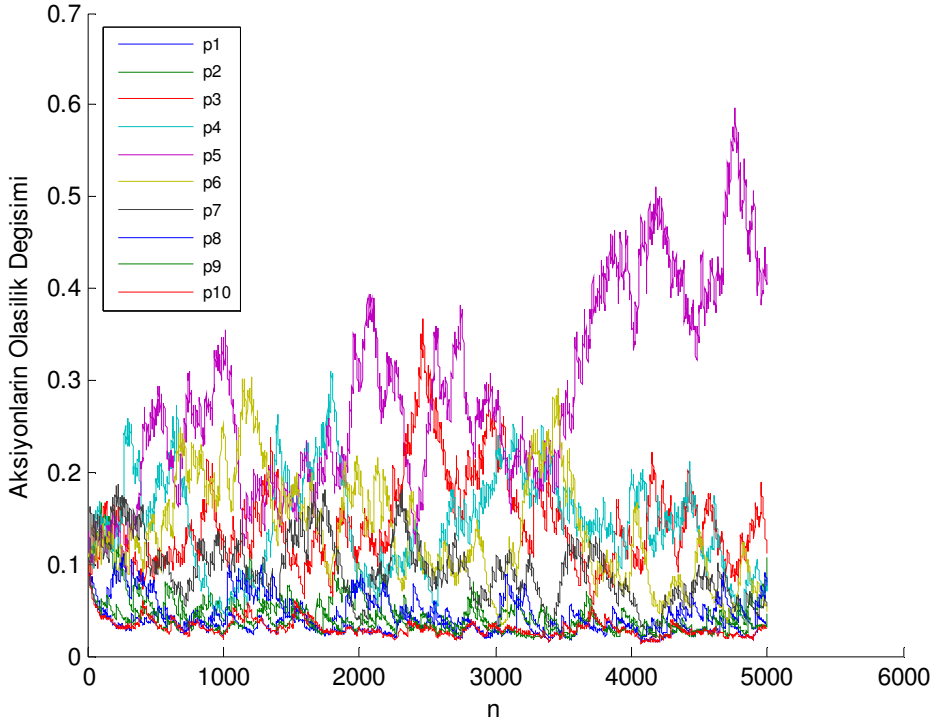
Şekil 7.9 LA kullanımında aksiyonların kaç defa birinci sırada göründüklerinin değeri



Şekil 7.10 LA çözümünde 5. aksiyona ait olasılık değerinin iterasyona bağlı değişimi



Şekil 7.11 LA için ortalama ceza değerinin değişimi



Şekil 7.12 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması

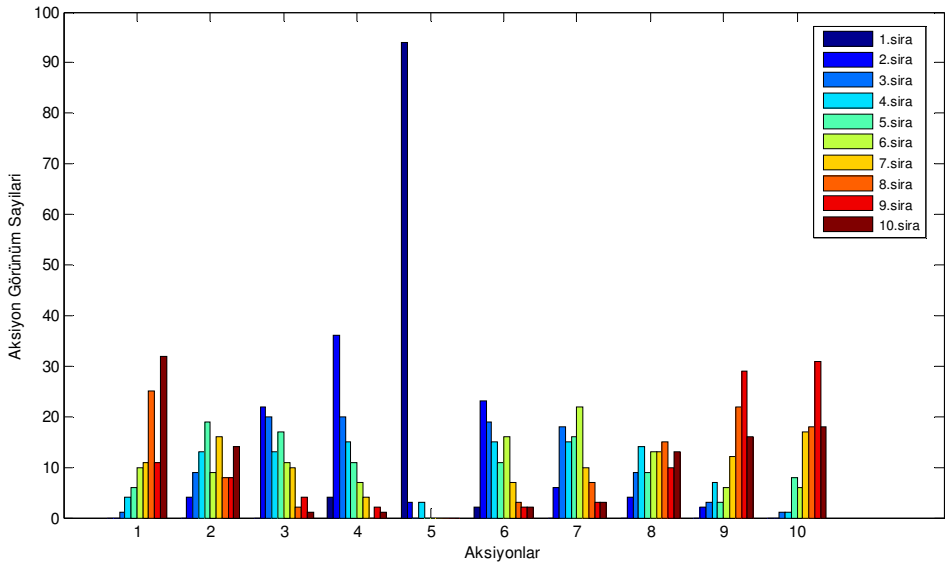
Şekil 7.11'de L_{R-P} pekiştirme düzeni için ortalama cezanın *M(n)* değişimi, Şekil 7.12'de ise tüm aksiyonların aksiyon olasılıkları (*p*) değişimi görülmektedir.

3.Durum

Öğrenme Metodu: $L_{R-\varepsilon P}$

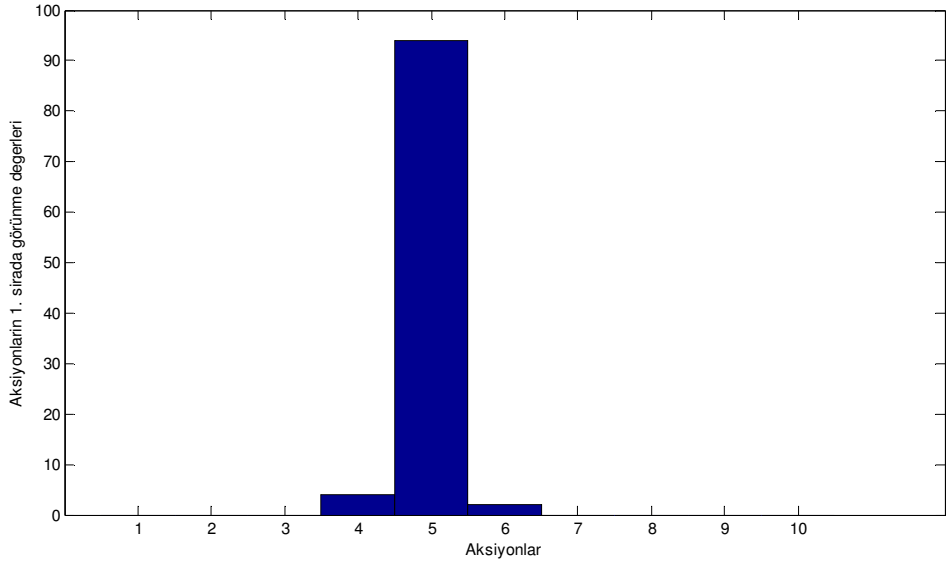
L_{R-EP} öğrenme planında, ödül (*a*) ve ceza (*b*) parametreleri birbirine eşit olmayıp sıfır değerinden de farklıdır ve aralarında *a>b* ilişkisi bulunmaktadır. Aksiyon olasılıkları (*p*), bu orana bağlı kalarak güncellenmektedir. Bu uygulama için *a* ve *b* parametrelerinin değeri aşağıdaki gibidir.

Öğrenme Parametreleri: a = 0.015 , b = 0.0015

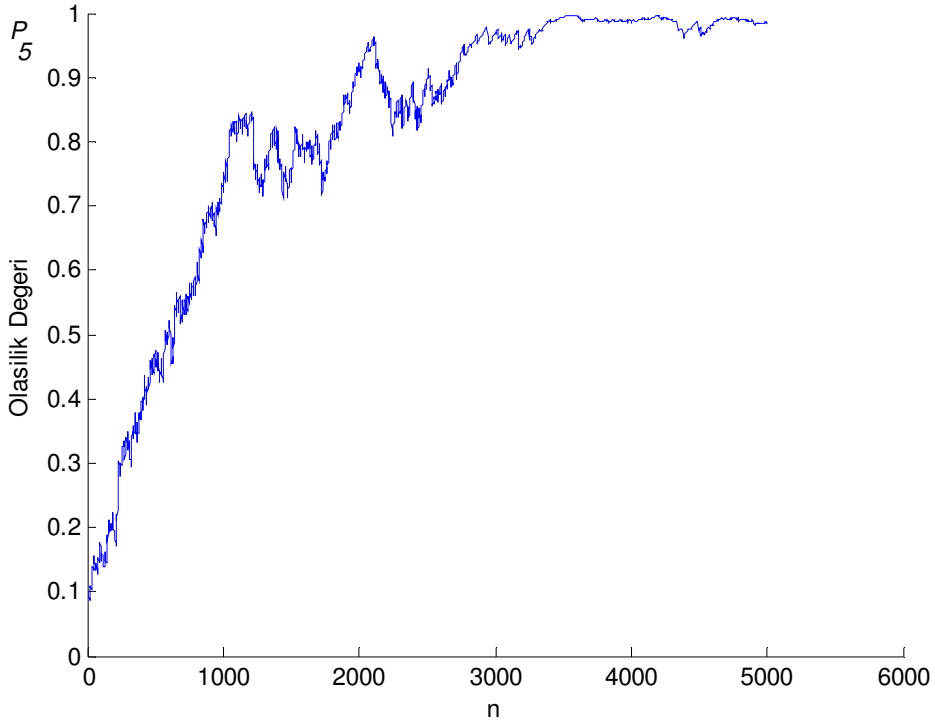


Şekil 7.13 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri

Şekil 7.13'de $L_{R-\varepsilon P}$ pekiştirme düzeneğinin kullanıldığı öğrenen automata için yapılan öğrenme işlemleri sonucunda, aksiyonların en iyiden en kötüye doğru sıralanmaları sonucu elde ettikleri yerlerin değerleri ve Şekil 7.14'de ise her öğrenme işlemi için en iyi gelen birinci aksiyon ve kaç defa en iyi sırada geldiği görülmektedir.

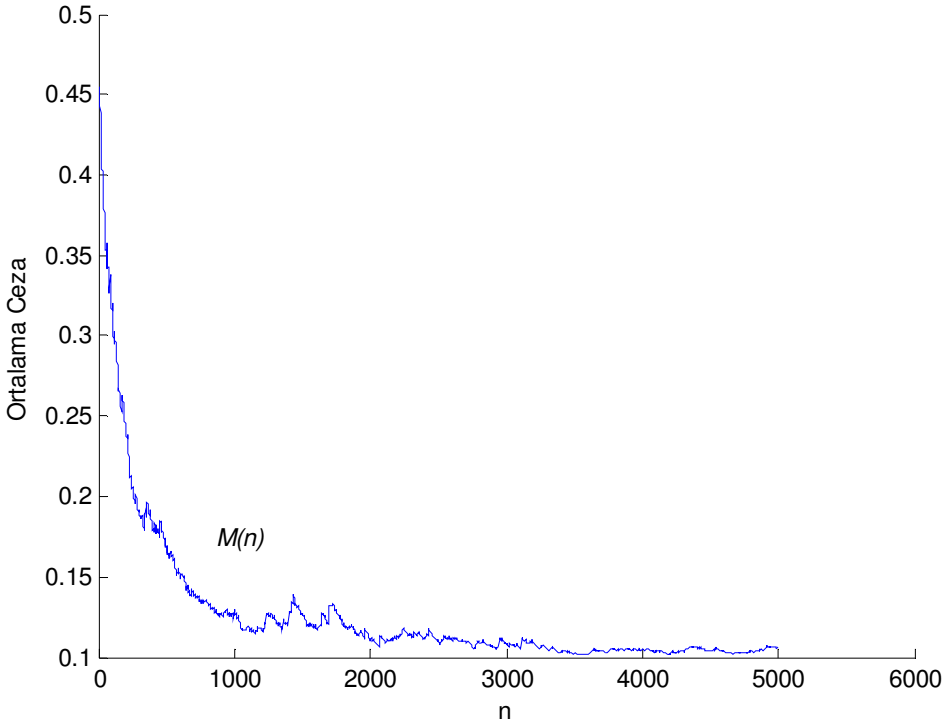


Şekil 7.14 LA kullanımında aksiyonların kaç defa birinci sırada göründüklerinin değeri

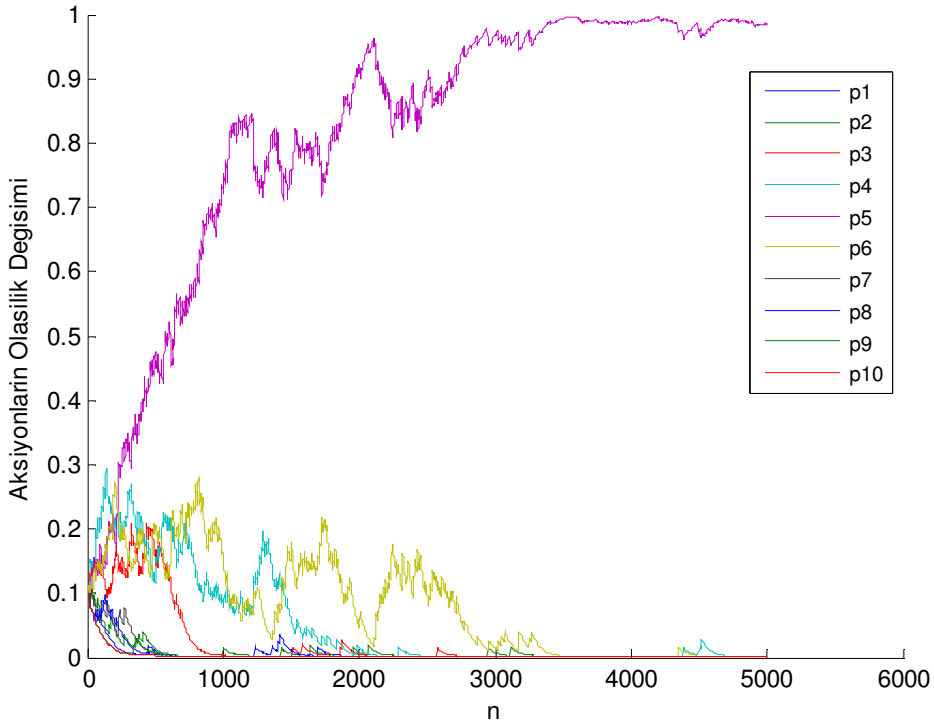


Şekil 7.15 LA çözümünde 5. aksiyona ait olasılık değerinin iterasyona bağlı değişimi

Şekil 7.15'de en iyi aksiyon olan 5 numaralı aksiyonun olasılık değişimi verilmiştir. $L_{R-\varepsilon P}$ pekiştirme düzeneği için ortalama ceza *M(n)*'in değişimi Şekil 7.16'te görülmektedir.



Şekil 7.16 LA için ortalama ceza değerinin değişimi



Şekil 7.17 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması

Yapılan deneyler ve analizler sonucunda 5 numaralı aksiyonun en iyi aksiyon olduğu belirlenmiştir. Böylece 2 ve 3 numaralı generatörlerin üretecekleri güç değerleri (P_2=85 MW, P_3=45 MW olarak) belirlenmiş olur. Sıra salınım barası olarak seçilmiş olan 1 numaralı generatörün değerini elde etmeye gelmektedir. Bunun için de yük akış analizi yapılarak sisteme ait birimlerin üreteceği güç miktarı (P_1=90 MW olarak) belirlenmiş olacaktır.

Yük akışı analizi sonucunda elde edilen değerler ve en iyi aksiyon numarası Tablo 7.6'da verilmiştir. Bu değerlere sahip olan durum, sistem için en önemli kriterlerden ekonomik maliyetli işletimi ve kararlı işletimi istenen düzeyde birlikte sağlayan işletim noktası olarak elde edilmiş olunmaktadır. Bu durum için hesaplanan maliyet değeri ise 2369,1 $ olarak belirlenmiştir.

Tablo 7.6 Öğrenen automata kullanımı ile belirlenen en iyi aksiyon ve güç değerleri

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
5	90	85	45

İncelenen üç durumun karşılaştırılması:

Öğrenen automata yöntemi için kullanmış olduğumuz üç farklı pekiştirme düzeneğini kıyasladığımızda şu sonuçları elde ederiz: Üç durum için de en iyi aksiyon, 5 numaralı aksiyon olarak belirlenmiştir. L_{R-I} pekiştirme düzeneğinde en iyi aksiyonun olasılık değeri çok hızlı bir şekilde 1'e otururken farklı durumları absorbe edebilmektedir. L_{R-P}, geniş salınımlara sahip en yavaş yakınsayan öğrenme planıdır. $L_{R-\varepsilon P}$ ise diğer pekiştirme düzenekleri ile kıyaslandığında herhangi bir durumu absorbe etmeyen ve yakınsama hızında da oldukça iyi olan öğrenme planıdır.

7.2 Sayısal Uygulama 2

Bu uygulamada 12 saatlik bir işletme dilimi için birim yüklenme probleminin analizi tezde ortaya konan metoda uygun olarak gerçekleştirilmiştir. Birim yüklenme problemi evrimsel programlama yöntemi kullanılarak çözülmüştür. Diğer uygulamada olduğu gibi bu uygulamada da ekonomik yük dağıtımı genetik algoritma ile elde edilmiştir. Sistemin farklı işletim durumlarını kararlılık anlamında kıyaslayabilmek için yine V-Q duyarlılık analizi bu uygulamada da kullanılmıştır. Öğrenen automata yöntemi ile de sistemin hem ekonomik hem de kararlı işletimi için belli kriterler dahilinde optimum işletme noktası elde edilmiştir.

Bu uygulama için bir önceki uygulamadaki test sistemi göz önüne alınmıştır. T = 12 saatlik bir zaman periyodu için birim yüklenme problemini evrimsel programlama tekniğini kullanarak çözelim. Generatörlerin devrede veya devre dışında olması durumunu evrimsel programla ile belirledikten sonra birimlere ekonomik yük dağıtımını bir önceki uygulamadaki gibi genetik algoritma yöntemi ile gerçekleyeceğiz. Genetik algoritma parametreleri birinci uygulamadaki gibi alınacaktır.

Birim yüklenme probleminin çözümü için gerekli verilerin belirlenmesi gerekmektedir. Bunun için, Tablo 7.7'de sistemdeki generatör birimlerine ait veriler ve kısıtlar verilmiştir. Yine Tablo 7.9'da sistem tarafından sağlanması istenen talep yük ve yedek güç değerleri verilmiştir. Tablo 7.8'de ise sistemdeki baralardan beslenen yüklerin, tüm talep yükü üzerinden dağılım profili oransal olarak gösterilmektedir.

Tablo 7.7 Sistemdeki generatörlerin verileri

Bara	P_{min}	P_{max}	Başlatma Maliyeti ($)	Durdurma Maliyeti ($)	Maliyet Katsayıları			Min up/down süreleri (h)	
					a ($)	b ($/MWh)	c ($/MW^{2}h)		
1	50	200	250	0	400	7,5	0,004	3	3
2	37,5	150	200	0	200	4	0,009	3	3
3	45	180	175	0	350	6,5	0,007	3	3

Tablo 7.8 Sistemdeki yüklerin dağılım profili

Bara	Toplam Yükün Oranı (%)
4	20
5	40
6	40

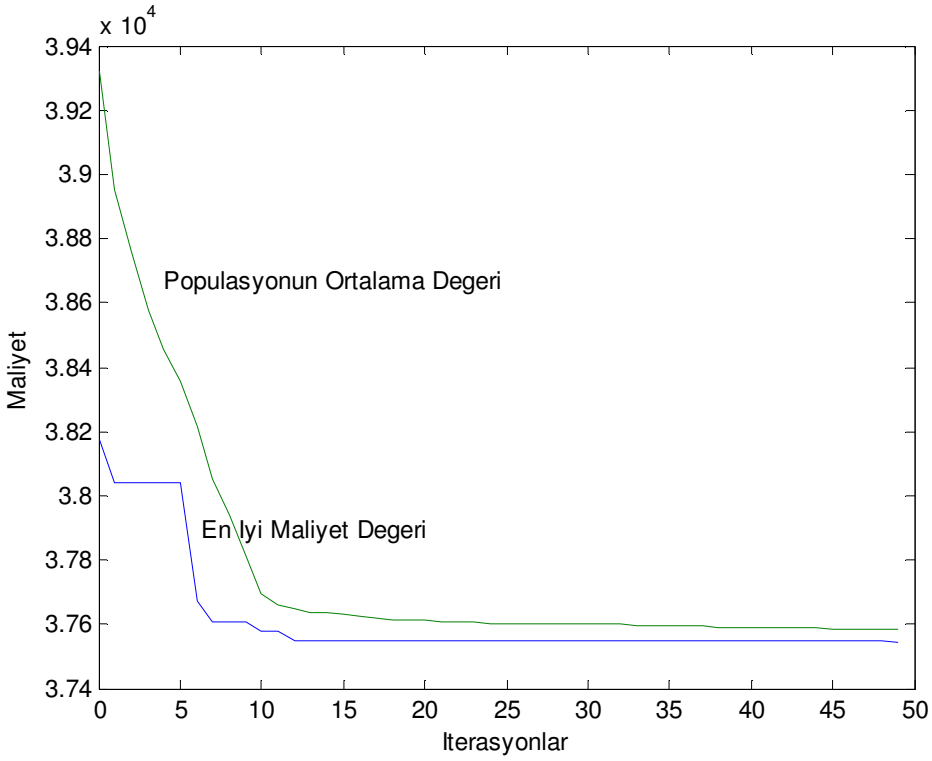
Tablo 7.9 İncelenen 12 Saatlik çalışma dilimi için tahmini yük ve yedek güç değerleri

Saat (t)	Tahmini Yük (MW)	Yedek Güç (MW)	Saat (t)	Tahmini Yük (MW)	Yedek Güç (MW)
1	145	5	7	346	29
2	191	9	8	366	34
3	237	13	9	385	40
4	258	17	10	405	45
5	280	20	11	423	52
6	325	25	12	441	59

Evrimsel programlama kullanarak birim yüklenme probleminin çözümü için yapılan analizlerde öncelikli olarak farklı popülasyon değerleri için elde edilen maliyet değerleri Tablo 7.10'da verilmiştir. Evrimsel programlama çözümü için popülasyon değeri 40 olarak belirlenmiştir. Bu popülasyon değeri için en iyi maliyet değerinin ve popülasyonun ortalama maliyet değerinin iterasyon ile değişimi Şekil 7.18'de verilmiştir.

Tablo 7.10 Evrimsel programlama çözümü için popülasyon sayısı ile maliyet değerinin değişimi

Popülasyon Değeri	Maliyet Değeri ($)
10	37861,50
20	37764,17
30	37505,82
40	37466,01
50	37506,52



Şekil 7.18 Evrimsel programlama çözümünde maliyet değerinin iterasyon ile değişimi

En ekonomik maliyeti elde etmek için evrimsel algoritma kullanılarak öncelikle sistemdeki generatör birimlerinin on/off durumları belirlenmiş olup, ardından da genetik algoritma kullanılarak devrede bulunan generatörlerin ne kadar güç üreteceği hesaplanmıştır. Tablo 7.11'de 12 saatlik periyot içinde birimlerin on/off durumları, Tablo 7.12'de ise ne kadar güç

üretecekleri gösterilmiştir.

Tablo 7.11 Evrimsel programlama kullanımı sonucunda generatör birimlerinin on/off durumları (birim yüklenme matrisi)

Birimler	t (Saat)											
	1	2	3	4	5	6	7	8	9	10	11	12
Gen.1	0	0	0	0	0	1	1	1	1	1	1	1
Gen.2	1	1	1	1	1	1	1	1	1	1	1	1
Gen.3	0	1	1	1	1	1	1	1	1	1	1	1

Tablo 7.12 Genetik algoritma sonucu generatör birimlerinin üretecekleri güç değerleri

Birimler	t (Saat) , P(MW)											
	1	2	3	4	5	6	7	8	9	10	11	12
Gen.1	0	0	0	0	0	80	95	95	125	140	155	170
Gen.2	150	138	138	149	149	144	147	145	150	144	149	150
Gen.3	0	62	112	126	151	126	133	160	150	166	171	180

Böylece her *t* saati için sistemin en ekonomik işletim noktası belirlenmiş olmaktadır. Şimdi bir önceki uygulamada yapılmış olduğu gibi öğrenen automata için aksiyonların belirlenmesi gerekmektedir. Aksiyonların belirlenmesi her *t* saati göz önüne alınarak ayrı ayrı yapılacaktır. Bunun için ilk yapılacak olan V-Q duyarlılık analizi kullanılarak her *t* için sistemin en kararlı işletim noktalarının belirlenmesi olacaktır. Ondan sonra, ekonomik ve kararlılık olmak üzere bu iki farklı optimum işletim noktaları arasında elde edilecek durumlar aksiyonlar olarak belirlenecek ve öğrenen automata kullanılarak en iyi aksiyon seçilmiş olacaktır. Öğrenen automata çözümü için aksiyon sayısı 5 adet olarak belirlenmiştir.

Bu uygulamada öğrenen automata için bir önceki uygulamada incelenen üç farklı pekiştirme düzeneğinden $L_{R-\varepsilon P}$ öğrenme planı kullanılacaktır. Aksiyon olasılıkları bu yapıya göre

güncellenecektir. Güncelleme sırasında ödül ve ceza parametreleri; a = 0.015, b= 0.0015 olarak seçilmiştir. Analizler her t saati için gerçeklenmiştir.

t = 1 saati için;

bu durumda sadece 2 nolu generatör devrededir ve maksimum güç limiti olan 150 MW gücünde işletmede bulunmaktadır. Bu değer sistem için en iyi ekonomik ve kararlılık değerlerini sağlamaktadır. Bundan dolayı bu değer direkt alınmıştır

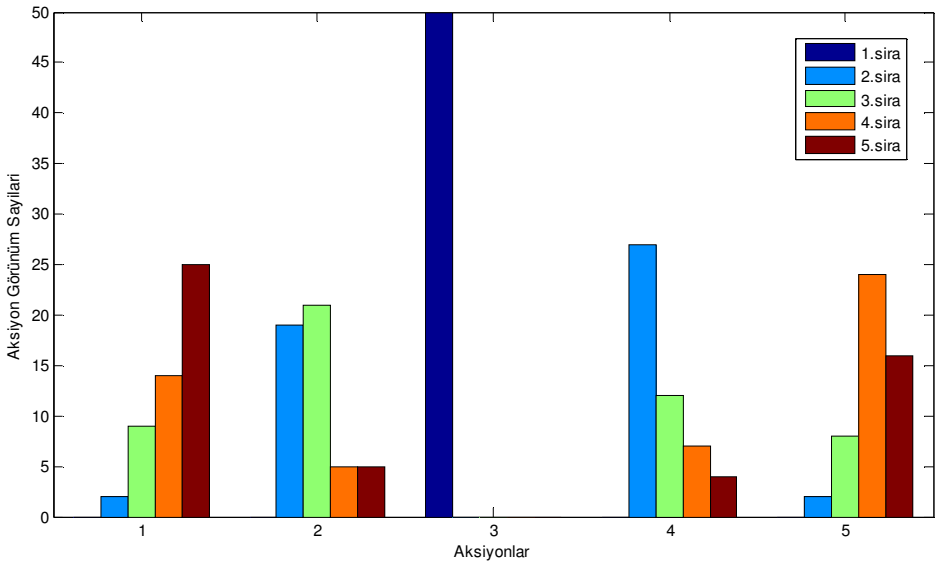
t = 2 saati için;

2 ve 3 nolu generatörler devrede olup, 1 nolu generatör devre dışındadır. 2 nolu generatör salınım barası olarak seçilmiş ve 3 nolu generatörün güç değerleri limitler içerisinde kalmak üzere ve sistem yük talep dengesi de sağlanmak üzere farklı değerlerde alınmıştır. 2 nolu generatörün değerleri yük akış analizi sonucunda belirlenmektedir. Tablo 7.6'da generatörlerin on/off durumu ve aksiyonların belirlenen değerleri verilmiştir.

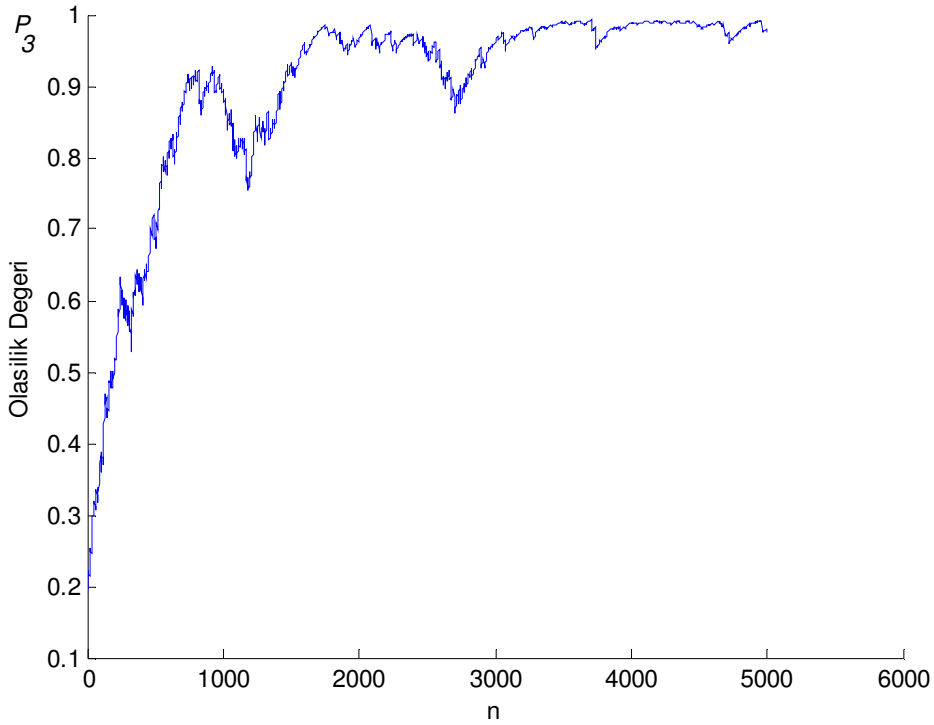
Tablo 7.13 Öğrenen automata için aksiyonların belirlenmesi (t = 2)

t (Saat)	2				
Generatörlerin Durumu	Generatör 1	Generatör 2	Generatör 3		
	Devre Dışı (0)	Devrede (1)	Devrede (1)		
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_3	62	70	100	110	120

Öğrenen automata için ilk olarak 50 adet deney yapılarak aksiyonların çevreye bağlı olarak almış oldukları sıralamalar elde edilmiştir. Her deneyin iterasyon adedi de 5000 olarak belirlenmiştir. Şekil 7.19'da aksiyonların yapılan deneyler sonucunda en iyiden en kötüye doğru sıralanmaları sonucu elde ettikleri yerlerin değerleri görülmektedir.



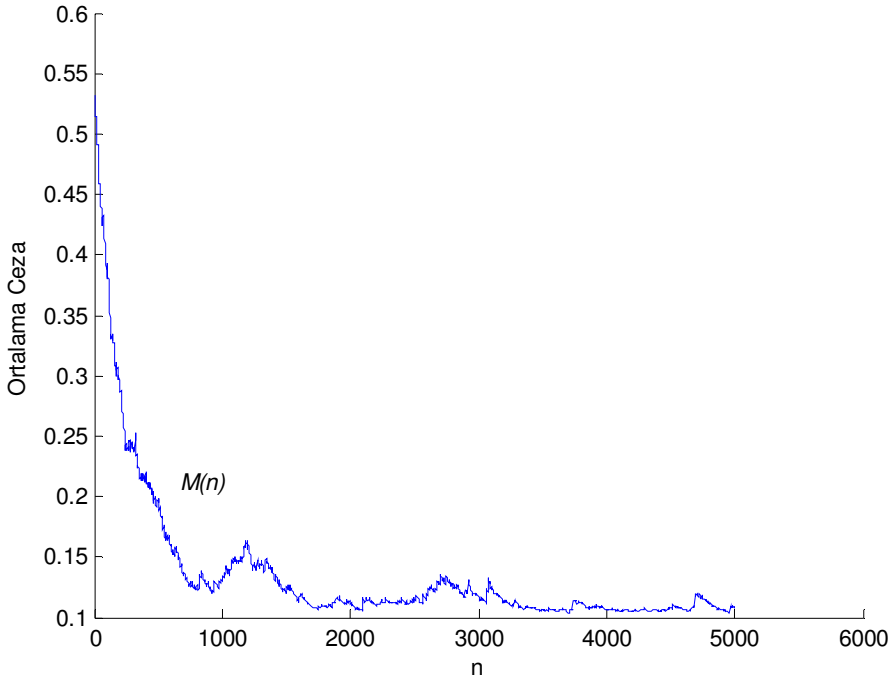
Şekil 7.19 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 2)



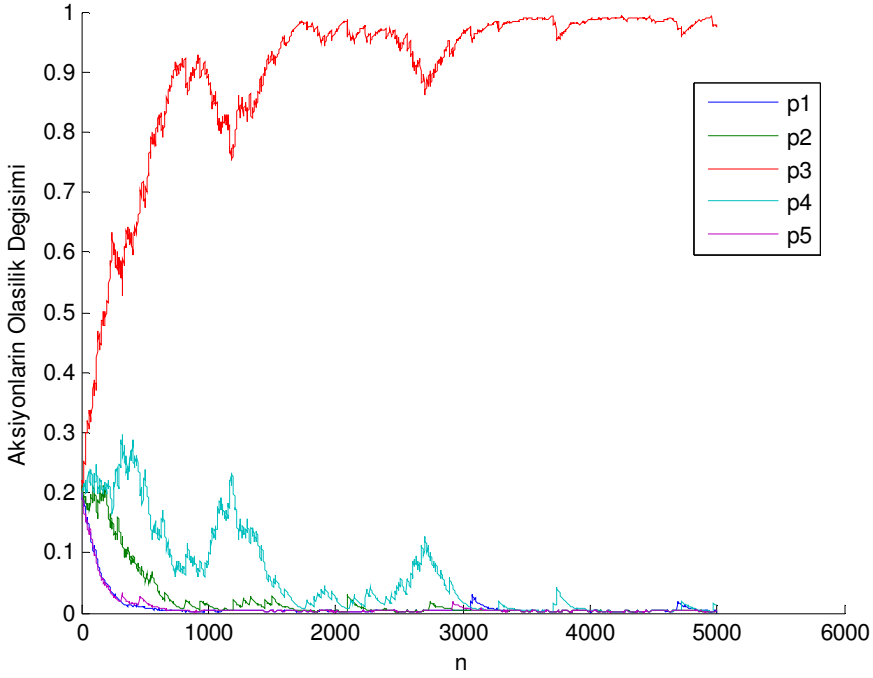
Şekil 7.20 LA çözümünde 3. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 2)

Öğrenen automata yöntemi kullanılarak t = 2 saati için en iyi aksiyon 3 numaralı aksiyon olarak belirlenmiştir. 3 numaralı aksiyonun iterasyona bağlı olasılık değeri değişimi Şekil

7.20'de verilmiştir. Bu durum için ortalama ceza değeri *M(n)*'in değişimi ise Şekil 7.21'de görülmektedir. Şekil 7.22'de ise tüm aksiyonların olasılık değişimleri gösterilmiştir.



Şekil 7.21 LA için ortalama ceza değerinin değişimi ($t = 2$)



Şekil 7.22 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 2$)

Tablo 7.14 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 2)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
3	-	100	100

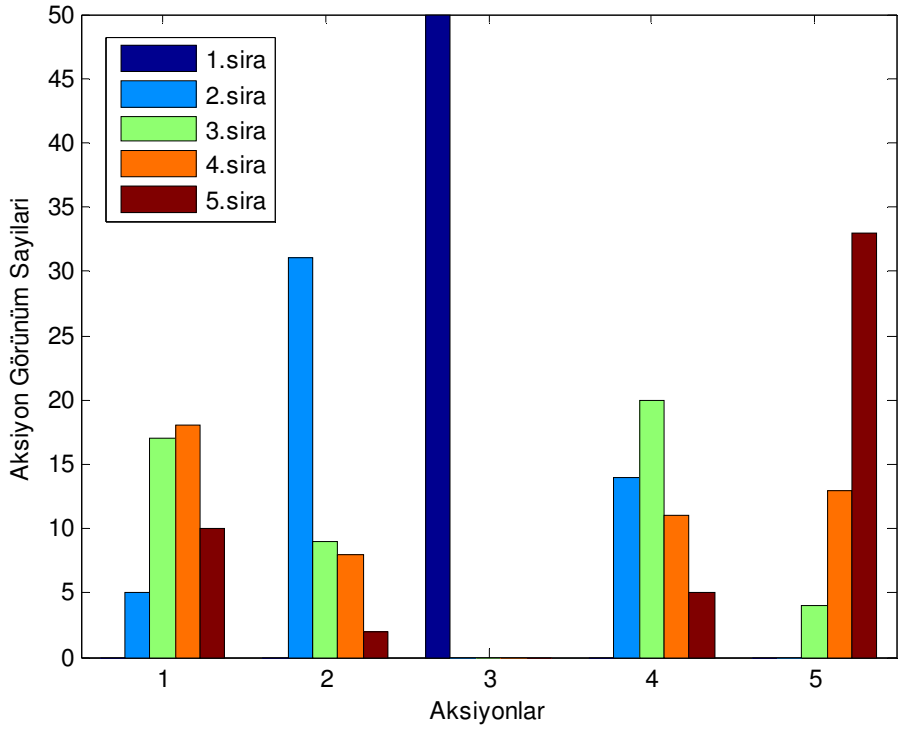
Tablo 7.14'te ise en iyi aksiyon numarası ve t=2 saati için devredeki generatörlerin üretecekleri güç değerleri verilmiştir.

t=2 saati için yapılan tüm işlemler sırasıyla her t anı için tekrarlanmıştır.

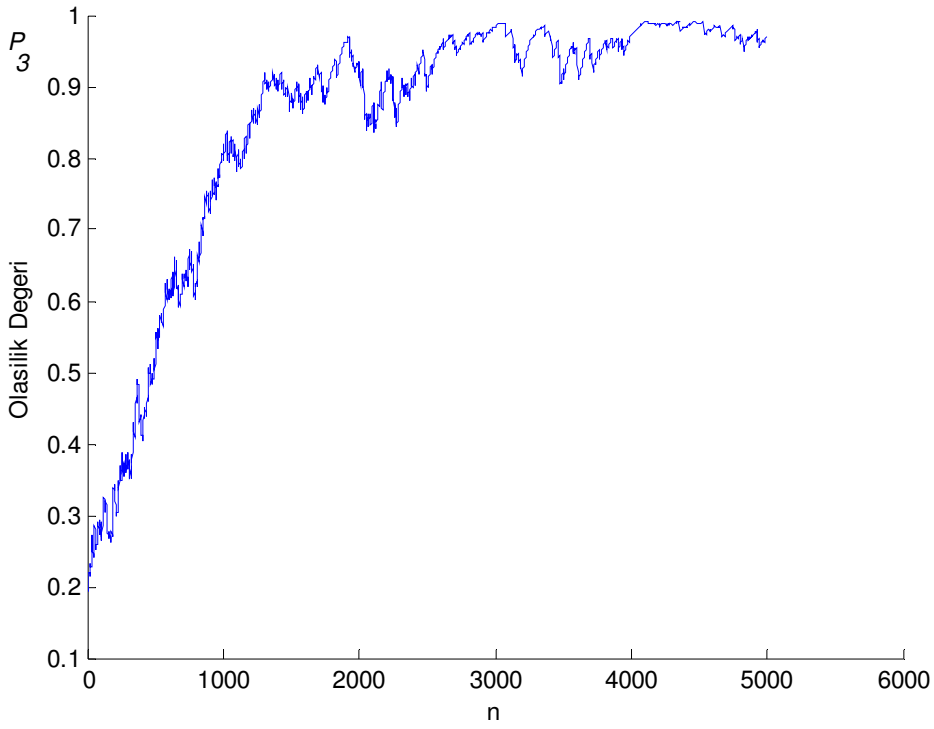
***t* = 3 saati için;**

Tablo 7.15 Öğrenen automata için aksiyonların belirlenmesi (t = 3)

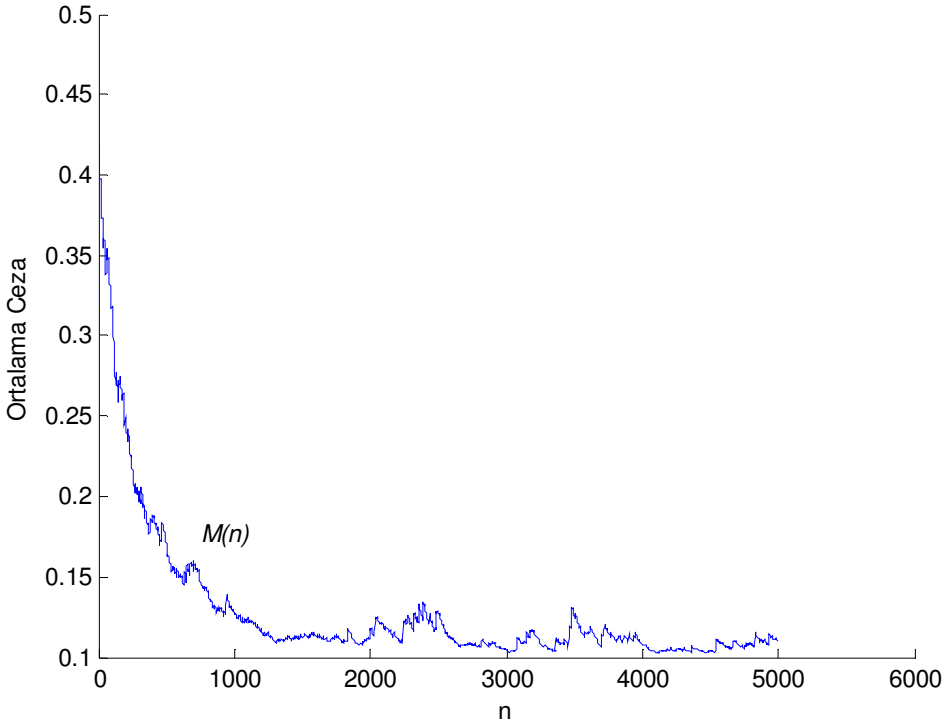
t (Saat)	3				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devre Dışı (0)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_3	112	120	140	150	160



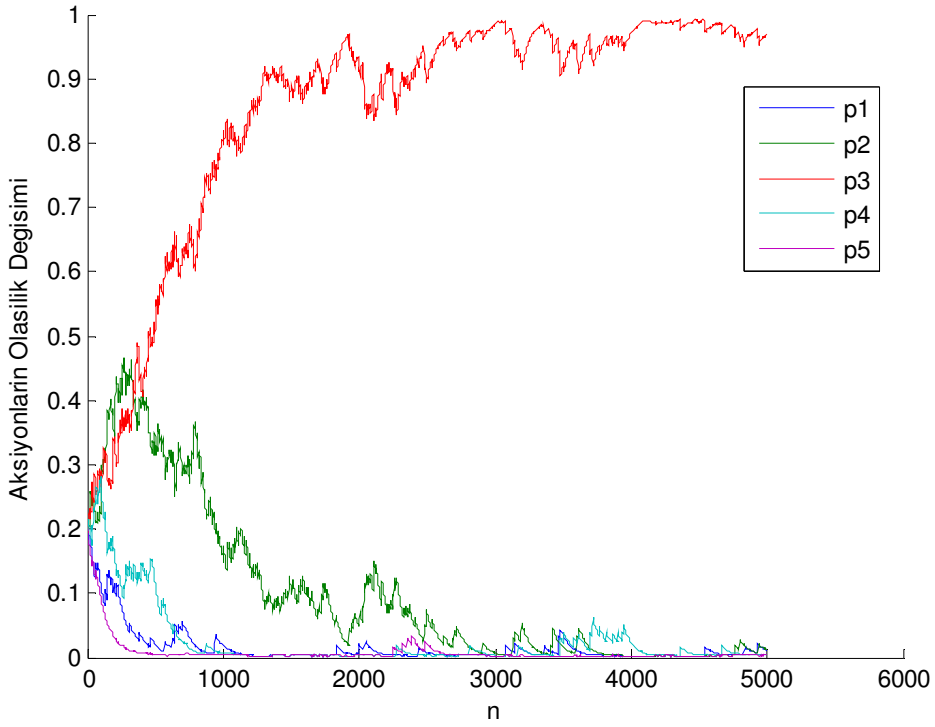
Şekil 7.23 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 3)



Şekil 7.24 LA çözümünde 3. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 3)



Şekil 7.25 LA için ortalama ceza değerinin değişimi (t = 3)



Şekil 7.26 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması (t = 3)

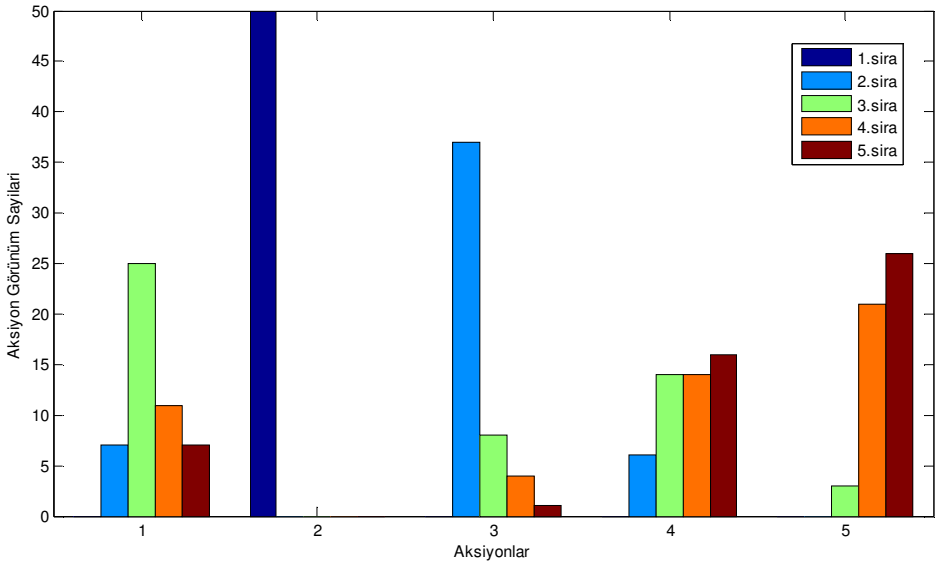
Tablo 7.16 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 3)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
3	-	110	140

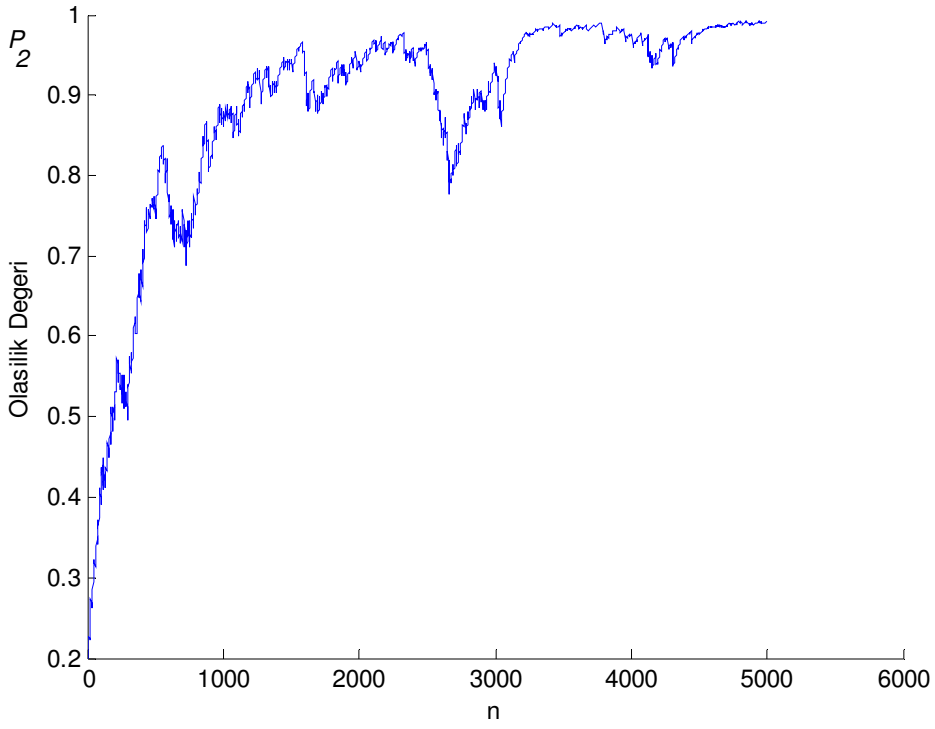
***t* = 4 saati için;**

Tablo 7.17 Öğrenen automata için aksiyonların belirlenmesi (t = 4)

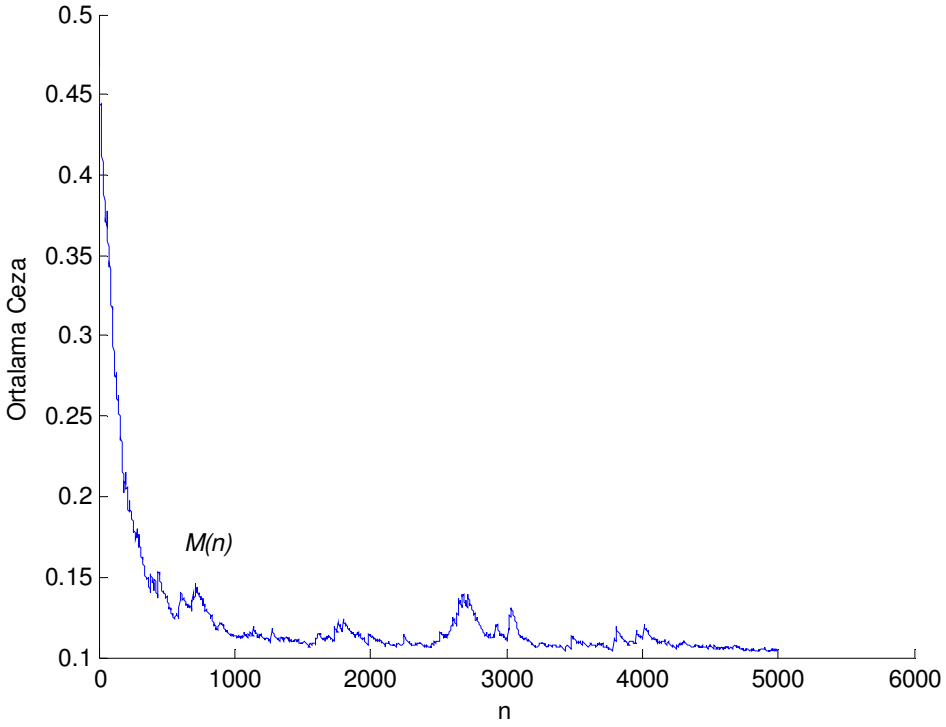
t (Saat)	4				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devre Dışı (0)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_3	126	150	155	165	170



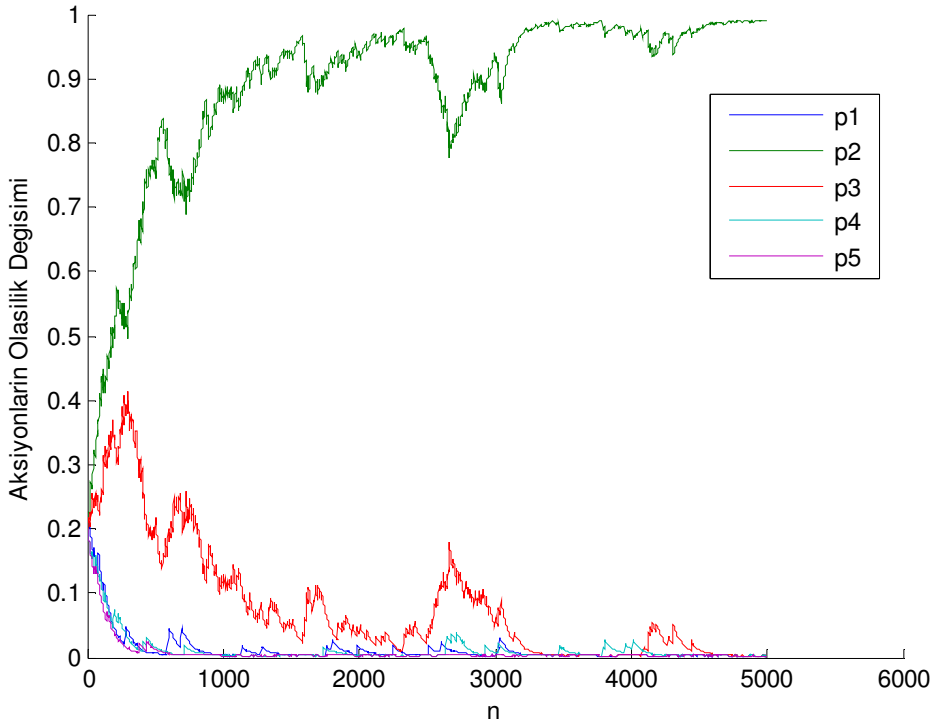
Şekil 7.27 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri ($t = 4$)



Şekil 7.28 LA çözümünde 2. aksiyona ait olasılık değerinin iterasyona bağlı değişimi ($t = 4$)



Şekil 7.29 LA için ortalama ceza değerinin değişimi ($t = 4$)



Şekil 7.30 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 4$)

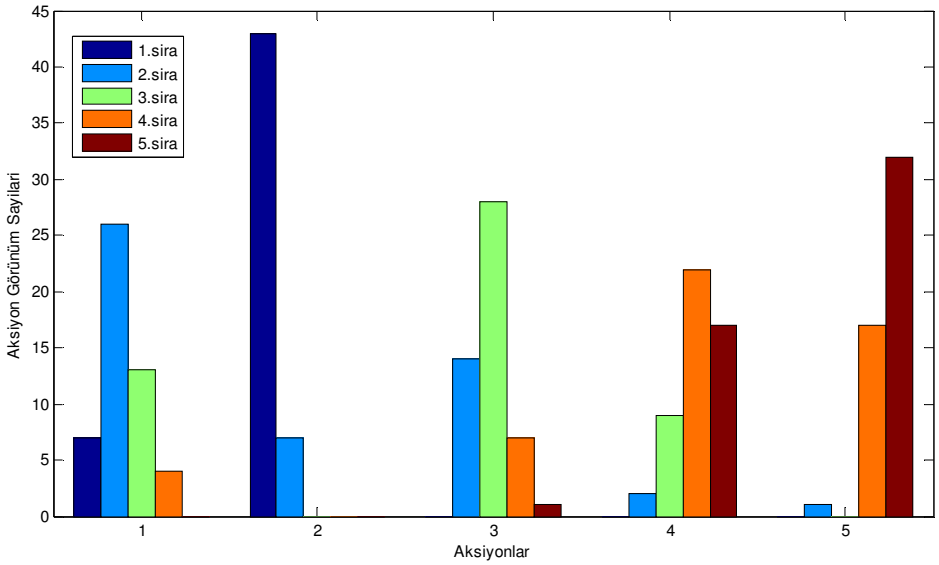
Tablo 7.18 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (*t* = 4)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
2	-	125	150

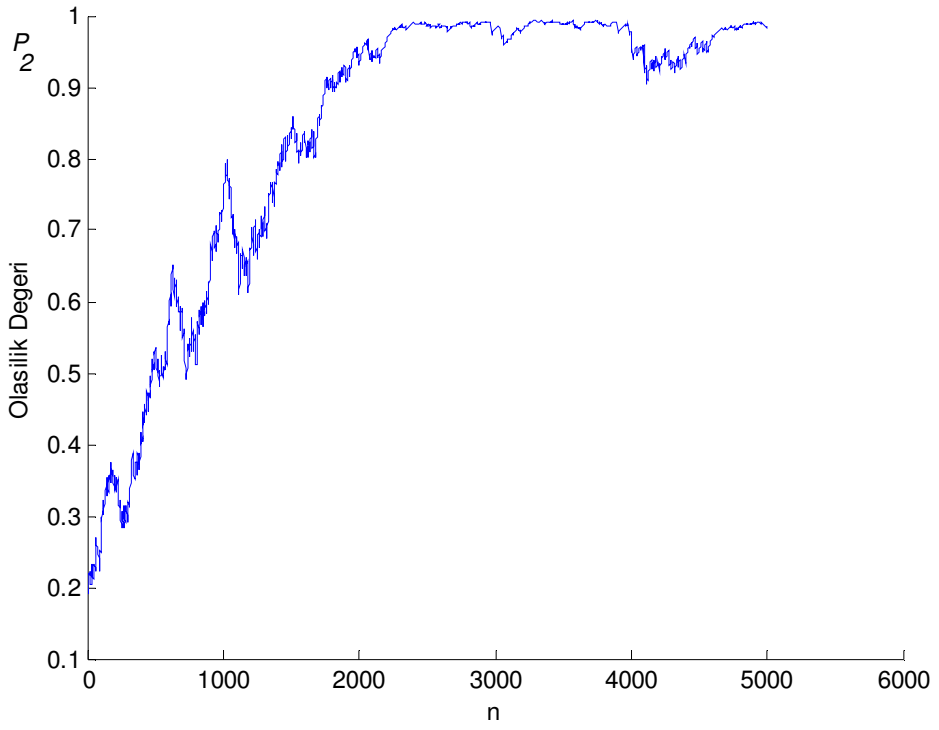
***t* = 5 saati için;**

Tablo 7.19 Öğrenen automata için aksiyonların belirlenmesi (*t* = 5)

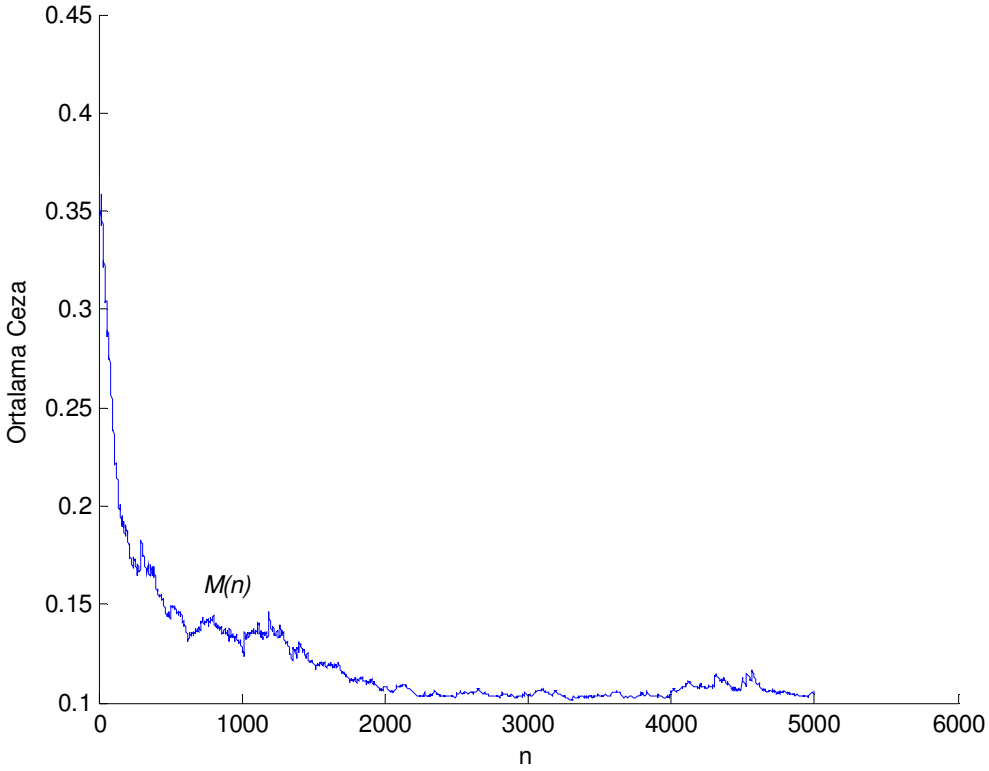
t (Saat)	5				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devre Dışı (0)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_3	151	160	165	170	175



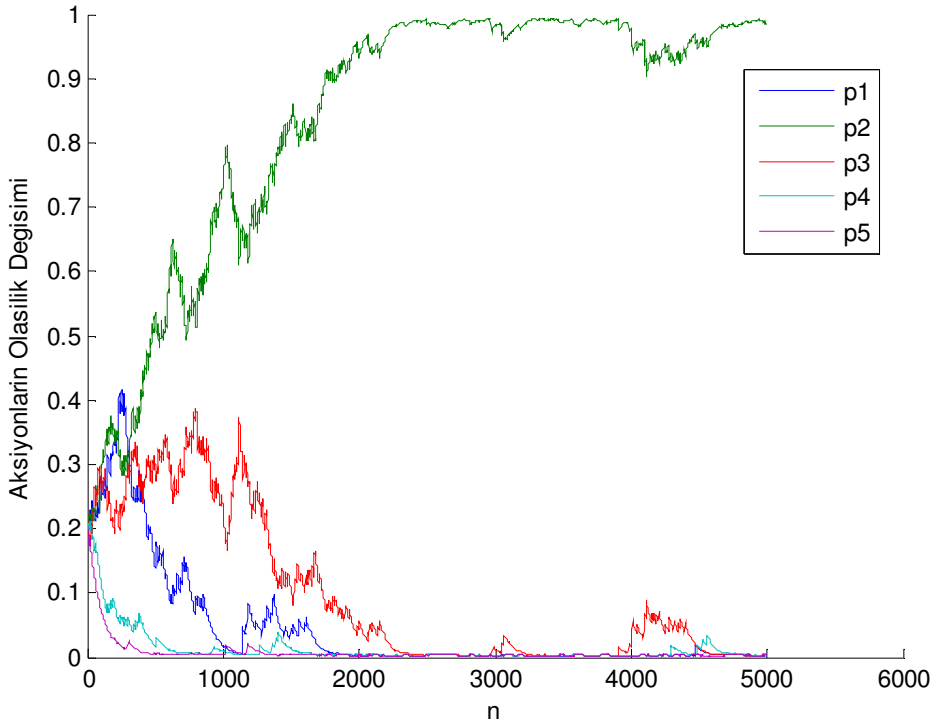
Şekil 7.31 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 5)



Şekil 7.32 LA çözümünde 2. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 5)



Şekil 7.33 LA için ortalama ceza değerinin değişimi ($t = 5$)



Şekil 7.34 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 5$)

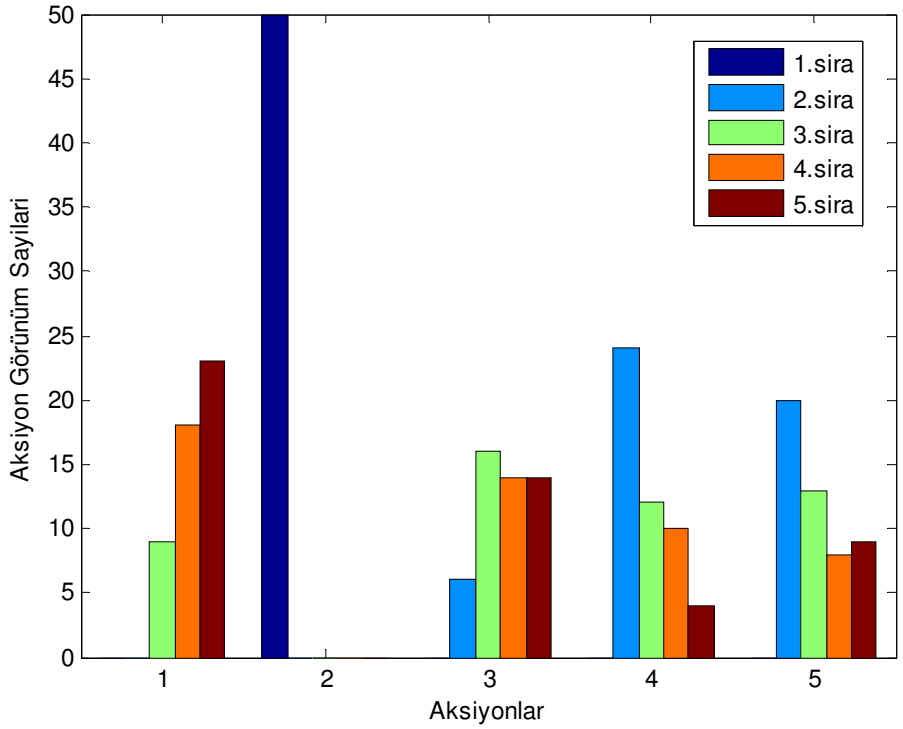
Tablo 7.20 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 5)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
2	-	140	160

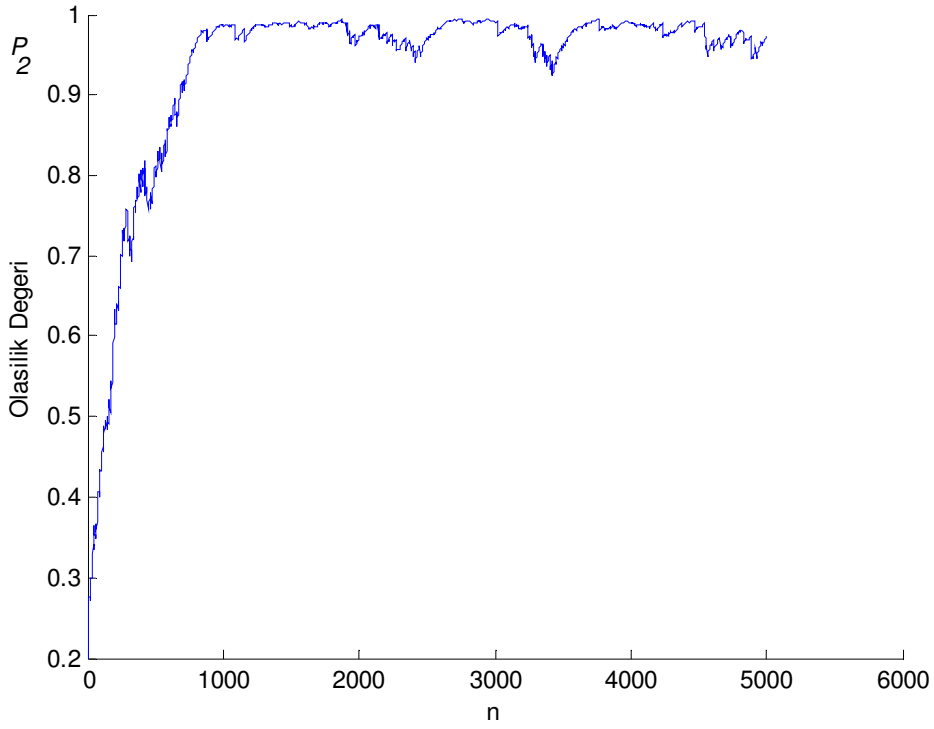
***t* = 6 saati için**

Tablo 7.21 Öğrenen automata için aksiyonların belirlenmesi (t = 6)

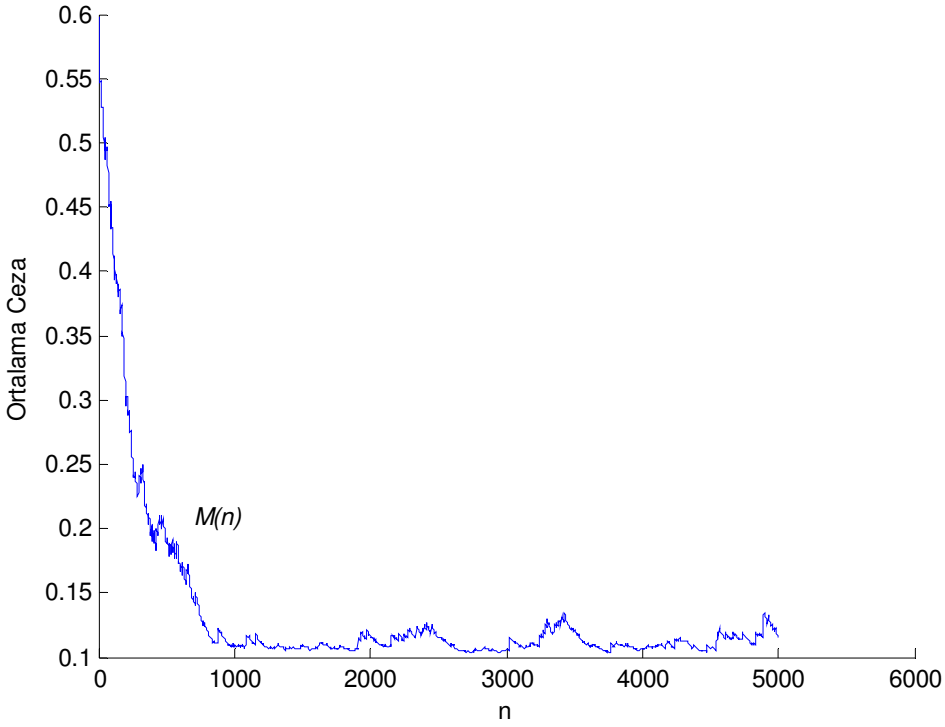
t (Saat)	6				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede (1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	144	120	95	144	144
P_3	126	126	126	135	145



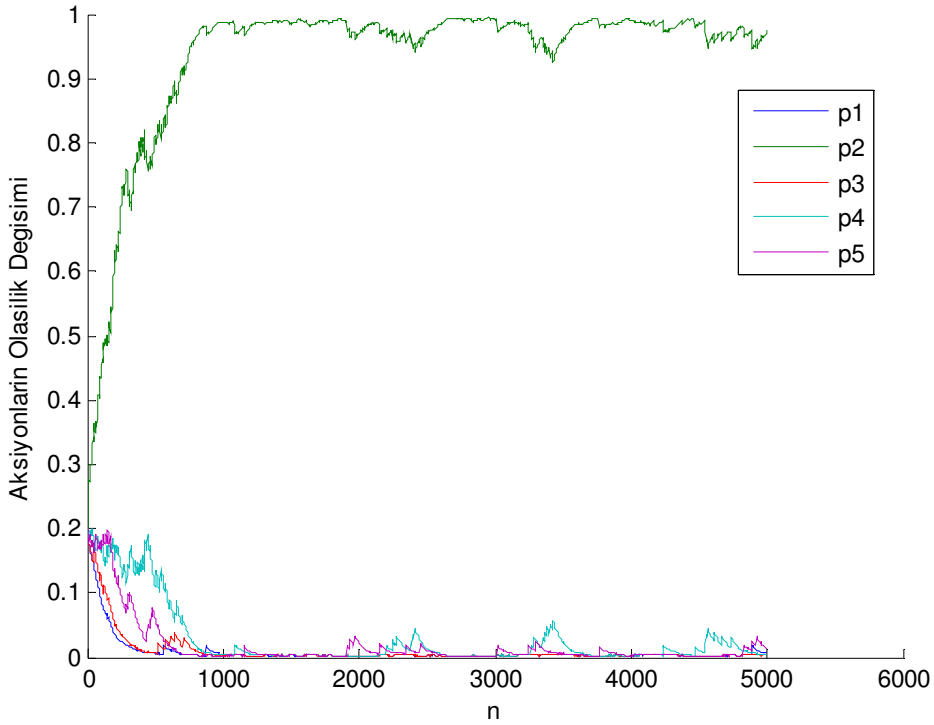
Şekil7.35 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 6)



Şekil 7.36 LA çözümünde 2. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 6)



Şekil 7.37 LA için ortalama ceza değerinin değişimi ($t = 6$)



Şekil 7.38 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 6$)

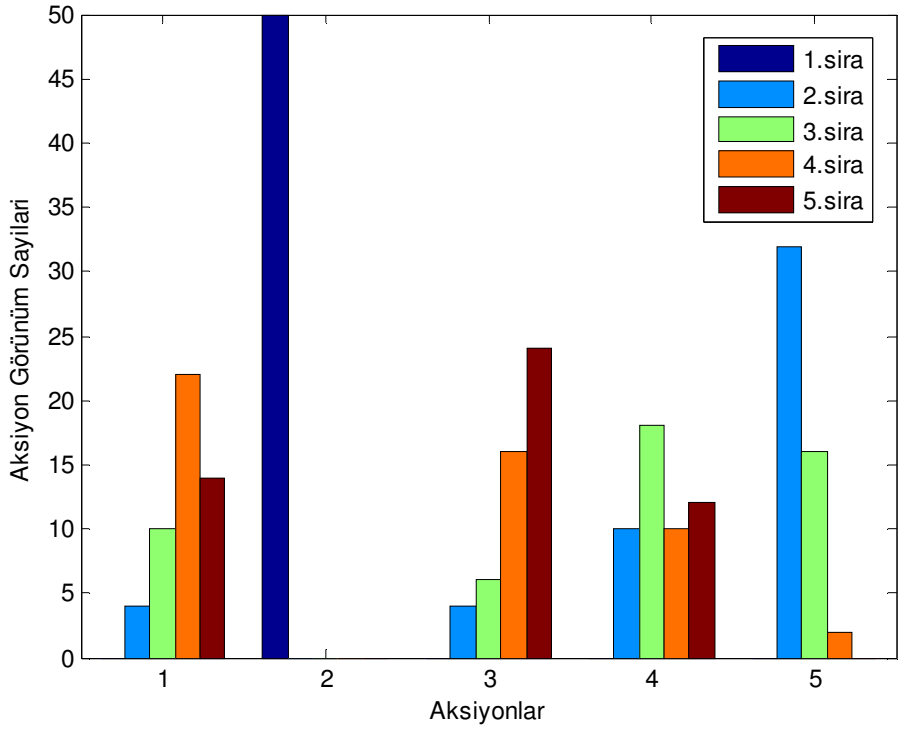
Tablo 7.22 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 6)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
2	104	120	126

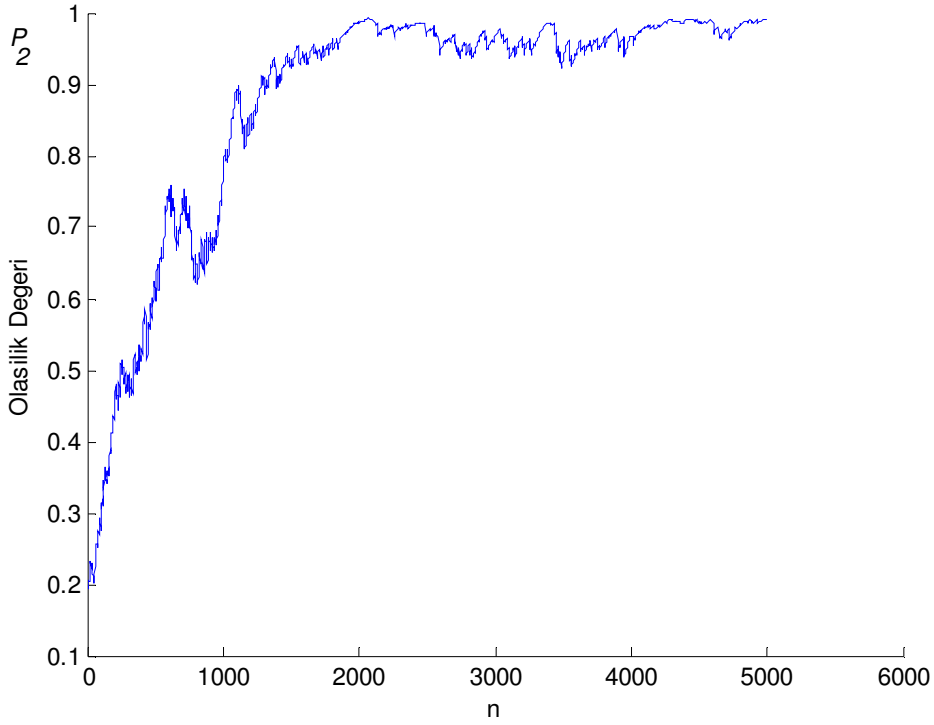
***t* =7 saati için**

Tablo 7.23 Öğrenen automata için aksiyonların belirlenmesi (t = 7)

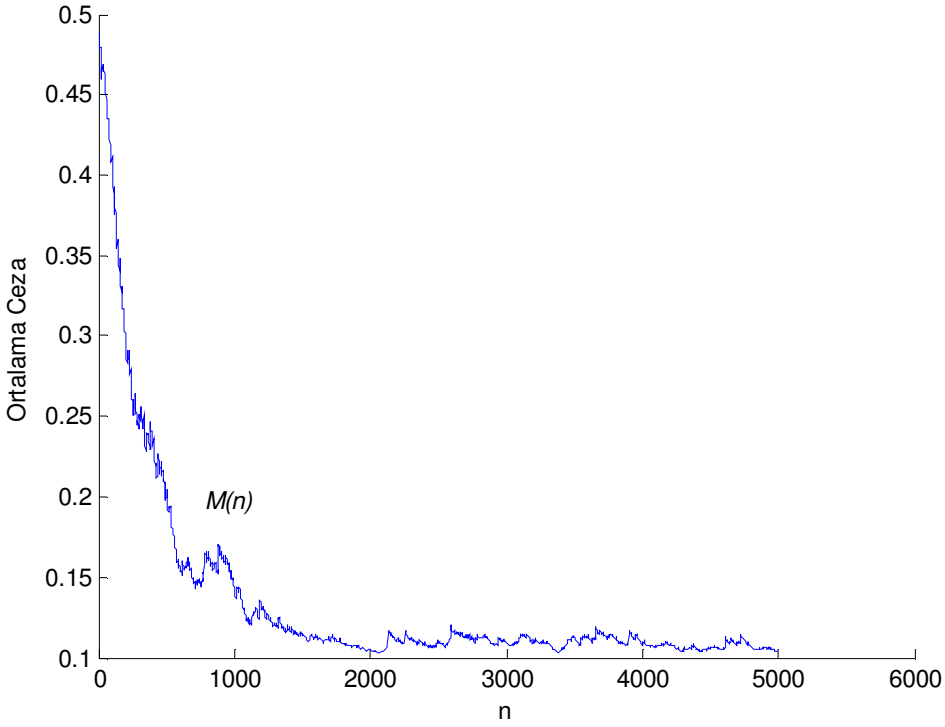
t (Saat)	7				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede(1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	147	147	90	100	110
P_3	133	165	133	133	133



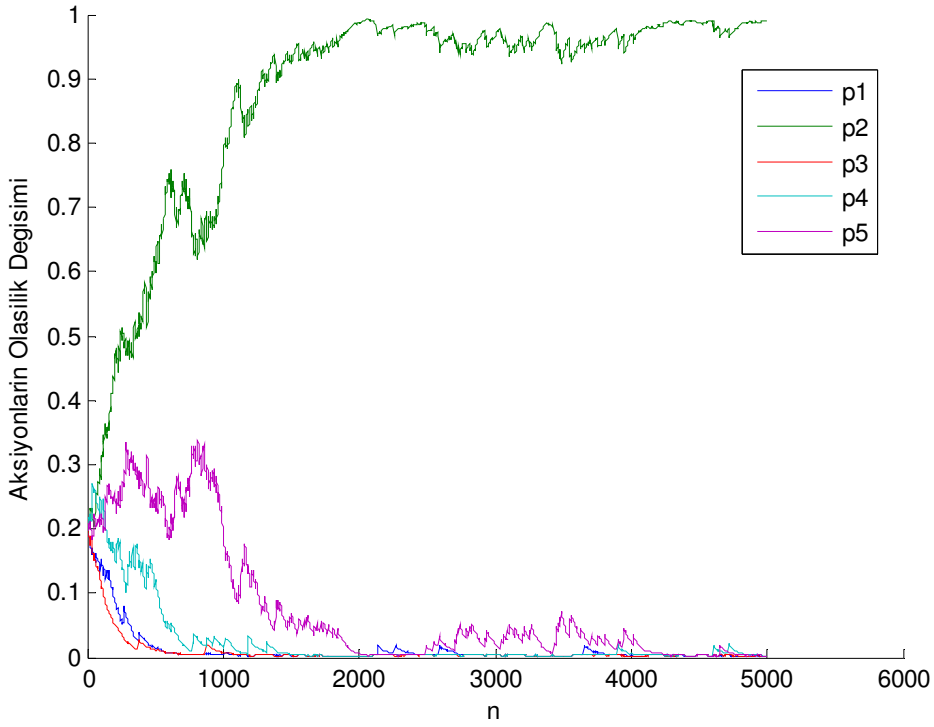
Şekil 7.39 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 7)



Şekil 7.40 LA çözümünde 2. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 7)



Şekil 7.41 LA için ortalama ceza değerinin değişimi ($t = 7$)



Şekil 7.42 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 7$)

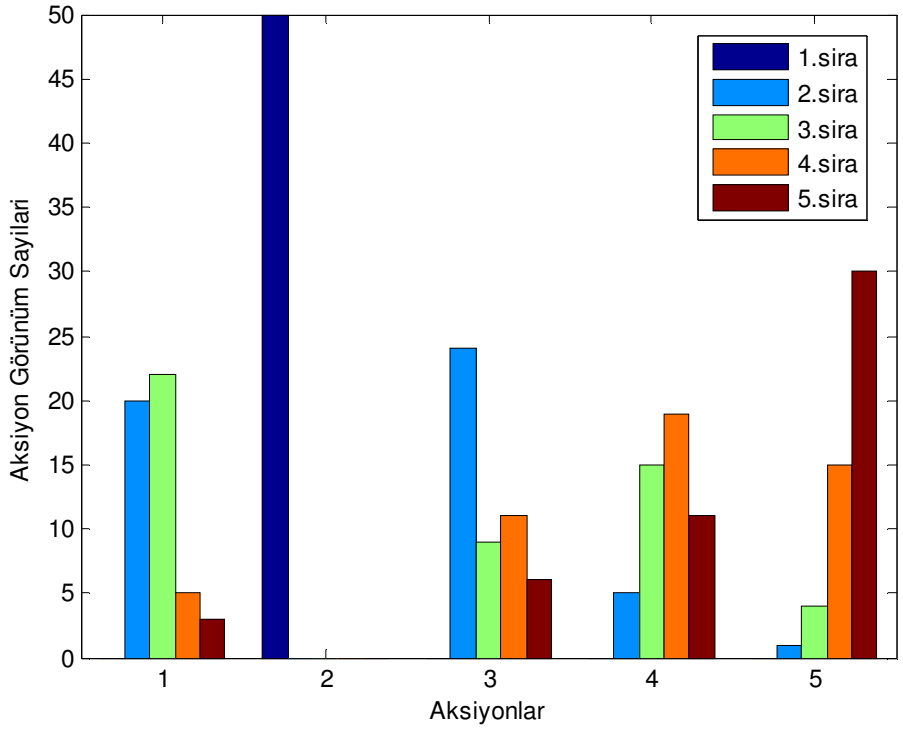
Tablo 7.24 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (*t* = 7)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
2	63	147	165

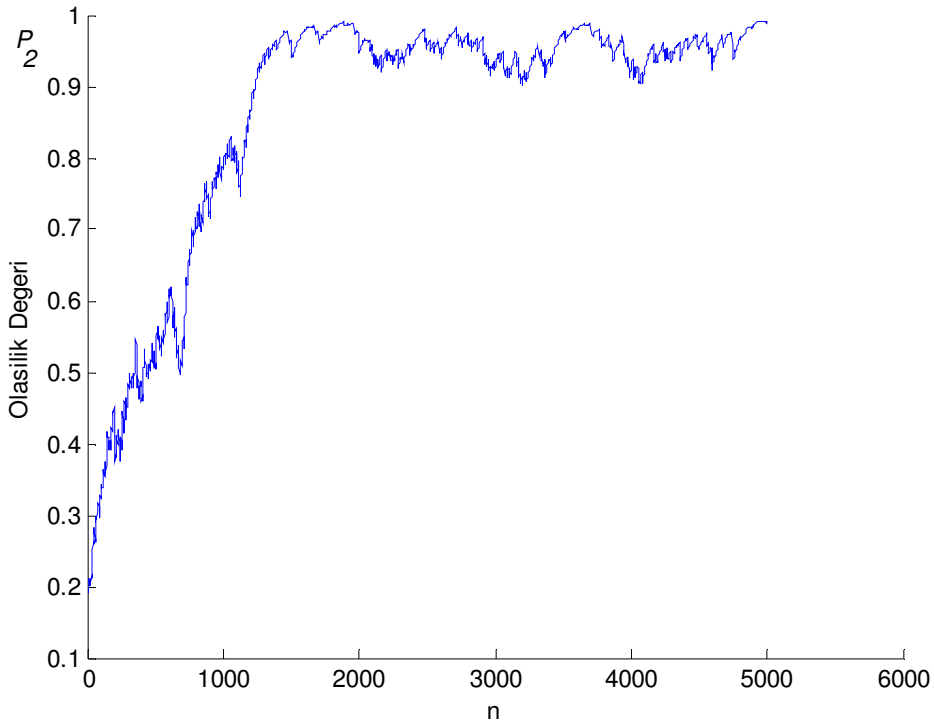
***t* = 8 saati için;**

Tablo 7.25 Öğrenen automata için aksiyonların belirlenmesi (*t* = 8)

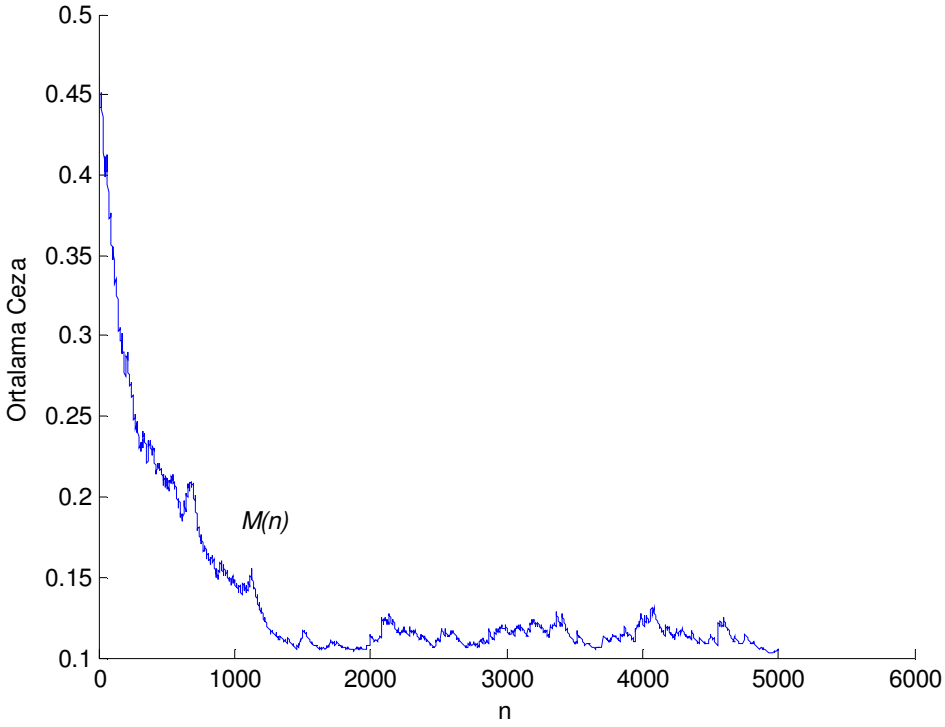
t (Saat)	8				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede (1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	145	110	90	80	70
P_3	160	160	160	160	160



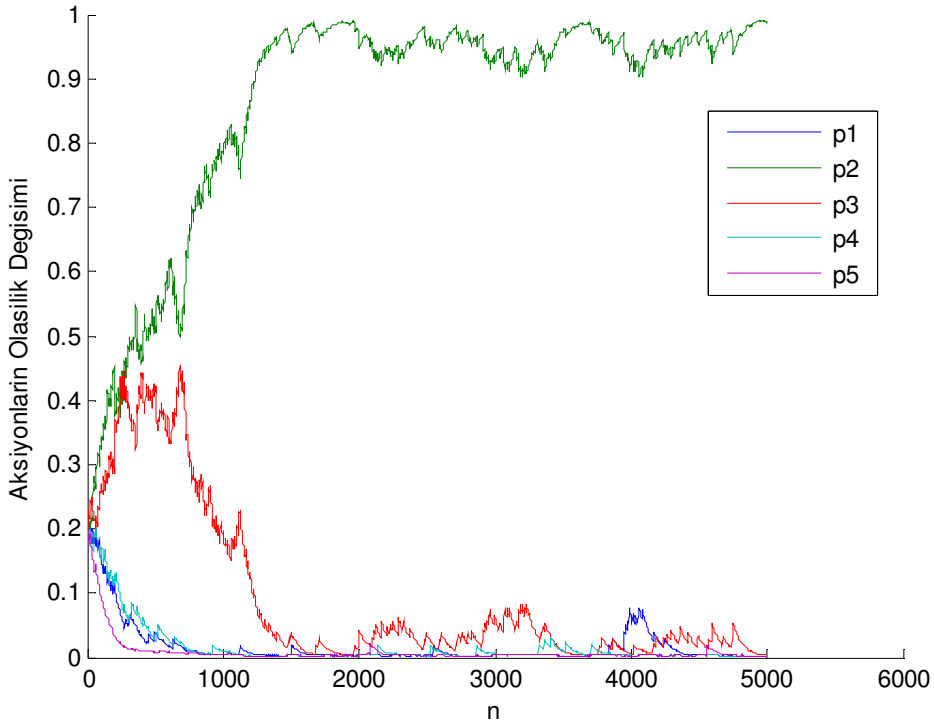
Şekil 7.43 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 8)



Şekil 7.44 LA çözümünde 2. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 8)



Şekil 7.45 LA için ortalama ceza değerinin değişimi ($t = 8$)



Şekil 7.46 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 8$)

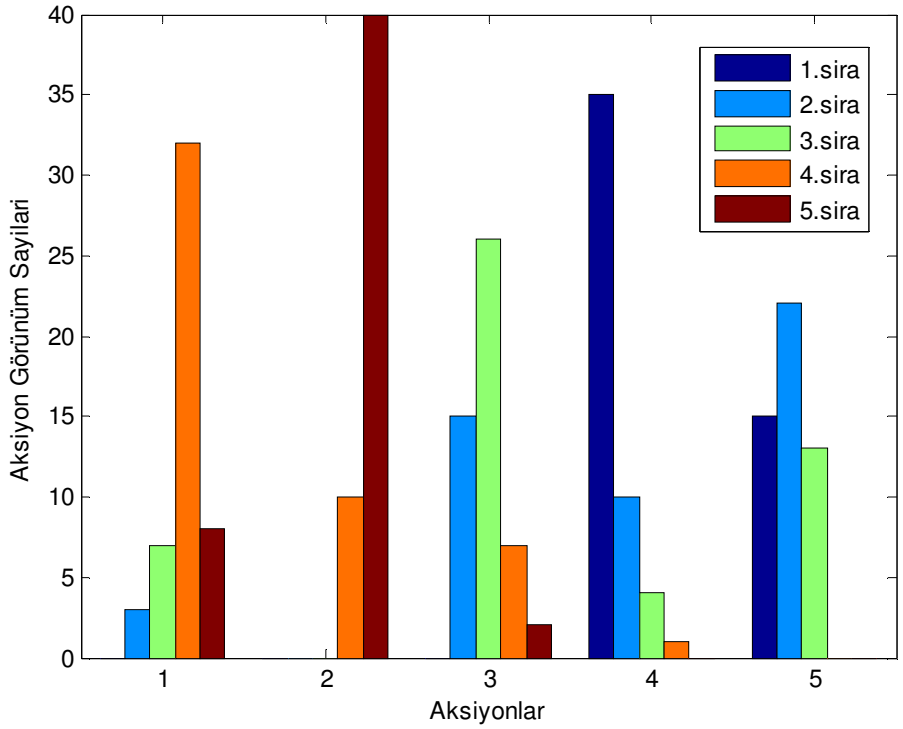
Tablo 7.26 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (*t* = 8)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
2	130	110	160

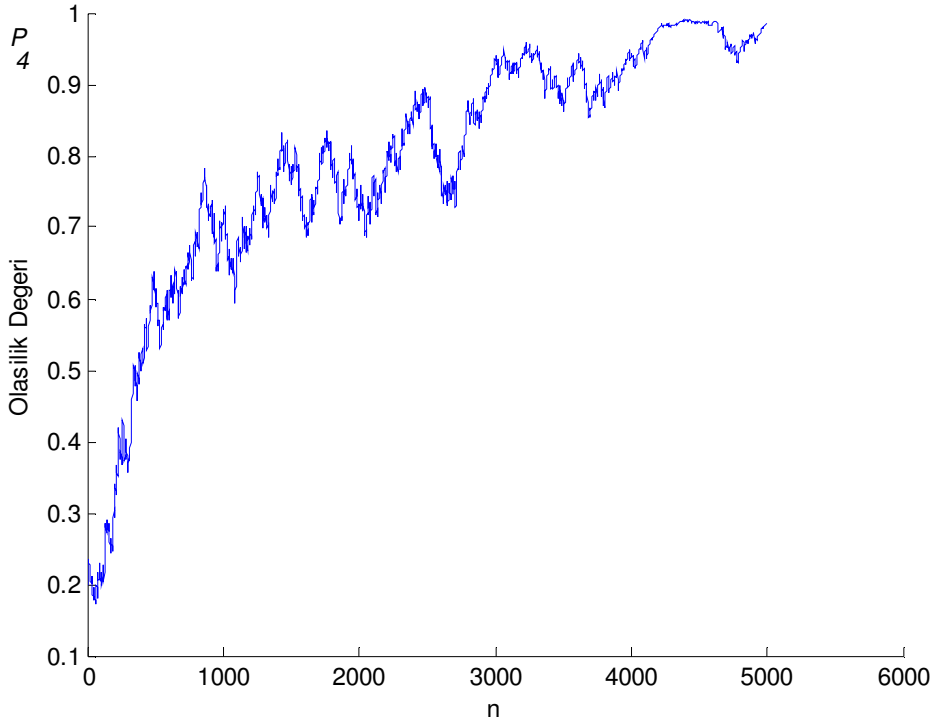
***t* = 9 saati için;**

Tablo 7.27 Öğrenen automata için aksiyonların belirlenmesi (*t* = 9)

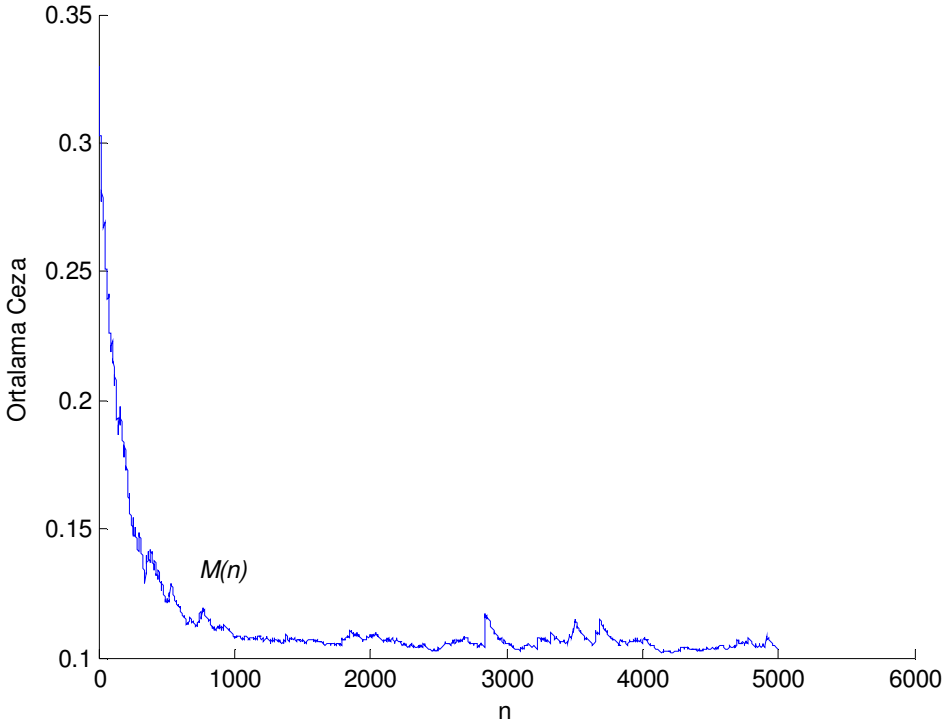
t (Saat)	9				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede (1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	150	135	150	150	150
P_3	150	150	160	175	180



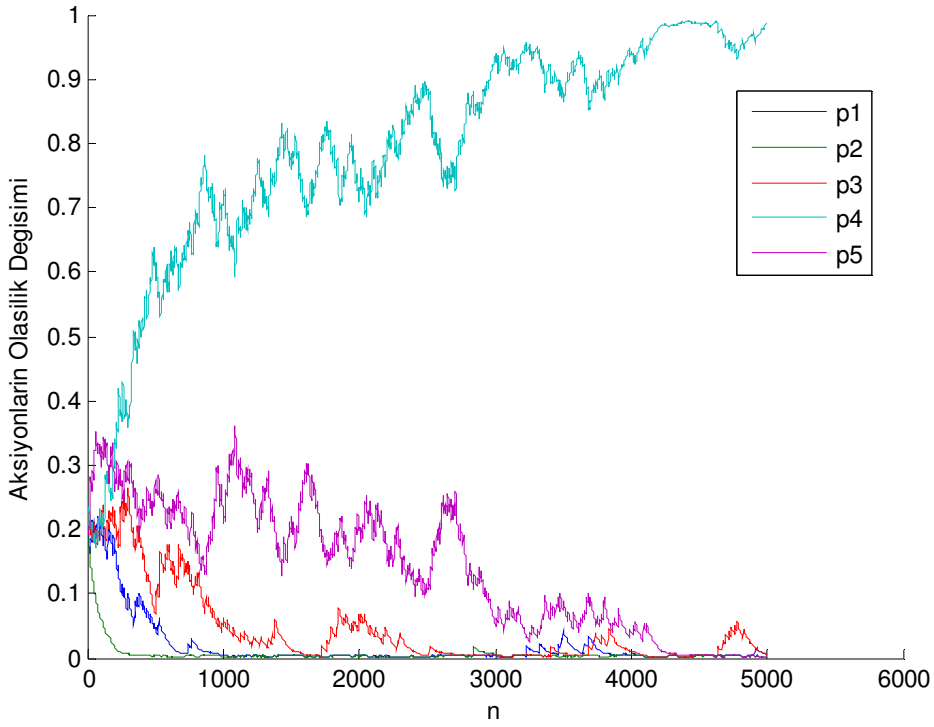
Şekil 7.47 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri ($t = 9$)



Şekil 7.48 LA çözümünde 4. aksiyona ait olasılık değerinin iterasyona bağlı değişimi ($t = 9$)



Şekil 7.49 LA için ortalama ceza değerinin değişimi ($t = 9$)



Şekil 7.50 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması ($t = 9$)

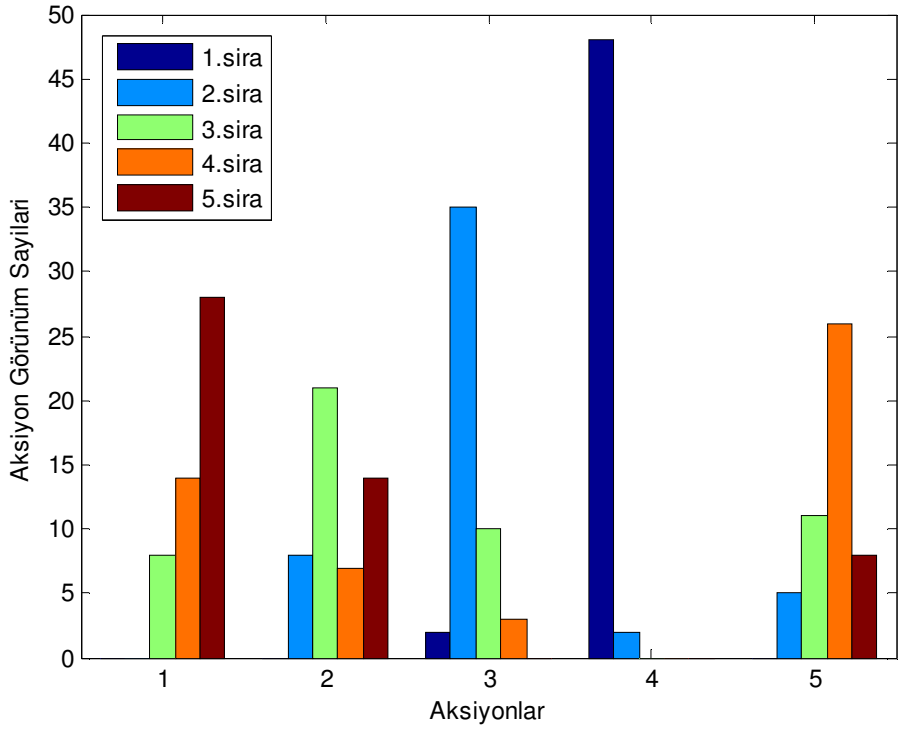
Tablo 7.28 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 9)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
4	100	150	175

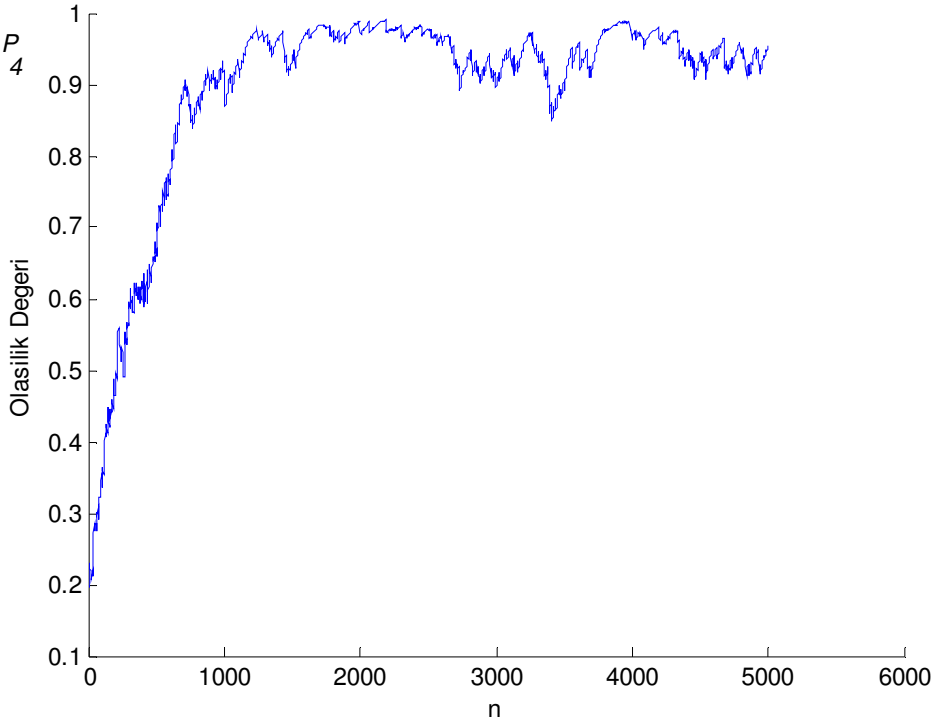
***t* =10 saati için;**

Tablo 7.29 Öğrenen automata için aksiyonların belirlenmesi (t = 10)

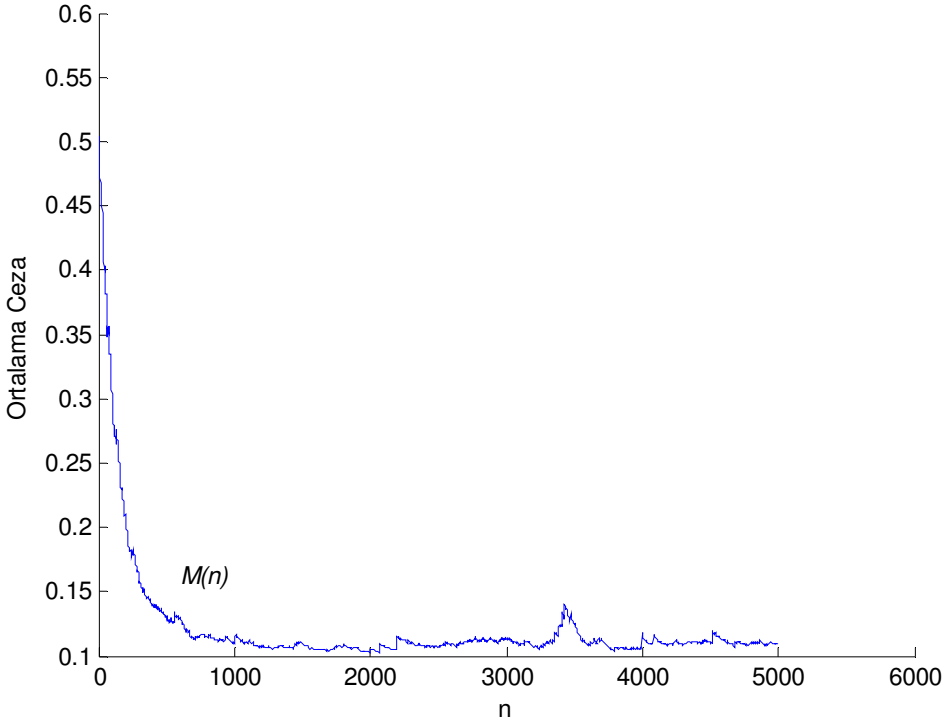
t (Saat)	10				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede (1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	144	144	144	144	144
P_3	166	168	170	174	180



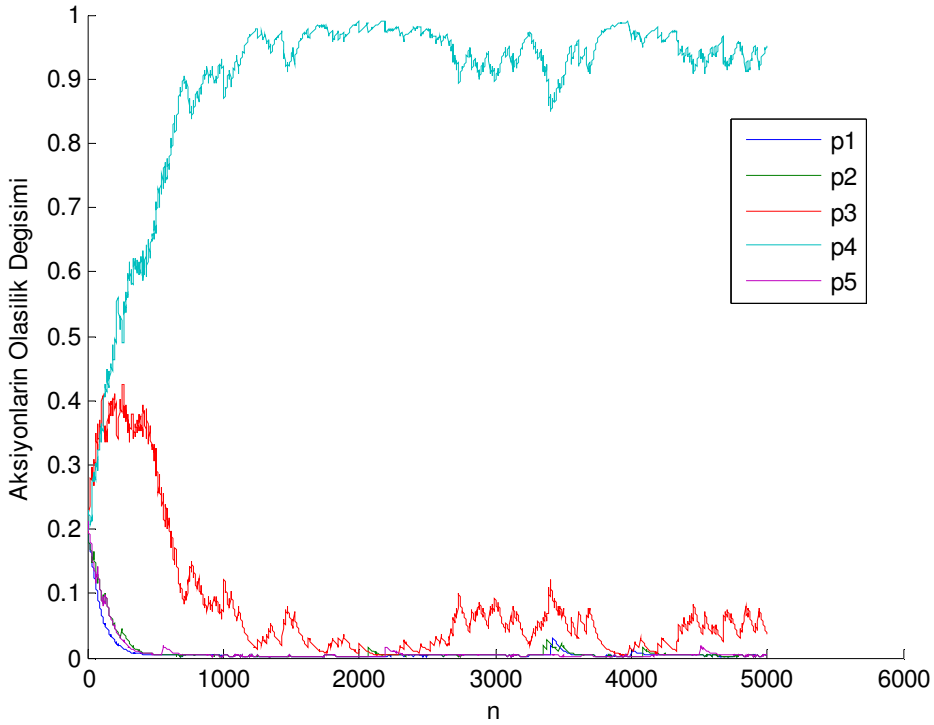
Şekil 7.51 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 10)



Şekil 7.52 LA çözümünde 4. aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 10)



Şekil 7.53 LA için ortalama ceza değerinin değişimi (t = 10)



Şekil 7.54 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması (t = 10)

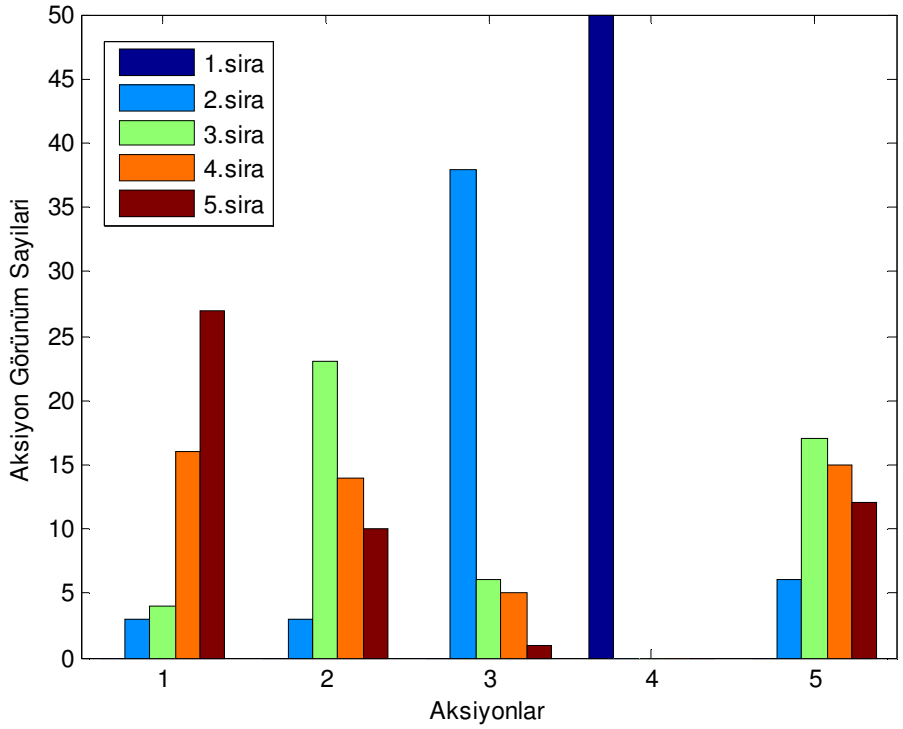
Tablo 7.30 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 10)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
4	132	144	174

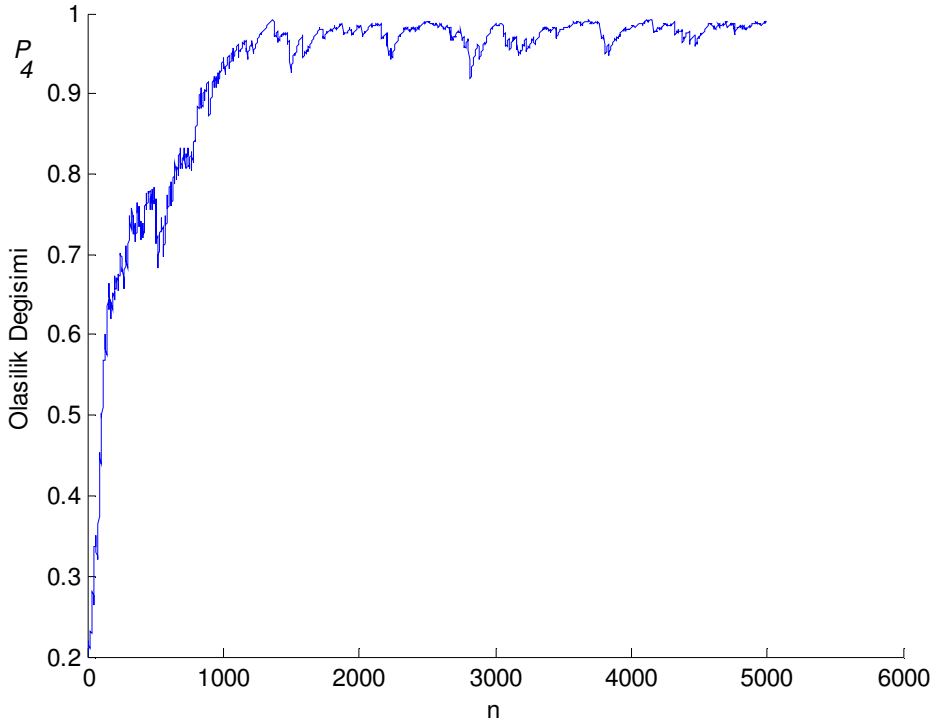
***t* =11 saati için;**

Tablo 7.31 Öğrenen automata için aksiyonların belirlenmesi (t = 11)

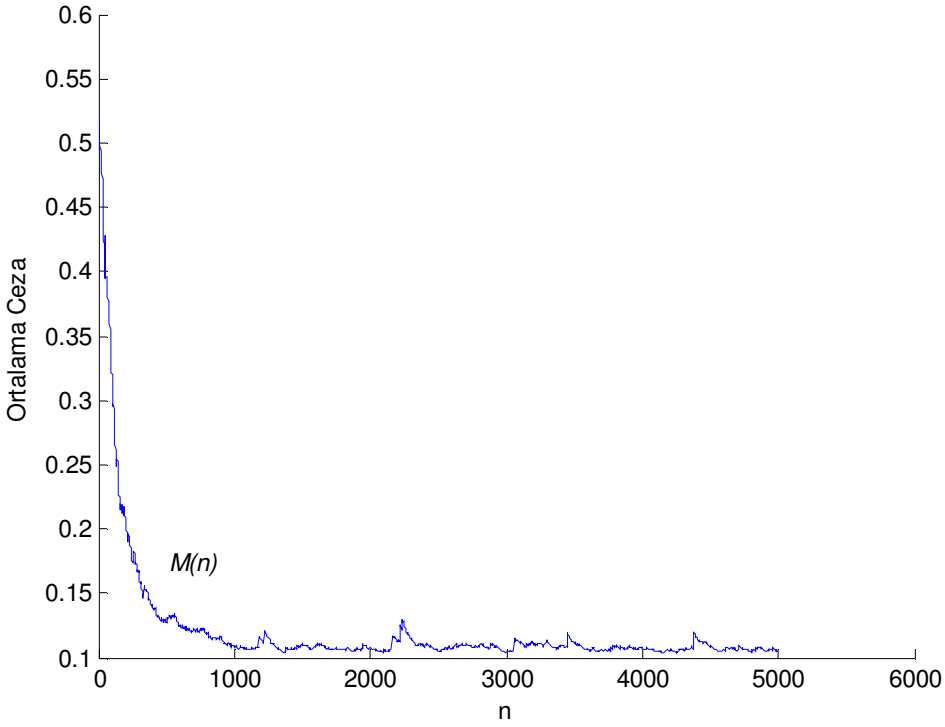
t (Saat)	11				
Generatörlerin Durumu	Generatör 1		Generatör 2		Generatör 3
	Devrede (1)		Devrede (1)		Devrede (1)
Generatör Güçleri (MW)	Aksiyonlar				
	1	2	3	4	5
P_2	149	149	149	149	149
P_3	171	172	174	178	180



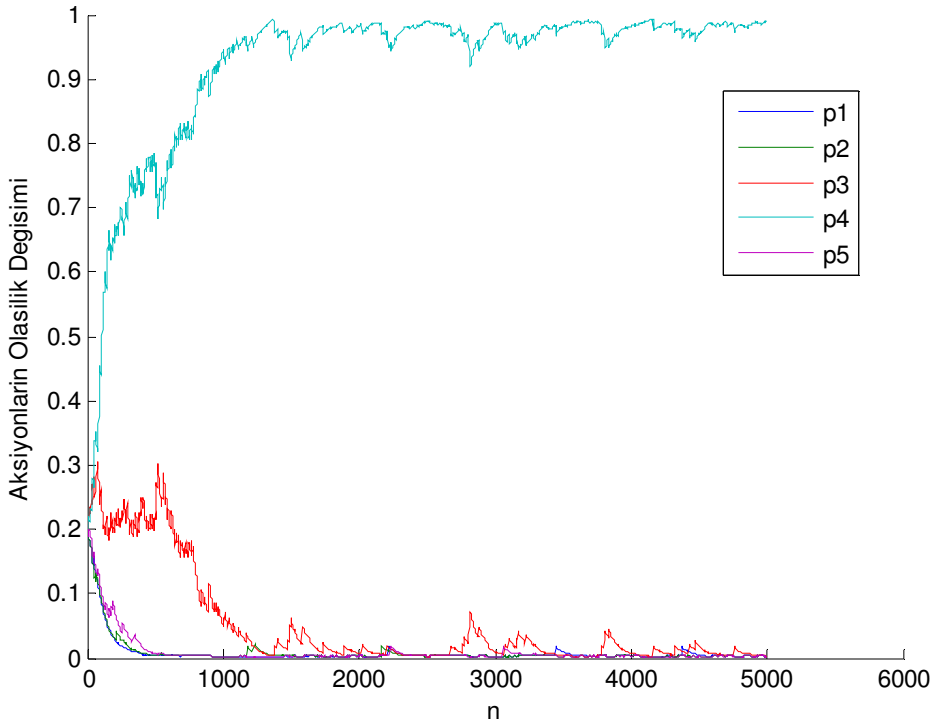
Şekil 7.55 LA ile çözüm için yapılan öğrenme işlemleri sonucunda aksiyonların aldıkları sıralama değerleri (t = 11)



Şekil 7.56 LA çözümünde 4 aksiyona ait olasılık değerinin iterasyona bağlı değişimi (t = 11)



Şekil 7.57 LA için ortalama ceza değerinin değişimi (t = 11)



Şekil 7.58 LA kullanımında aksiyonların olasılık değişimlerinin karşılaştırılması (t = 11)

Tablo 7.32 LA kullanımı sonucu elde edilen en iyi aksiyon ve güç değerleri (t = 11)

En İyi Aksiyon	Generatör Güçleri (MW)		
	P_1	P_2	P_3
4	148	149	178

***t* = 12 için;**

bu durum için de yapılan analizlerde, 2 ve 3 nolu generatörlerin maksimum güç limitinde olmaları ve 1 nolu generatörün ürettiği güç değerinin değişmesinden de kararlılık anlamında bir iyileştirme sağlanamadığından, mevcut durumdan daha iyi bir sonuç elde edilememiştir ve bu değerler göz önüne alınmıştır.

Elde Edilen Sonuçlar:

Her *t* saati için yapılan analizler sonucunda tezde geliştirilen metoda uygun olarak sistemdeki mevcut generatörlerin üretecekleri güç miktarları Tablo 7.33'de verilmiştir. Bu durum için elde edilen maliyet değeri 37791,69 $ olarak belirlenmiştir. Maliyet değerindeki bu artış ile birlikte sistem kararlılığında da iyileştirme sağlanmıştır. Geliştirilen metot ile böylece sistem için hem ekonomiklik hem de kararlılık olarak istenen kriterleri sağlayan optimum bir işletim noktası sağlanmıştır.

Tablo 7.33 Generatör birimlerinin üretim değerleri

Birimler	t (Saat) , P(MW)											
	1	2	3	4	5	6	7	8	9	10	11	12
Gen.1	0	0	0	0	0	104	63	130	100	132	148	170
Gen.2	150	100	110	125	140	120	147	110	150	144	149	150
Gen.3	0	100	140	150	160	126	165	160	175	174	178	180

8. SONUÇLAR VE ÖNERİLER

Günümüzde gelişen ve ilerleyen teknolojiye paralel olarak elektrik enerjisine duyulan ihtiyaç her geçen gün artmaktadır. Elektrik enerjisine duyulan bu gereksinim, enerji ile ilgili bazı parametrelerin de önem kazanmasına neden olmuştur. Üretilen enerjinin ekonomik bir maliyete sahip olması, daha kararlı ve daha güvenilir olması gibi özellikler, elektrik enerjisinin üretim, iletim ve dağıtımında göz önüne alınması gereken en önemli kriterlerdir. Bu çalışmada, birim yüklenme maliyeti olarak tanımlanan enerji üretiminde ekonomik yük dağıtımı sağlanmış ve bu ekonomik işletim durumu, iyileştirilmiş bir kararlılık yapısının elde edilmesi için incelenmiştir. Böylece sistem için hem ekonomik maliyete hem de daha iyi bir kararlılığa sahip optimum bir işletim noktasının elde edilmesi gerçekleştirilmiştir.

Bir elektrik güç sisteminde maliyet optimizasyonu çok önemli bir parametredir. Ekonomik yük dağıtımı, birim yüklenme probleminin bir parçası olup, sistemde optimum enerji üretiminin sağlanması için generatör birimlerinin her birinin ne kadar güç üreteceğinin belirlenmesidir. Diğer bir deyişle, birim yüklenme problemi, enterkonnekte bir güç sisteminde en ekonomik enerji üretimini sağlamak için sistemdeki generatör birimlerinin hangilerinin devrede, hangilerinin ise devre dışında olacaklarının planlanması olarak ifade edilebilir. Problemin çözümünde, sistemden talep edilen güç miktarının sağlanması gerekirken, generatör birimlerinin üreteceği güç miktarlarının ve minimum devrede veya devre dışında kalmasına ait zaman kısıtlarının da göz önünde tutulması gerekmektedir. Bu nedenle birim yüklenme problemi sistemin fiziksel ve dinamiksel işletim limitlerinden dolayı belli başlı kısıtlara sahip bir optimizasyon problemidir.

Güç sisteminde önemli bir diğer parametre de kararlı bir işletim noktasına sahip olunmasıdır. Bu çalışmada ekonomik işletimle birlikte gerilim kararlılığı da göz önüne alınmıştır. Gerilim kararlılığı, hem normal işletim şartlarında, hem de sistemde meydana gelen bir bozucu durum sonrasında enerji sistemindeki bütün baraların gerilim değerlerini belli limitler içerisinde tutabilme kabiliyeti olarak ifade edilebilir.

Tez çalışmasının sonucu olarak, güç sisteminin üretim ve işletimi planlaması için aşağıdaki tespitler yapılabilir:

Sistemin sürekliliği bakımından kararlılık çalışmalarının önemi bilinmektedir. Enerji üretiminin maliyeti göz önüne alınarak elde edilen en ekonomik çözümün yanında, kararlılık açısından da değerlendirilmesinin mutlaka gerektiği ortaya çıkmaktadır. Çünkü her bir ekonomik çalışmanın kararlılık değerleri farklı olmaktadır.

Yapılan bu çalışma sonucunda, birim yüklenme probleminin çözüm yönteminde sadece ekonomik kriterlerin değil, sistemin kararlılık bakımından davranışına ait incelemeleri de dahil eden bir yöntem önerilmiştir. Yapılan sayısal uygulamalarla ekonomik güç dağılımının yanında daha kararlı bir sistemin kullanılması sağlanmış ve bu yönüyle çalışma sonucunda olumlu katkılar görülmüştür.

Bu tez çalışmasında, birim yüklenme probleminin analizi kararlı çalışma koşulları göz önüne alınarak gerçekleştirilmiştir. Bu analizin gerçekleştirilmesi aşamalarında, birim yüklenme probleminin çözümünde evrimsel programlama yöntemi kullanılmıştır. Diğer yöntemlere göre daha hızlı, daha doğru ve diğer yöntemler kadar karmaşık olmayan bir yapıda olduğu ve sonuçların elde edilmesinde daha uygun bir yöntem olduğu görülmüştür.

Generatör birimlerinin üretecekleri güç miktarlarının belirlendiği ekonomik güç dağıtımı probleminin çözümünde ise genetik algoritma yöntemi kullanılmış ve diğer çözüm yöntemlerinden daha çabuk ve doğru bir şekilde sonuca ulaşıldığı tespit edilmiştir.

Güç sisteminin farklı işletim koşullarındaki kararlılığının kıyaslanmasında V-Q duyarlılık kriteri kullanılmış olup böylece sistem kararlılığı hakkında bilgi sahibi olunmuştur. Böylece sistemde hangi işletim durumlarının daha kararlı, hangi işletim durumlarının kararsızlığa yakın olduğu tespit edilmiş olmaktadır. Çalışmada kullanılan yöntemin sistemde kararlılık analizini daha hızlı ve detaylı analizlere girmeksizin gerçekleştirdiği belirlenmiştir.

Yine bu çalışmada, V-Q duyarlılık endeksinin kullanımı ile orijinal olarak birim yüklenme probleminin analizine farklı bir bakış açısı getirilmiştir. Sistem için, farklı işletim noktalarının sahip olduğu kararlılık durumları karşılaştırılarak incelenmiş ve generatör birimlerinin güç üretiminde bu kriter de göz önüne alınmıştır. Sistemin kararlılığı iyileştirilerek olası bir bozucu durum için sistemin daha dayanıklı hale gelmesinin mümkün olduğu gösterilmiştir. Kullanılan V-Q duyarlılık endeksi ile sistemin işletimine olumlu katkılarının olduğu gözlenmiştir.

Sistemin hem ekonomik hem de iyileştirilmiş bir kararlılığa sahip optimum işletme durumunun elde edilmesinde öğrenen automata yöntemi kullanılmıştır. Bu çalışmadaki gibi, farklı kontrol yapılarına sahip optimal çözümleri sağlayan toplam bir kontrol modeli oluşturarak analitik yoldan her birini sağlayan bir çözüm elde etmek zordur. Bu yöntem ile optimizasyon işlemi daha sade ve daha hızlı bir biçimde gerçekleştirilmiştir. Bu yönüyle yöntemin kullanılması çözüm için avantaj sağlamaktadır.

Tez çalışmasında öğrenen automatanın kullanımı ile tez içeriğinin özgün olması da sağlanmıştır. Değişken yapılı öğrenen automata kullanılarak, automata için belirlenen aksiyonlardan en iyisi seçilmiştir. En iyi aksiyon ile sistem için aynı anda hem ekonomik bir maliyet ve hem de daha iyi bir kararlılık için optimum işletim noktasına ait generatörlerin üretecekleri güç değerleri belirlenmiştir. Yöntem ile problemin optimizasyonunun elde edilmesinde kolaylık sağlanmıştır.

Bu tez çalışmasında evrimsel programlama, genetik algoritma ve öğrenen automata yöntemlerinin kullanıldığı metoda uygun olarak MATLAB programından yararlanılarak bir bilgisayar yazılımı ortaya konmuştur ve EK 1'de gerçekleştirilen bu yazılım verilmiştir.

Sistemin kararlılığının kıyaslanması sırasında kullanılan V-Q duyarlılık endeksi için gerekli olan mevcut sisteme ait jakobiyen matrisi bir matlab güç sistem simülasyon paketi olan MATPOWER 3.1.b2'nin modifiye edilmesi ile elde edilmiştir. Yine güç akış analizleri de bu programın kullanımı ile sağlanmıştır.

Yukarıda belirtildiği üzere, tez çalışmasında birim yüklenme problemi, evrimsel programlama ve genetik algoritma hibrit bir şekilde kullanılarak çözüm orijinal olarak elde edilmiştir. Minimum devrede veya devre dışında bulunma süreleri evrimsel programlama ile basit bir şekilde kodlanmıştır ve başarılı bir şekilde algoritmaya dahil edilmiştir.

Sayısal uygulamalarla ilgili olarak şunlar belirtilebilir:

Önerilen metodu gerçekleştirmek için yapılan ilk uygulamada, 1 saatlik bir periyot için ekonomik yük dağıtımının çözümünde genetik algoritma kullanılmıştır. Genetik algoritmanın çözümünde popülasyondaki birey sayısı 40 ve mutasyon değeri de 0,01 olarak belirlenmiştir. Sistemdeki generatörlerin ekonomik olarak üretecekleri güç miktarları ve buna bağlı maliyet değeri elde edilmiştir (Tablo 7.4). Daha sonra sistemin en iyi kararlı işletim noktasına sahip durumu belirlemek için analizler yapılmıştır. Ayrı ayrı en ekonomik ve en iyi kararlı işletim noktaları belirlendikten sonra sistemin bu iki kriteri de belli eşik değerleri ile sağlayan optimum işletim noktası öğrenen automata ile belirlenmiştir. Öğrenen automata kullanımı ile elde edilen en iyi aksiyon ve bu aksiyona ait generatörlerin üretecekleri güç değerleri verilmiştir (Tablo 7.6). Bu değerler sistemin ekonomik ve kararlı çalışmasını sağlayan işletim sistemine ait değerler olarak elde edilmiştir. Optimum işletim durumu için ekonomik yük dağıtım maliyeti, kararlılık analizi yapılmadan önce elde edilen değerinden artış göstermiştir ancak sistemin kararlılığı da daha elverişli hale gelmiştir.

İkinci uygulama için öncelikle en ekonomik maliyet açısından hangi generatörlerin devrede olmaları gerektiğini belirlemek için birim yüklenme problemi çözülmüştür. Bu problemin çözümünde evrimsel programlama kullanılmıştır. Evrimsel programlama problemini gerçekleştirirken popülasyon birey sayısı 40 olarak belirlenmiştir ve her birey mutasyona uğratılmıştır. Problemin çözümü ile birim yüklenme matrisi elde edilmiştir (Tablo 7.11). Generatörlerin işletme planı belirlendikten sonra, uygulama 1'deki gibi genetik algoritma kullanılarak generatörlerin üretecekleri güç değerleri T=12 periyotlu bir işletme için elde edilmiştir (Tablo 7.12). En ekonomik maliyet değeri belirlendikten sonra her işletim saati için kararlılık analizleri yapılmış ve sistemin en elverişli kararlı işletim noktaları belirlenmiştir. Öğrenen automata yöntemi ile bu iki kriter göz önüne alınarak sistemin ekonomiklik ve

kararlılık açısından optimum işletim noktaları her saat için elde edilmiştir (Tablo 7.33). Bu durumda da birim yüklenme maliyetinde bir artış gözlenmektedir ancak daha elverişli bir kararlılık yapısına sahip olmuştur. Bu değerler sistem için optimum işletme durumuna ait ekonomik ve kararlılık değerleri olarak belirlenmiştir.

Yukarıda verilmiş olan sayısal uygulamaların sonucunda farklı işletim durumları için V-Q duyarlılık endeksi ile sistemin kararlılığının kıyaslanması analizlerinde, kararlılığın iyileştirilmesi ile ekonomik maliyetin birbirine paralel bir şekilde değiştiği görülmüştür. Diğer bir deyişle, kararlılığın iyileştirilmesi için ekonomik işletim sisteminde maliyetin arttığı gözlenmiştir. Bu durum enerji planlama ve işletim çalışmalarında göz önüne alınmalı ve yapılacak çalışmalarda kabul edilebilir değerler içerisinde sistemin çalıştırılması gereğini ortaya koymaktadır.

Bu çalışma ile hem ekonomik hem de kararlılık açısından bir işletim sistemi elde edilmesine ilişkin analiz metodu önerilmiştir. Generatör güçlerinin dışında sistemin topolojisine bağlı olarak meydana gelecek hat kayıplarının bu sistemin içerisinde değerlendirilebilmesi mümkündür. Böylece bundan sonraki çalışmalarda yapılacak analizler için sistem kayıplarının dahil edilmesi sağlanabilir.

V-Q duyarlılık endeksinin kullanılması, farklı işlerim durumları için daha kararlı bir sistem elde edilmesini sağlamaktadır. Bu analizin önerilen yöntemle sürekli yapılması ekonomik ve daha kararlı sistem elde edilmesi bakımından gerekecektir.

KAYNAKLAR

Anderson, P.M., FOUAD, A.A., (1994), "Power System Control and Stability", Revised Printing, IEEE Press.

Aoki, K. Satoh, T. ve Itoh, M., (1987), "Unit Commitment in Large Scale Power Systems Including Fuel Constrained Thermal and Pumped Storage Hydro", IEEE Trans. Power Syst., vol. PWRS-2, 1077–1084.

Artaç, T., (2003), "Genetik Algoritmalar ile Dağıtım Şebekelerinin Optimum Tasarımı", Y.L. Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Aso, H., Kimura, M., (1976), "The Structures of Automata to Adapt to an Unknown Enviroment", IEEE Trans. Syst., Man and Cybern., SMC-6, 494-504.

Atkinson, R.C., Bower, G.H., Crothers, E.J., (1965), "An Introduction to Mathematical Learning Theory", New York: Wiley,.

Aygen, Z.E., (2002), "Elektrik Enerji Sistemlerinde Genetik Algoritma Kullanarak Optimizasyona Yeni Bir Yaklaşım", Doktora Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Baba, N., (1983), "The Absolutely Expedient nonlinear Reinforcement Schemes under the Unknown Multi-Teacher Environment", IEEE Transactions on Systems, Man and Cyber., SMC-13, 100-108.

Bertsekas, D.P., Lauer, G.S., Sandell, N.R., Posbergh, T.A., (1983), "Optimal Short-Term Scheduling of Large-Scale Power Systems", IEEE Transactions on Automatic Control, vol. AC-28, no.1, 1-11.

Burns, R.M. ve Gibson, C.A., (1975), "Optimization of Priority Lists for a Unit Commitment Program", Proc. IEEE Power Eng. Soc. Summer Meeting,.

Bush, R.R. ve Mosteller, F., (1958), "Stochastic Models for Learning", New York: Wiley.

Castillo, E. ve Alvarez, E., (1991), "Expert Systems: Uncertainity and Learning", U.K.: Computational Mechanics Publications.

Castellanos, F., Mohammed, A.F., (2003), "Voltage stability and reactive power compensation on the T&TEC system", IEEE Proceedings SoutheastCon, 134 – 139.

Chandrasekharan, B. ve Shen, D.W.C., (1968) "On Expediency and Convergence in Variable Structure Stochastic Automata", IEEE Transactions on System Science and Cybernetics, vol. SSC-5, 145-149.

Chandrasekharan, B. ve Shen, D.W.C., (1969), "Stochastic Automata Games", IEEE Transactions on System Science and Cybernetics, vol.SSC-5, 145-149.

Cheng, C.P., Liu, C.W. ve Liu C.C., (2000), "Unit Commitment by Lagrangian Relaxation and GEnetic Algorithms", IEEE Trans. On Power Systems, vol.15, no.2, 707-714.

Chowdhury, N. ve Billinton, R., (1990), "Unit Commitment in Interconnected Generating Systems Using a Probabilistic Technique", IEEE Trans. Power Syst., vol.5, 1231–1238.

Christober, C., Rajan, C.C.A., (2004) "An evolutionary programming based simulated annealing method for unit commitment problem with cooling-banking constraints",

Proceedings of the IEEE INDICON India Annual Conference, 435 – 440.

Cohen, A.I., ve Wan, S.H., (1987), "A Method for Solving The Fuel Constrained Unit Commitment Problem", IEEE Trans. Power Syst., vol. PWRS-2, 608–614.

Contreras- Hernandez, E. J., Cedeno-Maldonado, J. R., (2006), "A Sequential Evolutionary Programming Approach to Profit-Based Unit Commitment", IEEE/PES Transmission & Distribution Conf. and Exp., 1 – 8.

Cover, T. M. ve Hellman, M.E., (1970), "The Two-Armed Bandit Problem with Time Invariant Finite Memory", IEEE Trans. On Information Theory, vol.16, no.2, 185-195.

Dasgupta, D. ve McGregor, D.R., (1994), "Thermal Unit Commitment Using Genetic Algorithms", Proc. Inst. Elect. Eng., Gen. Transm. Dist., vol. 141, 459–465.

Dorigo, M., Sttzle, T., (2004), "Ant Colony Optimization", Bradford Books.

Elgerd, O.I., (1983), "Electric Energy Systems Theory", McGraw-Hill.

Fogel, L.J., Owens, A.J. ve Walsh M.J., (1966), "Artificial Intelligence Through Simulated Evolution", New York:Jhon Wiley & Sons.

Fu, K.S., (1967), "Stochastic Automata as Models of Learning Systems", in Computer and Information Sciences II, J.T. Lou, New York: Academic.

Garcia, H.E., Ray, A. ve Edwards, R.M., (1991), "Reconfigurable Control of Power Plants Using Learning Automata", IEEE Control Syst, vol .11, no.1, 85-92.

Geçkinli, E.A., (1989), "Metalografi", İTÜ Kütüphanesi Sayı 1391, Teknik Üniversite Matbaası, İstanbul.

Gilbert, V., Thibault, J., Najim, K., (1992), "Learning Automata for the Control and Optimization of a Continuous Stirred Tank Fermenter", IFAC Symposium on Adaptive systems in Control and Signal Processing, Grenoble, France.

Goldberg, D.E., (1989), "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, USA.

Guan, X., Luh, P.B., Yan, H., Amalfi, J.A., (1992), "An Optimization-Based Method for Unit Commitment", Electrical Power & Energy Systems, vol.14, no.1, 9-17.

Happ, H.H., Johnson, R.C., ve Wright, W.J., (1971), "Large scale hydro-thermal unit commitment-method and results", IEEE Trans. Power App. Syst., vol. PAS-90, 1373–1384.

Hara, K., Kimura, M. ve Honda, N., (1966), "A method for Planning Economic Unit Commitment and Maintenance of Thermal Power Systems", IEEE Trans. Power App. Syst., vol. PAS-85, 427–436, May.

Haupt, R.L., Haupt, S.E., (2004), "Practical Genetic Algorithms", John Wiley & Sons, Inc., New Jersey.

Holland, J.H., (1975), "Adaptation in Natural and Artificial systems", Ann Arbor: University of Michigan Press, USA.

Hong Y.H., Pan, C.T., Lin, W., (1997), "Fast calculation of a voltage stability index of power systems", IEEE Transactions on Power Systems, Vol.12, 1555 – 1560.

Huang, S.J., (2001), "Enhancement of Hydroelectric Generation Scheduling Using Ant Colony System Based Optimization Approaches", IEEE Trans. Energy Conversion, vol.16, 296–301.

Indulkar, C.S., Viswanathan, B., Venkata, S.S., (1989), "Maximum Power Transfer Limited by Voltage Stability in Series and Shunt Compensated Schemes for AC Transmission Systems", IEEE Transaction on Power Delivery, 2(4), 1246-1252.

Juste, K.A., Kita, H., Tanaka, E. ve Hasegawa, J., (1999), "An Evolutionary Programming Solution to The Unit Commitment Problem", IEEE Trans. Power Syst., vol.14, 1452–1459.

Kazarlis, S.A., Bakirtzis, A.G., Petridis, V., (1996), "A Genetic Algorithm Solution to The Unit Commitment Problem", IEEE Trans. Power Syst., vol.11, 29–36.

Kerr, R.H., Scheidt, J.L., Fontana, A.J., and Wiley, J.K., (1966), "Unit Commitment", IEEE Trans. Power App. Syst., vol. PAS-85, pp. 417–421.

Kundur, P., (1994), "Power System Stability and Control", Mc Graw-Hill, Inc.

Kushner, H.J., Thathachar, M.A.L. ve Lakshmivarahan, S., (1972), "Two-state Automaton:a Counterexample", Dec. 1979; Appendix to Viswanathan R, and K.S. Narendra, "A Note on the Linear Reinforcement Scheme for Variable Structure Stochastic Automata", IEEE Transactions on Systems, Man and Cybernetics, SMC-2, 292-294.

Lee, B. H., Park, J. K., (1998), "Multiple objective optimal operation of power system using learning automata", Proceedings of EMPD '98, vol.1, 247-252.

Lee, B. H., Lee, K.Y., (2005), "Application of S-model learning automata for multi-objective optimal operation of power systems", IEE Proceedings on Generation, Transmission and Distribution, vol.152, 295-300.

Lee, F.N., (1988),"Short-term unit commitment—a new method", IEEE Trans. Power Syst., vol.3, 421–428.

Lee, F.N., (1991), "The Application of Commitment Utilization Factor (CUF) to Thermal Unit Commitment", IEEE Trans. Power Syst., vol.6, 691–698.

Li, C., Johnson, R.B., Svoboda, A.J., Tseng, C. ve Hsu, E., (1998), "A Robust Unit Commitment Algorithm for Hydro-Thermal Optimization", IEEE Trans. Power Syst., vol.13, 1051–1056.

Lin, W.M., Cheng, F.S. ve Tsay, M.T., (2002), "An Improved Tabu Search for Economic Dispatch with Multiple Minima", IEEE Trans. Power Syst., vol. 17, 108–112.

Lof, P.A., Smed, T., Andersson, G., Hill, D.J., (1992), "Fast calculation of a voltage stability index", IEEE Transactions on Power Systems, Vol.7, 54 – 64.

Lowery, P.G., (1966), "Generating unit commitment by dynamic programming", IEEE Trans. Power App. Syst., vol. PAS-85, 422–426.

Ma, H. ve Shahidehpour, S.M., (1999), "Unit Commitment With Transmission Security and Voltage Constraints", IEEE Trans. Power Syst., vol.14, 757–764.

Maifeld, T.T. ve Sheble, G.B. (1996), "Genetic-based unit commitment algorithm", IEEE Trans. Power Syst., vol.11, 1359–1370.

Man, K.F., Tang, K.S. ve Kwong, S., (1996), "Genetic Algorithms: Concepts and

Applications", IEEE Trans. Of Industrial Electronics, vol.43, no.5, 519-533.

Mantawy, A.H., Abdel-Magid, Y.L., Selim, S.Z., (1998), "A Simulated Annealing Algorithm for Unit Commitment", IEEE Trans. Power Syst., vol.13, 197–204.

Mantawy, A.H., Abdel-Magid, Y.L., ve Selim, S.Z., (1998), "Unit Commitment by Tabu Search", Proc. Inst. Elect. Eng., Gen. Transm. Dist., vol.145, no.1, 56–64.

Marsh, C., Gordon, T. J., and Q. H. Wu, (1993), "Stochastic Optimal Control of Active Vehicle Suspensions using Learning Automata", Proceedings I. Mech. Eng. Part I, Journal of Systems and Control Engineering, vol.207, 143-152.

Marsh, C., Gordon, T.J., Wu, Q.H., (1993), "Stochastic Optimal Control of Active Vehicle Suspensions using Learning Automata", Proceedings I. Mech. Eng. Part I, Journal of Systems and Control Engineering, vol.207,143-152.

McLaren, R.W., (1966), "A Stochastic Automaton Model for Synthesis of Learning Systems", IEEE Transactions on System Science and Cybernetics, SSC-2, 109-114.

Merlin, A. ve Sandrin, P., (1983), "A New Method for Unit Commitment at Electricite De France", IEEE Trans. Power App. Syst., vol. PAS-102, 1218–1225.

Miller, Timothy J. E., ed., (1982), "Reactive Power Control in Electric Systems", John Wiley and Sons, Inc., N.Y.

Momoh, J.A., (2001), "Electric Power System Applications of Optimisation", Marcel Dekker Inc., New York.

Mori, H. ve Usami, T., (1996), "Unit Commitment Using Tabu Search with Restricted Neighborhood", in Proc. Intell. Syst. Applicat. Power Syst., 422–427.

Mori, H., ve Matsuzaki, O., (2000), "Embedding The Priority into Tabu Search for Unit Commitment", in Proc. IEEE Winter Meeting.

Mukhtari, S., Singh, J., Wollenberg, B., (1988), "A Unit Commitment Expert System", IEEE Trans. Power Syst., vol.3, 272–277.

Najim, K. ve Poznyak, A.S., (1994), "Learning Automata: Theory and Application", Tarrytown, NY: Elsevier Science Ltd.

Narendra, K.S. ve Thathachar, M.A.L., (1974), "Learning Automata:A Survey", IEEE Transactions in Systems, Man and Cybernetics, vol. SMC-4, no.4.

Narendra, K.S. ve Thathachar, M.L., (1989), "Learning Automata: An Introduction", Englewood Cliffs, NJ: Prentice Hall.

Nieva, R., Inda, A. ve Guillen, I., (1987), "Lagrangian Reduction of Search-Range for Large-Scale Unit Commitment", IEEE Trans. Power Syst., vol. PWRS-2, 465–473.

Oommen, B.J. ve Croix, E.V., (1994), "Graph Partitioning using Learning Automata", Technical Report TR-250, School of Computer Science, Carleton University, Ottawa, Canada.

Oommen, B.J. ve Croix, E.V., (1994), "String Taxonomy using Learning Automata", Technical Report TR-234, School of Computer Science, Carleton University, Ottawa, Canada.

Orero, S.O. ve Irving, M.R., (1997), "Large Scale Unit Commitment Using a Hybrid Genetic Algorithms", Int. J. Elect. Power Energy Syst., vol.19, no.1, 45–55.

Özdeş, A., (2003), "Radyal Enerji Dağıtım Sistemlerinde Yeniden Yapılandırma Problemlerine Genetik Algoritma Yaklaşımı", Doktora Tezi, Yıldız Teknik Üniversitesi, İstanbul.

Padhy, N.P., (2004), "Unit Commitment-A Bibliographical Survey", IEEE Trans. Power Syst., Vol.19, No.2, 1196-1205.

Peng, J. ve Williams, R.J., (1993), "Efficient Learning and Planning within the Dyna Framework", Adaptive Behavior, vol. 1, no.4, 437-454.

Rajan, A., Mohan, C.C., ve Manivannan, M.R., (2002), "Neural based tabu search method for solving unit commitment problem," in Proc. 5th Int. Conf. Power Syst. Manage., 180–185.

Rajan, C.C.A., Mohan, M.R., (2004), "An evolutionary programming-based tabu search method for solving the unit commitment problem", IEEE Transactions on Power Systems, vol.19, 577 – 585.

Rajan, C.C.A., Mohan, M.R., (2004), "An evolutionary programming method for solving the unit commitment problem", IEEE Power Engineering Society General Meeting, vol.1, 1149.

Salam, M.S., Abdul-Razak Hamdan, A.R., Khalid Mohamed Nor, K.M., (1991), "Integrating an Expert System into a Thermal Unit-Commitment Algorithm", Proc. Inst. Elect. Eng. C, vol. 138, 553–559.

Saneifard, S., Prasad, N.R., ve Smolleck, H.A., (1997), "A Fuzzy Logic Approach to Unit Commitment", IEEE Trans. Power Syst., vol.12, 988–995.

Sapmaz, M.E., (2004), "Generatör Birim Katkı Sorununun Genetik Algoritmalar ile Çözülmesi", Y.L. Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Sasaki, H., Watanabe, M., Yokoyama, R., (1992), "A Solution Method of Unit Commitment by Artificial Neural Networks", IEEE Trans. Power Syst., vol.7, 974–981.

Sekine, Y. Ve Ohtsuki, H., (1990), "Cascaded Voltage Collapse", IEEE Trans. on Power Systems, 1(5), 250-256.

Sequeira, J., Bispo, C. ve Sentieiro, J., (1992), "Distributed Control of a Flexible Manufacturing Plant Using Learning Automata", Proceedings of the IMACS/IFAC Intern. Symp., Parallel and Distributed Computing in Engineering Systems, Corfu, Greece, 327-332.

Shanthi, V., Jeyakumar, A.E., (2004), "Unit Commitment by Genetic Algorithms", IEEE PES Power Syst. Conf. And Exp., vol.3, 1329-1334.

Sheble, G.B. et al., (1996), "Unit Commitment by Genetic Algorithm with Penalty Methods and a Comparison of Lagrangian Search and Genetic Algorithm Economic Dispatch Example", Int. J. Elect. Power Energy Syst., vol.18, no.6, 339–346.

Shoults, R.R., Chang, S.K., Helmick, S. ve Grady, W.M., (1980), "A Practical Approach to Unit Commitment, Economic Dispatch and Savings Allocation for Multiple-Area Pool Operation with Import/Export Constraints", IEEE Trans. Power App. Syst., vol. PAS-99, 625–635.

Sisworahardjo, N.S. ve El-Kaib, A.A., (2002), "Unit Commitment Using Ant Colony Search Algorithm", in Proc. 2002 Large Eng. Syst. Conf. Power Eng., 2–6.

Snyder, W.L., Powell, H.D. ve Rayburn, J.C., (1987), "Dynamic Programming Approach to

Unit Commitment", IEEE Trans. Power Syst., vol.2, 339–347.

Sood, Y.R., Padhy, N.P. ve Gupta, H.O., (2003), "Discussion on Optimal Power Flow by Enhanced Genetic Algorithms", IEEE Trans. Power Syst., vol. 18, 1219.

Stevenson, W.D., (1988), "Elements of Power System Analysis", McGraw-Hill Ltd., 1988.

Sun, L., Zhang, Y., Jiang, C., (2006), "A Matrix Real-Coded Genetic Algorithm to The Unit Commitment Problem", Electric Power Systems Research, 76, 716–728.

Swarup, K.S., Yamashiro, S., (2002), "Unit Commitment Solution Methodology Using Genetic Algorithm," IEEE Trans. Power Syst., vol.17, 87–91.

Şenel, S., (2006), "Genetik Algoritmalar ile Enerji İletim Kayıpları ve Saatlik Yakıt Giderleri Optimizasyonu", Y.L. Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Tacer, E., (1990), "Enerji Sistemlerinde Kararlılık", İTÜ Kütüphanesi, Sayı 1407, E-E Fakültesi Ofset Baskı Atölyesi, İstanbul.

Takriti, S., Birge, J.R. ve Long, E., (1996), "Astochastic Model for The Unit Commitment Problem", IEEE Trans. Power Syst., vol. 11, 1497–1508.

Taylor, C.W., (1992), "Power System Voltage Stability", McGraw-Hill, Inc.

Tong, S.K. ve Shahidehpour, S.M., (1989), "Combination of Lagrangian-Relaxation and Linear-Programming Approaches for Fuel-Constrained Unit Commitment Problems", Proc. Inst. Elect. Eng., Gen. Transm. Dist., vol.36, 162–174.

Tseng, C.L., (1996), "On Power System Generation Unit Commitment", Ph.D. Thesis, University of California, Berkeley.

Tsetlin, M.L., (1961), "On the Behaviour of Finite Automata in Random Media", Automat. Telemekh., vol.22, 1345-1354.

Tsetlin, M.L., (1973), "Automaton Theory and Modeling of Biological Systems", Mathematics in Science and Engineering, New York: Academic Press.

Uzunoğlu, M., (2000), "Nonlineer Yükler İçeren Elektrik Güç Sistemlerinde Gerilim Kararlılık Analizi", Doktora Tezi, Yıldız Teknik Üniversitesi, İstanbul.

Ünsal, C., (1997), "Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles Approach", Ph. D. Thesis, Virginia Polytechnic Institute and State University.

Varshavski, V.I. ve Vorontsova, I.P., (1963), "On the Behavior of Stochastic Automata with Variable Structure", Automat. Telemekh., vol. 24, 253-360.

Viswanathan, R. ve Narendra, K.S., (1972), "A Note on the Linear Reinforcement Scheme for Variable Structure Stochastic Automata", IEEE Man and Cybern., SMC-2, 292-294.

Walsh, M.P. ve Malley, M.J., (1997), "Augmented Hopfield Network for Unit Commitment and Economic Dispatch", IEEE Trans. Power Syst., vol. 12, 1765–1774.

Wang, C. ve Shahidehpour, S.M., (1993), "Effects of Ramp-Rate Limits on Unit Commitment and Economic Dispatch", IEEE Trans. Power Syst., vol. 8, 1341–1350

Wang, C. ve Shahidehpour, S.M., (1994), "Ramp-rate Limits in Unit Commitment and

Economic Dispatch Incorporating Rotor Fatigue Effect", IEEE Trans. Power Syst., vol.9, 1539–1545.

Wood, A.J. ve Wollenberg, B.F., (1996), "Power Generation, Operation and Control", New York: Wiley.

Wu, Q.H., (1992), "Learning Control for Turbogenerators", Proc. 27th Universities Power Engineering Conference, University of Bath, UK, 436-439.

Wu, Q.H., (1995), "Learning Coordinated Control of Power Systems Using Interconnected Learning Automata", Electrical Power & Energy Systems, vol. 17, no.2, 91-99.

Wu, Q.H., Hogg, B.W., Irwin, G.W., (1992), "A Neural Network Regulator for Turbogenerators", IEEE Trans. Neural Netw., vol.3, no.1, 95-100.

Wu, Q.H., Pugh, A.C., (1993), "Reinforcement Learning Control of Unknown Dynamic Systems", IEE Proc., D, Control Theory Appl., vol.140, no.5, 313-322.

Xing, W., Wu, F.F., (2002), "Genetic algorithm based unit commitment with energy contracts", Electrical Power and Energy Systems, vol.24, 329-336

Yalçın, M.A., (1995), "Enerji İletim Sistemlerinde Gerilim Kararlılığının Yeni Bir Yaklaşımla İncelenmesi", Doktora Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Yang, H., Yang, P., Huang, C., (1997), "A Parallel Genetic Algorithm Approach to Solving The Unit Commitment Problem: Implementation on The Transputer Networks", IEEE Trans. Power Syst., vol.12, 661–668.

Yang, H.T., Yang, P.C., ve Huang, C.L., (1996), "Evolutionary Programming Based Economic Dispatch for Units with Non-Smooth Fuel Cost Functions", IEEE Trans. Power Syst., vol. 11, 112–118.

EK - 1 Tez çalışmasında önerilen metoda dayalı bilgisayar programı yazılımı

```
%************************************************************************
%
% Birim Yüklenme Probleminin Çözümünde Kullanılan
% Evrimsel Programlama İçin Bilgisayar Yazılımı
%
% Program İsmi: Main_EP.m
%
%
%************************************************************************
%
% Alt Programların Tanıtılması:
%
% veriler.m         : Probleme ait verilerin girilmesi
% populasyon_EP.m   : İlk popülasyonun Oluşturulması
% maliyet_EP.m      : Bireylerin uygunluk değerlerinin belirlenmesi
% mut_EP            : Bireylerin mutasyona uğratılması
% MUPMDP.m          : Minimum devrede/devre dışında kalma sürelerine
%                     bağlı olarak birim yüklenme matrislerinin üretilmesi
% kisit_EP.m        : Sistemden talep edilen gücün sağlanması
% main_GA.m         : Genetik Algoritma ile ekonomik yük dağıtımının
%                     sağlanması
%
%************************************************************************

global PP CC in
global T Pmin Pmax a b c PD ST
global N nvar npop nmut popsize Nt
global keep M ma pa

veriler.m
% Durdurma kriterlerinin belirlenmesi
Iterasyon=50;      %maksimum iterasyon sayısı
MinMal=-9999999;   %minimum maliyet
Iter=0;
% İlk populasyonun oluşturulması
[UUC DD]=populasyon_EP;
% İlk popülasyona ait bireylerin uygunluk değerlerinin belirlenmesi
[cost UUZ Guc DDx]=maliyet_EP(UUC, DD);
costp=cost;
UUCp=UUZ;
DDp=DDx;
Gucp=Guc;
% İterasyona (üretime) baslanması
while Iter<Iterasyon
    Iter=Iter+1
% Bireylerin mutasyona uğratılması
[UUX DDX]=mutasyon_EP(UUC,DD);
% Mutasyon işlemi sonucunda oluşan bireylerin uygunluk değerleri
[cost UUZ Guc DDx]=maliyet_EP(UUX, DDX);
costm=cost;
UUCm=UUZ;
DDm=DDx;
Gucm=Guc;
% Ebeveyn ve yavru bireylerin tek popülasyonda birleştirilip uygunluk
% değerlerine göre sıralanması ve en iyilerin yeni popülasyona seçilmesi
cost=[costp costm];
UUC={UUCp{:} UUCm{:}};
DD={DDp{:} DDm{:}};
```

```
Guc={Gucp{:} Gucm{:}};
[cost,ind]=sort(cost);
[UUC{:}]=UUC{ind};
[DD{:}]=DD{ind};
[Guc{:}]=Guc{ind};
costp=cost(1:in);
UUCp=UUC(1:in);
DDp=DD(1:in);
Gucp=Guc(1:in);
%populasyonun minimum maliyet değerinin saklanması
minc(Iter)=min(costp);
%populasyonun ortalama maliyet değerinin saklanması
meanc(Iter)=mean(costp);
%Durdurma kriterinin kontrolü
if Iter>Iterasyon | costp(1)<MinMal
    break
end
[Iter costp(1)];
end

% Sonuçlara ait verilerin ekrana verilmesi
day=clock;
disp(datestr(datenum(day(1),day(2),day(3),day(4),day(5),day(6)),0))
disp(['popsize= ' num2str(in)])
disp(['İterasyonlar= ' num2str(Iter) 'En Iyi Maliyet= ' num2str(costp(1))])
disp(['En İyi Çözüm'])
disp([num2str(Gucp{1})])
figure(24)
iters=0:length(minc)-1;
plot(iters,minc,iters,meanc,'-')
xlabel('Iterasyonlar');ylabel('Maliyet')
text(0,minc(1),' En Iyi Maliyet Degeri ')
text(1,meanc(2),' Populasyonun Ortalama Degeri ')

% ************************************************************************
% Alt Program İsmi  : veriler.m
% ************************************************************************

% Generatör değerlerinin girilmesi
Pmin=input('Generatörlerin minimum güç limiti: ');
Pmax=input('Generatörlerin maksimum güç limiti: ');
% Her generatörün yakıt maliyetine ait katsayıların girilmesi
a=input('a katsayıları: ');
b=input('b katsayıları: ');
c=input('c katsayıları: ');
ST=input('Başlangıç maliyetleri: ');;
PDT=input('Sistemden talep edilen güç değerleri (T periyot): ');
PDR=input('Sistemden talep edilen rezerv güç değerleri (T periyot): ');
PD=PDT+PDR;
% Minimum devrede ve devre dışında kalma süreleri için generatörlerin
% işletme durumlarının belirlenmesi
PP=input('İşletim Durumu: ');
CC=input('Uygun işletim kümesi: ');
T=input('İşletim Zamanı: ');
in=input('EP için populasyon değeri: ');

%************************************************************************
```

```
%*************************************************************************
% Alt Program İsmi  : populasyon_EP
% *************************************************************************

function [UUC DD]=populasyon_EP
global in
for ic=1:in
n=0;
while ~ n
%Unit Commitment Matrisinin elde edilmesi
    [UC d]=MUPMDP;
%Kisitlarin degerlendirilmesi
    [kisit]=kisit_EP(UC);
    n=kisit;
end
UUC{ic}=UC;
DD{ic}=d;
end
UUC{:};
DD{:};

% *************************************************************************
% Alt Program İsmi  : MUPMDP.m
% *************************************************************************

function [UC d]=MUPMDP
global PP CC T in
bb=6;   % minUT ve minDT kisitlarini saglayan kumelerin sayısı
nn=4;   % p sayısı p=T/Nbit (burada 12/3=4)
nb=3;    %bit sayısı
for i=1:nb
    ii=ceil(rand*bb);
    UC(i,1:3)=PP{i,ii};
    d(i,1)=ii;
    for nx=2:nn
        ix=CC{i,ii};
        ii=ix(ceil(rand*length(CC{i,ii})));
        d(i,nx)=ii;
        UC(i,(nx-1)*nb+1:nx*nb)=PP{i,ii};
    end
end
UC; %Unit Commitment matrisi
d;  %indisler

% *************************************************************************
% Alt Program İsmi  : kisit_EP.m
% *************************************************************************

function [kisit]=kisit_EP(UC)
global T Pmin Pmax a b c PD

nb=3;
TT1=0;
TT2=0;
iter=0;
nd=zeros(1,T);
for t=1:T
    for i=1:nb
        TT1=TT1+UC(i,t)*Pmax(i);
        TT2=TT2+UC(i,t)*Pmin(i);
    end
    if (TT1 >= PD(t)) & (TT2 < PD(t))
```

```
        nd(t)=1;
        Toplam1(t)=TT1;
        Toplam2(t)=TT2;
    end
    TT1=0;
    TT2=0;
end
kisit=prod(nd);


% *************************************************************************
% Alt Program İsmi  : maliyet_EP.m
% *************************************************************************

function [cost UUZ Guc DDx]=maliyet_EP(UUZ, Dx)
global in T
for icx=1:in
    UC=UUZ{icx};
    DDx{icx}=Dx{icx};
    for t=1:T
        [P M]=main_ED(UC,t);
        Gucler(t,:)=P;
        TP(t)=sum(P);
        Maliyet(t)=M;
    end
    TP;
    Gucler;
    Maliyet=sum(Maliyet);
    Guc{icx}=Gucler;
    cost(icx)=Maliyet;
end
[cost,ind]=sort(cost);
[UUZ{:}]=UUZ{ind};
[Guc{:}]=Guc{ind};
[DDx{:}]=DDx{ind};
cost;

% *************************************************************************
% Alt Program İsmi  : mutasyon_EP.m
% *************************************************************************

function [UUX DDX]=mutasyon_EP(UUC,DD)
global PP CC in

popi=in;
nn=4;
nb=3;
mut=ceil(1+(nn-1)*rand(popi,nb));
for ia=1:popi
    n=0;
%Mutasyona ugrayacak UC satırlarının indisleri belirleniyor
    nmut=mut(ia,:);
    UC=UUC{ia};
    D=DD{ia};
    while ~ n
        for i=1:nb
            ii=D(i,nmut(i)-1);
            aa=nmut(i);
            for nx=aa:nn
                ix=CC{i,ii};
                ii=ix(ceil(rand*length(CC{i,ii})));
                D(i,nx)=ii;
```

```
                UC(i,(nx-1)*nb+1:nx*nb)=PP{i,ii};
            end
        end
        [kisit]=kisit_01(UC);
        n=kisit;
    end
    UUX{ia}=UC;
    DDX{ia}=D;
end
UUX{:};
DDX{:};

%*************************************************************************
% Ekonomik Güç Dağıtımı Probleminin Çözümünde Kullanılan
% Genetik Algoritma İçin Bilgisayar Yazılımı
%
% Program İsmi: Main_GA.m
%
% %*************************************************************************
% Alt Programların Tanıtılması:
%
% malfonk_GA.m      : Amaç fonksiyonu
% populasyon_GA.m   : İlk popülasyonun oluşturulması
% cift_GA           : Çiftlestirilecek bireylerin belirlenmesi
% capraz_GA         : Çaprazlamanın yapılması
% mutfonk_GA        : Bireylerin mutasyona uğratılması
%
%*************************************************************************

function [P M]=UC_3(UC,t)
global T Pmin Pmax a b c PD ST
global N nvar npop nmut popsize Nt
global keep M ma pa

ff='malfonk_GA';       %Amac fonksiyonu
npar=3;      %Degisken sayisi
N=npar;
nvar=N;
% Durdurma Kriteri
maxit=50;      %Maksimum iterasyon sayısı
mincost=-9999999;    %minimum maliyet (cost)
EE=5e-1;
% GA Parametreleri
popsize=40;         %Popülasyon sayisi
mutrate=0.01;       %Mutasyon oranı
selection=0.5;      %Secim oranı
nbits=1;            %Her degiskenin sahip oldugu bit sayisi
Nt=npar;
keep=floor(selection*popsize);      %Saklanacak populasyon sayisi
nmut=ceil((popsize-1)*Nt*mutrate);  %Toplam Mutasyon Sayısı
M=ceil((popsize-keep)/2);           %Çiflestirme Sayisi
% İlk popülasyonun oluşturulması
iga=0;
par=populasyon_GA(popsize,npar,PD,Pmin,Pmax,t,UC);
cost=feval(ff,par,t,UC);
[cost,ind]=sort(cost);    r
par=par(ind,:);    %Populasyon en iyi maliyet degerlerine gore siralanıyor
minc(1)=min(cost);       %populasyonun minimum maliyetini saklar
meanc(1)=mean(cost);     %populasyonun ortalama maliyetini saklar
% İterasyona (üretime) baslama
while iga<maxit
    iga=iga+1;
```

```
% Çiftlestirilecek bireyler belirleniyor
[ma pa]=cift_GA(keep,M);
% Tek nokta çaprazlaması kullanılarak çiftleştirme yapılıyor
[par]=capraz_GA(keep,M,par,Nt,ma,pa,npar,popsize,PD,t,UC);
% Popülasyondaki bireylere mutasyon işleminin uygulanması
[par]=mutfonk_GA(par,nmut,Nt,popsize,Pmin,Pmax,PD,t,UC);
% Düzeltmelerin yapılması
[f g]=size(par);
    for ii=1:f
        if sum(par(ii,:))<PD(t)
        par(ii,:)=9999999*ones(1,npar);
        end
    end
% Maliyet (fitness fonksiyonu) için populasyonun tekrar degerlendirilmesi
cost=feval(ff,par,t,UC);
% Maliyet değerlerinin sıralanması
[cost,ind]=sort(cost);
par=par(ind,:);
% İstatistiklerin yapılması (for a single nonaveraging run)
minc(iga+1)=min(cost);
indc=find(cost<99999);
meanc(iga+1)=mean(cost(indc));
AA=abs([meanc(iga)-minc(iga+1)]);
% Durdurma Kriteri
if iga>maxit | AA<EE
    break
end
[iga cost(1)];
end

P=par(1,:);
M=cost(1);

% ************************************************************************
% Alt Program İsmi  : malfonk_GA.m
% ************************************************************************

function TC=malfonk_GA(par,t,UC)
global T Pmin Pmax a b c PD ST N

Ux=zeros(N,T+1);
Ux(:,2:T+1)=UC;
TC=0;
    for i=1:N
        P=par(:,i);
        ft=a(i)+b(i).*P+c(i).*(P.^2);
        f=[ft+ST(i)*(1-Ux(i,t))]*UC(i,t);
        TC=TC+f;
        P=[];
    end
TC;

% ************************************************************************
```

```
% ************************************************************************
% Alt Program İsmi  : populasyon_GA.m
% ************************************************************************

function Pop=populasyon_GA(t,UC)
global T Pmin Pmax a b c PD ST N nvar npop

rnd=0:0.1:1;
rn=length(rnd);
for jj=1:npop
  n=0;
  while ~ n
     for ii=1:nvar
        P(ii)=[Pmin(ii)+((Pmax(ii)-Pmin(ii))*rnd(ceil(rn*rand)))]*UC(ii,t);
        Popx(jj,ii)=P(ii);
     end
     if sum(Popx(jj,:))>=PD(t)
        Pop(jj,:)=Popx(jj,:);
        n=n+1;
     end
  end
end

Pop;

% ************************************************************************
% Alt Program İsmi  : cift_GA.m
% ************************************************************************

function [ma pa]=cift_GA
global keep M

prob=flipud([1:keep]'/sum([1:keep]));   %Rank weighting
odds=[0 cumsum(prob(1:keep))'];         %Toplam olasılık değerleri
pick1=rand(1,M);     %Birinci çiftleri olusturacak olasılıklar atandı
pick2=rand(1,M);     %İkinci çiftleri olusturacak olasılıklar atandı
ic=1;
while ic<=M                 %Çiftlestirme sayısına kadar devam edecek
    for id=2:keep+1
        if pick1(ic)<=odds(id) & pick1(ic)>odds(id-1)
            ma(ic)=id-1;    %Anne bireyleri elde ediliyor
        end
        if pick2(ic)<=odds(id) & pick2(ic)>odds(id-1)
            pa(ic)=id-1;    %Baba bireyleri elde ediliyor
        end
    end
    ic=ic+1;
end
ma;
pa;

% ************************************************************************
% Alt Program İsmi  : capraz_GA.m
% ************************************************************************

function [par]=capraz_GA(par,t,UC)
global PD N nvar npar nmut popsize Nt
global keep M ma pa

ix=1:2:2*M;
xp=ceil(rand(1,M)*Nt);      %Caprazlama noktaları belirleniyor
r=rand(1,M);    %çaprazlama parametresi
```

```
for ic=1:M
    iterc=0;
    nc=0;
    while ~nc
        iterc=iterc+1;
        xy=par(ma(ic),xp(ic))-par(pa(ic),xp(ic));
        par(keep+ix(ic),:)=par(ma(ic),:);
        par(keep+ix(ic)+1,:)=par(pa(ic),:);
        par(keep+ix(ic),xp(ic))=par(ma(ic),xp(ic))-ceil(r(ic).*xy);
        par(keep+ix(ic)+1,xp(ic))=par(pa(ic),xp(ic))+ceil(r(ic).*xy);
        if xp(ic)<npar
            parx(keep+ix(ic),:)=[par(keep+ix(ic),1:xp(ic))...
                par(keep+ix(ic)+1,xp(ic)+1:npar)];
            parx(keep+ix(ic)+1,:)=[par(keep+ix(ic)+1,1:xp(ic))...
                par(keep+ix(ic),xp(ic)+1:npar)];
            par(keep+ix(ic),:)=parx(keep+ix(ic),:);
            par(keep+ix(ic)+1,:)=parx(keep+ix(ic)+1,:);
        end
        A=sum(par(keep+ix(ic),:));
        B=sum(par(keep+ix(ic)+1,:));
        if (A >=PD(t)) & (B >=PD(t))
            nc=nc+1;
        elseif iterc >100
            nc=nc+1;
        end
    end
end
par=par(1:popsize,:);

% ************************************************************************
% Alt Program İsmi  : mutfonk_GA.m
% ************************************************************************

function [par]=mutfonk_GA(par,t,UC)
global Pmin Pmax PD
global N nvar npop nmut popsize Nt

mrow=sort(ceil(rand(1,nmut)*(popsize-1))+1);  %Mutasyona ugrayacak satır
mcol=ceil(rand(1,nmut)*Nt);             %Mutasyona ugrayacak sütun

iterm=0;
nm=0;
for ii=1:nmut
    while ~nm
        iterm=iterm+1;
        par(mrow(ii),mcol(ii))=ceil([(Pmax(mcol(ii))-Pmin(mcol(ii)))...
            *rand+Pmin(mcol(ii))])*UC(mcol(ii),t);
        if sum(par(mrow(ii),:))>=PD(t)
            nm=nm+1;
        elseif iterm > 100
            nm=nm+1;
        end
    end
end

par;


%************************************************************************
```

```
%*************************************************************************
%
% Öğrenen Automata Yöntemi İçin Bilgisayar Yazılımı
%
% Program İsmi: Main_LA.m
%
%*************************************************************************

% Verilerin girilmesi
r=input('Aksiyon Sayısı: ');
c=input('Penaltı Olasılıkları: ');
a=input('Ödül Parametresi: ');
b=input('Ceza Parametresi: ');
EN=input('İterasyon Sayısı: ');

iter=0;
M(1)=(1/r)*sum(c);
P=(1/r)*ones(1,r);
p=P;

i=ceil(rand*r);
olda=i;

for n=1:EN
    iter=iter+1;
    if c(olda)>rand
        Bta=1;
    else
        Bta=0;
    end

    if Bta==1
        nn=b/(r-1);
        p=p*(1-b)+nn;
        p(olda)=p(olda)-nn;
    elseif Bta==0
        p=p*(1-a);
        p(olda)=p(olda)+a;
    end

    P(n+1,:)=p;
    M(n+1)=sum(p.*c);
    ac=length(find(rand>cumsum(p)))+1;
    olda=ac;
end

P;
M;

% *************************************************************************
% Alt Program İsmi  : c_hesabi.m
% *************************************************************************

K_ED=input('Maliyet Performans İndisleri: ');
K_GK=input('Kararlılık Performans İndisleri');

n1=input('Maliyet için eşik değeri: ');
n2=input('Kararlılık için eşik değeri: ');

K_ED_N=min(K_ED)./K_ED;
K_ED_NN=[K_ED_N-min(K_ED_N)]/[max(K_ED_N)-min(K_ED_N)];
K1=sigmf(K_ED_NN,[1 n1]);
```

```
K_GK_N=min(K_GK)./K_GK;
K_GK_NN=[K_GK_N-min(K_GK_N)]/[max(K_GK_N)-min(K_GK_N)];
K2=sigmf(K_GK_NN,[1 n2]);

KK=K1+K2;
KS=[KK-min(KK)]/[max(KK)-min(KK)];
KD=sigmf(KS,[4.5 0.5]);

c=1-KD;

% ************************************************************************
% Alt Program İsmi  : VQ_analiz.m
% ************************************************************************

% Sistemin Jakobiyen matrisinin elde edilmesi ve yük akış analizi
% Matpower programının çalıştırılması
runpf('durum');
% Matpower programından değerlerinden alınması
load degerler.mat
% Jakobiyen matrisin elde edilmesi
load jac.mat
%V-Q duyarlılık endeksinin değerinin elde edilmesi
A=full(j22)-full(j12)'*inv(full((j11)))*full(j21)';
kriter=inv(det(A))

% ************************************************************************
```

ÖZGEÇMİŞ

Doğum tarihi	14.07.1976	
Doğum yeri	İstanbul	
Lise	1990-1993	Eyüp Lisesi
Lisans	1993-1998	İstanbul Teknik Üniversitesi Elek.-Elektronik Fak. Elektrik Mühendisliği Bölümü
Yüksek Lisans	1998-2000	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik Müh. Anabilim Dalı
Doktora	2000-	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik Müh. Anabilim Dalı

Çalıştığı kurum(lar)

1999-Devam ediyor	YTÜ Elektrik-Elektronik Fakültesi Elektrik Mühendisliği Bölümü Elektrik Tesisleri Anabilim Dalında Araş.Gör.