KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMALARLA KISA DÖNEM OPTİMUM ÜNİTE TAHSİSİ

YÜKSEK LİSANS TEZİ

Uğur ALKANOĞLU

Anabilim Dalı: Elektronik Bilgisayar Eğitimi

Danışman: Yrd. Doç. Dr. Mehmet YILDIRIM

KOCAELİ, 2007

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMALARLA KISA DÖNEM OPTİMUM ÜNİTE TAHSİSİ

YÜKSEK LİSANS TEZİ

Uğur ALKANOĞLU

Tezin Enstitüye Verildiği Tarih: 04 Temmuz 2007

Tezin Savunulduğu Tarih: 28 Temmuz 2007

Tez Danışmanı	Üye	Üye
Yrd. Doç. Dr. Mehmet YILDIRIM	Yrd. Doç. Dr. Melih İNAL	Yrd. Doç. Dr. Ferdi BOYNAK
(.....)	(.....)	(.....)

KOCAELİ, 2007

ÖNSÖZ ve TEŞEKKÜR

Modern toplumların en önemli ihtiyacı olan elektrik enerjisine olan talep nüfus artışı, sanayileşme ve şehirleşme ile hızla artmaktadır. Artan enerji ihtiyacı ile oluşan talebin karşılanması için trilyon dolarlara varan yatırımlara gereksinim olduğundan eldeki imkanlardan en uygun şekilde yararlanılması gereklidir. Bu tez çalışmasında fosil yakıtlarla çalışan mevcut sistemlerin çalışma takviminde genetik algoritmalarla optimizasyon yapılarak maliyetin en aza çekilmesi amaçlanmıştır.

Yaşamımın her alanında sürekli desteklerinden dolayı özlediğim sevgili aileme ve bu çalışmanın hazırlanması sırasında benden yardımlarını, desteğini ve zamanını esirgemeyen danışman hocam Yard. Doç. Dr. Mehmet YILDIRIMA' a sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER DİZİNİ

a ,b, c	: Ünitenin yakıt maliyeti katsayıları
σ , δ, τ	: Ünitenin soğuk başlatma katsayıları
ρ	: Yatık konum maliyet sabiti
i	: i=1,2,…N ünite numarası
j	: j=1,2,...T saati
P_D	: Talep edilen toplam güç
$P_{D,j}$	: j. saatte talep edilen güç
$P_{DownRamp}$	: Ünitenin artan güç eğim oranı
P_G	: Talep edilen toplam güç
$P_{G,j}^{max}$	: j. saatte ünitelerin verebileceği en yüksek güç
$P_{i,j}$	: i. üniteden talep edilen güç
$P_{min\ i,j}$	: i. üniteden alınması gereken en düşük güç miktarı
$P_{max\ i,j}$	: Ünitenin verebileceği en yüksek güç
P_{Rj}	: j. saat verilmesi gereken yedek güç
P_{UpRamp}	: Ünitenin artan güç eğim oranı
T^{OFF}	: Ünitenin kapalı kaldığı süre
T^{ON}	: Ünitenin açık kaldığı süre

Kısaltmalar

CSUC	: Ünitenin soğuk başlatma maliyeti
FC	: Ünitenin yakıt maliyeti
GA	: Genetik algoritma
HSUC	: Ünitenin sıcak başlatma maliyeti
MDT	: Ünitenin en az devrede dışı kalma süresi
MUT	: Ünitenin en az devrede kalma süresi
MW	: Mega watt
P	: Üniteden talep edilen güç
SDC	: Ünitenin kapatma maliyeti
TM	: Toplam maliyet

GENETİK ALGORİTMALARLA KISA DÖNEM OPTİMUM ÜNİTE TAHSİSİ

Uğur ALKANOĞLU

Anahtar Kelimeler: Genetik algoritma, Ünite tahsisi, Ekonomik yük dağılımı, Elektrik enerjisi üretimi

Özet: Bütün enerji sistemlerinde yapılması gereken en önemli iş ünite tahsisidir. Talep edilen enerjiye bakılmaksızın bütün üniteleri çalışır durumda bırakmak oldukça pahalıya mal olmaktadır. Talebe göre, çalışmasına gerek duyulmayan ünitelerin kapalı durumda bırakılması önemli miktarlarda maliyet tasarrufu sağlamaktadır. Talep ve yedek enerjiyi karşılamak üzere, ünitelerin hangi saatlerde devrede ya da devre dışında kalacakları planlanırken, ünitelerin ekonomiklik koşulları ile birlikte ünitelerin teknik çalışma şartlarını belirleyen kısıtlamalar da göz önünde bulundurulur.

Toplam üretim maliyetinde azalma sağlamak için yapılan ve enerji sisteminde yer alan bütün ünitelerin planlanan bir çalışma takvimine göre sisteme alınması veya sistemden çıkartılması ünite tahsis problemidir. Ünite tahsis problemi kısıtlamalı bir optimizasyon problemidir. Bu problemde, problemin giriş değerleri; talep edilen enerji, rezerv olarak bulundurulması gereken yedek enerji, ünitelerin karakteristik özellikleri olan ortalama yakıt maliyetleri, minimum devrede kalma ve minimum devre dışı kalma süreleri, başlatma maliyetleri, kapatma maliyetleridir. Problemin çıkışında ise, hangi ünitelerin hangi saat diliminde çalışacağını ve çalıştırılacak olanlardan ne kadar enerji alınacağını gösteren bir plan bulunur. Çok sayıda kısıtın olması, karmaşıklığı ve analitik çözüm yolunun olmaması gibi nedenlerle; çözümü zor, doğrusal olmayan, geniş ölçekli ve kombinasyonel bir problemdir. Bu nedenle, çözüm uzayında rastlantısal arama yapan yöntemlerden olan genetik algoritmalar bu problemin çözümünde kullanılmış ve yapılan uygulama programı ile bulunan sonuçlar gösterilmiştir.

SHORT TERM OPTIMUM UNİT COMMİTMENT BY GENETİC ALGORİTHMS

Uğur ALKANOĞLU

Keywords: Genetic algorithms, Unit commitment, Economic load dispatch, Production of Electrical Energy

Summary: The most important job is solving the unit commitment problem in the energy system. Working of all units without considering the power demand causes too much expenses. According to the power demand, shutting-down of unnecessary units saves a great deal of production cost. While planning which units are on or off in order to meet the consumers variable demands in a day, technical operating constraints of units should be obeyed as well as the economical constraints.

The unit commitment problem in a power system involves determining start-up and shut-down schedules of units to be used to meet forecasted demand over a future short term. Unit commitment is a constrained optimization problem. The inputs of the problem are power demand, spinning reserve, the characteristics of the units which are fuil cost, minimum up-time constraint, minimum down-time constraint, start-up cost, and shut-down cost. The outputs of the problem is a plan that shows hourly on or off states of units in a day and the power outputs of the units. Since it has some constraints, course of dimensionality and no analytical solution, the unit commitment problem is a non-linear, large-scale, combinetorial problem and so it is hard to solve. For this reason, genetic algorithm which is the one of randomized search methods is used to solve the unit commitment problem.

1. GİRİŞ

Elektrik enerjisi, tüketildiğinde herhangi bir atık üretmediğinden dolayı temiz bir enerjidir ve kullanımı bakımından evrensel bir özellik taşır. Diğer enerji biçimlerine de kolaylıkla dönüştürülebilir. Kullanım kolaylığından dolayı, günümüz modern toplumlarının vazgeçilmez bir ihtiyacıdır. Örneğin, evlerindeki insanlar sıcak su, ısınma, aydınlanma ve elektronik cihazları çalıştırma sebebiyle; endüstriyel kullanıcılar ise ısınma, üretim, işleme ve çekiş gücü ihtiyacını karşılama sebebiyle elektriğe neredeyse bağımlı konumdadırlar.

Elektrik enerjisinin üretimi için kullanılan enerji kaynakları temel olarak; nükleer enerji, yenilenebilir enerji kaynakları (güneş, hidrolik, rüzgar, biyokütle, jeotermal, vb.) ve fosil yakıtlardır. Nükleer enerji kurulum maliyetleri ve görülen kimi sakıncaları nedeniyle birçok ülkede sınırlandırılmıştır. Yenilenebilir enerji kaynakları arasında elektrik üretimi için en yaygın olarak kullanılan hidrolik enerjisidir ve gelişmiş ülkelerde potansiyelinin tamamına yakını devreye alınmış durumdadır. Petrol, kömür ve doğal gaz tükenir özellikteki fosil hidrokarbon enerji kaynaklarıdır. Diğer yenilenebilir enerji kaynaklarının işletme maliyetleri ucuz olmasına rağmen, kurulum maliyetleri fosil yakıtlara oranla daha pahalıdır.

Her ne kadar fosil kaynaklar, enerji dönüşüm sürecinde yaydıkları karbon emisyonları çevre kirliliği açısından değişen oranlarda zararlı ise de, enerji üretiminin yanı sıra sanayi hammaddesi olarak da yaşamsal önem arz etmesi ve alternatif kaynakların bu kaynakları ikame etme olanaklarının yakın gelecekte mümkün görünmemesi gibi nedenlerle, önümüzdeki on yıllarda da dünya enerji tüketiminde belirleyici enerji hammaddesi olacaktır [1].

Dünyada nüfus artışı, sanayileşme ve şehirleşme ile birlikte, küreselleşme sonucu artan ticaret ve üretim imkanlarına bağlı olarak, doğal kaynaklara ve enerjiye olan talep giderek artmaktadır. 2030 yılına kadar dünya enerji talebinin bugüne oranla

%60 oranında artması beklenmektedir. Fosil kaynaklar, bugün olduğu gibi yakın gelecekte de dünya enerji talebinde önemini sürdürmeye devam edecektir. 2002 yılında dünya toplam enerji talebinde fosil kaynakların payı %80 iken, bu oranın 2030 yılında %82 olması beklenmektedir [2].

Çevre etkileri ve ekonomik koşullar göz önüne alındığında, yeni üretim tesislerinin kurulmasından önce mevcut sistemlerden en uygun şekilde yararlanılmasının gerektiği açıkça görülmektedir. Enerji sistemlerinin verimliliği, enerji kaynaklarının hızla tükenmesi ve artan enerji ihtiyacı sebebiyle giderek daha da önem kazanmıştır. Elektrik enerjisinin bazı özellikleri onu farklı ve zor bir ürün haline getirmektedir. En önemlisi, depolanması oldukça masraflıdır. Elektriğin depolanması için gerekli olan hidroelektrik pompa ve pil gibi teknolojiler hiç etkin değildir. Bu sebeple, elektrik talebi ve arzının her saniye dengelenmesi gerekmektedir [3].

Evrensel düzeyde kullanılan bir enerji olması, hem nihai hem de ara mal olması, depolama imkanının, çok yüksek maliyetli olmasından dolayı, neredeyse imkansız olması, elektriği diğer ürünlerden farklı kılmaktadır. Bu nedenle özellikle üretim aşamasından itibaren çok planlı olarak tüketicilere aktarılması gereklidir. Burada planlama, ihtiyaca göre üretimin ve tüketimin düzenlenmesidir.

Elektrik enerjisinin üretildiği ünitelerin her birinin çalışma yapıları ve kullandığı yakıtlar, dolayısıyla da her birinin ürettiği elektrik enerjisinin maliyeti farklı olabilmektedir. Gün içinde üretim birimlerinden talep edilen enerji sabit olmadığı için, üretim birimlerinin saatlik periyotlar için devreye alınmaları veya devreden çıkarılmalarının düzenlenmesi gerekmektedir. Üretim birimlerinden talep edilen enerjinin toplam maliyetini en düşük seviyede tutmak için, belli bir zaman aralığında ünitelerin çalıştırılıp çalıştırılmayacağını belirleyen planlama çalışmasına ünite tahsis problemi (unit commitment problem) denir.

Ünite tahsis problemi kısıtlamalı bir optimizasyon problemidir. Bu problemde, problemin giriş değerleri; talep edilen enerji, rezerv olarak bulundurulması gereken yedek enerji, ünitelerin karakteristik özellikleri olan ortalama yakıt maliyetleri,

minimum devrede kalma ve minimum devre dışı kalma süreleri, başlatma maliyetleri, kapatma maliyetleridir. Problemin çıkışında ise, hangi ünitelerin hangi saat diliminde çalışacağını ve çalıştırılacak olanlardan ne kadar enerji alınacağını gösteren bir plan bulunur. Çok sayıda kısıtın olması, karmaşıklığı ve analitik çözüm yolunun olmaması gibi nedenlerle; çözümü zor, doğrusal olmayan, geniş ölçekli ve kombinasyonel bir problemdir.

Problemin çözümünde uygulanan optimizasyon tekniklerinden en bilinenleri arasında; dinamik programlama [4,5], tamsayı programlama [6,7] , öncelik listesi [8], lagrange-relaxation (ing.) [9-11] gibi matematiksel yöntemler yanında; tabu araması [12], benzetilmiş tavlama [13] ve genetik algoritmalar [14-29] gibi sezgisel yöntemler de sayılabilir.

Genetik algoritmaların, özellikle lineer olmayan, çok değişkenli, çok amaçlı optimizasyon problemlerinin çözümünde güçlü bir arama yöntemi olduğu literatürde bir çok çalışmada gösterilmiştir [14-29]. Bu optimizasyon yöntemi yerel minimuma takılmayan ve büyük boyutlu problemlerde diğerlerine oranla daha iyi sonuç veren bir tekniktir.

Ünite tahsis probleminin çözüm uzayının çok büyük olması ve analitik çözümünün olmaması nedeniyle çözüm uzayında rastlantısal arama yöntemleri daha başarılı olur. Genetik algoritmalar da rastlantısal arama yöntemlerinden en çok bilinen ve kullanılanıdır. Genetik algoritmalar arama işlemine bir nokta grubundan başlayarak yerel optimuma takılmadan çözüm uzayında rastlantısal olarak arama yapabilmeleri sayesinde bu tip kısıtlamalı problemlerin çözümünde başarısını kanıtlamıştır.

Genetik algoritma, doğadaki evrim mekanizmasını örnek alan bir arama metodudur ve bir veri grubundan problemin çözümü olan özel bir veriyi bulmak için kullanılır. Genetik algoritmalar 1970'lerin başında John Holland tarafından ortaya atılmıştır. Genetik Algoritmalar, Evrimsel Genetik ve Darwin'in Doğal seleksiyonuna benzerlik kurularak geliştirilmiş "iteratif " bir arama metodudur.

Genetik algoritmalar doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Yani doğadaki güçlü olan hayatta kalır mantığı genetik algoritmada da geçerlidir. Bu yöntemi sayılar üzerine uygularken dikkat edilecek nokta ise hangi çözümün daha iyi (güçlü) olduğunu belirleyebilmektir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için yeniden kopyalama (recombination) , değiştirme (mutation) gibi operatörleri kullanır. Genetik algoritmaların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir.

Genetik algoritmadaki en önemli özellik olayların tamamen rastlantısal olmasıdır. Örneğin bir çözümün diğer çözüme tercih edilmesi, çözümün istenen sonuca ne kadar yaklaştığıyla ilgili olsa da tamamen olasılıkla ilgilidir ve iyi çözümün seçilmesi hiçbir zaman garanti altında değildir. Bu durum genetik algoritmaların başarısının ardındaki sırdır. Yani doğada da olduğu gibi aslında her zaman güçlüler hayatta kalmaz. Bazen güçlülerin ölüp, güçsüzlerin hayatta kalması gereklidir.

Ünite tahsis probleminin genetik algoritmalarla çözümüyle ilgili çok sayıda yayın yapılmıştır, bu yayınların birbirinden en önemli farkı kromozomların kodlanmasında olmuştur. Ünite tahsis problemi kısıtlamaları nedeniyle karmaşık bir problem olduğundan önerilen kodlamalar genellikle kısıtları da kapsayacak şekilde olmakta, bu sayede mutasyon ve çaprazlama operatörlerinden sonra uygun olmayan çözümlerin ortaya çıkma ihtimali azaltılmakta veya bu çözümlerin kontrol edilerek düzeltilmesi kolaylaştırılmaktadır. Yapılan değişiklikler çözümün iyiliğine çok katkı sağlamasa da gelişen bilgisayar teknolojileri sayesiyle de işlem zamanlarında önemli azalmalar olmuştur.

A. H. Mantawy ve diğ. (1999) genetik algoritmayla ünite tahsis problemini çözerken farklı bir algoritma yapısı kullanmışlardır. Yaptıkları yayınlarında kromozom yapısında ikili ve onluk sayı sistemini birlikte kullanıp; çaprazlama işlemini ikili yapıda kromozomun gösterimini ise onlu yapıda oluşturmuşlardır. Bir ünite için belli bir periyot boyunca hangi saatte açık ya da kapalı kaldığını gösteren kromozom dizisini ikili olarak ifade edip gerçekte kromozomların onluk hallerini bilgisayar

hafızasında tutmuşlardır. Eğer bir kromozom genetik operatörlerde işleme tutulmak üzere seçilirse ikili sayı sistemine çevrilerek gerekli işlemler yapılır. Bu sayede işlem zamanından önemli derecede tasarruf edildiğini göstermişlerdir. Problemdeki kısıtların üstesinden gelmek için ise uygunluk fonksiyonunda kısıtları görmezden gelip daha sonra bulunan çözümler içinden uygun olmayanların elemesini yapmışlardır.

Yiying ve diğ. (2002) tarafından yapılan çalışmada en düşük devrede kalma ve en düşük devre dışı kalma süreleri de kromozomların yapılarına dahil edilmiştir. Kodlama yapılırken ünitelerin açık kapalı oldukları bir bit ile gösterildikten sonra diğer üç bitte ünitelerin ne kadar saat daha açık kalacağı gösterilmiştir. Ayrıca yaptıkları çalışmalarında genetik algoritmanın temel işlevleri olan seçim işlevinde, çaprazlama ve mutasyon oranlarında kullanılan değişik değerlerin, sonucu ve işlem zamanını ne oranda etkilediğini göstermişlerdir.

Swarup ve Yamashiro (2003) ise yaptıkları çalışmada ünitelerin açık kapalı durumları bir bit olarak verilmekte fakat kromozomu vektör olarak değil iki boyutlu matris olarak vermişlerdir. Burada bir ve sıfırlardan oluşturulan matriste, satırlar çalışma saatlerini, sütunlar üniteleri, matristeki sıfır ve birler ise belirli bir saatte bir ünitenin devrede ya da devre dışı olma durumunu göstermektedir.

Yang ve diğ. (1997) 8 işlemcili bilgisayar ile iki türlü paralel genetik algoritma gerçekleştirmişlerdir. Bunlardan birincisi ana-uydu genetik algoritmadır. Ana-uydu yönteminde; ana bilgisayar bütün popülasyonu rasgele üretip, uygunluk değeri yüksek olan bireylerin daha yüksek ihtimalle seçerek uydu işlemcilere alt popülasyon olarak gönderir, alt popülasyonu alan uydu işlemciler çaprazlama, mutasyon ve uygunluk değeri hesaplamalarını yaparak popülasyonu ana işlemciye geri gönderir ve durma koşulu sağlanıncaya kadar işlemleri tekrarlar. İkinci yöntem ise çift yönlü halka paralelleştirilmesidir. Çift yönlü halka metodunda ise her işlemci alt popülasyonunu oluşturarak kendi içinde çaprazlama, mutasyon işlemlerini yerine getirdikten sonra belirlediği en iyi bireyleri sağındaki ve solundaki işlemcilere gönderir, bu işlemciler ise aldıkları en iyi bireyleri kendilerindeki en kötü bireylerle değiştirir. Durdurma şartı sağlanana kadar işlemler devam ettirilir.

Arroyo ve Conejo (2002) yaptıkları çalışmalarında, Yang ve arkadaşlarının uyguladıkları paralelleştirme yöntemlerini birleştirerek paralel genetik algoritmayla buldukları çözümü, Lagrange rahatlamasından buldukları çözümlerle karşılaştırarak genetik algoritmanın üstünlüklerini göstermişlerdir.

Damousis ve diğ. (2004) ünite tahsis probleminin genetik algoritmalarla çözümünün uzun zaman aldığını ve bununda nedeninin kromozom kodlamasında kullanılan ikili sayı sistemi olduğunu savunmuşlardır. Yaptıkları çalışmada kromozomların gösteriminde gerçel sayıları kullanarak işlem zamanında gelişme kaydetmişlerdir.

Orero ve Irving (1997) ünite paylaşımı probleminde genetik algoritma ile Lagrange-relaxation bir araya getirerek hesaplama zamanını azaltıp daha iyi çözüm değerlerine ulaşmışlardır.

Qiang ve Lo (1997) genetik algoritmanın işlem basamakları içerisine dinamik programlamayı da katarak maliyette kayda değer ilerleme sağlamışlardır.

Kazarlis ve diğ. (1996) problemin çözümünde kullandıkları standart genetik algoritmaya değişik eklentiler yaparak işlem zamanında ve çözüm değerlerinde daha iyi sonuçlar bulmuşlardır.

Bu tez çalışmasında, MATLAB yazılımı kullanılarak geliştirilen bir benzetim programı ile, genetik algoritmalar kısa dönem elektrik enerjisi optimum ünite planlamasına uygulanmıştır. Planlamanın içeriğini oluşturan, ünite tahsis problemi çözüm uygulamasında, genetik algoritmanın parametreleri değiştirilerek bulunan sonuçlar karşılaştırılacaktır.

Çalışmanın ikinci bölümünde, ünite tahsis probleminin nasıl oluştuğu, tanımı, anlatılmış; ünite katkı maliyetleri ve ünite planlamasındaki kısıtlar ile ünite paylaşımında yaygın olarak kullanılan diğer metotlar olan dinamik programlama, doğrusal programlama, öncelik listesi, langrange-relaxation metotları açıklanmıştır.

Üçüncü bölümde, optimizasyon tekniği olarak, kısa dönem ünite planlamasında kullanılacak olan genetik algoritmalar yer almaktadır. Bu tekniğin diğer optimizasyon yöntemlerinden farkı ve üstünlükleri açıklanmış, genetik algoritmaların bölümleri ile problem çözümünde kullanılan genetik operatör yöntemlerine yer verilmiştir.

Dördüncü bölümde genetik algoritmalar ile yapılan optimizasyon çalışması yer almaktadır. Optimizasyon problemi için genetik algoritma modeli oluşturulmuş, kuşak sayısı, mutasyon oranı ve nüfus büyüklüğünün etkisi incelenmiştir.

Son olarak beşinci bölümde tez çalışmasının sonuçları değerlendirilmiş ve öneriler sunulmuştur.

2. ELEKTRİK ENERJİSİ KISA DÖNEM ÜNİTE PLANLAMASI

2.1. Giriş

Elektrik enerjisi, tüketildiğinde herhangi bir atık üretmediğinden dolayı temiz bir enerjidir ve kullanımı bakımından evrensel bir özellik taşır. Diğer enerji biçimlerine de kolaylıkla dönüştürülebilir. Kullanım kolaylığından dolayı, günümüz modern toplumlarının vazgeçilmez bir ihtiyacıdır.

Petrol, kömür ve doğal gaz, tükenir özellikteki fosil hidrokarbon kaynaklardır. Bugün, ispatlanmış üretilebilir petrol rezervleri 1189 milyar (1,2 trilyon) varildir. Mevcut üretim değerleri (günde 80 milyon varil) dikkate alındığında, petrol rezervlerinin ömrü yaklaşık 41 yıldır. Dünya üretilebilir doğal gaz rezervleri 180 trilyon metre küptür. Mevcut üretim değeri olan 2,7 trilyon metre küp dikkate alındığında, bu rezervin ömrü 67 yıldır. Kömür ise, diğer iki fosil kaynağa göre, dünya üzerinde çok daha homojen dağılmış bir kaynaktır. Diğer yandan, dünya kömür rezervleri 909 trilyon tondur ve mevcut üretim eğilimleri dikkate alınırsa, 2004 yılı verileriyle 164 yıllık ömrü vardır. Her ne kadar fosil kaynaklar, enerji dönüşüm sürecinde yaydıkları karbon emisyonları çevre kirliliği açısından değişen oranlarda zararlı ise de, enerji üretiminin yanı sıra sanayi hammaddesi olarak da yaşamsal önem arz etmesi ve alternatif kaynakların bu kaynakları ikame etme olanaklarının yakın gelecekte mümkün görünmemesi gibi nedenlerle, önümüzdeki on yıllarda da dünya enerji tüketiminde belirleyici enerji hammaddesi olacaktır [30].

Fosil kaynaklar ile diğer enerji kaynaklarının durumu ve toplumların gelecek yıllardaki enerji ihtiyacı ortadadır. Artan enerji ihtiyacı ile oluşan talebin karşılanması için trilyon dolarlara varan yatırımlara gereksinim vardır. Çevre etkileri ve ekonomik koşullar göz önüne alındığında, yeni üretim tesislerinin kurulmasından önce mevcut sistemlerden en uygun şekilde yararlanılmasının gerektiği açıkça görülmektedir. Enerji sistemlerinin verimliliği, enerji kaynaklarının hızla tükenmesi

ve artan enerji ihtiyacı sebebiyle giderek daha da önem kazanmıştır. Yapılan tahminlere göre, 10000 MW büyüklüğünde üretim gücüne sahip olan tesis için işletme maliyetinde yapılacak her %1'lik düşüş, yıllık ortalama 20 milyon Amerikan Doları tasarruf sağlamaktadır [31].

Elektrik enerjisinin bazı özellikleri onu farklı ve zor bir ürün haline getirmektedir. En önemlisi, depolanması oldukça masraflıdır. Elektriğin depolanması için gerekli olan hidroelektrik pompa ve pil gibi teknolojiler hiç etkin değildir. Bu sebeple, elektrik talebi ve arzının her saniye dengelenmesi gerekmektedir. Elektrik enerjisinin az ya da fazla olması, sadece bir kaç müşteriyi etkilemekle kalmayacak, aynı zamanda tüm elektrik şebekesinin düzenini tehlikeye atacaktır. Bu yüzden, şebeke operatörü zorunlu olarak tüketim kısıntısına gitmez ise, elektrik talep eden tüketicilerin arz/talep dengesinin korunabilmesi imkansız hale gelecektir [32].

Evrensel düzeyde kullanılan bir enerji olması, hem nihai hem de ara mal olması, depolama imkanının, çok yüksek maliyetli olmasından dolayı, neredeyse imkansız olması, elektriği diğer ürünlerden farklı kılmaktadır. Bu nedenle özellikle üretim aşamasından itibaren çok planlı olarak tüketicilere aktarılması gereklidir. Burada planlama, ihtiyaca göre üretimin ve tüketimin düzenlenmesidir.

Elektrik enerjisinin üretildiği ünitelerin her birinin çalışma yapıları ve kullandığı yakıtlar, dolayısıyla da her birinin ürettiği elektrik enerjisinin maliyeti farklı olabilmektedir. Gün içinde üretim birimlerinden talep edilen enerji sabit olmadığı için, üretim birimlerinin saatlik periyotlar için devreye alınmaları veya devreden çıkarılmalarının düzenlenmesi gerekmektedir. Üretim birimlerinden talep edilen enerjinin toplam maliyetini en düşük seviyede tutmak için, belli bir zaman aralığında ünitelerin çalıştırılıp çalıştırılmayacağını belirleyen planlama çalışmasına ünite tahsis problemi (unit commitment problem) denir.

Talep edilen enerjinin karşılanması için yapılan ve ünitelerin devreye alınıp alınmayacağına karar veren planlama oluşturulurken, devrede olan üniteler arasında, maliyetin en düşük seviyede tutulması için, yükün ekonomik olarak ünitelere paylaştırılması gereklidir. Bu nedenle ünite tahsis problemi, ekonomik yük dağılımı

sorununu da beraberinde getirir. Oluşturulan planlamada, devreye alınmasına karar verilen ünitelerin talep edilen yükü karşılamalarına rağmen, açma kapama maliyetleri gibi nedenlerden dolayı gerçekte ekonomik olmayabilir. Burada amaç, hem talep edilen yükün karşılanması hem de en ekonomik olması gibi iki kısıtlamayı da göz önünde bulundurarak en uygun çözüme ulaşmaktır.

Ünite tahsis problemi kısıtlamalı bir optimizasyon problemidir. Bu problemde, problemin giriş değerleri; talep edilen enerji, rezerv olarak bulundurulması gereken yedek enerji, ünitelerin karakteristik özellikleri olan ortalama yakıt maliyetleri, minimum devrede kalma ve minimum devre dışı kalma süreleri, başlatma maliyetleri ve kapatma maliyetleridir. Problemin çıkışında ise, hangi ünitelerin hangi saat diliminde çalışacağını ve çalıştırılacak olanlardan ne kadar enerji alınacağını gösteren bir plan bulunur. Çok sayıda kısıtın olması, karmaşıklığı ve analitik çözüm yolunun olmaması gibi nedenlerle; çözümü zor, doğrusal olmayan, geniş ölçekli ve kombinasyonel bir problemdir [33].

Bütün enerji sistemlerinde yapılması gereken en önemli iş ünite tahsisidir. Talep edilen enerjiye bakılmaksızın bütün üniteleri çalışır durumda bırakmak oldukça pahalıya mal olmaktadır. Talebe göre, çalışmasına gerek duyulmayan ünitelerin kapalı durumda bırakılması önemli miktarlarda maliyet tasarrufu sağlamaktadır.

Modern bir enerji üretim tesisinin karşılaması gereken çeşitli hizmetler vardır. Bunlardan en önemlileri kesintisiz, güvenli ve kaliteli enerji akışıdır. Burada kaliteden kastedilen, elektriğin sabit voltajda ve frekansta kalmasıdır. Kullanıcıların ihtiyaç duydukları alanlarda sıkıntıya düşmemeleri için elektrik enerjisinin kesintiye uğramaması gereklidir. Ayrıca sistemde oluşacak arızalar, önlemler alınarak kullanıcılara aktarılmamalıdır.

2.2. Enerji Sistemlerinin Planlanması

Öngörülen talebe göre üretimin planlanması zorunludur. Bu planlama çeşitli periyotlar halinde yapılır ve her aşamadaki amaç, sistemdeki mevcut ünitelerle üretim maliyetinin en aza indirilmesidir.

Bir enerji sistemindeki planlama aşağıdaki kısımlardan oluşur:
Çalışma planlaması
Çalışma kontrolü
Çalışma kaydı

Çalışma planlaması da kendi içerisinde uzun dönem, orta dönem ve kısa dönem olmak üzere üçe ayrılır.

Uzun dönem çalışma planlamasında aylık yük tahmini ve bakım takvimi oluşturulur. Bakım takvimi, ünitelerin karakteristik ve yapılış özelliklerine göre, aylık talebin kabataslak değerlerini karşılamak koşuluyla, bakıma alınacakları zamanları gösterir.

Orta dönem çalışma planlamasında, hidrotermal koordinasyonu ve kabaca haftalık yük tahmini yapılır. Yük tahmini haftalık enerji ihtiyacını belirler ve hidrotermal koordinasyon planı, öngörülen talebe göre en uygun olacak şekilde, hangi hidrolik ve termal ünitelerin devreye alınacağını belirler.

Kısa dönem çalışma planlaması ise kısa dönem yük tahmini, güvenlik analizi ve ekonomik yük dağılımı programını kapsar. Kısa dönem yük tahmini, saatlik periyotlar halinde günlük yük tahminini ve talep edilen enerjinin en uygun olacak şekilde üniteler arasında paylaştırılmasını kapsar.

Uzun dönem planlamanın sonucunda, yıllık olarak ve aylık bazda oluşacak yük talebinin karşılanması bulunur. Bulunan bu sonuç, orta dönem planlamanın amacını oluşturur. Orta dönem planlamanın sonucunda ise haftalık yük talebinin sağlanması bulunur ki bu değerler, günlük ve saatlik olarak elektrik enerjisini tüketicilere sağlamak için yapılan kısa dönem planlamanın giriş değerleridir.

Çalışma kontrolü, talep edilen yükü karşılamak için anlık olarak yapılan işlemleri kapsar. Burada yapılması gereken görevler üretim kontrolü, ünitelerin çalışma durumlarının değerlendirilmesi, anlık yük akışı, çalışma planı, ekonomik yük dağılımı, ayrılacak rezerv hesapları ve yük yönetim sistemidir.

Çalışma kaydında ise enerji sisteminde yapılan her işlem daha sonraki planlamalarda kullanmak üzere kayıt altına alınır [34].

2.3. Ünite Tahsis Problemi

Tüketicilerin talep ettikleri enerji gün içerisinde sürekli değişir. Örneğin, gündüzleri ve akşamları endüstriyel taleplerden ve lambaların yoğun kullanımından dolayı talep fazladır. Gece geç ve sabah erken saatlerde ise bütün toplum uykuda olduğundan talep daha azdır. Bu değişken talebi karşılamanın en kolay yolu, enerji üretim merkezlerindeki bütün üniteleri açık bırakmaktır. Fakat bu yöntem oldukça pahalıya mal olmaktadır ve bu nedenle sadece ihtiyaç olduğu kadar enerji üretilmelidir. Kısa dönem ünite paylaşımında, tüketicilerin enerji taleplerini karşılamak için elektrik üretiminin saat bazında, bir veya iki gün için planlanması gerekir. Bu planın yapılmasındaki amaç sistemin en uygun maliyet ile enerji üretmesidir. Enerji santralinin ekonomik şekilde işletilmesinin yolu, günlük değişken yük talebine göre hangi santralin devreye alınıp çıkarılacağı, devreye alınan enerji santralinin hangi kapasite ile çalıştırılacağına karar vermekten geçer. Bu kararların yanında enerji üretim birimlerinin kısıtlamaları, enerji sisteminin fiziksel kısıtlamaları gibi birçok kısıtlamadan oluşan bir problem ortaya çıkar, bu probleme ünite tahsis problemi adı verilir ve bu bir optimizasyon problemidir.

Gün içinde değişen enerji talebinin en ekonomik şekilde sağlanması için belli bir plan çerçevesinde herhangi bir anda, hangi üniteden ne kadar enerji alınacağının önceden belirlenmesi gerekir. Bu planlama ve belirleme yapılırken ünitelerin değişim kısıtlamaları ve enerji sisteminin fiziksel kısıtlamaları gibi bir problem ortaya çıkar. Ünite tahsis problemi çözülürken bu kısıtların da göz önünde bulundurulması gerekir.

Talep edilen enerjinin karşılanması için yapılan, ünitelerin devrede olup olmayacağına karar veren plan oluşturulurken, devrede olan üniteler arasında, maliyetin en düşük seviyede tutulması için yükün ekonomik olarak paylaştırılması gereklidir. Bu nedenle ünite tahsis problemi, ekonomik yük dağılımı sorununu da beraberinde getirir. Oluşturulan planda, devrede kalmasına karar verilen ünitelerin talep edilen yükü karşılaması açma, kapama maliyetleri gibi nedenlerden gerçekte

ekonomik olmayabilir. Burada amaç iki kısıtlamayı da göz önünde bulundurarak en uygun çözümü bulmaktır.

Ünite tahsis problemi enerji üretim maliyetlerinden ve sistem kısıtlarından oluşmaktadır. Üretim maliyetleri üniteler devrede iken oluşan yakıt maliyetleri, üniteler devreye girdiklerinde oluşan başlatma maliyetleri ve ünitelerin kapatılmasıyla oluşan durdurma maliyetleridir.

Problemdeki kısıtlar ise ünitelerin türbinlerinin ısıtılması ihtiyacı, dönmeye başlamasının ve durdurulmasının belli bir zaman alması, talep edilen gücün ve yedek gücün karşılanmasıdır [31].

2.3.1. Ünite maliyetleri

Ünite tahsis probleminde ünitelerin oluşturduğu maliyetler iki ana başlık altında toplanır. İlk gurupta oluşan maliyetler, ünitelerin durum değiştirme maliyetleridir. Burada oluşacak masraflar ünitelerin üretim için hazır hale gelene kadar geçen sürede oluşan maliyetler ve kapatma maliyetleridir. İkinci olarak, üretim maliyetleri ise talep edilen yükü karşılamak için harcanan yakıt maliyetlerinden oluşur.

2.3.1.1. Yakıt maliyeti

Yakıt maliyeti üniteler çalışır durumda iken ve ürettikleri güce bağlı olan bir fonksiyondur. Termik üretim birimlerinde genel olarak buhar türbinleri ve yanmalı türbinler olmak üzere iki tip türbin yer almaktadır. Buhar türbinleri fosil yakıtların yanmasıyla elde edilen buhar ile döndürülür. Yanmalı türbinlerde ise doğal gaz veya distile edilmiş petrolün yanması ile hareket elde edilir. Termik üretim birimlerinin yakıt maliyeti denklem 2.1'deki gibi ifade edilebilir.

$$FC = ai + bi\mathrm{P} + \mathrm{c}i\mathrm{P}^2 \qquad (2.1)$$

Burada; FC ünitenin yakıt maliyeti, ai, bi, ve ci yakıt maliyeti katsayıları, P ise üniteden talep edilen güç miktarıdır.

2.3.1.2. Başlatma maliyeti

Termik santrallerde elektrik enerjisi, buhar türbinlerinin sağladığı mekanik enerjiyle elde edilir. Yakıt kazanlarda yakılarak ısı enerjisine çevrilir. Bunun sonucunda, yüksek sıcaklık ve basınçta oluşan buhar, türbinlerde mekanik enerjiye çevrilir ve ünitelerden elektrik üretimi başlar. Bu nedenle ünitelerin enerji üretimine başlamaları için ulaşmaları gereken sıcaklık ve basınç değerleri farklı zamanlar almaktadır.

Başlangıç maliyeti, üniteler boşta beklerken devreye alındığında oluşan maliyettir. Termik üretim birimleri iki türlü devre dışı bırakılabilir, bunlardan birincisi ünite üretimden çıkarıldığında türbinlerin soğumaya bırakılmasıdır. Eğer ünitenin türbinleri ortam sıcaklığına kadar soğuduktan sonra tekrar ısıtılıp devreye alınırsa oluşan maliyete soğuk başlatma adı verilir. Soğuk başlatma maliyeti denklem 2.2 ile verilmiştir.

$$CSUC = \sigma_i + \delta_i \left[1 - e^{(-T_{i,j}^{OFF} / \tau_i)} \right] \tag{2.2}$$

Burada *CSUC* ünitenin soğuk başlatma maliyeti, σ_i , δ_i, τ_i başlatma katsayıları, T^{OFF} ise kapalı kalınan zamandır.

Termik birimleri devre dışı bırakmanın bir diğer yolu da türbinleri yatık konumda bırakmaktır. Yatık konumda iken ünite üretimde değildir fakat türbinlerinin soğumasına da izin verilmez. Bu sayede ünitenin devreye alınması kolaylaşır. Üretim birimleri kısa süreli olarak devre dışı kalacakları zaman yatık konuma bırakılırlar. Bu konumdaki üretim biriminin başlatma maliyeti denklem 2.3 ile verilmiştir.

$$HSUC = \rho^{TOFF} \tag{2.3}$$

Burada; *HSUC* ünitenin soğuk başlatma maliyeti, ρ yatık konum maliyet sabiti ve T^{OFF} yatık konumda kalma süresidir.

2.3.1.3. Durdurma maliyeti

Üniteler çalışır durumda iken devre dışına alındığında oluşan maliyettir ve birim çalışanları haricinde fazladan işgücüne gerek olmadığından denklem 2.4'de görüldüğü gibi genellikle sıfır kabul edilir.

$$SDC=0 \tag{2.4}$$

SDC üniteyi kapatırken oluşan maliyettir.

2.3.2. Ünite tahsisi planlamasındaki kısıtlamalar

Bu problemde çözümü zorlaştıran kısıtlardır ve iki ana başlık altında toplanabilir. Bunlar sistem kısıtları ve ünitenin kısıtlarıdır. Sistem kısıtları, talep edilen enerjinin sağlanması ve yedek yük ihtiyacıdır. Ünite kısıtları ünitenin fiziksel ve karakteristik özelliklerinden kaynaklanan kısıtlardır.

2.3.2.1. Yük dengesi kısıtlaması

Bir enerji üretim merkezinin her t anı için ürettiği güç tüketicilerin talep ettiği güce eşit veya fazla olmalıdır. Sistemden istenen güç sadece tüketicilerin istediği güç olmayıp buna sistemdeki bütün kayıplar da dahildir. Yük dengesi (2.5)'deki gibi ifade edilebilir.

$$\sum_{j=1}^{j} P_{G,J}-P_{D,J}\geq 0 \tag{2.5}$$

Burada $P_{G,j}$ ünitelerin toplam gücü, $P_{D,j}$ sistemin çektiği toplam güç ve $j=1,2,...T$ saatleridir.

2.3.2.2. Yedek enerji kısıtlaması

Bir enerji üretim sisteminde üretimin talep edilen yükü karşılamasının yanında, sistemin güvenilirliği açısından belirli bir orandaki enerjiyi gerektiğinde karşılaması istenir. Ünitelerin bir veya birkaçının devre dışı kalması durumunda kullanılmak

üzere ayrılan bu güce yedek güç (spinning reserve) denir ve denklem 2.6'da hesaplanması gösterilmiştir.

$$\sum_{j=1}^{j} P_{G,J}^{\max} \geq P_{D,J} + P_{R,J} \tag{2.6}$$

Burada $P_{G,j}^{\max}$ j. saatte ünitelerin verebileceği en yüksek gücü, $P_{R,j}$, j. saat verilmesi gereken yedek gücü, $P_{D,j}$ ise j. saatte talep edilen gücü, ve $j=1,2,...T$ ise saatleri göstermektedir.

2.3.2.3. Ünite çıkış limitleri kısıtlaması

Her ünitenin kendine has özelliklerinden ve fiziksel kısıtlamalardan dolayı verebileceği enerji miktarları belirli değerler arasında kalır. Ünitelerin üretecekleri güç denklem 2.7'de ifade edildiği gibi ilgili ünitenin sağlayabileceği en düşük ve en yüksek güç değerleri arasında olmalıdır.

$$P_{min\ i,j} \leq P_{i,j} \leq P_{max\ i,j} \tag{2.7}$$

Bu denklemde $P_{min\ i,j}$ i. üniteden alınması gereken en düşük enerji miktarı, $P_{i,j}$ i. üniteden talep edilen gücü, $P_{maxi,j}$ ise ünitenin verebileceği en yüksek gücü göstermektedir ve $j=1,2,...T$ saatleridir.

2.3.2.4. En düşük devrede ve devre dışı kalma süresi kısıtlaması

Ünitelerin yapılış özelliklerinden dolayı üretime başladıklarından sonra belirli bir süre boyunca çalışır durumda kalma zorunlulukları vardır. Aynı şekilde termal ve mekanik sınırlamalar nedeniyle ünite devre dışına alındığında belirli bir süre daha kapalı kalması gerekir. Ünitenin türbinlerini ani sıcaklık değişimlerinden korumak ve ünitelerin başlatılmasının maliyetini düşük tutmak için başlama ve durdurmadan sonra belli bir süre geçmesi istenir. Ünitelerin en az devrede kalma süresi denklem 2.8'de verilmiştir.

$$\sum_{j=1}^{T} \left| T_{i,j}^{ON} - MUT_i \right| \geq 0 \tag{2.8}$$

Burada *MUT*, ilgili ünitenin en az devrede kalma süresi, $T_{i,j}^{ON}$ ünitenin devrede kaldığı zamanlardır.

Ünitenin en az devre dışı kalma süresi denklem 2.9'da verilmiştir.

$$\sum_{j=1}^{T} \left| T_{i,j}^{OFF} - MDT_i \right| \geq 0 \tag{2.9}$$

Burada *MDT*, ilgili ünitenin en az devre dışı kalma süresi $T_{i,j}^{OFF}$ ünitenin devre dışı kaldığı zamanlardır.

2.3.2.5. Eğim oranı kısıtlaması

Üniteler termal ve mekanik sınırlamalar nedeniyle üretim seviyelerini hızla değiştiremezler. Bu nedenle ünitelerden alınan gücün saatlik olarak değişimi belirli bir değeri aşmamalıdır. Bir sonraki saatte ünite çıkışından alınacak enerji miktarı, ünitenin o anki üretimine, kaç saattir devrede olduğuna ve üretimin artma ya da azalma durumuna bağlı olan fonksiyonla bulunur. Bazı üniteler için çekilen gücün artması ve azalması farklı oranlarda olmalıdır. Buna göre en genel hali ile ünitelerin eğim oranı kısıtları aşağıdaki gibi ifade edilebilir.

$$P_j - P_{j-1} \leq P_{UpRamp} \tag{2.10}$$

$$P_{j-1} - P_j \leq P_{DownRamp} \tag{2.11}$$

Burada P_{UpRamp} ilgili ünitenin bir saat içinde, gücünde meydana gelebilecek en fazla artış miktarıdır, $P_{DownRamp}$ ise ilgili ünitenin bir saat içinde, gücünde meydana gelebilecek en fazla azalma miktarıdır.

2.3.3. Ünite çeşitleri

Bir enerji sisteminde çalışma şekilleri ve üretim maliyetlerine göre çeşitli üniteler yer almaktadır ve bu ünite çeşitleri aşağıda açıklanmıştır.

Sürekli çalıştırılan üniteler, bir enerji sisteminde çalışma güvenliklerinden ve ekonomik sebeplerden dolayı sürekli çalıştırılırlar, bu nedenle de çalışma planında her zaman yer alırlar.

Sabit üretimdeki üniteler başlatma ve kapatma maliyetleri gibi ekonomik nedenlerden dolayı belirli periyotlarda sürekli çalışır durumda bırakılan ünitelerdir.

Periyodik üniteler en az devrede ve devre dışı kalma zamanları göz önünde bulundurularak üretime katılan ya da çalıştırılmayan ünitelerdir.

Limit değer üniteleri ise en az devrede ve devre dışı kalma gibi kısıtları olmayan, hızla üretime başlayabilen, gaz türbinleri gibi, ünitelerdir. Enerji tesislerinde talep miktarı tepe değerlere ulaştığında arzı karşılamak için ya da rezerv enerjiyi sağlama amacıyla kullanılırlar. [4,33]

2.3.4. Amaç fonksiyonu

Ünite paylaşımı probleminin amacı kısıtları göz önünde bulundurarak toplam üretim maliyetini en aza indirmektir. Buna göre daha önce bahsedilen kısıtlar değerlendirmeye alınarak denklem 2.12'deki toplam üretim maliyeti en düşük olmalıdır.

$$\text{TM}=\sum_{j=1}^{T}\sum_{i=1}^{N}\left(a_i + b_i P_{ij} + c_i P_{ij}^2\right)+\sum_{j=1}^{T}\sum_{i=1}^{N}\sigma_i + \delta_i\left[1-e^{(-T_{i,j}^{OFF}/\tau_i)}\right] \qquad (2.12)$$

2.4. Çözüm yöntemleri

Ünite paylaşımı problemi çözümü zor, doğrusal olmayan, geniş ölçekli, karışık tam sayı kombinasyonlu bir problemdir. Problemin çözümünde öncelik listesi, dinamik programlama, lagrange-relaxation, dal-sınır metodu ve benders ayrıştırması gibi çeşitli çözüm yöntemleri geçmişte oldukça yoğun olarak kullanılmışlardır. Bu çözüm yöntemleri, problemi daha kolaylaştırmak için matematiksel programlama yaklaşımlarını bazı kesinlikler ve kabullenmelerle bir arada kullanırlar.

2.4.1. Öncelik listesi

Bu metotta üniteler maliyetlerinin düşüklüğüne göre sıraya dizilir. Daha sonra da her saat için talep edilen ve rezerv enerji karşılanana kadar en düşük maliyetli üniteden başlamak üzere sırasıyla devreye alınırlar. Bu şekilde çalışma planı oluşturulduktan sonra, en az devrede kalma zamanlarına bakılarak ihtiyaç duyulmadıkları saatlerde dahi devreye alınırlar. Sonuçta çok hızlı bir şekilde talep karşılanmış olsa da tüm planın oluşturduğu maliyet optimallikten uzaktır.

2.4.2. Dinamik programlama

Biri diğerini izleyen ve karşılıklı etkileri olan bir dizi kararın bütünüyle ele alındığı problemler için geliştirilen karar modelleri ve bunların çözümleri dinamik programlama başlığı altında incelenir. Bu yöntem, basitçe tüm mümkün çizelgeleri listeler veya dallandırır ve daha sonra uygun olmayanları listeden çıkarır ve sonuçları uygun olanları bırakır.

Dinamik programlama, problemleri tekrarlayan alt problemler şeklinde ele alıp çözen bir tekniktir. Bu yöntemde amaç, tekrarlayan problem çözümlerini saklayarak, bu çözümleri gerektiğinde tekrar kullanmaktır. Her üretilen çözüm bir tabloda (veri yapısında) saklanarak gerektiğinde tekrar tekrar kullanılmaktadır.

Ünite tahsis probleminde ilk saat için talep ve yedek enerjiyi sağlamak amacıyla devreye alınacak ünitelere karar verilir. Bu işlem ikinci saat içinde tekrarlanır, birinci

saatten ikinci saate geçildiğinde oluşacak durum değiştirme maliyetleri hesaplanarak ikinci saatte kullanılan ünitelerin durumlarını en uygun şekilde karşılayacak olan ilk saatteki ünite durumları bulunur. Bu işlem süreçteki her saat için tekrarlanarak sonuçta en uygun plan elde edilir.

Ünite sayısı ve saatlik periyotlar arttıkça hesaplanması gereken çok fazla işlem oluşacaktır ve bu da dinamik programlama ile çözümün bulunmasını oldukça zorlaştırır [35].

2.4.3. Lagrange-relaxation

Genel olarak Lagrange-relaxation ünite tahsisi problemini, çalışma takvimindeki her saat ve bütün üniteler için maliyet fonksiyonunu, her ünitenin kısıtları ve yük kısıtlamaları türünden ifade eder. Problemin yaklaşık çözümü, yük kısıtlamalarını Lagrange çarpanlarıyla maliyete eklenmesiyle bulunabileceği gösterilmiştir. Oluşturulan bu rahatlatılmış problem ünite kısıtlamalarına göre minimize edilir [11].

2.4.4. Dal ve sınır metodu

Dal sınır metodu, asıl problemden elde edilen hafifletilmiş problemler dizisini ayrı ayrı çözme tekniğidir. Çözümü arama işlemi dal sınır ağacı içerisinde yapılır. Bu ağacın tepe noktasında problemin çözülmesi en kolay olan hafifletilmiş hali bulunur. Her biri ayrı çözüm uzayına sahip problemler kümesi tepeden itibaren yer alır ve bu problemlerin çözüm uzaylarının birleşimi de tepe noktadaki problemin çözüm uzayını oluşturur. Ağaçtaki her nokta aynı şekilde problem kümelerine ayrılır, her bir kümenin çözüm uzaylarının birleşimi de bağlı oldukları noktanın gene çözüm uzayını oluşturur. Bu işlem ağacın en altındaki noktaya kadar tekrarlanır. Mevcut noktanın çözüm uzayı kendinden sonra gelen noktaların hafifletilmiş hali olduğundan, bulacağı en uygun çözüm, altındakilerden mutlaka daha az olmalıdır. Bu nedenle mevcut noktadan bulunan sonuç kendisini takip eden noktalar için alt sınır olur.

Dal sınır metodunda ana fikir, prosedürün herhangi bir kısmında, bir noktada kendinden öncekinde bulunan çözümden daha büyük bir değer bulunursa bu

noktadan aşağıda bulunan noktaların bulduğu çözümler en uygun değerler olmaz. Bu nedenle arkadan gelen noktaların değerlendirilmesine gerek kalmaz [36,37].

2.4.5. Benders ayrıştırması

Bu metot ünite paylaşımı probleminin karışık tam sayı doğrusal programlama şeklinde formülleştirilmesini gerektirir. Benders ayrıştırması problemin ünite maliyetlerinin bulunmasını içeren ana problem ve yük dağılımını belirleyen alt problem olarak ikiye ayırır. Her saat için alt problemde belirlenen üretim değerleri ana problemde çalışmasına karar verilen üniteler için kısıt olarak kullanılır. Ana problem ise ekonomik yük dağılımı için hangi ünitelerin kullanılacağını belirler. Böylelikle çözüm bulunana kadar bu iki problem tekrar tekrar çözülür [38].

3. GENETİK ALGORİTMALAR

3.1. Giriş

Genetik algoritmalar Darwin'in evrim teorisinden ilham alınarak geliştirilmiş olan evrimsel hesaplamanın bir koludur. Evrimsel hesaplama fikri ilk olarak 1960 yılında Rechenberg'in "Evolutionsstrategie" (evrimsel stratejiler) çalışmasıyla ortaya atılmıştır. Temel olarak evrim stratejileri tekniği, olası çözüm adayı olarak, ebeveyn birey ve ebeveyn bireyden mutasyon yoluyla üretilmiş olan çocuk bireyleri içermekteydi. Bu tekniğin bir sonraki evresi olan evrimsel programlama (Evolutionary Programming) günümüzde kullanılan Genetik Algoritmalara benzeyen bir yöntem olup Fogel, Owens, ve Walsh tarafından geliştirilmiştir. Burada artı olarak, üretilen çözüm adaylarının belirli bir uygunluk fonksiyonu altında değerlendirilmesi eklenmiştir.

Genetik Algoritma kelimesini ilk kullanan ve çalışmalarını 1967 yılında yayımlayan Bagel olmuştur. Aynı tarihte bağımsız olarak Rossenberg de biyolojik ve benzetim esaslı bir çalışma yapmıştır.

Michigan Üniversitesinde psikoloji ve bilgisayar bilimi uzmanı olan John Holland genetik algoritmalar konusunda ilk çalışmaları yapan kişilerdendir. Mekanik öğrenme (machine learning) konusunda çalışan Holland, Darwin'in evrim kuramından etkilenerek, canlılardaki genetik aktarım ve değişim olaylarını mekanik ve bilgisayar ortamına aktarmayı düşündü. Çünkü öğrenmenin tek bir süreçte olmasından ziyade, nesilden nesile bilgi ve özellik aktarımının kendi başına öğrenebilen devrelerin yapılmasında daha başarılı olacağını öngörüyordu. Tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine böyle yapılardan oluşan bir topluluğun çoğalma, çiftleşme, mutasyon, vb. genetik süreçlerden geçerek başarılı (öğrenebilen) yeni bireyler oluşturabildiğini gördü. Çalışmalarının sonucunu açıkladığı kitabının 1975'te yayınlanmasından sonra geliştirdiği yöntemin adı

Genetik Algoritmalar (GA) olarak yerleşti. Ancak 1985 yılında Holland'ın öğrencisi olarak doktorasını veren David E. Goldberg adlı inşaat mühendisi 1989'da konusunda bir klasik sayılan kitabını yayımlayana dek genetik algoritmaların pek pratik yararı olmayan bir araştırma konusu olduğu düşünülüyordu. Halbuki Goldberg'in gaz boru hatlarının denetimi üzerine yaptığı doktora tezi ona sadece 1985 National Science Foundation Genç Araştırmacı ödülünü kazandırmakla kalmadı, genetik algoritmaların pratik kullanımının da olabilirliğini kanıtladı. Goldberg sayesinde GA hak ettiği ilgiye kavuştu ve o tarihten itibaren birçok uygulama alanı buldu. Günümüzde birçok rastlantısal algoritma bulunmuş olmasına rağmen, GA halen popülerliğini koruyan ve çoğunlukla başarılı olan bir yöntem olmaya devam etmektedir [39-41].

3.2. En İyiyi Arama Yöntemleri

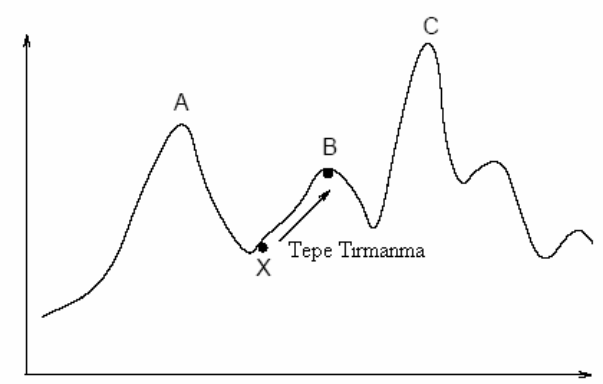
Bir problemin bilgisayar aracılığıyla çözülmesi isteniyorsa, önce sorun tespit edilmeli ve bu soruna ait denklem bulunmalıdır. Sadece fonksiyonu doğru çıkarılmış problemlerin çözümleri istenildiği gibi çıkar. Bu yüzden sorunun denklemini çıkarmak, fonksiyonu çözerken harcanacak gayretten daha çok çaba gerektirebilir.

Problem fonksiyonunu doğru bir şekilde elde ettikten sonra yapılacak ilk iş, denklemi çözecek (istenen değere yaklaştıracak) bir yöntem seçmektir. Kullanılan birçok eniyileme yöntemi vardır. Bunlardan bazıları aşağıda anlatılmıştır.

1. Türev-İntegral hesabına (calculus) dayananlar,
2. Numaralamaya (enumeration) dayananlar,
3. Rastgele aramalar (random searches) olmak üzere üç tip çözümden bahsedilebilir.

Türev-İntegral hesaplamalarına dayanan hesaplama yöntemleri üzerine çok derinlemesine çalışılmıştır. Bu yöntemler, fonksiyonun türevi alındığında bulunan köklerin, fonksiyonun en küçük ve en büyük değer veren noktaları olmasından yararlanır. Gerçek problemler için sıfır veren noktaları bulmak da ayrı bir problemdir.

Diğer bir yöntem ise alınan bir noktadan sadece yukarı ilerleyerek en iyi sonucu bulmayı hedefler. Tepe tırmanma (hill climbing) denen bu yöntem fonksiyon grafiğinin tepelerini tırmanır. Ancak çok sayıda dönme noktası içeren bir fonksiyonda şekil 3.1'de görüldüğü gibi çok sayıda tepe oluşur. Hangi tepenin en iyi çözüm olduğu bilinemez.



Şekil 3.1: Tepe Tırmanma.

Numaralama yöntemleri ise oldukça alışılagelmiştir. Sürekli olan gerçel sayı aralıkları belli sayıda parçaya ayrılarak parçalar denenir. Ancak daha büyük problemlerin çözümünde bu yöntem yeterli gelmeyebilir. Bu yöntemin biraz daha geliştirilmiş şekli Dinamik programlamayla (dynamic programming) oluşturulur. Parçalar arasından iyi görünenler seçilir. Bu parçalar, tekrar parçalara ayrılarak işlem tekrarlanır. Bu yöntem de tepe tırmanma yöntemi gibi yanlış tepeleri çözüm olarak bulabilir. Dinamik programlama tepelerin olmadığı aralıklarda başarılı ve hızlıdır.

Eniyilemenin, bir işin daha iyi yapılması ve en doğru şekilde yapılması olmak üzere iki amacı vardır. Günümüzde rastgele aramaların kullanımı artmaktadır. Bu tip aramalar eniyilemeyi daha iyi yapma amacını sağlamakta daha başarılıdırlar.

3.2.1. Genetik algoritmaların diğer yöntemlerden farkları

Genetik algoritmalar, klasik optimizasyon algoritmalarından beş temel noktada ayrılır:

1. Genetik algoritma parametrelerin kendileri ile değil, kodlarıyla uğraşır. Parametreler kodlanabildiği sürece fark etmez.
2. Genetik algoritma bir alana bağlı kalarak çözüm aramaz. Arama işlemine bir nokta grubundan başlayarak yerel optimuma takılmadan arama yapabilirler.
3. Genetik algoritmalar arama uzayında bireylerin uygunluk değerini bulmak için sadece amaç-uygunluk fonksiyonuna (objective-fitness function) ihtiyaç duyar. Bu sayede türev, diferansiyel ve diğer ek bilgilerine gerek kalmadan doğrudan amaç fonksiyonun kendisini kullanırlar.
4. Genetik algoritma ne yaptığı konusunda bilgi içermez, nasıl yaptığını bilir. Bu nedenle bir kör arama (blind search) metodudur.
5. Genetik algoritmalar olasılık kurallarına göre çalışır. Programın ne kadar iyi çalışacağı önceden kesin olarak belirlenemez [41-45].

3.3. Genetik Algoritmanın Tanımı

Genetik algoritma, doğadaki evrim mekanizmasını örnek alan bir arama metodudur ve bir veri grubundan özel bir veriyi bulmak için kullanılır. Genetik algoritmalar 1970'lerin başında John Holland tarafından ortaya atılmıştır. Genetik Algoritmalar, Evrimsel Genetik ve Darwin'in Doğal seleksiyonuna benzerlik kurularak geliştirilmiş "iteratif " ihtimalli bir arama metodudur.

Genetik algoritmalar doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler üretir. Yani doğadaki, güçlü olan hayatta kalır mantığı genetik algoritmada da geçerlidir. Bu yöntemi sayılar üzerine uygularken dikkat edilecek nokta ise hangi çözümün daha iyi (güçlü) olduğunu belirleyebilmektir. Bunun için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için yeniden kopyalama (recombination), değiştirme (mutation) gibi operatörleri kullanır. Genetik algoritmaların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilir.

Genetik algoritmadaki en önemli özellik olayların tamamen rastlantısal olmasıdır. Örneğin, bir çözümün diğer çözüme tercih edilmesi, çözümün istenen sonuca ne

kadar yaklaştığıyla ilgili olsa da, tamamen olasılığa bağlıdır ve iyi çözümün seçilmesi hiçbir zaman garanti altında değildir. Bu durum genetik algoritmaların başarısının ardındaki sırdır. Yani doğada da olduğu gibi aslında her zaman güçlüler hayatta kalmaz. Bazen güçlülerin ölüp, güçsüzlerin hayatta kalması gereklidir.

3.3.1. Şema teoremi

Genetik algoritmaların çalışmasını açıklamaya yönelik pek çok çalışma yapılmıştır. Bu çalışmalardan en çok bilineni John Holland tarafından ortaya atılan şema (schema ya da schemata) kuramıdır. Şema ikili dizileri göstermek için kullanılan bir gösterimdir. Holland bu kuramı ikili diziler üzerinde temel bir genetik algoritma için tanımlamıştır. Bu yönteme göre genetik algoritma iyi yapılar ortaya çıkarır, çoğaltır ve birleştirir. Holland'ın şema teoremi hala genetik algoritmaların başarısını açıklamak için en başarılı teorem olarak kullanılır. Genel olarak bir genetik algoritma ikili kodlama (binary strings) sistemi üzerinde çalışır. Bu kodlamalar doğal sistemlerdeki kodlamalar ile ilgilidir. Genetik algoritmalarda oluşan başarılı bireyler incelenirse, bu bireyler arasındaki benzerlikler bulunabilir. Bu benzerliklerden yola çıkarak şemalar oluşturulabilir. İkili dizi kodlaması için 0,1 ve * (* o konumda 0 veya 1 olmasının önemsiz olduğunu gösterir) ifadelerinin kullanıldığı yöntem önerilebilir. Örnek olarak H alt dizisi ikinci ve dördüncü bitleri 1, altıncı biti 0 olan kromozomlar kümesi kabul edilirse: H= * 1 * 1 * 0 olur.

Eğer bir 3 dizisi, alt dizinin kalıbına uyarsa 3 dizisine "H'nin bir örneğidir" denir. Alt dizilerin iki özelliği mevcuttur. Bu özellikler aşağıda verilmiştir.

Alt dizi derecesi: Bir H alt dizisinin derecesi o(H) ile gösterilir ve mevcut alt dizi kalıbında bulunan sabit konumların sayısıdır. Bu sayı ikili alfabede 0 ve 1 değerlerinin sayısının toplamına eşittir.

Alt dizi uzunluğu: Bir H alt dizisinin uzunluğu δ(H) ile gösterilir ve mevcut alt dizi kalıbında bulunan belirli ilk ve son konumlar arasındaki uzaklıktır.

Alt dizi derecesi ve alt dizi uzunluğu kavramlarının genetik algoritmaların temel teoreminde son derece önemli bir yeri vardır. Alt dizi derecesi düşük, alt dizi uzunluğu kısa olan diziler "yapı blokları" olarak adlandırılır. John Holland, genetik algoritmaların işleyişinde uygun yapı bloklarının tanımlanmasını ve bu yapı bloklarının daha uygun yapı blokları elde etmek amacıyla birleştirilmesini önermektedir. Bu fikir yapı blokları hipotezi olarak bilinmektedir. Genetik algoritmanın temel teoremi ise şöyle açıklanmaktadır.

Popülasyon ortalamasının üstünde uyum gücü gösteren, kısa uzunluğa ve düşük dereceye sahip alt diziler zamanın ilerlemesiyle üstsel olarak çoğalırlar. Bu çoğalma, genetik işlemler aracılığı ile gerçekleşmektedir ve sonucunda ana-babadan daha üstün özellikler taşıyan bireyler ortaya çıkmaktadır. Bu çözüm kalitesinin kuşaktan kuşağa artması iki nedene bağlanmaktadır. Bu nedenler şöyle açıklanabilir:

Başarısız olan bireylerin üreme şansları azaltıldığı için kötüye gidiş zorlaşmaktadır.
Genetik algoritmaların yapısı kötüye gidişi engellemekle kalmamakta, genetik algoritmaların temel teoremi uyarınca, zaman içinde hızlı bir iyiye gidiş de sağlayabilmektedir.

Genetik algoritmalar yapısı gereği, kötü bireyleri yani uygun olmayan çözümleri, operatörleri sayesinde elemektedir. Bu işlemler bir döngü içerisinde durdurma kriteri sağlanana kadar devam etmektedir [44,45].

3.4. Genetik Algoritmada Kullanılan Terimler

Genetik algoritmaların çalışmasında ve başarılı çözüm değerlerine ulaşmada algoritma yapısında kullanılan kavramların ve bu kavram değerlerinin iyi belirlenmesi gerekmektedir. Aşağıda bu kavramların açıklamaları verilmiştir.

3.4.1. Gen

Kromozom yapısında kendi başına birer genetik bilgi taşıyan en ufak yapı birimine gen denir.gen denir. Çözümle ilgili kısmi bilgiler içeren bu ufak yapıların bir araya gelmesiyle bütün bir çözüm kümesini oluşturan kromozomlar meydana gelir.

Genetik algoritmaların kullanıldığı programlama yapısında gen yapıları tasarımcının tanımlamasına bağlıdır. Bir gen dizisi ikilik tabandaki sayılardan oluşabileceği gibi onluk veya onaltılık tabandaki sayı değerlerinden de oluşabilir.

3.4.2. Kromozom

Birden fazla gen yapısının bir araya gelerek oluşturduğu problemin çözümüne ait tüm bilgiyi içeren dizilere kromozom denir. Kromozomların bir araya gelmesiyle popülasyon (nüfus) oluşturulur. Nüfusu oluşturan her bireye kromozom, kromozomdaki her bilgiye de gen denilir. Kromozomlar, üzerinde çalışılan problemin olası çözüm bilgilerini içerir, çözüm kümesidir.

3.4.3. Bireyler

Bir bireyin oluşabilmesi için en az bir kromozom gereklidir. Kromozomların ne kadar çok olacağını hata fonksiyonunun giriş değişken sayısı belirler. Yani birey (fert) denilince anlaşılması gereken hata fonksiyonunun yalnızca bir çözüm üretmesi için girilmesi gereken sayıların topluluğudur.

3.4.4. Popülasyon (Nüfus)

Popülasyon, problemle ilgili çözüm bilgilerini içeren kromozomların bir araya gelerek oluşturduğu yapıdır. Nüfus içerisindeki kromozom sayısı sabit olup problemin özelliğine göre tasarımcı tarafından başlangıçta kaç tane olacaklarına karar verilir.

Genetik algoritmanın çalışması süresince nüfus içerisinde yer alan kromozomların büyük bir çoğunluğu yok olarak yerlerine yeni üretilen kromozomlar gelir ve bu sayede de nüfus büyüklüğü her zaman sabit kalmış olur.

Popülasyonun büyüklüğü çözüme ulaşmada geçen süreyi doğrudan etkilemektedir. Fazla sayıda kromozomdan oluşan popülasyon problemin çözüm süresini uzatırken, az sayıdaki kromozomdan oluşan popülasyon en iyi çözüm değerine ulaşılamamasına neden olabilir.

3.4.5. Yeniden üretim

Mevcut popülasyonda bulunan bireylerden gelecek nesile yeni bireyler aktarmak için yapılan işlemdir. Seçilen bireyler genetik olarak mevcut popülasyonda çözüm için en iyi değere sahip olanlardır. Bu işlem belirlenen uygunluk değerlerine sahip bireylerden daha iyi uygunluk değerine sahip bireylerin sonraki nesile aktarılmasını sağlar.

3.4.6. Uygunluk değeri

Bir nüfus oluşturulduktan sonraki ilk adım, popülasyondaki her üyenin uygunluk değerini hesaplamadır. Genetik algoritma ile problem çözümünde, probleme uygun çözüm kümesinin elde edilebilmesi için uygunluk fonksiyonunun (fitness function) belirlenmesi gerekmektedir. Bu uygunluk fonksiyonu aşağıdaki kurala uygun seçilmelidir.

$$Pi = \frac{Fi}{\sum_{j=1}^{n} Fj} \tag{3.1}$$

Burada n popülasyon büyüklüğünü Fi, i. bireyin, o popülasyon içindeki uygunluk değerini, $\sum_{j=1}^{n} Fj$ popülasyonun toplam uygunluk değerini ve Pi ise i. birey için seçilme olasılığını gösterir [4].

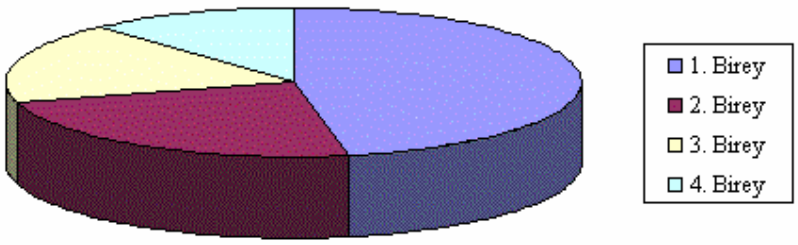
Verilen belirli bir kromozom için uygunluk fonksiyonu, o kromozomun temsil ettiği çözümün kullanımıyla veya yeteneğiyle orantılı olan sayısal bir uygunluk değeri verir. Uygunluk fonksiyonunun genetik algoritma yapısı içerisinde kullanılması ile problemin çözümünde yarar sağlayacak olan kromozomların yaşama şansının artırılması, problemin çözümünde yarar sağlamayan kromozomların ise popülasyon içerisinden ayıklanmasına olanak sağlanır (doğal seleksiyon ilkesi). Böylece algoritma, başarısız olan aday çözümlerini elimine eder.

3.4.7. Seçim mekanizmaları

Başlangıç popülasyonu oluşturulduktan sonra yeni nesillerin yaratılabilmesi için mevcut popülasyondan en iyi uygunluk değerine sahip bireyler arasından seçim yapılması gereklidir. Seçimdeki amaç yüksek uygunluk değerine sahip bireylerin seçim mekanizmasıyla çaprazlama havuzuna alınarak diğer bireylerin yok olmalarının sağlanmasıdır. Bu sayede yüksek uyuma sahip bireylerin iyi özellikleri sonraki nesillere aktarılmış olunur.[41]

3.4.7.1. Rulet tekerleği

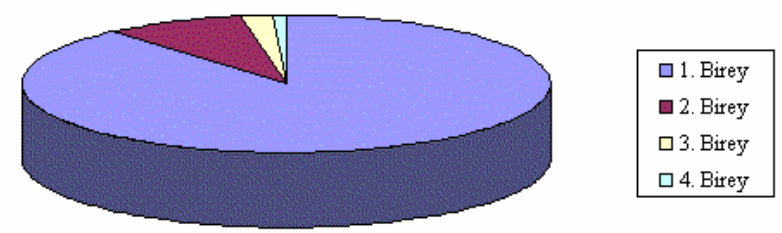
Rulet tekerleği seçiminde, rulet birey adedi kadar dilime ayrılmıştır ve dilimlerin her biri bir bireyin seçilme olasılığını temsil eder. Her bireyin uygunluk değeri toplam uygunluk değerine bölünerek, rulette yüzdelik dilim olarak alacakları dilim büyüklüğü belirlenir. Şekil 3.2' bireylerin uygunluklarına göre kapladıkları alanlar gösterilmiştir.



Şekil 3.2: Rulet tekerleği.

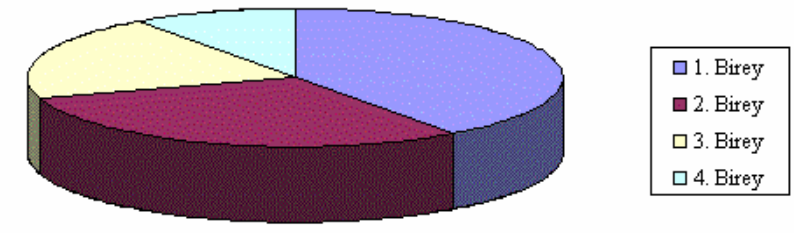
Tekrar üreme için rulet tekerleğinin döndürülmesi gerekmektedir. Bunun için sıfırla toplam uygunluk arasında rastgele bir sayı üretilerek bu sayının ruletin hangi parçasına karşılık geldiğine bakılarak birey seçilir. Benzer şekilde diğer bireylerin de belirlenmesiyle uygunluk değerleri en yüksek olan bireyler çaprazlama havuzuna alınır. Aynı işlem her döngüde devam ederek yeni nesiller elde edilir. Uygunluk değeri yüksek olan bireyin seçilme şansı yüksek gözükmesine rağmen seçilmesi garanti altında değildir.

Rulet tekerleği yönteminde bireylerin uygunluk değerleri arasında çok büyük farklar varsa seçim işlemi amacına uygun hareket etmemeye başlayacaktır. Örneğin en iyi bireyin uygunluğu tüm rulet tekerleğinin %90'ı ise şekil 3.3'de gösterildiği gibi diğer bireylerin seçilme şansları çok az olacaktır. Bunu önlemek için sıralı seçim kullanılmalıdır.



Şekil 3.3: Sıralı seçimden önceki kromozomların rulet tekerleğindeki durumu.

Sıralı seçimde en kötü uygunluk değerine sahip olan kromozoma 1 değeri, sonraki kromozoma 2 değeri ve en iyi kromozoma da popülasyondaki kromozom sayısının değeri verilerek şekil 4.4'de görüldüğü üzere diğerlerine seçilme şansı verilmiş olur. Fakat bu yöntem de daha geç olarak çözüme yaklaşılmasına neden olabilir.



Şekil 3.4: Sıralı seçimden sonraki kromozomların rulet çemberindeki durumu.

3.4.7.2. Turnuva seçimi

Bu metotta popülasyondan rastgele bir grup kromozom seçilir. Grup içindeki en iyi uygunluk değerine sahip kromozom ebeveyn olarak seçilir. Popülasyon genişliğine ulaşılıncaya kadar bu işleme devam edilir ve ayrıca yeni oluşan nesil içerisinde aynı kromozomdan birden fazla olabilir.

3.4.7.3. Kararlı hal seçimi

Bu seçim metodunda her nesilde sadece birkaç kromozom yenileriyle yer değiştirir. Bunun için rastgele bir grup kromozom seçilir ve bunlardan uygunluk değeri kötü olanlar genetik operatörlerle yeniden üretilerek yeni nesilde yer alırlar.

3.4.7.4. Elitizm

Genetik operatörler işletilirken mevcut dönemdeki en iyi uygunluk değerine sahip birey bir sonraki popülasyona aktarılamayabilir. Bunun önüne geçmek için mevcut bireylerden en iyisi ya da en iyileri bir sonraki popülasyona doğrudan eklenmelidir.

Elitizm, uygunluk değerleri en iyi olan belirli sayıdaki ebeveyn bireylerin, çaprazlama ve mutasyon işlemi ile iyi özellikleri bozulmadan bir sonraki popülasyona aktarılmalarıdır.

Elitizmin uygulanmadığı standart genetik algoritmalarda, mutasyon işleminden sonra, çocuk bireyler yeni popülasyonu oluşturmaktadır. Yeni nüfus oluşturulurken, çocuk bireylerin sayısının ebeveyn bireylerin sayısına eşit olmasına dikkat edilir. Yani, ebeveynlerin tamamı nüfustan çıkartılıp, yerlerine çocuk bireyler konulur. Böylece, her bir bireyin sadece bir kuşak yaşamasına izin verilmektedir. Ancak, oldukça basit olan bu yöntemde, iyi ebeveynler iyi çocuklar üretmeden popülasyondan çıkabilmekte ve çözüm için iyi sonuç verebilecek faydalı bilgiler kaybolabilmektedir.

Elitizm uygulandığında, her bir kuşakta ebeveyn bireyler içerisinden, önceden belirlenmiş sayıda en iyi birey, en kötü çocuk bireylerle yer değiştirerek yeni

popülasyona katılmaktadır. Böylece, iyi bireylerin bir kuşak yerine, birkaç kuşak veya daha iyi bireyler üretilene kadar yaşaması sağlanmış olmaktadır. Elitizm ile bireylerin iyi özelliklerinin kaybolması engellenmektedir [54].

3.4.8. Çaprazlama

Çaprazlama genetik algoritmalarda kullanılan en önemli mekanizmadır. Çaprazlamadaki amaç popülasyon içerisindeki iyi özellikli bireylerden daha iyi çözümler bulan bireylerin yaratılmasıdır. Uygulamada çok fazla çaprazlama yöntemi olmakla beraber hepsinde yeni birey oluşturmak için ebeveyn kromozomlar arasında gen alış verişi yapılır [46].

3.4.8.1. Binary çaprazlama

Bu çaprazlama mekanizmasında kromozomlardaki genleri ifade etmek için sadece binary sayılar kullanılır.

3.4.8.1.1. Tek noktalı çaprazlama

Bu metot rastgele seçilen bir çaprazlama noktasından itibaren önceki veya sonraki genlerin yerlerinin değiştirilmesi işlemidir. Eşleşen iki kromozomda, bu çaprazlama noktasının sağında kalan bölümlerin yeri değiştirilerek yeni kromozomlar elde edilmiş olur.

Çaprazlama işleminde göz önünde bulundurulması gereken husus çaprazlama noktasının seçimidir. Çaprazlama noktasının yeri oluşacak yeni kromozomların ebeveynlerinden ne kadar farklı olacağını gösterir.

E1:	1	1	0	0	1	1	0	1	0	1	1	1	1	0	1	1
E2:	1	0	0	1	0	1	0	1	1	1	0	0	0	1	1	1
Çaprazlama işleminden sonra oluşan bireyler																
E1':	1	1	0	0	1	1	0	1	0	1	1	0	0	1	1	1
E2':	1	0	0	1	0	1	0	1	1	1	0	1	1	0	1	1

Şekil 3.5: Tek noktalı çaprazlama.

3.4.8.1.2. Çift noktalı çaprazlama

Bu metotta tek noktalı çaprazlamadan farklı olarak çaprazlama için iki adet çaprazlama noktası belirlenir ve daha sonra eşleşen iki kromozomun bu çaprazlama noktaları arasında kalan bölümleri yer değiştirilerek yeni kromozomlar elde edilir.

E1:	1	1	0	0	1	1	0	1	0	1	1	1	1	0	1	1
E2:	1	0	0	1	0	1	0	1	1	1	0	0	0	1	1	1

Çaprazlama işleminden sonra oluşan bireyler

E1:	1	1	0	0	1	1	0	1	1	1	1	1	1	0	1	1
E2:	1	0	0	1	0	1	0	1	0	1	0	0	0	1	1	1

Şekil 3.6: Çift noktalı çaprazlama.

3.4.8.1.3. Çok noktalı çaprazlama

Eşleşen kromozomların rastgele seçilmiş iki veya daha fazla noktalarından parçalanarak yapılan çaprazlamaya çok noktalı çaprazlama denir. Şekil 3.7'de nasıl yapıldığı gösterilmiştir.

E1:	1	1	0	0	1	1	0	1	0	1	1	1	1	0	1	1
E2:	1	0	0	1	0	1	0	1	1	1	0	0	0	1	1	1

Çaprazlama işleminden sonra oluşan bireyler

E1:	1	1	0	1	1	1	0	1	0	1	0	0	1	0	1	1
E2:	1	0	0	0	0	1	0	1	1	1	1	1	0	1	1	1

Şekil 3.7: Çok noktalı çaprazlama.

3.4.8.1.4. Düzgün çaprazlama

Diğer bir çaprazlama türü ise düzgün çaprazlamadır. Düzgün çaprazlama işlemi için öncelikle çaprazlama maskesi tanımlanır. Tek noktalı ve çok noktalı çaprazlamada, rastgele çaprazlama noktası seçilmektedir. Düzgün çaprazlamada ise, her nokta bir çaprazlama noktası olarak alınmaktadır. Düzgün çaprazlamada ebeveyn bireylerle eşit uzunlukta maskeler kullanılmaktadır. Maske rastgele üretilen bitleri içermektedir. Birinci çocuk birey için bir maske üretilmekte, ikinci çocuk birey için birinci maskenin tersi alınmaktadır. Düzgün çaprazlama için aşağıda bir örnek verilmiştir.

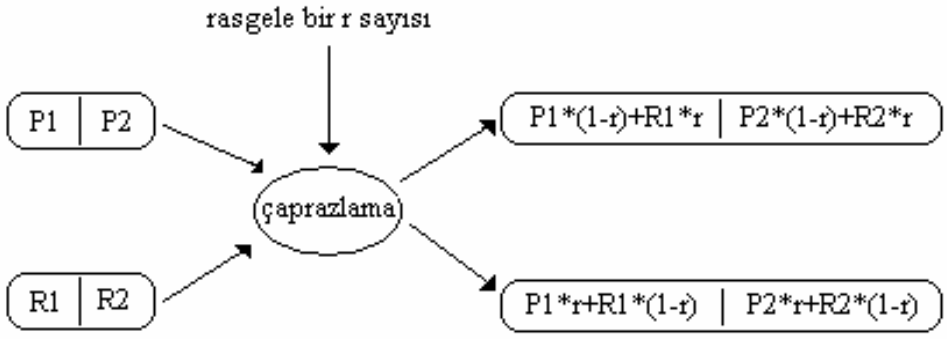
Örnekte, çocuk bireyler oluşturulurken, maskedeki 1'ler için birinci ebeveynin bitleri, 0'lar için ikinci ebeveynin bitleri çocuk bireylere taşınmaktadır.

Ebeveyn bireyler																
E1:	1	1	0	0	1	1	0	1	0	1	1	1	1	0	1	1
E2:	1	0	0	1	0	1	0	1	1	1	0	0	0	1	1	1
Maskeler																
M1:	0	1	1	0	0	0	1	1	0	1	0	1	1	0	0	1
M2:	1	0	0	1	1	1	0	0	1	0	1	0	0	1	1	0
Çocuk Bireyler																
Ç1:	1	1	0	1	0	1	0	1	1	1	0	1	1	1	1	1
Ç2:	1	0	0	0	1	1	0	1	0	1	1	0	0	0	1	1

Şekil 3.8: Düzgün çaprazlama.

3.4.8.2. Gerçel sayı çaprazlama

Gerçel sayı çaprazlamasında ebeveyn bireyler parametrelerine ayrılırlar ve her bir parametre etiketlenir. Örnek olarak şekil 3.9'daki gibi iki ebeveyn alınabilir ve bu iki ebeveynin parametreleri P1,P2,R1 ve R2 olarak etiketlenir.



Şekil 3.9: Gerçel sayı çaprazlaması formülasyonu.

Örnekteki r sayısı 0 ile 1 aralığında rastgele üretilir. Çaprazlamayla üretilen çocuk bireylerin parametre değerleri, etiketlenmiş ebeveyn parametre değerleri ile "r" rastgele sayısının aşağıda görülen formülün kullanılmasıyla elde edilir.

Gerçel sayı çaprazlamasını uygulanmış hali şekil 3.10'da verilmiştir.

E1:	5,3	4,1
E2:	2,3	1,5

r = 0.77 rastgele sayısıyla yapılan çaprazlamada oluşan bireyler

Ç1:	4,61	3,5
Ç2:	2,99	2,09

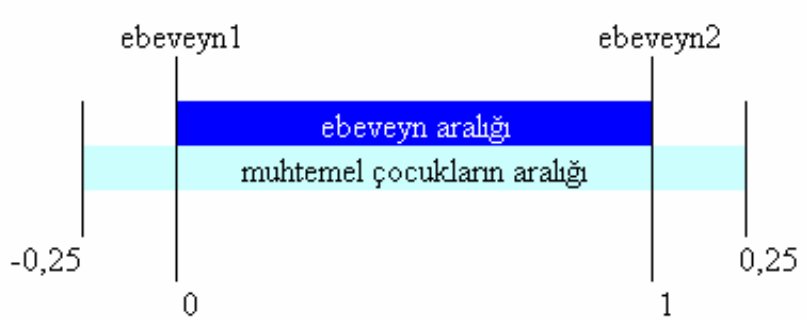
Şekil 3.10: Gerçel sayı çaprazlaması.

3.4.8.2.1. Ölçekli çaprazlama

Ölçekli çaprazlama, sadece gerçel sayı kodlaması yapılmış bireylerin çaprazlamasında kullanılabilir. Bu tip çaprazlamada, yeni oluşturulan çocuk bireylerin parametre değerleri, ebeveyn bireylerin parametre değerleri civarında olmaktadır. Yeni bireylerin parametre değerleri denklem 3.2 kullanılarak hesaplanır.

$$\text{çocuk}=\text{ebeveyn1}+\alpha(\text{ebeveyn2}-\text{ebeveyn1}) \quad (3.2)$$

Burada, α değeri [-d,1+d] aralığında rastgele seçilen ölçeklendirme faktörüdür. Her yeni çocuk bireyin parametreleri için farklı α değerleri üretilmesi gerekmektedir. Şekil 3.11'de d değeri 0,25 alınarak ve ölçekli çaprazlama kullanılarak üretilen çocuk bireyler ile ebeveyn bireylerin değişim aralığı gösterilmiştir.



Şekil 3.11: Ölçekli çaprazlama.

Ölçekli çaprazlama uygulandıktan sonra oluşacak yeni birey şekil 3.12'de verilmiştir.

Ebeveyn bireyler			
E1:	12	25	5
E2:	123	4	34
α değerleri:			
Birey1	0,5	1,1	-0,1
Birey2	0,1	0,8	0,5
Çocuk Bireyler			
Ç1:	67,5	1,9	2,1
Ç2:	23,1	8,2	19,5

Şekil 3.12: Ölçekli çaprazlamayla elde edilecek bireyler.

3.4.9. Mutasyon

Mutasyonda amaç, nüfusu oluşturan bireylerin çeşitliliğini artırmaktır. İyi oluşturulamayan başlangıç nüfusu, çözüm kümesi içerisinde çeşitliliği sağlayamaz. Ayrıca, evrim süreci içerisinde bireylerin birbirine çok yaklaşmaları da çeşitliliğin azalması demektir. Çeşitliliğin sağlanması için, mutasyon seçim aşamasında veya çaprazlamadan sonra uygulanabilmektedir. Yaygın kullanım çaprazlamadan sonra kullanma yönündedir.

Genetik algoritma ile çözüm üretme işleminde, çaprazlamanın gerçekleştirilmesinden sonra, yeni bireyler mutasyona tabi tutulurlar. Bu nüfus içerisindeki bireylerin çoğunluğunun yerel minimuma takılmasını engeller. Mutasyon işleminde, yeni bireylerin değişkenleri mutasyon oranı adı verilen düşük bir olasılıkla değişime uğratılır. Değişim biçimi, bireylerin kodlama biçimine bağlıdır. Gerçel kodlamada, bireyin mutasyona uğratılacak parametrelerine çok küçük bir sayı olan mutasyon olasılıklar eklenir veya çıkartılır. İkili kodlamada ise, bireyi oluşturan bitlerin tek tek mutasyon olasılıklarına göre tersi alınarak mutasyona uğratılır [39].

Mutasyon oranı değişken sayısıyla ters orantılıdır. Değişken sayısı arttıkça mutasyon oranı azalmaktadır. Mutasyon oranını belirleyen kesin bir koşul veya kriter yoktur. Farklı çalışmalarda mutasyon oranı olarak 0,1 ile 0,001 arasında değerler kullanılıştır. Standart genetik algoritmalarda, mutasyon ikinci derecede rol oynamaktadır. Görevi çaprazlama sonucunda kaybedilen iyi özellikli bilgileri tekrar kazanabilmektir. Bazı araştırmacılar, mutasyon işleminin görevinin sadece çaprazlama kullanılarak

gerçekleştirilebileceği düşüncesiyle mutasyona gerek olmadığını belirtmektedirler. Bazı araştırmacılar ise mutasyon oranının çok yüksek, hatta çaprazlama oranından da fazla olması gerekliliğini savunmaktadırlar [47].

3.4.9.1. İkili kodlamada mutasyon işlemi

İkili kodlamada, mutasyona uğratılacak bitin tersi alınmaktadır. Bireyin hangi bitlerinin mutasyona uğrayacağı mutasyon oranı ile belirlenmektedir. Her bir bit için [0,1] aralığında rastgele bir sayı üretilir. Üretilen sayı önceden belirlenen mutasyon oranından küçük ise, tersi alınarak bit mutasyona uğratılır. Şekil 3.13'de bu işleme bir örnek verilmiştir.

Mutasyon oranı:0,005								
Mutasyon olasılıkları	0,561	0,814	0,207	0,361	0,147	0,019	**0,003**	0,963
Mutasyondan önce	1	1	0	0	1	1	**0**	1
Mutasyondan sonra	1	1	0	0	1	1	**1**	1

Şekil 3.13: Binary kodlu için mutasyon işlemi.

34.9.2. Gerçel kodlamada mutasyon işlemi

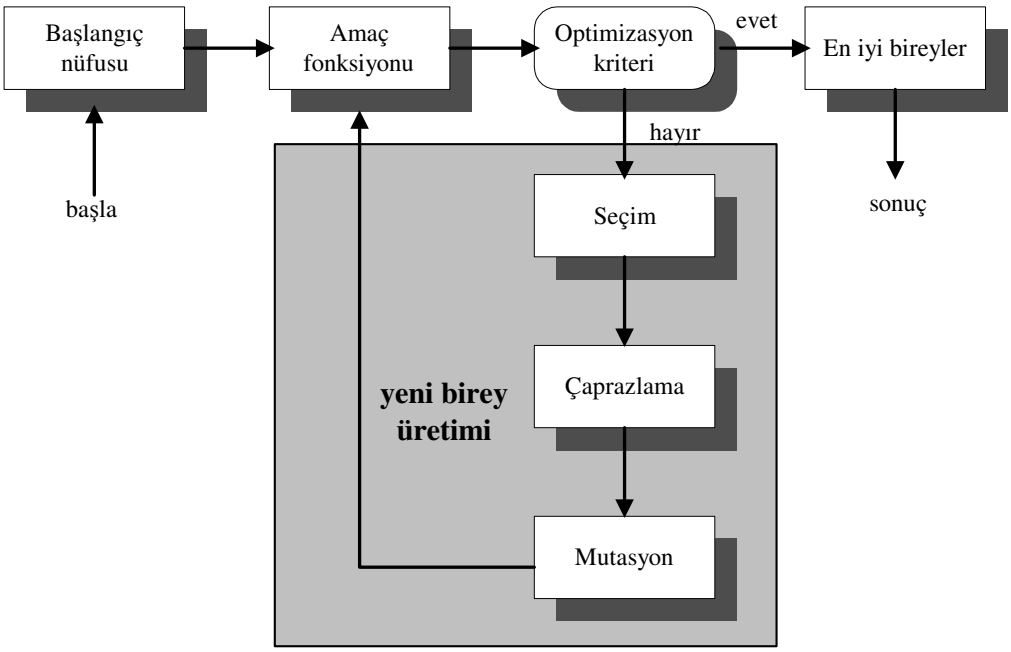
Bu metotta mutasyona uğratılacak değişkenin değerine rastgele bir sayı eklenir veya çıkartılır. Bireyin hangi değişkeninin mutasyona uğrayacağı mutasyon oranı ile belirlenmektedir. Her bir değişken için [0,1] aralığında rastgele bir sayı üretilir. Üretilen sayı mutasyon oranından küçük ise, sayı değişkene eklenerek değişken mutasyona uğratılır. Şekil 3.14'de bu işleme örnek verilmiştir.

Mutasyon oranı:0,005								
Mutasyon olasılıkları	0,561	0,814	0,207	0,361	0,147	0,019	**0,003**	0,963
Mutasyondan önce	4,384	2,756	0,547	-3,18	6,831	0,214	**0,351**	0,412
Mutasyondan sonra	4,384	2,756	0,547	-3,18	6,831	0,214	**0,354**	0,412

Şekil 3.14: Gerçel kodlu birey için mutasyon işlemi.

3.5. Genetik Algoritmanın Çalışması

Genetik algoritmalar ile problem çözümünde ilk adım, rastgele üretilen, büyüklüğü veya ne kadar birey içereceği probleme bağlı olan, başlangıç nüfusunun oluşturulmasıdır (Şekil 3.15). Nüfus içerisindeki bireyler, parametre veya değişken bilgilerini temsil eden kodları içermektedir.



Şekil 3.15: Standart genetik algoritma.

İkinci adımda, başlangıç nüfusunun bireyleri için, amaç fonksiyon değerleri hesaplanır ve bu değerlere bağlı olarak uygunluk değerleri atanır. Uygunluk değerleri, nüfus içerisindeki bireylerin ne oranda çözüm sağladıklarını gösteren sayısal değerlerdir.

Üçüncü adımda, nüfus içerisindeki bireylerin optimizasyon kriterini sağlayıp sağlamadıkları kontrol edilir. Optimizasyon kriteri, bireylerin amaç fonksiyon ve uygunluk değerleri için bir eşik değeri olabileceği gibi, önceden belirlenmiş bir üretim döngü sayısı da olabilmektedir. Optimizasyon kriteri sağlanmıyorsa, yeni nüfus üretimine geçilir.

Yeni (çocuk) bireylerin üretiminde kullanılacak olan mevcut (ebeveyn) bireylerin seçiminde, uygunluk değerleri esas alınır. Çocuk bireylerin üretimi için önce çaprazlama, sonra da mutasyon adımları izlenir. Çocuk bireylerin uygunluk değerleri hesaplanır ve ebeveyn bireylerle beraber yeni nüfus içerisine katılır. Yeni nüfus üretim döngüsü optimizasyon kriteri sağlanıncaya kadar sürdürülür. Optimizasyon kriteri sağlanıyor ise, nüfus içerisindeki uygunluk değeri en iyi olan birey problemin çözümünü oluşturmaktadır. Bu işlem problem çözümünün son adımıdır.

3.5.1. Genetik algoritmanın çalışma adımları

Genetik algoritmalarda çözüme ulaşmak için yapılması gereken adımlar şekil 3.16'da gösterilmiştir.

1 Başlangıç popülasyonunu rastgele oluştur
2 (Kuşak sayısı<maksimum kuşak sayısı) olduğu sürece tekrarla
    a. Popülasyondaki kromozomların uygunluk değerlerini bul
    b. Yeni boş popülasyon oluştur
    c. En iyi bireyleri yeni popülasyona kopyala
    d. Yeni popülasyon doluncaya kadar tekrarla
        i. Seçme mekanizmasıyla iki birey al
        ii. Çaprazlama uygula
        iii. Mutasyon uygula
        iv. Yeni popülasyona ekle
    e. Eski popülasyonu yok et
3 Kuşak sayısı=Kuşak sayısı+1
4 En iyi bireyleri göster

Şekil 3.16: Genetik algoritma.

Yapılan işlemler açıklanacak olursa;

1) Tasarımcının önceden kararlaştırdığı miktarlara göre rastgele bir çözüm grubu oluşturulur. Arama uzayındaki bütün çözüm noktalarını gösteren bilgileri

kromozomlar saklar ve ilk popülasyondaki kromozomları oluşturan genler rastgele seçilir.

2) Kuşak değeri sayısı sıfırlanarak çözümü bulmak amacıyla önceden karar verilen iterasyon sayısı kadar tekrarlama yapılır.

a) Dizilerin ne kadar iyi çözüm olduklarını gösteren işleve uygunluk değeri denir. Her bir kromozom için uygunluk değeri hesaplanır.

b) Her iterasyonda üreme işleminden sonra yeni popülasyon oluşturulacağından yer ayrılır.

c) Karar verilen elitizm oranına göre mevcut kuşaktaki en iyi uygunluk değerine sahip birey ya da bireyler genetik operatörlere takılmadan doğrudan yeni popülasyona gönderilir. Bu sayede mevcut dönemdeki en iyi bireylerin sahip olduğu bilgilerin yok olması engellenmiş olur.

d) i. Kromozomların seçilmesi, kullanılan seçim mekanizmasıyla uygunluk değerlerine göre yapılır. Seçme işleminde uygunluk değerleri en yüksek olan kromozomların seçilme oranları diğerlerine oranla daha yüksek olmalıdır.

ii. Rastgele seçilmiş olan iki kromozom çaprazlama oranına göre çaprazlanarak, yeni popülasyonda yer alacak iki yeni kromozom belirlenmiş olur. Çaprazlama işlemi yeni popülasyonda yer alacak birey sayısına ulaşana dek sürer.

iii. Çaprazlama sonucunda elde edilen kromozomlardaki genlerden bazıları mutasyon oranına göre rastlantısal olarak değiştirilir. Bu sayede çözümlerin yerel minimumlardan kurtulması sağlanmış olur.

iv. Genetik operatörlerle oluşturulan yeni kromozomlar yeni popülasyona eklenirler.

e) Yeni popülasyon tamamen oluşturulduğunda mevcut dönemde yer alan eski kromozomlar yok edilirler.

3) Kuşak sayısı bir artırılarak üçüncü adıma dönülür, uygunluklarına göre kromozomlar tekrar seçilerek genetik üreme işlemlerine katılırlar.

4) İstenilen kuşak sayısına kadar yukarıdaki işlem basamakları tekrar edilir ve sonuçta en iyi uygunluk değerine sahip olan kromozom çözüm olarak seçilir [31].

4. GENETİK ALGORİTMALARLA KISA DÖNEM ELEKTRİK ENERJİSİ ÜRETİM PLANLAMASI UYGULAMASI

İlk üç bölümde ünite paylaşımı problemi, çözümü için uygulanan metotlara ve genetik algoritmalara değinilmiştir. Bu bölümde ise çalışmanın amacı olan örnek uygulama hakkında bilgi verilecektir.

Bu çalışmada genetik algoritmalar, klasik optimizasyon yöntemlerine olan üstünlüklerinden dolayı problemin çözümünde kullanılacaktır. Bu bölümde öncelikle ünite paylaşımı probleminin uygulaması yapılacağı ünite ve yük bilgileri verilecek, daha sonra da matematiksel modelde kullanılan formüller açıklanacaktır. Sonraki aşamada ise uygulamada kullanılan farklı kuşak sayısı, mutasyon oranı ve nüfus büyüklüğünün sonuçlara nasıl etki ettiği detaylı olarak açıklanacaktır.

4.1. Uygulamada Kullanılacak Test Sistemi

Çözüm metodu olarak genetik algoritmada kullanılacak test sistemindeki ünitelere ait katsayılar ve günlük yük talebi aşağıdaki tablo 4.1 ve tablo 4.2' de verilmiştir [27].

Tablo 4.1: Çözülecek probleme ait katsayılar

	Ünite1	Ünite2	Ünite3	Ünite4	Ünite5
Pmax (MW)	455	130	130	80	55
Pmin (MW)	150	20	20	20	55
a ($/h)	1000	700	680	370	660
b ($/MWh)	16.19	16.60	16.50	22.26	25.92
c ($/MW h)	0.00048	0.002	0.00211	0.00712	0.00413
en az devrede (s)	8	5	5	3	1
en az devre dışı (s)	8	5	5	3	1
sıcak başlatma maliyeti ($)	4500	550	560	170	30
soğuk başlatma maliyeti ($)	9000	1100	1120	340	60
soğuk başlatma saati (s)	5	4	4	2	0
mevcut durumlar (s)	8	-5	-5	-3	-1

Tablo 4.2: Çözülecek probleme ait saatlere göre talep edilen enerji miktarları.

Saat (s)	1	2	3	4	5	6	7	8	9	10	11	12
Talep MW)	400	450	480	500	530	550	580	600	620	650	680	700
Saat (s)	13	14	15	16	17	18	19	20	21	22	23	24
Talep (MW)	650	620	600	550	500	550	600	650	600	550	500	450

Tablo 4.1'de gösterilen ifadeleri açıklamak gerekirse;

Pmax : ünitenin verebileceği maksimum enerji miktarı

Pmin : üniteden alınması gereken minimum enerji miktarı

a, b, c : ünitenin yakıt maliyeti katsayıları

en az devrede : ünitenin başlatıldıktan sonra çalışır durumda kalması gereken süre

en az devre dışı : ünitenin durdurulduktan sonra kapalı durumda kalması gereken süre

sıcak başlatma maliyeti : üniteler tamamen soğumadan devreye alındıklarında oluşan maliyet

soğuk başlatma saati : ünitenin durdurulmasından sonra tamamen soğuması için gereken süre

mevcut durum : ünitelerin çalışmada yada beklemede geçirdikleri süreler

saatler : hesaplamanın yapılacağı periyotlar

talep : sistemin karşılaması istenen enerji miktarı

4.2. Ünite Paylaşımı Probleminin Matematiksel Modeli

Bu çalışmada ünitelerin ve sistemin gerektirdiği kısıtları göz önünde bulundurarak yakıt maliyetinin en aza indirgenmesi amaçlanmıştır. Problemin çözümünün uygulanmasında dikkate alınan kısıtlar ve model aşağıdaki gibidir.

1) Yük dengesi kısıtlaması: Bir enerji üretim merkezinin her saat için ürettiği güç tüketicilerin talep ettiği güce eşit veya fazla olmalıdır.

$$\sum_{j=1}^{j} P_{G,J}\text{-}P_{D,J} \geq 0 \tag{4.1}$$

P_G jeneratörlerin toplam gücü, P_D sistemin çektiği toplam güç ve $j=1,2,...T$ saatleridir.

2)Yedek enerji kısıtlaması: Bir enerji üretim sisteminde üretimin talep edilen yükü karşılamasının yanında sistemin güvenilirliği açısından belirli bir orandaki enerjiyi istenildiğinde karşılaması istenir.

$$\sum_{j=1}^{j} P_{G,J}^{\max} \geq P_{D,J} + P_{R,J} \tag{4.2}$$

Burada $P_{G,J}^{\max}$ j. ünitenin verebileceği en yüksek gücü $P_{R,J}$, j. saat verilmesi gereken yedek gücü, $P_{D,J}$ ise j. Saatte talep edilmesi gereken gücü göstermektedir ve $j=1,2,...T$ saatleridir.

3) En düşük devrede devre dışı kalma süresi kısıtlaması: Ünitelerin yapılış özelliklerinden dolayı üretime başladıklarından sonra belirli bir süre boyunca çalışır durumda kalma zorunlulukları vardır. Aynı şekilde termal ve mekanik sınırlamalar nedeniyle ünite devre dışına alındığında belirli bir süre daha kapalı kalması gerekir.

$$\sum_{j=1}^{T} \left| T_{i,j}^{ON} - MUT_i \right| \geq 0 \tag{4.3}$$

$$\sum_{j=1}^{T} \left| T_{i,j}^{OFF} - MDT_i \right| \geq 0 \tag{4.4}$$

Burada *MUT,* ünitenin en az devrede kalma süresi, $T_{i,j}^{ON}$ ünitenin devrede kaldığı süresi. *MDT*, ünitenin en az devre dışı kalma süresi $T_{i,j}^{ON}$ ünitenin devre dışı kaldığı sürelerdir

4) Ünite çıkış limitleri kısıtlaması: Her ünitenin kendine has özelliklerinden ve fiziksel kısıtlamalardan dolayı verebileceği enerji miktarları belirli değerler arasında olmalıdır.

$$P_{min\ i,j} \leq P_{i,j} \leq P_{max\ i,j} \tag{4.5}$$

Bu denklemde $P_{min\ i,j}$ i. üniteden alınması gereken en düşük güç miktarı, $P_{i,j}$ i. üniteden talep edilen gücü, $P_{max\ i,j}$ ise ünitenin verebileceği en yüksek gücü göstermektedir ve $j=1,2,...T$ saatleridir

Gün içinde değişen enerji talebinin en ekonomik şekilde sağlanması için ünitelerin oluşturduğu maliyetler en aza indirilmelidir. Oluşacak masraflar ünitelerin üretim için hazır hale gelene kadar geçen sürede oluşan maliyetler, kapatma maliyetleri ve talep edilen yükü karşılamak için harcanan yakıtlardan oluşan maliyetlerdir.

Ünite paylaşımı probleminde oluşan maliyetler ve matematiksel gösterimleri aşağıda verilmiştir.

1) Yakıt maliyeti: Yakıt maliyeti üniteler çalışır durumda iken ve ürettikleri güce bağlı olan bir fonksiyondur.

$$FC = a_i + b_i \mathrm{P} + c_i \mathrm{P}^2 \tag{4.6}$$

Burada; *FC* ünitenin yakıt maliyeti, a_i, b_i, ve c_i yakıt maliyeti katsayıları, P ise üniteden talep edilen güç miktarıdır.

2) Başlatma maliyeti: Başlangıç maliyeti, üniteler boşta beklerken devreye alındığında oluşan maliyettir. Eğer ünitenin türbinleri ortam sıcaklığına kadar

soğuduktan sonra tekrar ısıtılıp devreye alınırsa oluşan maliyete soğuk başlatma adı verilir.

$$CSUC = \sigma_i + \delta_i \left[1 - e^{(-T_{i,j}^{OFF} / \tau_i)} \right] \tag{4.7}$$

Burada *CSUC* ünitenin soğuk başlatma maliyeti, σ_i , δ_i, τ_i başlatma katsayıları, T^{OFF} ise kapalı kalınan zamandır.

Ünitenin bekleme durumuna göre oluşan bir diğer başlatma maliyeti de sıcak başlatma maliyetidir.

$$HSUC = \rho T^{OFF} \tag{4.8}$$

Burada; ρ yatık konum maliyet sabiti ve T^{OFF} yatık konumda kalma süresidir.

3) Durdurma maliyeti: Üniteler çalışır durumda iken devre dışına alındığında oluşan maliyettir ve birim çalışanları haricinde fazladan işgücüne gerek olmadığından bu çalışmada sıfır kabul edilmiştir.

$$SDC=0 \tag{4.9}$$

SDC üniteyi kapatırken oluşan maliyettir.

Buna göre (4.5)-(4.7) eşitliklerinden toplam maliyet;

Toplam Maliyet = Yakıt Maliyeti + Başlangıç Maliyeti + Durdurma Maliyeti olur ve

$$\text{TM} = \sum_{j=1}^{T} \sum_{i=1}^{N} \left(a_i + b_i P_{ij} + c_i P_{ij}^2 \right) + \sum_{j=1}^{T} \sum_{i=1}^{N} \sigma_i + \delta_i \left[1 - e^{(-T_{i,j}^{OFF} / \tau_i)} \right] \tag{4.10}$$

Denklemiyle ifade edilir. Bu uygulamada amaç ünite paylaşımı problemindeki kısıtları da göz önünde bulundurarak 24 saatlik sürede toplam üretim maliyetini en aza indirmektir.

4.3. Problemin Çözümünde Kullanılan Genetik Algoritma

Genetik algoritmalarda çözülecek problemin değişkenleri sabit uzunlukta bir bit dizisi olarak ifade edilir. Bu dizideki bitler 1 ile 0'lardan oluşur ve kromozom olarak adlandırılır. İlk başta rastgele olarak oluşturulan bu kromozomlar bir araya gelerek popülasyonu yani problemin çözüm kümesini oluştururlar. Çözülmesi istenen amaç fonksiyon ise uygunluk fonksiyonu olarak sürece katılır. Rasgele yaratılan ve çözümleri ifade eden bu kromozomların uygunluk değerleri bulunur ve genetik algoritmanın temel operatörleri olan seçim, çaprazlama ve mutasyon işlemlerinden geçirilerek yeni popülasyonların oluşumunda kullanılırlar.

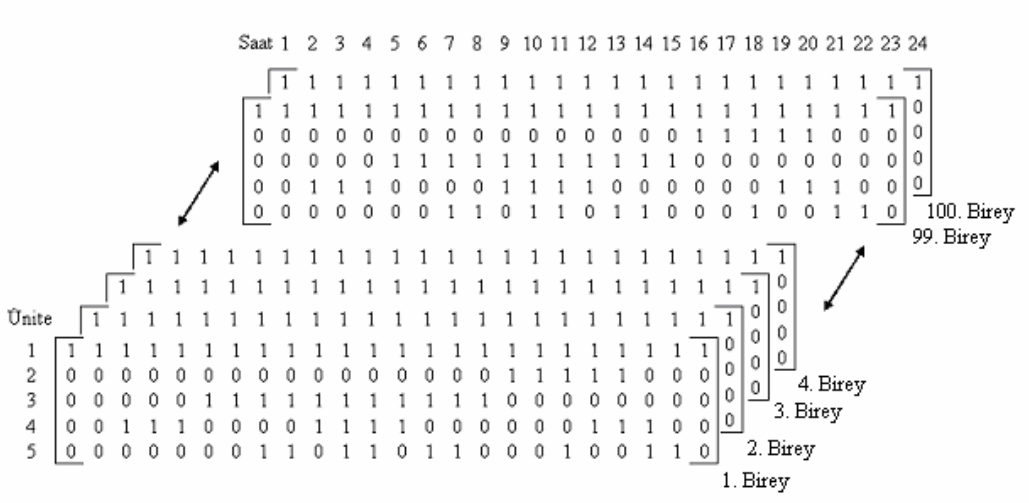
4.3.1. Başlangıç popülasyonu ve kromozom kodlaması

Kromozom yapısı genetik algoritmada kullanılan operatörlerin uygulanış şeklini belirler. Bu çalışmada basitliği ve genetik işlevlerde daha hızlı işlem yapmaya olanak sağlaması nedeniyle bit dizisi kodlaması kullanılmıştır. Süreç başlatılmadan önce rastgele olarak üretilen, 1 ve 0'lardan oluşan bireyler üretilir. Kuşak sayısı ilerledikçe kısıtlara uymayan bireyler yok olur ve yerlerine daha iyi çözümleri ifade nesiller gelir. Bir bireye ait bit dizisinin matris gösterimi şekil 4.1'de verilmiştir.

Saat/ Ünite	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
3	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
4	0	0	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	1	1	1	0	0
5	0	0	0	0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	1	0	0	1	1	0

Şekil 4.1: Uygulamada kullanılan kromozom gösterimi.

Bu gösterimde 5 adet ünitenin 24 saatlik periyotta hangi durumda olacağı ifade edilmiştir. Eğer bir ünite belirli bir saatte çalışır durumda ise matriste 1 değerini, beklemede ise 0 değerini alır. Örnek olarak popülasyonda 100 adet birey kullanıldığında oluşacak yapı şekil 4.2'deki gibi üç boyutlu bir matristir.



Şekil 4.2: Uygulamada kullanılan popülasyon gösterimi.

4.3.2. Uygunluk değeri

Bir nüfus oluşturulduktan sonraki ilk adım, popülasyondaki her üyenin uygunluk değerini hesaplamadır. Bu çalışmada amaç en düşük maliyetli üretim takvimi oluşturmak olduğundan, yüksek maliyetli bireylerin seçilme olasılıklarının azaltılması gerekir. Bu nedenle uygunluk fonksiyonu denklem 4.11'deki gibi seçilmiştir.

$$\text{Uygunluk}=\frac{1}{Maliyet+\varepsilon} \tag{4.11}$$

4.3.3. Seçim işlevi

Başlangıç popülasyonu oluşturulduktan sonra yeni nesillerin yaratılabilmesi için mevcut popülasyondan en iyi uygunluk değerine sahip bireyler arasından seçim yapılması gereklidir. Bu çalışmada kullanılan sıralı seçim metodunda, her bireyin uygunluk değeri toplam uygunluk değerine bölünerek, en kötü uygunluk değerine sahip olan kromozoma 1 değeri, sonraki kromozoma 2 değeri ve en iyi kromozoma da popülasyondaki kromozom sayısının değeri verilerek bütün bireylere seçilme şansı verilir.

Ünite tahsisi problemindeki amaç maliyeti, kısıtları da göz önünde bulundurarak en aza indirmektir. Eğer bireyler oluşturdukları takvimde kısıtları sağlamazsa maliyet değerine her uymadığı kısıt için ceza olarak 1500000 eklenerek seçilme şansı sıfıra yakın bir değere indirilir

En yüksek uygunluk değerine sahip bireyin iyi özellikleri bozulmadan bir sonraki popülasyona aktarılması için elitizm de uygulanmıştır.

4.3.4. Çaprazlama

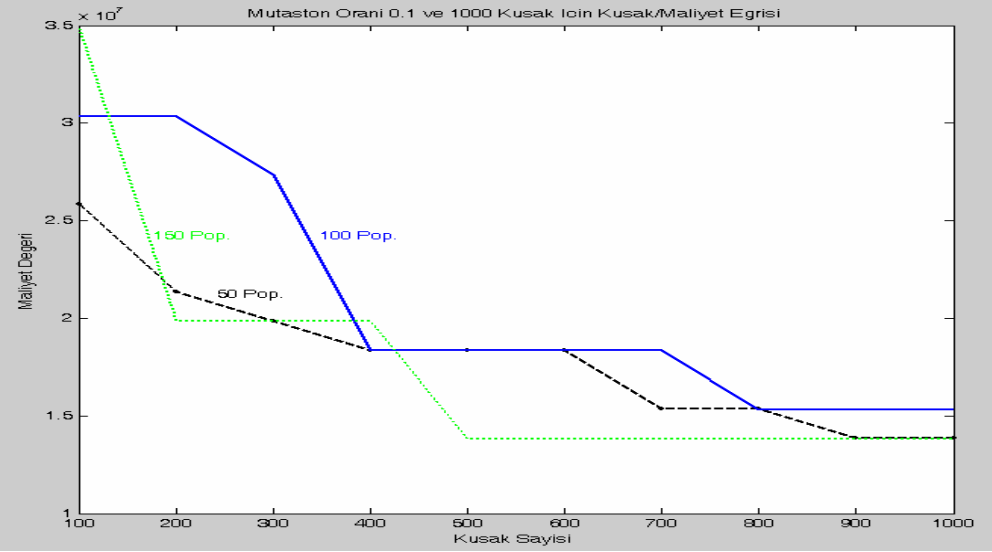
Çaprazlamadaki amaç popülasyon içerisindeki iyi özellikli bireylerden daha iyi çözümler bulan bireylerin yaratılmasıdır. Eşleşen iki bireyde, bu çaprazlama noktasının sağında kalan bölümlerin yeri değiştirilerek yeni bireyler elde edilmiş olur. Bu çalışmada, poülasyon içerisinden seçim işleviyle belirlenmiş olan ana baba çiftlerine, her çift için rastgele nokta belirlendikten sonra, tek noktalı çaprazlama uygulanmıştır.

4.3.5. Mutasyon

Mutasyonda amaç, nüfusu oluşturan bireylerin çeşitliliğini artırmaktır. İyi oluşturulamayan başlangıç nüfusu, çözüm kümesi içerisinde çeşitliliği sağlayamaz. Yerel minimumlara takılmayı engellemek ve çözüm uzayında farklı noktalara atlamayı sağlayan mutasyon oranı en uygun şekilde seçilmelidir.

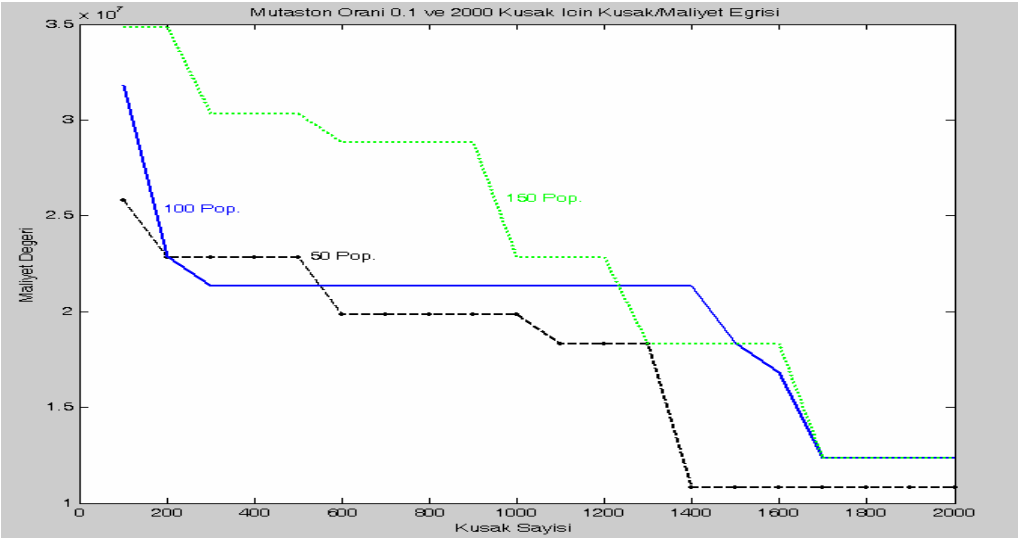
4.4. Mutasyon Oranının, Popülasyon Büyüklüğünün ve Kuşak Sayısının Etkisi

Genetik algoritma kullanılarak yapılan optimizasyon çalışmasında; mutasyon oranı, popülasyon büyüklüğü ve kuşak sayısının etkisi gösterilmiş ve sonuçları karşılaştırılmıştır. Yapılan çözümlerden elde edilen değerlere göre ünite tahsisi problemi için en uygun genetik algoritma parametrelerine karar verilmiştir.



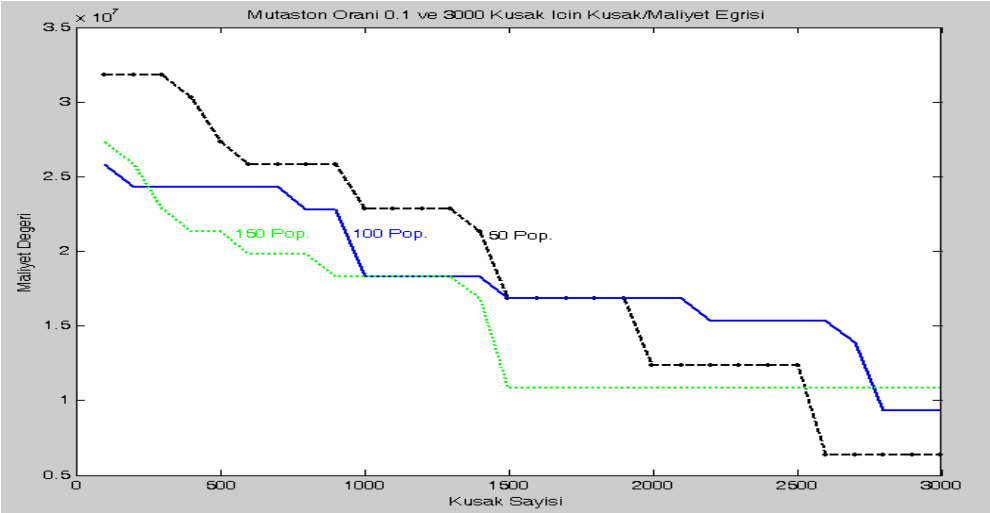
Şekil 4.3: Mutasyon oranı 0,1 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

Şekil 4.3'de, mutasyon oranı 0,1 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 1000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $13861379,73, popülasyon büyüklüğü 100 iken maliyet değeri $15327678,94, popülasyon büyüklüğü 150 iken maliyet değeri $13834950,66 bulunmuştur. Bu değerlere göre mutasyon oranı 0,1 ve 1000 kuşak ile bulunan çözümlerin tümünün kısıtlamaları sağlamadığı için ceza değerleri aldıkları ve gerçek bir takvim sağlamadığı görülmüştür.



Şekil 4.4: Mutasyon oranı 0,1 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

Şekil 4.4'de, mutasyon oranı 0,1 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 2000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $10820563,63, popülasyon büyüklüğü 100 iken maliyet değeri $12340065,39, popülasyon büyüklüğü 150 iken maliyet değeri $12328755,97 bulunmuştur. Bu değerlere göre mutasyon oranı 0,1 ve 2000 kuşak ile bulunan çözümlerin tümünün kısıtlamaları sağlamadığı için ceza değerleri aldıkları ve gerçek bir takvim sağlamadığı görülmüştür.

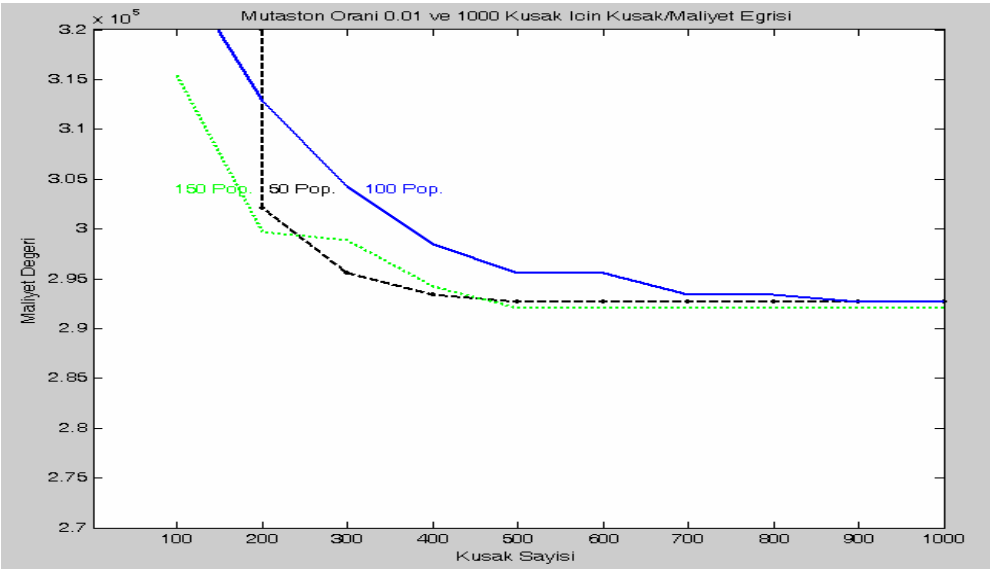


Şekil 4.5: Mutasyon oranı 0,1 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

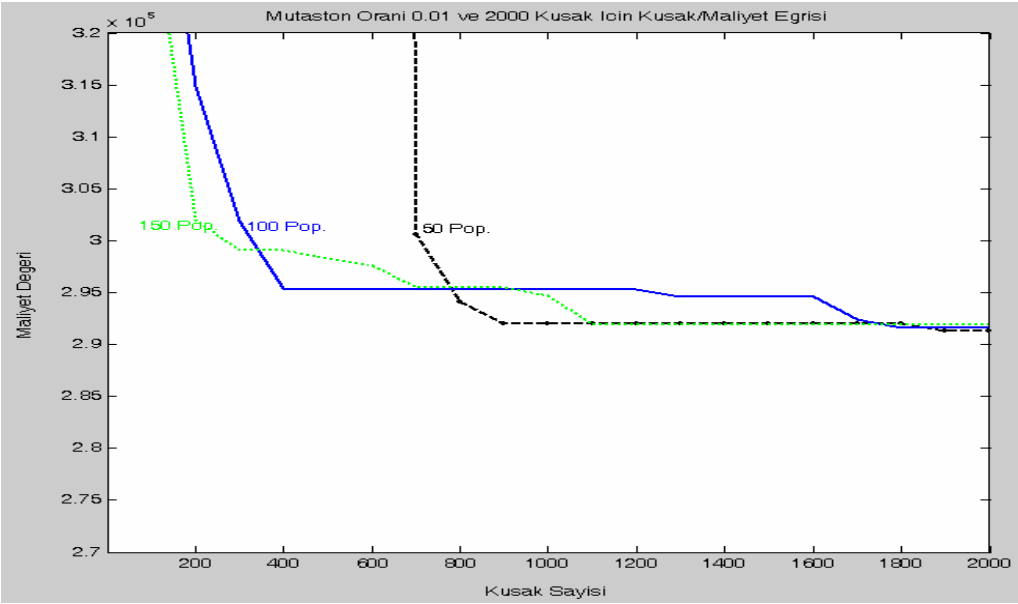
Şekil 4.5'de, mutasyon oranı 0,1 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 3000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $6331054,19, popülasyon büyüklüğü 100 iken maliyet değeri $9346257,66, popülasyon büyüklüğü 150 iken maliyet değeri $10841963,96 bulunmuştur. Bu değerlere göre mutasyon oranı 0,1 ve 3000 kuşak ile bulunan çözümlerin tümünün kısıtlamaları sağlamadığı için ceza değerleri aldıkları ve gerçek bir takvim sağlamadığı görülmüştür.

Şekil 4.6'da, mutasyon oranı 0,01 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 1000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $292691,04, popülasyon büyüklüğü 100 iken maliyet değeri $292665,91, popülasyon büyüklüğü 150 iken

maliyet değeri $292027,91 bulunmuştur. Bu değerlere göre mutasyon oranı 0,01 ve 1000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 150 iken bulunmuştur.



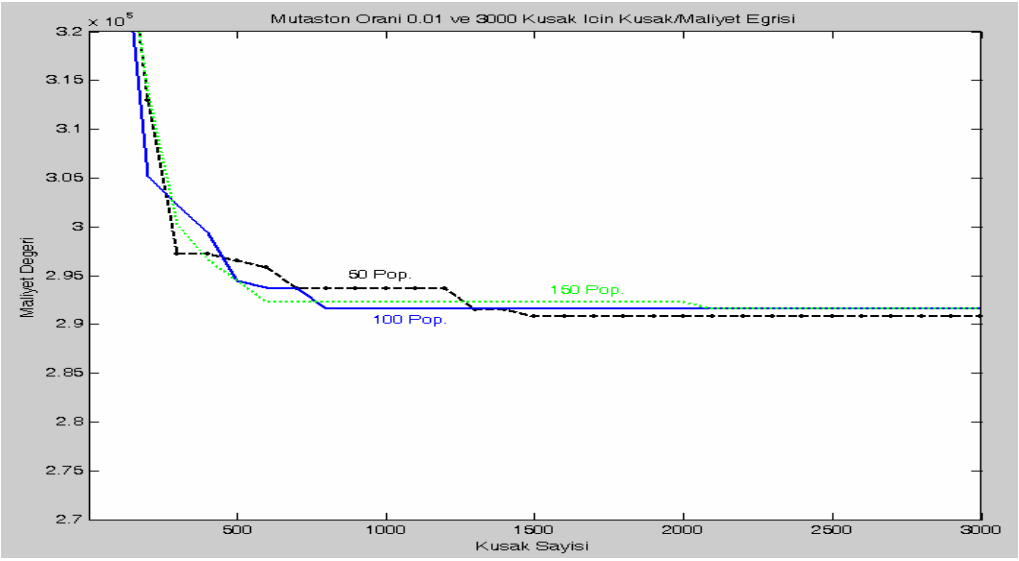
Şekil 4.6: Mutasyon oranı 0,01 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi



Şekil 4.7: Mutasyon oranı 0,01 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

Şekil 4.7'de, mutasyon oranı 0,01 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 2000 kuşak

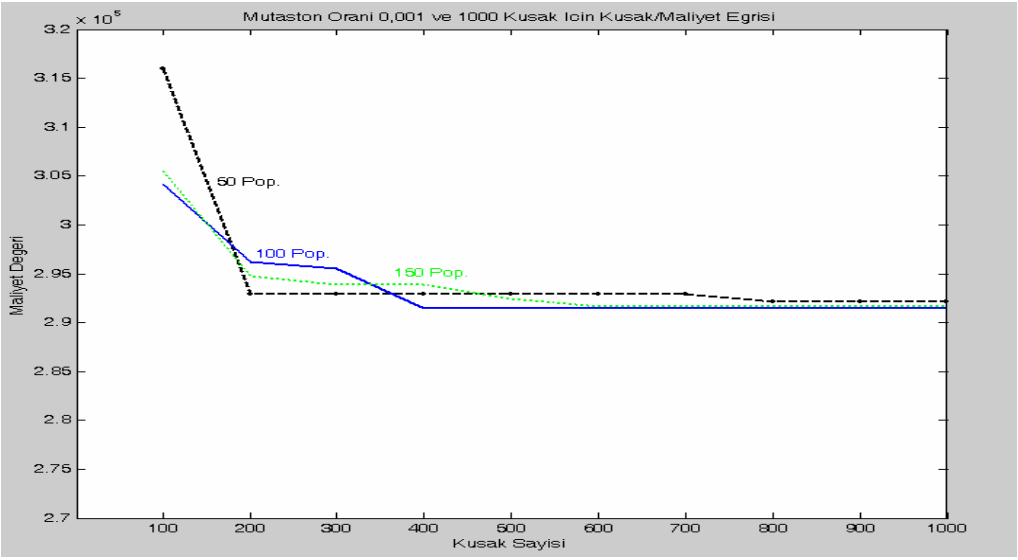
sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $291294,2, popülasyon büyüklüğü 100 iken maliyet değeri $291654,22, popülasyon büyüklüğü 150 iken maliyet değeri $291869,92 bulunmuştur. Bu değerlere göre mutasyon oranı 0,01 ve 2000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 50 iken bulunmuştur.



Şekil 4.8: Mutasyon oranı 0,01 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

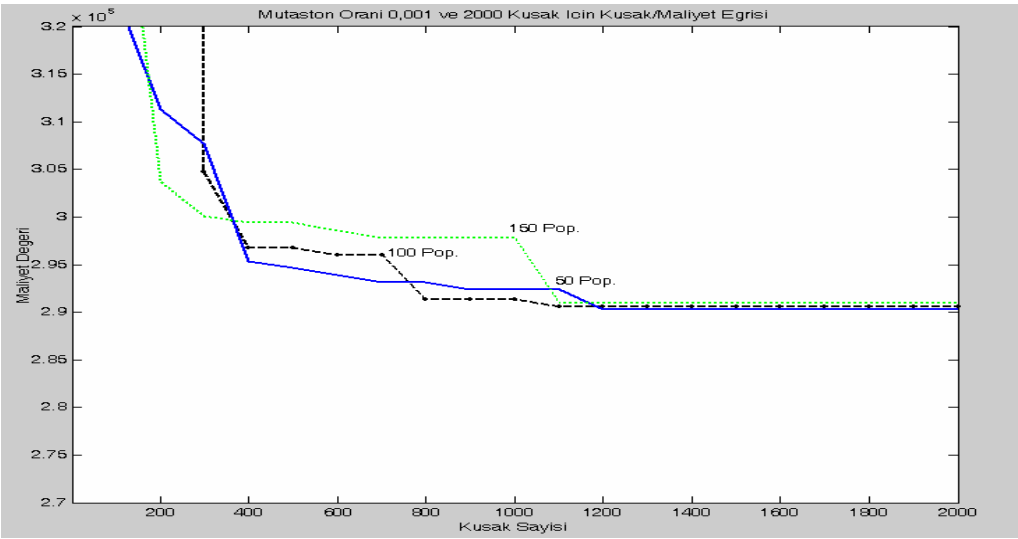
Şekil 4.8'de, mutasyon oranı 0,01 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 3000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $290795,05, popülasyon büyüklüğü 100 iken maliyet değeri $291583,72, popülasyon büyüklüğü 150 iken maliyet değeri $291623,08 bulunmuştur. Bu değerlere göre mutasyon oranı 0,01 ve 3000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 50 iken bulunmuştur.

Şekil 4.9'da, mutasyon oranı 0,001 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 1000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $292105,08, popülasyon büyüklüğü 100 iken maliyet değeri $291428,01, popülasyon büyüklüğü 150 iken maliyet değeri $291646,9 bulunmuştur. Bu değerlere göre mutasyon oranı 0,001 ve 1000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 100 iken bulunmuştur.

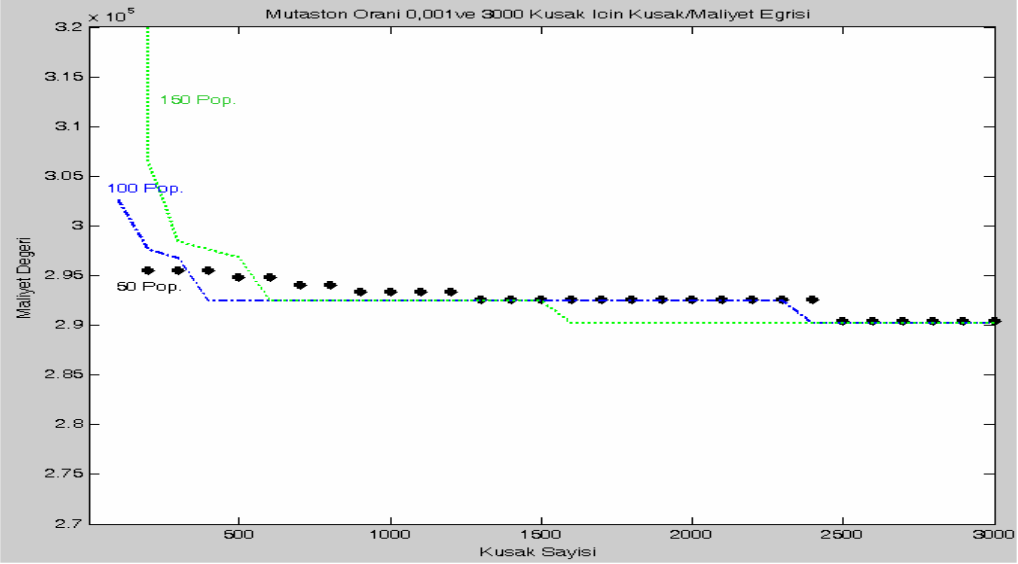


Şekil 4.9: Mutasyon oranı 0,001 ve 1000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

Şekil 4.10'da, mutasyon oranı 0,001 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 2000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $290585,63, popülasyon büyüklüğü 100 iken maliyet değeri $290251,37, popülasyon büyüklüğü 150 iken maliyet değeri $290920,51 bulunmuştur. Bu değerlere göre mutasyon oranı 0,001 ve 2000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 100 iken bulunmuştur.



Şekil 4.10: Mutasyon oranı 0,001 ve 2000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

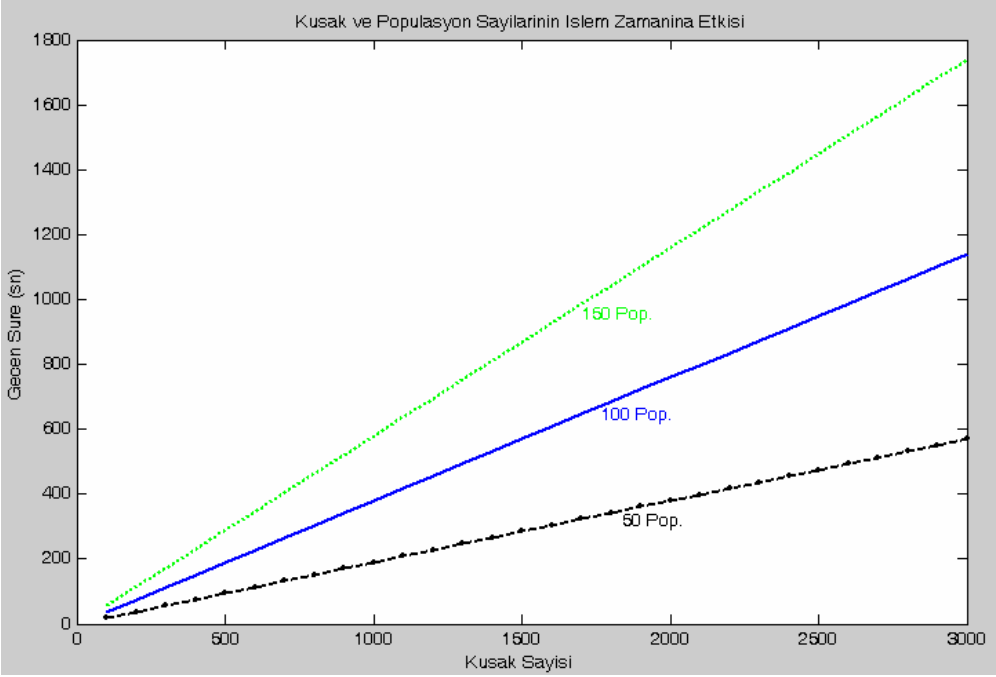


Şekil 4.11: Mutasyon oranı 0,001 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi

Şekil 4.11'de, mutasyon oranı 0,001 ve 3000 kuşak için değişik popülasyon büyüklüklerine göre maliyet değerlerinin değişimi görülmektedir. 3000 kuşak sonunda, popülasyon büyüklüğü 50 iken maliyet değeri $290375,93, popülasyon büyüklüğü 100 iken maliyet değeri $290251,37, popülasyon büyüklüğü 150 iken maliyet değeri $290251,37 bulunmuştur. Bu değerlere göre mutasyon oranı 0,001 ve 3000 kuşak ile bulunan en iyi çözüm popülasyon büyüklüğü 100 ve 150 iken bulunmuştur.

4.4.1. Kuşak ve popülasyon sayısının işlem zamanına etkisi

Genetik algoritmalarla problemin çözümü yapılırken işlem zamanı popülasyon büyüklüğü ve kuşak sayısıyla orantılı olarak artmaktadır. Kuşak sayısı arttıkça algoritma en iyi çözüme yakınsamaktadır, fakat çözümdeki değişmenin süreye oranı da oldukça azalmaktadır. Bir önceki bölümde gösterilen değerlere göre kuşak ve popülasyon sayısı fazla bir etki sağlamadığından daha fazla kuşak ve popülasyon üretilmesi gereksizdir. Şekil 4.12'de AMD 2600 işlemci, 1 GB bellek özelliklerine sahip bilgisayarla bulunan popülasyon ve kuşak sayılarına göre geçen süreler gösterilmiştir.



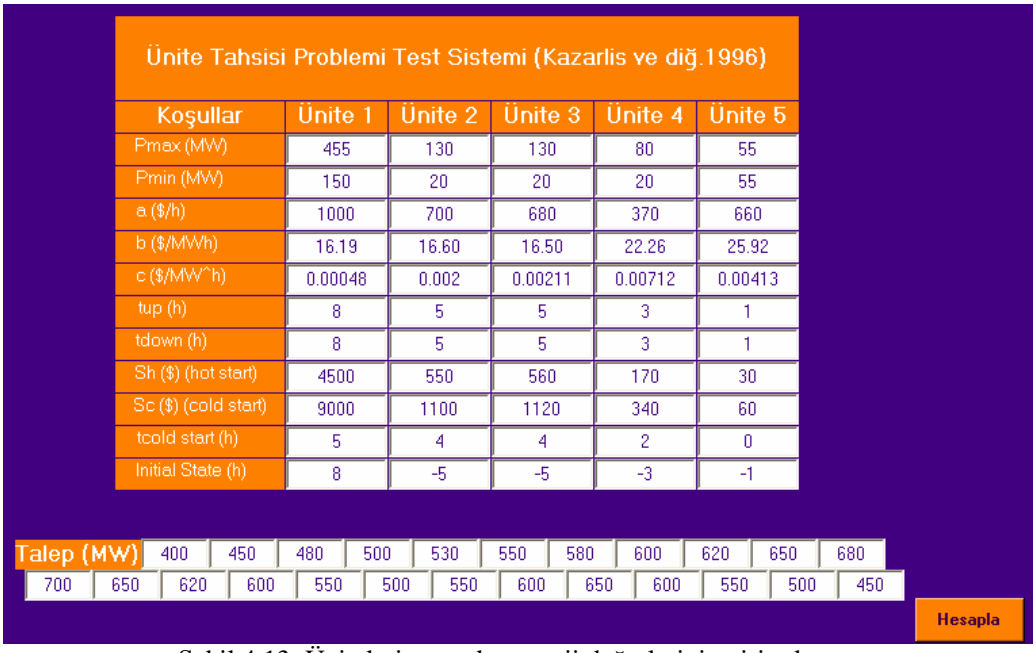
Şekil 4.12: Kuşak ve popülasyon sayısının işlem zamanına etkisi

4.5. Uygulama Programı

Uygulama için hazırlanan grafik arayüzdeki değerleri değiştirme imkanı olduğu gibi bu çalışmada referans alınan makaledeki değerler standart olarak giriş ekranında gösterilmiştir.

Şekil 4.13'de görüldüğü gibi oluşturulan tabloda sırasıyla 5 ünite için üretebilecekleri enerji aralıkları, üretim maliyetinin hesaplanmasında kullanılan katsayılar, çalışma kısıtları, başlatma maliyetleri, sıcak ve soğuk başlatma saatleri, mevcut durumları ile 24 saatlik enerji talepleri yer almaktadır.

Optimizasyonda kullanılacak değerler girildikten sonra "Hesapla" butonuna basıldığında tasarımı yapılan genetik algoritma optimizasyon işlemine başlar. Sonlandırma kriteri olan 2000 kuşak sayısına ulaşıldığında süreç sona erdirilir ve şekil 4.14'de görüldüğü gibi ünitelerin saat başına üretim maliyetleri ekran çıktısı olarak gösterilir.



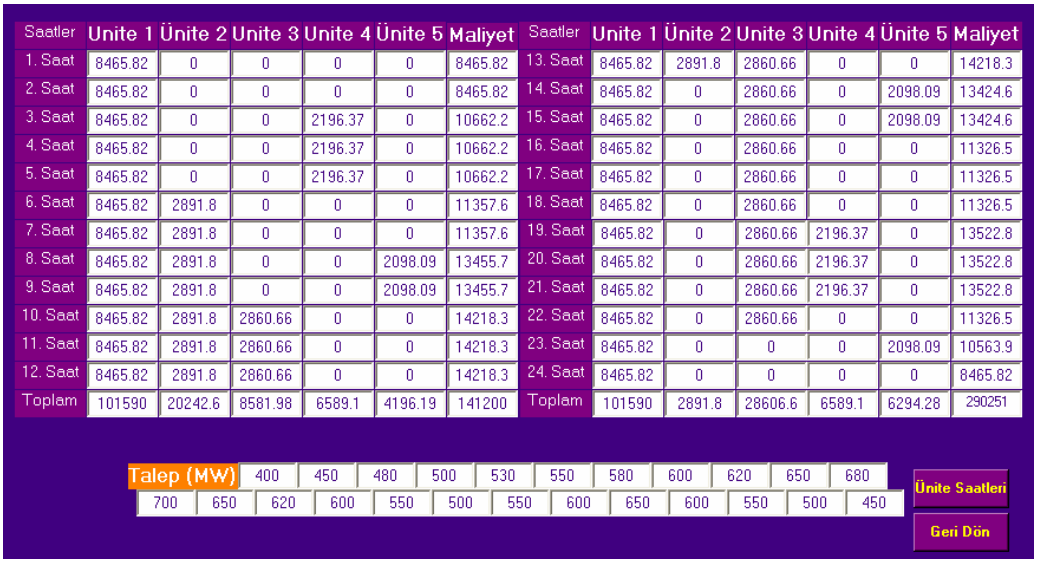
Ünite Tahsisi Problemi Test Sistemi (Kazarlis ve diğ.1996)

Koşullar	Ünite 1	Ünite 2	Ünite 3	Ünite 4	Ünite 5
Pmax (MW)	455	130	130	80	55
Pmin (MW)	150	20	20	20	55
a ($/h)	1000	700	680	370	660
b ($/MWh)	16.19	16.60	16.50	22.26	25.92
c ($/MW^h)	0.00048	0.002	0.00211	0.00712	0.00413
tup (h)	8	5	5	3	1
tdown (h)	8	5	5	3	1
Sh ($) (hot start)	4500	550	560	170	30
Sc ($) (cold start)	9000	1100	1120	340	60
tcold start (h)	5	4	4	2	0
Initial State (h)	8	-5	-5	-3	-1

Talep (MW)											
400	450	480	500	530	550	580	600	620	650	680	
700	650	620	600	550	500	550	600	650	600	550	500	450

Hesapla

Şekil 4.13: Ünitelerin ve talep enerji değerlerinin giriş ekranı.



Saatler	Unite 1	Ünite 2	Unite 3	Unite 4	Ünite 5	Maliyet	Saatler	Unite 1	Ünite 2	Unite 3	Unite 4	Ünite 5	Maliyet
1. Saat	8465.82	0	0	0	0	8465.82	13. Saat	8465.82	2891.8	2860.66	0	0	14218.3
2. Saat	8465.82	0	0	0	0	8465.82	14. Saat	8465.82	0	2860.66	0	2098.09	13424.6
3. Saat	8465.82	0	0	2196.37	0	10662.2	15. Saat	8465.82	0	2860.66	0	2098.09	13424.6
4. Saat	8465.82	0	0	2196.37	0	10662.2	16. Saat	8465.82	0	2860.66	0	0	11326.5
5. Saat	8465.82	0	0	2196.37	0	10662.2	17. Saat	8465.82	0	2860.66	0	0	11326.5
6. Saat	8465.82	2891.8	0	0	0	11357.6	18. Saat	8465.82	0	2860.66	0	0	11326.5
7. Saat	8465.82	2891.8	0	0	0	11357.6	19. Saat	8465.82	0	2860.66	2196.37	0	13522.8
8. Saat	8465.82	2891.8	0	0	2098.09	13455.7	20. Saat	8465.82	0	2860.66	2196.37	0	13522.8
9. Saat	8465.82	2891.8	0	0	2098.09	13455.7	21. Saat	8465.82	0	2860.66	2196.37	0	13522.8
10. Saat	8465.82	2891.8	2860.66	0	0	14218.3	22. Saat	8465.82	0	2860.66	0	0	11326.5
11. Saat	8465.82	2891.8	2860.66	0	0	14218.3	23. Saat	8465.82	0	0	0	2098.09	10563.9
12. Saat	8465.82	2891.8	2860.66	0	0	14218.3	24. Saat	8465.82	0	0	0	0	8465.82
Toplam	101590	20242.6	8581.98	6589.1	4196.19	141200	Toplam	101590	2891.8	28606.6	6589.1	6294.28	290251

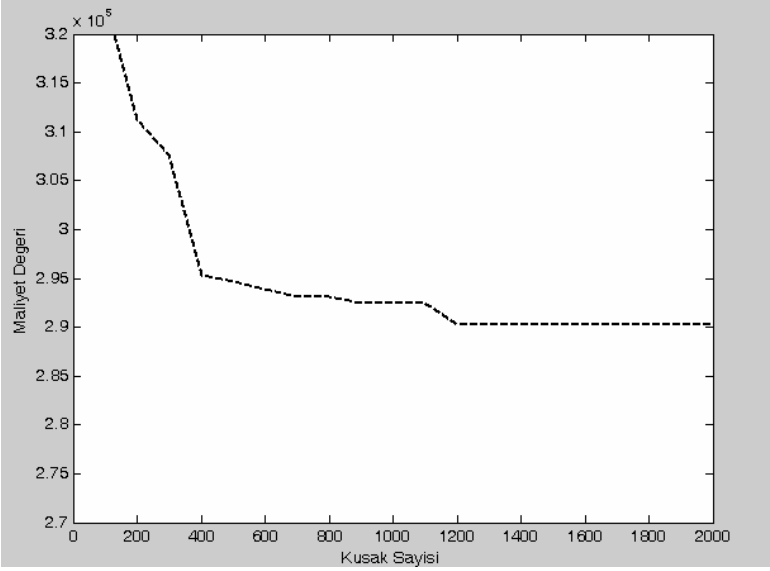
Talep (MW)											
400	450	480	500	530	550	580	600	620	650	680	
700	650	620	600	550	500	550	600	650	600	550	500	450

Ünite Saatleri

Geri Dön

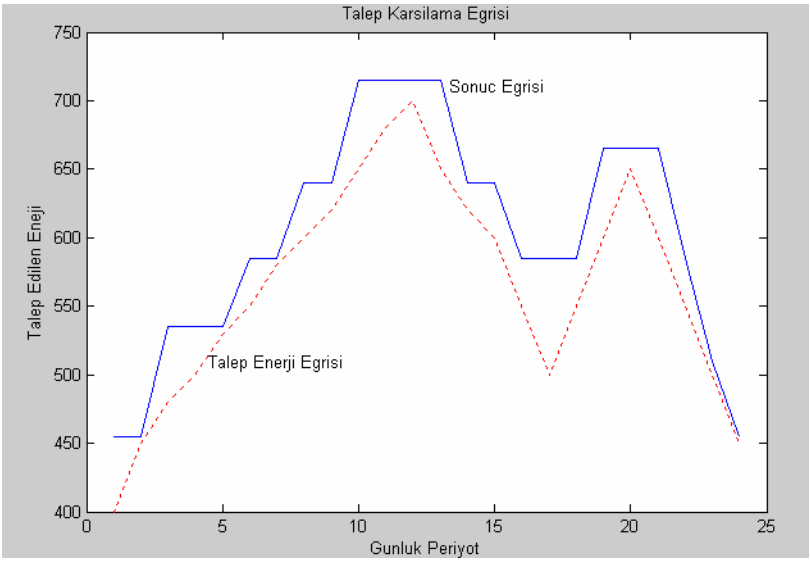
Şekil 4.14: Hesaplama sonucu bulunan maliyet değerleri.

Hesaplama yapıldığı süreç içerisinde kuşak sayısı arttıkça kısıtları sağlamayan ve yüksek maliyet oluşturan bireyler hızla popülasyondan çıkarılarak yerlerine daha uygun bireyler getirilir. Şekil 4.15'deki 100 adet bireyden oluşan popülasyonla, kuşak sayısı ve maliyet değeri eğrisinden görüleceği üzere nesil yenilendikçe minimuma doğru hızlı bir yaklaşma vardır.



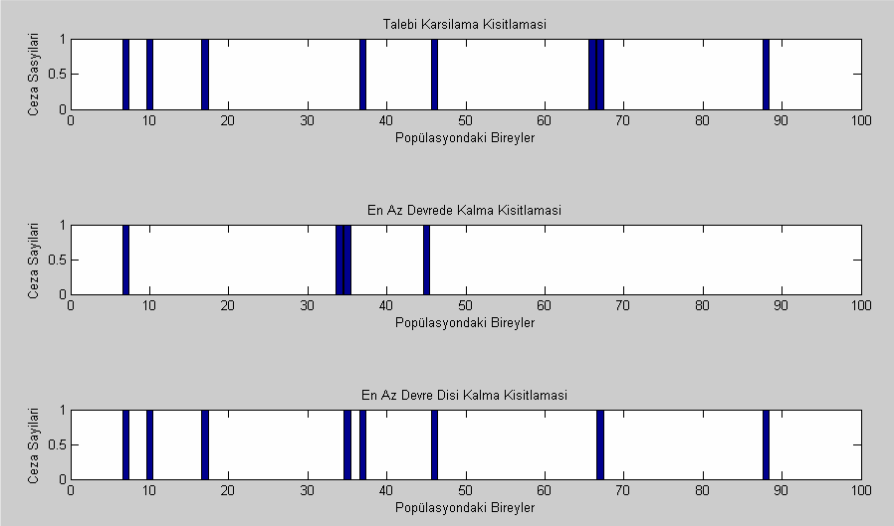
Şekil 4.15: Kuşak sayısı ilerledikçe bulunan maliyet.

Şekil 4.16'daki grafikte kesikli çizgilerle ifade edilen değerler 24 saatlik periyot boyunca kullanıcı tarafından girilen talep eğrisi, sürekli olarak çizilen eğri ise hesaplama sonucu bulunan üretim eğrisidir.



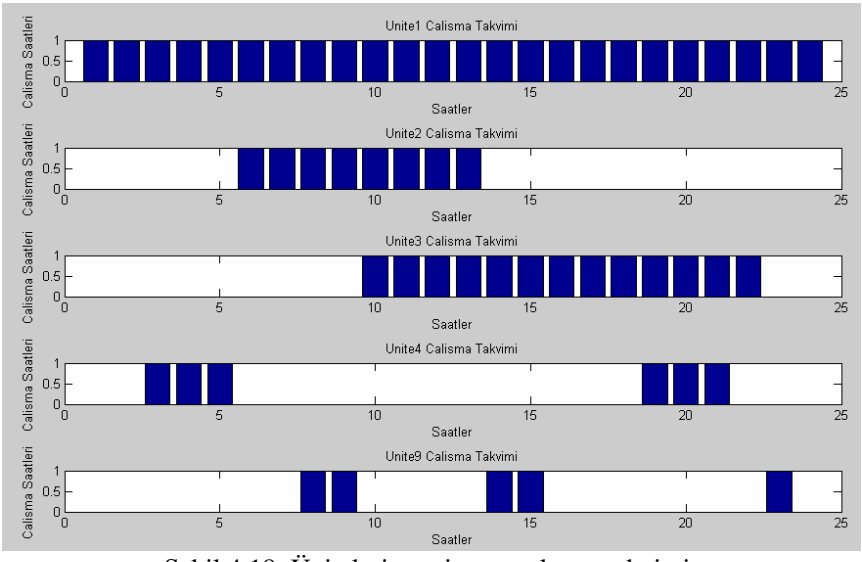
Şekil 4.16: Talebe göre bulunan üretim değerleri.

Çözüm için kullanılan 100 adet bireyden önceki bölümlerde açıklanmış olan ünite kısıtlamalarını sağlamayanları da mevcut popülasyon içerisinde yer alabilirler. Şekil 4.17'de popülasyon içerisinde kısıtları sağlamayan bireyler gösterilmiştir.



Şekil 4.17: Popülasyonda kısıtları sağlamayan bireyler.

Hesaplama işlemi sonlandırıldığında ortaya çıkan sonuç ekranında ünitelerin üretim maliyetlerinin yanında 24 saatlik periyotta hangi saatlerde çalışır durumda olduklarını gösteren şekil 4.18'deki grafik "Ünite Saatleri" butonuna basıldığında ekrana gelir.



Şekil 4.18: Ünitelerin optimum çalışma takvimi.

4.6. Uygulama Sonucu

Yapılan çalışmada bulunan sonuçlar aşağıdaki gibidir.

Bulunan en iyi toplam maliyet değeri $290251,37 olmuştur. Bulunan sonuç referans alınan makaledeki referans değerine yakındır.

Yaklaşık 2000 kuşaktan sonra çözüm en iyi sonuca ulaşmaktadır. Genetik algoritma çözüm uzayında rastgele dolaşarak çözüm aradığından, kuşak sayısı arttıkça çözümü bulma olasılığı da artmaktadır.Kullanılan ceza işlevi ile kısıtları sağlamayan bireylerin hızla popülasyonun dışına itilmesi sağlandı bu sayede optimum çözüm uzayı daraltılarak kuşak sayısı azaltılmış oldu.

Yapılan denemelerde kuşak sayısı ve popülasyon sayısı artırıldığında işlem zamanı çok fazla uzamıştır. Buna rağmen bulunan sonuç gene $290251,37 olmuştur.

Ünite tahsisi probleminin çözümü için en uygun mutasyon oranı 0,001 ve popülasyon büyüklüğü 100 adet birey olarak bulunmuştur. Mutasyon oranının artırılması bireylerin ifade ettiği çözümleri çok sık rastgele noktalara attığından çözüme yakınsama sağlanamamıştır. Popülasyon büyüklüğü ise azaltıldığında optimum çözüm bulunamamış, artırıldığında da daha iyi bir sonuç bulunamamıştır.

Uygulama için tasarlanan program sayesinde her onuncu kuşakta üretim maliyeti, talebi karşılama oranı, popülasyonda kısıtları sağlamayan bireyler ve ünitelerin saatlik çalışma takvimleri gösterildi. Bu sayede sürecin iyiye gidişi anlık olarak takip edildi.

5. SONUÇLAR ve ÖNERİLER

Elektrik enerjisi, kullanım kolaylığından dolayı, günümüz modern toplumlarının vazgeçilmez bir ihtiyacıdır. Elektrik enerjisinin üretimi için kullanılan enerji kaynakları temel olarak; nükleer enerji, yenilenebilir enerji kaynakları (güneş, hidrolik, rüzgar, biyokütle, jeotermal, vb.) ve fosil yakıtlardır. Diğer yenilenebilir enerji kaynaklarının işletme maliyetleri ucuz olmasına rağmen, kurulum maliyetleri fosil yakıtlara oranla daha pahalıdır. Fosil yakıtlar, önümüzdeki on yıllarda da dünya enerji tüketiminde belirleyici enerji hammaddesi olacaktır.

Elektrik enerjisinin depolanması için gerekli olan hidroelektrik pompa ve pil gibi teknolojiler hiç etkin değildir. Bu sebeple, elektrik talebi ve arzının her saniye dengelenmesi gerekmektedir. Üretim birimlerinden talep edilen enerjinin toplam maliyetini en düşük seviyede tutmak için, belli bir zaman aralığında ünitelerin çalıştırılıp çalıştırılmayacağını belirleyen planlama çalışmasına ihtiyaç vardır.

Kısa dönem veya günlük enerji sistemi planlamasında kullanılan ünite tahsis problemi kısıtlamalı bir optimizasyon problemidir. Bu problemde, problemin giriş değerleri; talep edilen enerji, rezerv olarak bulundurulması gereken yedek enerji, ünitelerin karakteristik özellikleri olan ortalama yakıt maliyetleri, minimum devrede kalma ve minimum devre dışı kalma süreleri, başlatma maliyetleri, kapatma maliyetleridir. Problemin çıkışında ise, hangi ünitelerin hangi saat diliminde çalışacağını ve çalıştırılacak olanlardan ne kadar enerji alınacağını gösteren bir plan bulunur. Çok sayıda kısıtın olması, karmaşıklığı ve analitik çözüm yolunun olmaması gibi nedenlerle; çözümü zor, doğrusal olmayan, geniş ölçekli ve kombinasyonel bir problemdir.

Ünite tahsis probleminin çözüm uzayının çok büyük olması ve analitik çözümünün olmaması nedeniyle çözüm uzayında rastlantısal arama yöntemleri daha başarılı olur. Genetik algoritmalar da rastlantısal arama yöntemlerinden en çok bilinen ve

kullanılanıdır. Genetik algoritmalar arama işlemine bir nokta grubundan başlayarak yerel optimuma takılmadan çözüm uzayında rastlantısal olarak arama yapabilmeleri sayesinde bu tip kısıtlamalı problemlerin çözümünde başarısını kanıtlamıştır.

Bu tez çalışmasında, ünite tahsis probleminin çözümü için genetik algoritmalar kullanılmıştır. Genetik algoritmalarda kromozom yapısı, kullanılan operatörlerin uygulanış şeklini belirler. Bu çalışmada basitliği ve genetik işlevlerde daha hızlı işlem yapmaya olanak sağlaması nedeniyle bit dizisi kodlaması kullanılmıştır. Bu gösterimde 5 adet ünitenin 24 saatlik periyotta hangi durumda olacağı ifade edilmiştir. Eğer bir ünite belirli bir saatte çalışır durumda ise matriste 1 değerini, beklemede ise 0 değerini alır.

Çalışmada, oluşturulan bir tablo ile, sırasıyla 5 ünite için üretebilecekleri enerji aralıkları, üretim maliyetinin hesaplanmasında kullanılan katsayılar, çalışma kısıtları, başlatma maliyetleri, sıcak ve soğuk başlatma saatleri, mevcut durumları ile 24 saatlik enerji talepleri geliştirdiğimiz optimizasyon programına giriş olarak verilmiştir.

Optimizasyonda, 2000 kuşak sayısı, 0,001 mutasyon oranı, 100 nüfus büyüklüğü kullanılmıştır. Optimizasyon işlemi sona erdiğinde; ünitelerin üretim maliyetlerinin yanı sıra, kuşak sayısına göre toplam maliyet, her kuşak için ünitelerin sağlamadıkları kısıt sayıları ve ünitelerin 24 saatlik periyotta hangi saatlerde çalışır durumda olduklarını gösteren grafik çıktılar elde edilmektedir.

5.1. Sonuçlar

Bulunan en iyi toplam maliyet değeri $290251,37 olmuştur. Bulunan sonuç referans alınan makaledeki referans değerine yakındır.

Yaklaşık 2000 kuşaktan sonra çözüm en iyi sonuca ulaşmaktadır.Genetik algoritma çözüm uzayında rastgele dolaşarak çözüm aradığından, kuşak sayısı arttıkça çözümü bulma olasılığı da artmaktadır.Kullanılan ceza işlevi ile kısıtları sağlamayan

bireylerin hızla popülasyonun dışına itilmesi sağlandı bu sayede optimum çözüm uzayı daraltılarak kuşak sayısı azaltılmış oldu.

Yapılan denemelerde kuşak sayısı ve popülasyon sayısı artırıldığında işlem zamanı çok fazla uzamıştır. Buna rağmen bulunan sonuç gene $290251,37 olmuştur.

Ünite tahsisi probleminin çözümü için en uygun mutasyon oranı 0,001 ve popülasyon büyüklüğü 100 adet birey olarak bulunmuştur. Mutasyon oranının artırılması bireylerin ifade ettiği çözümleri çok sık rastgele noktalara attığından çözüme yakınsama sağlanamamıştır. Popülasyon büyüklüğü ise azaltıldığında optimum çözüm bulunamamış, artırıldığında da daha iyi bir sonuç bulunamamıştır.

Uygulama için tasarlanan program sayesinde her onuncu kuşakta üretim maliyeti, talebi karşılama oranı, popülasyonda kısıtları sağlamayan bireyler ve ünitelerin saatlik çalışma takvimleri gösterildi. Bu sayede sürecin iyiye gidişi anlık olarak takip edildi.

5.2. Öneriler

Bu çalışmada ünite tahsis probleminin çözümü genetik algoritmalarla yapılmıştır. Çözüm bulunurken ekonomik yük dağılımı göz önünde bulundurulmadığından üretilen toplam güç miktarı talep edilen yükten fazla olmuştur. Planlama yapılırken bulanık mantık veya tam sayı kodlamalı genetik algoritma kullanılarak ekonomik yük dağılımını yapıldığında toplam maliyette azalma sağlanabilir.

Çözüme daha kısa sürede ulaşmak için başlangıç popülasyonunu rastgele üretmek yerine sezgisel yöntemler kullanılarak bulunması kuşak sayısında önemli azalmalar sağlayarak aynı sonuçlara ulaşılmayı sağlar

İşlem zamanını azaltmak için kullanılabilecek bir diğer yol olarak da elitizm işlevinde her kuşaktaki en iyi bireyin bütün genlerini almak yerine belli kuşaklar

boyunca en iyi bireylerin ortak genlerini almak kullanılabilir. Bu sayede en uygun çözüme katkısı olmayan genler devre dışı bırakılmış olur.

KAYNAKLAR

[1] Pamir, N., 2006, *Enerji politikaları ve küresel gelişmeler* [online], Avrasya Stratejik Araştırma Merkezi, www.asam.org.tr/tr/yazidosyagoster.asp?ID=11 **(Ziyaret tarihi: 20 Şubat 2007).**

[2] Akçöllü, F.Y., "Elektrik sektöründe rekabet ve regülasyon", Rekabet Kurumu Uzmanlık Tezi, ***Rekabet Kurumu***, Ankara, 1-10, (2003).

[3] BP, 2005, *BP Statistical review of world energy full report* [online], www.bp.com/liveassets/bp_internet/globalbp/globalbp_uk_english/publications/ energy_reviews_2005/STAGING/local_assets/downloads/pdf/statistical_review_of_ world_energy_full_report_2005.pdf **(Ziyaret tarihi: 20 Şubat 2007).**

[4] Gökçay, E., "Short-term thermal unit commitment by dynamic programming approach", Yüksek Lisans Tezi, ***Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 5-36, (1991).

[5] Takriti, S., Birge, J.R., "Using integer programming to Lagrangian-based unit commitment solutions", ***IEEE Transactions on Power Systems***, 15, 151–156, (2000).

[6] Damousis I.G., Bakirtzis,A.G., Dokopoulos, P.S., "A solution to the unit-commitment problem using integer-coded genetic algorithm", ***IEEE Transactions on Power Systems***, vol. 9, no. 2, 1165-1171, (2004).

[7] Dillon, T.S., "Integer programming approach to the problem of optimal unit commitment with probabelistic reserve determination", ***IEEE Transactions on Power Apparatus and Systems***, 2154–2164, (1978).

[8] Yamin, H.Y., "Review on methods of generation scheduling in electric power systems"***, Electric Power Systems Research***, 69, 227–248, (2004).

[9] Orero, S.O., Irving, M.R., "A combination of the genetic algorithm and Lagrangian relaxation decomposition techniques for the generation unit commitment problem", ***Electric Power Systems Research,*** 43, 149-156, (1997).

[10] Virmani, S., Imhof, K., Mukherjee, S., "Implementation of a Lagrangian relaxation based unit commitment problem", ***IEEE Transactions on Power Systems***, vol. 4, no. 4, 1373–1379, (1989) .

[11] Cheng, C.P., C.W., Liu, C.C., "Unit commitment by Lagrangian relaxation and genetic algorithms", ***IEEE Transactions on Power Systems,*** vol. 15, no. 2, 707–714, (2000).

[12] Mantawy, A.H., Abdel-Magid, S.Z. Selim, "Unit commitment by tabu search", ***IEEE Proceedings on Generation, Transmission and Distribution,*** 145, 56–64, (1998).

[13] Mantawy, A.H., Abdel-Magid, Y.L., Selim, S.Z., "A simulated annealing algorithm for unit commitment", ***IEEE Transactions on Power Systems,*** vol. 13, no. 1, 197–204, (1998).

[14] Rudolf, A., Bayrleithner, R., "A genetic algorithm for solving the unit commitment problem of a hydro-thermal power system", ***IEEE Transactions on Power Systems,*** vol. 14, no. 4, 1460–1468, (1999).

[15] Dudek, G., "Unit commitment by genetic algorithm with specialized search operators", ***Electric Power Systems Research,*** 72, 299–308, (2004).

[16] Yamin, H.Y., Shahidehpour, S.M., "Unit commitment using a hybrid model between Lagrangian relaxation and genetic algorithm in competitive electricity markets", ***Electric Power Systems Research***, 68, 83-92, (2004).

[17] Cai X.Q., Lo, K.M., "Unit commitment by a genetic algortihm nonlinear analysis", ***Theory Methods & App1ications,*** 30, 4289-4299, (1997).

[18] 0rero, S.O., Irving, M,R, "Large scale unit commitment using a hybrid genetic algorithm", **ElectricalPower & Energy Systems,** vol. 19, no. 1, 45-55, (1997).

[19] Mantawy, A.H., Abdel-Magid, Y.L., Selim, S.Z., "A new genetic-based tabu search algorithm for unit commitment problem, ***Electric Power Systems Research,*** 49, 71–78, (1999).

[20] Damousis I.G., Bakirtzis,A.G., Dokopoulos, P.S., "A solution to the unit-commitment problem using integer-coded genetic algorithm", ***IEEE Transactions on Power Systems***, vol. 9, no. 2, 1165-1171, (2004).

[21] Arroyo, J.S., Conejo, A.J., "A Parallel Repair Genetic Algorithm to Solve the Unit Commitment Problem", ***IEEE Transactions on Power Systems,*** vol. 17, no. 4, 1216-1224 , (2002).

[22] Yang, H.T., Yang P.C, Huang, C.L., "A Parallel Genetic Algorithm Approach to Solving the Unit Commitment Problem: Implementation on the Transputer Networks", ***IEEE Transactions on Power Systems,*** vol. 12, no. 2, 661-668, (1997).

[23] Swarup, K.S., Yamashiro, S., "Unit Commitment Solution Methodology Using Genetic Algorithm", ***IEEE Transactions on Power Systems ,*** vol. 17, no. 2, 87-91, (2002).

[24] Yiying, Z., Feng, W., Xiaomin, B., "Unit Commitment Problems Using GA", ***PowerCon 2002 İnternational Conference on Power System Technology,*** 1, 625-631, (2002).

[25] Swarup, K.S., Yamashiro, S., "A Genetic Algorithm Approach to Generator Unit Commitment", ***Electrical Power and Energy Systems,*** 25, 679–687, (2003).

[26] Yang, P.C., Yang, H.T., Huang, C.L., "A genetic algorithm for generator scheduling in power systems", ***Electrical Power and Energy Systems,*** 18, 19–26, (1996).

[27] Kazarlis, S.A., Bakirtzis, A.G., Petridis, V., "A genetic algorithm solution to the unit commitment problem", ***IEEE Transactions on Power Systems,*** vol. 11, no. 1, 83–92, (1996).

[28] Valenzuela J., Smith, A.E., "A seeded memetic algorithm for large unit commitment problems", ***Journal of Heuristics***, vol. 8, no.2 , 173–195, (2002).

[29] Yang, P.C., Yang, H.T., Huang, C.L., "A parallel genetic algorithm approach for solving the unit commitment problem: implementation on the transputer Networks", ***IEEE Transactions on Power Systems,*** 12, 661–669, (1997).

[30] Enerji ve Tabii Kaynaklar Bakanlığı, 2003, *Enerji ve tabii kaynaklar kamu araştırma programı* [online], www.tubitak.gov.tr/tubitak_content_files/ARDEB/kamag/Turkiye_Ulusal_Enerji_ve_Tabii_Kaynaklar_Arastirma_Programi.pdf **(Ziyaret tarihi: 20 Şubat 2007).**

[31] Sapmaz, M.E., "Generatör birim katkı sorununun genetik algoritmalar ile çözülmesi", Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, 4-8, (2004).

[32] Borenstein, S., Bushnell, J., "Electricity Restructuring: Deregulation or Reregulation ?", ***Regulation***, 2, 46-52, (2000).

[33] Eğri, R., "Optimized fuzzy approach to unit commitment problem in power system", Yüksek Lisans Tezi, ***Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 6-25, (1999).

[34] Aygen, Z.E, "A new approach to electrical power system optimization by using genetic algorithm", Doktora Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, 5-17, (2002).

[35] Snyder, W.L., Powell, H.D., Rayburn, J.C., "Dynamic programming approach to unit commitment", ***IEEE Transactions on Power Systems,*** vol. 2, no. 2, 339–350, (1987).

[36] Cohen , A.I., Yoshimura, M., "A branch-and-bound algorithm for unit commitment"***, IEEE Transactions on Power Apparatus and Systems,*** 102, 44–451, (1983).

[37] Chen , C.L., Wang, S.C., "Branch-and-bound scheduling for thermal generating units", ***IEEE Transactions on Energy Conversion,*** vol. 8, no. 2, 184–189, (1993).

[38] Geoffrion, A.M., "Generalized benders decomposition", ***Journal of Optimization Theory and Applications***, no. 4, 237–260, (1972).

[39] Işık, Y., **"**Genetik algoritma tabanlı bulanık kontrolün uçuş kontrol sistem tasarımına uygulanması", Doktora Tezi, ***Anadolu Üniversitesi Fen Bilimleri Enstitüsü,*** Eskişehir, 29-42, (2006).

[40] Tosun, E., " Frezeleme işlemlerinde genetik algoritma yaklaşımı ile kesme koşullarının optimizasyonu", Yüksek Lisans Tezi, ***Selçuk Üniversitesi Fen Bilimleri Enstitüsü,*** Konya, 25-42, (2006).

[41] Ceran, P., "Esnek akış tipi çizelgeleme problemlerinin veri madenciliği ve genetik algoritma kullanılarak çözülmesi", Yüksek Lisans Tezi, ***Selçuk Üniversitesi Fen Bilimleri Enstitüsü,*** Konya, 13-24, (2004).

[42] Biroğul, S., "Genetik algoritma yaklaşımıyla atölye çizelgeleme", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 28-51, (2005).

[43] Bolat, B., "Asansör kontrol sistemlerinin genetik algoritma", Yüksek Lisans Tezi, ***Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü,*** İstanbul, 23-27 (2006).

[44] Akoğlu, K., "Konteyner limanının depolama sahasının genetik algoritma ile optimizasyonu", Yüksek Lisans Tezi, ***Celal Bayar Üniversitesi Fen Bilimleri Enstitüsü,*** Manisa, 12-40, (2006).

[45] Akyol,. A.P., "Doğrusal olmayan ekonometrik modellerin genetik algoritma yaklaşımı ile parametre tahmini", Yüksek Lisans Tezi, ***Gazi Üniversitesi Sosyal Bilimler Enstitüsü,*** Ankara, 4-30, (2006).

[46] Kert, M., "Gerçek görüntüden elde edilen koordinatlarla robot kol hareket optimizasyonu", Yüksek Lisans Tezi, ***Mustafa Kemal Üniversitesi Fen Bilimleri Enstitüsü,*** Antakya, 33-45, (2006).

[47] Spears, W.M., "Crossover or Mutation?", ***Proceedings of the Second Founfations of Genetic Algorithms Workshop, Morgan Kaufmann Publishers***, San Mateo, California, 221-237, (1992).

EK-A: PROGRAM LİSTELERİ

Bu ekte, kısa dönem elektrik enerjisi optimum ünite tahsisi için, MATLAB'da yazılmış genetik algoritma program listesi yer almaktadır.

ANA PROGRAM

```
% Grafik arayüzdeki değerlerin alınması ve hesaplamanın başlatılması
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Ünite1 değerlerinin tablodan okunması
P1max=str2num(get(handles.edit1,'string'));
P1min=str2num(get(handles.edit2,'string'));
a1=str2num(get(handles.edit3,'string'));
b1=str2num(get(handles.edit4,'string'));
c1=str2num(get(handles.edit5,'string'));
tu1=str2num(get(handles.edit6,'string'));
td1=str2num(get(handles.edit7,'string'));
hs1=str2num(get(handles.edit8,'string'));
cs1=str2num(get(handles.edit9,'string'));
tcs1=str2num(get(handles.edit10,'string'));
ini1=str2num(get(handles.edit11,'string'));

%Ünite2 değerlerinin tablodan okunması
P2max=str2num(get(handles.edit12,'string'));
P2min=str2num(get(handles.edit13,'string'));
a2=str2num(get(handles.edit14,'string'));
b2=str2num(get(handles.edit15,'string'));
c2=str2num(get(handles.edit16,'string'));
tu2=str2num(get(handles.edit17,'string'));
td2=str2num(get(handles.edit18,'string'));
hs2=str2num(get(handles.edit19,'string'));
cs2=str2num(get(handles.edit20,'string'));
tcs2=str2num(get(handles.edit21,'string'));
ini2=str2num(get(handles.edit22,'string'));

%Ünite3 değerlerinin tablodan okunması
P3max=str2num(get(handles.edit23,'string'));
P3min=str2num(get(handles.edit24,'string'));
a3=str2num(get(handles.edit25,'string'));
```

```
b3=str2num(get(handles.edit26,'string'));
c3=str2num(get(handles.edit27,'string'));
tu3=str2num(get(handles.edit28,'string'));
td3=str2num(get(handles.edit29,'string'));
hs3=str2num(get(handles.edit30,'string'));
cs3=str2num(get(handles.edit31,'string'));
tcs3=str2num(get(handles.edit32,'string'));
ini3=str2num(get(handles.edit33,'string'));

%Ünite4 değerlerinin tablodan okunması
P4max=str2num(get(handles.edit34,'string'));
P4min=str2num(get(handles.edit35,'string'));
a4=str2num(get(handles.edit36,'string'));
b4=str2num(get(handles.edit37,'string'));
c4=str2num(get(handles.edit38,'string'));
tu4=str2num(get(handles.edit39,'string'));
td4=str2num(get(handles.edit40,'string'));
hs4=str2num(get(handles.edit41,'string'));
cs4=str2num(get(handles.edit42,'string'));
tcs4=str2num(get(handles.edit43,'string'));
ini4=str2num(get(handles.edit44,'string'));

%Ünite5 değerlerinin tablodan okunması
P5max=str2num(get(handles.edit45,'string'));
P5min=str2num(get(handles.edit46,'string'));
a5=str2num(get(handles.edit47,'string'));
b5=str2num(get(handles.edit48,'string'));
c5=str2num(get(handles.edit49,'string'));
tu5=str2num(get(handles.edit50,'string'));
td5=str2num(get(handles.edit51,'string'));
hs5=str2num(get(handles.edit52,'string'));
cs5=str2num(get(handles.edit53,'string'));
tcs5=str2num(get(handles.edit54,'string'));
ini5=str2num(get(handles.edit55,'string'));

%Talep değerlerinin tablodan okunması
s1=str2num(get(handles.edit56,'string'));
s2=str2num(get(handles.edit57,'string'));
s3=str2num(get(handles.edit58,'string'));
s4=str2num(get(handles.edit59,'string'));
s5=str2num(get(handles.edit60,'string'));
s6=str2num(get(handles.edit61,'string'));
s7=str2num(get(handles.edit62,'string'));
s8=str2num(get(handles.edit63,'string'));
s9=str2num(get(handles.edit64,'string'));
s10=str2num(get(handles.edit65,'string'));
s11=str2num(get(handles.edit66,'string'));
s12=str2num(get(handles.edit67,'string'));
s13=str2num(get(handles.edit68,'string'));
```

```
s14=str2num(get(handles.edit69,'string'));
s15=str2num(get(handles.edit70,'string'));
s16=str2num(get(handles.edit71,'string'));
s17=str2num(get(handles.edit72,'string'));
s18=str2num(get(handles.edit73,'string'));
s19=str2num(get(handles.edit74,'string'));
s20=str2num(get(handles.edit75,'string'));
s21=str2num(get(handles.edit76,'string'));
s22=str2num(get(handles.edit77,'string'));
s23=str2num(get(handles.edit78,'string'));
s24=str2num(get(handles.edit79,'string'));

%Ünite özellikleri tablosu
table1=[P1max   P2max   P3max   P4max   P5max;
        P1min   P2min   P3min   P4min   P5min;
           a1      a2      a3      a4      a5;
           b1      b2      b3      b4      b5;
           c1      c2      c3      c4      c5;
          tu1     tu2     tu3     tu4     tu5;
          td1     td2     td3     td4     td5;
          hs1     hs2     hs3     hs4     hs5;
          cs1     cs2      cs3     cs4     cs5;
          tcs1    tcs2    tcs3    tcs4    tcs5;
          ini1    ini2    ini3     ini4    ini5];

%Talep edilen enerji tablosu
table2=[s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 s19 s20 s21
s22 s23 s24];

%Talep değerlerinin kontrolü
for i=1:24
if table2(i)>P1max+P2max+P3max+P4max+P5max
s=sprintf('%d. saat için üretilebilinecek maksimum değeri aştınız.Lütfen tekrar değer
giriniz',i);
disp(s)
errordlg('Talep değerleri ünitelerin en yüksek üretim değerlerinden fazla
olamaz.Lütfen tekrar değer giriniz','Hata')
return;
end
end
for i=1:24
if table2(i)<min(table1(2,:))
s=sprintf('%d. saat için üretilebilinecek minimum değeri geçmediniz.Lütfen tekrar
değer giriniz',i);
disp(s)
errordlg('Üretilebilecek minimum değeri geçmediniz.Lütfen tekrar değer
giriniz','Hata')
return;
end
```

```
end

%Tanımlar
pop_size=100;                                   %birey sayısı
mate_size=pop_size;                             %seçilecek cift sayısı
max_gener=2000;                                 %maximum generation
mut_rate=0.001;                                 %mutasyon oranı.

%Parametreler
for i=1:pop_size
POP(:,:,i)=round(rand(U,T));
end
for generation=1:max_gener

%Toplam maliyet
[obj,ceza1,ceza2,ceza3]=f_amac_fonk(POP,table1,table2,U,T,pop_size);
[fitness]=f_fitness1(obj,pop_size);
[p r]=max(fitness);

%Seçim
[selected]=f_select(fitness,mate_size);
[parent1,parent2]=f_parents(POP,selected,mate_size);

%Çaprazlama
[cr_POP]=f_cross1(parent1,parent2,mate_size,U,T);

%Mutasyon
[mt_POP]=f_mutation(cr_POP,pop_size,U,T,mut_rate);
for k=1:pop_size
for u=1:U
for t=1:T
POP(u,t,k)=mt_POP(k,((u-1)*T)+t);
end
end
end

%Ceza fonksiyonu
[obj,ceza1,ceza2,ceza3]=f_amac_fonk(POP,table1,table2,U,T,pop_size);
[fitness]=f_fitness1(obj,pop_size);

%Elitzm
[p r]=min(fitness);
POP(:,:,r)=elitist;

[obj,ceza1,ceza2,ceza3]=f_amac_fonk(POP,table1,table2,U,T,pop_size);
[fitness]=f_fitness1(obj,pop_size);

%Gözlem parametreleri
if mod(generation,10)==0
```

```
clc;
iterasyon_no=generation
minimum=obj(:,r)
[sonuc,a]=f_sonuc(POP,table1,table2,U,T,r);
nufus=POP(:,:,r)        %popülasyondaki en iyi birey
sonuc
a
fprintf('power=%d   up=%d   down=%d',ceza1(r),ceza2(r),ceza3(r));
u1=nufus(1,:).*table1(1,1);
u2=nufus(2,:).*table1(1,2);
u3=nufus(3,:).*table1(1,3);
u4=nufus(4,:).*table1(1,4);
u5=nufus(5,:).*table1(1,5);
saat=1:1:24;
uretim=u1+u2+u3+u4+u5;

%Maliyet eğrisi
figure(1);
hold on
plot(iterasyon_no,minimum,'r-')
axis([1 iterasyon_no 260000 400000])

%Talep ve üretilen
figure(2);
plot(saat,table2,'r:',saat,uretim,'b-');
xlabel('Gunluk Periyot');
ylabel('Talep Edilen Eneji');
title('Talep Karsilama Egrisi');

%Kısıtlamalar
figure(3);
subplot(6,1,1);
bar(ceza1);
xlabel('Popülasyondaki Bireyler');
ylabel('Ceza Sasyilari');
title('Talebi Karsilama Kisitlamasi');

subplot(6,1,3);
bar(ceza2);
xlabel('Popülasyondaki Bireyler');
ylabel('Ceza Sayilari');
title('En Az Devrede Kalma Kisitlamasi');

subplot(6,1,5);
bar(ceza3);
xlabel('Popülasyondaki Bireyler');
ylabel('Ceza Sayilari');
title('En Az Devre Disi Kalma Kisitlamasi');
end
```

```
end

u1=nufus(1,:).*table1(1,1);
u2=nufus(2,:).*table1(1,2);
u3=nufus(3,:).*table1(1,3);
u4=nufus(4,:).*table1(1,4);
u5=nufus(5,:).*table1(1,5);
m1=zeros(1,24);
m2=zeros(1,24);
m3=zeros(1,24);
m4=zeros(1,24);
m5=zeros(1,24);

%Her ünitenin 24 saatlik maliyeti
for i=1:24
if(u1(i)~=0)
m1(i)=(table1(3,1))+(table1(4,1)*table1(1,1))+(table1(5,1)*table1(1,1)^2);
else
m1(i)=0;
end
end
for i=1:24
if(u2(i)~=0)
m2(i)=(table1(3,2))+(table1(4,2)*table1(1,2))+(table1(5,2)*table1(1,2)^2);
else
m2(i)=0;
end
end
for i=1:24
if(u3(i)~=0)
m3(i)=(table1(3,3))+(table1(4,3)*table1(1,3))+(table1(5,3)*table1(1,3)^2);
else
m3(i)=0;
end
end
for i=1:24
if(u4(i)~=0)
m4(i)=(table1(3,4))+(table1(4,4)*table1(1,4))+(table1(5,4)*table1(1,4)^2);
else
m4(i)=0;
end
end
for i=1:24
if(u5(i)~=0)
m5(i)=(table1(3,5))+(table1(4,5)*table1(1,5))+(table1(5,5)*table1(1,5)^2);
else
m5(i)=0;
end
end
```

```
%Saatlik toplam
su=zeros(1,24);
for i=1:24
su(i)=m1(i)+m2(i)+m3(i)+m4(i)+m5(i);
end

%Sonucların yazdırılması
%Her saat için ünite
for i=80:103
str=['set(handles.edit' int2str(i) ',"string",m1(' int2str(i-79) '))'];
eval(str);
end
for i=104:127
str=['set(handles.edit' int2str(i) ',"string",m2(' int2str(i-103) '))'];
eval(str);
end
for i=128:151
str=['set(handles.edit' int2str(i) ',"string",m3(' int2str(i-127) '))'];
eval(str);
end
for i=152:175
str=['set(handles.edit' int2str(i) ',"string",m4(' int2str(i-151) '))'];
eval(str);
end
for i=176:199
str=['set(handles.edit' int2str(i) ',"string",m5(' int2str(i-175) '))'];
eval(str);
end
%Her saat için toplam ünite
for i=200:223
str=['set(handles.edit' int2str(i) ',"string",su(' int2str(i-199) '))'];
eval(str);
end

%Toplamlar
t12(1)=sum(m1(1:12));
t12(2)=sum(m2(1:12));
t12(3)=sum(m3(1:12));
t12(4)=sum(m4(1:12));
t12(5)=sum(m5(1:12));
for i=224:228
str=['set(handles.edit' int2str(i) ',"string",t12(' int2str(i-223) '))'];
eval(str);
end
tm1=sum(t12(1:5));
set(handles.edit229,'string',tm1);
t12(1)=sum(m1(13:24));
```

```
t12(2)=sum(m2(13:24));
t12(3)=sum(m3(13:24));
t12(4)=sum(m4(13:24));
t12(5)=sum(m5(13:24));
for i=230:234
str=['set(handles.edit' int2str(i) ',"string",t12(' int2str(i-229) '))'];
eval(str);
end
set(handles.edit235,'string',minimum);

%Diğer butonun gösterilmesi
set(handles.pushbutton3,'visible','on')
set(handles.pushbutton2,'visible','on')
set(handles.pushbutton1,'visible','off')

%Ünite değerleri tablosunun gizlenmesi
for i=1:18
str=['set(handles.text' int2str(i) ',"visible","off")'];
eval(str);
end
for i=1:55
str=['set(handles.edit' int2str(i) ',"visible","off")'];
eval(str);
end

%Sonuc değerlerinin gösterilmesi
for i=22:41
str=['set(handles.text' int2str(i) ',"visible","on")'];
eval(str);
end
for i=238:257
str=['set(handles.text' int2str(i) ',"visible","on")'];
eval(str);
end
for i=80:235
str=['set(handles.edit' int2str(i) ',"visible","on")'];
eval(str);
end
-------------------------------------------------------------------------------------------------------
% Sonuç ekranından giriş ekranına geri dönme
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Diğer butonun gösterilmesi
set(handles.pushbutton3,'visible','off')
set(handles.pushbutton2,'visible','off')
set(handles.pushbutton1,'visible','on')
```

```
%Sonuc değerleri tablosunun gizlenmesi
for i=22:41
str=['set(handles.text' int2str(i) ',"visible","off")'];
eval(str);
end
for i=238:257
str=['set(handles.text' int2str(i) ',"visible","off")'];
eval(str);
end
for i=80:235
str=['set(handles.edit' int2str(i) ',"visible","off")'];
eval(str);
end
%Ünite değerleri tablosunun gizlenmesi
for i=1:18
str=['set(handles.text' int2str(i) ',"visible","on")'];
eval(str);
end
for i=1:55
str=['set(handles.edit' int2str(i) ',"visible","on")'];
eval(str);
end
-------------------------------------------------------------------------------------------------
% Ünitelerin çalışma saatleri
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
u1=zeros(1,24);
u2=zeros(1,24);
u3=zeros(1,24);
u4=zeros(1,24);
for i=1:24
str=['u1(' int2str(i) ')=str2num(get(handles.edit' int2str(i+79) ',"string"));' ];
eval(str);
end
for i=1:24
str=['u2(' int2str(i) ')=str2num(get(handles.edit' int2str(i+103) ',"string"));'];
eval(str);
end
for i=1:24
str=['u3(' int2str(i) ')=str2num(get(handles.edit' int2str(i+127) ',"string"));'];
eval(str);
end
for i=1:24
str=['u4(' int2str(i) ')=str2num(get(handles.edit' int2str(i+151) ',"string"));'];
eval(str);
end
```

```
for i=1:24
str=['u5(' int2str(i) ')=str2num(get(handles.edit' int2str(i+175) ',"string"));'];
eval(str);
end

%Ünite tahsis  takvimi
figure(4)
subplot(10,1,1);
bar(u1);
xlabel('Saatler');
ylabel('Calisma Saatleri');
title('Unite1 Calisma Takvimi');
subplot(10,1,3);
bar(u2);
xlabel('Saatler');
ylabel('Calisma Saatleri');
title('Unite2 Calisma Takvimi');
subplot(10,1,5);
bar(u3);
xlabel('Saatler');
ylabel('Calisma Saatleri');
title('Unite3 Calisma Takvimi');

subplot(10,1,7);
bar(u4);
xlabel('Saatler');
ylabel('Calisma Saatleri');
title('Unite4 Calisma Takvimi');
subplot(10,1,9);
bar(u5);
xlabel('Saatler');
ylabel('Calisma Saatleri');
title('Unite9 Calisma Takvimi');
```

ALT PROGRAMLAR

```
%Başlatma maliyeti
function [HM,CM]=f_ac_mal(POP,table1,U,T,pop_size)
for k=1:pop_size
X=table1(11,:);
Y=max(abs(X));
MASK=zeros(U,T+Y);
MASK(:,Y+1:T+Y)=POP(:,:,k);
for i=1:U
if table1(11,i)>0 MASK(i,Y-table1(11,i)+1:Y)=1;
end
end
CSN=zeros(U,1);
```

```
HSN=zeros(U,1);

for i=1:U
for j=Y+1:Y+T
if MASK(i,j)==1
if MASK(i,j-table1(10,i)-1:j-1)==0 CSN(i,1)=CSN(i,1)+1;
else if MASK(i,j-1)==1 HSN(i,1)=HSN(i,1)+0;
else HSN(i,1)=HSN(i,1)+1;
end
end
end
end
end
HM(k)=table1(8,1:U)*HSN;
CM(k)=table1(9,1:U)*CSN;
end
```

```
%Ceza fonksiyonu
function [amac,ceza1,ceza2,ceza3]=f_amac_fonk(POP,table1,table2,U,T,pop_size)
YM=f_yak_mal(POP,table1,U,T,pop_size);
[HM,CM]=f_ac_mal(POP,table1,U,T,pop_size);
[ceza1,ceza2,ceza3]=f_constraint(POP,table1,table2,U,T,pop_size);
amac=YM+HM+CM+(1500000*ceza1+1500000*ceza2+1500000*ceza3);

unction [kisit1,kisit2,kisit3]=f_constraint(POP,table1,table2,U,T,pop_size)

%kisit #1: demand'in karşılanması
sup=zeros(T,pop_size);
for k=1:pop_size
a=0;
for t=1:T
for u=1:U
sup(t,k)=sup(t,k)+(POP(u,t,k)*table1(1,u));
end
if sup(t,k)<table2(t) a=a+1;
end
end
kisit1(k)=a;
end

%kisit #2: min up time
for k=1:pop_size
X=table1(11,:);
Y=max(abs(X));
MASK=zeros(U,T+Y);
MASK(:,Y+1:T+Y)=POP(:,:,k);
for i=1:U
if table1(11,i)>0 MASK(i,Y-table1(11,i)+1:Y)=1;
end
```

```
end

for i=1:U
for j=1:Y+T
if MASK(i,j)==1 MASK1(i,j)=0;
else MASK1(i,j)=1;
end
end
end

MUTD=zeros(U,T);
for i=1:U
for j=Y+1:Y+T
s=1;
while MASK(i,j-s)==1
MUTD(i,j-Y)=MUTD(i,j-Y)+1;
s=s+1;
if s==j break;
end
end
end
end

sup=0;
for i=1:U
for j=Y+1:Y+T
if (MUTD(i,j-Y)-table1(6,i))*(MASK(i,j-1)-MASK(i,j))<0 sup=sup+1;
end
end
end
kisit2(k)=sup;

%kisit #2: min down time
MDTD=zeros(U,T);
for i=1:U
for j=Y+1:Y+T
s=1;
while MASK1(i,j-s)==1
MDTD(i,j-Y)=MDTD(i,j-Y)+1;
s=s+1;
if s==j break;
end
end
end
end

sup=0;
for i=1:U
for j=Y+1:Y+T
```

```
if (MDTD(i,j-Y)-table1(7,i))*(MASK1(i,j-1)-MASK1(i,j))<0 sup=sup+1;
end
end
end

kisit3(k)=sup;
end
```

```
%Çaprazlama
function [cocuk_bin]=f_cross1(dizi1,dizi2,s_number,U,T)
y=U*T;
for n=1:(s_number/2),
for k=1:U,
for m=1:T,
ebeveyn1(n,((k-1)*T)+m)=dizi1(k,m,n);
end;
end;
end;

for n=1:(s_number/2),
for k=1:U,
for m=1:T,
ebeveyn2(n,((k-1)*T)+m)=dizi2(k,m,n);
end;
end;
end;

nokta=ceil(rand((s_number/2),1)*y);
for i=1:(s_number/2),
cocuk1(i,1:nokta(i,1))=ebeveyn1(i,1:nokta(i,1));cocuk1(i,nokta(i,1)+1:y)=
ebeveyn2(i,nokta(i,1)+1:y);
cocuk2(i,1:nokta(i,1))=ebeveyn2(i,1:nokta(i,1));cocuk2(i,nokta(i,1)+1:y)=
ebeveyn1(i,nokta(i,1)+1:y);
end,
cocuk_bin=cat(1,cocuk1,cocuk2);
```

```
%Uygunluk fonksiyonu
function [fitnes]=f_fitness1(mal_fonk,x)
%fitnes=1./(mal_fonk+eps);
[Y Z]=sort(mal_fonk);
[K L]=sort(Y);
fitnes1(Z(:))=((x-L(:))./(x-1))+eps;
fitnes=fitnes1';
```

```
%Mutasyon
function [mu_bin]=f_mutation(cr_POP,pop_size,U,T,mutasyon_rate)
y=U*T;
mo=rand(pop_size,y);
```

```
mo=round(mo+(0.5-mutasyon_rate));
mu_bin=mod((cr_POP+1+mo),2);

function [ebeveyn1,ebeveyn2]=f_parents(nufus_bin,secim,s_number)
ebeveyn1(:,:,1:(s_number/2))=nufus_bin(:,:,secim(1:(s_number/2),1));
ebeveyn2(:,:,1:(s_number/2))=nufus_bin(:,:,secim((s_number/2)+1:s_number,1));

function [secim]=f_select(fitnes,s_number)
rullet=sum(fitnes)*rand(s_number,1);
for j=1:s_number,
dilim=0;i=0;
while dilim<rullet(j,1),
i=i+1;
dilim=dilim+fitnes(i,1);
end,
secim(j,1)=i;
end,
```

```
%Sonuçlar
function [sup,a]=f_sonuc(POP,table1,table2,U,T,r)
sup=zeros(1,T);
for t=1:T,
for u=1:U,
sup(1,t)=sup(1,t)+(POP(u,t,r)*table1(1,u));
end
sup(1,t)=sup(1,t)-table2(t);
end
a=sum(sup);
```

```
%Yakıt maliyeti
function [yak_mal]=f_yak_mal(POP,table1,U,T,pop_size)
for k=1:pop_size,
for j=1:T,
for i=1:U,
FC(i,j)=(table1(3,i)*POP(i,j,k))+(table1(4,i)*POP(i,j,k)*table1(1,i))+(table1(5,i)*
(POP(i,j,k)*table1(1,i))^2);
end
end

yak_mal(k)=sum(sum(FC(:,:)));
end;
```

ÖZGEÇMİŞ

1980 yılında İstanbul'da doğdu. İlk ve orta öğrenimini İstanbul'da tamamladı. Lise öğrenimini Tuzla Anadolu Meslek Lisesi Elektronik Bölümü'nde yaptı.1999 yılında girdiği Kocaeli Üniversitesi Teknik Eğitim Fakültesi, Elektronik Öğretmenliği Programından 2003 yılında mezun oldu. Aynı yıl başladığı Elektronik Öğretmenliği görevine halen Köseköy Anadolu Teknik Lisesi'nde Elektrik Elektronik Teknolojisi Alanı şefi olarak devam etmektedir.

2003 yılında başladığı Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Elektronik Bilgisayar Eğitimi Ana Bilim Dalı'nda yüksek lisans öğrenimine devam etmektedir.