T.C.
VAN YÜZÜNCÜ YIL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

REEL-DEĞER KODLAMA İÇEREN GENETİK ALGORİTMALAR İÇİN ADAPTASYON SÜRECİ GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

HAZIRLAYAN: Ahmet Fatih KAZANKAYA
DANIŞMAN : Doç. Dr. Rıdvan SARAÇOĞLU

VAN-2019

T.C.
VAN YÜZÜNCÜ YIL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

REEL-DEĞER KODLAMA İÇEREN GENETİK ALGORİTMALAR İÇİN ADAPTASYON SÜRECİ GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

HAZIRLAYAN: Ahmet Fatih KAZANKAYA

VAN-2019

KABUL VE ONAY SAYFASI

Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda Doç. Dr. Rıdvan SARAÇOĞLU danışmanlığında, Ahmet Fatih KAZANKAYA tarafından sunulan "Reel-Değer Kodlama içeren Genetik Algoritmalar için Adaptasyon Süreci Geliştirilmesi" isimli bu çalışma Lisansüstü Eğitim ve Öğretim Yönetmeliği'nin ilgili hükümleri gereğince 05/07/2019 tarihinde aşağıdaki jüri tarafından oy birliği ile başarılı bulunmuş ve Yüksek Lisans tezi olarak kabul edilmiştir.

Başkan:.Doç Dr. Rıdvan SARAÇOĞLU İmza:

Üye:.Dr. Öğr. Üyesi İlker Ali ÖZKAN İmza:

Üye:.Dr. Öğr. Üyesi A. Oğuz KIZILÇAY İmza:

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./..../….. tarih ve ..... sayılı kararı ile onaylanmıştır.

İmza
Prof. Dr. Suat ŞENSOY
Enstitü Müdürü

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

(İmza)

Ahmet Fatih KAZANKAYA

ÖZET

REEL-DEĞER KODLAMA İÇEREN GENETİK ALGORİTMALAR İÇİN ADAPTASYON SÜRECİ GELİŞTİRİLMESİ

KAZANKAYA, Ahmet Fatih
Yüksek Lisans Tezi, Elektrik-Elektronik Mühendisliği Anabilim Dalı
Tez Danışmanı: Doç. Dr. Rıdvan SARAÇOĞLU
Temmuz 2019, 69 sayfa

Genetik algoritma (GA), optimizasyon problemlerini çözmek için doğadaki yaşam döngüsünü ve doğal seçilimi simüle ederek geliştirilmiş sezgi ötesi bir yöntemdir. Bu çalışmada, GA'nın mevcut yapısına ilave bir süreç eklenmesiyle algoritmanın geliştirilmesi amaçlanmıştır. Eklenen bu yeni süreç bir adaptasyon yöntemidir. Popülasyonun bireylerinin algoritmanın mevcut en iyi çözümüne uyum sağlaması bu yöntemin temelini oluşturnaktadır. Bu adaptasyon metodu ile istenilen optimizasyon değerlerine daha kısa nesilde ulaşılmaktadır. Belirli bir nesil sayısı için ise daha iyi sonuçlara ulaşmak mümkündür. Çeşitli optimizasyon test fonksiyonları ile yapılan deneysel çalışmaların sonuçları da bu yeni sürecin GA'yı geliştirdiğini ve hızlandırdığını göstermiştir.

Anahtar kelimeler: Adaptasyon, Genetik algoritma, Geliştirilmiş genetik algoritma

ABSTRACT

DEVELOPING ADAPTATION PROCESS FOR REAL CODED GENETIC ALGORITHM

KAZANKAYA, Ahmet Fatih
M. Sc. Thesis, Electrical-Electronics Engineering
Supervisor: Assoc. Prof. Dr. Rıdvan SARAÇOĞLU
July 2019, 69 pages

The genetic algorithm (GA) is a nature-inspired method that simulates biological phenomena like the life cycle and the natural selection in order to solve optimization problems. This study aimed to enhance GA by adding an additional operation on it's current state. This new process is also a biological phenomenon that includes an adaptation method. Making individuals to adapt to the best solution of the algorithm is the very basis of this method. With this adaptation method, it is possible to reach better results in a shorter time. Experimental results, which was evaluated by using a variety of optimization test functions show that this new process accelerated the algorithm and made solutions to be found in fewer generations especially for some specific cases.

Keywords: Adaptation, Genetic algorithm, Enhanced genetic algorithm

ÖN SÖZ

Tezimin tüm aşamalarında hiçbir desteğini esirgemeyen danışmanım Sayın Doç. Dr. Rıdvan SARAÇOĞLU'na yürekten şükranlarımı sunmak istiyorum. Ayrıca babam Prof. Dr. Ahmet KAZANKAYA'ya ve bana her şekilde yardım eden arkadaşım Burak Kaan ERCEDOĞAN'a teşekkür etmek istiyorum.

2019

Ahmet Fatih KAZANKAYA

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ŞEKİLLER LİSTESİ

Şekil **Sayfa**

ÇİZELGELER LİSTESİ

Çizelge **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılan bazı kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklama
GA	Genetik Algoritma
GGA	Geliştirilmiş Genetik Algoritma
KGA	Klasik Genetik Algoritma
YD	Yük Dengeleme
YDMO	Yük Dengeleme Maliyet Oranı
YDS	Yük Dengeleme Stratejisi
SDÖ	Sezgisel Deneyim Örnekleme
SDM	Su Dağıtım Merkezi
GSP	Gezgin Satıcı Problemi
HGA	Hibrit Genetik Algoritma
AGA	Aralık Genetik Algoritma
HAGA	Hibrit Aralık Genetik Algoritma
KKO	Karınca Kolonisi Optimizasyonu
PSO	Parçacık Sürü Optimizasyonu
TSMY	Transfer Sertliği Matrisi Yöntemi
MCSY	Monte Carlo Simülasyon Yöntemi
DNA	Deoksiribo Nükleik Asit
AGSP	Asimetrik Gezgin Satıcı Problemi
BİDA	Bulut İçerik Dağıtım Ağı
BDSİD	Bulut Depolamada Statik İçerik Dağıtımı

1. GİRİŞ

İlk olarak doğal evrimde gözlemlenen bazı süreçleri taklit etmek amacıyla geliştirilen genetik algoritma (GA), biyologlar da dahil birçok insanın, devasa karmaşıklık seviyesindeki yaşantımızda gözlemlenen evrimleşmeye merak duymalarıyla, optimizasyon problemlerinin çözümünde kullanılmaya başlanmıştır. GA, babası kabul edilen John Holland'ın 1970'teki ilk adımlarıyla başlayarak gelişimini artan bir tempoyla günümüze dek devam ettirmiştir (McCall, 2005). Bilgisayar bilimleri ve optimizasyon araştırmalarında GA, daha geniş bir algoritma sınıfına ait olan, doğal seleksiyon sürecinden esinlenen ve evrimsel sürece benzeyen arama ve eniyileme yöntemidir. GA, seçilim, çaprazlama ve mutasyon gibi biyolojiden ilham alan olgulara dayanarak optimizasyon ve arama sorunlarına, yüksek kalitede çözümler üretmek amacıyla yaygın olarak kullanılmaktadır.

GA'da, optimizasyon probleminin çözümü için belirlenen aday bireylerin, mevcut popülasyonu daha iyi genlere evrimleştirmesi beklenir (Mitchell, 1996). Her aday, mutasyona uğratılabilen ve değiştirilebilen bir dizi özelliğe sahip olmalıdır. Evrim, genellikle rastgele oluşturulmuş bireylerden oluşan bir popülasyonla başlar, her bireyin uygunluk değeri hesaplanır ve yeni bir nesil olarak devam eder (Sadeghi ve ark., 2016). Daha fazla uyum gösteren bireyler, mevcut popülasyondan seçilir ve bireylerin genleri, yeni bir nesil oluşturmak üzere değiştirilir. Yeni nesiller, algoritmanın daha sonraki yinelemesinde kullanılır (Whitley, 1994; Koza, 1992). Genetik uygunluk fonksiyonu tanımlandıktan sonra seçilim, evrimleme, çaprazlama ve mutasyon evrelerinin tekrarlı uygulanması yoluyla bir gen popülasyonu oluşturulmaya başlanır (Ting, 2005).

Popülasyon büyüklüğü, tamamen duruma ve parametrelere bağlıdır. İlk popülasyon çoğunlukla rastgele olarak üretilir ve tüm olası gen varyasyonlarına izin verilir (Banzhaf ve ark., 1998). Her nesilde, mevcut nüfusun bir kısmı yeni bir nesil üretecek şekilde seçilir. Bireyler, uygunluk değeri düşük olandan yüksek olana doğru sıralanır ve seçilim olasılığı daha yüksek olan bireyler, parametrelere bağlı olarak seçilmeye başlanır (Echegoyen ve ark., 2012). Her bir genin uygunluğunu değerlendiren uygunluk fonksiyonları, en iyi genleri seçmekle görevlidir. Problemin formülüne bağlı olarak işlem yapan uygunluk fonksiyonu, temsili birey üzerinde tanımlanırken, temsil

edilen bireyin kalitesini de ölçer (Taherdangkoo ve ark., 2012). Daha sonra, çaprazlama ve mutasyonun sıralı kombinasyonu ile seçilen bireylerden yeni nesil popülasyonlar üretilir (Goldberg, 1989).

Üzerinde çalışılan problemin makul sonuçlarına ulaşırken, mutasyon oranı, çaprazlama oranı ve popülasyon büyüklüğü parametrelerinin ayarlanması son derece önemlidir. Üretilecek yeni bireyler için havuzdan bir çift ebeveyn seçilir (Coffin ve ark., 2008; McCall, 2005). Çaprazlama yöntemi, ebeveynlerin tipik özelliklerinin çoğunu paylaşan yeni bir birey yaratmak için kullanılmaktadır (Poon ve Carter, 1995). Her yeni çocuk için yeni ebeveynler seçilir ve süreç uygun büyüklükte yeni bir çözüm bulununcaya kadar devam eder (Cha ve Tappert, 2009). Bu süreçler sonucunda, ilk jenerasyondan çoğunlukla farklı genlere sahip yeni nesil bireyler oluşur (Shir ve Ofer, 2012). Popülasyondaki uygunluğun bu süreçlerle artması beklenir çünkü ilk neslin en iyi bireyleri ile onlardan biraz daha az uygunluk oranına sahip diğer bireyler birbiriyle çaprazlanmak üzere seçilmiştir (Janikow ve Michalewicz, 1991; Patrascu ve Pop, 2014).

Daha az uygunluk oranına sahip bu bireylerin popülasyon hala bünyesinde bulunmasının asıl amacı ise hem genetik havuzdaki hem de gelecek nesillerdeki genetik çeşitliliği arttırmaktır (Baluja ve ark., 1995). Oluşan yeni yavrular ve rastgele seçilen bazı bireyler, düşük olasılıklı mutasyonlara tabi tutulur (Chen ve ark., 2017). Bu da bireylerin bazı genlerinin değiştirilebilmesini sağlar (Harik, 1997). Mutasyon evresi genellikle nüfustaki çeşitliliği korumak ve yerel en iyiye takılmayı önlemek için kullanılır (Zhang ve Chung, 2007). Algoritma ya maksimum sayıda nesil üretildiğinde ya da popülasyon için tatmin edici bir skor seviyesine ulaştığında sonlanır (McCall, 2005).

1.1. Amaç ve Tezin Önemi

Bu çalışmada, doğal seleksiyon sürecinden esinlenen, evrimsel sürece benzeyen, optimizasyon ve arama sorunlarına, mümkün olan en iyi çözümleri üretmek amacıyla kullanılan arama ve eniyileme yöntemi olan GA'nın geliştirilmesi amaçlanmıştır. Ana genetik evresinden ikisi çaprazlama ve mutasyon olarak bilinen GA'ya, çaprazlama ve mutasyon evreleri arasında gerçekleşen bir adaptasyon evresi eklenmiştir. Bu evre ile algoritmada o ana kadarki en iyi birey donor olarak belirlenip, tamamen rastgele

belirlenecek diğer genlere donorların ilgili genleri klonlanarak mutasyon evresi öncesi daha optimize genlerden oluşan bireyler oluşturulmuştur. Böylelikle algoritmanın hedefine daha az nesil sayısıyla ve dolayısıyla daha erken ve daha iyi bireylerle ulaştırılması hedeflenmiştir.

Çalışmada kullanılan bireyler ve bireylerin genleri tıpkı Klasik Genetik Algoritma (KGA)'da olduğu gibi belirlenen boyut doğrultusunda rastgele olarak oluşturulmuştur. Oluşturulan bireylerin genleri uygunluk fonksiyona tabi tutulması yoluyla uygunluk değerleri hesaplanmaktadır. Elde edilen uygunluk değerleri, belirlenen optimum değer aralığına en yakın olandan en uzak olana doğru sıralanmaktadır (Holland, 1992). Belirlenen bu bireyler seçilim evresi adı altında, algoritma içerisinde uygun görülmüş katsayılar doğrultusunda 3'lü turnuva metoduna tabi tutulmaktadır. Seçilen bu üç birey içinde en iyi uygunluk değerine sahip birey popülasyonu oluşturmak üzere birey havuzuna atılmaktadır. Bu işlem popülasyon sayısı kadar tekrar edip popülasyon boyutuna ulaşıldığında sona ermektedir.

Seçilen bu bireyler çaprazlama aşamasına alınır ve bu işlem ikişer ebeveynden ikişer yavru oluşturulacak şekilde maksimum yavru sayısı popülasyon sayısına eşit olana kadar devam eder (Bakirtzis ve ark., 2002; Gürbüz, 2010). Çaprazlama evresiyle oluşan yavru bireyler arasındaki genetik çeşitliliği arttırmak amacıyla oluşturulan yavrular belirli mutasyonlara tabi tutulurlar (Guo ve ark., 2010). Mutasyona uğrama olasılığı tamamen rastgele olup belirlenen mutasyon oranı ile doğru orantılıdır. Bu evreyle nesiller arasındaki gen tekrarı minimum seviyelere indirilir.

Çaprazlama ve mutasyon evrelerinden oluşan GA'ya, literatürden farklı olarak önerilen adaptasyon evresi eklenmiştir. Yeni olan bu evre çaprazlama ve mutasyon evrelerinin arasında yer almaktadır. Bu yeni evre ile çaprazlama yoluyla oluşan yavru bireylerin bazı genleri adaptasyon evresinin getireceği yenilik ile donor genler ile değiştirilmiştir. Bu sayede optimum aralığa çok daha yakın gen değerlerine sahip olunmuş ve böylelikle algoritmanın hedefe daha erken ve daha iyi bireylerle ulaştırılması sağlanmıştır.

2. KAYNAK BİLDİRİŞLERİ

GA, istenilen çözüme daha kısa ve verimli bir yolla ulaşmak için yıllarıdır çeşitli faktörler göz önünde tutularak, geliştirilip genişletilmeye çalışılmıştır. Bunlar araştırıcının amacına göre değişmekle beraber optimizasyon tasarımı, uygunluk fonksiyonu ve aralığı, seçilim metodu, popülasyon tipi ve boyutu, nesil sayısı, çaprazlama ve mutasyon değerleri, elit birey katsayısı, durdurma kriterleri ve gen havuzu gibi önemli faktörler olmaktadır.

Ali ve ark. (2018), farklı optimizasyon problemlerinin üstesinden gelmek için birçok Geliştirilmiş Genetik Algoritma (GGA)'nın önerildiğini ancak diğer algoritmalar arasından GA'nın karmaşık özelliklere sahip optimizasyon problemlerinde tutarlı performans gösteremediğini belirtmiştirler. Çalışmalarında, karmaşık optimizasyon problemlerini çözmek için gerçek kodlanmış bir GA sınıfı geliştirmiştirler ve bu algoritmanın temel amacının, problem arama alanında daha fazla çözüm keşfederek erken yakınsama ve durgunluk senaryolarını önlemek için yeni bir diferansiyel evrim geçidini GA'yla birleştirerek algoritmanın arama yeteneğini arttırmak olduğunu belirtmişlerdir. Sonuçların, GA çeşitlerine kıyasla önerilen algoritmaların etkinliğini ve sağlamlığını onayladığını belirtmişlerdir.

Akbari ve ark. (2017), heterojen bilgisayar sistemlerinde önemli sorunlardan birisi olan görev zamanlamasını, işlem süresini en aza indirgeyerek ve işlemlerin işlemcilere atanmasında paralelleştirmeyi en üst düzeye çıkararak sistemin genel performansını en iyi duruma getirmeyi amaçlayan geliştirilmiş bir GA sunmuşlardır. Sonuçların, diğer görev zamanlama algoritmalarına kıyasla önemli gelişmeler gösterdiğini belirtmişlerdir.

Alam ve Raza (2018), hem Yük Dengeleme (YD) hem de Yük Dengeleme Maliyet Oranı (YDMO)'nı optimize etmek amacıyla, iş akışı uygulaması için kuantumdan ilham alan GA tabanlı Yük Dengeleme Stratejisi (YDS)'ni önermişlerdir. Strateji planlama kararlarında iletişim maliyetinin ve kritik yolun dikkate alınmasını sağlayan bu algoritmada, kalite değerlendirmeleri yapmış ve olumlu sonuçlar aldıklarını söylemişlerdir.

Bakirtzis ve ark. (2002), ayrı kontrol değişkenleri ile optimal güç akışı sağlayan ve hem sürekli hem de aktif güç çıkışları ile değiştirilebilir geliştirilmiş bir GA sunmuşlardır. Algoritmanın verimliliğini ve doğruluğunu arttırmak için probleme özel operatörler tanıtmış olsalar da şebeke akış limitleri, bar gerilim sınırları ve jeneratör reaktif kabiliyetleri gibi işlem kısılmalarının GA fonksiyonunda büyük bir dezavantaj olduğunu belirtmişlerdir.

Bi ve ark. (2015), makalelerinde ilk popülasyonun örneklemesinde sezgisel alan bilgisi kullanılarak GA'ların etkinliğinin nasıl geliştirilebileceğini göstermek için örnek olarak su dağıtım sistemlerinin optimal tasarımını seçmişlerdir. Sezgisel Deneyim Örnekleme (SDÖ) adı verilen yeni bir sezgisel prosedür önermişler ve değişen büyüklükteki yedi Su Dağıtım Merkezi (SDM) üzerinde test etmişlerdir. Sonuçların, SDÖ'nun hem hesaplama verimliliği hem de optimum çözüm bulma yeteneği açısından açıkça en iyi performansı gösterdiğini belirtmişlerdir.

Bu ve ark. (2013), kirlilik kaynaklarını araştırmak amacıyla, hafif kirli alanları çok kirli bölgelere oranlayan, bireyler arasındaki rekabete müdahale edebilen, rehber bireyler ve özel iyileştirme varyasyonları ekleyebilen geliştirilmiş bir GA sunmuşlardır.

Carr (2014), yazısında GA'ların maksimum veya minimumu bulmak için kullanılan bir tür optimizasyon algoritması olduğunu belirterek yeni başlayanlar için GA'ları tanıtmış ve açıklamıştır. GA'ların bileşenlerini ve nasıl yazıldığını göstermiştir. MATLAB kullanarak, klasik Gezgin Satıcı Problemi (GSP)'ni çözen bir GA da dahil olmak üzere birkaç örnek programı ve ayrıca GA'ların tarihçesini, güncel uygulamaları ve gelecekteki muhtemel gelişmelerini tartışmıştır.

Elsayed ve ark. (2011), GA'nın performansının diğer evrimsel algoritmalardan kısıtlı optimizasyon değerlerinden dolayı daha düşük olduğu sonucuna ulaşarak çoklu ebeveyn çaprazlama ve yerel arama tekniği ile geliştirilmiş bir GA sunmuşlardır. Bu algoritmada, mutasyon yerine çeşitlilik operatörü kullanmışlardır. Sonuçlara baktıklarında, algoritmanın daha hızlı yakınsama değerlerine ulaştığını gözlemlemişlerdir.

Guo ve ark. (2010), Hibrit Genetik Algoritma (HGA), Aralık Genetik Algoritma (AGA) ve Hibrit Aralık Genetik Algoritma (HAGA) dahil olmak üzere yeni geliştirilmiş GA prosedürlerinin üç farklı türünü sunmuşlardır. Kanıtlanmış sistemlerin sonuçlarına

göre, HGA'nın, KGA'dan optimum tasarımı daha iyi belirleyebildiğini söylemişlerdir. AGA ve HAGA'nın, sistem eğiminin hesaplanmasını önleyebileceğinden, klasik aralık analizi ve izin verilen hata sınırı altında parametrelerin optimum aralığını belirleyebileceğinden ve aralık optimizasyon işlemine GA'nın ilk kez uygulandığını belirtmiştirler.

Gürbüz (2010), Karınca Kolonisi Optimizasyonu (KKO) ve Parçacık Sürü Optimizasyonu (PSO) gibi yeni iki farklı algoritma kullanarak GA'nın çözüm kalitesini geliştirilmeyi ve daha akıllı bir algoritma meydana getirmeyi amaçlamıştır. Önerilen algoritmaların KGA'dan daha iyi sonuçlar verdiğini istatistiksel yöntemler kullanarak ispatlamıştır.

Huan ve ark. (2018), çalışmalarının oto-gövde gelişimini teşvik etmek için geliştirilmiş bir GA kullanarak şekil optimizasyonu sunduğunu belirtmişlerdir. Şekil optimizasyonlarının sorunlarından birkaçı olan statik sertlikten, dinamik öz frekansından, imalat kısıtlamalarından, kütle minimizasyon problemlerinden, önceki çalışmalarında önerdikleri Transfer Sertliği Matrisi Yöntemi (TSMY)'nden ve beyaz gövde çerçevesinin dinamik analizlerinin neden benimsediğinden bahsetmişlerdir. Ek olarak, ölçek vektör yönteminin, tasarım değişkenlerini önemli ölçüde azalttığını belirtmişler ve sonuçları ilerleyen zamanlarda karşılaştıracaklarından bahsetmişlerdir.

Hu ve ark. (2004), serbest uçuşun, bir uçuş yol optimizasyonunda çok önemli olan hava trafik yönetim sistemlerinin öneminden bahsetmişlerdir. Makalelerinde, geliştirilmiş GA kullanarak serbest uçuş yolu optimizasyonu için gerçek zamanlı özelliklere sahip yenilikçi bir algoritma önermişlerdir. Çevrimiçi uçuş yolu optimizasyon problemi için yakın ve uzak gelecekteki uygulamaları kapsayacak şekilde iki tür matematiksel model önerilmişlerdir. GA'ya yakınsama sürecini hızlandırmak ve performansını artırmak için çeşitli iyileştirmeler yapıldığından bahsetmişlerdir. Simülasyon sonuçlarında, yeni algoritmanın yeterince etkili olduğundan ve çevrimiçi serbest uçuş yolu optimizasyon probleminin gerçek zamanlı çözülebilme potansiyeline sahip olduğundan belirtilmiştir.

Hu ve ark. (2014), GA destekli, Kendinden Uyarlamalı Çok Amaçlı Ekolojik Rezervuar İşletme Modeli'ni önermişlerdir. Bu modelin, istatistiksel su kalitesi modellerini, çok amaçlı rezervuar işlemlerini ve kendinden uyarlamalı bir GA'yı genel

bir çerçeve içinde birleştiren bir model olduğunu belirtmişlerdir. Bunlar arasında, Xiangxi Nehri'nin istatistiksel su kalitesi modelleri, rezervuar işletmesi arasındaki ilişkileri ele almak için formüle etmişlerdir. Sonuçlarında, geliştirdikleri algoritmanın ekolojik operasyonda faydaların çoğunu göstermekte olduğunu ve geleneksel veya pratik işlemlerde KGA'dan daha iyi olduğunu belirtmişlerdir. Bunun nedeni ekolojik operasyonda taşkın kontrolü ve çevre korumanın makul bir şekilde göz önünde bulundurulmasıdır.

Huynh ve ark. (2018), rehabilitasyon hastalarının programlanmasında 2D olarak geliştirilmiş bir HGA sunmuşlardır. Ek olarak, dizinin en iyi çözüme verimli şekilde ulaşması için bir zaman çizelgesi algoritması geliştirilmişlerdir. Algoritma sonuçlarının, yaklaşımlarının geleneksel yaklaşımdan daha iyi bir performans gösterdiğini belirtmişlerdir.

Király ve Abonyi (2015), yazılarında servis yerlerinin genişletilmesiyle malzeme taşıma faaliyetlerini azaltmak için yeni bir yöntem sunmuşlardır. Tedarik sisteminin tasarımını, merkezi depolardan gelen talepleri yerine getirirken kapasiteyi ve diğer kısıtlamaları minimum rota maliyetine göre hesaplayan karmaşık bir kombinasyon optimizasyon problemi olarak tanımlamışlardır. Bu entegrenin, Macaristan'ın en büyük enerji sağlayıcılarından birinde mobil mekanik tedarikçisi olarak gerçek bir lojistik problemin çözümünde başarıyla uyguladıklarını belirtmişlerdir.

Park ve ark. (2008), uygunluk fonksiyon değerlendirmelerinin sayısını azaltmak için çok disiplinli optimizasyonlara uygulanabilen geliştirilmiş bir GA önermişlerdir. Dizi tasarımındaki lamine yapıların tasarım kriterlerinin, bağımlı kriterler ve katman dizilerine bağımlı kriterler olmak üzere iki kombinasyon grubuna ayırmış ve bireylerin arama esnasında uygunluk fonksiyon değerlendirme sayısını azaltmak için, kompozit lamine plakanın çoklu tasarım kısıtlamaları altında ağırlık minimizasyonu ile doğrulandığını belirtmişlerdir.

Shao ve ark. (2018), makalelerinde, yapısal stabiliteleri sistematik olarak incelemek için geliştirilmiş bir GA önermişlerdir. Bu algoritmada, başlatma sırasında yapısal kararlılığı arttırmak için katmanlı bir koordinat sıralama yöntemi benimsenmişlerdir. Buna ek olarak, nüfus çeşitliliğini ve algoritmanın en iyi bireyini korumak için yeni bir geçiş spor işlevi tanımlamışlardr. Ayrıca, genel arama ve yerel

optimizasyon hızını arttırmak için GA'daki klasik çaprazlamanın yerini almak üzere bir küre biçimli çaprazlama operasyonun kullanıldığını belirtmişlerdir. Algoritmanın performansı Monte Carlo Simülasyon Yöntemi (MCSY) ve Parçacık Sürü Optimizasyonu (PSO) ile karşılaştırıldığında, algoritmalarının üstün yakınsama değerine ve yüksek kararlılığa sahip olduğunun altını çizmişlerdir.

Shen (2018), elektrikli araçlarda hızlı bir şekilde gelişmekte olan, kapsamlı hesaplama gereklilikleri ve yakınsama sorunları gibi birtakım zorluklar barındıran lityum piller için düşük hesaplama karmaşıklığı ve yüksek başlangıç kararlılığı parametreleriyle geliştirilmiş GA tabanlı bir yöntem önermiştir. Shen, algoritmanın düşük hesaplama karmaşıklığı ile büyük bir performans gösterdiğini ve verilen başlangıç değerinden çok az etkilendiğini ifade etmiştir.

Shrestha ve Mahmood (2016), çeşitliliğin seyreltilme oranının azaltabilir olabileceğine ve buna gecikmiş bir yakınsamanın neden olabileceğini düşünerek, GA örneklerinin bağımsız olarak çalıştırıldığı ve popülasyon üyelerinin periyodik olarak bir Kıta Modeli'ne dönüştürdüğü Ada Modeli'ni ölçeklendirmişlerdir. Bu modelde, ada modelini birden çok web hizmetiyle birlikte yürütmüşlerdir. Bu yeni kavramların GA'nın optimizasyon kalitesini göreceli olarak iyileştirdiğini ifade etmişlerdir.

Song ve ark. (2019), makalelerinin, uydu aralığı programlama problemini analiz etmekte ve matematiksel modeller ve kısıtlamalar getirmekte olduğundan bahsetmişlerdir ve yerel arama yöntemlerini birleştiren etkili bir GGA önermiştirler. GGA, planlama şemasının kalitesini hızlı bir şekilde geliştirmek ve küçük ölçekli optimizasyonlardaki yerel araştırma tekniği için kullanılmıştır. Arama hızını arttırmak için algoritmalarında yeniden düzenlemeler yapılmış ve yineleme sayısına göre ayarlanmış bir mutasyon işlemi kullanılmıştır. Algoritmanın etkinliğini test etmek için, farklı görev ölçeklerinde çeşitli uydu türlerinin hesaplamalarının deneysel doğrulamasını yapmışlar ve bunları diğer algoritmalar ile karşılaştırmıştırlar.

Soni ve Kumar, (2014), GA'ların, doğal evrim sürecini taklit eden popülasyon temelli arama ve optimizasyon tekniği olduğundan ve büyük popülasyon büyüklüğünü içeren sorunları çözmek için farklı çaprazlama ve mutasyon operatörleri bulunması gerektiğini belirtmişlerdir. Böyle bir soruna örnek olarak, büyük bir çözüme sahip olan

GSP'yi vermişlerdir. Çalışmalarında problemin çözümünde yardımcı olan farklı mutasyon operatörleri tartışmışlardır.

Tian ve ark. (2018), çalışmasında, üç boyut örgülü kompozit birleşme biriminin optimizasyon tasarımı için bir algoritma geliştirmiştirler. Tasarımları, objektif değerlendirmeler ve çok boyutlu tasarım alanları açısından yüksek hesaplama problemidir. GA'nın, küresel bir arama yaptığından, bunun da sonucun yerel olarak birleşmesini önlediğinden ve mükemmel ilk çözümler sunabildiğinden bahsetmiştirler. Bunun da çözümlerin doğruluğunu arttırdığını ve yöntemin belirtilen optimizasyon problemi için kesin ve etkili olduğun altını çizmiştirler.

Wang ve ark. (2003), kullanılan matematiksel modellerin doğrusal ve konveks olmayan ve çoğu zaman süreksiz doğası nedeniyle zor bir görev olan proses optimizasyonunun iyileştirilmesi ve kısıtlı problemlere karşı doğası gereğince uygun olmayan KGA'yı, boyut, yer ve bölge yoluyla ilgili bilgi verebilmesi için bir GGA sunmuşlardır. Algoritma sonuçlarının, görselleştirme, küme analizi ve GA kombinasyonunu parametreleri açısından geliştirme gösterdiğinin altını çizmişlerdir.

Wang ve ark. (2018), çok amaçlı optimizasyon problemleri için en çok kullanılan algoritmalardan biri olan dominant sıralanmamış bir GA'yı, DNA kod sözcüklerinin tasarımı için geliştirmişlerdir. Metotlarının, dominant olmayan sıralamalara kısıtlamalar getirdiğini ve performansını diğer DNA kod tasarım yöntemleri ile karşılaştırdıklarında güvenilebilirlik ve kontrol edilebilirlik açısından daha yüksek bir yakınsama hızına ve daha iyi bir popülasyon çeşitliliğine sahip olduğundan bahsetmişlerdir.

Xing ve ark. (2008), Asimetrik Gezgin Satıcı Problemi (AGSP)'nin çeşitli uygulamalarda hala ortaya çıkmasından ve çözümüne yönelik çeşitli sezgisel yaklaşımlar olmasına rağmen hala zor bir kombinasyon optimizasyon sorunu olduğundan bahsetmişlerdir. Çalışmalarında AGSP için uzmanlaşmış yeni bir hibrit yaklaşım önermektedirler. Önerdikleri bu yöntem, bir GGA ve bazı optimizasyon stratejileri içermektedir. Sonuçların, önerilen yaklaşımlarının yayınlanmış diğer birkaç algoritmayı geride bıraktığını gösterdiğini belirtmişlerdir.

Xu ve ark. (2014), görev önceliği ve işlemci seçimi basamaklarından oluşan paralel dağıtılmış heterojen bilgisayar sistemlerinde görev önceliğini en iyi şekilde ayarlayabilen bir görev zamanlama şemasıyla geliştirilmiş bir GA sunmuşlardır.

Yaklaşımlarının temel düşüncesini hem evrimsel hem de sezgisel tabanlı algoritmaların avantajlarından yararlanmak ve aynı zamanda da sakıncalarından kaçınmak olduğunu ifade etmişlerdir. Önerilen algoritmanın, program dışı kaliteyi ve rastgele araştırma yöntemini arttırdığını belirtmişlerdir.

Yang ve XuHu (2016), kısa vadeli trafik akışının tahmin doğruluğunu geliştirmek için, kümeleme stratejisine dayanan tahmin modelli geliştirilmiş bir GA önermişlerdir. Başlangıç bağlantı ağırlıklarını, çeviri ve ölçekleme faktörünü optimize etmek için kullanılan bu algoritmanın sonuçlarının, KGA'ya kıyasla daha yüksek tahmin doğruluğu ve daha iyi doğrusal olmayan montaj kabiliyetine sahip olduğunu belirtmişlerdir.

Zheng ve Zheng (2018), Bulut İçerik Dağıtım Ağı (BİDA)'ndaki karmaşık yol oluşturma, önbellek güncellemesi ve yük dengeleme gibi sorunları çözmek için, Bulut Depolamada Statik İçerik Dağıtımı (BDSİD) adı altında geliştirilmiş sezgisel bir GA önermişlerdir. BDSİD, GA ile modellerine ayrılmış ve en uygun çözümü en uygun içeriğe indirgeyebilmiştir.

3. GENETİK ALGORİTMA

GA, Darwin'in türlerin kökeninde belirtilen doğal evrim teorisi olan en uygun olanın hayatta kalması kavramına dayanan karmaşık optimizasyon problemlerini çözmek için kullanılan bir yöntemdir (Goldberg, 1989). Doğada olduğu gibi, uygun olan türler bozulmadan kalırken uygun olmayan türler elimine edilir. Mevcut olan çok sayıda çözümden uygunluk oranının yüksek olduğu çözümlerin elde edilmesi asıl amaç iken, daha az uygunluğa sahip çözümler ise yine aynı şekilde elimine edildiği GA, genlerin ve kromozomların birbirleriyle olan etkileşimleri doğrultusunda optimum bireylerin hayatta kalmasıdır. Geleneksel yaklaşımdan bazı farklılıklar içeren genetik yaklaşım, organizmaları optimize ederek, doğadaki evrilme sürecini araştırmak ve yeni yaklaşımlar yapabilmek amacıyla kullanılmaktadır (Goldberg, 1989).

GA, kromozom formundaki çözüm kümelerini temsil eder ve bu bağlamda kromozomların uygunluğunu değerlendirir. GA, popülasyondaki tür ve çeşit bilgilerini önemsiz kılan rastgele bireyler oluşturur. Evrimdeki nesil devamı için çaprazlama operatörü kullanırken, popülasyonun çeşitliliğini korumak için de mutasyon operatörü kullanır. Daha uygun kromozomlar daha az uygun kromozomların yerini alır ve önceden belirlenmiş bazı kriterler temelinde optimal çözüm bulunana kadar süreç devam eder.

GA, bireyleri ebeveyn olarak kullanarak her adımda mevcut nüfustan rastgele seçilimler gerçekleştirir ve gelecek nesiller için yavru bireyler üretir. Ardışık nesiller boyunca, nüfus optimal bir çözüme doğru ilerler. Evrimsel bir algoritma olduğu için, her yinelemenin içindeki ilerleme kolayca görülebilir. GA, optimizasyon, tasarım, robotik, görüntü işleme, makine öğrenmesi, otomatik programlama gibi birçok uygulama alanında kullanılabilir. GA, optimizasyon problemlerini çözmek için kullanıldığında, sonuçlar oldukça hızlı bir şekilde elde edilir. Sezgisellik, algoritma tarafından toplanan bilgileri kullanan ve hangi çözüm adayının daha sonra test edileceğine veya bir sonraki kişinin nasıl üretileceğine karar vermek için taşıyıcı görevi gören bir optimizasyon parçasıdır. GA, rastgele arama ve çok amaçlı optimizasyon problemleri için evrimsel algoritmalar arasında en popüler optimizasyon tekniklerinden biridir (Goldberg, 1989).

Doğada var olan popülasyonlardaki her bir birey, yemek, barınak gibi yaşamın temel kaynakları için birbiriyle rekabet eder. Aynı türe ait bireyler üreme ve nesil devamı

için karşı cinsin dikkatini çekmek için yarışırlar. Bu yarış sonucunda galip gelen birey neslinin devamının sağlanması için ilk adımı atmış olacaktır. Sürecin devamında genişlemesi beklenen bu popülasyonda nesil devamı için adına seçilim diyebileceğimiz bir prosedür başlar. Bu seçilim prosedüründe, kötü performans gösteren bireylerin hayatta kalma şansı daha azdır ve en adapte olmuş veya "uygun" bireyler nispeten fazla sayıda yavru üretir. Üreme sırasında, her bir ataya ait iyi özelliklerin rastgele bir biçimde yavrulara aktarıldığı ve ondan sonraki nesillere de aktarımını mümkün kıldığı gözlemlenmektedir. Birkaç kuşaktan sonra, türler yaşamlarına daha uzun bir süre boyunca ve daha adapte olmuş bir şekilde devam edebilir (Goldberg, 2002).

Bu bölümde, en büyük avantajı, sayısal veya matematiksel modellerin başarısız olduğu çoğu durumlarda başarısını kanıtlamış olan, tek noktaya dayanan geleneksel yaklaşımlara kıyasla, çoklu popülasyonlara dayanan GA'dan, algoritmanın kodlanmasına, test fonksiyonlarından, seçilim, çaprazlama, mutasyon gibi genetik operatörlere genel bir bakış sunulmaktadır.

3.1. Tarihi Geçmişi

Holland'ın GA'yı geliştirmedeki etkisi çok önemliydi, ancak farklı geçmişlere sahip diğer birkaç bilim adamı da benzer fikirlerin geliştirilmesinde yer aldı. 1975, Holland'ın kitabının ve lisansüstü öğrencilerden biri olan Ken DeJong'un doktora tezinin yayınlanması açısından GA tarihinde önemli bir yıldı (Michalewicz, 1996). Holland'ın diğer öğrencileri bu alanda daha önce tezlerini tamamlamışlardı, ancak Ken DeJong, GA'nın optimizasyondaki yeteneklerini tam olarak sergileyebilen ilk kişi oldu. Holland'ın bir diğer yüksek lisans öğrencisi olan David Goldberg, ilk önce gaz boru hattı optimizasyonuna başvurusu üzerine ödüllü bir doktora tezi hazırladı ve daha sonra 1989'da Arama, Optimizasyon ve Makine Öğrenmesinde GA'lar başlıklı bir tez çalışması yaptı. Hocasıyla birlikte, doğal evrim prensiplerinin optimizasyon problemlerine nasıl uygulanabilir hale getirebileceklerini ve ilk GA'ları nasıl oluşturduklarını anlattı. Holland'ın teorisi böylelikle sıçrama noktasına ulaştı ve sınırlarını aşmaya başladı (Michalewicz, 1996).

3.2. Doğal Seçilim

Türlerin kökeni “uygun varyasyonların korunması ve uygun olmayanların reddedilmesi” üzerine kuruludur. Çeşitlilik, bir türün bireyleri ve aynı ebeveynleri yavrularından ayıran değişiklikleri ifade eder. Hayata gelen her birey hayatta kalmayı başarabilmek adına sürekli bir mücadele içindedir. Avantajlı bireyler, yani en güçlüleri, hayatta kalmak için daha büyük bir şansa sahip olurken nispeten daha güçsüz olanları hayatta kalma mücadelesini kazanamayacaktır. Örneğin, uzun boyunlu zürafa, uzun boylu ağaçlardan ve topraklardan gelen yiyeceklere sahipken, diğer yandan keçi gibi küçük boylu türler ise yalnızca topraklardan gelen yiyeceklere sahiptir. Sonuç olarak, doğal seleksiyon bu hayatta kalma sürecinde önemli bir rol oynar. GA da her yinelemede benzer çizgiler üzerine kurulmuştur, uygun birey hayatta kalır ve uygun olmayan ise ölür. İşlem her yinelemede, kararlı veya optimize edilmiş çözümlere ulaşılabilinen bir aşamaya gelene kadar devam eder ve bu da adapte olabilirlik olarak adlandırılabilir (Sivanandam ve ark. 2008).

3.3. Genetik Algoritma’nın Temel Prensibi

Çözüm popülasyonunun oluşturulması, uygunluk fonksiyonunun bulunması ve genetik operatörlerin uygulanmasından oluşan algoritma işleyişinin temel adımları aşağıda verilmiştir.

- [başlangıç] n bireyli rastgele popülasyon oluşumu
- [uygunluk] popülasyondaki her bireyin f(x) uygunluk değerinin hesaplanması
- [yeni popülasyon] yeni bir popülasyon tamamlanana kadar aşağıdaki adamların tekrarlanarak yeni bir popülasyon oluşturulması
    - [seçilim] popülasyondan uygunluk değerine göre 2 ebeveyn bireyin seçilmesi
    - [çaprazlama] çaprazlama oranı dahilinde yeni çocuklar oluşturmak için ebeveynlerin çaprazlanması
    - [mutasyon] mutasyon oranı dahilinde oluşturulan çocukların mutasyona uğratılması

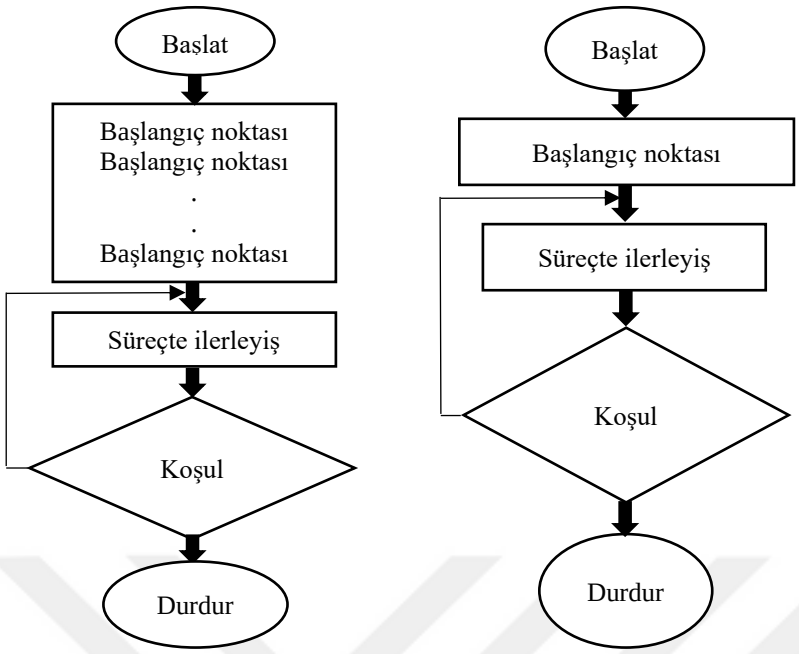
- [onaylama] oluşturulan yeni bireylerin popülasyona yerleştirilmesi

- [yeni bireylerin eskileriyle değiştirilmesi] algoritmanın ileriki fazlarında oluşturulan yeni popülasyonun kullanılması
- [test] sonlandırma kriteri sağlanması ve mevcut popülasyonun en iyi çözümüne geri dönülmesi
- [döngü] uygunluk hesabı için ikinci adıma gidilmesi

Kromozomlarla temsil edilen bir birey popülasyonu oluşturmaları ve bunu sürdürebilmeleri GA'ların ardında yatan temel ilkedir. Bireyin genleri, DNA'da görülen kromozomlara benzeyen birer karakter dizileridir ve tipik olarak bir soruna kodlanmış çözümlerdir. Kromozomlar daha sonra seçilim, çaprazlama ve mutasyon kurallarına göre bir evrim sürecinden geçer. Ortamdaki her birey (bir kromozomla temsil edilir), ortamdaki uygunluğunun bir ölçüsünü alır. Algoritma üreme ile, popülasyonda yüksek uygunluk değerlerine sahip bireyleri seçer ve bu tür bireylerin çaprazlanması ve mutasyonu yoluyla, bireylerin çevrelerine daha iyi uyum sağlamaları için yeni bir popülasyon elde eder. Çaprazlama işlemi, veri parçalarını değiştiren iki kromozom içerir ve doğadaki üreme sürecinin bir benzeridir. Mutasyon ise nüfusun küçük bir kısmına hafif değişiklikler getirir ve evrimsel bir basamağı temsil etmektedir (Sivanandam ve ark. 2008).

3.3.1. Geleneksel ve genetik yaklaşım arasındaki farklılıklar

Algoritma, problemleri çözmek için oluşturulan bir dizi adımlar bütünüdür. GA ise optimizasyon sorunlarına yaklaşık çözümler bulmak için genetiği problem çözme modeli olarak kullanan bir arama tekniğidir. Geleneksel bir algoritma ile genetik bir algoritma arasındaki fark Şekil 3.1'de olduğu gibi kolayca ayırt edilebilmektedir.



(a) GA (b) Geleneksel Yöntem

Şekil 3.1. Geleneksel ve genetik yaklaşımların karşılaştırılması.

3.4. Genetik Algoritma'nın Uygulanması

GA, arama probleminin karar değişkenlerini belirli niceliğe sahip alfabelerin sonlu uzunluktaki dizelerine kodlar. Doğal seleksiyonu uygularken, iyi çözümlerin geliştirilip kötü çözümlerden ayıklandığı fazda, bir takım hesaplamaların yapılması gerekir. Ölçüm, matematiksel bir model veya bilgisayar simülasyonu olan herhangi bir nesnesel işlev olabilir. Uygunluk ölçüsü özünde, daha sonra iyi çözümlerin evrimini yönlendirmek için GA tarafından kullanılacak bir aday çözümün göreceli uygunluğunu belirlemektedir. GA'larda bir diğer önemli kavramı ise popülasyon kavramıdır. Genellikle kullanıcı tarafından belirlenen bir parametre olan popülasyon büyüklüğü, GA'ların ölçeklenebilirliğini ve performansını etkileyen önemli faktörlerden biridir. Örneğin, küçük nüfus boyutları erken yakınsamaya ve standardın altında çözümlere neden olabilirken, büyük nüfus büyüklükleri karmaşık hesaplamalar ile gereksiz süre kaybına

yol açmaktadır. Sorun kromozomsal bir şekilde kodlandıktan ve iyi çözümler kötü olanlardan ayırt edildikten sonra, sorun çözülmeye başlanabilir (Sastry ve ark., 2005).

3.4.1.Başlangıç popülasyonu

GA'ların ana bileşenlerinden ikisi problemi kodlama ve değerlendirme fonksiyonudur. Kâr hedefinin en üst düzeye çıkarılıp, maliyetin en aza indirgendiği bir optimizasyon problemi düşünüldüğünde, amaç bazı çıktıları optimize edebilecek çeşitli parametreleri ayarlamaktır. Daha geleneksel bir dille ifade edilecek olursa, 'f' fonksiyonu en aza indirilmeli veya en üst seviyeye çıkarılmalıdır. Optimizasyon problemleri çözümünde algoritma, optimal çözüm ve asimptotik birleşenlerden oluşan bir dizi hesaplamasıdır. Klasik optimizasyon yöntemlerinin çoğu, fonksiyonun gradyan veya daha yüksek dereceli türevlerine dayanan deterministik hesaplama dizilerinden oluşur. Yöntemlerin arama alanında yalnızca tek bir noktaya uygulanması, yerel optimumda başarısız olma olasılığını da beraberinde getirir. GA, potansiyel çözüm popülasyonunu koruyarak çok yönlü aramalar yapar ve arama aralığını yerel optimumdan uzaklaştırma konusunda başarı göstermektedir. Popülasyon, her nesilde nispeten iyi çözümlerin üretilip, kötü çözümlerin de öldüğü simule edilmiş bir evrim geçirir (Carr, 2014).

İkili rakamlardan oluşan bit dizileri, çözümün kromozom olarak temsil edildiği en yaygın halidir. Bu dizgideki her bit bir geni temsil etmektedir. Çözümü orijinal formundan bit dizisine dönüştürme işlemi de kodlama olarak bilinmektedir. Kullanılan spesifik kodlama şeması tamamen uygulamaya bağlı olmakla birlikte, çözüm bit dizeleri, değerlendirmelerini etkinleştirmek için uygunluk ölçüleri kullanılarak çözülmektedir. Kromozomların yani bireylerin hepsi aynı tip ve aynı uzunlukta olurken popülasyon büyüklüğü de nesil değişiminden etkilenmeksizin sürekli sabit kalmaktadır (Anonim, 2019).

3.5. Kodlama

Kromozomların kodlanması, alternatif olarak kodlanmış alan ve çözüm alanı üzerinde çalışan GA ile bir sorunu çözmeye başlarken sorulacak ilk sorudur.

Kromozomlar gibi kodlanmış alanlar üzerinde gerçekleştirilen genetik işlemler, değerlendirme ve çözüm alanı üzerinde seçim yapmakla yükümlüdür (Gen, 1996). Bir kromozom temsil ettiği çözüm kümesi hakkında mutlaka bazı bilgiler içermelidir. GA'da en çok kullanılan kodlama yöntemleri ikili bit dizileri ve değer kodlama yöntemleridir. Dizideki her bit, çözümün birçok özelliğini temsil edebilir. Başka bir olasılık ise, tüm dizinin bir sayıyı temsil edebilmesidir ve bu da değer kodlama yöntemidir. Kodlamanın başka birçok yolu daha vardır. Kodlamanın nasıl olacağı tamamen çözülecek soruna bağlıdır. Örneğin, kodlama yöntemlerinden biri doğrudan tam sayı veya gerçek sayıları kodlayabilirken; bazıları da permütasyonları ve benzeri ifade biçimlerini kodlayabilmektedir (Samanta, 2016).

GA'larda kodlama yöntemleri seçilirken iki temel prensip izlenir:

- Anlamlı yapı taşları prensibi: Seçilen şema kısa, düşük dereceli ve diğer sabit pozisyonlar üzerindeki şema ile nispeten ilişkili olmalıdır.
- Minimal alfabe prensibi: Kodlamanın alfabesi, çözümlerin doğal olarak temsil edilmesine izin verirken mümkün olduğu kadar da küçük olmalıdır.

Birinci ilke kullanıcının temel problemin yapı taşlarının küçük ve diğer konumlardaki yapı taşlarıyla göreceli olarak ilgisiz olduğu bir kod seçmesi gerektiğidir. Anlamlı yapı taşları ilkesi, şema teoremi tarafından doğrudan harekete geçirilir ve şemalar çok formda, kısa ve düşük sıralı ise, sayıları nesiller boyunca üssel olarak artar. Eğer yüksek kaliteli şemalar uzun veya yüksek dereceli ise, çaprazlama ve mutasyon tarafından bozulurlar ve uygun şekilde çoğaltılamazlar. İkinci ilke, sorunu kullanıcının tanımlanmasına izin veren bir yöntemdir. Kullanıcının en küçük alfabeyi seçmesi gerektiğini, böylece işleme sokulabilir şemaların sayısının en üst düzeye çıkarılması gerektiğini belirtmektedir (Goldberg, 1989). Minimal alfabelerin prensibi, alfabenin niceliğini azaltarak potansiyel şema sayısının arttırılmasını söylemektedir. Minimal alfabe kullanırken, olası şemaların sayısı maksimum olmalıdır. Goldberg'in bitli dize sunumları kullanmasını tavsiye etmesinin nedeni de budur, çünkü niceliği yüksek alfabe kullanırken yüksek kaliteli şemaları bulmak daha zordur. Sorunun niteliğine göre seçilebilecek çeşitli yaygın kodlama türleri aşağıda verilmiştir.

3.5.1. İkili kodlama

Kodlamaların en yaygın olduğu yol, Çizelge 3.1.'deki gibi gösterilebilen ikili bit dizileridir. Her kromozom ikili bir bit dizesini kodlamaktadır. Dizideki her bir bit, çözümün bazı özelliklerini temsil edebilmektedir. Bu nedenle her bit dizesi bir çözümdür, ancak mutlaka en iyi çözüm olmak zorunda değildir. Başka bir olasılık da, tüm dizenin bir sayıyı temsil edebilmesidir. Bit dizelerinin kodlanma şekli, problemden probleme çeşitli farklılıklar göstermektedir.

İkili kodlama, az sayıda allel geni olan birçok olası kromozom içerebilmektedir. Diğer taraftan, çoğunlukla 0 ve 1'leri içeren bu ikili kodlama, birçok çözüm kümesi için doğal olmamakla birlikte, bazı genetik işlemlerin tamamlanmasından sonra bir takım düzeltmelerin yapılmasını gerektirebilmektedir ve dizelerin doğruluğu uzunluğuna bağlıdır. Bu kodlamada, tamsayılar tam bir şekilde temsil edilirken, gerçek sayılar sınırlı sayıda temsil edilir ve temsil edilen gerçek sayıların sayısı dize uzunluğu ile doğru orantılı olarak artar (Samanta, 2016).

Çizelge 3.1. İkili kodlama

1.Kromozom	1100011101110010
2.Kromozom	0110010101110011

3.5.2. Değer kodlama

Bit değerlerinden oluşan dizilerin kromozomlarla temsil edildiği bu kodlama yöntemi diğer kodlama yöntemlerine kıyasla Çizelge 3.2.'te gösterildiği gibi çok daha geniş bir çözüm yelpazesi sunmakta ve özellikle bazı özel problem türleri için çok iyi sonuçlar verebilmektedir (Samanta, 2016). Problemin doğasına göre yeni genetik operatörlerin geliştirilmesinin mümkün kılındığı değer kodlama, tam sayılar, gerçek sayılar veya karmaşık sayılar gibi birçok karmaşık matematiksel değerlerin kullanıldığı problemlerde oldukça kullanışlıdır çünkü bu tür problemler ikili kodlamanın baş edemediği içinden çıkılması zor olan döngü ve hesaplamaların kolaylıkla idare edebilmektedir (Haupt ve Haupt, 1998).

Çizelge 3.2. Değer kodlama

1.Kromozom	2.125 0.2398 8.0127
2.Kromozom	AFJBADCEHISTK
3.Kromozom	(geri)(sağ)(sol)(ileri)

3.5.3. Sekizli kodlama

Bu kodlama, Çizelge 3.3.'te gösterildiği gibi sekizli sayılardan (0–7) oluşan bir dizi kullanır. Bu kodlama türünün ikili kodlamaya göre temel avantajı daha küçük kodlama şemasına sahip olmasıdır (Haupt ve Haupt, 1998).

Çizelge 3.3. Sekizli kodlama

1.Kromozom	20346151
2.Kromozom	12670231

3.5.4. On Altılı kodlama

Bu kodlama, Çizelge 3.4.'te gösterildiği gibi onaltılık sayılardan (0–9, A – F) oluşan bir dize kullanır. Bu kodlama şemasının ikili ve sekizli kodlamaya göre avantajı daha küçük boyutta olmasıdır (Haupt ve Haupt, 1998).

Çizelge 3.4 On altılı kodlama

1.Kromozom	A09B
2.Kromozom	932F

3.5.5. Gray kodlama

Değişken değerlerin ikili sayı gösterimi, GA'nın yakınsaklığını yavaşlatabilir. Değişken gösterimindeki bit sayısının arttırılması sorunu büyütmektedir (Haupt ve Haupt, 1998). Gray kodlama, ikili sayıları yeniden tanımlayarak ardışık sayıları bir kod

mesafesine sahip olacak şekilde düzenleyerek bu sorunu önleyebilmektedir (Taub ve Schilling, 1986). Gray kodları, algoritmanın çözüme yaklaşırken yakınsama süresini dikkat çekecek seviyede hızlandırır. Gray kodlama, çözümleri bulmak için dörtlü arama metodunu kullanır. İkili dizelerde olduğu gibi, gray kodlu dizelerde herhangi bir değişiklik yapılması, kodu çözülmüş tamsayı değerinde büyük bir değişikliğe uğratabilmektedir. Gray kodlu dizilerin karşılık geldiği değişken kodların çözülmesi, dizi ve kodu çözülen değer arasındaki ilişkide doğrusal olmayan bir yapaylığı ortaya çıkarmaktadır.

3.5.6. Permütasyon kodlama

Her kromozom, Çizelge 3.5.'de gösterildiği gibi sayıların sırayla temsil edildiği bir sayı dizisinde toplamaktadır. Genetik işlem tamamlandıktan sonra bazen birtakım düzeltmelerin yapılması gerekebilmektedir. Permütasyon kodlamada her kromozom, bir dizideki sayıyı temsil eden bir tamsayı dizisinden oluşmaktadır. Permütasyon kodlama, sadece sipariş verme sorunları için kullanışlıdır. Bu problemler için bile bazı geçit tipleri ve kromozomun tutarlı bırakılması için mutasyon düzeltmeleri yapılmalıdır (Starkweather ve ark., 1991).

Çizelge 3.5. Permütasyon kodlama

1.Kromozom	142739865
2.Kromozom	231937458

3.5.7. Ağaç Tipi kodlama

Genetik programlama için sıklıkla kullanılan bir diğer kodlama çeşidi ise ağaç kodlamadır. Her bir kromozomun şifrelendiği ağaçlarda, fonksiyonlar veya komutlar gibi bazı nesnesel ağaçlar vardır. Dizi işlevleri ve uç noktalar tanımlandıktan sonra ağaçlardaki düğümleri etiketlemek için analitik işlevleri yerine getiren bir takım zengin gösterimler sağlanır. Ayrıştırma ağacı, dinamik olarak boyutlandırılmış yapılara izin veren ve doğal özyinelemeli tanımları içeren popüler bir yöntemdir (Back ve ark., 1996).

Ayrıştırma ağacı temsillerinin çoğunun, değişen programların boyutuyla ilgili kısıtlamaları içermektedir. Bir ayrıştırma ağacı, içerik, gösterim gücü ve uygunluk belirlemekle yükümlüdür. Ayrıştırma ağaçlarının asiklik yapıları nedeniyle, yinelemeli hesaplamaların gösterilmesi pek de mümkün değildir ve durma kriterlerini belirlemek de oldukça zordur. Bu nedenle gelişmiş işlev, önceden belirlenmiş bazı durma kriterleri yerine getirilinceye kadar, yeniden belirlenen dolaylı bir döngü içinde değerlendirilir.

Ağaç kodlaması, arama alanının açık uçlu olmasına izin vermektedir. Ancak bu da ağacın kontrolsüz bir şekilde büyüyebilmesine neden olmaktadır. Büyük ağaçlar yapı ve hiyerarşik nicelikleri önlemekle birlikte, anlaşılması ve basitleştirilmesi zor olan kodlama kriterleridir. Ayrıştırma ağacı, dinamik olarak boyutlandırılmış yapılara izin veren doğal özyinelemeli tanımları içermektedir. Ayrıştırma ağacı temsillerinin çoğu değişen programların boyutuyla ilgili bazı kısıtlamalar içermektedir. Eğer herhangi bir kısıtlama yoksa, o zaman gelişen programların boyutunun artması ve daha sonra mevcut hesaplama kaynaklarının kullanılması muhtemeldir (Angeline, 1996).

3.6. Uygunluk Fonksiyonu

Genetik uygunluk fonksiyonu, algoritmadaki bir çözümün optimizasyonunu belirten, belirli bir kromozomun diğer tüm kromozomlara göre sıralanabilmesi için gerekli olan nesnel bir işlev türüdür. Optimal kromozomların veya en azından en uygun kromozomların veri kümelerini birkaç teknikten herhangi biriyle üreyip karıştırmasına izin verilir, bu da daha iyi optimize olmuş yeni bir popülasyon oluşturur. İdeal bir uygunluk fonksiyonu algoritmanın hedefiyle yakından ilişkilidir ve hızlı bir şekilde hesaplanabilir. Bu tür kodlamalarda uygulama hızı çok önemlidir, çünkü tipik bir GA herhangi bir probleme kullanılabilir bir sonuç üretmek için birçok kez yinelenmelidir. Bu da GA'ların gerçek dünya uygulamalarındaki ana dezavantajlarından biridir ve bazı endüstrilerde uygulanabilirliklerini sınırlamaktadır.

Yaklaşık modeller özellikle aşağıdaki durumlarda en çok umut vaat eden yaklaşımlardan olmaktadır;

- Tek bir çözümün uygunluk hesaplama süresinin oldukça yüksek olduğu durumlarda,

- Uygunluk hesaplaması için gerekli bir modelin eksik olduğu durumlarda,
- Uygunluk fonksiyonun belirsiz veya karmaşık olduğu durumlarda.

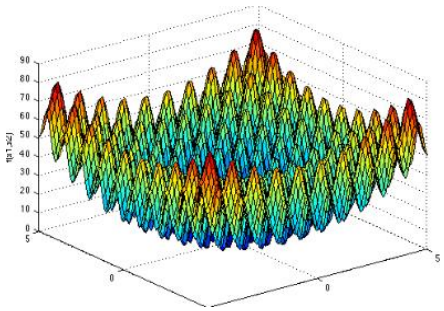
Uygunluk fonksiyonlarına bakmanın bir başka yolu, her olası kromozom için uygunluğu gösteren bir spor modudur. Uygunluk fonksiyonunun tanımı pek çok durumda karmaşıktır ve GA tarafından üretilen en uygun çözümler sıklıkla tekrarlanır şekilde yapılır. Bazı durumlarda, uygunluk fonksiyon tanımının ne olabileceğine dair bir tahminde bulunabilmek çok zor veya imkansızdır. İnteraktif GA'lar bu karmaşıklığa bir dereceye kadar yanıt vermektedir (DeJong, 2006).

3.7. Optimizasyon Test Fonksiyonları

Yapay tabiat olarak da bilinen test fonksiyonları, yakınsama oranı, hassaslık, sağlamlık ve genel performans gibi optimizasyon algoritmalarının özelliklerini değerlendirmek için kullanılmaktadır. Bu bölüm, bu çalışmada kullanılan test fonksiyonları ve bu fonksiyonların GA'daki performansları hakkında bilgi vermektedir.

3.7.1. Rastrigin test fonksiyonu

Matematiksel optimizasyonda rastrigin fonksiyonu, optimizasyon algoritmaları için performans test problemleri olarak kullanılan ve dışbükey olmaktan uzak bir fonksiyon olmasından dolayı oldukça kullanışlıdır. Doğrusal olmayan bir fonksiyon olan Rastrigin, birkaç yerel minimuma sahip olmakla birlikte, çok modüllü bir fonksiyondur. Birden fazla yerel minimum değerlerine sahip olması, noktaların düzenli olarak dağılmasına engel olmamaktadır (Bingham, 2017). Ancak devasa arama alanlarına ve çoklu genel minimun değerlerine sahip olması fonksiyonun minimumu bulurken kısmen zorlandığını göstermektedir (Mühlenbein ve ark., 1991). Şekil 3.2'de 2 boyutlu olarak gösterilmiştir. Genel minimum değeri 0'dır ve d boyutludur.

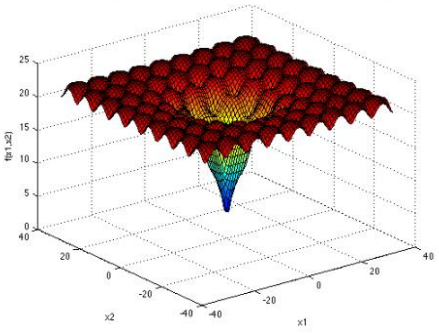


Şekil 3.2. Rastrigin test fonksiyonu.

$$f(x) = 10d + \sum_{i=1}^{d}[x_i^2 - 10\cos(2\pi x_i)] \quad (3.1)$$

3.7.2. Ackley test fonksiyonu

Ackley test fonksiyonu, optimizasyon algoritmalarını test etmek için yaygın olarak kullanılmaktadır. Şekil 3.3'te gösterildiği gibi iki boyutlu formda, neredeyse düz bir dış bölge ve merkezde büyük bir delik ile karakterize edilir (Bingham, 2017). Yerel minimum değeri 0'dır. Bu fonksiyon, optimizasyon algoritmaları için birçok yerel minimumundan birinde sıkışıp kalma riski taşımaktadır.

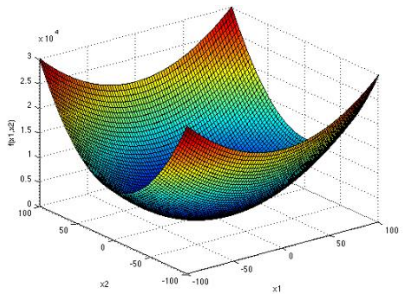


Şekil 3.3. Ackley test fonksiyonu.

$$f(x) = -a\exp\left(-b\sqrt{\frac{1}{d\sum_{i=1}^{d}x_i^2}}\right) - \exp\left(\frac{1}{d\sum_{i=1}^{d}\cos(cx_i)}\right) + a + \exp(1) \quad (3.2)$$

3.7.3. Bohachevsky test fonksiyonu

Bohachevsky fonksiyonlarının neredeyse hepsi kâse biçimine sahiptir. Yerel minimum değeri 0'dır ve Şekil 3.4'teki örnek grafik iki boyut için verilmiştir.

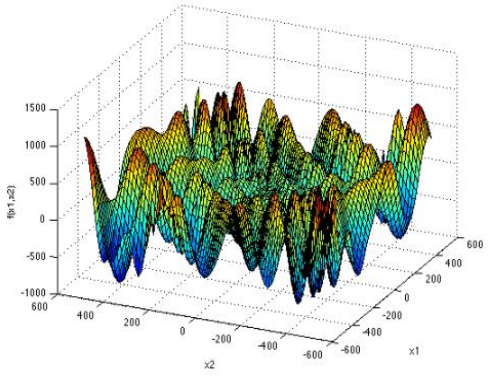


Şekil 3.4. Bohachevsky test fonksiyonu.

$$f(x) = x_1^2 + 2x_2^2 - 0{,}3\cos(3\pi x_1) - 0{,}4\cos(4\pi x_2) + 0{,}7 \quad (3.3)$$

3.7.4. Eggholder test fonksiyonu

İki boyutlu gösterimi Şekil 3.5'deki gibi olan Eggholder test fonksiyonu, çok sayıda yerel minimuma sahip olması nedeniyle optimize edilmesi zor bir fonksiyondur. Yerel minimum değeri -959.6407'dir.

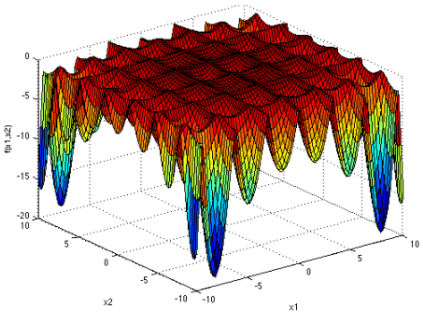


Şekil 3.5. Eggholder test fonksiyonu.

$$f(x) = -(x_2 + 47)\sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1\sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right) \quad (3.4)$$

3.7.5. HolderTable test fonksiyonu

Holdertable, dört genel minimuma ve birçok yerel minimuma sahiptir. Genel minimum değeri -19.2085'tir, iki boyutlu gösterimi Şekil 3.6'daki gibi olmaktadır.



Şekil 3.6. Holdetable test fonksiyonu.

$$f(x) = -|\sin(x_1)\cos(x_2)\exp(\left|1 - \sqrt{(x_1^2 + x_2^2)/\pi}\right|)| \quad (3.5)$$

4. GENETİK OPERATÖRLER

Genetik operatör, genetik çeşitliliğin sağlanması ve gen tekrarının önlenmesi açısından GA'larda kullanılması bir bakıma şart operatörlerdir. Genetik çeşitlilik, evrim sürecinin bir gereğidir. GA'larda kullanılan, en güçlülerin hayatta kalması olarak belirtilen seçilim (Yadav, 2017); üreme (çaprazlama, rekombinasyon olarak da bilinir); ve mutasyon gibi genetik operatörler, doğal dünyada meydana gelen olayların birer simülasyonudur. Biyoçeşitlilik seviyesi olarak da bilinen genetik çeşitlilik, bir türün genetik yapısındaki toplam genetik nicelik sayısını ifade etmektedir.

GA sürecinde ilerleme kat edildiğinde, arama sahasında keşif ve sömürü arasında bir denge sağlamak için arama fonksiyonunun optimum çözüm olarak kabul edilmesi gerekmektedir. GA araştırmalarından birikmiş bilgilerin kullanılması, seçilim mekanizması tarafından yapılırken, araştırma alanının yeni bölgelerine yapılan araştırmalar ise genetik operatörler tarafından gerçekleştirilmektedir. Genetik operatörler, her nesilde yeni yavrular oluşturmak için genlerin kalıtım sürecini taklit etmektedir. Operatörler, temsil sırasında bireylerin genetik kompozisyonunu değiştirmek için kullanılır ve temelde rastgele aramalar yaptığı halde gelişmiş bir yavru vermeyi garanti edemez. Seçilim, çaprazlama ve mutasyon olmak üzere üç ana operatörden oluşan genetik operatörlere aşağıda geniş bir bakış açısı sağlanmıştır.

4.1. Seçilim

Seçilim operatörü, daha sonraki çaprazlama fazı için nüfustan iki veya daha fazla ebeveynin seçilim sürecidir. Kodlama yöntemine karar verildikten sonraki ilk adım, seçilimin nasıl yapılacağına, yani, gelecek nesil için yavru yaratacak bireylerin nasıl seçileceğine ve her birinin kaç yavru yaratacağına karar vermektir. Seçilimin amacı, doğal seleksiyon sürecinin daha yüksek uygunluğa sahip olması umuduyla popülasyondaki nispeten daha üstün bireylerin seçilmesidir. Başka bir deyişle popülasyonun büyüklüğünü korurken iyi bireyleri bir sonraki nesle yaymaktır (Yadav, 2017). Kromozomlar, ilk popülasyondan üreme için ebeveyn olarak seçilir. Sorun, bu

kromozomların nasıl seçileceğidir. Darwin'in evrim teorisine göre, en iyileri yeni yavrular yaratmak için hayatta kalır (Goldberg, 1989).

Seçilim, değerlendirme işlevlerine göre kromozomları popülasyondan rastgele alan bir yöntemdir. Bir bireyin seçilim şansı, uygunluk değeri ne kadar yüksek olursa o kadar fazladır. Seçilim baskınlığı, daha iyi bireylerin tercih edilme derecesi olarak da tanımlanılabilir. Seçilim baskınlığı ne kadar yüksek olursa, yüksek uygunluk değerine sahip bireylerin tercih edilme ihtimali de o kadar fazladır. Böylelikle seçilim baskınlığı, GA'yı art arda gelen nesillerdeki popülasyon uygunluğunu daha iyi noktalara taşımaktadır.

GA'nın yakınsama hızı, yakınsama oranları ve seçilim baskınlığı ile belirlenir. GA'lar, geniş bir seçilim baskınlık şeması altında optimal veya optimala en yakın çözümleri tanımlayabilmelidir. Bununla birlikte, eğer seçilim baskınlığı çok düşükse, yakınsama oranı yavaş olacaktır ve GA, optimum çözümü bulmak için gereksiz yere zaman harcayacaktır. Eğer seçilim baskınlığı çok yüksekse, zamanından önce yanlış bir çözüme yaklaşan GA'da birtakım değişiklikler meydana gelmektedir. Seçilim baskınlığı sağlamanın yanı sıra, seçilim şemalarındaki erken yakınsamayı önlemeye yardımcı olduğu için popülasyon çeşitliliğini de korumaktadır (Sivanandam ve ark., 2008).

Tipik olarak orantılı seçilim ve sıra tabanlı seçilim olmak üzere iki tip seçilim şeması vardır. Oran temelli seçilim, bireyleri, popülasyon içindeki diğer bireylerin uygunluğuna göre uygunluk değerlerini esas alarak seçmektedir. Sıralama tabanlı seçilim, bireyleri kendi ham formlarına göre değil, nüfus içindeki sıralarına göre seçer. Bu da seçilim baskınlığının, popülasyonun uygunluk dağılımından bağımsız olmasını ve yalnızca popülasyonun göreceli sıralamasına dayanmasını gerektirmektedir. Seçilim baskınlığının uyarlanması ve popülasyonun uygunluk değer aralığına göre yeniden derecelendirilmesi için bir ölçeklendirme operasyonu kullanılması gerekmektedir. Çok güçlü bir seçilim, en uygun bireylerin nüfusu ele geçireceği, değişim ve ilerleme için gereken çeşitliliği azaltacağı; çok zayıf seçilim ise, evrimin çok yavaş gerçekleşeceği anlamına gelmektedir. Sıklıkla kullanılan çeşitli seçilim yöntemleri aşağıda verilmiştir.

4.1.1. Rulet çarkı

Rulet çarkı, en yaygın KGA seçilim tekniklerinden biridir. Bu üreme operatörü, üreme havuzundan uygunlukla orantılı bir dizenin seçildiği, orantılı bir üreme operatörüdür. Rulet seçiliminin prensibi, rulet çarkında bulunan ve bireyin uygunluk değerleri ile orantılı olarak tartılmış bir çark üzerinden yapılan doğrusal bir aramadır. Popülasyondaki bireylerin uygunluklarının toplamının rastgele bir hedef değer oranı belirlenir. Popülasyon hedef değere ulaşana kadar sürece devam edilir. Bu, sadece orta derecede güçlü bir seçilim tekniğidir, çünkü uygun bireylerin seçilmesi garanti edilmez. Yüksek uygunluk değerine sahip bir birey hedef değere daha fazla katkıda bulunacaktır, ancak bunu gerçekleştiremezse, sıradaki kromozomun tek bir şansı vardır ve bu kromozom düşük uygunluk değerine sahip zayıf bir birey de olabilir. Popülasyonun uygunluk değerine göre sıralanmaması esastır, çünkü bu seçilimi önemli ölçüde ön yargılı hale getirecektir.

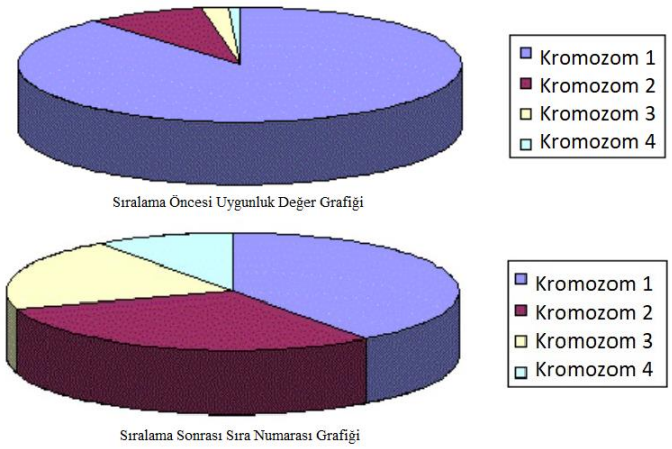
Bir veya daha fazla bireyin birden fazla kez seçilmesinin mümkün olması, geliştirmeye elverişli bazı tekniklerin uygulanmasının kolay olması ve her bireye adil şansın verilip çeşitliliğin korunması dolayısıyla önyargısız bir teknik olması bakımından avantajlara sahip bir seçilim tekniğidir. Ancak, baskın olan ve her zaman seçilen bireylerin varlığından dolayı erken yerel minimuma ulaşılması ve en uygun bireyin çarkta daha çok yer kaplayarak diğer bireylerin seçilim şansını nispeten azaltması bakımından birtakım dezavantajlara da sahiptir (Yadav, 2017).

4.1.2. Rastgele seçilim

Bu seçilim tekniği popülasyondan rastgele bir ebeveyn seçerek seçilim işlemine başlar. Rastgele seçilim genetik kodların bozulması açısından rulet çarkından ortalama olarak daha kötü sonuçlar vermektedir.

4.1.3. Sıralı seçilim

Sıralı seçilim, bu teknikteki seçilim olasılığının mutlak uygunluktan ziyade uygunluk ile orantılı olması dışında rulet çarkına benzer. Bu teknikte bir bireyin seçilim olasılığı, uygunluktan çok bir rütbe işlevidir. Ancak rulet seçiliminde uygunluk değerinin çok değişken olduğu durumlarda ciddi sorunlar yaşanacaktır. Örneğin en iyi kromozom uygunluğu tüm rulet çarkının %90'ı ise, diğer kromozomlar seçilim için çok az şansa sahip olacaktır. Sıralı seçim ilk önce popülasyonu sıralar, ardından her kromozoma bu sıralamadan uygunluk değeri atar. Sıralı seçilim iki aşamalı bir süreçten oluşmaktadır. İlk önce bireylerin listesi sıralanmalı ve daha sonra bireylere rütbelerin atanması gerekir. En kötü uygunluğa sahip birey 1.uygunluk, ikinci en kötü 2.uygunluk vs. olacak şekilde sıralanır ve en iyi birey popülasyondaki en iyi uygunluk değerine sahip olacaktır. Şekil 4.1'de sıra numaralarındaki uygunluk derecelerinin, algoritmayı nasıl etkilediği gösterilmiştir.



Şekil 4.1. Sıralı seçilim öncesi ve sonrası.

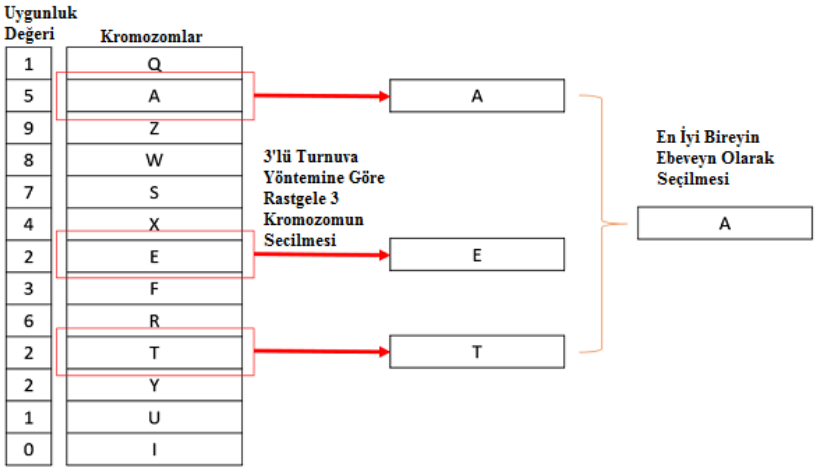
Sıralı seçilim çeşitliliği korurken ön yargısız olmayı da başarabilen bir seçilim yöntemidir. Ancak daha yavaş yakınlaşmaya neden olması nedeniyle iyi bireyler arasındaki farkın az olması ve karmaşık bir hesaplama sistemine sahip olması bakımından bazı dezavantajları da beraberinde getirmektedir (Yadav, 2017).

4.1.4. Turnuva seçilimi

Turnuva seçiliminde, popülasyondaki her birey diğer bireyler ile rastgele eşlenir ve her bir çiftin uygunluk değerleri karşılaştırılır. Eşleştirilen çift bireylerden uygunluğu daha iyi olan bir sonraki aşamaya alınırken diğerleri elimine edilir (Miller ve Goldberg, 1995). Bu süreç turnuvayı kazanan birey sayısı hedefteki ebeveyn sayısına ulaşana kadar devam eder ve son kazanan grup yeni bireylerin ebeveyni olarak eşleştirilir (Shah ve Chudasama, 2011). Bu seçilim tekniğinde seçilim baskınlığı, turnuva büyüklüğüne göre kontrol edilir. Turnuvanın büyüklüğü popülasyonun büyüklüğüne eşitse, kazanan her zaman aynı olacaktır (Yadav, 2017).

GA arama performansının ideal seçilim stratejisinin hassas bir şekilde ayarlanabilmesi için seçilim baskınlığı ve popülasyon çeşitliliği ölçeklendirilebilir olmalıdır. Rulet çarkından farklı olarak, turnuva seçilim stratejisi, belirli bir sayıdaki birey arasında bir turnuva düzenler. Turnuvanın en iyi bireyi turnuvanın galibi ve en yüksek uygunluğa sahip olan bireydir. Turnuvayı kazanan birey, daha sonra çaprazlama havuzuna yerleştirilir ve yeni yavrular üretmek için havuz dolana kadar bu prosedür tekrar edilir. Böylelikle turnuvanın galiplerinden oluşan çaprazlama havuzunun uygunluk değer ortalaması giderek yükselecektir.

Turnuva boyutunun büyük olduğu durumlarda kazanan birey hep aynı olacağından çeşitlilik kaybolsa da popülasyonun sıralanma kriterine tabi tutulmadan çeşitliğin korunması ve paralel olarak uygulanabilmesinden dolayı bu yöntem bütün seçilim türleri içindeki en verimli ve en uygun çözüme sahip olan seçilim türüdür. Turnuva seçiliminin temsili bir şeması Şekil 4.2’de verilmiştir.



Şekil 4.2. Üçlü turnuva yöntemi.

4.1.5. Elitizm seçilim

Elitizm fikri, bir popülasyonun en iyi bireyini korumaktır çünkü çaprazlama ve mutasyon operatörü sayesinde en iyi bireylerin kaybedilme riski mevcuttur. Bu seçilim metodunun amacı sadece en iyi kişiyi korumak değil, aynı zamanda en iyi özellikleri gelecek nesillere yaymak için bu iyi özelliklerin üremeye katılmasını sağlamaktır (Yadav, 2017).

4.2. Çaprazlama

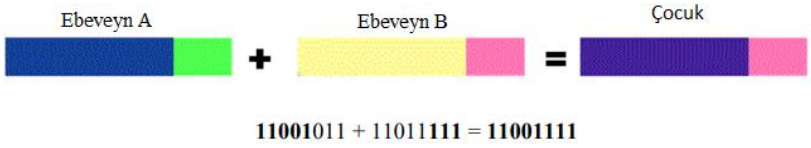
GA'larda gen ve bireylerin rekombinasyonu için kullanılan yöntem olan çaprazlama, iki ebeveynin genetik bilgilerini kullanarak yeni yavrular üretmek için kullanılan genetik bir operatördür. GA'da en iyi çözümleri veya yavruları oluşturmak için kullanılan temel strateji, ana genleri çaprazlamaktır. Evrimsel hesaplamadaki birçok farklı algoritma, genetik bilgileri depolamak için farklı veri yapılarını kullanabilir ve her genetik temsil, farklı çaprazlama operatörleri ile yeniden birleştirilebilir (Tian ve ark., 2018). Bu operatörde, algoritma önce ilk iki ebeveyn bireyi seçer ve seçilen bireyleri belirlenen çaprazlama işlemine tabi tutarak özel gen varyasyonuna göre iki çocuk yaratır. Daha önce belirlenmiş olan ebeveyn ve popülasyon sayısına göre yeni bireyler yaratılmaya devam edilir. Başka bir deyişle çaprazlama, problemin çözümünde kullanılan

kodlama şemasına bağlı olarak bireylerden seçilen genler üzerinde çalışır ve yeni yavrular yaratır (Gwiazda, 2006).

Çaprazlamayı gerçekleştirmenin en basit yolu, rastgele bazı çaprazlama noktaları seçmek ve bu noktadan önceki en iyi birey ve genleri kopyalamaktır. Daha sonra seçilen çaprazlama noktasından sonraki bütün bireyleri kopyalanan bir önceki birey ve genlerle değiştirmektir. K noktalı çaprazlama, tek ve çift noktalı çaprazlama ve aritmetik çaprazlama gibi birçok çaprazlama metodu vardır. Çaprazlama oldukça karmaşık olabildiği gibi, temel olarak kromozomların kodlanmasına bağlıdır (Beasley ve ark., 1993). Evrimleşme sürecinin ilk kuşaklarında, mümkün olan en iyi çözümden iyi şekilde faydalanmak için çeşitli çaprazlama teknikleri geliştirilmiştir. Çaprazlama yönteminin seçilimi, GA'nın performansı üzerinde oldukça etkilidir çünkü erken yakınsamanın önüne uygun üreme operatörleri seçilerek geçilebilir (Umbarkar ve Sheth, 2015). Günümüzde çok kullanılan çaprazlama yöntemleri aşağıda verilmiştir.

4.2.1. Tek Noktalı çaprazlama

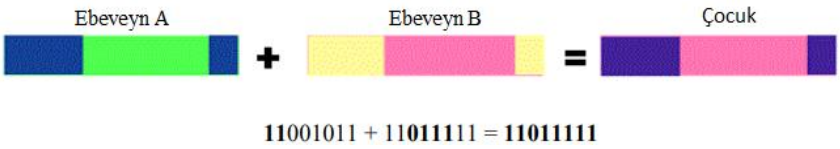
Uygulaması en basit çaprazlama metotlarından biri olan tek noktalı çaprazlama, eşleşen iki kromozomun karşılık geldiği noktalarda bir kez kesilen ve bu bölümlerin tek nokta ile çaprazlamaya uğratıldığı çaprazlama yöntemidir. Bu çaprazlama yönteminde, ebeveynlerin kromozomları parçalanırken tek bir nokta kullanır ve daha sonra yavruları oluşturmak için ebeveynler bu çaprazlama noktalarından birleştirir (Umbarkar ve Sheth, 2015). Eşleştirilen dizilerin uzunluğu boyunca rastgele bir çapraz nokta seçilmekte ve çaprazlama noktalarının yanındaki değerler değiştirilmektedir. Çaprazlama noktaları rastgele seçilmektedir, eğer uygun yer seçilirse, birleştirilen ebeveynler daha iyi genlere sahip olacak ve dolayısıyla elde edilen çocuklar daha optimum değerlere sahip olacaktır. Şekil 4.3'te tek noktalı çaprazlama gösterilmektedir ve çaprazlama noktalarının yanındaki bitlerin çocuk üretmek için nasıl değiştirildiği gösterilmektedir (Soni ve Kumar, 2014).



Şekil 4.3. Tek noktalı çaprazlama.

4.2.2. Çift Noktalı çaprazlama

Tek nokta çaprazlamadan sonra birden fazla kesme noktası içeren birçok farklı çaprazlama algoritması geliştirilmiştir. Ancak fazladan eklenen çaprazlama noktalarının eklenmesi GA performansını düşürmektedir. Ek çaprazlama noktalarının eklemesinin dezavantajı, gen yapı taşlarındaki bozulma ihtimalinin daha yüksek olmasıdır. Sorunlu alanlarda daha ayrıntılı aramaların yapılabilmesi çoklu çaprazlama noktalarına sahip olmanın getirdiği avantajlardandır. Çift noktalı çaprazlamalarda, iki çaprazlama noktası seçilir ve bu noktalar arasındaki içerik Şekil 4.4'te gösterildiği gibi eşlenmiş iki ebeveyn arasında değiştirilir. Tek noktalı çaprazlama ile ilgili temel problem, bir kromozomun başının ve kuyruğunun aynı anda yavrulara geçememesidir. Bir kromozomun hem başı hem de kuyruğu iyi genlerden oluşuyorsa, tek noktalı çaprazlama ile elde edilen yavruların hiçbiri iki iyi özelliği aynı anda paylaşamamaktadır. Tek noktalı çaprazlamanın geliştirilmiş hali olan ve ondan daha iyi kabul edilen bu çaprazlama metodunun kullanımıyla, bahsi geçen olumsuzluk ortadan kaldırılmıştır (Umbarkar ve Shelth, 2015).



Şekil 4.4. Çift noktalı çaprazlama.

4.2.3. Çok noktalı çaprazlama (k-noktalı çaprazlama)

Tek noktalı çaprazlamada bahsedilen sorunlar iki noktalı çaprazlamada da ortaya çıkabilmektedir ve bu problem aslında bir kromozomdaki her gen pozisyonuna

genelleştirilebilir. Bir kromozomda nispeten birbirine yakın olan genlerin elde edilen yavrulara hep birlikte geçme şansı K-noktalı çaprazlamayla daha fazla olacağında bu sorunlar da önüne geçilmiş olunacaktır. Sonuç olarak, K-noktalı bir çaprazlamanın algoritma içinde etkinliği, kromozom içindeki genlerin pozisyonuna bağlı olacaktır. Ebeveynleri birleştirmek için tek ve çift noktalı çaprazlamalarda olduğu gibi rastgele çapraz nokta kullan K-noktalı çaprazlama, ebeveynlerin mümkün olan en iyi birleşimini sağlamak ve yavru oluşturmak için birden fazla çaprazlama noktası seçer. Genetik varsayımda, çözümlerin bağımlı özelliklerini kodlayan genler birbirine yakın olacağından, seçilen bu noktalar da birbirine yakın olmalıdır (Umbarkar ve Sheth, 2015).

4.2.4. Tekdüze çaprazlama

Tekdüze çaprazlama, yavrudaki her gene karşılık gelen genin, kromozomlarla aynı uzunlukta rastgele oluşturularak ikili çaprazlama maskesine göre seçilen ebeveynden kopyalanmasıyla oluşturulmasından dolayı diğer çaprazlama metotlarından oldukça farklıdır. Çaprazlama maskesinde değer "1" olduğu durumlarda, gen birinci ebeveynden kopyalanırken, maskede değer "0" olduğunda, gen ikinci ebeveynden kopyalanır. Her bir ebeveyn çifti için rastgele yeni bir çaprazlama maskesi oluşturulur. Yavrular bu nedenle her bir ebeveynden birer gen karışımı içerir (Umbarkar ve Sheth, 2015).

4.2.5. Aritmetik Çaprazlama

Aritmetik çaprazlamada, ağırlıklı aritmetik ortalaması alınmış iki ebeveynden meydana gelen iki çocuk oluşturulmaktadır. Çocukların oluşumu, belirli doğrusal kısıtlamalar doğrultusunda olmaktadır. Alfa (a), [0,1] arasında rastgele herhangi bir değer olup oluşturulan çocuklar uygunluk değeri daha iyi olan ebevne yakın karakteristikte olmaktadır (Kaya ve ark., 2011).

$$1.çocuk = a * 1.ebeveyn + (1 - a) * 2.\ ebeveyn$$
$$2.çocuk = (1 - a) * 1\ ebeveyn. + a * 2.\ ebeveyn \quad (4.1)$$

4.3. Mutasyon

Mutasyon, çeşitli kromozomlarda rastgele değişiklikler yaparak popülasyondaki gen çeşitliliğini sağlayan genetik bir operatörüdür. Mutasyonu elde etmenin en basit yolu, bir veya daha fazla geni değiştirmek olacaktır. GA'da mutasyon, seçilim sürecinde popülasyondan elimine edilen genlerin tekrar denenebilmeleri için değiştirilmelerini ya da ilk popülasyonda bulunmayan genlerin oluşturulması için kritik bir rol oynar (Srinivas ve Patnaik, 1994). Mutasyon olasılığı, popülasyona yeni genlerin dahil edilme olasılığını kontrol eder ve popülasyondaki toplam gen sayısının yüzdesi olarak tanımlanmaktadır. Çok düşük olduğu durumlarda faydalı olabilecek birçok genin oluşturulma olasılığı oldukça az olacakken, çok yüksek olduğu durumlarda ise rastgele bozulma çok yüksek oranlarda olacak ve yavrular ebeveynlerine olan benzerliklerini gittikçe yitirerek algoritmanın amacının dışına çıkmasına sebep olacaktır (Knuth, 1997).

İkili ifade biçimi ile oluşturulmuş ve mutasyona uğratılacak olan bireylerin genlerinde bulunan 0'ların 1'lere, 1'lerin ise 0'laraa çevrildiği mutasyon biçimi olan çevirmeli mutasyon, bir bireyin kromozomunda bulunan genlerden ikisinin yerinin değiştiği takas mutasyonu ve rastgele bir pozisyonun seçildiği ve sonrasında bu pozisyonun yanındaki genlerin tersine çevrildiği çevirme mutasyonu gibi yaygın mutasyon çeşitleri aşağıda verilmiştir.

4.3.1. Ekleme mutasyonu

Her şeyden önce, birbirini takip eden rastgele iki allel gen değerinin seçildiği, daha sonra ilkini takip etmek üzere hareket ettirilen ikinci genin, kromozomun geri kalanına uyum sağlaması için kaydırıldığı mutasyon çeşitli olan ekleme mutasyonu, permütasyon kodlamalarında sıklıkla kullanılır (Soni ve Kumar, 2014).

4.3.2. Evirme mutasyonu

Evirme mutasyonu, permütasyon kodlamasıyla oluşturulmuş kromozomlar için kullanılmaktadır. Evirmenin gerçekleştirilmesi için rastgele iki gen seçilir ve daha sonra

aralarında kalan alt diziler mutasyona uğratılarak evrim gerçekleştirilir (Sivanandam ve Deepa, 2008). Bu mutasyon yöntemi birçok bitişik bilginin saklandığı alt diziyi iki noktadan keserek sipariş bilgilerinin bozulmasına yol açabilmektedir (Soni ve Kumar, 2014).

4.3.3. Çırpma mutasyonu

Çırpma mutasyonu da yukarıdaki mutasyon yöntemleri gibi permütasyon kodlamasıyla oluşturulan kromozomlar için kullanışlı bir diğer mutasyon yöntemidir. Bu mutasyonda, rastgele bir gen kümesi seçilir ve seçilen bu kümedeki genlerin değerleri bazı değişikliklere uğrayarak düzenlenir ve böylelikle mutasyon gerçekleşmiş olur. Bu mutasyonda dikkat edilmesi gereken şey ise kümelerin bitişik olması gerekmediğidir (Soni ve Kumar, 2014).

4.3.4. Takas mutasyonu

Birçok kodlama yöntemiyle verimli bir şekilde çalışabilen bu mutasyon yöntemini gerçekleştirmek için rastgele iki gen seçilir ve bu genlerin konumları değiştirilir. Komşu genlerdeki bilgilerin çoğunun korunduğu takas mutasyonu, çok daha fazla sipariş kapasitesine sahip olması bakımından kullanışlıdır (Sivanandam ve Deepa, 2008; Soni ve Kumar, 2014).

4.3.5. Çevirme mutasyonu

Çoğu kodlama yönteminde ve özellikle ikili kodlamada kullanılan çevirme mutasyonu, kromozomda meydana gelen bir mutasyona dayanarak, değeri “0” olan bitlerin “1” ve değeri “1” olan bitlerin ise “0” yapılığı mutasyon çeşitidir. Ebeveyn birey göz önünde bulundurularak rastgele bir mutasyon kromozomu oluşturulur (Sivanandam ve Deepa, 2008). Oluşturulan mutasyon kromozomlarındaki “1” değerleri, karşılık gelen ana kromozomdaki bit değerlerine çevrilir ve böylelikle mutasyon gerçekleşmiş olur (Soni ve Kumar, 2014).

4.3.6. Akış mutasyonu

Akış mutasyonunda, rastgele bir gen seçiminin ardından ve seçilen bu genin değeri, alt ve üst sınır arasında rastgele bir değerle değiştirilir. Gerçek değer kodlamada kullanılan bir mutasyon metodudur (Soni ve Kumar, 2014).

4.3.7. Tekdüze mutasyon

Seçilen genin değerini, kullanıcı tarafından belirlenen üst ve alt sınırlar dahilinde, tekdüze bir değer ile rastgele olarak değiştiren ve sadece gerçek değer kodlamada kullanılan bir mutasyon operatörüdür (Sivanandam ve Deepa, 2009).

4.4. Yeni Bireylerin Eskileriyle Değiştirilmesi

Bu faz üreme döngüsünün son aşamasıdır ve temel olarak bir sonraki yineleme için bir yapı taşı görevi görmektedir. Yeni nesil yavrular üretilirken asıl soru, bu yavruların hangilerinin gelecek nesillere aktarılacağı ve mevcut kuşaktaki hangi kromozomlarının yerini alacağıdır. Cevap ise, daha iyi uygunluk değerine sahip bireylerin hayatta kalmak için daha fazla şansa sahip olduğu ve daha az uygunluğa sahip olanları geride bırakarak gelecek nesillere taşındıklarıdır. Bazı yavruların veya ebeveynlerin değiştirilerek yeni nesillerin oluşturulması süreci değişim şeması ile yapılmaktadır (Sivanandam ve ark., 2008). Popülasyon korunumu için temelde, nesil ve elitizm değişimi olmak üzere iki tür değişim stratejisi vardır. Nesil değişiminde popülasyon tamamının genleri her nesilde değiştirilir. Elitizm ise popülasyonun bütün genleri değiştirilmeden sadece her neslin en iyi bireyleri dışında değişimlerin yapıldığı değişim stratejisidir (Affenzeller ve ark., 2009). Elitizm, tüm yinelemelerde, nüfusun sadece küçük bir kısmının değiştirildiği üst üste binen popülasyonlar içerir ve yeni bireyler oluşturulduktan hemen sonra popülasyona yerleştirilir (Sarma ve De Jong, 1997).

4.5. Sonlandırma

GA'nın sonlanma koşulu, programın ne zaman ve hangi şartlar altıda sonlandırılacağını belirlemede oldukça önemlidir. GA'lar çözüme ulaşma süreçlerinin ilk fazlarında, sürecin sonlarına kıyasla oldukça hızlı çözümler üretmektedirler. Ancak nesil sayısı artıp program yerel minimum aralığına yaklaştıkça daha iyi çözümleri bulması zorlaşmakta ve bazı durumlarda da hiç bulamamaktadır. Bu tür senaryolar da kullanıcıya zaman kaybı ve verimsizlik olarak geri dönmektedir. Bu nedenle optimum çözüm aralığına varılabilecek en yakın noktada belirlenen sonlandırma kriterleri genetik operatörler kadar önemli bir yere sahiptir.

Sonlandırma kriterleri tamamen kullanıcın belirlediği kurallar doğrultusunda çalışıp istenilen şartlar sağlandığında programı sonlandırmaktadır. Program, belirlenen nesil sayısına ulaştığında, çalışmasına izin verilen süreyi doldurduğunda ve bulduğu uygunluk değerleriyle genel minimuma artık yaklaşamıyorsa sonlanır.

4.6. Parametreler

Algoritmanın çözüme ulaşılabilirliğini ve kesinliğini belirleyen birçok parametre bulunmaktadır. Bunlardan en yaygın olanları popülasyon büyüklüğü, çaprazlama, mutasyon ve nesil sayısı parametreleridir.

Çaprazlama parametresi bu operatörün ne sıklıkta gerçekleşeceğini belirler. Mevcut popülasyona çaprazlama yoluyla oluşturulacak birey sayısı çaprazlama oranıyla orantılı olarak artar. Olasılığın %100 olduğu durumlarda, tüm yavrular bu operatör ile oluşturulur. Eğer olasılık %0 ise bütün bireyler bir önceki neslin aynısı olacak şekilde hiçbir genetik operatöre maruz bırakılmadan olduğu gibi bırakılır. Bu parametre, oluşturulan yeni kromozomların eski olanların daha optimize parçalarını içereceği ve bu nedenle daha optimize olacağı için tasarlanmıştır. Çaprazlama parametresi ne kadar yüksek olursa o kadar optimize gen oluşturulacağı için genelde 0,8 olarak belirlenmektedir.

Mutasyon parametresi ise belirlenen değer doğrultusunda, nesildeki mevcut bireylerin ne sıklıkla mutasyona uğratılacağına karar vermektedir. Oran ne kadar yüksek

olursa bireylerin mutasyona uğratılma olasılığı o kadar yüksektir. Gen çeşitliliğinin aşırı değerlere çıkıp, oluşturulan yeni nesillerin atalarıyla olan gen benzerliğinin ortadan kalkmasına engel olmak için mutasyon parametresi genellikle 1/popülasyon boyutu olarak belirlenmektedir (Soni ve Kumar, 2014; Sivanandam ve Deepa, 2008).

Popülasyon büyüklüğü parametresi ise popülasyondaki birey sayısına karar vererek büyüklüğünü belirler. Popülasyon büyüklüğünün düşük belirlendiği durumlarda program çaprazlama için çok az bir varyasyona sahip olacak ve çözümün ufak bir kısmı bulunmuş olacaktır. Popülasyonun fazla belirlendiği durumlarda ise çaprazlama için oluşturulacak gen kombinasyonu ciddi miktarda artıp çözüm aralığı daha tatminkar olacak olsa da programın fark edilebilir seviyede yavaşlamasına sebep olacağından, genelde 30-50 aralığında belirlenmektedir.

5. MATERYAL VE YÖNTEM

5.1. Genetik Algoritma İçin Önerilen Adaptasyon Süreci

Bu bölümde Holland'ın 1970'teki ilk adımlarıyla başlanarak günümüze kadar birçok problem çözümünde ve optimizasyon sürecinde yaygın olarak kullanılıp, geliştirilen GA'ya, doğadaki adaptasyon teorisinden ilham alan yeni bir adaptasyon sürecinin geliştirilmesinden ve adaptasyonun ne olduğundan bahsedilmiştir. Bahsi geçen bu yeni adaptasyon süreci ile algoritmanın genel minimum değer aralığına hem daha erken ve hem de daha uygun bireylerle ulaştırılması amaçlanmıştır.

5.1.1. Adaptasyon nedir?

Canlılar, dünyanın oluşumunu tamamladığı ilk günden bu yana, yaşamlarını sürdürdüğü habitatlarda hayatta kalma mücadelelerinde onlara yardımcı olan bir takım özel özelliklerle yaşamlarına uyarlanırlar. Bu uyarlanma süreci adaptasyon teorisi olarak adlandırılmaktadır. Biyolojide adaptasyon, bir hayvan veya bitki türünün çevresine uygun hale gelmesi; doğal seleksiyonun kalıtsal varyasyonuna göre hareket etmesinin sonucudur. Örneğin bir afrika fili, sıcak bir yaşam alanı içinde yaşadığından kendisini serin tutmak için çırparak kullandığı çok büyük kulaklara sahiptir. Öte yandan, bir kutup ayısı soğuk bir yaşam alanı içinde yaşadığından kendisini sıcak tutmak için oldukça kalın bir tüy yapısına sahiptir (Margery, 1993).

Çevrelerine adapte olan bireyler sadece hayvanlar değildir ve aksine bitkiler de öyledir. Bir kaktüs çölde hayatta kalmak için iyi bir şekilde adapte edilmiştir. Oldukça geniş bölgelerden su toplamak için uzun kökleri ve uzun süre suyu depolayabilen gövdeleri vardır. Daha basit organizmalar bile yapıları, fizyolojileri ve genetiği, hareketleri veya dağılımları, savunma ve saldırı şekilleri, üreme ve gelişme durumlarında ve diğer açılardan çok çeşitli şekillerde uyarlanmaktadırlar. Bir habitatı paylaşan hayvanlar ve bitkiler özellikle o alanda yaşamak için uygundur ve diğer habitatlarda hayatta kalamayabilirler. Bu da demektir ki, bir habitat değişiminde, orada yaşayan hayvanlar ve bitkiler değişimden birlikte etkilenmektedir (Margery, 1993).

Adaptasyon teorisinin şaşırtıcı bulgularından ilham alan çalışmalar, son on yılda büyük bir patlama göstermektedir. Olgun bir adaptasyon teorisinin yavaş gelişmesinin nedenlerine odaklanan, şaşırtıcı bulguların dayandığı uyum teorisinin tarihini detaylı bir şekilde araştıran ve çağdaş adaptasyon teorisinin karşılaştığı birtakım zorlukları da dile getiren Allen Orr (2005)'un da belirttiği gibi patlama noktasına gelen bu teori, günümüzde bilgisayar birimleri de dahil olmak üzere birçok alana hızla uyarlanmaktadır.

5.1.2. Reel değer kodlama için adaptasyon süreci geliştirilmesi

GA, bir problemin optimum çözümünü ararken iki farklı yaklaşımı bir araya getirir. İlki, yerel optimumun arama sırasında daha hızlı bulunmasını sağlayan ve mevcut çözümlerin harmanlanması yoluyla yeni bireyler oluşturan çaprazlamadır. Diğer yaklaşım ise, mevcut çözümlere çeşitlilik ve yenilenebilirlik katan mutasyondur. Bu sayede algoritma genel minimum veya maksimumlarda istenilen çözüm aralıklarında gereken çözümleri sunabilmektedir.

GA doğal evrimsel sürecin bir modelini oluşturma çabası içindedir. Eski nesilden yeni nesle ait bir bireyin oluşmasını da çaprazlama üstlenir. Bir neslin sonunda ortaya çıkan sıra dışı (ebeveynlerde görülmeyen) değişiklikler ise mutasyon ile gösterilir. Bu değişiklikler genotip değişiklik olarak adlandırılır. Mutasyona uğramış bireylerden hayatta kalanlar çeşitliliği artırırlar. Ancak bireyler üzerinde çevresel etkilerden dolayı fenotip değişiklikler de olmaktadır. Bu değişiklikler doğrudan bireyin çevreye adaptasyonudur. Bu değişiklikler yeni nesillere aktarılmaz. Ancak ortama daha uyumlu bireylerin oluşmasını sağlar. Ayrıca bu bireylerin hayatta kalma olasılıkları da artar. Çevresel koşullar bireyin ömründen daha kısa bir zaman dilimi içinde fizyolojik veya morfolojik değişiklikler meydana getirir. Örneğin çevresel şartların etkisi ile yetişkin organizmaların organ sistemlerindeki boyut değişiklikleri ile ilgili çeşitli çalışmalar mevcuttur (Piersma and Drent, 2003).

Bu noktadan yola çıkarak varılan sonuç, popülasyon içindeki bireylerin çaprazlanıp ve mutasyona uğradığıdır. Bu süreçlere ek olarak bireyler bulundukları ortama uyum gösterirler. Bu olay bireyin ortama adaptasyonu olarak düşünülebilir. Yani çaprazlama ve mutasyon GA'nın vazgeçilmez süreçleridir ancak doğal süreci yansıtmakta

yetersiz kalmaktadırlar. Bu çalışmanın amacı bu eksikliği gidermektir. Doğadaki genetik sürecin tam anlamı ile optimizasyon algoritmasına dönüştürülmesini sağlamak için adaptasyon süreci tasarlamak ana hedefimizdir.

Adaptasyonun özü bireyin çevreye daha uyumlu hale gelmesidir. Ancak GA süreçlerinde bu durum uygunluk fonksiyonu ile ölçülür ve değerlendirilir. Böylece eğer bir gen çevresel şartlara daha uyumlu olacaksa yapılması gereken şöyle sırlanabilir. Habitatın (önceki ve mevcut popülasyonların) en iyi bireyi hangisi ise çevreye en uyumlu olan odur. Adapte edilmek istenen birey de doğal olarak bu en iyi bireye benzemelidir. Metnin devamında bu en iyi birey donor birey olarak adlandırılacaktır.

Mutasyon ve çaprazlama işlemlerinde olduğu gibi adaptasyon işlemi de geni temel alır. İlk olarak yapılması gereken hangi genlerin adaptasyona uğrayacağını belirlemektir. Genellikle bu işlem kodlama yönteminden bağımsız olarak standart bir işlemdir. Mutasyon ve çaprazlama işlemlerinde olduğu gibi adaptasyon işlemi için de bir operatör önerilmiştir. Adaptasyon oranı dediğimiz bu parametre [0,1] aralığında bir reel sayıdır. Genlerin adaptasyona uğrayıp uğramayacaklarına bu değere dayalı olarak karar verilir. Her bir gen için rastgele bir sayı üretilecek ve eğer bu sayı adaptasyon oranı değerinden küçük ise o gen adaptasyona uğrayacaktır.

İkinci olarak yapılması gereken adaptasyona uğrayacak bir genin nasıl değişikliğe uğrayacağıdır. Bu aşamada yapılması gereken ilk şey popülasyon içinde çevreye en iyi uyumu sağlamış bireyi belirlemektir. Bunun için de popülasyon genelindeki bütün bireylerin uygunluk değerleri gözden geçirilir ve en iyi uygunluk değerine sahip birey işaretlenir. En iyi uygunluk değerine sahip işaretlenmiş bu birey donor birey olarak adlandırılırken, genleri de donor gen olarak adlandırılmaktadır. Çevreye ve koşullara en iyi uyumu gösterip popülasyondaki en uygun birey olacağı ve adaptasyon süresi boyunca adapte edilmesine karar verilmiş bütün bireylere kendi genlerini vereceği için de bu adla adlandırılmaktadır.

Adapte edilmesine karar verilmiş tüm genler, donor bireyin ilgili genleri ile değiştirildikten sonra adaptasyon süreci o nesil için sona ermiş olacaktır. Her nesilde bir adım daha adapte olarak uygunluk değerlerini daha uygun noktalara taşıyan bireyler, optimizasyon süreci boyunca algoritmaya ve kullanıcıya zaman kazandırarak sonlandırma koşuluna daha uygun bireylerle ulaşma fırsatı sağlamaktadır.

Aşağıda, sırasıyla seçilim, çaprazlama ve mutasyon operatörlerinden oluşan KGA üzerinde geliştirilen adaptasyon sürecinin gen ve uygunluk değerleri üzerindeki olumlu etkilerine ve sürecin geneline olan katkısı gösterilmiştir.

Çizelge 5.1. Rastgele popülasyon oluşumu

Uygunluk Değeri	*1.Gen*	*2.Gen*
32.8521	4.1900	1.2100
55.0407	-4.3500	-0.5100
42.0433	2.5900	-4.0500
23.2785	0.8900	-0.5200
43.4156	-2.1500	-4.7100
57.9671	4.6300	4.1200
34.7491	-1.9500	4.7900
16.5183	-0.8300	-2.8900
46.8000	-4.6000	0.8000
26.6433	-2.4400	0.9500

Her bireyin iki genden ve popülasyonun da on bireyden oluştuğu minimizasyon probleminde herhangi bir neslin rastgele gösteriminin Çizelge 5.1'deki gibi olduğu düşünülürse, bireylerin turnuva metodundan sonraki dizilimi de Çizelge 5.2'deki gibi gibi olacaktır.

Çizelge 5.2. Turnuva sonrası popülasyon

Uygunluk Değeri	*1.Gen*	*2.Gen*
16.5183	-0.8300	-2.8900
16.5183	-0.8300	-2.8900
26.6433	-2.4400	0.9500
16.5183	-0.8300	-2.8900
16.5183	-0.8300	-2.8900
34.7491	-1.9500	4.7900
16.5183	-0.8300	-2.8900
26.6433	-2.4400	0.9500
16.5183	-0.8300	-2.8900
23.2785	0.8900	-0.5200

Seçilim operatörünün tamamlanmasıyla, çaprazlama operatörüne göre çaprazlanması sağlanan ve akabinde mutasyon operatörüne alınacak bireylerden meydana gelecek yeni neslin ve donor bireyin de gösterimi Çizelge 5.3.'deki gibi olacaktır.

Çizelge 5.3. Çaprazlama sonrası popülasyon ve donorün belirlenmesi

Uygunluk Değeri	*1.Gen*	*2.Gen*
16.5183	-0.8300	-2.8900
16.5183	-0.8300	-2.8900
23.2785	-0.5200	0.8900
12.2545	-1.1967	-2.0153
7.5341	-1.0851	-1.1407
25.8373	-1.6949	3.0407
12.2545	-1.1967	-2.0153
6.4456(donor birey)	-2.0733(1.donor gen)	0.0753(2.donor gen)
16.5183	-0.8300	-2.8900
23.2785	0.8900	-0.5200

Ancak çaprazlanma evresi tamamlanarak oluşan yeni bireyler, KGA'da olduğu gibi direk mutasyona alınmadan, geliştirilen adaptasyon evresiyle adapte edilecek ve daha uygun hale getirilecektir. Çizelge 5.3'deki popülasyonda görüldüğü üzere en iyi uygunluk değerine sahip 6.4456 uygunluk değerli bireyin donor olarak belirlenmesinin ardından adaptasyon oranı dahilince adapte edilmesine karar verilen genlerin donor birey genleriyle değiştirilmesi ve neslin adaptasyon operatörü sonrasındaki hali de Çizelge 5.4'teki gibi olacaktır.

Çizelge 5.4. Adaptasyon sonrası popülasyon ve adapte edilen genler

Uygunluk Değeri	*1.Gen*	*2.Gen*
9.5314	-2.0733	-2.8900
16.5183	-0.8300	-2.8900
12.7453	-0.5200	0.0753
12.2545	-1.1967	-2.0153
10.3007	-2.0733	-1.1407
25.8373	-1.6949	3.0407
12.2545	-1.1967	-2.0153
6.4456	-2.0733	0.0753
11.9769	-0.8300	0.0753
8.8527	-2.0733	-0.5200

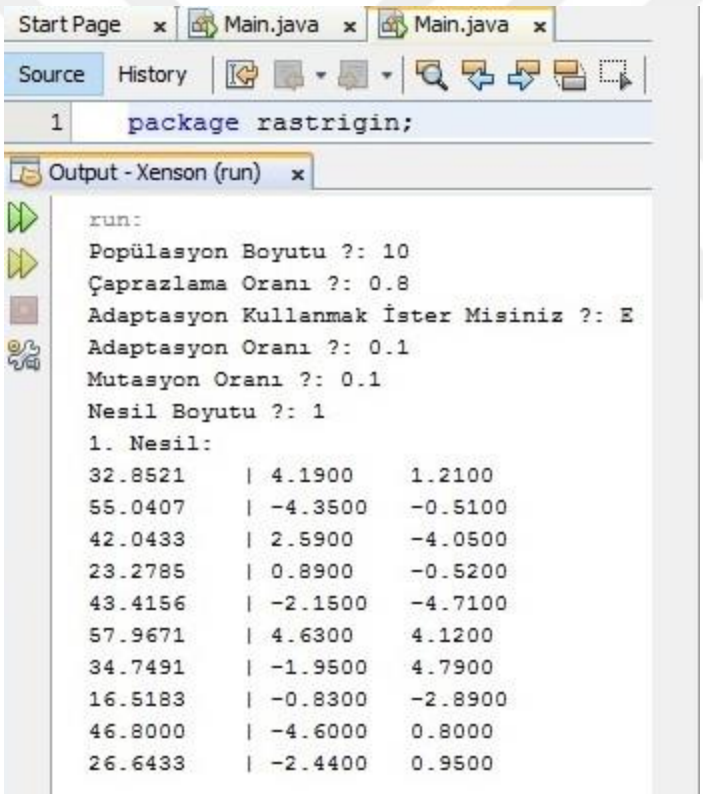
Rastgele popülasyon oluşumu ve seçilim evresinin tamamlanmasıyla doğadaki üreme teorisinin bir simülasyonu olan çaprazlama evresine alınan bireyler, oluşturulan yavru birey sayısı popülasyon sayısına eşit olana kadar çaprazlama oranınca çaprazlanmaya devam edilir. Çaprazlanma evresinin tamamlanmasıyla oluşturulan yavru

birey genleri daha önce belirlenen en iyi uygunluk değerine sahip donor birey genleri ile adaptasyon oranınca değiştirilir. Geliştirilen adaptasyon evresinin uygulanmasındaki asıl amaç mutasyon evresi ile birtakım değişikliklere uğrayıp mevcut uygunluk değer aralığından bir miktar daha uzaklaşacak olan bireyler, ilgili evre gerçekleşmeden daha iyi uygunluk değerine sahip donor bireyin değerlerleriyle değiştirilerek, daha uygun değer aralığına getirilmiş ve dolayısıyla daha adapte olmuş bir hale getirilecektir. Böylece olası bir mutasyon ihtimaline karşı genler, uygun değerlere adapte edilmesiyle ortama daha iyi uyum sağlayacaklarından dolayı, mevcut değer aralığından uzaklaşma ihtimalinin önüne de ciddi ölçüde geçilmiş olunacaktır.

6. BULGULAR

6.1. Adaptasyon Süreçli GA'nın Çalışma Prensibi ve Evreleri

Çalışmanın ilk aşamasında kodlanan KGA'nın işleyişinden emin olunduktan sonra bir takım modifikasyon ve eklentiler dahilinde geliştirilen bu algoritma, literatürden farklı olarak optimizasyon sürecine olumlu birçok etkiyi beraberinde getiren bir adaptasyon süreci içermektedir. Bu süreç ile ulaşılmak istenen optimum değer aralığına daha kısa sürede ve daha uygun bireylerle ulaşılması amaçlanmıştır.



Şekil 6.1. Örnek başlangıç popülasyonu.

Ackley, Bohachevsky, Eggholder, Holder Table ve Rastrigin test fonksiyonlarından hangisiyle çalışılacağına karar verildikten sonra algoritma, kullanıcıdan parametre değerlerini girmesini ister. Bu parametreler popülasyon boyutu ile başlayarak, sırası ile çaprazlama oranı, adaptasyon evresinin olup olmayacağını soran

parametre, adaptasyon oranı, mutasyon oranı ve son olarak nesil sayısının girildiği parametrelerdir.

Rastrigin test fonksiyonu seçilip program çalıştırıldıktan sonra, Şekil 6.1'de görüldüğü gibi popülasyon boyutu, genler ve uygunluk değerlerindeki değişimlerin rahatça gözlemlenmesi açısından 10 olarak belirlenmiştir. Çaprazlanma oranı 0,8, adaptasyon oranı 0,1 ve mutasyon koranı da 0,1 olarak belirlenmiştir.

Örnek bir başlangıç popülasyonunun oluşturulmasından sonra bireyler seçilim evresine alınır. Bu evrede, her seferinde popülasyondan rastgele 3 tane birey seçilir ve içlerinde en iyi uygunluk değerine sahip birey seçilimini tamamlar ve bu 3 birey tekrar popülasyona bırakılır. Popülasyondan 3 bireyin alıp, içlerinden birinin seçilip ve tekrar popülasyona bırakılma döngüsü seçilen birey sayısı popülasyon büyüklüğüne eşit oluncaya kadar devam eder. Bireylerin 3'lü turnuva metodunca seçilim evresini tamamlaması Şekil 6.2'deki gibidir.

Şekil 6.2. Seçilim evresi sonrası popülasyon.

Bu evreyi tamamlayan bireyler, sırasıyla X, Y, Z ve W olarak gösterilirse, X ile Y, Z ile W çaprazlanıp ikişer yavru oluşturacak şekilde popülasyon sonuna kadar devam eden çaprazlama evresine alınacaktır. Çaprazlama oranı bu evrenin gerçekleşme sıklığına karar vermektedir. Girilen oran, popülasyon boyutu ile çarpılır. Çarpımın sonucu kadar birey çaprazlanır ve yeni bireyler oluşturulur. Popülasyon boyutundan çıkarılan çarpım sonucu kadar birey de olduğu gibi kalır. Örnek verilecek olursa, popülasyon boyutu 10 ve çaprazlama oranı da 0,8 olarak belirlenirse, 8 birey (10 x 0,8) çaprazlama metodu ile

oluşturulacak, 2 birey (10 - (10 x 0,8)) ise olduğu gibi kalacaktır. Çaprazlama evresiyle oluşturulan bireyler Şekil 6.3'te görülebilmektedir.

Çaprazlama Sonrası:		
16.5183	-0.8300	-2.8900
16.5183	-0.8300	-2.8900
23.2785	-0.8900	-0.5200
12.2545	-1.1967	-2.0153
7.5341	-1.0851	-1.1407
25.8373	-1.6949	3.0407
12.2545	-1.1967	-2.0153
6.4456	-2.0733	0.0753
16.5183	-0.8300	-2.8900
23.2785	0.8900	-0.5200

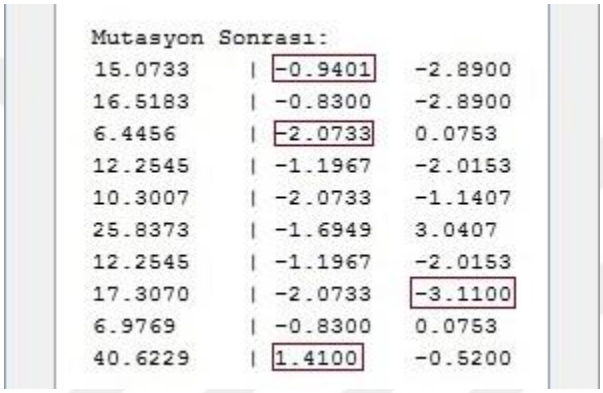
Adaptasyon Sonrası:		
9.5314	-2.0733	-2.8900
16.518	-0.8300	-2.8900
12.7453	-0.8900	0.0753
12.2545	-1.1967	-2.0153
10.3007	-2.0733	-1.1407
25.8373	-1.6949	3.0407
12.2545	-1.1967	-2.0153
6.4456	-2.0733	0.0753
11.9769	-0.8300	0.0753
23.2785	0.8900	-0.5200

Şekil 6.3. Çaprazlama ve geliştirilen adaptasyon evresi sonrası popülasyon.

Literatürde, çaprazlama evresini tamamlayan bireyler mutasyon evresine alınarak, popülasyondaki bireyler mümkün olan en uygun noktaya ulaştırılır ve nesil oluşumu tamamlanmaktadır. Literatürden farklı olarak geliştirilen ve çaprazlama evresini tamamlayan bireylere kendilerini adapte etme imkanı sağlayan adaptasyon evresi ile bireyler çok daha uygun değerlerle mutasyon evresine alınır ve gerçekleşmesi muhtemel mutasyonlara daha dirençli hale gelerek mevcut neslin oluşumu çok daha optimize genlerle sağlanır. Şekil 6.3'te görüldüğü üzere, çaprazlama evresinden çıkan bireyler içinde en uygun değere sahip, 6.4456 uygunluk değerli birey, donor birey olarak belirlenir. Bu aşamadan sonra, belirlenen donor bireyin ilgili geni, bireylerin adapte edilmesine karar verilen genleriyle değiştirilir.

Bu aşamada önemli olan bireyin adapte edilip edilmeyeceğine nasıl karar verileceğidir. Karar aşamasında, belirlenen adaptasyon oranı, adaptasyonun hangi sıklıkla gerçekleşeceğine karar verir. Her gene geçici bir rastgele değer atanmasının ardından, atanan bu değerler girilen adaptasyon oranıyla karşılaştırılır. Atanan bu rastgele sayı girilen adaptasyon oranından düşük ise bu gen belirlenmiş donor bireyin ilgili geniyle güncellenir ve birey böylece adapte edilmiş olur. Eğer atanan geçici rastgele sayı, adaptasyon oranından büyük iste, birey adapte edilmez ve hiçbir değişime uğramadan bu evreyi tamamlar.

Adaptasyon evresinin tamamlanmasıyla bireyler mutasyon evresine alınır. Bu evrede belirlenen mutasyon oranı mutasyonun hangi sıklıkla gerçekleşeceğine karar verir. Tıpkı adaptasyon evresinde olduğu gibi her gene geçici bir rastgele değer atanmasının ardından, atanan bu değerler girilen mutasyon oranıyla karşılaştırılır. Atanan bu rastgele sayı girilen mutasyon oranından düşük ise gene tekrardan rastgele bir değer atanır ve atanan bu gen değerlerine göre tekrardan uygunluk değer hesaplaması yapılır. Birey böylece mutasyona uğratılmış olur. Eğer, atanan geçici rastgele sayı, mutasyon oranından büyük iste, gen mutasyona uğramaz ve hiçbir değişime uğramadan bu evreyi tamamlar.



Şekil 6.4. Mutasyon evresi sonrası popülasyon.

Genetik operatörlerin tamamlanmasının ardından mevcut nesil oluşumu tamamlanır ve neslin son hali Şekil 6.4'teki gibi olur. Bu aşama aynı zamanda bir sonraki neslin başlangıç popülasyonudur. Bir sonraki nesil de aynı genetik süreçlere tabi tutulur ve girilen popülasyon boyutuna ulaşılana dek bu döngü devam eder.

6.2. Çalışma Sonuçlarının Karşılaştırılması

Bu bölümde, önerilen adaptasyon sürecinin GA üzerindeki etkisi beş farklı test fonksiyonu kullanılarak değerlendirilmiştir. Bu fonksiyonlardan dördü çok modüllü (Ackley, Eggholder, Holdertable, Rastrigin) ve biri de tek modüllü (Bohachevsky) test fonksiyonlarıdır. Test fonksiyonlarında, boyut iki olarak belirlenmiştir. Önerilen adaptasyon sürecinin performansını doğrulamak için, KKA ve adaptasyon işlemini içeren GGA aynı parametrelerle çalıştırılmıştır. Belirlenen popülasyon büyüklüğü her iki

algoritma için de 50'dir. Her iki algoritma da, test fonksiyonları optimum çözüme ulaşana kadar çalıştırılmıştır. Her test fonksiyonunun istatistiksel sonuçları, 30 bağımsız çalışma ile belirlenmiştir. GA için kullanılan parametre listesi Çizelge 6.1'de gösterilmiştir. Uygulama için kullanılan işlemci Intel (R) Core (TM) i5-6500 CPU 3.20 GHz'dir

Çizelge 6.1. KGA ve GGA'nın parametreleri

Parametre	*KGA*	*GGA*
Gen Sayısı	2	2
Kodlama	Gerçek-Değer Kodlama	Gerçek-Değer Kodlama
Popülasyon Boyutu	50	50
Seçilim Metodu	Turnuva (3'lü)	Turnuva (3'lü)
Çaprazlama Metodu ve Oranı	Tekdüze (0,8)	Tekdüze (0,8)
Adaptasyon Oranı	Yok	0,1
Mutasyon Metodu ve Oranı	Tekdüze (0,1)	Tekdüze (0,1)
Sonlandırma Ölçütü	Nesil Sayısı (400 nesil)	Nesil Sayısı (400 nesil)

Bu tez çalışmasında programlama dili olarak java, programlama ortamı olarak NetBeans IDE 8.2 kullanılmıştır. Ayrıca sonuçların karşılaştırılması için ortalama ve standart sapma operasyonlarının hesaplandığı Microsoft Excel ve grafiklerin oluşturulduğu MATLAB ortamlarından yararlanılmıştır. Aşağıda bu tezde kullanılan yöntemler, aşamalar ve algoritma hakkında bilgiler verilmiştir.

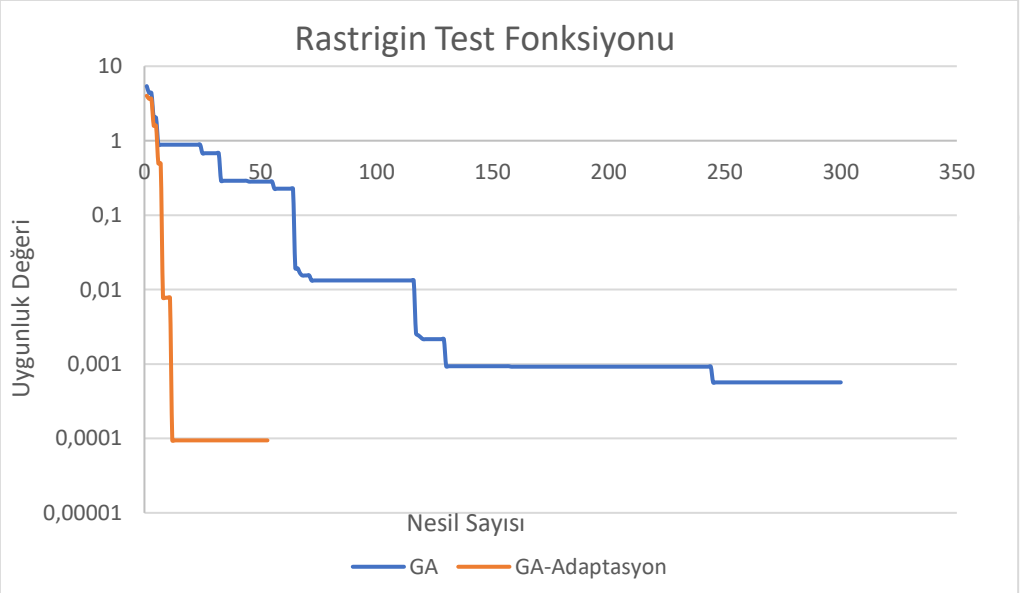
Bu çalışmada reel sayılardan oluşan reel değer kodlama yöntemi kullanılmıştır. Kodlama aşaması sırasıyla rastgele popülasyon oluşturulması, çaprazlama, literatürden farklı olarak geliştirilen adaptasyon ve mutasyon süreçlerinden oluşan genetik operatörleridir.

Seçilim metodu olarak 3'lü turnuva yöntemi kullanılmıştır. Bu yönteme göre, oluşturulan rastgele nüfusun uygunluk değerleri test fonksiyonlarınca hesaplandıktan sonra bireyler 3'lü turnuva metoduna tabi tutularak çaprazlama operatörüne hazır hale getirilmiştir. Çaprazlama metodu olarak aritmetik çaprazlanma kullanılmıştır. Bu metot, bir takım doğrusal kısıtlamalarla aritmetik ortalaması alınan iki ebeveynden iki çocuk oluşturulacak şekilde uygulanmıştır.

Bireyleri daha uyumlu hale getirmek amacıyla geliştirilen adaptasyon evresi, algoritmanın bu aşamasında kullanılmıştır. Çaprazlama evresi sonrası donor birey

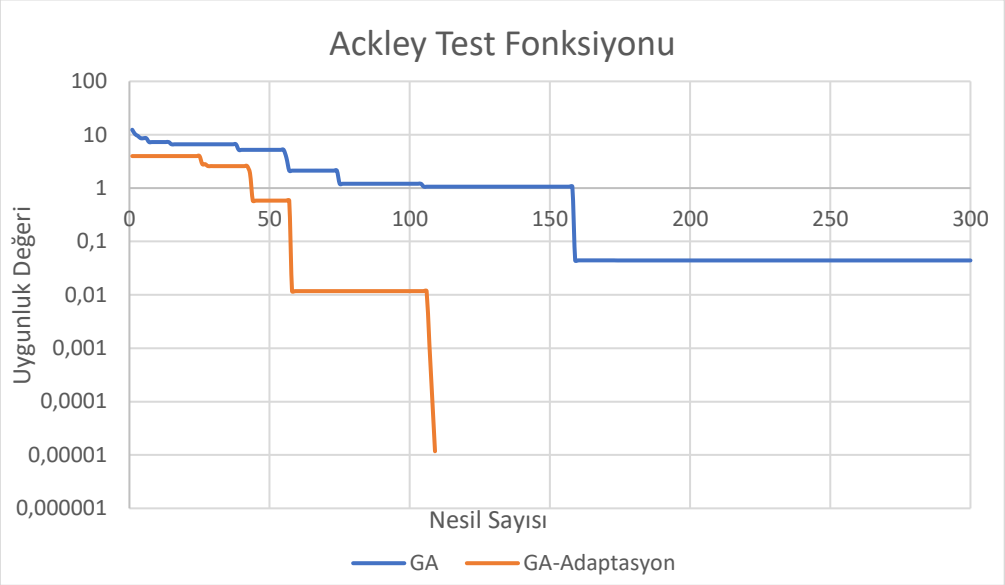
belirlenmiş ve daha sonra her gene [0.1] aralığında rastgele değerler atanmıştır. Eğer atanan rastgele değer adaptasyon oranından küçük ise gen donor genin ilgili geniyle değiştirilmiş, büyük ise gen üzerinde hiçbir değişiklik yapılmamıştır. Bu işlem bütün genler üzerinde tek tek uygulanmıştır. Mutasyon metodu olarak ise tekdüze mutasyon kullanılmıştır. Bu metot gereğince tıpkı adaptasyon evresindeki gibi genlere rastgele değerler atanmış ve atanan rastgele değerlerle mutasyon oranı karşılaştırılarak genin mutasyona uğrayıp uğramayacağına karar verilmiştir. Sonlandırma kriteri olarak da nesil sayısı kullanılmıştır. Algoritma her çalıştırıldığında kullanıcıya programı hangi nesilde sonlandırmak istediğini sormakta ve böylece kullanıcı programı istediği nesil sayısında sonlandırabilmektedir.

6.2.1 En iyi bireylerin test fonksiyonlarına göre karşılaştırılması



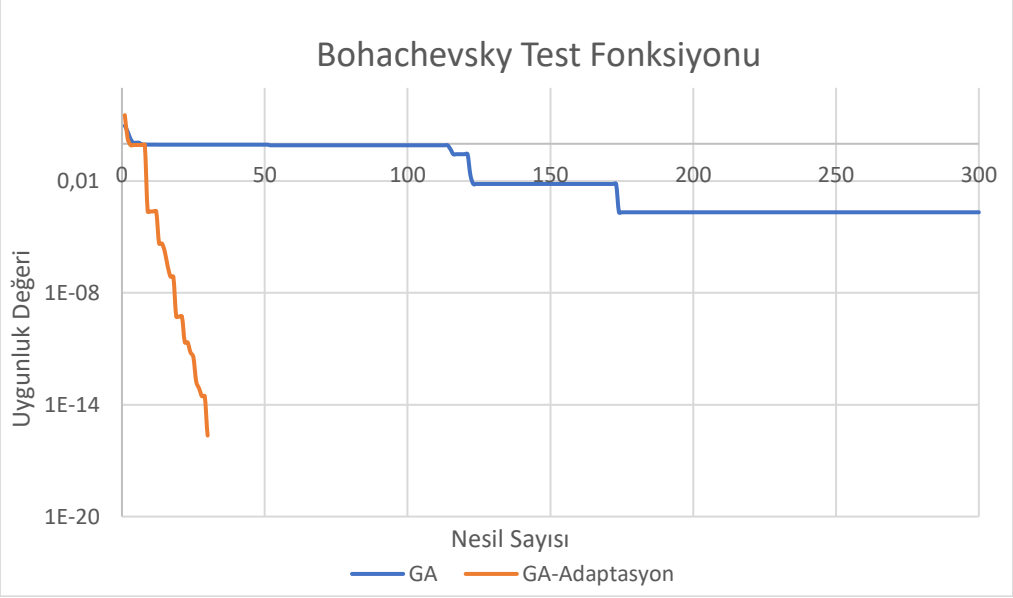
Şekil 6.5. Rastrigin test fonksiyonu karşılaştırması.

Rastrigin test fonksiyonuyla elde edilen Şekil 6.5'teki sonuçlar, tıpkı diğer test fonksiyonlarında olduğu gibi örnek bir çalıştırmadır ve ulaşılmak istenen 0 değerli genel minimum noktasına, geliştirilen adaptasyon süreciyle çok daha kolay ve stabil bir şekilde ulaşılabildiğini göstermektedir.



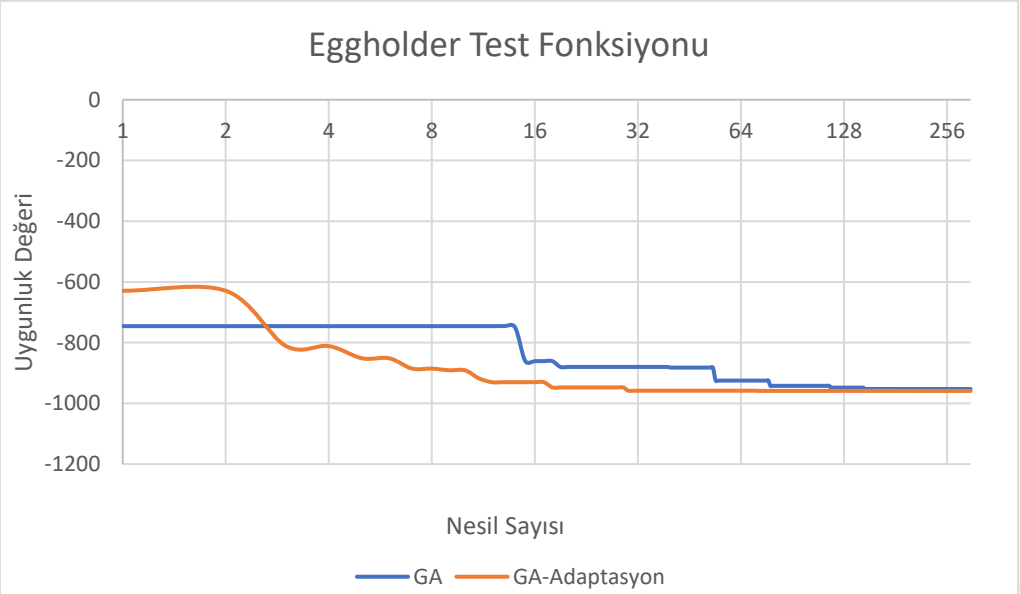
Şekil 6.6. Ackley test fonksiyonu karşılaştırması.

Ackley test fonksiyonu ile elde edilen sonuçlar, programın, fonksiyonun genel minimum noktası olan 0'a, daha erken ve daha iyi sonuçlara ulaştığını göstermektedir. Şekil 6.6'da görüldüğü üzere, KGA ile elde edilen uygunluk değerleri tam olarak 0'a ulaşamazken, geliştirilen adaptasyon evresi ile genel minimum noktası olan 0'a programın erken fazlarında ulaşmak mümkün olmuştur.



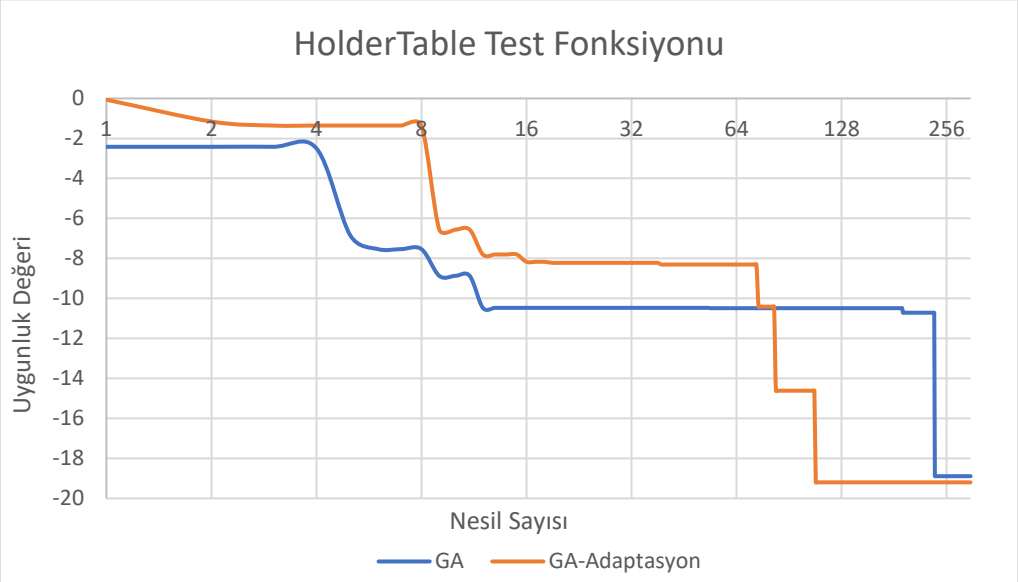
Şekil 6.7. Bohachevsky test fonksiyonu karşılaştırması.

Şekil 6.7'de görüleceği üzere, rastgele oluşturulan popülasyonlar için GGA KGA'dan daha büyük uygunluk değerlerinde başlamasına rağmen, geliştirilen adaptasyon süreci popülasyona ait en iyi uygunluk değerlerini kısa bir sürede genel minimum seviyesi olan 0'a indirmiş ve kararlılığını koruyarak KGA'dan daha iyi uygunluk değerleriyle programı sonlandırmıştır.



Şekil 6.8. Eggholder test fonksiyonu karşılaştırması.

Bohachevsky test fonksiyonunda olduğu gibi programa yine yüksek rastgele değerlerle başlamasına rağmen kısa süre içinde KGA'yı yakalayıp öne geçen GGA, Şekil 6.8'de gösterildiği gibi genel minimum değeri -959.6407'a ulaşamadan programı sonlandıran KGA'ya kıyasla daha iyi sonlandırmıştır.



Şekil 6.9. Holder table test fonksiyonu karşılaştırması.

Genel minimum değeri -19.2085 olan HolderTable test fonksiyonuyla Şekil 6.9'da elde edilen sonuçlara göre, geliştirilen adaptasyon süreciyle birlikte, ulaşılmak istenen uygunluk değerine KGA'ya kıyasla daha erken ulaşılmıştır.

6.2.2. Optimal çözümlerin nesil sayısı sonuçları

Çizelge 6.2'de gösterildiği gibi, önerilen adaptasyon süreci algoritmayı önemli ölçüde hızlandırırken, optimum çözümü elde etmek için gereken nesil sayısı da önemli ölçüde azalmıştır. Bu hem çok modüllü hem de tek modüllü test fonksiyonları için geçerli olurken, adaptasyon sürecinin gerekliliğini de ortaya koymaktadır.

Çizelge 6.2. Optimal çözümlerin nesil sayısı sonuçları

Test Fonksiyonu	*KGA*		*GGA*	
	Ortalama	*Standart Sapma*	*Ortalama*	*Standart Sapma*
Rastrigin	240,2	113,32	80,4	33,26
Ackley	179,7	93,69	57,6	43,21
Bohachevsky	234,9	124,43	92,5	81,76
Eggholder	352,1	24,42	252	74,27
Holder Table	203,1	98,38	139,3	104,55

Getirilen ek operasyonlarla önerilen süreç, optimum uygunluk değerlerinin bulunduğu nesil sayılarındaki düşüş olarak dikkat çekmektedir. KGA'da bulunan optimum çözümler her seferinde genel minimuma inemezken, GGA ile ulaşılmak istenen genel minimum değerine neredeyse her seferinde ulaşılmış ve ulaşılan nesil sayısı da her seferinde çok daha düşük sayılarla olmuştur. Çizelge 6.2'de yöntemin, genel minimum noktalarına ulaşılan neslin sayısına olan etkisi incelenmiştir.

6.2.3. Optimal çözümlerin çalışma süresi sonuçları

Çizelge 6.3. Optimal çözümlerin çalışma süresi sonuçları

Test Fonksiyonu	*KGA*		*GGA*		*Çalışma Süresi Azalım Yüzdesi*
	Ortalama	*Standart Sapma*	*Ortalama*	*Standart Sapma*	
Rastrigin	0,0666	0,0209	0,0537	0,0129	%19,369
Ackley	0,0628	0,0069	0,0519	0,0064	%17,356
Bohachevsky	0,0715	0,0266	0,0522	0,0083	%26,993
Eggholder	0,0762	0,0104	0,0694	0,0057	%8,923
Holder Table	0,0612	0,0069	0,0588	0,0103	%3,921

Önerilen süreç GA'ya ek operasyon ve karmaşıklıklar getirse de, GA'nın çalışma zamanı üzerindeki etkisi dikkat çekmektedir Çizelge 6.3'te yöntemin çalışma süresi açısından etkisi incelenmiştir. Bu çizelge her uygunluk fonksiyonu için CPU süresini saniye cinsinden sunmuştur. Algoritma geneline her ne kadar karmaşıklık ekleniyor olsa da, optimum çözüme ulaşmak için gereken nesil sayısındaki azalma sebebiyle çalışma süresin de azaldığı görülmektedir. Çizelgede ayrıca çalışma sürelerinin yüzdelik cinsten azalımı da verilmiştir.

6.2.4. Sabit nesil sayılarıyla test fonksiyon sonuçları

Çizelge 6.4. 300 nesilden oluşan GA'nın test fonksiyon sonuçları

Test Fonksiyonu	*KGA*		*GAA*		*Fonksiyonun Optimum Değeri*
	Ortalama	*Standart Sapma*	*Ortalama*	*Standart Sapma*	
Rastrigin	0,00069145	0,00118720	0,00000021	0,0000006	0
Ackley	0,01876631	0,02593699	0,00113715	0,0027556	0
Bohachevsky	0,00563769	0,01124438	0,00001821	0,0000389	0
Eggholder	-940,6947	0,44191292	-959,4643	0,1150899	-959,6407
Holder Table	-18,9693	0,26289615	-19,1419	0,0192346	-19,2085

Sabit nesil sayısıyla çalıştırılmış yöntemlerin performans karşılaştırılması Çizelge 6.4'te yer almaktadır. Tabloda gösterilen veriler göz önünde bulunduğunda adaptasyon işlemi test fonksiyonlarının sonuç değerlerini azaltmakta ve bu da başarının arttığı anlamına gelmektedir.

Tabloda verilen veriler neticesinde, her fonksiyonda farklılık gösterebilen genel minimum noktalarına KGA ile ulaşmak, özellikle programın belirli fazlarından sonra oldukça zor hale gelmektedir. Bunun sebebi belirli nesillerden sonra uygunluk değerlerinin belirli değerlerde sıkışarak daha uygun noktalara gidememesi veya yerel minimuma takılması olarak gösterilebilir. Böylelikle KGA çoğu çalıştırmada tam

manasıyla genel minimum noktalarına inememekte ancak bu noktalara oldukça yakın değerler vermektedir.

Tablo ve grafiklerdeki veriler göz önüne alındığında, sınırlı sayıdaki nesilde bile geliştirilen adaptasyon süreci ile bu sıkışmaların önüne geçilerek program istenilen genel minimum noktalarına tam manasıyla inebilir hale getirilmiştir. Ulaşılmak istenen bu uygunluk değerleri KGA'nın bunu başarabildiği durumlardaki istatistiklerine oranla çok daha iyi ve erken sonuçlar vermektedir.

7. TARTIŞMA VE SONUÇ

Bu çalışmada, GA için yeni bir ek operasyon olarak önerilen ve adaptasyon olarak adlandırılan süreç ile asıl hedeflenen, popülasyondaki bireyleri algoritmanın ilgili aşamaya kadarki en iyi çözümüne adapte etmektir. Bu şekilde, doğal seleksiyon, çaprazlama ve mutasyon gibi doğada gerçekleşen olgulardan ilham alan genetik süreçlere ek olarak geliştirilen adaptasyon süreci, tıpkı diğer operatörler gibi GA içine dahil edilmiştir. Seçilim, çaprazlama ve mutasyon gibi algoritma içinde bağımsız olarak çalışan operasyonlar gibi adaptasyon süreci için de bir operatör geliştirilmiştir. Bu operatör çaprazalama ve mutasyon evreleri arasonda gerçekleşmekte ve donor genin belirlenip adapte edilmesine karar verilen genleri bu gene göre adapte etme mantığıla çalışmaktadır. Bu operatör ile amaçlanan algoritmanın doğadan ilham almakta ve simule etmekte yetersiz kaldığı noktaları da dahil ederek algorimayı daha verimli ve daha efektif bir biçimde kullanabilmektir. Çeşitli test fonksiyonlarıyla yapılan değerlendirme sonucunda algoritmanın açıkça geliştirildiği ve amaçlanan hedeflere ulaşıldığı görülmektedir. Deneysel sonuçlar bu yeni sürecin algoritmayı hızlandırdığını, çözüme daha az nesil sayılarıyla ulaştığını ve özellikle küçük değerli nesiller için daha iyi çözümler elde edildiğini göstermektedir.

Genel minimuma ulaşma süreci geliştirilen bu yeni yaklaşım ile hem daha az nesille hem de daha az zamanda başarılmıştır. Bu başarımlar gerek grafikler gerek tablolar ile ıspatlanmış ve algoritmanın çalışma süresi açısından karşılaştırmaları getirilen ilave operasyon ve işlem karmaşıklıklarına rağmen göstermektedir ki algoritmanın çalışma süresini farkedilir bir biçimde azaltmıştır. Çalışma sürelerinde olduğu gibi bulunan en iyi değerlerde de gözle görülür başarımlar yine tablo ve verilere bağlı olarak elde edilmiştir. Bu durum adaptasyon sürecinin evrimsel algoritmaların doğal ve gerekli bir parçası olduğu fikrini desteklemektedir. Daha verimli optimizasyon modelleri olarak genellikle hibrit yapıların kullanıldığı literatüre önerdiğimiz adaptasyon sürecinin, GA dahil olmak üzere tüm evrim tabanlı hibrit sistemlerin başarısını arttırabileceği düşünülmektedir. Yani, adaptasyon süreciyle diğer evrimsel algoritmalar da (diferansiyel evrim, vb.) bu şekilde geliştirilebilecektir.

Çaprazlama, mutasyon gibi genetik operatörler ve kodlama yöntemleri gibi çeşitli evrelerde birçok farklılığı bünyesinde barındırabilen KGA'ya bu uyarlama işlemi sadece değer kodlaması için önerilmiş olsa da ikili kodlama, permütasyon kodlaması gibi diğer kodlama yöntemleri için de birer adaptasyon işlemi geliştirilebileceği gibi adaptanson evresi de yine kendi içerisinde geliştirilebilir. Donor bireyin bulunmasında başka yöntemlerin geliştirilmesi veya kullanılmış mecvut yöntemin başka hesaplamalarla genişletilmesi, donor birey üzerindeki bir takım değişiklikler, adaptasyon oranının son derecede hassas rakamlarla detaylıca test edilmesiyle en uygun adaptasyon oranının tekrar belirlenmesi ve son olara adaptasyon algoritması üzerinde kalıtsallığı arttıracak ya da azaltacak şekilde çözülmek istenen problemin çözümüne bağlı olarak bir takım değişiklik ve eklemeler gibi yöntemlerle adaptasyon operasyonu geliştirilebilir.

KAYNAKLAR

Affenzeller, M., Winkner, S., Wagner, S., Beham, A., 2009. ***Genetic Algorithms and Genetic Programming- Modern Concepts and Partical Applications***. Numerical Insights. Crc Press.

Akbari, M., Rashidi, H., Alizadeh, S., 2017. An enhanced genetic algorithm with new operators for task scheduling in heterogeneous computing systems. ***Engineering Applications of Artificial Intelligence,*** **61**: 35–46.

Alam, T., Raza, Z., 2018. Quantum genetic algorithm based scheduler for batch of precedence constrained jobs on heterogeneous computing systems. ***The Journal of Systems and Software*** **,135**: 126–142.

Ali, M., Awad, N., Suganthan, P., Shatnawi, A., 2018. An improved class of real-coded genetic algorithms for numerical optimization, ***Neurocomputing,*** **275**: 155-166.

AllenOrr, H., 2005. The Genetic theory of adaptation: a brief story, ***Nature Reviews Genetics,*** **6**: 119-127.

Angeline, P., J., 1996. ***Two Self-Adaptive Crossover Operators for Genetic Programming. Cambrige***, MA. 89-110.

Anonim, 2019. ***http://shodhganga.inflibnet.ac.in/bitstream/10603/41504/12/12_chapter%202.pdf*** . Erişim Tarihi: 02.04.2019.

Back, W., Bohak, I., Ehrmann, M., Ludwig, W., Schleifer, K., H., 1996. Revival of the species lactobacillus lindneri and the design of a species specific oligonucleotide probe, ***System Application Microbiology,*** **19**: 322-325.

Baluja, S., Caruana, R., 2014. Removing the genetics from the standard genetic algorithm, ***Machine Learning Proceeding,*** **10**: 38–46.

Bakirtzis, A., Biskas, P., Zoumas, C., Petridis, V., 2002. Optimal power flow by enhanced genetic algorithm, ***IEEE Transactions on Power Systems***, **17**: 229-236.

Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D., 1998. ***Genetic Programming: An Introduction***, San Francisco: Morgan Kaufmann.

Beasley, D., Bull, D., R., Martin, R., R., 1993. An overview of genetic algorithms. 1. fundamentals. ***University Computing,*** **15** ; 58-96.

Bi, W., Dany, G., C., Maier, H., R., 2015. Improved genetic algorithm optimization of water distribution system design by incorporating domain knowledge. ***Environmental Modelling & Software***, **69**: 370-381

Bies, R.R., Muldoon, M., Pollock, B., Manuck, S., Smith, G., Sale, M., 2006. A genetic algorithm-based, hybrid machine learning approach to model selection. ***Journal of Pharmacokinetics and Pharmacodynamics***, **33**: 196–221.

Bingham, D., 2017 Optimization Test Problems, ***https://www.sfu.ca/~ssurjano/optimizatio n.html***. Erişim Tarihi: 10.04.2019.

Bu, Q., Wang, Z., Tong, X., 2013. An improved genetic algorithm for searching for pollution sources. ***Water Science and Engineering,*** **6**: 392-401.

Budin, L., Golub, M., Budin, A., 2003. ***Traditional Techniques of Genetic Algorithms Applied to Floating-Point Chromosome Representations***. University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb.

Carr, J., 2014. ***An Introduction to Genetic Algorithms***.

Cha, S., Tappert, C., 2009. Genetic algorithm for constructing compact binary decision trees. ***Journal of Pattern Recognition Research***, **48**: 1–13.

Chen, S.Y., Zheng, F., Wu, S.Q., Zhu, Z.Z., 2017. Improved genetic algorithm for crystal structure prediction. ***Current Applied Physics***, **17**: 454-460.

DeJong, K., A., 2006. ***Evolutionary Computation. A Unified Approach***. Cambridge, MA, USA: MIT Press.

Echegoyen, C., Mendiburu, A., Santana, R., Lozano, J. A., 2012. On the taxonomy of optimization problems under estimation of distribution algorithms. ***Evolutionary Computation,* 21**: 471 – 495.

Elsayed, S., Sarker, R., Essam D., 2011. ***Improved Genetic Algorithm for Constrained Optimization***. University of New South Wales, School of Engineering and lnfonnation Technology, Canberra.111-115.

Fogel, D., 2005. ***Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*** (3rd ed.), Piscataway, NJ. 210.

Goldberg, D., 1989. ***Genetic Algorithms in Search, Optimization and Machine Learning***. Reading, MA. 225.

Goldberg, D., 2002. ***The Design of Innovation: Lessons from and for Competent Genetic Algorithms***. Norwell, MA. 120.

Guo, P., Wang X., Han, Y., 2010. The enhanced genetic algorithms for the optimization design. ***3rd International Conference on Biomedical Engineering and Informatics***. 2010, USA. 2990-2994.

Gürbüz, A., 2010. ***Improved Genetic Algorithm***, (Yayınlanmamış Doktora Tezi). Marmara Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Gwiazda. D., T., 2006. ***Genetic Algorithms Reference Vol.1 Crossover for Single-Objective Numerical Optimization Problems***, Tomasz Gwiazda, Lomianki. 157-160.

Harik, G., R., 1997. ***Learning linkage to efficiently solve problems of bounded difficulty using genetic algorithms***, (Yayınlanmamış Doktora Tezi), Ann Arbor, MI.

Haupt R., Haupt, S., E., 1998. ***Practical Genetic Algorithm***, New Jersey. 87-135.

Holland, J., 1992. ***Adaptation in Natural and Artificial Systems***, Cambridge, MA. 54-120.

Hu, M., Huang, G., H., Sun, W., Li, Y., Ding, X., An, C., Zhang, X., Li, T., 2014. Multi-objective ecological reservoir operation based on water quality response models and improved genetic algorithm: a case study in three gorges reservoir, ***China. Engineering Applications of Artificial Intelligence***, **36**; 332–346.

Hu, X., B., Wu, S., F., Jiang, J., 2004. On-line free-flight path optimization based on improved genetic algorithms. ***Engineering Applications of Artificial Intelligence***, **17**; 897–907.

Huan, Q., Yi, G., Zijian, L., Yu, L., Haolong, Z., 2018. Shape optimization of automotive body frame using an improved genetic algorithm optimizer, ***Advances in Engineering Software,* 121**: 235-249.

Huynh, N., Huang, Y., Chien, C., 2018. A hybrid genetic algorithm with 2d encoding for the scheduling of rehabilitation patients. ***Computers & Industrial Engineering,* 125**: 221-231.

Gen, M., Cheng, R., 1996***. Genetic Algorithms and Engineering Design***, 45-67.

Janikow, C., Michalewicz, Z., 1990. Specialized genetic algorithms for numerical optimization problems. ***Proceedings of the International Conference on Tools for AI***, Washington, 6–9 November, 798–804.

Jebari, K., Madiafi, M., 2013. Selection methods for genetic algorithms. ***International Journal of Emerging Science and Engineering***, **4**: 333-334.

Kaya Y., Uyar, M., Tekin, R., 2011. ***A Novel Crossover Operator for Genetic Algorithms: Ring Crossover***.

Király, A., Abonyi, J., 2015. Redesign of the supply of mobile mechanics based on a novel genetic optimization algorithm using google maps api**.** ***Engineering Applications of Artificial Intelligence,*** **38**; 122–130.

Koza, J., 1992. ***Genetic Programming: On the Programming of Computers by Means of Natural Selection***. Cambridge, MA. 74.

Knuth, D., E., 1997. ***The Art of Computer Programming. Volume 1: Fundamental Algorithms, 1st edition***, Addison-Wesley, Reading, MA, 1968.

Margery, G., 1993. ***Exploring Your World: The Adventure of Geography***, Washington, D.C.: National Geographic Society.

McCall, J., 2005. Genetic algorithms for modelling and optimization. ***Journal of Computational and Applied Mathematics,*** **184**: 205-222.

Michalewicz, Z., 1996. ***Genetic Algorithms + Data Structures = Evolution Programs***.

Miller, B.L., Goldberg, D.E., 1995. Genetic algorithms, tournament selection, and the effects of noise**.** ***Complex Systems,*** **9**: 193- 212.

Mitchell, M., 1996. ***An Introduction to Genetic Algorithms***, Cambridge, MA. 24-110.

Mühlenbein, H., Schomisch, D., Born, J., 1991. The parallel genetic algorithm as function optimizer**.** ***Parallel Computing,*** **17**: 619–632.

Park, H., Lee, W., Han, W., Vautrin, A., 2008. Improved genetic algorithm for multidisciplinary optimization of composite laminates. ***Computers and Structures,*** **86**: 1894–1903.

Patrascu, M., Stancu, A., F., Pop, F., 2014. A heterogeneous encoding lifelike genetic algorithm for population evolution modeling and simulation. ***Soft Computing***, **18**: 2565-2579.

Piersma T., Drent J., 2003. Phenotypic flexibility and the evolution of organismal design. ***Trends in Ecology & Evolution,*****18**: 228-233.

Poon, P.W., Carter, J.N., 1995. Genetic algorithm crossover operators for ordering applications. ***Computers & Operations Research***, **22**: 135-147.

Sadeghi, E., Sleno, L., Sabally, K., Khairallah, J., Azadi, B., Rodes, L., Prakash, S., Donelly, DJ., Kubow, S., 2016. Biotransformation of polyphenols in a dynamic multistage gastrointestinal model**.** ***Food Chemistry***, **204**: 453-462.

Sarma, J., DeJong, K., 1997. ***An analysis of local selection algorithms in a spatially structured evolutionary algorithm***. San-Francisco: Morgan Kauffmann.

Sastry, K., Johnson D., D., Goldberk, D., E., 2005. ***Bellon P. Phys Rev***.; 72:0805438-085439.

Samanta, D., 2016. ***Encoding Techniques in Genetic Algorithms***. Indian Institute of Technology Kharagpur, India.

Sivanandam, S. N., Sumathi, S., Deepa, S. N., 2007. ***Introduction to Fuzzy Logic using MATLAB***, Springer- Verlag Berlin Heidelberg.

Sivanandam, S. N., Deepa, S. N., 2008. ***Introduction to Genetic Algorithm***, Berlin: Springer.

Sivanandam, S., N., Deepa, S., N, 2009. Comparative study using genetic algorithm and particle swarm optimization for lower order system modelling, ***International Journal of the Computer, the Internet and Management,*** **17**: 1-10.

Shah, M., S., Chudasama M., P., C., 2011. Chudasama, comparison of parents selection methods of genetic glgorithm for tsp, ***International Journal of Computer Applications***: 85-87.

Shao, G., Shangguan, Y., Tao, J., Zheng, J., Liu, T., Wen, Y., 2018. An improved genetic algorithm for structural optimization of au–ag bimetallic nanoparticles, ***Applied Soft Computing Journal,*** **73**: 39–49

Shen, Y., 2018. Improved chaos genetic algorithm based state of charge determination for lithium batteries in electric vehicles. ***Energy***, **152**: 576-585.

Shir, O., 2012. ***Niching in Evolutionary Algorithms***. Rozenberg, Grzegorz; Bäck, Thomas; Kok, Joost N. Handbook of Natural Computing. Springer Berlin Heidelberg: 1035–1069.

Shrestha, A., Mahmood, A., 2016. ***Improving Genetic Algorithm with Fine-Tuned Crossover and Scaled Architecture***. University of Bridgeport, Department of Computer Science and Engineering, Bridgeport.

Soni, N., Kumar, T., 2014. Study of various mutation operators in genetic algorithms. ***International Journal of Computer Science and Information Technologies***, **5**: 519-521.

Song, S., Zhang, Z., Song, B., Chen, Y., 2019. Improved genetic algorithm with local search for satellite range scheduling system and its application in environmental monitoring, ***Sustainable Computing: Informatics and Systems,*** **21:** 19–27.

Srinivas, M., Patnaik, L.M., 1994. Adaptive probabilities of crossover and mutation in genetic algorithms. ***IEEE Transactions on System, Man and Cybernetics***, **17**: 19-23.

Starkweather, T., S., McDaniel, K., Mathias, D., Whitley, C., 1991. ***A comparison of genetic sequencing operators Proc. 4th International Conference on Genetic Algorithms***, 69-76.

Taherdangkoo, M., Yazdi, M., Bagheri, M. H., 2012. A powerful and efficient evolutionary optimization algorithm based on stem cells algorithm for data clustering. ***Central European Journal of Computer Science***, **2**: 47-59.

Taub, H., Schilling, D., I., 1986. ***Princeples of Communication Systems***, Signapore.

Tian, Z., Yan, Y., Hong, Y., Guo, F., Ye, Z., Li, J., 2018. Improved genetic algorithm for optimization design of a three-dimensional braided composite joint. ***Composite Structures,*** **206**: 668-680.

Ting, C-H. 2005. On the mean convergence time of multi-parent genetic algorithms without selection. ***Advances in Artificial Life*** 403-412.

Umbarkar, A., J., Sheth, P., D., 2015. Corssover operations in genetic algorithm: a review. ***Ictac Journar on Soft Computing***, **6**: 1083-1092.

Wang, K., Salhi, A., Fraga, E., 2003. Cluster analysis and visualisation enhanced genetic algorithm. ***Computer Aided Chemical Engineering***, **15**: 642-647.

Wang, Y., Shen, Y., Zhang, X., Cui, G., Sun, J., 2018. An improved non-dominated sorting genetic algorithm-II (INSGA-II) applied to the design of DNA codewords, ***Mathematics and Computers in Simulation,*** **151**: 131–139.

Whitley, D., 1994, A genetic algorithm tutorial. ***Statistics and Computing,*** **4**: 65-85.

Xing, L., N., Chen, Y., W., Yang, K., W., Hou, F., Shen, X., S., Cai, H., P., 2008. A hybrid approach combining An improved genetic algorithm And optimization strategies for the asymmetric traveling salesman problem. ***Engineering Applications of Artificial Intelligence***, **21**: 1370–1380.

Xu, Y., Li, K., Hu J., Li, K., 2014. Genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues. ***Information Sciences***, **270**: 255–287.

Yadav, S., L., 2017. ***Study of the Various Selection Techniques in Genetic Algorithm***, Department of Computer Science and Engineering, K.R. Mangalam University, Gurgaon.

Yang, H., Hu, X., 2016. Wavelet neural network with improved genetic algorithm for traffic flow time series prediction. ***Optik***, **127**: 8103–8110.

Zhang, J., Chung, H. S., H., Lo, W., 2007. Clustering-based adaptive crossover and mutation probabilities for genetic algorithm. ***IEEE Transactions on Evolutionary Computation,*** **11**: 326-335.

Zheng, Z., Zheng, Z., 2018. Towards an improved heuristic genetic algorithm for static content delivery in cloud storage. ***Computers and Electrical Engineering,*** **69**: 422–434.

ÖZGEÇMİŞ

Ahmet Fatih KAZANKAYA 1993 yılında Ankara'da doğmuş, ilk ve orta öğrenimini Van'da tamamlamıştır. 2017 yılında, Kadir Has Üniversitesi, Bilgisayar Mühendisliği Bölümü'nde lisans eğitimini tamamlamıştır. Yüksek lisansa yine 2017 yılında Van Yüzüncü Yıl Üniversitesi Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda başlamıştır.