T.C.
DÜZCE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HİBRİT ÇOK AMAÇLI RÜZGAR GÜDÜMLÜ OPTİMİZASYON ALGORİTMASI

FETHİYE SULTAN ÖZPEHLİVAN AY

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN
PROF. DR. PAKİZE ERDOĞMUŞ

DÜZCE, 2020

T.C.
DÜZCE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HİBRİT ÇOK AMAÇLI RÜZGAR GÜDÜMLÜ OPTİMİZASYON ALGORİTMASI

Fethiye Sultan ÖZPEHLİVAN AY tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Pakize ERDOĞMUŞ

Düzce Üniversitesi

Jüri Üyeleri

Prof. Dr. Pakize ERDOĞMUŞ
Düzce Üniversitesi _____

Prof. Dr. Cihan KARAKUZU
Bilecik Üniversitesi _____

Prof. Dr. Uğur GÜVENÇ
Düzce Üniversitesi _____

Tez Savunma Tarihi: 09/10/2020

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

09 Ekim 2020

Fethiye Sultan ÖZPEHLİVAN AY

TEŞEKKÜR

Yüksek lisans öğrenimim ve bu tezin hazırlanmasında süresince gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Prof. Dr. Pakize ERDOĞMUŞ'a en içten dileklerimle teşekkür ederim.

Bu çalışma boyunca yardımlarını ve desteklerini esirgemeyen değerli eşim Çağatay AY, sevgili ailem ve çalışma arkadaşlarıma sonsuz teşekkürlerimi sunarım.

09 Ekim 2020 **Fethiye Sultan ÖZPEHLİVAN AY**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ÇİZELGE LİSTESİ

Sayfa No

KISALTMALAR

GWO	Gri Kurt Optimizasyon Algoritması
HÇA-RGO	Hibrit Çok-Amaçlı Rüzgar Güdümlü Optimizasyon Algoritması
HV	Hiperhacim
MOGA	Çok-Amaçlı Genetik Algoritmalar
MOPSO	Çok-Amaçlı Parçacık Sürü Optimizasyonu
NPGA	Yerleştirilmiş Pareto Genetik Algoritma
NSGA	Baskınlanamayan Sıralamalı Genetik Algoritma
PAES	Pareto Arşivlemeli Evrim Stratejisi
PSO	Parçacık Sürü Optimizasyonu
RGO	Rüzgar Güdümlü Optimizasyon Algoritması
S	Boşluk

SİMGELER

α	Alfa
β	Beta
δ	Delta
ω	Omega

ÖZET

HİBRİT ÇOK AMAÇLI RÜZGAR GÜDÜMLÜ OPTİMİZASYON ALGORİTMASI

Fethiye Sultan ÖZPEHLİVAN AY
Düzce Üniversitesi
Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi
Danışman: Prof. Dr. Pakize ERDOĞMUŞ
Ekim 2020, 62 sayfa

Temel anlamda optimizasyon, bir veya birden fazla problemin belirli koşullar altındaki en iyi çözümlerini bulma işlemidir. Günümüzde bu problemlerin çözümü için klasik yöntemler ve sezgisel yöntemler kullanılmaktadır. Sezgisel yöntemlerden biri olan Rüzgar Güdümlü Optimizasyon algoritması, rüzgarın atmosfer içerisindeki hareketini temel alarak atmosferik dinamik eşitlikten yararlanan tek-amaçlı optimizasyon problemlerine çözüm arayan bir algoritmadır. Bu çalışmada çok-amaçlı optimizasyon problemlerinin çözümü için Rüzgar Güdümlü Optimizasyon algoritması yeniden düzenlenmiştir. Çok-amaçlı optimizasyon problemlerinde elde edilen sonuçların gerçek sonuçlara ne kadar yakınsadığı ve bu sonuçların ne kadar çeşitli olduğu kullanılan yöntemlerin performansı hakkında bilgi vermektedir. Baskın olmayan sıralama, ağırlıklı toplam, normal sınır kesişimi gibi metotlar çok-amaçlı optimizasyon problemlerinde sıklıkla kullanılan yaklaşımlardır. Bu yaklaşımlardan bazıları çeşitlilik açısından ön plana çıkarken bazılarının ise en iyi sonuca daha iyi yakınsadığı gözlenmiştir. Bu çalışmanın temel amacı elde edilen çözümleri hem çeşitlilik hem de yakınsama açısından en iyi hale getirmektir. Bu amaç kapsamında baskın olmayan sıralama ve adaptif ızgara yaklaşımları bir arada kullanılarak yeni bir hibrit yaklaşım geliştirilmiştir. Daha iyi bir yakınsama için baskın olmayan sıralama, çeşitlilik için adaptif ızgara yaklaşımı bir arada kullanılmıştır. Geliştirilen bu hibrit yaklaşım test problemleri ve doğrusal olmayan denklem sistemlerinde test edilerek sonuçları literatürde iyi bilinen Baskın Olmayan Sıralamalı Genetik Algoritma (NSGA-II) ve Çok-Amaçlı Parçacık Sürü Optimizasyonu (MOPSO) algoritmaları ile karşılaştırılmıştır. Deneysel sonuçlar incelendiğinde çeşitlilik ve yakınsama performansı açısından geliştirilen hibrit yaklaşımın kabul edilebilir olduğu gözlenmiştir.

Anahtar sözcükler: Çok amaçlı optimizasyon, Optimizasyon, Rüzgar güdümlü optimizasyon algoritması.

ABSTRACT

A HYBRID MULTI-OBJECTIVE WIND DRIVEN OPTIMIZATION ALGORITHM

Fethiye Sultan ÖZPEHLİVAN AY
Düzce University
Graduate School of Natural and Applied Sciences, Department of Computer Engineering
Master's Thesis
Supervisor: Prof. Dr. Pakize ERDOĞMUŞ
October 2020, 62 pages

Basically, optimization is the process of finding the best solutions for one or more problems under certain conditions. Today, classical methods and heuristic methods are used to solve these problems. Wind Driven Optimization algorithm, which is one of the heuristic methods, is an algorithm that seeks solutions for single-objective optimization problems that benefit from atmospheric dynamic equality based on the movement of the wind in the atmosphere. In this study, Wind Driven Optimization algorithm was rearranged to solve multi-objective optimization problems. In multi-objective optimization problems, the performance of the used method depends on how closely the results obtained converge to the actual results and how diverse these results are. Methods such as non-dominant sorting, weighted sum, normal boundary intersection are frequently used approaches in multi-objective optimization problems. While some of these approaches have more diverse to results, others have been observed to better converge. The main purpose of this study is to optimize the solutions obtained in terms of both diversity and convergence. Within this scope, a new hybrid approach has been developed by using non-dominant sorting and adaptive grid approaches. Non-dominant sorting used for better convergence and adaptive grid approach is used for diversity. The developed hybrid approach has been tested in test problems and nonlinear equation systems. Results have been compared with Non-Dominated Sorting Genetic Algorithm (NSGA-II) and Multi-Objective Particle Swarm Optimization (MOPSO). When the experimental results were examined, it was observed that the hybrid approach developed in terms of diversity and convergence performance was acceptable.

Keywords: Multi-objective optimization, Optimization, Wind driven optimization algorithm.

1. GİRİŞ

Teknolojinin gelişmesi ve hesaplama maliyetinin kolaylaşması ile birlikte optimizasyon problemlerinin çözümünde kullanılan klasik yöntemler yerini sezgisel araştırma yöntemlerine bırakmıştır. Ekosistemde yer alan canlıların davranışlarında optimum çözüme ulaşması bilim adamlarının araştırmalarına konu olmakta ve doğadan esinlenen algoritmaların artışı ile sonuçlanmaktadır [1]. Bu nedenle problemlerin çözümünde doğadan esinlenmek geçerlilik kazanmıştır. 1977 yılında optimizasyon alanında Holland tarafından devrim niteliğinde bir fikir ortaya atılmış ve doğadaki evrimselliğin bilgisayarlar aracılığıyla benzetimi yapılarak optimizasyon problemlerinin çözümü sağlanmıştır. Sezgisel algoritmaların en bilineni olan bu yöntem Genetik Algoritma (GA) olarak adlandırılmış ve karmaşık problemlerin çözümüne yeni bir bakış açısı getiren bir çalışma alanı haline gelmiştir [2].

Doğadan esinlenme ve evrimsellik kavramlarıyla birlikte GA'nın yanında birçok optimizasyon algoritması geliştirilmiştir. Parçacık Sürü Optimizasyonu (PSO) [3], Karınca Kolonisi Optimizasyonu (ACO) [4] gibi birçok algoritma tek-amaçlı optimizasyon problemlerini başarı ile çözebilmektedir. Bu algoritmaların düzenlenmiş versiyonları çok-amaçlı optimizasyon problemlerinin çözümünde kullanılmaktadır.

Çok-amaçlı optimizasyon birden fazla amacın en iyilenmesini amaçlar. Ancak bu amaçlar çoğu zaman bir birleri ile çelişirler. Bir fabrikada hem karın maksimize edilmesi, hem de maliyetin minimize edilmesi bir noktada denge durumuna ulaşır. Çok amaçlı optimizasyon problemlerinin çözümünde çeşitlilik ve yakınsama önemli iki kavramdır [5]. Bu amaçla çok amaçlı optimizasyonu için yeniden düzenmiş algoritmaların, problem çözümünde elde ettikleri pareto noktaların optimum çözüme mümkün olduğu kadar yakınsamış ve çeşitlilik göstermiş olması gerekmektedir. Yakınsama ve çeşitliliği sağlamak için baskın olmayan sıralama (non-dominated sorting), adaptif ızgara gibi birçok metot geliştirilmiştir [6], [7].

Genel olarak çok-amaçlı optimizasyon algoritmalarında uygunluk değerleri belirlenirken kümeleme ve pareto baskınlığı [8] tabanlı iki yaklaşım kullanılmaktadır [9]. Goldenberg'in baskınlık kavramını ortaya atmasından sonra birçok araştırmacı aynı

yöntemi kullanarak çok-amaçlı optimizasyon algoritması geliştirmiştir. Fonseca ve Fleming, Çok-Amaçlı Genetik Algoritmalar (MOGAs) için sıralama tabanlı uygunluk atama metodu geliştirmişlerdir [10]. Horn ve arkadaşları çok-amaçlı problemlerin optimize edilmesinde pareto baskınlığından yararlanmış ve GA'yı yeniden düzenleyerek Yerleştirilmiş Pareto Genetik Algoritma (NPGA) adlı yeni bir yöntem sunmuştur [11]. Srinvas ve Deb, daha önce bahsedilen yaklaşımlardan farklı olarak pareto baskınlığı ve sıralama tabanlı yaklaşımları bir arada kullanarak Baskın Olmayan Sıralamalı Genetik Algoritma (NSGA) adlı bir yöntem geliştirmiştir [6]. Geliştirilen bu yöntem daha çok Goldenberg'in yaklaşımına benzerlik göstermektedir. İlerleyen yıllarda Deb, baskınlık tabanlı yaklaşımlarda çok tartışılan hesaplama karmaşıklığını azaltarak NSGA-II algoritmasını öne sürmüştür [12]. Mirjalili ve arkadaşları doğadaki gri kurtların liderlik ve avlanma mekanizmasını taklit eden sezgisel bir yöntem olan Gri Kurt Algoritmasını [13] öne sürmüşler ve bu yöntemi geliştirerek çok-amaçlı problemlerin çözümüne uygun hale getirmişlerdir [14]. Aynı şekilde Saremi ve arkadaşları çekirge sürülerinin doğadaki davranışlarını taklit ederek çok değişkenli optimizasyon problemlerini çözen popülasyon tabanlı Çekirge Optimizasyon Algoritmasını [15] geliştirilmiştir. 2018 yılında Mirjalili ve arkadaşları [16] bu algoritmayı yeniden düzenleyerek çok-amaçlı problemleri çözebilir hale getirmiştir. Zeng ve arkadaşları [17], Reddy ve Kumar [18], Al Jadaan ve arkadaşları [19], Ghiasi ve arkadaşları [20], Nobahari ve arkadaşları [21], Costa ve arkadaşları [9], Salcedo-Sanz ve arkadaşları [22] tarafından önerilen çok-amaçlı optimizasyon yaklaşımlarının hepsi aynı yöntemi kullanarak pareto cephesini daha yakınsamış ve çeşitli hale getirmeye çalışmışlardır.

Yapılan bu çalışmada popülasyon tabanlı Rüzgar Güdümlü Optimizasyon (RGO) algoritmasının çok-amaçlı optimizasyon problemlerinin çözümünde kullanılabilmesini amaçlayan bir uyarlama içermektedir. Literatürde yer alan çok-amaçlı RGO, pareto-optimum noktaların elde edilmesinde hızlı bastırılamayan sıralama yöntemini kullanmaktadır [23]. Bu çalışmada ise Bayraktar ve arkadaşlarının geliştirmiş olduğu çok-amaçlı RGO'ya alternatif olarak, pareto arşivlemeli depolama ve adaptif ızgara yaklaşımı bir arada kullanılarak çözüme daha hızlı ve başarılı yakınsayacak hale getirilmiştir. Geliştirilen Hibrit Çok-Amaçlı Rüzgar Güdümlü Optimizasyon Algoritması (HÇA-RGO) ile pareto-mesafe tabanlı bir yaklaşım geliştirilerek daha çeşitli çözümler elde edilmiştir.

1.1. TEZ ORGANİZASYONU

Tez çalışmasının ikinci bölümünde optimizasyon detaylı bir şekilde açıklanmış ve optimizasyon türleri hakkında bilgi verilmiştir. Üçüncü bölümde tek-amaçlı optimizasyon ve buna bağlı algoritmalar tanıtılmaktadır. Dördüncü bölümde tez çalışmasının temeli oluşturan RGO algoritması tanıtılmıştır. Beşinci bölümde çok-amaçlı optimizasyon anlatılmış ve deneysel çalışmada kullanılan algoritmalar hakkında detaylı bilgi verilmiştir. Altıncı bölümde yapılan çalışma olan HÇA-RGO algoritması detaylı bir şekilde anlatılmaktadır. Son bölümde ise, elde edilen bulgular neticesinde diğer algoritmalar ile karşılaştırmalar yapılmıştır. Elde edilen sonuçlar geliştirilen yöntemin çok-amaçlı optimizasyon problemlerinde çeşitlilik ve yakınsama açısından başarılı olduğunu göstermektedir.

2. OPTİMİZASYON

Temel anlamda optimizasyon; belirli bir amaç veya amaçlar doğrultusunda belirlenmiş kısıtlarla birlikte bir problemin en iyi çözümünü bulma işlemi olarak ifade edilebilir. Bir optimizasyon problemin en iyi çözümünü bulma işlemi olarak ifade edilebilir. Bir optimizasyon problemi Denklem 2.1 ve Denklem 2.2'deki gibi matematiksel olarak modellenebilir.

$$min\, f(x_i)\ i = 1, \dots, n \tag{2.1}$$

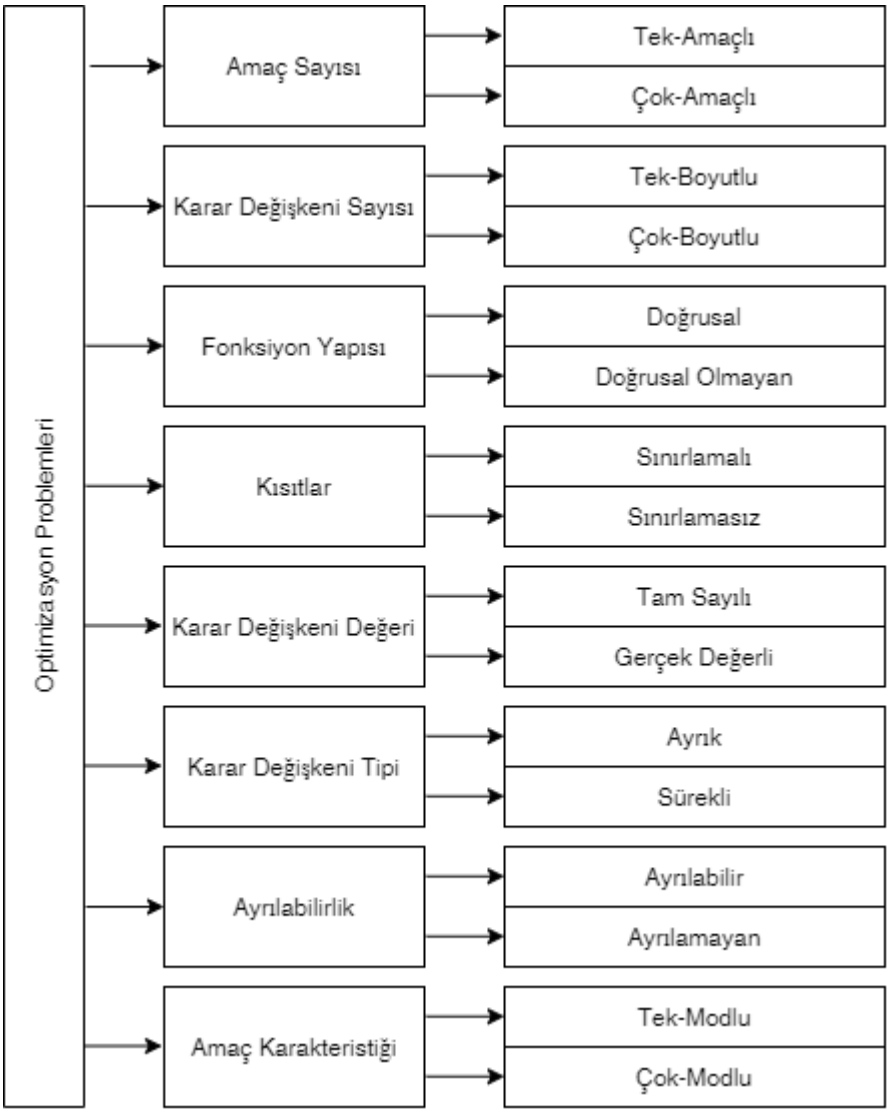
$$g(x_i) \geq 0 \tag{2.2}$$

Denklem 2.1'e göre f amaç fonksiyonunu, x karar verme değişkeni ve Denklem 2.2'de g kısıtları ifade etmektedir. Amaç fonksiyonu modellenen problemin türüne göre maksimum ve minimum değer veya değerlere sahip olabilir. Kısıtlayıcılar ile karar verme değişkenlerinin sınırları belirlenebilir ve bu kısıtlar eşitlik ve eşitsizlik şeklinde olabilir. Bununla birlikte tüm kısıtları sağlayan fonksiyon uygunluk fonksiyonu olarak adlandırılır. Bir problem için en uygun çözüm optimum çözüm olarak ifade edilir. Optimizasyonun temel amacı optimum çözümü bulmaktır. Optimum çözüm problemin türüne göre minimum veya maksimum olabilir.

Optimizasyonun temelleri Newton, Lagrange ve Cauchy tarafından atılmıştır [24]. Newton yapmış olduğu çalışmalar ile diferansiyel yöntemlerin geliştirilmesine katkı sağlarken, Lagrange kısıtlı optimizasyon problemlerini çözen ve kendi adıyla anılan bir yöntem geliştirmiştir. Cauchy ise; kısıtsız optimizasyon problemleri üzerine çalışarak bu alanda bir optimizasyon yöntem geliştirilmiştir. Temellerinin atılmasından sonra oldukça yavaş ilerleyen gelişmeler sayısal bilgisayarların gelişmesi ve hesaplamayı kolaylaştıran icatlar ile birlikte 20. yüzyılın ortalarında tekrar hız kazanmıştır. 20. yüzyılın sonlarına doğru kişisel bilgisayarların yaygınlaşmasına paralel olarak optimizasyon yöntemleri ve kullanım alanları da büyük bir oranda artmıştır.

Bir optimizasyon problemi amaç ve kısıtlama fonksiyonu ile karar verme değişkenlerinin türüne göre sınıflandırılabilir (Şekil 2.1). Optimizasyon problemleri amaç sayısına göre tek-amaçlı ve çok amaçlı olmak üzere ikiye ayrılır. Modellenen problemlerde birden fazla amacın minimum veya maksimum çözümlerin aranması gerekebilir. Problemin türüne

göre birden fazla amacı minimize veya maksimize etmek tek-amaçlı bir problemin optimize edilmesine göre daha zor olabilir.



Şekil 2.1. Optimizasyon problem sınıfları.

Amaç sayısının yanı sıra karar değişkenlerinin sayısı da bir optimizasyon problemi için oldukça önemlidir. Karar değişkenlerinin sayısı problemin boyutu hakkında bilgi verir. Bir optimizasyon problemi tek bir karar değişkenine sahip ise; tek boyutlu, bir den fazla karar değişkenine sahip ise; çok boyutlu optimizasyon problemi olarak adlandırılır. Genel olarak incelendiğinde tek ve çok-amaçlı optimizasyon problemlerinde olduğu gibi çok-boyutlu optimizasyon problemlerinin çözümü tek-boyutlu optimizasyon problemlerine göre daha maliyetli olacaktır.

Amaç ve kısıt fonksiyonlarında ter alan karar değişkenlerinin derecesine göre optimizasyon problemi iki sınıfa ayrılır. Derecesi bir olan optimizasyon problemi

doğrusal, derecesinin birden fazla olması durumunda doğrusal olmayan şeklinde adlandırılır. Yapılan çalışmalar incelendiğinde doğrusal olmayan denklem sistemlerinin optimize edilmesi, doğrusal sistemlere göre daha maliyetli olmaktadır.

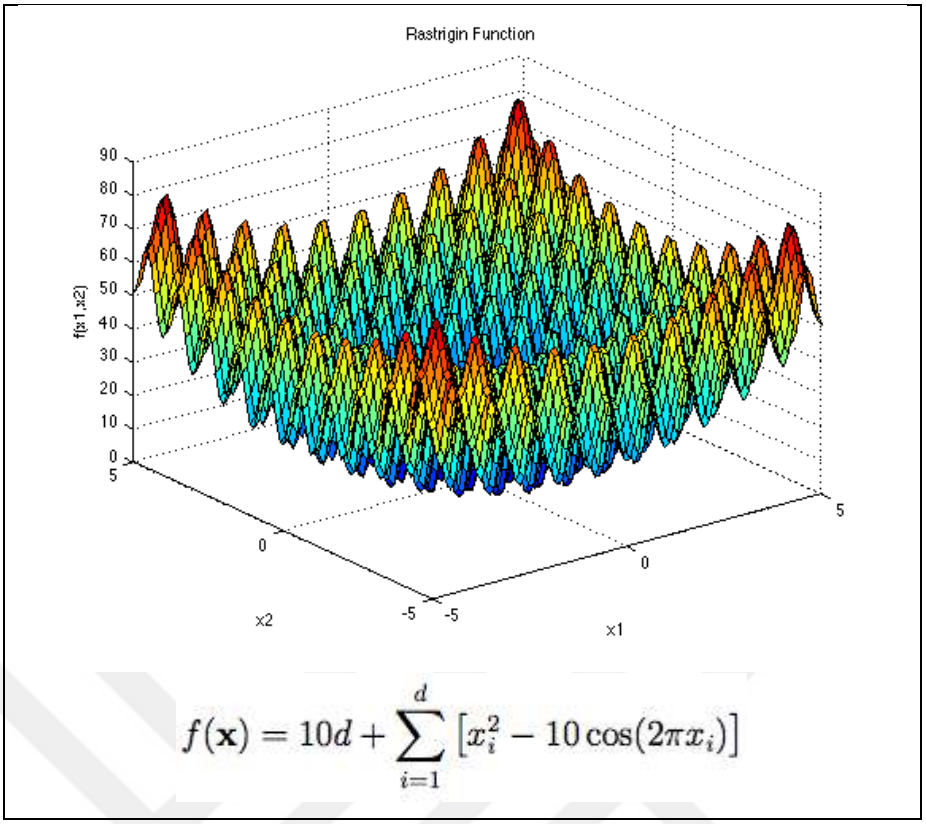
Optimizasyon problemleri amaç fonksiyonunu oluşturan karar değişkenlerinin kısıt durumuna göre iki sınıfa ayrılır. Eğer bir amaç fonksiyonu herhangi bir kısıt fonksiyonu ile sınırlandırılıyorsa kısıtlı, sınırlandırılmıyor ise kısıtsız optimizasyon problemi olarak adlandırılır. Aynı zamanda karar değişkenleri tam sayı olacak şekilde kısıtlanıyorsa tam sayılı programlama, gerçek değer ile kısıtlanıyorsa gerçek değerli programlama olarak iki alt sınıfa ayrılır.

Optimizasyon problemleri alt fonksiyonlara ayrılabilme durumuna göre, ayrılabilir ve ayrılmayan optimizasyon problemi olarak iki sınıfa ayrılır.

Karar verme değişkenlerinin türüne göre optimizasyon problemleri kesikli ve sürekli olmak üzere iki sınıfa ayrılır. En iyi sıralama gruplama, düzenleme, seçilme gibi çözümlerin yer aldığı problemler kesikli optimizasyon problemi olarak adlandırılır. Karar değişkenleri ile ilgili bir kısıt olmaması ve sürekli değerler olması durumunda optimizasyon problemi sürekli olarak adlandırılır.

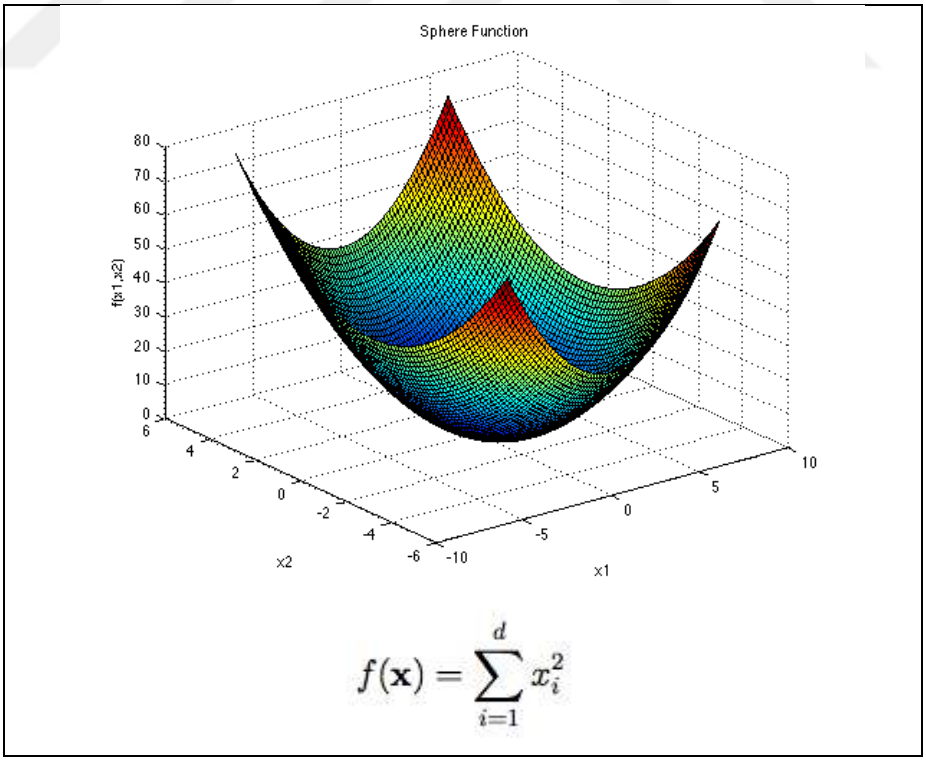
Bazı optimizasyon problemlerinde optimum çözüm birden fazla olabilir. Bir adet global ve birden fazla lokal optimum çözümü olan problemler çok-modlu optimizasyon problemi olarak adlandırılır. Bu problemlerde birden fazla tepe ve dip noktası bulunur. Şekil 2.2'de yer alan Rastrigin Benchmark problemi birden fazla lokal ve tek bir global optimum noktası olan çok-modlu optimizasyon problemlerine örnek olmaktadır.

Tek bir lokal ve global optimum noktası olan optimizasyon problemleri tek-modlu optimizasyon problemi olarak adlandırılır. Şekil 2.3'de tek bir global optimum noktası olan Sphere Benchmark fonksiyonu yer almaktadır.



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

Şekil 2.2. Rastrigin benchmark fonksiyonu.



$$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2$$

Şekil 2.3. Sphere benchmark fonksiyonu.

2.1. KLASİK OPTİMİZASYON

Sürekli ve türevlenebilir fonksiyonların diferansiyel analiz yöntemleri ile optimum çözümlerinin bulunması klasik optimizasyon olarak ifade edilir. Klasik optimizasyonda fonksiyonun yapısı optimum çözüm tarafından belirlenir. Klasik optimizasyon problemleri kısıtlamaların türüne göre eşitlik kısıtlayıcı (equality constraint) ve eşitsizlik kısıtlayıcı (inequality constraint) olarak iki sınıfa ayrılır ve optimum çözümünün elde edilmesinde farklı yöntemler kullanılır. Sürekli olmayan ve türevlenemeyen problemler için klasik optimizasyon pratikte sınırlıdır.

2.2. SEZGİSEL OPTİMİZASYON

Optimizasyon problemlerinin büyük bir kısmı karmaşık ve çözümlemesi oldukça zor problemlerdir. Bu tarz problemlerin geçerli bir zaman içerisinde çözümlenmesi mümkün olmayabilir. Gerçek zamanlı uygulamalar için zaman oldukça önemli bir etkendir. Zaman ve karmaşıklık faktörlerinden dolayı bu tarz problemlerin çözümünde sonuca kesin ulaşmak yerine yaklaşık çözümler kabul edilmiş ve yaklaşık algoritmalar kullanılmaya başlanmıştır. Sezgisel optimizasyon klasik optimizasyon yöntemlerinin uygun bir çözüm üretemediği problemler için kabul edilebilir zaman dilimi içerisinde yaklaşık çözüm elde etme işlemidir. Bir başka ifade ile belirli bir amaca ulaşmak veya hedefe varmak için doğal olaylardan esinlenmek sezgisel optimizasyon olarak adlandırılır. Sezgisel optimizasyonda kesin çözüm garanti edilmez. Bunun yerine problemin optimum çözüme en iyi şekilde yakınsamayı amaçlar.

Sezgisel optimizasyon için bir çok yöntem geliştirilmiştir. Sezgisel yöntemlerin en belirgin özelliği arama uzayının farklı noktalarında çözüm arayabilmesidir.

Sezgisel yöntemler fizik, biyolojik, kimya, sosyal ve müzik tabanlı olmak üzere altı başlık altında incelenebilir. Bunların yanı sıra hibrit yaklaşımlarda mevcuttur [25].

Sezgisel yöntemlerin işleyişi genel olarak dört adımda listelenebilir:

- Rastgele başlangıç değerleri ile rastgele çözümlerin elde edilmesi
- Parametrelerin bağlı olarak çözümlere ait komşulukların araştırılması
- Anlık durum güncellemesi
- Kurallar çerçevesinde kabul edilebilir çözüme kadar sürecin tekrar edilmesi

3. TEK-AMAÇLI OPTİMİZASYON

Tek-amaçlı optimizasyon matematiksel olarak bir problemin tek bir amaç fonksiyonu olacak şekilde modellenmesidir. Tek-amaçlı optimizasyonun temel amacı modellenmiş bir amacın belli koşullar altında en iyi çözümü bulmaktır. Bu çözüm minimum veya maksimum olabilir. Maliyet ve zamanı en aza indirme, üretim ve karı maksimize etme gibi problemler tek-amaçlı optimizasyon problemledir.

3.1. TEK-AMAÇLI OPTİMİZASYON ALGORİTMALARI

3.1.1. Parçacık Sürü Optimizasyonu (PSO)

Parçacık Sürü Optimizasyonu Algoritması, 1995 yılında Russell C. Eberhart ve James Kennedy tarafından önerilen doğadaki kuş ve balıkların sürü psikolojisinden yararlanılan sürü tabanlı sezgisel bir algoritmadır [3]. Algoritma temel olarak balık ve kuş sürülerinin yanı sıra sürü psikolojisine sahip çeşitli hayvanların yaşamsal faaliyetlerini sürdürebilmek için geliştirmiş oldukları davranışsal hareketleri baz almaktadır. Bu davranışlar sosyal, bilişsel ve keşifsel olarak üçe ayrılır. Yiyecek aramakta olan bir sürü için sürü içerisinde yer alan her bir bireyin hafızalarında son belirledikleri yiyecek kaynağına hareket etmesi bilişsel davranışı; kaynağı bulan bir bireyi takip etme biçimi sosyal davranışı; son olarak rastgele bir şekilde kaynak arama biçimi keşifsel davranış olarak adlandırılır. Algoritma bu üç davranışsal modeli bir arada kullanarak kaynağa ulaşmayı hedefler [26].

PSO algoritmasında sürüyü oluşturan her bir birey parçacık, sürü ise popülasyon olarak adlandırılır [3], [27].

Bir başka ifade ile belirli bir amaca ulaşmak veya hedefe varmak için doğal olaylardan esinlenerek sezgisel optimizasyon olarak adlandırılır. Sezgisel optimizasyonda kesin çözüm garanti edilmez. Bunun yerine problemin optimizasyon çözümüne en iyi şekilde yakınsamasına amaçlanır.

3.1.1.1. *Konum Hesaplama*

Davranışsal modelde yer alan bilişsel davranış algoritma içerisinde yerel en iyi , sosyal davranış ise global en iyi olarak adlandırılır.

Parçacıklara ait sonraki konum bilgisi hesaplanırken her bir parçacığın o ana kadar sahip olduğu en iyi değer yerel en iyi, tüm parçacıkların birbirleri ile kıyaslanması sonucu elde edilen en iyi değer ise küresel en iyiyi ifade eder. Güncel konum bilgisi için parçacığın bir önceki konumu ve o an ki hız bilgisi kullanılmaktadır. Anlık hız bilgisi Denkem 3.1'de güncel konum bilgisi ise Denklem 3.2'de gösterilmektedir.

$$v_i^d(k+1) = wv_i^d(k) + c_1 r_1\left[x^d_{pbest_i} - x_i^d(k)\right] + c_2 r_2\left[x^d_{gbest} - x_i^d(k)\right] = \mathrm{w}v_i^d(k) \quad (3.1)$$

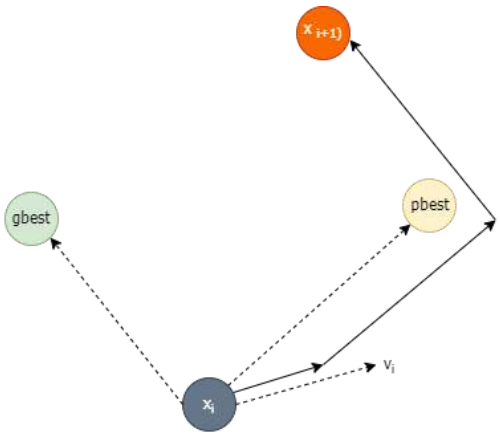
$$x_i^d(k+1) = x_i^d(k)v_i^d(k)\Delta t \quad (3.2)$$

Denklem 3.1 ve 3.2'e göre x parçacığın bir sonraki hızı, v ise parçacığın güncellenmiş hızını ifade eder. Popülasyonda yer alan her parçacık i ile numaralandırılır. PSO iterasyon tabanlı bir algoritmadır. Denklemlerde yer alan k iterasyonun o an ki sayısını ifade eder. Hız denklemlerinde yer alan r_1 ve r_2 0 ile 1 arasında rastgelene değer alan keşifsel davranış için kullanılan katsayılardır. Aynı şekilde bilişsel ve sosyal davranışların ağırlıklandırılmasında c_1 ve c_2 katsayıları kullanılmaktadır. Clerc ve Kennedy'nin yapmış oldukları çalışmada bu katsayılar için kullanılacak en uygun değerin 1.494 olduğu önerilmektedir [28]. w sürünün başlangıçta geniş bir alanı taraması, ilerleyen aşamada ise çözüm uzayını daraltmakta kullanılan değişken bir çarpandır. w çarpanı Denklem 3.3'de gösterilmektedir.

$$w = w_{max} - \frac{w_{max} - w_{min}}{k_{max}} k \quad (3.3)$$

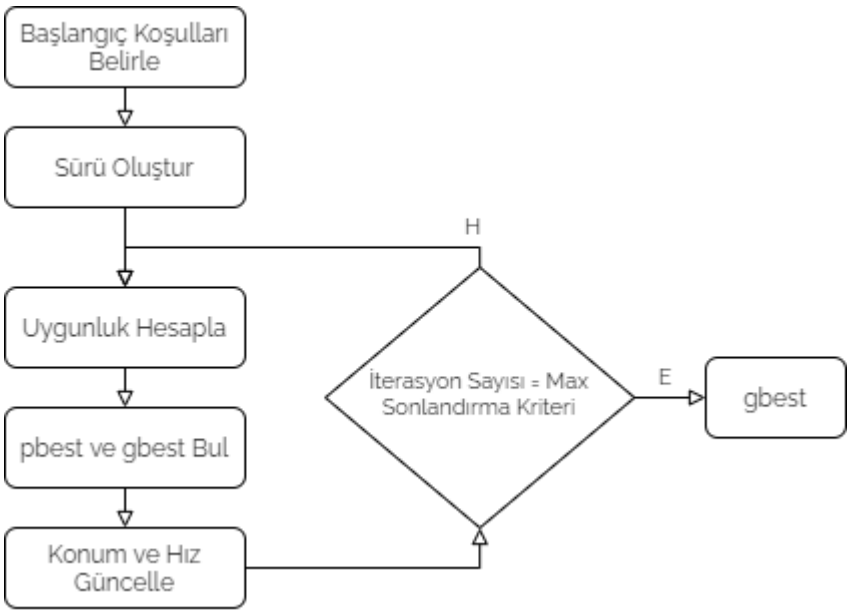
Eşitlik yer olan k_{max} maksimum iterasyon sayısı, k ise iterasyonun o anki değerini ifade eder. w_{max}ve w_{min} genel olarak Shi ve Eberhart tarafından yapılan bir çalışma baz alınarak 0.4 ve 0.9 olmaktadır [29].

PSO algoritmasının parçacık konumlandırma işlemi Şekil 3.1'de gösterilmektedir.



Şekil 3.1. PSO parçacık konum ve hız güncelleme.

Şekil 3.1'e göre parçacığın güncel konumu küresel en iyi $gbest$, yerel en iyi $pbest$ ve parçacığa ait anlık hızının vektörel olarak toplanmasıyla elde edilir. Herhangi bir sonlandırma kriteri veya iterasyonun sonuna kadar bu işlemler devam eder. Bunun sonucunda elde edilen küresel en iyi optimum çözüm olarak ifade edilir. Algoritmaya ait akış diyagramı Şekil 3.2'de gösterilmektedir.



Şekil 3.2. PSO akış şeması.

3.1.2. Genetik Algoritma (GA)

Genetik algoritmalar karmaşık çözüm uzayına sahip problemleri çözmek için doğal seleksiyon ve evrim teorisinden yararlanan deterministik olmayan stokastik arama optimizasyon algoritmaladır. Genetik algoritmayı daha iyi anlamak için Darwin'in evrim teorisi hakkında bilgi sahibi olmak gerekir. Genetik algoritmanın temelinde doğal çevreye en iyi şekilde uyum sağlayan ve en zeki olarak adlandırılan bireylerin hayatta kalması, bunların dışında kalıp zorluk çeken bireylerin ise ayıklanması vardır. Bununla birlikte hayatta kalan bireylerin özelliklerini yeni kuşaklara aktarılarak en üstün bireylerin elde edilmesi amaçlanır. Sürecin sonunda doğal çevreye uyum sağlamış güçlü bireylerden oluşan bir popülasyon elde edilmiş olur.

Genetik Algoritma, PSO'ya benzer şekilde popülasyon tabanlı yinelemeli bir algoritmadır ve en iyi bireyin hayatta kalmasını amaçlar.

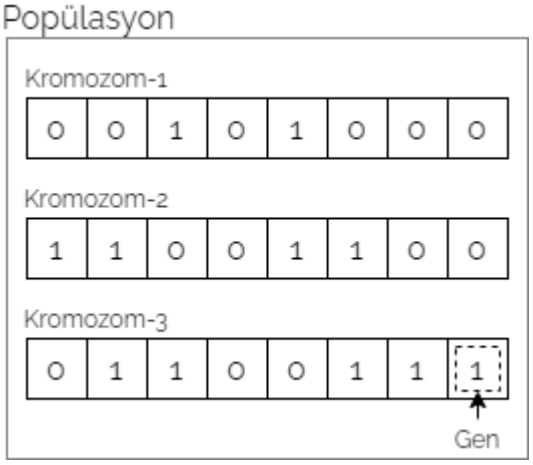
Evrimsel teoriyi temel alan algoritmalar ile karmaşık yapıya ve kısıtlara sahip zorluk derecesi yüksek problemlerin çözüm maliyetinin azaltılması istenmiştir. Holland

tarafından genetik algoritmanın temellerinin atılmasıyla popüler hale gelmiştir [30].

Yapılan çalışmalar Genetik Algoritmanın klasik yöntemlere göre karmaşık problemlerinin çözümünde daha başarılı olduğunu göstermiştir. Bununla birlikte robotik, görüntü işleme, yapay zekâ vb. birçok alanda bilim insanları tarafından tercih edilmiştir.

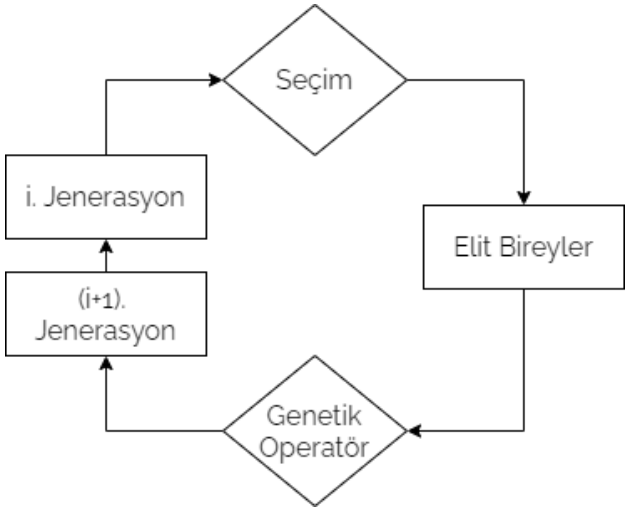
Genetik algoritmayı diğer klasik yöntemlerden ayıran en önemli özelliği çözüm uzayının belirli bir kısmını tarayarak optimum çözüme en kısa sürede ulaştırmasıdır.

Genetik algoritmayı oluşturan yapı taşları; gen, kromozon ve popülasyon olmak üzere üçe ayrılır. Burada çözüm kümesi popülasyon olarak adlandırılır ve popülasyon kromozomlar tarafından oluşturulur. Kromozomlar ise gen adı verilen çözüm paramatrelerinden oluşur. Bir başka ifade ile genlerin kromozomları, kromozomların ise popülasyonun oluşturduğu hiyerarşik bir yapı söz konusudur (Şekil 3.3).



Şekil 3.3. Genetik Algoritma yapı.

Genetik algoritmada bir nesli oluşturan her bir kromozom için uygunluk değeri hesaplanarak elit bireyler elde edilir. Bu elit bireyler çaprazlama ve mutasyon gibi genetik operatörler yardımıyla yeni popülasyonun oluşturulmasını sağlar. Bu evrimsel döngü olarak ifade edilir. Başlangıçta rastgele oluşturulan kromozomlar ile evrim sürecinin sonunda elit bireyler elde edilir.



Şekil 3.4. Genetik Algoritma evrim süreci.

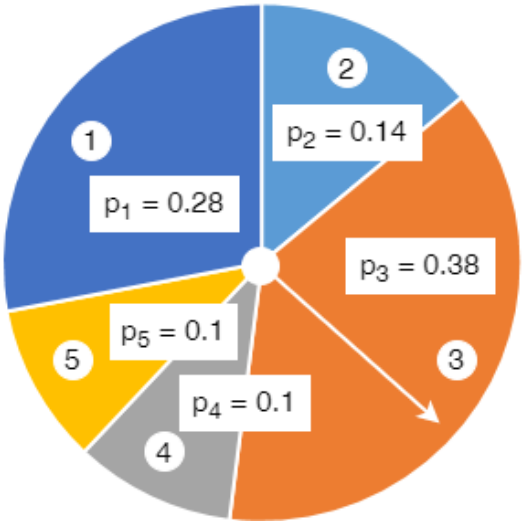
3.1.2.1. *Seçim İşlemi*

Genetik algoritma ve birçok algoritmanın ilk adımı başlangıç koşullarının oluşturulmasıdır. Genetik algoritmanın bu aşamasında rastgele kromozomlardan bir popülasyon oluşturulur. Daha sonra bu popülasyonda yer alan her bir kromozom için uygunluk değeri hesaplanır. Bu uygunluk değeri kromozomların hayatta kalma ihtimalini belirler. Uygunluk değeri kullanılarak hayatta kalacak bireyler için seçim işlemi yapılır. Seçim işlemi için birden fazla yöntem bulunmaktadır. Bunlardan en çok tercih edileni turnuva ve rulet tekerleği yöntemleridir.

Turnuva seçim yönteminde rastgele kromozomlardan oluşan gruplar oluşturulur. Bu gruplar içerisinde yine rastgele seçilen birkaç kromozom uygunluk değerine göre yarıştırılır ve grup lideri seçilir. Her bir grup için aynı işlemler uygulanır. Bu işlemlerin sonunda her bir grup lideri genetik faktörler kullanılarak yeni nesiller oluşturulmak üzere seçilmiş olur.

Rulet tekerleği turnuva seçim yöntemine göre uygulanması daha zordur. Bu sebepten dolayı daha az tercih edilir. Bu yöntemde kromozomların uygunluk değerine göre olasılık hesabı yapılır. Uygunluk değeri yüksek olan bireylerin seçilme ihtimali aynı oranda yüksektir [31].

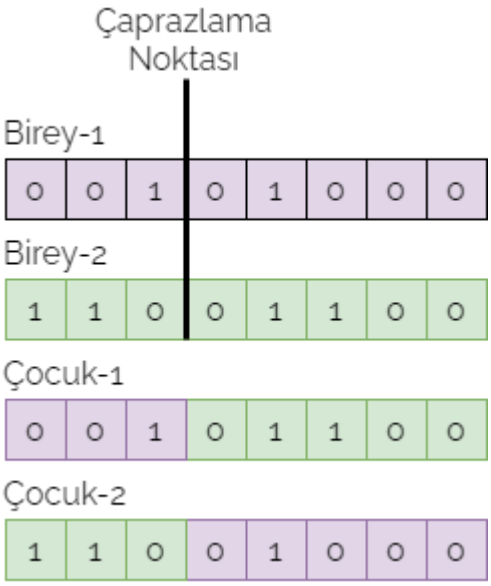
Tüm kromozomların kümülatif olasılığı hesaplanır ve bir olasılık tekerleği oluşturulur (Şekil 3.5). Seçilmesi planlanan kromozom sayısı kadar 0 ile 1 arasında rastgele değer üretilir. Bu değerleri kapsayan kromozomlar yeni nesli oluşturmak üzere lider olarak seçilir.



Şekil 3.5. Rulet tekerleği.

3.1.2.2. *Çaprazlama*

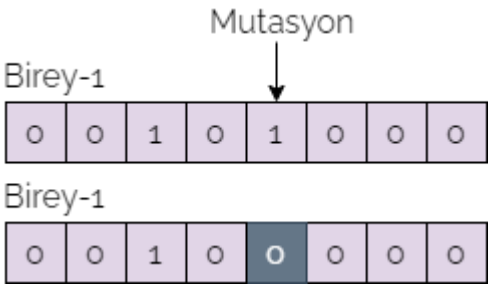
Çaprazlama işleminde seçilmiş liderler arasında gen değişimi yapılarak yeni çocuk bireyler elde edilir. Bu aşamada ilk olarak çaprazlama işlemi için yeni liderler arasından iki kromozom seçilir. Daha sonra kromozomu oluşturan genler arasında rastgele bir gen seçilerek çaprazlama noktası belirlenir. Bu çaprazlama noktası temel alınarak iki kromozom arasında gen değişimi yapılır (Şekil 3.6). Oluşan çocuk bireylerin ebeveynlerinden daha iyi olması beklenir. Çaprazlama işlemi tek noktalı ve çift noktalı olmak üzere iki farklı şekilde yapılabilir.



Şekil 3.6. Çaprazlama.

3.1.2.3. *Mutasyon*

Bir diğer genetik operatör olan mutasyon işleminde çaprazlama işlemi sonucu oluşan çocuk bireylerde bazı genler değişime uğrar. Bu değişim mutasyon katsayısı adı verilen değişkene göre belirlenir. Çocuk bireyler üzerinde rastgele bir gen seçilir. Daha sonra 0 ile 1 arasında rastgele bir sayı üretilir. Eğer bu sayı mutasyon katsayısından büyük ise o gen 0 ise 1, 1 ise 0 olacak şekilde mutasyona uğrar.



Şekil 3.7. Mutasyon.

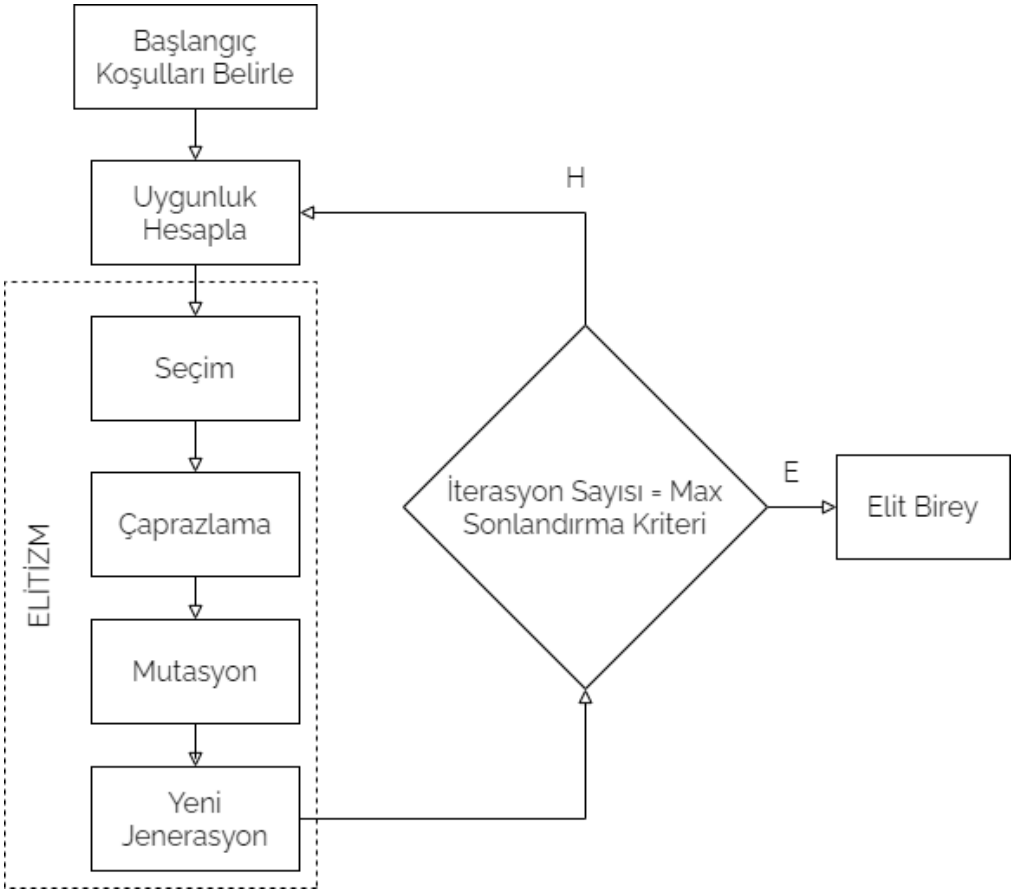
Mutasyon işlemi çeşitliliği azaltmamak için az sıklıkta yapılmaktadır. Yineleme işleminin sonunda bireyler birbirine benzemeye başladığından çaprazlama etkisini yitirmektedir. Bu aşamada arama uzayını genişletmek için mutasyon işlemi oldukça önemlidir [32].

3.1.2.4. *Elitizim*

Genetik algoritmada, yeni neslin oluşturulmasında kullanılan seçilme, çaprazlama ve mutasyon adımları elitizm aşaması olarak adlandırılır. Bu süreç sonunda elde edilen bireyler elit birey olarak ifade edilir. Elit bireylerin elde edilmesi sonlandırma kriteri ya da yineleme sayısı kadar devam eder. Elde edilen bireyler arasından en elit birey seçilerek optimum çözüme ulaşılmış olur.

Algoritmanın temeli oluştururken popülasyon sayısı dikkatlice belirlenmelidir. Popülasyon sayısının olması gerekenden küçük olması arama uzayını azaltacağından yerel noktalara takılma ihtimalini arttıracaktır. Popülasyon sayısının çok yüksek olması ise arama uzayını genişletecek fakat optimum çözüme ulaşmayı aynı oranda da geciktirecektir.

Algoritmanın genel akış şeması Şekil 3.8'de gösterilmiştir. Buna göre; başlangıç olarak rastgele bireylerden oluşan bir popülasyonda doğal yaşama en uygun lider bireyler seçilir. Bu bireyler arasında sırayla çaprazlama ve mutasyon yapılarak bu bireylerin en iyi özelliklerini alan yeni elit bireyler belirlenir. Buna optimum çözüm adı verilir. Ve algoritma bu aşamada sonlandırılır.



Şekil 3.8. Genetik Algoritma akış diyagramı.

3.1.3. Gri Kurt Optimizasyon Algoritması (GWO)

Gri Kurt Optimizasyon Algoritması, S. Mirjalili ve arkadaşları tarafından 2014 yılında öne sürülen doğadaki gri kurtların sosyal davranış ve avlanma stratejisinin matematiksel olarak modellenmesiyle optimizasyon işlemi gerçekleştiren popülasyon tabanlı bir algoritmadır [13].

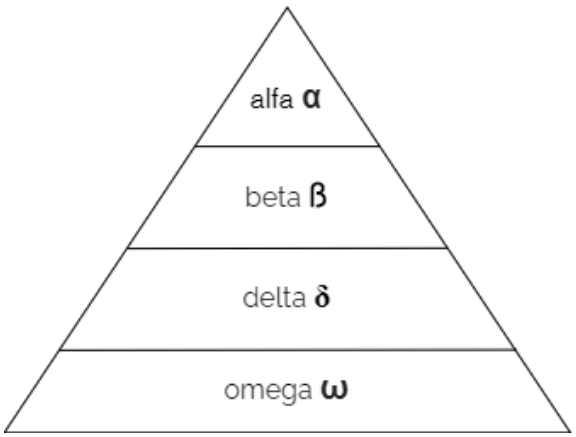
Gri kurtlar grup halinde yaşar ve kendi aralarında belli bir hiyerarşisi mevcuttur. Bu hiyerarşinin en tepesinde alfa olarak adlandırılan lider kurtlar yer alır. Alfa sürüdeki en baskın kurttur. Sürünün avlanma, dinlenme, uyuma ve konaklama gibi önemli kararları alfa kurt tarafından alınır ve sürüye aktarılır [33]. Sürüyü yönetmekten sorumlu alfa kurtları sürünün en güçlü bireyi olmak zorunda değildir. Buradan sürüyü yönetmek fiziksel anlamda güçlü olmaktan daha önemli olduğu sonucu çıkarılabilir.

Piramit'in ikinci basamağı beta olarak adlandırılır. Beta'nın sürüdeki en temek göre alfa tarafından alınacak kararlara yardım etmektir. Aynı zamanda sürüdeki diğer faaliyetlerden sorumludur. Alfa'nın ölmesi veya karar alamayacak duruma gelmesiyle

Beta, alfa görevi için sürüdeki en uygun kurt konumundadır. Beta sürüdeki diğer kurtlar ile alfa arasından tampon olarak görev yapar.

Piramit'in üçüncü basamağında delta kurtları yer alır. Hiyerarşik sistemin temel prensibi olarak üst basamaktan alınan kararları alt basamaktaki kurtlara iletir. İzci, bekçi, gözcü, avcı ve yaşlı kurtlar deltayı oluşturur. İzci kurtlar bölge denetlemesi yaparak sürüyü olası tehlikelerden korur. Aynı şekilde gözcüler sürünün güvenli bir şekilde hareket etmesini sağlar. Avcılar av durumunda alfa ve beta kurtlarına yardım eder ve sürünün beslenmesinden sorumludur. Bekçiler sürüdeki zayıf kurtlar ile ilgilenmekten sorumlu olan kurtlardır. Son olarak yaşlılar, alfa veya beta olmuş tecrübeli kurtlardır.

Piramit'in en alt basamağında omega kurtlar yer alır. Omega, alfa, beta ve deltadan aldıkları emirleri uygular ve sürüye her durumda yardım eder. Gri kurtların sosyal davranış hiyerarşisi Şekil 3.9'te gösterilmiştir. Hiyerarşide baskınlık aşağıdan yukarıya doğru artar.



Şekil 3.9. GWO sosyal davranış hiyerarşisi.

GWO algoritması kurtların avlanma stratejisinden esinlenir. Bu strateji de gri kurtlar önce avın konumunu belirler ve sonrasında alfa liderliğinde avın etrafı sarılır. GWO'nun matematiksel modellenmesinde avın konumu hakkında en iyi bilgilerin alfa, beta ve delta tarafından sağlandığı varsayılır. Bu varsayımda beta ve delta olur. Diğer bütün çözümler ise omegadır.

3.1.3.1. *Kuşatma*

Gri kurtların avlanma sırasındaki kuşatma davranışı aşağıda verilen denklemlerdeki gibi matematiksel olarak modellenmiştir.

$$w = \alpha, \beta, \delta \tag{3.4}$$

$$\overrightarrow{D_w} = \left|\overrightarrow{C_w} * \overrightarrow{X_w} - \overrightarrow{X_\iota}\right| \tag{3.5}$$

$$\overrightarrow{U_w} = \overrightarrow{X_w} - \overrightarrow{A_w} * \overrightarrow{D_w} \tag{3.6}$$

$$\overrightarrow{X_\iota} = \frac{\sum \overrightarrow{U_w}}{3} \tag{3.7}$$

Matematiksel modellemeye göre; i iterasyon sayısını, $\overrightarrow{X_w}$ alfa, beta ve delta tarafından belirlenen avın konumunu $\overrightarrow{X_\iota}$ vektörü ise gri kurdun o anki konumunu ifade etmektedir. $\overrightarrow{A_w}$ ve $\overrightarrow{C_w}$ vektörleri alfa, beta ve delta kurtları için katsayı vektörüdür ve Denklem 3.8 ile Denklem 3.9'da matematiksel olarak hesaplanarak gösterilmiştir.

$$\overrightarrow{A_w} = 2 * \vec{a} * \overrightarrow{r_{1w}} - \vec{a} \tag{3.8}$$

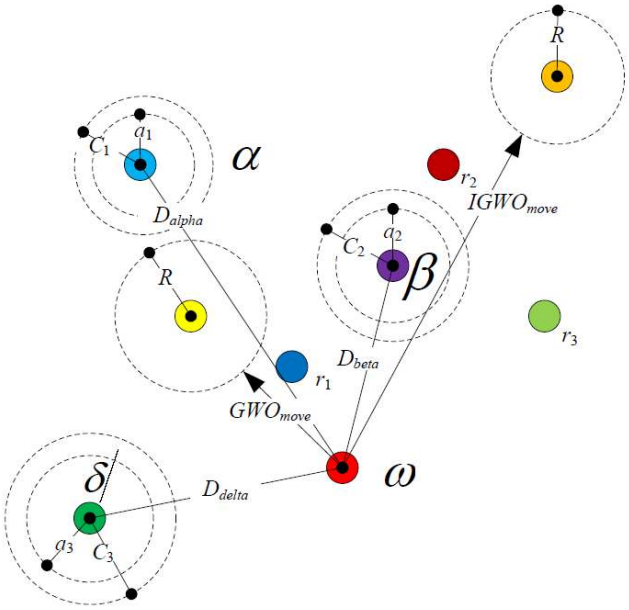
$$\overrightarrow{C_w} = 2 * \overrightarrow{r_{2w}} \tag{3.9}$$

$\overrightarrow{r_1}$ ve $\overrightarrow{r_2}$ vektörleri lider kurtlar için (alfa, beta ve delta) için 0 ile 1 arasında rastgele sayılardan oluşan vektörlerdir. $\vec{a}$ vektörü optimizasyon sürecinde 2'den 0'a doğrusal şekilde azalır.

Matematiksel modelleme sonucunda lider kurtlar ile av arasındaki mesafe $\vec{D}$ hesaplanır ve bu vektör kullanılarak kurtların yeni konumu $\overrightarrow{X_{\iota+1}}$ belirlenir.]

3.1.3.2. *Avlanma*

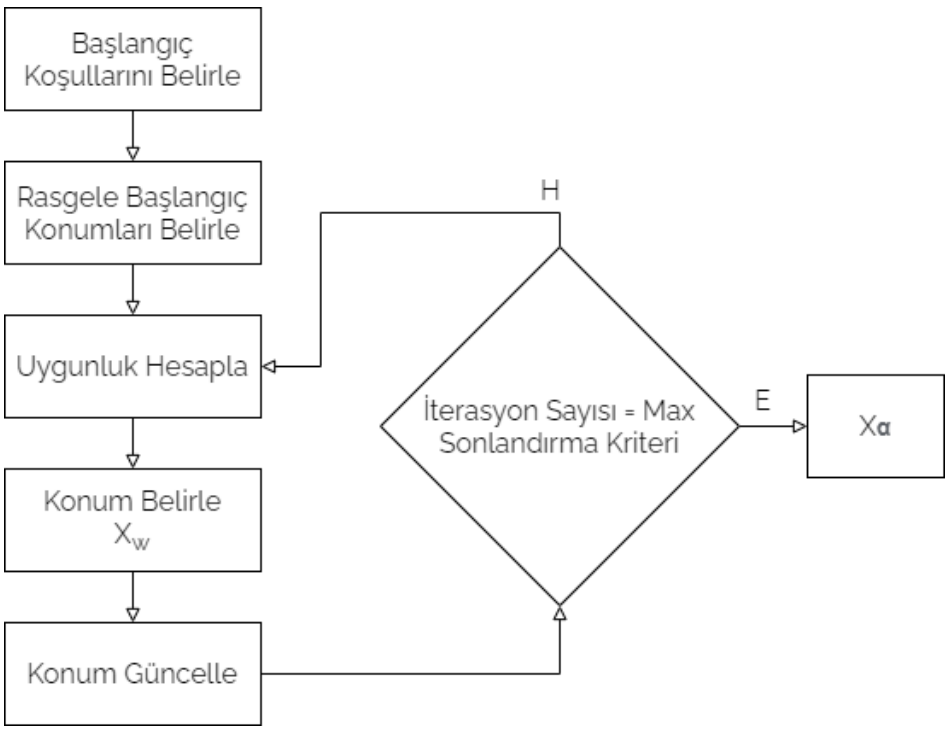
Gri kurtların avlanma süreci genellikle alfa kurt tarafından yönetilir. Bazı durumlarda beta ve delta bu süreci genellikle alfa kurt tarafından yönetilir. Bazı durumlarda beta ve delta da bu sürece dâhil olabilir. Av sürecinin matematiksel modellenmesinde alfa, beta ve deltanın avın muhtemel konumu hakkında en iyi bilgiye sahip olduğu varsayılır. Elde edilen en iyi üç çözüm diğer kurtların konumlarını güncellemek için kullanılır. Bir başka ifade ile en iyi üç konum alfa, beta ve delta olarak saklanır ve diğer tüm kurtlar bu konumlara doğru rastgele hareket eder (Şekil 3.10).



Şekil 3.10. GWO konum güncelleme [34].

3.1.3.3. *Saldırma*

Gri kurtların avlanma süreci ava saldırma ile son bulur. Bu aşamada ava sistematik yaklaşma söz konusudur. Matematiksel modelleme $\vec{a}$ vektörü 2'den 0'a doğrusal olarak azaltılarak ava saldırma işlemi gerçekleştirilir. GWO akış diyagramı Şekil 3.11'de gösterilmiştir.



Şekil 3.11. GWO akış diyagramı.

4. RÜZGAR GÜDÜMLÜ OPTİMİZASYON ALGORİTMASI

Bayraktar ve arkadaşları tarafından 2010 yılında geliştirilen RGO, doğadan esinlenen popülasyon tabanlı bir global optimizasyon algoritmasıdır [35]. RGO, hidrostatik denge içerisinde yer alan, atmosferik dinamik eşitlikten türetilmiş ve rüzgarın atmosfer içerisindeki hareketinden etkilenmiştir. Rüzgarı oluşturan en küçük parçacığa hava parseli denilmektedir. RGO algoritması bu sonsuz sayıda hava parselinin fiziksel denklemlere dayalı olacak şekilde hareketini temel alır. Bu hava parselleri N-boyutlu bir problem uzayında rasgele dağıtılır ve anlık konum ve hız bilgisi Newton'un ikinci hareket kanunu temel alınarak her yinelemede güncellenmektedir. Bir hava parseline uygulanan kuvvetlerin toplamı hava parselinin o anki hızı ve bir sonraki konumunu ifade etmektedir.

Rüzgar, atmosferde hava basıncındaki dengesizliği dengelemek için gerçekleşen bir doğa olayıdır. Bir başka şekilde ifade etmek gerekirse, yüksek basınçtan alçak basınca belirli bir basınç gradyanı ile doğru orantılı olacak şekilde hava parsellerinin hareketi rüzgar olarak adlandırılır. RGO, rüzgarın bu hareketini temel alarak çok-boyutlu problemlere çözüm arayan bir algoritmadır.

RGO algoritmasının başlangıç noktası, atmosferik hareket analizine uyarlandığında etkili sonuçlar veren Newton'un ikinci hareket kanunudur. Newton'un ikinci kanununa göre bir nesne üzerine uygulanan kuvvet ile nesneye kazandıracağı ivme arasında bir bağlantı vardır. Bir başka ifade ile bir cismin ivmesi, ona etki eden bileşke kuvvet ile doğru orantılıdır. Atmosferik hareket analizine uyarlanan Newton'un ikinci kanunu Denklem 4.1'de gösterilmiştir.

$$\rho\vec{a} = \sum F_i \tag{4.1}$$

Yukarıdaki denklemde ρ hava yoğunluğu, $\vec{a}$ ivme vektörü, F_i ise kütleye etki eden bileşke kuvvet olarak ifade edilmektedir. Hava yoğunluğu ideal gaz denklemine göre hava basıncı ile doğrudan ilişkilidir. İdeal gaz denklemi Denklem 4.2'de gösterilmiştir.

$$P = \rho RT \tag{4.2}$$

Denkleme göre P hava basıncı, R evrensel gaz sabiti, T ise sıcaklığı ifade etmektedir.

Denklem 4.1'de, rüzgarın belirli bir yönde hareket etmesine neden olan veya onu yolundan saptıran dört ana kuvvet vardır. Havanın hareket etmesine neden olan en

gözlemlenebilir kuvvetler basınç (Denklem 4.3) ve buna karşı koyan sürtünme kuvvetidir. Sürtünme kuvveti oldukça karmaşık olduğundan dolayı RGO algoritmasında basitleştirilmiş şekilde kullanılmaktadır (Denklem 4.4). Bir başka etken olan yerçekimi kuvveti, 3-boyutlu bir uzayda dikey bir kuvvet olsa da RGO algoritması N-boyutlu bir uzay olduğundan koordinat sisteminin merkezine doğru çeken bir kuvvet haline gelmektedir (Denklem 4.5). Coriolis kuvveti bir başka ifade ile savrulma kuvveti, dünyanın dönmesinden kaynaklanan ve birim kütleye etki eden saptırıcı güç olarak adlandırılabilir. RGO algoritmasında bu güç rüzgarı oluşturan hava parselinin hızına doğrudan etki eden kuvvet olarak kullanılmıştır (Denklem 4.6).

$$\overrightarrow{F_{PG}} = -\nabla \mathrm{P}\delta \mathrm{V} \tag{4.3}$$

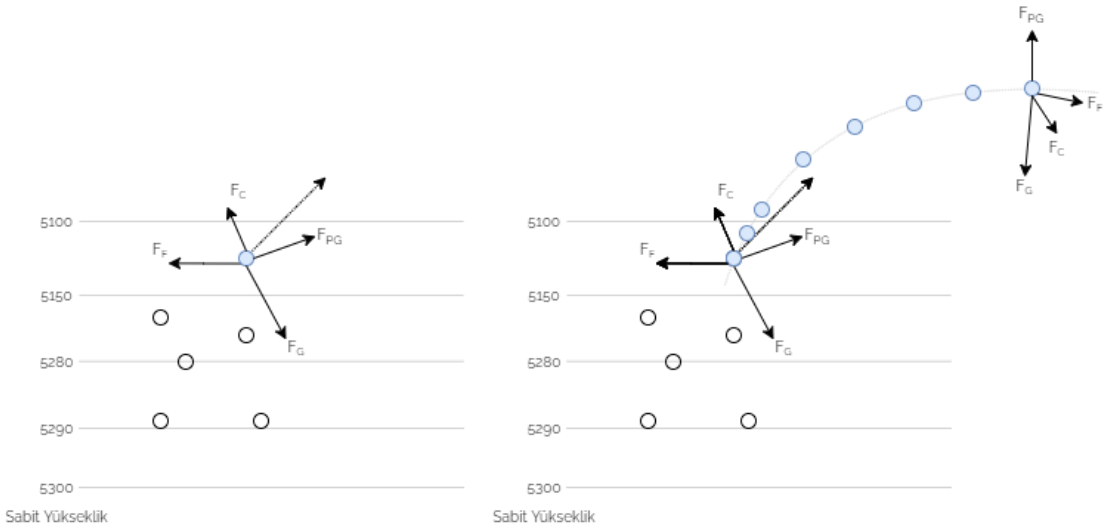
$$\overrightarrow{F_F} = -p\alpha\vec{u} \tag{4.4}$$

$$\overrightarrow{F_{PG}} = p\delta \mathrm{V}\vec{g} \tag{4.5}$$

$$\overrightarrow{F_C} = -2\Omega\vec{u} \tag{4.6}$$

Tüm bu kuvvetlerin Denklem 4.1'da yer alan Newton'un ikinci hareket kanununda bileşke kuvvet toplamı şeklinde ifade edilmesiyle Denklem 4.7'de yer alan eşitlik elde edilir.

$$p\vec{u}\Delta \mathrm{t} = (p\delta \mathrm{V}\vec{g}) + (-\nabla \mathrm{P}\delta \mathrm{V}) + (-p\alpha\vec{u}) + (-2\Omega\vec{u}) \tag{4.7}$$



Şekil 4.1. Hava parselinin atmosferik hareketi.

4.1. BAŞLANGIÇ KOŞULLARI

RGO algoritması diğer popülasyon tabanlı optimizasyon algoritmalarında olduğu gibi belirli başlangıç koşulları ile optimizasyon süreci başlatılır. Problemin tanımlanmasından sonra hız ve konum güncellemede kullanılan sabit parametreler belirlenerek değer ataması yapılır. Popülasyon boyutu, problem boyutu, maksimum iterasyon sayısı, yer çekimi katsayısı g, α sabiti ve ideal gaz denkleminde (Denklem 4.2) yer alan RT katsayıları başlangıçta belirlenir.

4.2. KONUM GÜNCELLEME

Başlangıç koşulları belirlendikten sonra popülasyon boyutu kadar oluşturulan hava parselleri çözüm uzayında rasgele konumlandırılır. Bir sonraki aşamada bu hava parselleri yeni konum ve hızları Denklem 4.7'de yer alan eşitliğe göre güncellenir. Eşitliğin algoritmaya uyarlanmış şekli Denklem 4.8'de gösterilmiştir.

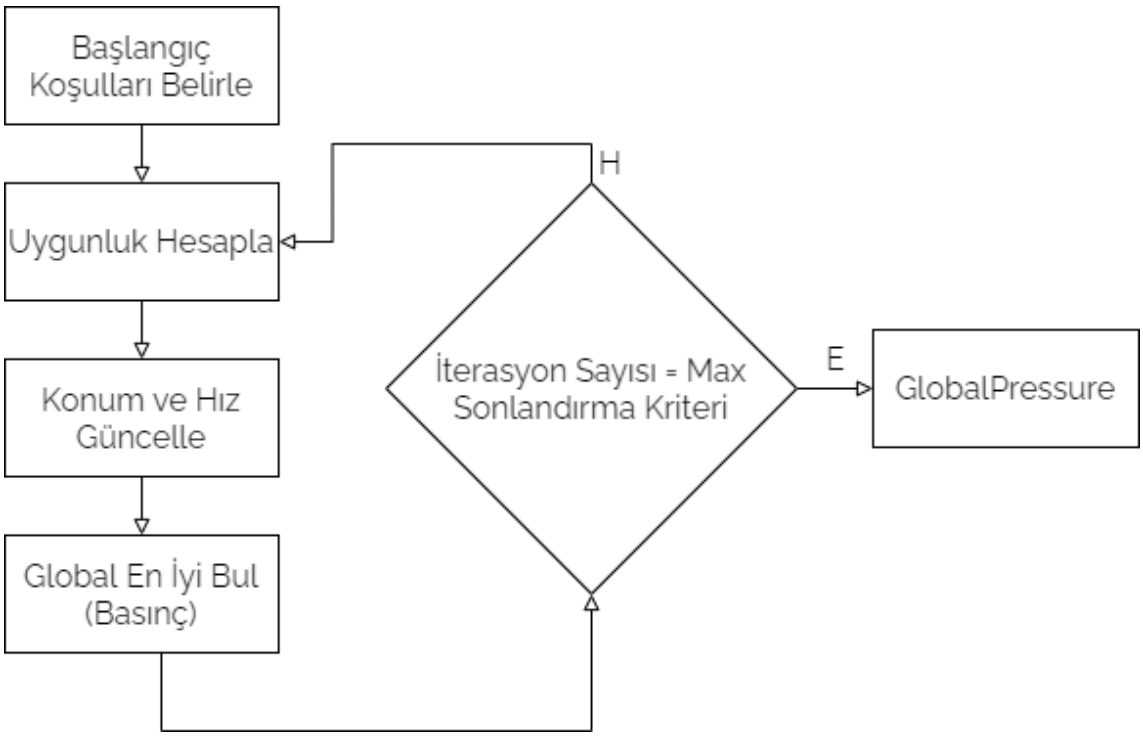
$$u_n = (1-\alpha)u_c - gx_c + \left|\frac{i-1}{i}\right| RT(x_{max} - x_c) + \frac{cu_c^{otherd}}{i} \tag{4.8}$$

Yukarıdaki eşitlikte u_n, bir sonraki iterasyona aktarılacak hız bilgisi, u_c ise a parselin anlık hızını ifade etmektedir. Hava parselinin anlık konum bilgisi x_c, o ana kadar bulunduğu konumlardan en iyisi x_{max} olarak gösterilmektedir. i değişkeni hava parselinin popülasyondaki sırasını ifade etmektedir. Hava parselleri kendilerine uygulanan basınç değerlerine göre sıralanmaktadır. Minimizasyon işleminde alçak basınç iyi, yüksek basınç kötü sonuç olarak kabul edilmektedir. u_c^{otherd} anlık hız değerini etkileyen bir başka boyuttaki hız değeridir. Klasik RGO kullanıcı tanımlamalı birkaç katsayı ile ilişkilidir. Yine Bayraktar ve arkadaşları tarafından 2015 yılında geliştirilen Uyarlanır Rüzgar Güdümlü Optimizasyon (AWDO) algoritması kullanılan bu katsayıları kullanıcılardan bağımsız dinamik bir şekilde güncelleyebilmektedir [25]. Denklem 3.10'da sürtünme katsayısı α, yerçekimi sabiti g, coriolis kuvveti c, evrensel gaz sabiti R ve sıcaklık T ile gösterilmiştir. Algoritma içerisinde evrensel gaz sabiti ve sıcaklık tek bir katsayıda RT birleştirilerek kullanılmaktadır. Denklem 4.8'de yer alan katsayılar ve değişkenler ile elde edilen yeni hız değeri hava parselinin bir sonraki pozisyonunu belirlemekte kullanılır. Hava parselinin bir sonraki pozisyonu Denklem 4.9'da gösterilmiştir.

$$x_n = x_c + u_n \Delta t \tag{4.9}$$

Yukarıdaki denkleme göre hava parselinin yeni konumu x_n, mevcut konumu x_c, güncellenmiş hız vektörü u_n ve zamana bağlı değişim katsayısı Δt ile ifade edilmektedir. Her iterasyonda hız ve konum bilgileri güncellenerek hava parselleri mevcut olan en iyi konuma, alçak basınca doğru hareket eder.

RGO uygulanış biçimi olarak PSO'ya benzerlik göstermektedir. PSO'da olduğu gibi her iterasyonda parsellerin konum bilgisi güncellenir ve en iyi konuma doğru bir yakınsama söz konusudur. RGO parçacık tabanlı diğer algoritmalarla karşılaştırıldığında hızlı yakınsaması ile ön plana çıkmaktadır. RGO'ya ait akış diyagramı Şekil 4.1'de gösterilmiştir.



Şekil 4.2. RGO akış diyagramı.

Yapılan bu tez çalışması hızlı yakınsaması ile bilinen RGO algoritmasının çok-amaçlı problemlerin çözümünde etkin bir şekilde kullanımını hedeflemektedir. Bastırılamayan hızlı sıralama ve adaptif ızgara yaklaşımlarının birlikte kullanılmasıyla elde edilen yöntem diğer algoritmaların çok-amaçlı problemler için uyarlanabilmesini kolaylaştırmaktadır.

5. ÇOK-AMAÇLI OPTİMİZASYON

Çok-amaçlı optimizasyon, birden fazla amacın eş zamanlı olarak optimize edilmesi olarak adlandırılır. Optimizasyon problemlerinde ele alınan problemin tek amaç olacak şekilde modellenmesi çoğu zaman mümkün olmayabilir. Birden fazla kriterin bulunduğu ve bu kriterlerin birbirleri ile çeliştiği durumlarda alternatif çözümler söz konusudur.

Çok-amaçlı problemlerin optimize edilmesinde tek-amaçlı optimizasyon algoritmalarının kullanılması arama uzayının yeterince taranmaması ve efektif sonuçların elde edilmemesine neden olabilir. Bu nedenle çok-amaçlı problemlerin çözümlerinde efektif sonuçlara ulaşmak için arama uzayının tamamıyla kapsayacak yöntemler geliştirilmiş ve bu yöntemler tek-amaçlı optimizasyon algoritmalarına uyarlanmıştır.

Genel olarak çok–amaçlı bir optimizasyon problemi n adet amaç fonksiyonu, m adet karar değişkeni ve k adet kısıt fonksiyonundan oluşur. Çok-amaçlı bir optimizasyon problemi aşağıdaki gibi matematiksel olarak ifade edilebilir.

$$minimize\ y = \{f_1(X), f_2(X) \dots, f_n(X)\} \tag{5.1}$$

$$const.\ G = \{g_1(X), g_2(X) \dots, g_m(X)\} \tag{5.2}$$

$$X = \{1,2 \dots, k\}, x\ \in \mathbb{R}\ \text{x} \tag{5.3}$$

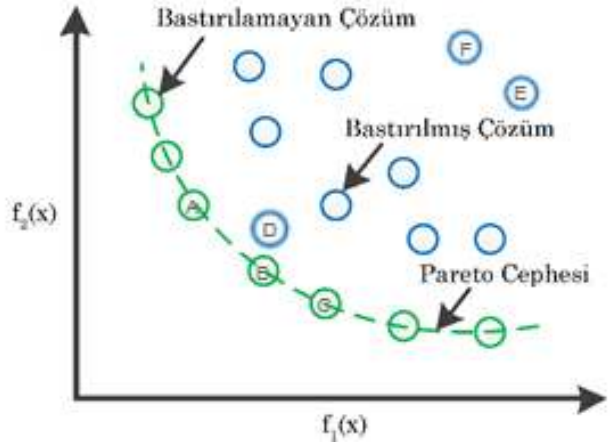
Yukarıdaki denklemlere göre y amaç, G kısıt X ise karar vektörlerünü ifade etmektedir.

Çok-amaçlı optimizasyon problemlerinde genellikle tek bir çözümden bahsedilmez. Birden fazla çözümün oluşturduğu küme optimal küme, bir başka ifade ile pareto-optimal çözüm kümesi olarak adlandırılır.

5.1. PARETO-OPTİMAL

Pareto-optimal, 1900'lü yılların başında V. Pareto tarafından öne sürülmüştür. Arama uzayında herhangi bir çözüm, en iyi, en kötü veya normal bir çözüm olabilir. Amaçlardan biri için elde edilen en iyi çözüm bir başka amaç için en kötü çözüm olabilir. Pareto-optimal çözüm ise arama uzayında herhangi bir çözüm tarafından baskınlanamayan çözümdür [36]. İki-amaçlı bir optimizasyon problemine ait baskınlık ilişkisi Şekil 5.1'de

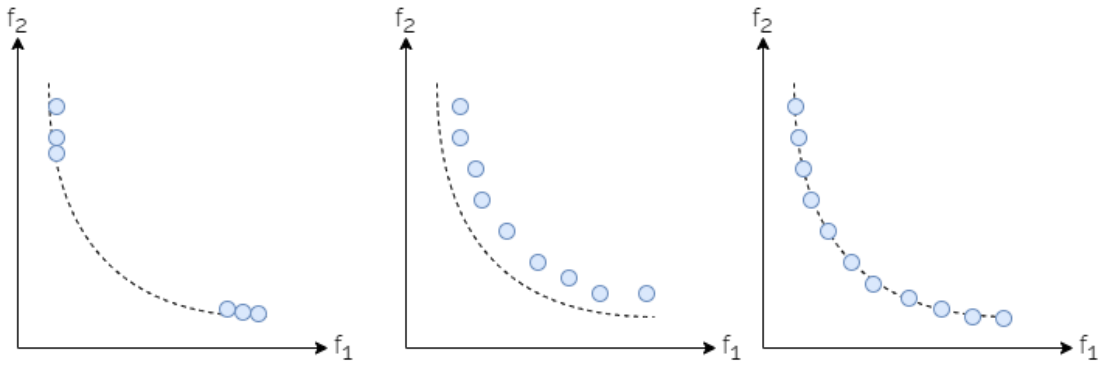
gösterilmiştir.



Şekil 5.1. Baskınlık ilişkisi.

Şekil 5.1'e göre ok ile ifade edilen eğri o problemin ulaşılabilecek çözüm sınırını ifade eder ve pareto cephesi olarak adlandırılır. Pareto cephesi üzerinde yer alan A, B ve C noktaları diğer çözümler tarafından baskınlanamayan yani pareto-optimal çözümlerdir. D noktası E ve F noktalarına göre baskın iken A, B ve C tarafından baskınlandığı için baskılanan çözüm olarak ifade edilir. Bir maksimizasyon probleminde ise E ve F noktaları pareto optimal çözüm olur. A, B ve C noktaları ise ütopik nokta olarak adlandırılır.

Çok-amaçlı optimizasyon problemlerinde amaç pareto-cephesini bulmak veya mümkün olan en uzaklıkta yakınsamaktır. Çeşitlemek, çözümlerin bir nokta üzerinde yoğunlaşması değil, pareto cephesi üzerinde düzgün bir dağılım göstermesidir. Yakınsama ise optimal çözümlerin pareto cephesine ne kadar uzak olduğu anlamına gelir (Şekil 5.2).



a) b) c)

Şekil 5.2. Çok-amaçlı optimizasyonda baskınlık durumları: a) Yakınsama b) Çeşitlilik c) Yakınsama ve çeşitlilik.

5.2. ÇOK-AMAÇLI OPTİMİZASYON ALGORİTMALARI

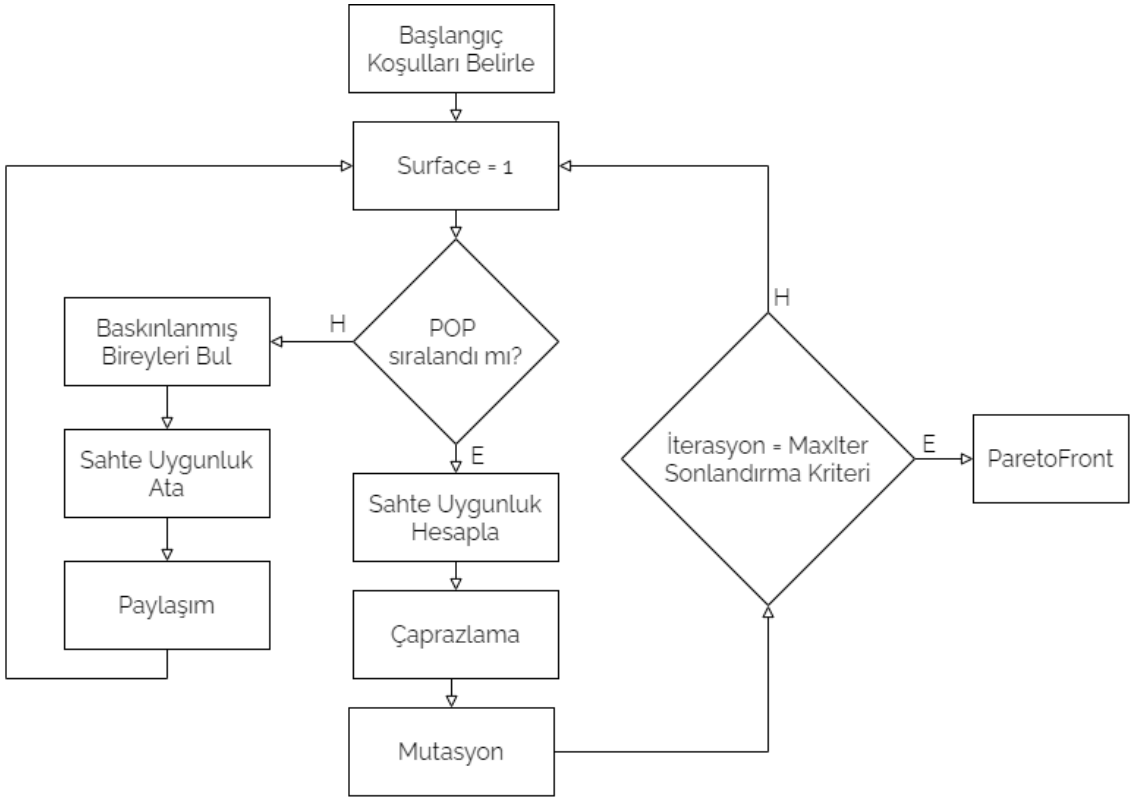
Tek-amaçlıda olduğu gibi çok-amaçlı optimizasyon probleminin çözümünde metasezgisel algoritmalar daha az maliyetli olmalarından dolayı sıklıkla tercih edilmektedir. Bu bölümde literatürde iyi bilinen ve tez organizasyonunda yer alan bazı algoritmalar ele alınmıştır.

5.2.1. Baskınlanamayan Sıralamalı Genetik Algoritma-II (NSGA-II)

Baskınlanamayan Sıralamalı Genetik algoritma (NSGA) Deb ve arkadaşları tarafından 1994 yılında öne sürülmüş, klasik genetik algoritmada yer alan çaprazlama ve mutosyan operatörlerini kullanan aynı zamanda kendine özgü seçim süreci olan popülasyon ve evrimsellik tabanlı çok-amaçlı optimizasyon algoritmasıdır. NSGA algoritmasında mutasyon ve çaprazlama genetik algoritmada olduğu gibi kullanılır. Lider seçiminde tek bir lider olmayacağından dolayı popülasyonda yer alan bireyler pareto seviyesine göre sıralanır. İlk aşamada baskınlanamayan çözümler elde edilir. Bu ilk pareto yüzeyi olarak adlandırılır. Sonraki aşamada bireylere sahte uygunluk değeri atanır. Böylelikle sınıflandırılmış bireylerin eşit olduğu varsayılır. Popülasyonda çeşitlilik elde etmek için bireyle sahte uygunluk değeri ile paylaşılırlar. Bireylerin gerçek uygunluk değerleri birey sayısına bölünerek yeni bir uygunluk değeri hesaplanır. Bu değer kullanılarak seçim işlemi yapılır.

Baskınlanmış bireyler göz ardı edilerek diğer bireyler üzerinden ikinci bir pareto yüzey elde edilir. Bu işlem tüm bireyler bir yüzeye ait olana kadar devam eder ve çözüm uzayı sıralanmış olur [6].

Popülasyon bir sonraki aşamada sahte uygunluk değerine göre yeniden oluşturulur. İlk yüzeydeki bireylerin uygunluk değerleri yüksek olduğundan diğer yüzeylere göre oranla yeni jenerasyona aktarılma daha fazladır. Yapılan bu sınıflandırma işlemi gerçek pareto cephesine popülasyonun daha hızlı yakınsamasını sağlar. Şekil 5.3'de NSGA akış diyagramı gösterilmektedir.

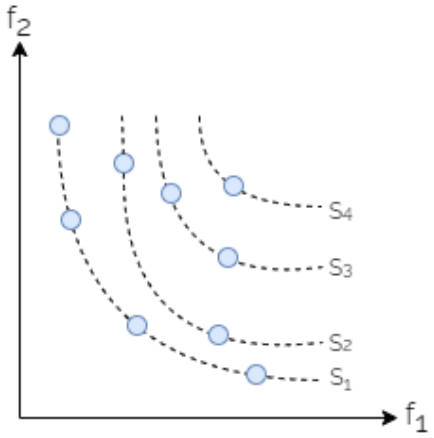


Şekil 5.3. NSGA akış diyagramı.

Deb ve arkadaşları tarafından geliştirilen NSGA, hesaplama maliyeti başta olmak üzere bazı zayıf yönlerinin keşfedilmesiyle 2002 yılında yine Deb ve arkadaşları tarafından yeniden düzenlenerek NSGA-II algoritması ortaya konmuştur [12]. Algoritma NSGA'e benzerlik göstermesinin yanı sıra birçok yenilik ve yöntem içermektedir. Bunlardan en bilinen ve referans olarak kullanılan yöntem hızlı baskınlanamayan sıralama yöntemidir.

5.2.1.1. *Hızlı Baskınlanamayan Sıralama (Fast Non-Dominated Sorting)*

Hızlı baskınlanamayan sıralama yaklaşımında tüm bireyler birbirleri ile karşılaştırılarak baskınlanmamış bireyler bulunur. Bu bireyler bir sonraki aşamada göz ardı edilerek baskınlanmamış birey bulma işlemi tekrarlanır. Tüm yüzeyler bulunup baskınlanmış birey kalmayana kadar süreç devam eder ve sıralama işlemi tamamlanmış olur (Şekil 5.4). Algoritma için en kötü senaryo her yüzeyde bir çözümün olduğu durumda gerçekleşir. N elemanlı bir popülasyon için N tane yüzey olacağı anlamına gelir.

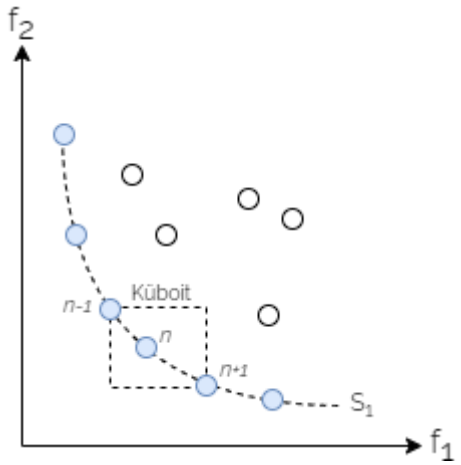


Şekil 5.4. Sıralama örneği.

5.2.1.2. *Kalabalık Mesafesi (Crowding Distance)*

Metasezgisel algoritmalarda çözümlerin pareto-optimal çözüm kümesine ne kadar yakınsadığı ve çeşitlilik ile ne kadar etkin bir yayılıma sahip olduğu oldukça önemlidir. NSGA çözümlerin daha çeşitli olabilmesi için paylaşım adı verilen bir yöntem kullanır. Bu paylaşım yönteminin başta karmaşık hesaplama maliyeti olmak üzere zayıf yönleri bulunmaktadır. NSGA-II algoritmasında bu zayıflığın önüne geçebilmek için paylaşım yöntemi yerine kalabalık mesafesi yöntemi geliştirilmiştir. Bu yaklaşım ile hesaplama maliyeti azaltılmış ve bireysel parametre kullanımı ortadan kaldırılmıştır.

Kalabalık mesafesi aynı zamanda yoğunluk mesafesi anlamına gelir. Çözüm kümesinde yer alan bir noktaya ait yoğunluğu hesaplayabilmek için o noktanın altında ve üstünde bulunan iki noktaya olan ortalama uzaklık kaydedilir. Bu çözüm kümesinde yer alan tüm noktalar için gerçekleştirilir. Kaydedilen bu değer kullanılarak küboit oluşturulur ve bu küboitin çevre uzaklığı hesaplanır. Bu değere kalabalık mesafesi adı verilir (Şekil 5.5).



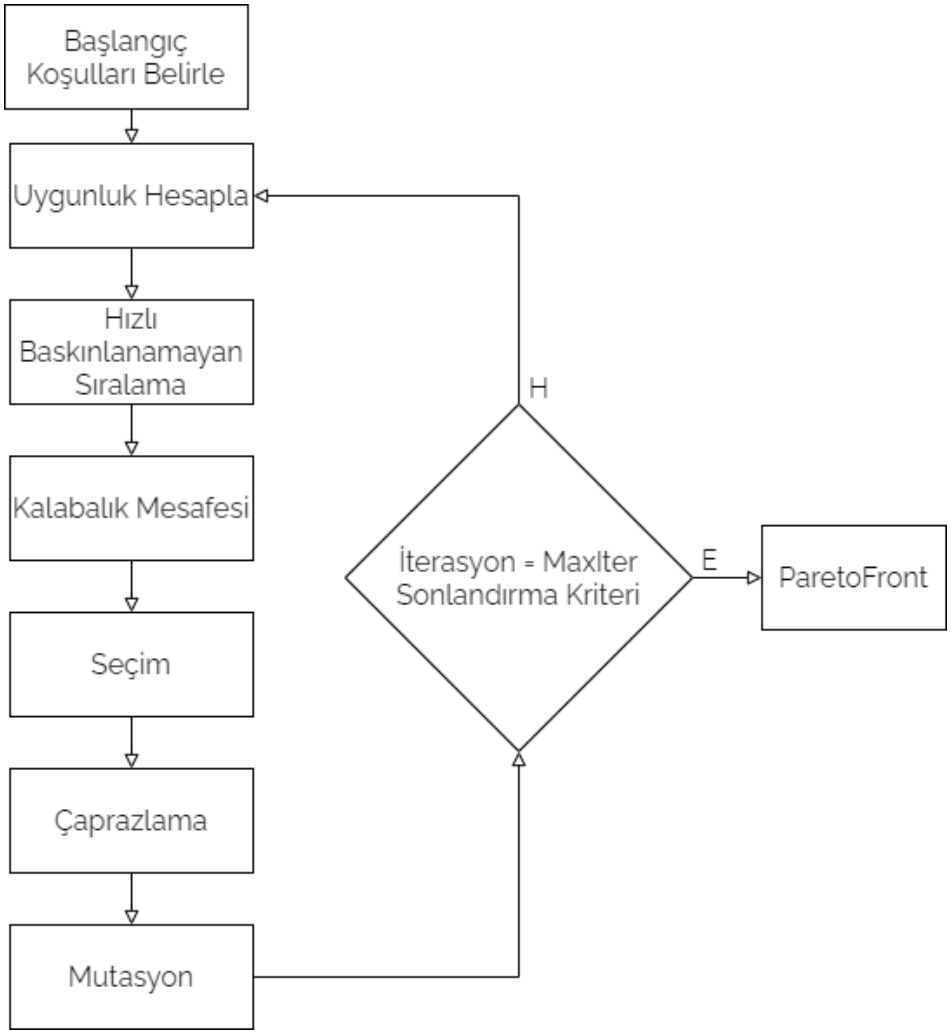
Şekil 5.5. Kalabalık mesafesi.

Kalabalık mesafe hesabı aynı yüzeydeki bireyleri rütbelendirmek için kullanılır. Her birey için kalabalık mesafe hesabı yapıldıktan sonra popülasyon büyükten küçüğe doğru sıralanır. Liste başı çeken bireyler içerisinden yeni jenerasyona aktarılacak bireyler seçilir. Kalabalık mesafesine ait sözde kod Şekil 5.6'da gösterilmiştir.

1. $l = |F_n|$
2. **While** $i \leq 1$
3. $\quad F_n[i]_{dist} = 0$
4. **While** $m \leq M$
5. $\quad F_n = sort(F_n, m)$
6. $\quad F_n[1]_{dist} = F_n[l]_{dist} = \infty$
7. $\quad i = 2$
8. $\quad$ **While** $i \leq l - 1$
9. $\quad\quad F_n[1]_{dist} + F_n[1]_{dist} + \frac{F_n[i+1]_m - F_n[i-1]_m}{f_m^{max} - f_m^{min}}$
10. $\quad i++$

Şekil 5.6. Kalabalık mesafesine ait sözde kod [37].

Bir sonraki aşamada klasik genetik algoritma operatörleri kullanılarak çocuk bireyler oluşturulur. İterasyon ve sonlandırma kriterine bağlı olarak yeni jenerasyon bireylerin uygunluk değerleri, rütbeleme ve kalabalık mesafe hesabı tekrarlı bir şekilde yapılarak optimum çözüm kümesi elde edilir. NSGA-II algoritma akış şeması Şekil 5.7'da gösterilmiştir.



Şekil 5.7. NSGA-II akış diyagramı.

5.2.2. Çok-Amaçlı Parçacık Sürü Optimizasyon Algoritması (MOPSO)

Parçacık Sürü Optimizasyonu R. Eberhart ve J. Kennedy tarafından 1995 yılında geliştirilmiş, kuş ve balık sürülerinin yiyecek arama davranışlarından esinlenen popülasyon tabanlı bir algoritmadır [3]. Algoritma tek-amaçlı optimizasyon problemlerine etkili çözümler sunmaktadır. Algoritmanın çok-amaçlı problemlerle başa çıkabilmesi için C. Coello ve arkadaşları tarafından 2002 yılında Çok-Amaçlı Parçacık Sürü Optimizasyon Algoritması (MOPSO) önerilmiştir [38]. Önerilen bu algoritmada klasik PSO operatörlerinin yanı sıra NSGA-II ve Pareto Arşivlemeli Evrim Stratejisi (PAES) algoritmalarında önerilen bazı metotlar kullanılmıştır [39].

MOPSO algoritmasında ilk aşama başlangıç koşullarının belirlenmesidir. Başlangıç koşulları oluşturulduktan sonra her bir amaç için tüm parçacıkların uygunluk değeri hesaplanır. Uygunluk değeri hesaplanan parçacıklar için baskın olan parçacıklar depoya

(repository) kaydedilir. Depo algoritmanın son aşamasına kadar baskın parçacıkları saklayan bir kaydedicidir.

Klasik PSO'da sürünün sosyal ve bilişsel davranışlarından esinlenerek sürüyü oluşturan her bir bireyin (parçacığın) en iyi konumları kişisel en iyi ve bu kişisel en iyilerin en iyisi küresel en iyi olarak kaydedilir. Kişisel ve küresel en iyi bilgisi bireylerin yeni pozisyonunu belirlemekte kullanılır.

MOPSO algoritmasında her bir parçacığın kişisel en iyisi, parçacığın anlık uygunluk değerinin bir önceki uygunluk değerini baskınlayıp baskınlamadığına göre belirlenir. Eğer yeni uygunluk değeri önceki kişisel en iyiyi baskınlıyorsa o değer parçacığın kişisel en iyisi olarak kaydedilir.

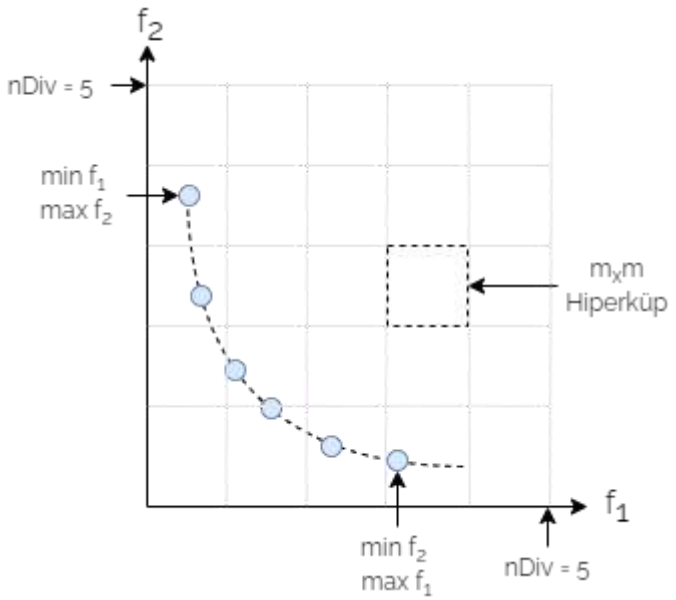
Küresel en iyi her bir iterasyon sonunda depoya kaydedilen baskınlanamayan parçacıklardır. Klasik PSO'da tek bir küresel en iyi olurken MOPSO'da birden fazla küresel en iyi mevcuttur. Algoritmanın sonunda depoda bulunan küresel en iyiler pareto-optimal çözüm kümesini oluşturur.

PSO'da parçacıklar kendi en iyisi ve küresel en iyinin konum bilgisini kullanarak küresel en iyiye doğru hareket eder ve arama uzayında yeni çözümler aranır. MOPSO'da tek bir küresel en iyi olmadığından liderler seçimi yapılır. Lider seçimi PAES algoritmasında önerilen adaptif ızgara yaklaşımına göre yapılır.

5.2.2.1. *Lider Seçimi*

MOPSO'da lider seçimi PAES tarafından önerilen adaptif ızgara yaklaşımı ile yapılmaktadır. Bu yaklaşım çok-amaçlı diğer algoritmalarda kullanılan yoğunluk metodlarına göre iki önemli avantaja sahiptir. Bunlar hesaplama maliyetinin düşük ve adaptif olmasından dolayı herhangi bir parametreye ihtiyaç duymamasıdır. Adaptif ızgara yaklaşımı d-boyutlu amaç uzayını yinelemeli olarak ızgaralara böler.

Her parçacığa ait uygunluk değerleri elde edildikten sonra amaç uzayı boyutuna göre ızgaralara bölünür. Bu ızgara hiperküp olarak adlandırılır. Depoda bulunan baskın çözümlerin ızgarada karşılık gelen konumları parçacığın ızgara konumu olarak kaydedilir. Aynı ızgaraya denk gelen parçacıklar arasından rulet tekerleği yöntemi ile lider seçimi yapılır. Şekil 5.8'de iki-amaçlı bir optimizasyon problemi için adaptif ızgara yaklaşımı gösterilmektedir.

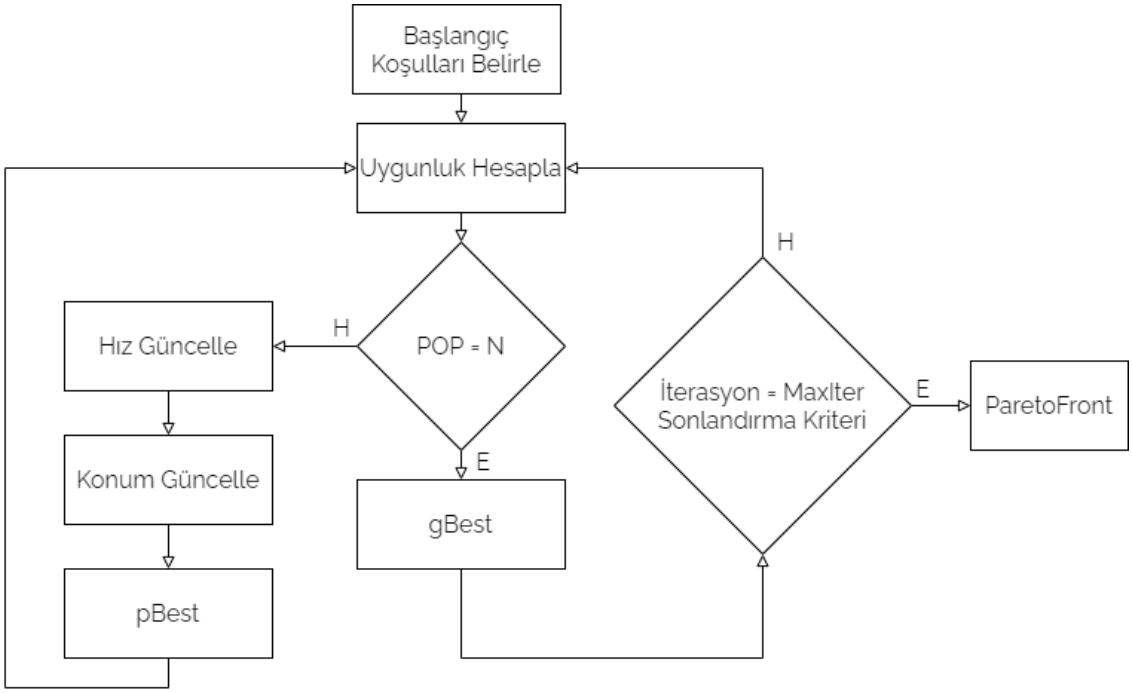


Şekil 5.8. PAES adaptif ızgara.

5.2.2.2. *Konum Güncelleme*

Lider seçiminden sonra her bir parçacığın konum ve hız bilgisi hesaplanır. Her bir parçacık lider deposundan rasgele bir lideri küresel en iyi olarak belirler ve konum ile hızını kişisel en iyi konum bilgisi ile birlikte günceller. Konum ve hız güncelleme operatörleri PSO ile aynıdır.

Algoritmanın sonraki aşamaları iteratif olarak sonlanma kriteri ve maksimum iterasyıon sayısında kadar devam eder. Her iterasyonda uygunluk değerleri hesaplanır ve baskınlanamayan parçacıklar depoya atılır. Depoda yer alan parçacıkların ızgara konumları belirlenir ve liderler seçimi yapılır. Son olarak bu liderlere ait konum bilgisi kullanılarak parçacıkların konum ve hızları güncellenir. Sonlanma kriterinin ardından depo problemin optimal çözüm kümesi olmuş olur. Şekil 5.9'de MOPSO akış şeması gösterilmektedir.



Şekil 5.9. MOPSO akış diyagramı.

5.2.3. Çok-Amaçlı Gri Kurt Optimizasyon Algoritması (MOGWO)

Gri Kurt Algoritması (GWO), 2014 yılında S. Mirjalili ve arkadaşları tarafından önerilen, gri kurtların avlanma ve sosyal liderlik hiyerarşisinden esinlene popülasyon tabanlı tek-amaçlı optimizasyon algoritmasıdır [13]. GWO algoritması geliştirilirken kurtları sosyal hişerarşisi matematiksel olarak modellenmiştir. Bu matematiksel modele göre en uygun çözüm alfa (α) kurt olarak kabul edilir. Devamında yer alan ikinci ve üçüncü çözümler sırasıyla beta (β) ve delta (δ) kurtlar olarak adlandırılır. Bunların dışında geriye kalan tüm çözümler omega (ω) kurtlar olduğu varsayılmaktadır. GWO algoritmasında avlanma yani optimizasyon α, β ve δ tarafından yönlendirilir. Ω kurtlar ise küresel optimum arayışında bu üç kurdu takip eder [14].

Çok-Amaçlı Gri Kurt Algoritması (MOGWO) mevcut GWO algoritmasına iki yeni metot eklenmesiyle çok-amaçlı problemlere çözüm aramaktadır. Algoritma kullanılan metotlar ve akış şeması incelendiğinde MOPSO algoritmasına oldukça benzerlik göstermektedir. Bu metotlardan ilki baskınlanamayan pareto-optimal çözümleri depolamaktan sorumlu arşivlemedir. Diğer yöntem ise baskınlanamayan çözümlerin yer aldığı arşivden avlanma süreci liderleri olarak alfa, beta ve delta çözümlerin seçilmesine yardımcı olan lider seçim stratejisidir.

5.2.3.1. *Arşivleme*

Arşiv, şimdiye kadar elde edilen baskınlanamayan pareto-optimal çözümlerin yer aldığı basit denetlenebilir bir depolama birimidir. Arşivlemenin en önemli özelliği bir çözümün arşive girmek istediğinde veya arşivde yeteri kadar çözüm olduğunda hangi çözümlerin arşive alınacağını ve aynı şekilde arşivden çıkarılacağına karar veren bir denetleyici mekanizmasına sahip olmasıdır. Arşiv, algoritmanın başlangıç aşmasında belirlenen kapasite değeri ile sınırlıdır ve iterasyon aşamasında elde edilen her baskınlanamayan çözüm arşivde yer alan çözümler ile karşılaştırılarak arşiv güncellenir. Arşivleme üç denetim aşamadan oluşmaktadır:

- Arşive eklenmesi planlanan çözüm, arşivde en az bir üye tarafından baskınlanıyorsa o çözüm arşive alınmaz.
- Çözüm arşivde bir veya daha fazla üyeyi baskınlıyorsa baskınlanan çözümler arşivden atılır ve yeni çözüm arşive eklenir.
- Eğer çözüm ne baskınlanıyor ne de baskınlıyorsa direk arşive alınır.

Arşivin dolu olması durumunda adaptif ızgara yardımı ile kalabalık hesabı yapılır ve eklenecek yeni çözüm en az çözümün bulunduğu hiperküp adı verilen ızgaraya yerleştirilir. Aynı şekilde çıkarılacak çözüm en kalabalık ızgaradan yapılır.

Arşivden bir üyenin çıkarılması belirli bir olasılığa bağlıdır. Bu olasılık hiperküpler içerisinde yer alan çözüm sayısı ile doğru orantılıdır. Bir başka ifade ile en kalabalık hiperküpler seçilir ve bunlar içerisinden en kalabalık olandan rasgele bir çözümün silinme olasılığı diğer hiperküplere göre daha yüksektir. Bu işlem maksimum iterasyon sayısı veya sonlandırma kriterine kadar devam eder.

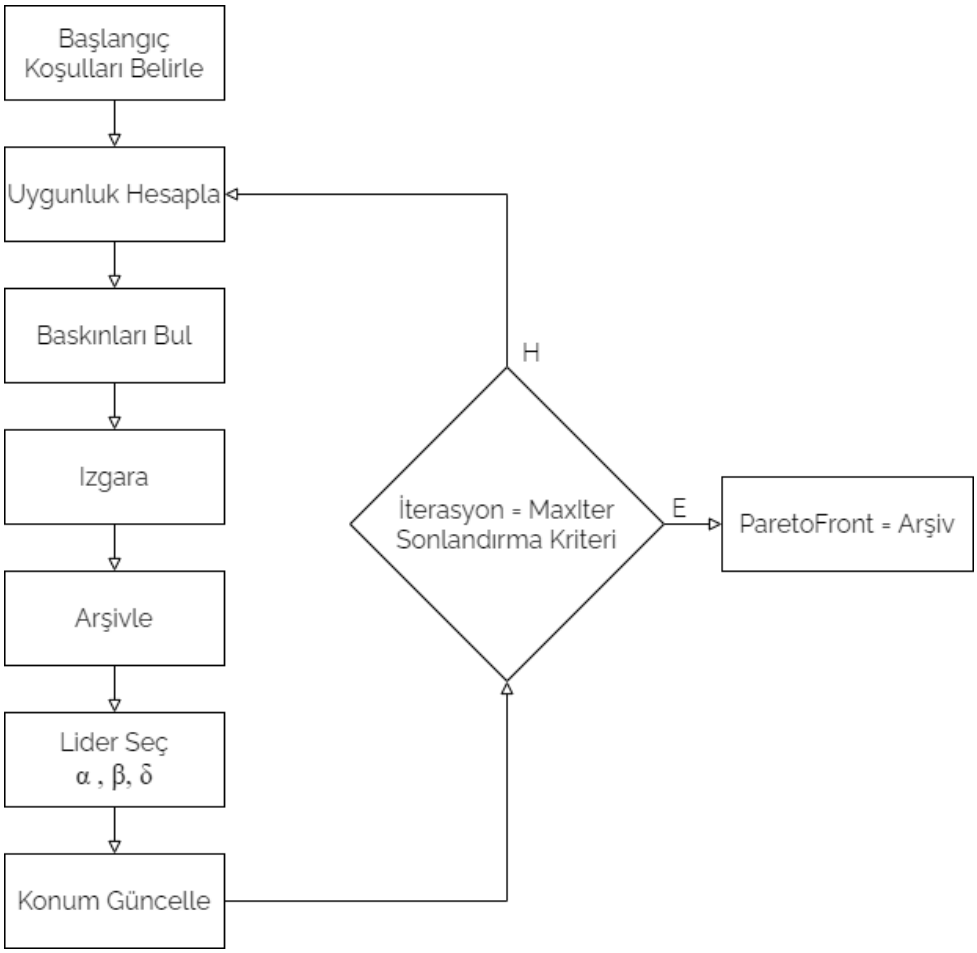
5.2.3.2. *Lider Seçimi*

MOGWO'yu klasik GWO'dan ayıran bir diğer unsur lider seçim mekanizmasıdır. GWO algoirtmasında elde edilen en iyi çözümler sırayla alfa, beta, delta kurtları olarak temsil edilmektedir. Bu liderler diğer kurtları optimum çözüme yakın bölgelere arama yapmaya yönlendirir.

Çok-amaçlı problemlerde en iyi çözüm yerine pareto baskınlığı kavramı mevcuttur. Pareto baskınlığında tüm çözümler en iyi çözüm olarak ifade edilir. Bunların arasından rasgele seçim yapmak arama uzayını belli bir bölgeye yoğunlaştırabilir. Bu durum çeşitlilik açısında kabul edilen bir durum değildir.

MOGWO'da lider seçimi ızgara yaklaşımı ile yapılır. Arşivde yer alan çözümler ızgara yöntemi ile gruplandırılır. Gruplar arasında en az kalabalığa sahip hiperküpler içerisinden alfa, beta ve delta kurt olmak üzere seçim yapılır. Bu seçim rasgelelik ve olasılığa dayalı rulet çarkı yöntemi ile gerçekleştirilir.

Algoritmanın diğer aşaması GWO algoritmasıyla aynı olup sonlandırma kriteri veya maksimum iterasyon sayısına kadar bu işlemler devam eder. Elde edilen arşiv pareto-optimum çözüm kümesidir. Algoritmanın akış diyagramı Şekil 5.10'da gösterilmiştir.



Şekil 5.10. MOGWO akış diyagramı.

6. HİBRİT ÇOK AMAÇLI RÜZGAR GÜDÜMLÜ OPTİMİZASYON ALGORİTMASI

Tek-amaçlı problemlerin çözümünde başarılı sonuçlar veren RGO, bu bölümde çok-amaçlı problemleri optimize edebilmesi için uyarlanmıştır. Çok-amaçlı sezgisel algoritmaların birçoğu pareto baskınlığı tabanlı yöntemler kullanmaktadır. Bayraktar ve arkadaşları tarafından geliştirilen Çok-amaçlı RGO algoritmasında ise pareto cephesinin elde edilmesinde hızlı bastırılamayan sıralama kullanılmaktadır. Bu çalışmada ise [23] nolu çalışmaya ek olarak çeşitlilik ve yakınsamayı daha fazla arttırabilmek için pareto arşivlemeli depo ile adaptif ızgara yaklaşımı bir arada kullanılmaktadır. Aynı zamanda çözüm uzayının pareto cephesine yakınsarken belli noktalara takılıp kalmaması için yakın komşulukları ortadan kaldıran ve yakınsama yapılan çözümü derecelendiren bir algoritma geliştirilmiştir. Bu algoritmaya göre daha önceden belirlenmiş bir mesafe içerisinde yer alan çözümler çözüm uzayından çıkarılır. Böylelikle yakınsama gerçekleşirken belli bölgede yoğunlaşma ortadan kaldırılmış olacaktır. Yakınsanan çözümler her yakınsamada derecelendirilip en düşük dereceli çözümler arasından rasgele seçim yapılarak çeşitlilik sağlanmış olur.

6.1. PAREMETRELERİN TANIMLANMASI

HÇA-RGO algoritması, popülasyon boyutu, karar değişkenleri sayısı, maksimum iterasyon sayısı, amaç sayısı, depo kapasitesi ve alt-üst limiteler gibi parametreler ile optimizasyon işlemini gerçekleştirmektedir. Bunların dışında tek-amaçlı RGO algoritmasında yer alan temel parametreler de kullanılmaktadır. Popülasyon ve karar değişkenlerinin sayısı, çalışma performansı ve sonuçları etkileyen en önemli parametrelerdir. Bunların dışında depo kapasitesi ve maksimum bastırılamayan pareto-optimal nokta sayısı belirlenmektedir.

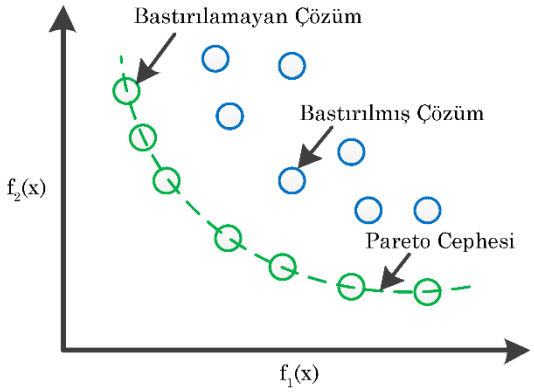
6.2. BAŞLANGIÇ KOŞULLARININ OLUŞTURULMASI

HÇA-RGO algoritması başlangıç aşamasında diğer popülasyon tabanlı algoritmalara benzer şekilde alt ve üst limitler içerisinde rasgele hız ve konum değerleri ile

optimizasyon sürecini başlatmaktadır. Konum bilgisi optimize edilecek amaç fonksiyonlarını oluşturan karar değişkenleridir. Popülasyon sayısı kadar oluşturulan konumsal bilgiler amaç fonksiyonuna aktarılarak her bir konuma karşılık gelen değerler hesaplanır. Algoritmanın ilerleyen aşamalarında konum bilgisi, kurulum aşamasında belirlenmiş olan parametreler ve her bir konuma karşılık gelen değerlerden elde edilen hız bilgisi ile güncellenerek amaç fonksiyonu optimize edilir.

Çok-amaçlı optimizasyon algoritmalarında pareto verimliliği, pareto optimumu veya optimum pareto gibi kavramlar ön plana çıkmaktadır. Arama uzayında yer alan bir çözümün diğer çözümler tarafından bastırılamıyor olması o çözümün optimum çözüm olduğunu ifade eder. Bu çözüme pareto-optimal çözüm, tüm pareto-optimal çözümlerin oluşturduğu kümeye pareto-optimal küme adı verilir [40].

Çok-amaçlı optimizasyon algoritmalarında amaç, pareto-optimal kümeyi oluşturan çözüm noktalarını elde edebilmektir. Bu küme aynı zamanda bastırılamayan çözüm kümesi olarak adlandırılır [41]. Şekil 6.1'de çözüm kümesinde yer alan bastırılmış ve bastırılamayan çözümler bir arada gösterilmiştir.



Şekil 6.1. Pareto Cephesi, bastırılan ve bastırılamayan çözümler.

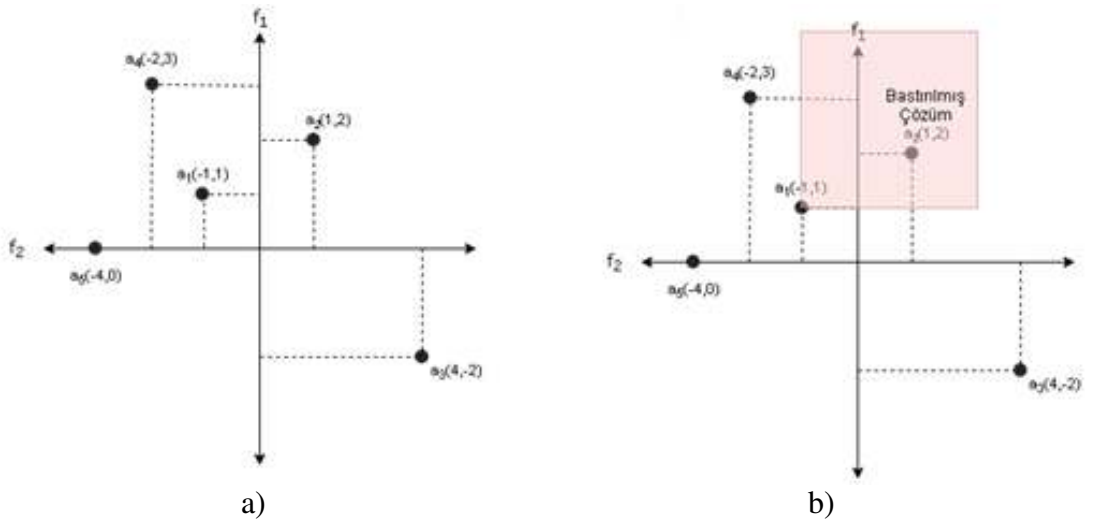
6.3. BASTIRILAMAYAN ÇÖZÜMLERİN BELİRLENMESİ

Tek-amaçlı optimizasyon problemlerinde genel olarak elde edilen çözüm kümesinden bir veya birkaç çözüm seçilerek optimum çözüme yakınsama işlemi yapılır. Seçilen çözümler problemin amacına göre minimum veya maksimum değerlikte çözümler arasından yapılır. Çok-amaçlı optimizasyon problemlerinde tek bir amaç olmadığından seçim işlemi pareto-optimum çözüm kümesinde yer alan çözümler arasından yapılır.

Çözüm kümesinde bastırılamayan çözümlerin elde edilmesi iki aşamada gerçekleşir.

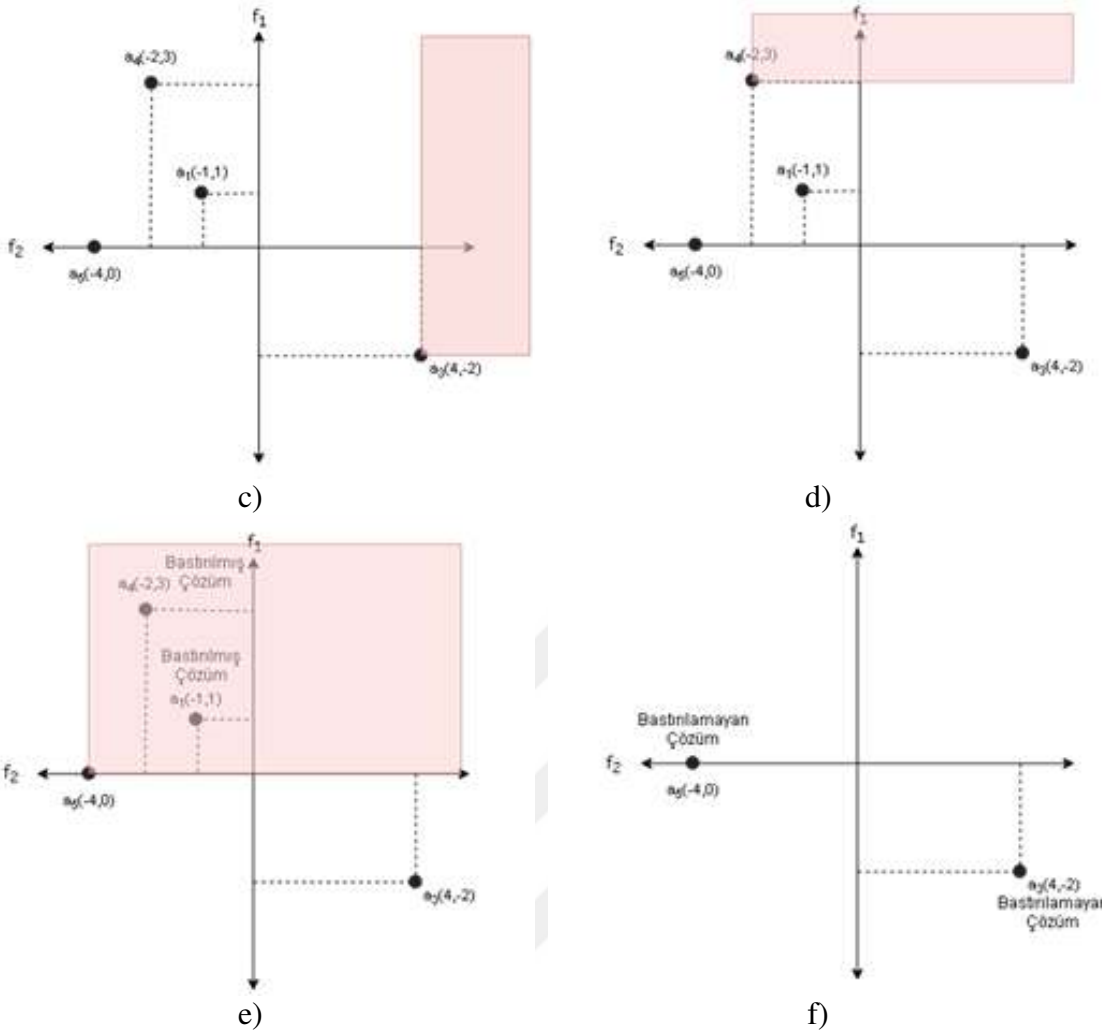
Birinci aşamada popülasyonda yer alan bir çözümün diğer çözümlere baskın olup olmadığı kontrol edilir. İki-amaçlı bir minimizasyon probleminde bir çözüm, çözüm uzayında bir A(f2,f1) noktası şeklinde ifade edilir. f1 ve f2 belirli koşullar altında problemin çözüm noktasını ifade etmektedir. Eğer diğer tüm çözüm noktaları f1, f2 noktasından büyük bir değere sahipse bu çözüm bastırılamayan çözüm olarak ifade edilir.

Bastırılamayan çözümlerin elde edilmesini bir örnek ile ifade etmek gerekirse; a1(-1,1), a2(1,2), a3(4,-2), a4(-2,3) ve a5(-4,0) noktaları önceden belirlenmiş konum ve hız değerlerine ait çözüm kümesini oluşturan noktalar olsun. Bu çözümlerden hangisi veya hangilerinin pareto-optimal çözüm olduğunu belirlemek için sırasıyla her bir noktanın baskın olduğu bölge belirlenir ve o bölgede yer alan noktalar çözüm kümesinden çıkartılır. a1 noktası ele alındığında f2 > -1 ve f1 > 1 bölgesi a1 noktasına ait bastırılmış bölge olarak ifade edilir ve bu bölge içerisinde yer alan noktalar çözüm kümesinden çıkartılır. a1 noktasına ait baskın bölgede a2 noktası yer alındığından bu nokta çözüm kümesinden çıkartılacaktır. Bir sonraki adımda a2 noktası çözüm kümesinden çıkartıldığından pareto-optimal çözüm arayışına a3 noktası ile devam edilir. Geriye kalan tüm noktalara ait baskın bölgeler incelendiğinde başta baskın olan a1 noktası da dahil olmak üzere a4 noktası çözüm kümesinden çıkartılarak pareto-optimal çözümler elde edilmiş olur. Pareto-optimal çözümlerin belirlenme aşaması Şekil 6.2'de gösterilmiştir.



a) b)

Şekil 6.2. Pareto baskınlık örneği: a) Çözüm kümesi b) a1 noktasına ait baskın bölge c) a3 noktasına ait baskın bölge d) a4 noktasına ait baskın bölge e) a5 noktasına ait baskın bölge f) Pareto-optimal çözüm kümesi.



Şekil 6.2. (devam) Pareto baskınlık örneği: a) Çözüm kümesi b) a1 noktasına ait baskın bölge c) a3 noktasına ait baskın bölge d) a4 noktasına ait baskın bölge e) a5 noktasına ait baskın bölge f) Pareto-optimal çözüm kümesi.

Elde edilen pareto-optimal çözüm kümesi diğer çözümlerden bağımsız tutularak algoritmanın başlangıç aşaması tamamlanmış olur. Bir sonraki aşamada konum ve hız bilgileri iteratif bir şekilde güncellenerek yeni pareto-optimal çözümler elde edilir.

6.4. LİDER SEÇİMİ

Baskın olmayan sıralama yaklaşımında çözüm kümesinde yer alan her bir çözüm için baskınlık değerine göre bir derece verilir. En düşük dereceli çözüm baskın çözüm olarak ifade edilir. Yapılan bu çalışmada çözümlerin baskınlıklarını derecelendirmek yerine her iterasyonda en baskın çözümler depoya kaydedilir. Daha sonra depodan lider seçimi yapılarak çözüm kümesinde yer alan çözümlerin konum ve hız bilgileri güncellenir.

Lider seçiminde depoda yer alan her çözüm lider olarak kabul edilir. Adaptif ızgara yaklaşımına göre çözüm uzayı eşit-aralıklı ızgaralara bölünür. Bu ızgaralar arasında yer alan her bir çözüm yine o ızgarada yer alan baskın çözümü lider kabul eder. Eğer ızgara içerisinde birden fazla baskın çözüm mevcut ise liderler arasında rasgele seçim yapılır ve lider sayacı adı verilen bir değişken o çözümün kaç kere lider seçildiğini kaydeder. Bir sonraki aşamada aynı ızgarada yer alan baskın çözümler arasında lider seçimi yapılırken lider sayacının değerine göre rasgele seçim yapılır. Lider sayacı yüksek olan bir baskın çözümün seçilme ihtimali lider sayacı ile ters orantılı olacak şekilde düşük olacaktır.

Başlangıçta geniş bir alana sahip olan ızgara, iterasyon sayısına bağlı olarak küçülerek arama uzayı daraltılır. Bir sonraki aşamadan lider çözümler kullanılarak elde edilen tüm çözümlerin konum ve hız bilgisi güncellenir.

6.5. KONUM VE HIZ GÜNCELLEME

RGO algoritması, hareketin dinamiği kanununu temel alarak rüzgârı oluşturan en küçük hava parselini konumlandırmaktadır. Algoritmada konumsal değişiklik diğer parçacık tabanlı algoritmalarda olduğu gibi en iyi veya en iyiye yakın çözümlere doğru olmaktadır. En iyi çözümlere doğru gerçekleşen hareket yeni en iyi çözümler elde edilmesine olanak sağlamaktadır. Bu aşamada konumsal değişiklik lider seçilmiş çözümlere doğru olmaktadır.

Konum ve hız bilgisi güncellenirken Denklem 3.10'da yer alan formül kullanılmaktadır. Eşitlik kullanılarak elde edilen hız bilgisi hava parselinin o anki konumuna eklenerek yeni konum elde edilir.

Bir sonraki aşamada konum bilgisi güncellenen hava parseline ait yeni uygunluk değerleri hesaplanır ve çözüm kümesinde yer alan yeni baskın çözümler depoya eklenir.

6.6. YAKIN KOMŞULUKLARIN SİLİNMESİ

Pareto-optimal çözümlerin daha çeşitli ve gerçek çözüme yakınsamış olabilmesi için elde edilen pareto-optimal çözüm kümesinde belli bir mesafenin altında bulunan komşuluklarda temizleme işlemi yapılır. Çözüm kümesinde yer alan her bir çözümün diğer çözümler ile olan Öklid uzaklığı hesaplanır. A ve B noktaları pareto-optimal çözüm kümesinde yer alan iki farklı nokta olsun. Bu iki noktanın arasındaki mesafe Denklem

6.1'de yer alan formül ile hesaplanır.

$$d(A,B) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (6.1)$$

Birbirine oldukça yakın olan komşuluklar çıkarılarak çözüm kümesinin daha çeşitli ve yakınsamış olması sağlanır.

6.7. PARETO-OPTİMAL ÇÖZÜM KÜMESİNİN ELDE EDİLMESİ

Tüm bu işlemler önceden belirlenmiş olan iterasyon sayısına kadar devam eder. Her iterasyonda pareto-optimal çözüm kümesi içerisinden liderler seçimi yapılır. Konum ve hız bilgileri bu liderler temel alınarak güncellenir. Yeni çözümler elde edildikten sonra baskın olmayan çözümler çıkartılır, baskınlar ise bir sonraki iterasyonda kullanılmak üzere depo adı verilen kümeye eklenir. İterasyon işlemi sınıra ulaştığında depoda yer alan çözümler çok amaçlı problemin pareto-optimal çözümleri olacaktır. Algoritmaya ait işlem adımları Şekil 6.3'de detaylı olarak gösterilmiştir.

Başlangıç
1. Parametreleri Belirle
2. Parçacıklara Ait Konum ve Hız Bilgisi Oluştur
3. Konum ve Hız Bilgilerine Göre Uygunluk Değerlerini Bul
4. Baskınları Bul Depola

Ana Döngü

While $i < MaxIt$
  While $j < PopSize$
    1.Depodan Lider Seç
    2.Izgara Oluştur
    3.Liderlere Göre Konum ve Hız Güncelle
    4.Konum ve Hız Bilgilerine Göre Uygunluk Değerini Bul
  End
  1.Baskınları Bul Depola
  2.Yakın Komşulukları Sil
End

5. Baskınları Görüntüle

Şekil 6.3. HÇA-RGO sözde kod.

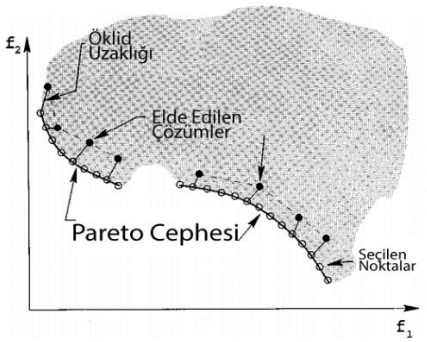
7. DENEYSEL SONUÇLAR

7.1. PERFORMANS METRİKLERİ

Geliştirilen algoritmayı performans açısından değerlendirme için literatürde iyi bilinen algoritmalar ile karşılaştırma yapılmıştır. İlk aşamada kullanılan test fonksiyonuna ait gerçek pareto-optimal çözümlerden eşit-aralıklı 100 çözüm seçilir. Sonrasında bu çözümler ile NSGA-II, MOPSO ve önerilen metot HÇA-RGO algoritması çok-amaçlı optimizasyon algoritmalarında değerlendirmede kullanılan Mesafe Metriği, Maksimum Yayılım, Boşluk ve Hiperhacim adı verilen performans metrikleri ile karşılaştırılmış ve değerler kaydedilmiştir.

7.1.1. Mesafe Metriği (GD)

Mesafe metriği, 1998 yılında Veldhuizen tarafından öne sürülmüş gerçek pareto cephesi ile optimal çözüm kümesi arasında kalan mesafeyi ölçen bir performans kriteridir. Çok-amaçlı optimizasyon algoritmalarında algoritmanın ne kadar yakınsadığını değerlendirmek için kullanılır . Çözüm kümesinde [42] yer alan her bir değer ile gerçek pareto cephesindeki değerler arasındaki öklid mesafesi ölçülerek minimum mesafeler kaydedilir (Şekil 7.1).



Şekil 7.1. Mesafe metriği (GD).

Minimum mesafenin ortalaması sıfıra ne kadar yakın ise algoritma o kadar yakınsamış demektir. Mesafe metriğine ait matematiksel ifade Denklem 7,1'de gösterilmiştir [12], [42].

$$GD = (\sum_{i=1}^{|Q|} d_i^p)^{\frac{1}{p}} / |Q| \quad (7.1)$$

$$d_i = \min_{k=1}^{|P|} \sqrt{\sum_{m=1}^{M} (f_m^i - f_m^k)^2} \quad (7.2)$$

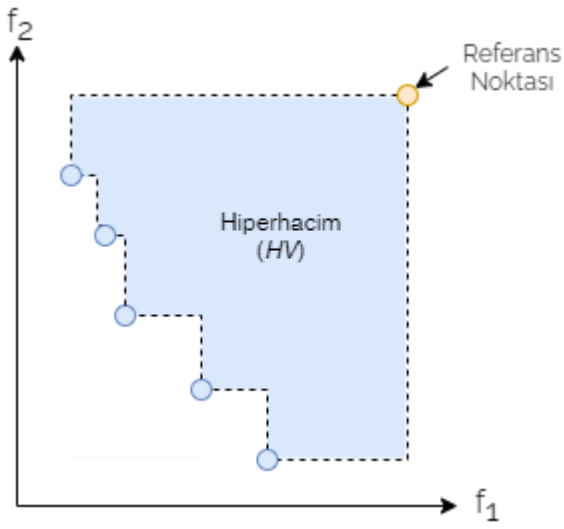
7.1.2. Boşluk (S)

Boşluk metriği pareto-optimal çözümdeki komşu çözümler arasındaki mesafeyi ölçmek için kullanılan bir performans metriğidir. Sonucun sıfıra yakın olması pareto-optimal çözümlerin eşit dağılım yaptığını ifade eder [43]. Boşluk metriği Denklem 7.4'deki gibi matematiksel olarak ifade edilmektedir.

$$S \triangleq \sqrt{\frac{1}{n} - 1 \sum_{i=1}^{m} \max_{k=1} f_m^i - \min_{k=1} f_m^i} \quad (7.3)$$

7.1.3. Hiperhacim (HV)

Hiperhacim metriği, pareto-optimal çözüm kümesinin yakınsama ve çeşitlilik derecesini belirleyen performans metriklerinden biridir. Elde edilen değerin 1'e yakın olması beklenir. Hiperhacim hesaplanırken çözüm kümesinde bulunan en kötü nokta referans noktası olarak belirlenir (Şekil 7.2).



Şekil 7.2. Hiperhacim (HV).

Bu referans nokta ile çözüm kümesindeki tüm noktalar arasında kalan alan hesaplanır. Performans ölçümü gerçek pareto cephesine ait hiperhacim değeri ile optimal-çözüm kümesinin hiperhacim değeri arasındaki oran ile ölçülür. Hiperhacim Denklem 7.5'deki şekilde hesaplanır [44].

$$HV_{Q,P^*} = vol(U_{i=1}^{|Q|} v_i) \tag{7.4}$$

$$HV = \frac{HV_Q}{HV_{P^*}} \tag{7.5}$$

7.2. TEST PROBLEMLERİNİN ÇOK-AMAÇLI RGO İLE ÇÖZÜMLERİ

HÇA-RGO algoritmasını performans açısından değerlendirmek için literatürde yer alan 7 farklı kısıtsız çok amaçlı test problemi kullanılmıştır. Kullanılan test fonksiyonları Çizelge 7.1'de gösterilmiştir. Bununla birlikte HÇA-RGO algoritmasına ait sonuçlar, NSGA-II ve Çok-Amaçlı Parçacık Sürü Optimizasyonu (MOPSO) gibi iyi bilinen algoritmalar kullanılarak elde edilen sonuçlar ile karşılaştırılmıştır [12], [38]. Her bir fonksiyon için kullanılan parametreler Çizelge 7.2'de gösterilmiştir. Deneysel sonuçların elde edilmesinde kullanılan sistemin temel özellikleri şu şekildedir:

- İşlemci : Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz
- RAM : 16.0 GB
- İşletim Sistemi : Windows 10 Enterprise N, x64
- Harddisk : 256 GB SSD

Aynı zamanda hata payını minimize etmek için her bir optimizasyon işlemi 30 tekrarlı test şeklinde uygulanmıştır.

Çizelge 7.1. Çok-amaçlı test problemleri.

#	Problem	Formül	Arama Uzayı
F1	Fonseca-Fleming [45]	$f_1(x) = 1 - e^{-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2}$ $f_2(x) = 1 - e^{-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2}$	$-4 \leq x_i \leq 4$ $1 \leq i \leq n$
F2	Kursawe [46]	$f_1(x) = \sum_{i=1}^{2} -10e^{-0.2\sqrt{x_i^2 + x_{i+1}^2}}$ $f_2(x) = \sum_{i=1}^{3} \lvert x_i \rvert^{0.8} + 5\sin(x_i^3)$	$-5 \leq x_i \leq 5$ $1 \leq i \leq 3$

Çizelge 7.1. (devam) Çok-amaçlı test problemleri.

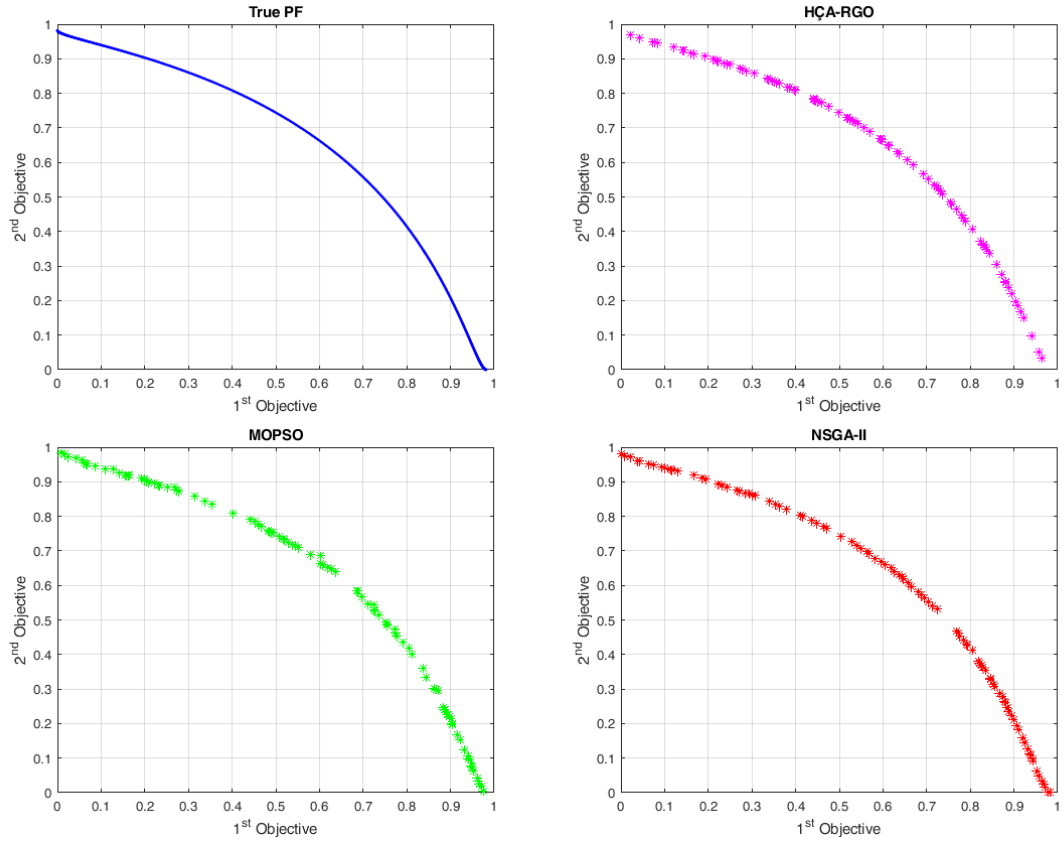
#	Problem	Formül	Arama Uzayı
F3	Poloni [47]	$f_1(x,y) = 1 + \left(A_1 - B_1(x,y)\right)^2 + \left(A_2 - B_2(x,y)\right)^2$ $f_2(x,y) = (x+3)^2 + (y+1)^2$ $A_1 = 0.5\sin(1) - 2\cos(1) + \sin(2) - 1.5\cos(2)$ $A_2 = 1.5\sin(1) - \cos(1) + 2\sin(2) - 0.5\cos(2)$ $B_1(x,y) = 0.5\sin(x) - 2\cos(x) + \sin(y) - 1.5\cos(y)$ $B_2(x,y) = 1.5\sin(x) - \cos(x) + 2\sin(y) - 0.5\cos(y)$	$-\pi \leq x, y \leq \pi$
F4	Schaffer N.2 [48]	$f_1(x) = \begin{cases} -x, & x \leq 1 \\ x-2, & 1 < x \leq 3 \\ 4-x, & 3 < x \leq 4 \\ x-4, & x > 4 \end{cases}$ $f_2(x) = (x-5)^2$	$-5 \leq x \leq 10$
F5	ZDT 1 [49]	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i$ $h\left(f_1(x), g(x)\right) = 1 - \sqrt{\frac{f_1(x)}{g(x)}}$	$0 \leq x_i \leq 1$ $1 \leq i \leq 30$
F6	ZDT 2 [49]	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i$ $h\left(f_1(x), g(x)\right) = 1 - \left(\sqrt{\frac{f_1(x)}{g(x)}}\right)^2$	$0 \leq x_i \leq 1$ $1 \leq i \leq 30$
F7	ZDT 3 [49]	$f_1(x) = x_1$ $f_2(x) = g(x)h(f_1(x), g(x))$ $g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i$ $h\left(f_1(x), g(x)\right) = 1 - \sqrt{\frac{f_1(x)}{g(x)}} - \frac{f_1(x)}{g(x)}\sin(10\pi f_1(x))$	$0 \leq x_i \leq 1$ $1 \leq i \leq 30$

Çizelge 7.2. Kullanılan parametreler.

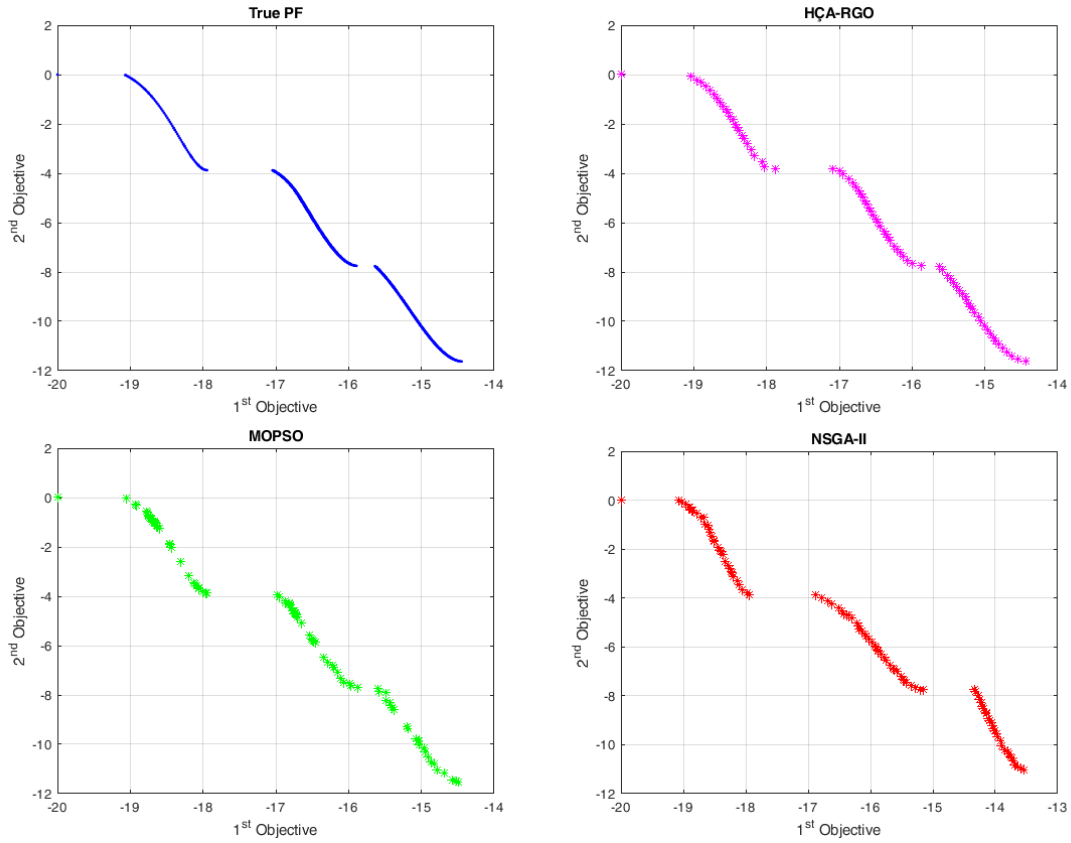
Parametreler	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
Popülasyon	100	100	100	100	100	100	100
Maks. İter.	300	300	300	300	500	500	500
MaxV.	0.1	0.3	0.3	0.1	1	1	1
MinV.	0.05	0.01	0.05	0.05	0.01	0.01	0.01
MinDist.	0.01	0.1	0.1	0.1	0.01	0.01	0.01

Kullanılan parametreler her test fonksiyonu için benzerlik göstermektedir. Her fonksiyon için popülasyon sayısı 100 olarak tanımlanmıştır. Maksimum iterasyon sayısı 300 olarak tanımlandığında pareto-optimal çözümler doyuma ulaşmaktadır. Karar verme değişkenlerinin fazla olduğu test fonksiyonlarında çözüm kümesini daha çeşitli hale getirmek için iterasyon sayısı arttırılmıştır. Diğer çok amaçlı algoritmalar incelendiğinde maksimum iterasyon sayısının 500 olması kabul edilebilir. Maksimum ve minimum hız değerleri algoritmanın alt ve üst limitlerine bağlı olarak değişmektedir. Minimum mesafe pareto-optimal çözümlerin birbirine ne kadar yakın olması gerektiğini göstermektedir. Yakınsama ve çeşitlilik için bu değer oldukça önemlidir. Algoritmanın belli noktalara takılıp kalmaması için minimum mesafe tanımlaması uygun şekilde yapılmalıdır.

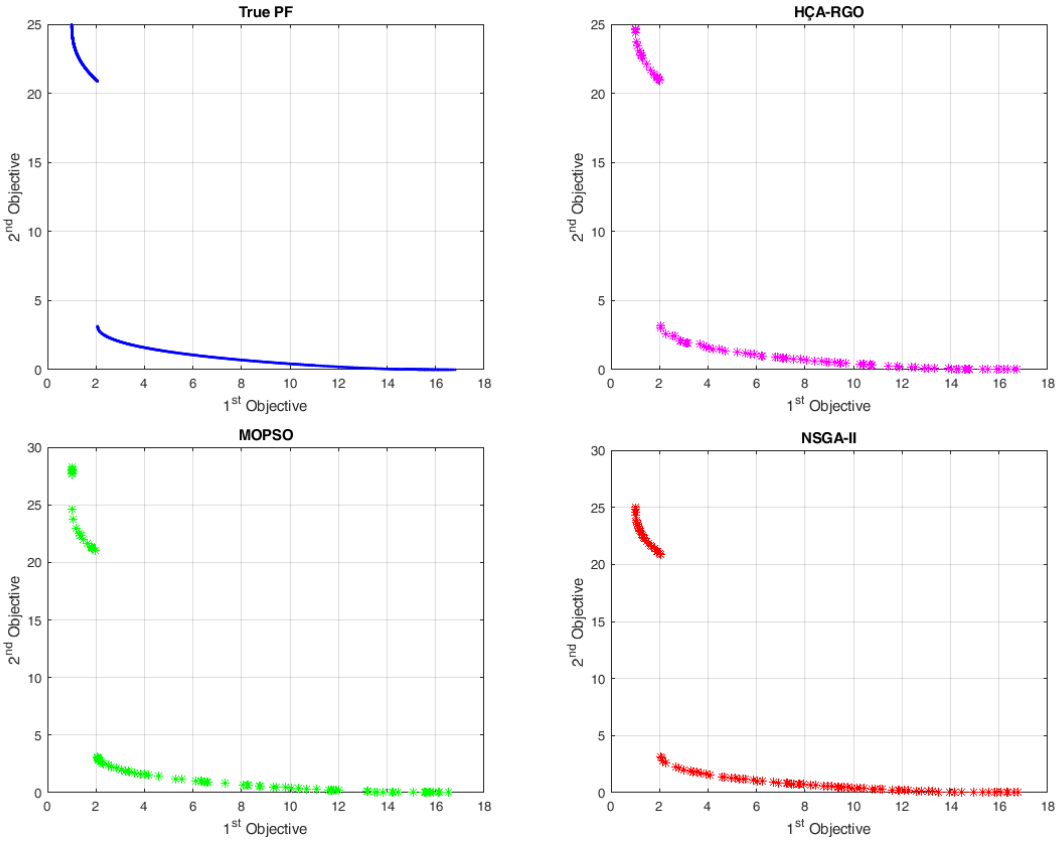
Her bir test fonksiyonu Çizelge 7.2'de yer alan parametreler kullanılarak NSGA-II, MOPSO ve HÇA-RGO algoritmalarına uygulanmıştır ve test sonuçlarına ait pareto cepheleri aşağıda yer almaktadır.



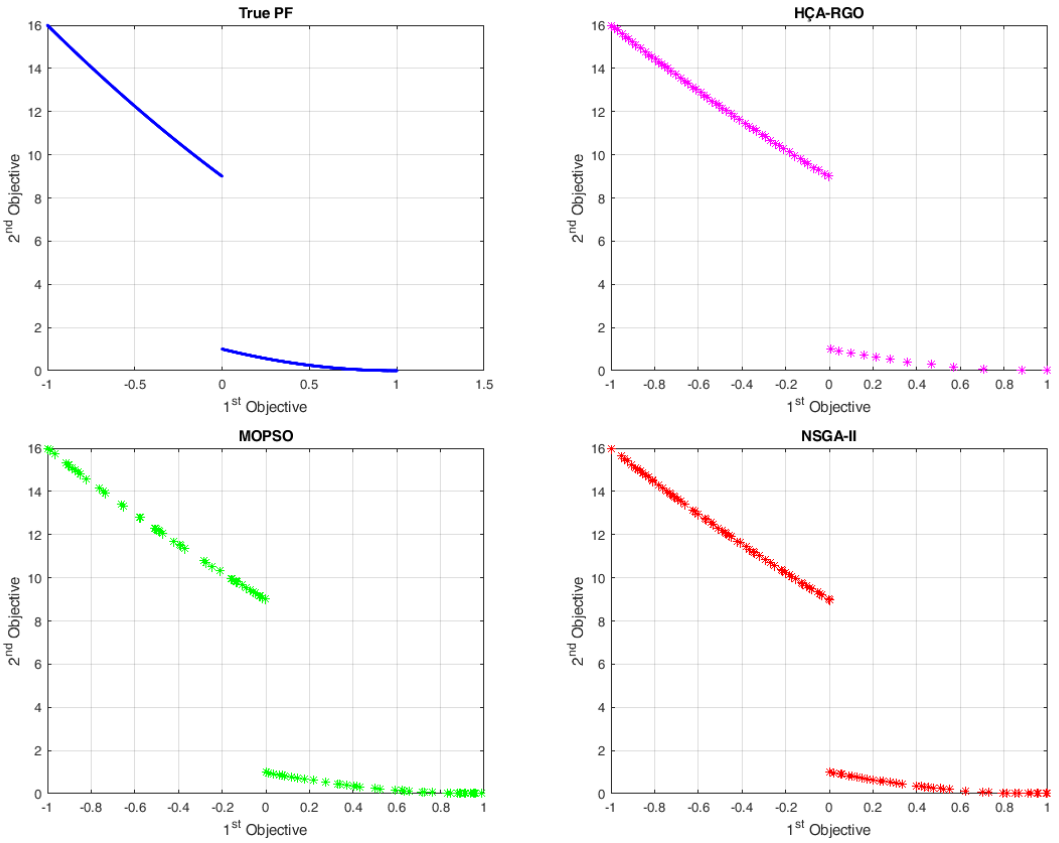
Şekil 7.3. Fonseca pareto cephesi.



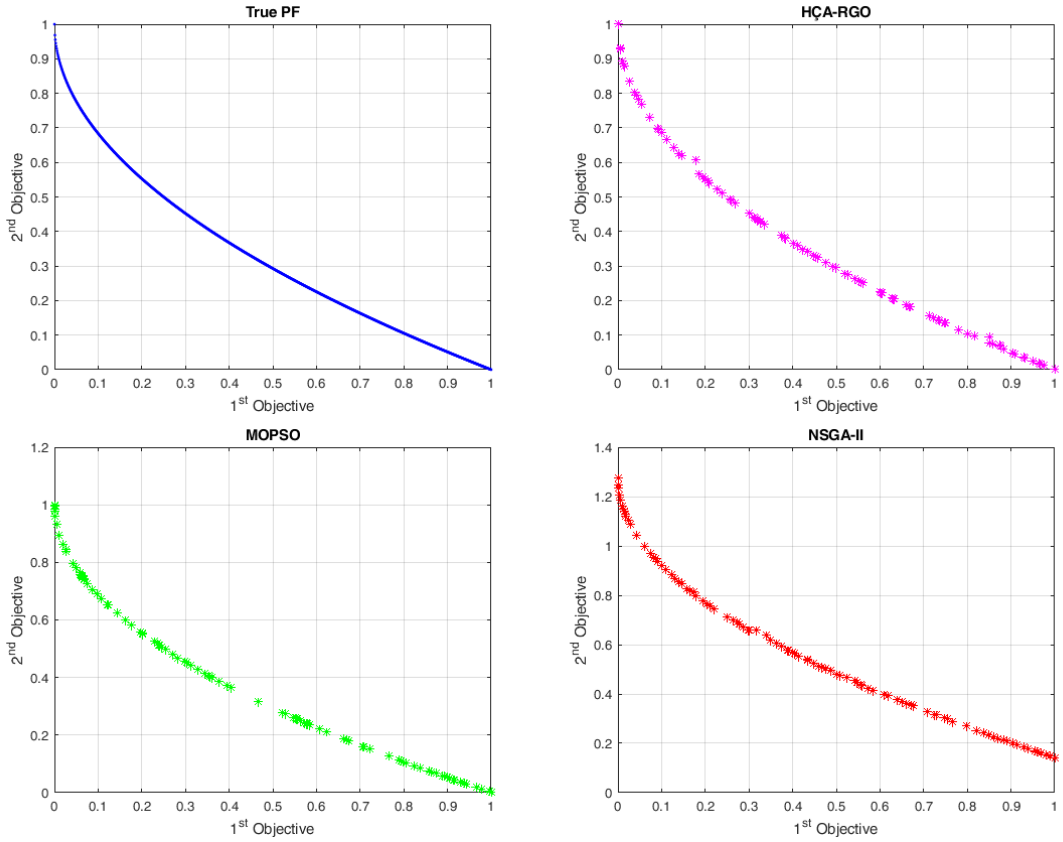
Şekil 7.4. Kursawe pareto cephesi.



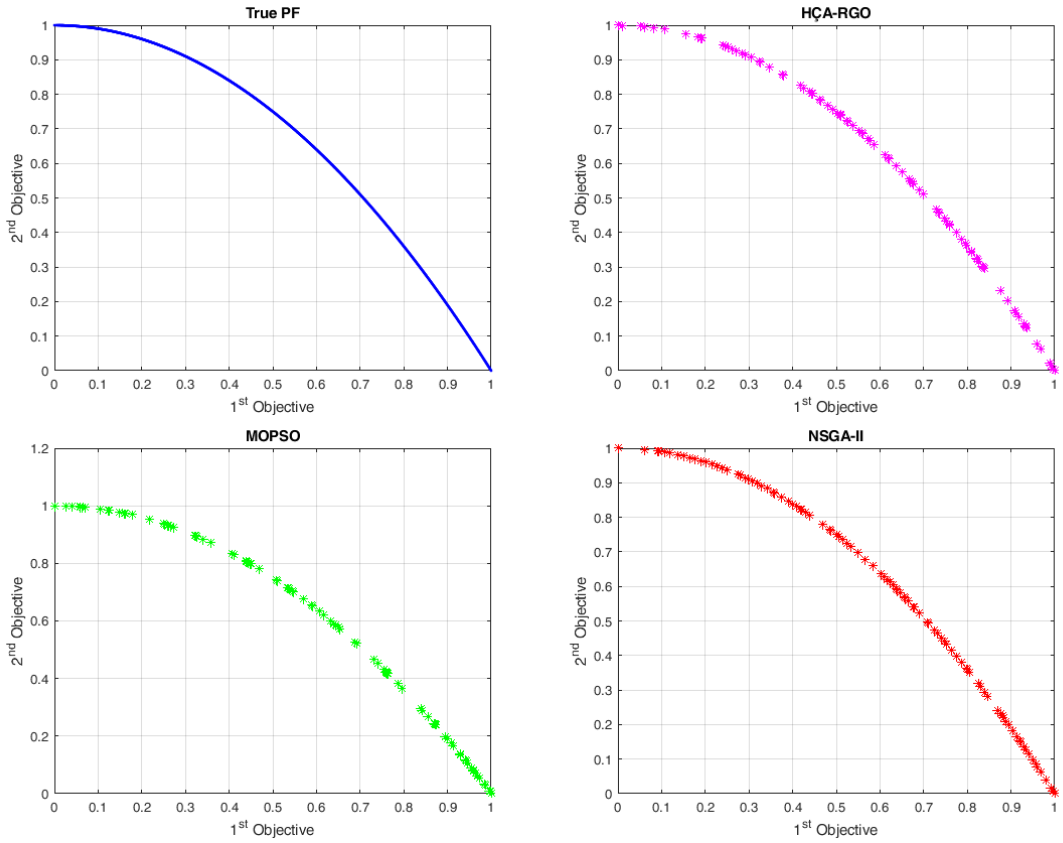
Şekil 7.5. Poloni pareto cephesi.



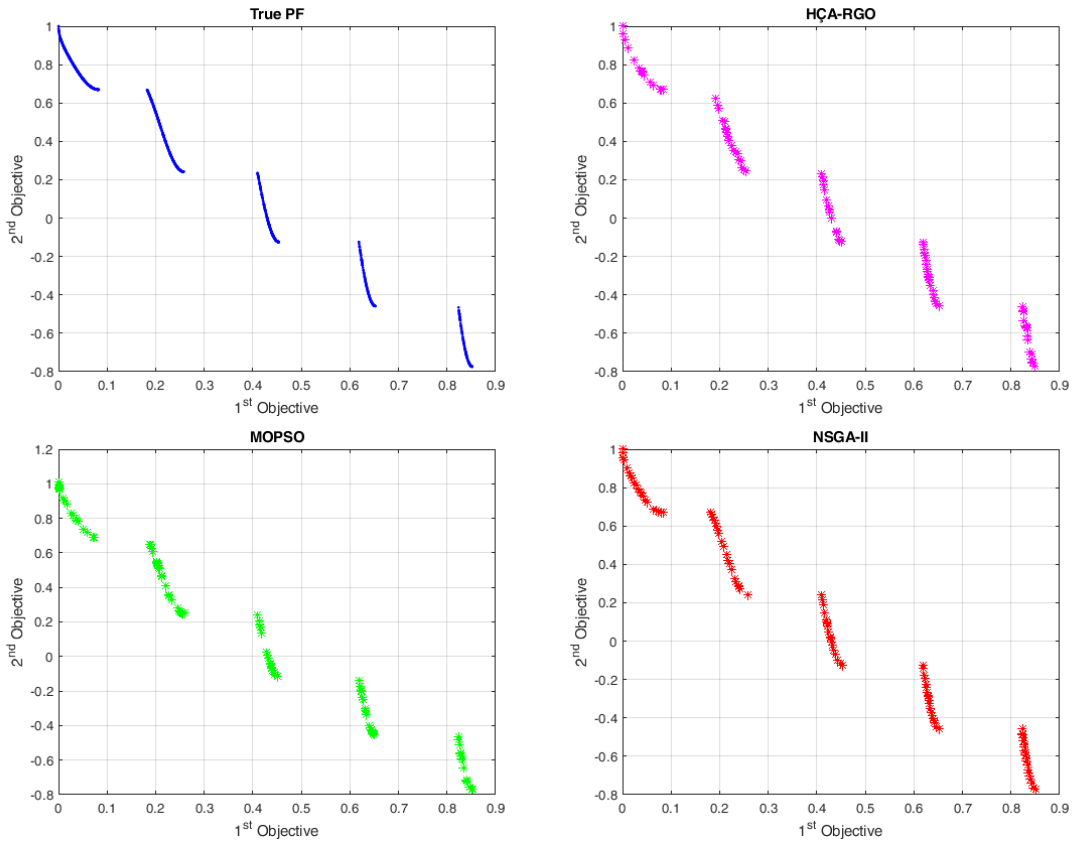
Şekil 7.6. SchafferN2 pareto cephesi.



Şekil 7.7. ZDT-1 pareto cephesi.



Şekil 7.8. ZDT-2 pareto cephesi.



Şekil 7.9. ZDT-3 pareto cephesi.

Mesafe (GD) metriğine ait veriler Çizelge 7.3'de gösterilmiştir. Ortalama ve standart sapma değerleri 30 tekrarlı test sonucunda elde edilmiştir. Ortalama değer her bir çözümün gerçek pareto-optimal çözüme ne kadar yakınsadığını ifade eder. Her bir algoritmaya ait ortalama minimum mesafe değerleri incelendiğinde 4 adet test fonksiyonunda HÇA-RGO diğer algoritmalardan daha iyi sonuçlar vermiştir. Karar değişkeni sayısı fazla olan F6 ve F7 fonksiyonlarında NSGA-II algoritması ön plana çıkarken HÇA- RGO ile kıyaslandığında sonuçlar oldukça yakındır. Standart sapma değeri pareto-optimal çözümlerin ne kadar çeşitli olduğunu ifade eder. Ortalama standart sapma değerleri incelendiğinde F1, F2 ve F4 fonksiyonlarında en iyi, diğer fonksiyonlarda ise en iyi çözüme yakın değerler elde edildiği gözlenmiştir.

Çizelge 7.3. Mesafe (GD) metriğine ait deneysel sonuçlar: Ortalama ($\bar{x}$) ve Standart Sapma (σ).

Algoritma		**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
HÇA-RGO	$\bar{x}$	**0.00408248**	**0.0390115**	**0.050263877**	0.028397946	**0.004625615**	0.006653173	0.017426835
	σ	**0.00238080**	**0.01936969**	0.060936042	**0.019493732**	0.005031273	0.004855082	0.015186426
NSGA-II	$\bar{x}$	0.00515361	0.85089600	0.062463868	**0.023169799**	0.004756987	**0.004728225**	**0.005777209**
	σ	0.00345436	0.33532186	**0.041788833**	0.021920297	**0.003577742**	**0.003737699**	**0.005372786**
MOPSO	$\bar{x}$	0.00731285	0.1681837	0.12612483	0.041097475	0.009171657	0.008377408	0.010020761
	σ	0.00445555	0.15326405	0.13389233	0.053504649	0.009676832	0.00906622	0.010876826

Hiperhacim (HV) metriği elde edilen pareto cephesinin ne kadar yakınsadığı ve ne kadar çeşitli olduğunu gösterir. Hiperhacim metriğine ait deneysel sonuçlar Çizelge 7.4'de gösterilmiştir. Gerçek paretoya ait hiperhacim değeri, elde edilen hiperhacim değerine bölünerek ikisi arasındaki orana bakılarak algoritmanın yakınsama ve çeşitlilik gösterdiği ifade edilebilir. Bu değer 1'e ne kadar yakın ise algoritmanın başarılı olduğu söylenebilir. Deneysel sonuçlarda yer alan ∇ sonucun 1'e ne kadar yaklaştığını gösteren bir metriktir. Buna göre önerilen algoritma HÇA-RGO F3, F5 ve F6 fonksiyonlarında en iyi F1, F2, F4 ve F7 fonksiyonlarında ise en iyi değere sahip algoritmaya oldukça yakın olduğu gözlenmiştir. Buna göre önerilen algoritmanın bilinen algoritmalarla karşılaştırıldığında yayılma ve çeşitlilik açısından başarılı sonuçlar verdiği gözlenmiştir.

Çizelge 7.4. Hiperhacim (HV) metriğine ait deneysel sonuçlar.

Algoritma		**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
	Ref.	**[1,1]**	**[-14,0]**	**[18,25]**	**[1,16]**	**[1,1]**	**[1,1]**	**[1,1]**
	HV_{P^*}	0.50329188	48.0152458	488.636277	19.3553298	0.66716012	0.33500768	0.46170975
HÇA-RGO	HV_Q	0.46369569	39.4516758	**288.438007**	16.6427558	**0.55017487**	**0.50380226**	0.76177852
	HV	0.92132558	0.8216489	**0.5902918**	0.85985390	**0.82465191**	**1.50385285**	1.64990779
	Δ	0.07867441	0.1783510	**0.4097081**	0.14014609	**0.17534808**	**0.50385285**	0.64990779

Çizelge 7.4. (devam) Hiperhacim (HV) metriğine ait deneysel sonuçlar.

Algoritma		**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
NSGA-II	HV_Q	0.54178414	42.8979437	266.207961	**13.6606601**	0.50028632	0.558190778	**0.773597578**
	HV	0.92333967	0.59572257	0.46578900	**0.8982297**	0.79596468	1.562294608	**1.611538281**
	Δ	0.07666032	0.40427742	0.53421099	**0.1017702**	0.20403531	0.562294608	**0.611538281**
MOPSO	HV_Q	**0.46470936**	**28.6037656**	227.601404	17.3855338	0.53103589	0.523380698	0.744062941
	HV	**1.07648098**	**0.8934233**	0.54479778	0.70578286	0.74987443	1.666202913	1.675506253
	Δ	**0.07648098**	**0.1065766**	0.45520221	0.29421713	0.25012556	0.666202913	0.675506253

Boşluk (S), komşu çözümler arasındaki mesafeyi ölçerek pareto cephesinin ne kadar iyi dağılım yaptığını ölçen performans metriğidir. Elde edilen değerin 0'a yakın olması pareto cephesinin eşit dağılım yaptığını gösterir. Boşluk metriğine ait deneysel sonuçlar Çizelge 7.5'de gösterilmiştir. Deneysel sonuçlara göre NSGA-II algoritması tüm test fonksiyonlarında en iyi değeri üretmiştir. Önerilen algoritma F1, F2, F4, F5, F6 ve F7 fonksiyonlarında NSGA-II algoritmasına oldukça yakın ve MOPSO'dan daha iyi sonuç ürettiği gözlenmiştir. Bu sonuçlara göre önerilen algoritmaya ait pareto-optimal çözümlerim başarılı bir şekilde eşit dağılım gösterdiği söylenebilir.

Çizelge 7.5. Boşluk (S) metriğine ait deneysel sonuçlar.

Algoritma		**F1**	**F3**	**F4**	**F5**	**F6**	**F7**
	S_{P^*}	0.001456406	0.000761082	6.89E-05	0.001239953	0.000394279	0.001489247
HÇA-RGO	S_Q	0.007101033	0.004963174	0.010670461	0.007839916	0.006812957	0.009404027
	Δ	0.005644627	0.004202093	0.010601573	0.006599963	0.006418678	0.00791478
NSGA-II	S_Q	**0.005355909**	**0.004182585**	**0.010139444**	**0.004999985**	**0.005986096**	**0.007441341**
	Δ	**0.003899503**	**0.003421504**	**0.010070555**	**0.003760032**	**0.005591817**	**0.005952094**

Çizelge 7.5. (devam) Boşluk (S) metriğine ait deneysel sonuçlar.

Algoritma		F1	F3	F4	F5	F6	F7
MOPSO	S_Q	0.007569572	0.00778685	0.011622765	0.009425079	0.006864305	0.00960395
	Δ	0.006113166	0.007025768	0.011553877	0.008185126	0.006470027	0.008114703

7.3. DOĞRUSAL OLMAYAN DENKLEM SİSTEMLERİNİN ÇOK-AMAÇLI RGO İLE ÇÖZÜMLERİ

Doğrusal olmayan denklem sistemleri; n adet bağımsız değişkeni ve n adet doğrusal olmayan denklemi olan sistemlerdir.

$$f_1(x_1, x_2, \dots, x_n) = 0\,, \tag{7.6}$$

$$f_2(x_1, x_2, \dots, x_n) = 0\,, \tag{7.7}$$

……

$$f_n(x_1, x_2, \dots, x_n) = 0 \tag{7.8}$$

Eşitlik değerleri sol tarafa atılacak olursa(denklemler sıfıra eşit olacak), doğrusal olmayan denklem sisteminin çözümü; n adet denklemi birlikte çözecek n adet kökün bulunması demektir. Bir denklemin kökü onu sıfır yapan veya mutlak minimum yapan değerdir [50].

Doğrusal olmayan denklem sistemlerinin çözümünde çok çeşitli yöntemler kullanılmaktadır. Bazı çalışmalarda doğrusal olmayan denklem sistemleri kısıtsız tek amaçlı optimizasyon problem olarak ele alınır. Bu tür problemlerde minimize edilecek amaç fonksiyonu, Denklem 7.9'dAki gibi doğrusal olmayan denklemlerin kareleri toplamı olarak kabul edilir.

$$fmin = \sum_{i=1}^{n} f_i\,(x_1, x_2, \dots, x_n)^2 \tag{7.9}$$

Ancak bu yaklaşım tüm denklemlerin kökünü bulmayı değil, toplam hatayı minimize etmeyi garanti eder.

Bir çok çalışmada ise doğrusal olmayan denklem sistemleri çok amaçlı optimizasyon problem olarak ele alınır [51], [52]. Her bir denklem mutlak olarak minimize edilmesi gereken bir amaç fonksiyonudur. Doğrusal olmayan denklem sistemlerini çok amaçlı optimizasyon problem olarak çözmenin avantajı, denklem sisteminin tek bir çözümü yerine alternatif çözümlerinin de optimizasyon algoritmaları ile bulunabilmesidir.

Doğrusal olmayan denklem sistemlerinin çözümünde sıklıkla kullanılan yöntemler [53]-[57]'de verilmiştir. Bu yöntemlerden en çok kullanılanlar aşağıda sıralanmıştır;

- Trust-Region [58]
- Broyden [59]
- Secant [55]
- Halley [57]

HÇA-RGO algoritmasını doğrusal olmayan denklem sistemlerinin çözümündeki performansını değerlendirmek için literatürde yer alan 4 farkı denklem sistemi kullanılmıştır. Kullanılan denklem sistemleri Çizelge 7.6'de gösterilmiştir.

Çizelge 7.6. Doğrusal olmayan denklem sistemleri.

Problem		**Formül**	**Arama Uzayı**
D1	[52]	$x_1{}^2 + x_2{}^2 - 1 = 0$ $x_1 - x_2 = 0$	$-1 \leq x_i \leq 1$ $1 \leq i \leq 2$
D2	[60]	$x_1{}^2 + 2x_2{}^2 + \cos(x_3) - x_4{}^2 = 0$ $3x_1{}^2 + x_2{}^2 + sin^2(x_3) - x_4{}^2 = 0$ $-2x_1{}^2 - x_2{}^2 - \cos(x_3) + x_4{}^2 = 0$ $-x_1{}^2 - x_2{}^2 - cos^2(x_3) + x_4{}^2 = 0$	$-2 \leq x_i \leq 2$ $1 \leq i \leq 4$
D3	Interval Aritmetic Benchmark [60]	$x_1 - 0.25428722 - 0.18324757x_4x_3x_9 = 0$ $x_2 - 0.37842197 - 0.16275449x_1x_{10}x_6 = 0$ $x_3 - 0.27162577 - 0.16955071x_1x_2x_{10} = 0$ $x_4 - 0.19807914 - 0.15585316x_7x_1x_6 = 0$ $x_5 - 0.44166728 - 0.19950920x_7x_6x_3 = 0$ $x_6 - 0.14654113 - 0.18922793x_8x_5x_{10} = 0$ $x_7 - 0.42937161 - 0.21180486x_2x_5x_8 = 0$ $x_8 - 0.07056438 - 0.17081208x_1x_7x_6 = 0$ $x_9 - 0.34504906 - 0.19612740x_{10}x_6x_8 = 0$ $x_{10} - 0.42651102 - 0.21466544x_4x_8x_1 = 0$	$-2 \leq x_i \leq 2$ $1 \leq i \leq 10$
D4	Neuro-physology Application [60]	$x_1{}^2 + x_3{}^2 = 1$ $x_2{}^2 + x_4{}^2 = 1$ $x_5x_3{}^3 + x_6x_4{}^3 = c_1$ $x_5x_1{}^3 + x_6x_2{}^3 = c_2$ $x_5x_1x_3{}^2 + x_6x_4{}^2x_2 = c_3$ $x_5x_1{}^2x_3 + x_6x_2{}^2x_4 = c_4$	$-1 \leq x_i \leq 1$ $1 \leq i \leq 6$ $1 \leq c \leq 4$ $c_i = 0$

Denklemlerin klasik bir yöntem olan Trust-Region metodu kullanılarak elde edilen çözümleri Çizelge 7.7'de gösterilmiştir [61].

Çizelge 7.7. Trust-Region ile doğrusal olmayan denklem sistemlerinin çözümleri.

Problem	**Karar Değişkeni**	**Çözüm**
D1	2	$x = (0.707107\,, 0.707107)$ $eq_1 = 0$ $eq_2 = 2.01E-7$
D2	4	$x = (-9.6E-08\,, -2.8E-09, 1.394741, 0.5349)$ $eq_1 = -0.11097$ $eq_2 = 0.683206$ $eq_3 = -0.46126$ $eq_4 = 0.110971$
D3	10	$x = (0.257833, 0.381097, 0.278745, 0.200669, 0.445251,$ $0.149184, 0.43201, 0.073403, 0.345967, 0.427326)$ $eq_1 = 1.88E-11$ $eq_2 = -3E-14$ $eq_3 = -3.7E-14$ $eq_4 = -2E-13$ $eq_5 = 1.09E-11$ $eq_6 = 2.06E-12$ $eq_7 = 4.7E-13$ $eq_8 = -2.2E-13$ $eq_9 = -1.3E-12$ $eq_{10} = -2.3E-12$
D4	6	$x = (0.754488, 0.954881, 0.656314, 0.296989, -1.4E11{,}4E-07)$ $eq_1 = 6.88E-15$ $eq_2 = 3E-09$ $eq_3 = 1.05E-08$ $eq_4 = 3.48E-07$ $eq_5 = 3.37E-08$ $eq_6 = 2.06E-12$

Bununla birlikte HÇA-RGO algoritmasına ait sonuçlar, test problemlerinde olduğu gibi NSGA-II ve Çok-Amaçlı Parçacık Sürü Optimizasyonu (MOPSO) kullanılarak elde edilen sonuçlar ile karşılaştırılarak Çizelge 7.8'de gösterilmiştir. Deneysel sonuçlar incelendiğinde test problemlerinin dışında doğrusal olmayan denklem sistemlerinin çözülmesinde başarılı sonuçlar elde ettiği ve algoritmanın kullanılabilir olduğu gözlenmiştir.

Çizelge 7.8. Doğrusal olmayan denklem sistemleri karşılaştırmalı sonuçlar: Ortalama ($\bar{x}$), Standart Sapma (σ) ve Toplam (Σ).

Denklemler		**HÇA- RGO**		**NSGA-II**		**MOPSO**	
		$\bar{x}$	Σ	$\bar{x}$	σ	$\bar{x}$	σ
D1	eq_1	0.000220	0.000422	**3.654401e-10**	**2.001598e-09**	4.171823e-05	9.180501e-05
	eq_2	0.000892	0.001860	**4.229778e-10**	**2.316744e-09**	2.081175e-05	4.296842e-05
	Σ	0.001113	0.001839	**7.884180e-10**	**4.318343e-09**	6.252999e-05	0.000105
D2	eq_1	**0.045320**	**0.051632**	0.211931	0.166559	0.194218	0.152565
	eq_2	**0.389621**	**0.068497**	0.586158	0.366147	0.435695	0.122648
	eq_3	0.352602	**0.088929**	**0.089188**	0.144438	0.174631	0.147708
	eq_4	**0.046171**	**0.052305**	0.272755	0.202010	0.316751	0.150192
	Σ	**0.833715**	**0.105726**	1.160033	0.243731	1.121297	0.178830
D3	eq_1	**0.117913**	**0.171911**	1.549991	1.381743	0.624084	0.734583
	eq_2	**0.114158**	**0.174627**	1.093895	1.288475	0.626450	0.823823
	eq_3	**0.093223**	**0.156028**	0.846264	1.041580	0.675210	0.718247
	eq_4	**0.097002**	**0.113068**	1.084293	0.973015	0.455211	0.565928
	eq_5	**0.080523**	**0.096309**	0.949058	1.135044	0.827122	0.704204
	eq_6	**0.114201**	**0.207663**	0.889087	1.176545	0.748125	0.765581
	eq_7	**0.090627**	**0.119581**	1.211447	1.303337	0.764480	0.912182
	eq_8	**0.107067**	**0.204127**	0.928754	1.094620	0.534452	0.523205
	eq_9	**0.123530**	**0.205318**	1.027816	1.160161	0.621151	0.574184
	eq_{10}	**0.150019**	**0.183516**	0.730268	0.920892	0.387760	0.666612
	Σ	**1.088267**	**0.370239**	10.31087	3.192165	6.264050	2.184322
D4	eq_1	0.035533	0.042967	0.131760	0.168764	**0.012220**	**0.016821**
	eq_2	0.026702	0.025672	0.124733	0.182261	**0.010031**	**0.016915**
	eq_3	0.020143	0.030866	0.113107	0.103906	**0.004673**	**0.006881**
	eq_4	0.020556	0.030583	0.117179	0.112201	**0.010950**	**0.014421**
	eq_5	0.013941	**0.021771**	0.167275	0.151491	**0.012111**	0.021989
	eq_6	0.012505	0.015368	0.178582	0.169187	**0.007654**	**0.011480**
	Σ	0.129382	0.065639	0.832637	0.361976	**0.057642**	**0.040228**

8. SONUÇLAR VE ÖNERİLER

Bu çalışmada, çok-amaçlı optimizasyon problemlerini çözmek için RGO algoritması kullanılarak hibrit bir yaklaşım geliştirilmiştir. Geliştirilen bu hibrit yaklaşımda yakınsama için baskın olmayan sıralama algoritması, çeşitlik için ise adaptif ızgara yaklaşımı kullanılmıştır. Aynı zamanda yakın komşuluk temizleme ve yakınsama derecelendirme ile algoritmanın tek bir çözüme takılmayıp çözüm uzayını çeşitlendirmesi sağlanmıştır. HÇA-RGO, literatürde iyi bilinen NSGA-II ve MOPSO algoritmaları ile karşılaştırılmıştır. Çok-amaçlı test problemlerine ait deneysel sonuçlar incelendiğinde hibrit yaklaşımın 7 test probleminin 4'ünde en iyi sonucu verdiği, doğrusal olmayan denklem sistemlerinin çözümlerine ait deneysel sonuçlar incelendiğinde ise hibrit yaklaşımın 4 problemin 2'sinde en iyi, diğer 2 problemde ise en iyiye oldukça yakın sonuçlar verdiği gözlenmiştir. Geliştirilen hibrit yaklaşımın elde edilen bulgular neticesinde yakınsama ve çeşitlilik kriterleri temel alındığında literatürde kabul edilebilir olduğunu gözlenmiştir.

9. KAYNAKLAR

[1] P. Erdoğmuş, "Doğadan esinlenen optimizasyon algoritmaları ve optimizasyon algoritmalarının optimizasyonu," *Düzce Üniversitesi Bilim ve Teknoloji Dergisi,* c. 4, sayı 1, ss. 293-304, 2016.

[2] J. H. Holland, "Control and artificial intelligence," *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology*, Cambridge, İngiltere: MIT Press, 1992.

[3] J. Kennedy & R. Eberhart, "Particle swarm optimization," *International Conference on Neural Networks*, Perth, WA, Avustralya, 1995, ss. 1942-1948.

[4] M. Dorigo, M. Birattari & T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine,* c. 1, sayı 4, ss. 28-39, 2006.

[5] M. Laumanns, L. Thiele, K. Deb & E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, c. 10, sayı 3, ss. 263-282, 2002.

[6] N. Srinivas & K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, c. 2, sayı 3, ss. 221-248, 1994.

[7] S. Rostami & A. Shenfield, "A multi-tier adaptive grid algorithm for the evolutionary multi-objective optimisation of complex problems," *Soft Computing*, c. 21, sayı 17, ss. 4963–4979, 2017.

[8] D. E. Goldberg, "Genetic algorithms and classifier systems," *Genetic Algorithms in Search, Optimization and Machine Learning*, New York, ABD: Addison-Wesley Publishing Company, 1989, böl. 3, ss. 60-61.

[9] M. F. P. Costa, A. M. A. C. Rocha & E. M. G. P. Fernandes, "Combining nondominance, objective-order and spread metric to extend Firefly Algorithm to multi-objective optimization," *International Conference On Evolutionary Multi-Criterion Optimization*, Guimares, Portekiz, 2015, ss. 292-306.

[10] C. M. Fonseca & P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation discussion and generalization," *International Computer Games Association*, c. 93, ss. 416-423, 1993.

[11] J. Horn, N. Nafpliotis & D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," *IEEE World Congress on Computational Intelligence*, Orlando, ABD, 1994, ss. 82-87.

[12] K. Deb, S. Agrawal, A. Pratap & T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," *International Conference On Parallel Problem Solving From Nature*, Berlin, Almanya, 2000, ss. 849-858.

[13] S. Mirjalili, S. M. Mirjalili & A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, c. 69, ss. 46-61, 2014.

[14] S. Mirjalili, S. Saremi, S. M. Mirjalili & L. S. Coelho, "Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization," *Expert Systems with Applications*, c. 47, ss. 106-119, 2016.

[15] S. Saremi, S. Mirjalili & A. Lewis, "Grasshopper optimization algorithm: Theory and application," *Advances in Engineering Software*, c. 105, ss. 30-47, 2017.

[16] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris & I. Aljarah, "Grasshopper optimization algorithm for multi-objective optimization problems," *Applied Intelligence*, c. 48, ss. 805-820, 2018.

[17] S. Y. Zeng, L. S. Kang & L. X. Ding, "An orthogonal multi-objective evolutionary algorithm for multi-objective optimization problems with constraints," *Evolutionary Computation*, c. 12, sayı 1, ss. 77-98, 2004.

[18] M. Janga Reddy & D. Nagesh Kumar, "An efficient multi-objective optimization algorithm based on swarm intelligence for engineering design," *Engineering Optimization*, c. 39, sayı 1, ss. 49-68, 2007.

[19] O. Jadaan, L. Rajamani & C. Rao, "Non-dominated ranked genetic algorithm for solving multi-objective optimization problems: NRGA," *Journal of Theoretical and Applied Information Technology*, ss. 60-67, 2008.

[20] H. Ghiasi, D. Pasini & L. Lessard, "A non-dominated sorting hybrid algorithm for multi-objective optimization of engineering problems," *Engineering Optimization*, c. 43, ss. 39-59, 2011.

[21] H. Nobahari, M. Nikusokhan & P. Siarry, "A multi-objective gravitational search algorithm based on non-dominated sorting," *International Journal of Swarm Intelligence Research*, c. 3, sayı 3, ss. 32-49, 2012.

[22] S. Salcedo-Sanz, A. Pastor-Sánchez, J. A. Portilla-Figueras ve L. Prieto, "Effective multi-objective optimization with the coral reefs optimization algorithm," *Engineering Optimization*, c. 48, ss. 966-984, 2015.

[23] Z. Bayraktar & M. Komurcu, "Multiobjective adaptive wind driven optimization," *The 8th International Conference on Computational Intelligence (IJCCI)*, Porto, Portekiz, 2016, ss. 115-120.

[24] J. L. Lagrange & A. L. Cauchy, "Abschnitt. Partielle differentialgleichungen erster ordnung mit zwei unabhängigen veränderlichen," *Einführung die heorie der partiellen Differentialgleichungen*, Berlin, Almanya: De Gruyter, 1910.

[25] M. Canayaz, "Cırcır böceği algoritması: Yeni bir meta-sezgisel yaklaşım ve uygulamaları," Doktora tezi, Bilgisayar Mühendisliği, Fen Bilimleri Enstitüsü, İnönü Üniversitesi, Malatya, Türkiye, 2015.

[26] Ö. T. Altınöz & A. E. Yılmaz, "Parçacık sürü optimizasyonunda yeni bir birey davranış biçimi önerisi," *13. Elektrik, Elektronik, Bilgisayar, Biyomedikal Mühendisliği Ulusal Kongresi*, Ankara, Türkiye, 2009.

[27] R. Eberhart & J. Kennedy, "A new optimizer using particle swarm theory," *The Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japonya, 1995, ss. 39-43.

[28] M. Clerc & J. Kennedy, "The particle swarm: Explosion, stability and convergence in a multi-dimensional complex space," *Evolutionary Computation*, c. 6, ss. 58-73, 2002.

[29] Y. Shi & R. Eberhart, "A modified particle swarm optimizer," *1998 IEEE International Conference on Evolutionary Computation Proceedings,* ABD, 1998, ss. 69-73.

[30] J. H. Holland, "Genetic algorithms and adaptation." *Adaptive Control of Ill-Defined Systems*, Boston, ABD: Springer, 1984, ss. 317-333.

[31] D. E. Goldberg & K. Deb, "A comparative analysis of selection schemes used in

genetic algorithms," *The First Workshop on Foundations of Genetic Algorithms*, Bloomington Campus, Indiana, ABD, 1991, ss. 69-93.

[32] F. Altıparmak, "Genetik algoritma ile haberleşme şebekelerinin topolojik optimizasyonu," Doktora tezi, Endüstri Mühendisliği, Fen Bilimleri Enstitüsü, Gazi Üniversitesi, Ankara, Türkiye, 1996.

[33] L. D. Mech, "Alpha status, dominance, and division of labor in wolf packs," *Canadian Journal of Zoology*, c. 77, sayı 8, ss. 1196-1203, 1999.

[34] N. Muangkote, K. Sunat & S. Chiewchanwattana, "An improved grey wolf optimizer for training q-gaussian radial basis functional-link nets," *International Computer Science and Engineering Conference ICSEC*, Khon Kaen, Tayland, 2014.

[35] Z. Bayraktar, M. Kömürcü, J. A. Bossard & D. H. Werner, "Wind driven optimization (WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics," *Antennas and Propagation Society International Symposium (APSURSI)*, Toronto, Kanada, 2010, ss. 1-4.

[36] K. Deb, "Pareto-optimal tecniques" *Multiobjective Optimization Using Evolutionary Algorithms*, New York, ABD: Wiley, 2001.

[37] B. Min, A. Y. Sun, M. F. Wheeler & H. Jeong, "Utilization of multiobjective optimization for pulse testing dataset from a CO2-EOR/sequestration field," *Journal of Petroleum Science and Engineering*, c. 170, ss. 244-266, 2018.

[38] K. E. Parsopoulos & M. N. Vrahatis, "Particle swarm optimization method in multiobjective problems," *Association for Computing Machinery Symposium on Applied Computing*, Madrid, İspanya, 2002, ss. 603-607.

[39] J. Knowles & D. Corne, "The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimisation," *Congress on Evolutionary Computation*, Washington, DC, ABD, 1999, ss. 98-105.

[40] A. H. F. Dias & J. A. Vasconcelos, "Multiobjective genetic algorithms applied to solve optimization problems," *IEEE Transactions On Magnetics,* c. 38, sayı 2, ss. 1133-1136, 2002.

[41] K. Mahesh, P. Nallagownden & I. Elamvazuthi, "Advanced pareto front non-dominated sorting multi-objective particle swarm optimization for optimal placement and sizing of distributed generation," *Energies*, c. 9, sayı 12, 2016.

[42] D. A. V. Veldhuizen & G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Air Force Institute of Technology, ABD, Rap. TR-98-03, 1998.

[43] C. Coello & N. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, c. 6, ss. 163-190, 2005.

[44] S. Mirjalili, A. Lewis & J. Dong, "Confidence-based robust optimisation using multi-objective meta-heuristics," *Swarm and Evolutionary Computation*, c. 43, ss. 109-126, 2018.

[45] C. M. Fonseca & P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms-part II: Application example," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, c. 28, ss. 38-47, 1998.

[46] F. Kursawe, "A variant of evolution strategies for vector optimization," *The 1st Workshop On Parallel Problem Solving From Nature*, Berlin, Almanya, 1991,

ss. 193-197.

[47] C. Poloni, “Hybrid GA for multiobjective aerodynamic shape optimization,” *Genetic Algorithms in Engineering and Computer Science*, ss. 397-415, 1997.

[48] J. D. Schaffer, “Multiple objective optimization with Vector Evaluated Genetic Algorithms,” *The First International Conference On Genetic Algorithms*, ABD, 1985, ss. 93-100.

[49] E. Zitzler, K. Deb & L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” *Evolutionary Computation*, c. 8, sayı 2, ss. 173-195, 2000.

[50] P. Erdogmus, “Parçacık sürü optimizasyonu ile doğrusal olmayan denklem köklerinin bulunması ve genetik algoritma ile mukayesesi,” *İleri Teknoloji Bilimleri Dergisi*, c. 5, sayı 1, 2016.

[51] S. Qin, S. Zeng, W. Dong & X. Li, “Nonlinear equation system solved by many-objective hype,” *IEEE Congress On Evolutionary Computation (CEC)*, Sendai, Japonya, 2015, ss. 2691-2696.

[52] W. Song, Y. Wang, H. X. Li & Z. Cai, “Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization,” *IEEE Transactions On Evolutionary Computation*, c. 19, sayı 3, ss. 414-431, 2015.

[53] C. Brezinski, “Projection methods for systems of equations,” *Journal of Computational and Applied Mathematics,* c. 7, ss. 35-51, 1997.

[54] J. E. Dennis, “On newton like methods,” *Numerische Mathematik*, c. 11, ss. 324-330, 1968.

[55] J. E. Dennis, “On the convergence of broyden's method for nonlinear systems of equations,” *Mathematics of Computation*, c. 25, ss. 559-567, 1971.

[56] J. E. Dennis, M. El-Alem & K. Williamson, “A trust-region approach to nonlinear systems of equalities and inequalities,” *Society for Industrial and Applied Mathematics Journal on Optimization*, c. 9, sayı 2, ss. 291-315, 1999.

[57] J. M. Ortega & W. C. Rheinboldt, “On the convergence of Halley's method,” *Iterative Solution of Nonlinear Equations in Several Variables*, New York, ABD: Academic Press, 1970, böl. 2, ss. 179-179.

[58] J. Nocedal, S. J. Wright, “Trust-Region methods,” *Numerical Optimization*, New York, ABD: Springer, 2006, böl. 2, ss. 66-100.

[59] C. G. Broyden, “A class of methods for solving nonlinear simultaneous equations,” *Mathematics of Computation*, c. 19, ss. 577-593, 1965.

[60] C. Grosan & A. Abraham, “A new approach for solving nonlinear equations systems,” *IEEE Transactions On Systems, Man, and Cybernetics - Part A: Systems and Humans*, c. 38, ss. 698-714, 2008.

[61] P. Erdoğmuş, “A new solution approach for non-linear equation systems with grey wolf optimizer,” *Sakarya University Journal of Computer and Information Sciences*, c. 1, sayı 3, ss. 1-11, 2018.

[62] Z. Bayraktar & M. Kömürcü, “Adaptive wind driven optimization,” *9th International Conference On Bio-Inspired Information and Communications Technologies*, New York, ABD, 2016, ss. 124-127.

[63] J. J. E. Dennis, “On Newton's method and nonlinear simultaneous replacements,” *Society for Industrial and Applied Mathematics Journal on Numerical Analysis*, c. 4, ss. 103-108, 1967.

[64] J. J. E. Dennis & H. Wolkowicz, “Least change secant methods sizing and shifting,” *Society for Industrial and Applied Mathematics Journal on Numerical Analysis*, c. 30, ss. 1291-1314, 1993.

[65] C. A. C. Coello & M. S. Lechuga, “MOPSO: a proposal for multiple objective particle swarm optimization,” *Congress on Evolutionary Computation*, Honolulu, HI, ABD, 2002, ss. 1051-1056.

[66] L. Dasheng, “Multi objective particle swarm optimization: algorithms and applications,” Doktora tezi, National University of Singapore, Singapur, 2008.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Fethiye Sultan ÖZPEHLİVAN AY

Doğum Tarihi ve Yeri : 30.05.1991 / Bakırköy

Yabancı Dili : İngilizce

E-posta : ozpehlivanfs@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Yüksek Lisans	Bilgisayar Müh.	Düzce Üniversitesi	2020
Lisans	Bilgisayar Müh.	Düzce Üniversitesi	2015
Lise		Fatih Kız Lisesi	2009

YAYINLAR

F. Özpehlivan Ay ve P. Erdoğmuş , "Hibrit Çok-Amaçlı Rüzgar Güdümlü Optimizasyon Algoritması", Düzce Üniversitesi Bilim ve Teknoloji Dergisi, c. 8, sayı. 4, ss. 2566-2582, 2020. doi:10.29130/dubited.765012.