

**T.C.
ARTVİN ÇORUH ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
İSTATİSTİK ANA BİLİM DALI
İSTATİSTİK PROGRAMI**

**DOĞADAN ESİNLENMİŞ HİBRİT BİR OPTİMİZASYON
ALGORİTMASININ YOL PLANLAMA PROBLEMİNE
UYGULANMASININ İSTATİSTİKSEL SONUÇLARI**

YÜKSEK LİSANS

Merve GENÇ

Danışman

Doç. Dr. KADRI DOĞAN

OCAK, 2026

BİLİMSEL ETİK BEYANNAMESİ

Artvin Çoruh Üniversitesi Lisansüstü Eğitim Enstitüsüne Yüksek Lisans Tezi olarak sunduğum “Doğadan Esinlenmiş Hibrit Bir Optimizasyon Algoritmasının Yol Planlama Problemine Uygulanmasının İstatistiksel Sonuçları başlıklı bu çalışmayı;

☒ Baştan sona kadar danışmanım Doç. Dr. Kadri Doğan ’ın sorumluluğunda tamamladığımı, başka kaynaklardan aldığım bilgileri metin içerisinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı taahhüt ederim.

☐ Turnitin benzerlik programından alınmış olan özgünlük raporuna göre tezin benzerlik oranının % 8 olduğunu beyan ederim.

Aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim.

Lisansüstü Eğitim-Öğretim yönetmeliğinin ilgili maddeleri uyarınca gereğinin yapılmasını arz ederim.

☒ Tezimin tamamı her yerden erişime açılabilir.

☐ Tezim sadece Artvin Çoruh Üniversitesi yerleşkelerinden erişime açılabilir.

☐ Tezimin ... ay süreyle erişime açılmasını istemiyorum. Bu sürenin sonunda uzatma için başvuruda bulunmadığım takdirde, tezimin tamamı her yerden erişime açılabilir.

...../...../20.....

Merve GENÇ

ORCID NO: 0000-0001-5090-2342

İmza

T.C.
ARTVİN ÇORUH ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ
İSTATİSTİK ANA BİLİM DALI
İSTATİSTİK PROGRAMI

DOĞADAN ESİNLENMİŞ HİBRİT BİR OPTİMİZASYON ALGORİTMASININ
YOL PLANLAMA PROBLEMİNE UYGULANMASININ İSTATİSTİKSEL
SONUÇLARI

Merve Genç
ORCID NO: 0000-0001-5090-2342

Başkan	: Ünvanı Adı SOYADI	Prof. Dr. Faik GÜRSOY
Jüri Üyesi	: Ünvanı Adı SOYADI	Prof. Dr. Mustafa Çağatay KORKMAZ
Jüri Üyesi	: Ünvanı Adı SOYADI	Doç. Dr. Kadri DOĞAN

ONAY:

Bu Yüksek Lisans / Doktora tezi, Artvin Çoruh Üniversitesi Lisansüstü Eğitim Enstitüsü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından .../.../..... tarihinde oybirliği/oyçokluğu ile uygun görülmüş ve kabul edilmiştir.

.../.../.....

Prof. Dr. Kerem ÇOŞKUN

Enstitü Müdürü

ÖN SÖZ

“Doğadan Esinlenmiş Hibrit Bir Optimizasyon Algoritmasının Yol Planlama Problemine Uygulanmasının İstatistiksel Sonuçları” konusunda yapılan bu çalışma; Artvin Çoruh Üniversitesi Lisansüstü Eğitim Enstitüsü İstatistik Anabilim Dalında yüksek lisans tezi olarak hazırlanmıştır.

Tez konusundaki bu araştırma için beni yönlendiren, literatür araştırmalarımnda karşılaştığım zorlukları bilgi ve tecrübesi ile aşmamda yardımcı olan, aynı zamanda elde edilen verilerin analiz edilmesinde ve tezin yazım aşamasında yardımlarını esirgemeyen, sadece bilgiyle sınırlı kalmayıp hayata bakış açımı fener olan ve mesleğimle ilgili yolumu aydınlatan değerli danışman hocam, Doç. Dr. Kadri DOĞAN ’a teşekkürlerimi sunarım.

Lisansüstü eğitim süreci içerisinde verdikleri değerli bilgilere ek olarak güler yüzü ve samimiyetlerini eksik etmeyen hocalarım Prof. Dr. Mustafa Çağatay KORKMAZ ve Doç. Dr. Zeynep ÖZTÜRK ’e teşekkür ederim.

İlime olan inanç, öğrenmeye olan ilgi ve hevesleri ile her zaman desteğini benden esirgemeyen sevgili aileme teşekkür ederim.

Merve GENÇ

Artvin – 2026

İÇİNDEKİLER

KISALTMA DİZİNİ	VIII
ŞEKİL LİSTESİ	X
TABLO LİSTESİ.....	XII
ÖZET	XIII
ABSTRACT	XIV
1 GİRİŞ.....	1
1.1 Optimizasyon	1
1.2 Metasezgisel Algoritmalar ve Optimizasyon	6
1.2.1 Balina Optimizasyon Algoritması (Whale Optimization Algorithm - WOA) 10	
1.2.2 Çoklu Stratejili Balina Optimizasyon Algoritması (Multi-Strategy Whale Optimization Algorithm-MSWOA)	12
1.2.3 Yapay Arı Koloni Algoritması (Artificial Bee Colony Algorithm - ABC).....	14
1.2.4 Karınca Kolonisi Algoritması (Ant Colony Optimization Algorithm - ACO) 15	
1.2.5 Gri Kurt Optimizasyonu (Grey Wolf Optimizer-GWO).....	17
1.2.6 Geliştirilmiş Gri Kurt Optimizasyonu (Improved Grey Wolf Optimization-IGWO).....	20
1.2.7 Parçacık Sürü Optimizasyonu (Particle Swarm Optimization-PSO)	21
1.2.8 Sinüs Kosinüs Algoritması (Sine Cosine Algorithm - SCA)	25
1.2.9 Yer çekimsel Arama Algoritması (A Gravitational Search Algorithm – GSA)	26
1.2.10 Yapay Atom Algoritması	26
1.2.11 Farksal Gelişim Algoritması (Differential Evolution-DE).....	26
1.2.12 Genetik Algoritması (Genetic Algorithm –GA).....	27
1.2.13 Yarasa Optimizasyonu (Bat Algorithm - BA).....	30
1.2.14 Karga Arama Algoritması	31
1.2.15 Optik Mikroskop Algoritması (Optical Microscope Algorithm - OMA).....	32
1.2.16 Gübre Böceği Optimizasyon Algoritması (Dung Beetle Optimizer-DBO)	33
1.2.17 Kum Kedisi Sürüsü Algoritması (Sand Cat Swarm Optimization-SCSO)	33
1.2.18 Dikenli Kertenkele Optimizasyon Algoritması (Horned Lizard Optimization Algorithm-HLOA).....	34
1.2.19 Doğrusal Olmayan Tabanlı Kaotik Harris Şahinleri Optimizasyon Algoritması (Nonlinear Chaotic Harris Hawks Optimization Algorithm-NCHHO).....	35

1.2.20 Uyarlamalı Spiral Uçuşlu Serçe Arama Algoritması (Adaptive Spiral Flying Sparrow Search Algorithm-ASFSSA).....	35
1.2.21 Kara Kanatlı Yırtıcı Kuşu (Sungur, Doğan vb.) Algoritması (Black Winged Kite Algorithm-BKA)	36
1.2.22 Altın Çakal Optimizasyonu Algoritması (Golden Jackal Optimization Algorithm-GJO)	36
1.2.23 Pelikan Optimizasyonu Algoritması (Pelican Optimization Algorithm-POA)37	
1.2.24 Şempanze Optimizasyonu Algoritması (Chimp Optimization Algorithm-ChOA)	38
1.2.25 Cüce Mangus Arama Algoritması (Dwarf Mongoose Optimization Algorithm-DMOA)	38
1.2.26 Popülasyon Tabanlı Çok Amaçlı Güve Algoritması (Population based Multi Objective Moth Algorithm - PMOMA)	39
1.2.27 Zebra Optimizasyon Algoritması (Zebra Optimization Algorithm - ZOA)....	40
1.2.28 Coati Optimizasyon Algoritması (COA).....	41
1.2.29 Walrus Optimizasyon Algoritması (WaOA).....	44
1.2.30 Dijkstra Algoritması	44
1.3 Hibrit Algoritmalar.....	49
1.3.1 Genetik-Simpleks Hibrit Algoritması	50
1.3.2 Geliştirilmiş Hibrit Zebra Algoritması (Zebra Hunt Optimization-ZHO)	53
1.3.3 Yarasa Optimizasyonu (BA) ve Farksal Gelişim Algoritması (Differential Evolution-DE) ile Oluşturulan Hibrit Algoritma (MBADE)	55
1.3.4 BA ve ABC ile Oluşturulan Hibrit Algoritma (BA_ABC)	56
1.3.5 Hibrit Genetik-Karga Algoritma	58
1.3.6 Hibrit Genetik Algoritma (HGA)	58
2 YOL PLANLAMA	60
2.1 Yol Planlama Algoritmalarının Sınıflandırılması.....	62
2.1.1 Global (Küresel – Çevrimdışı) Yol planlama.....	64
2.1.2 Yerel (Bölgesel – Çevrimiçi) Yol Planlama.....	64
2.2 Yol Planlama Teknikleri	65
2.2.1 Klasik teknikler	65
2.2.2 Olasılıksal Teknikler	65
2.2.3 Sezgisel Teknikler	65
3 DENEYSEL ÇALIŞMALAR VE TARTIŞMA	67
3.1 CEC 2017 Sonuçları ve Diğer Metasezgizeller ile Karşılaştırılması	67
3.1.1 CEC2017 Fonksiyon Rank Karşılaştırma	78
3.2 CEC2022 Sonuçları ve Diğer Metasezgizeller ile Karşılaştırılması	86

3.2.1 CEC2022 Fonksiyon Rank Karşılaştırma	92
3.3 UAV ile Deneysel Çalışmalar	96
3.3.1 Ele Alınan Senaryolar ve Simülasyon Parametreleri	98
3.3.2 Algoritmaların Karşılaştırıldığı UAV Yol Planlama Sonuçları	98
3.3.2.1 GA Referans Alınarak Karşılaştırılan Algoritmalar ve Sonuçları	99
3.3.2.2 GWO ve SCSO Referans Alınarak Karşılaştırılan Algoritmalar ve Sonuçları	102
4 SONUÇ VE ÖNERİLER	105
KAYNAKLAR	106
İNTİHAL RAPORU İLK SAYFASI	115



KISALTMA DİZİNİ

ABC	Yapay Arı Koloni Algoritması (Artificial Bee Colony Algorithm)
ACO	Karınca Kolonisi Algoritması (Ant Colony Optimization Algorithm)
ASFSSA	Uyarlamalı Spiral Uçuşlu Serçe Arama Algoritması (Adaptive Spiral Flying Sparrow Search Algorithm)
BA	Yarasa Optimizasyonu (Bat Algorithm)
BA_ABC	Ba Ve Abc İle Oluşturulan Hibrit Algoritma
BKA	Kara Kanatlı Yırtıcı Kuşu (Sungur, Doğan Vb.) Algoritması (Black Winged Kite Algorithm)
CHOA	Şempanze Optimizasyonu Algoritması (Chimp Optimization Algorithm)
COA	Coati Optimizasyon Algoritması
DBO	Gübre Böceği Optimizasyon Algoritması (Dung Beetle Optimizer)
DE	Farksal Gelişim Algoritması (Differential Evolution)
DMOA	Cüce Mangus Arama Algoritması (Dwarf Mongoose Optimization Algorithm)
GA	Genetik Algoritması (Genetic Algorithm)
GJO	Altın Çakal Optimizasyonu Algoritması (Golden Jackal Optimization Algorithm)
GSA	Yerçekimsel Arama Algoritması (A Gravitational Search Algorithm)
GWO	Gri Kurt Optimizasyonu (Grey Wolf Optimizer)
HGA	Hibrit Genetik Algoritma

HLOA	Dikenli Kertenkele Optimizasyon Algoritması (Horned Lizard Optimization Algorithm)
IGWO	Geliştirilmiş Gri Kurt Optimizasyonu (Improved Grey Wolf Optimization)
MBADE	Yarasa Optimizasyonu (Ba) Ve Farksal Gelişim Algoritması (Differential Evolution-De) İle Oluşturulan Hibrit Algoritma
MSWOA	Çoklu Stratejili Balina Optimizasyon Algoritması (Multi-Strategy Whale Optimization Algorithm)
NCHHO	Doğrusal Olmayan Tabanlı Kaotik Harris Şahinleri Optimizasyon Algoritması (Nonlinear Chaotic Harris Hawks Optimization Algorithm)
NMS	Nelder – Mead Simplex
OMA	Optik Mikroskop Algoritması (Optical Microscope Algorithm)
PMOMA	Popülasyon Tabanlı Çok Amaçlı Güve Algoritması (Population Based Multi Objective Moth Algorithm)
POA	Pelikan Optimizasyonu Algoritması (Pelican Optimization Algorithm)
PSO	Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)
SCA	Sinüs Kosinüs Algoritması (Sine Cosine Algorithm)
SCSO	Kum Kedisi Sürüsü Algoritması (Sand Cat Swarm Optimization)
WaOA	Walrus Optimizasyon Algoritması
WOA	Balina Optimizasyon Algoritması (Whale Optimization Algorithm).
ZHO	Geliştirilmiş Hibrit Zebra Algoritması (Zebra Hunt Optimization)
ZOA	Zebra Optimizasyon Algoritması (Zebra Optimization Algorithm)
UAV	Unmanned Aerial Vehicle (İnsansız Hava Aracı-İHA)

ŞEKİL LİSTESİ

	<u>Sayfa No</u>
Şekil 1.1 Optimizasyon problemlerinin sınıflandırılması.....	2
Şekil 1.2 Optimizasyon algoritmalarının gruplandırılması.....	3
Şekil 1.3 Sezgisel yöntemlerin sınıflandırılması.....	9
Şekil 1.4 Kambur balının avlanma yöntemi	10
Şekil 1.5 Spiral hareket ve WOA’ nın global arama davranışı	10
Şekil 1.6 Yapay arı koloni algoritması	14
Şekil 1.7 Besin kaynağını tarif eden arıların dansı	15
Şekil 1.8 Besin kaynağına ulaşmaya çalışan karıncaların en kısa yol keşif aşamaları	16
Şekil 1.9 Gri kurt hiyerarşisi	17
Şekil 1.10 Gri kurtların avlanma stratejileri ve ava göre güncellenen konumları	18
Şekil 1.11 Parçacıkların Arama Uzayındaki Hareketleri	23
Şekil 1.12 PSO akış diyagramı	24
Şekil 1.13 Genetik Algoritma akış diyagramı	30
Şekil 1.14 Dijkstra algoritmasının anlatımı için kullanılan çizge	45
Şekil 1.15 Dijkstra algoritması 1. adım görseli	46
Şekil 1.16 Dijkstra algoritması 2. adım görseli	46
Şekil 1.17 Dijkstra algoritması 3. adım görseli	46
Şekil 1.18 Dijkstra algoritması 4. adım görseli	47
Şekil 1.19 Dijkstra algoritması 5. adım görseli	47
Şekil 1.20 Dijkstra algoritması 6. adım görseli	47
Şekil 1.21 Dijkstra algoritması 7. adım görseli	48

Şekil 1.22 Dijkstra algoritması 8. adım görseli	48
Şekil 1.23 Dijkstra algoritması 9. adım görseli	48
Şekil 1.24 Dijkstra algoritması 10. adım görseli	49
Şekil 1.25 Genetik-Simpleks hibrit algoritması akış diyagramı	52
Şekil 2.1 Yol planlama problemine ait kriterler	61
Şekil 2.2 Yol planlama algoritmalarının sınıflandırılması	62
Şekil 2.3 Arama yapılacak çevrenin durumuna göre algoritmaların sınıflandırılması	63
Şekil 2.4 Yol planlama teknikleri	66
Şekil 3.1 CEC2017 test fonksiyonlarda karşılaştırma yapılan yöntemlerin yakınsama performansı	78
Şekil 3.2 CEC2022 test fonksiyonlarda karşılaştırma yapılan yöntemlerin yakınsama performansı	91
Şekil 3.3 GA-HLOA-MSWOA-ASFSSA-GWO-COA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	99
Şekil 3.4 GA-SCSO-DBO-NCHHO-PSO karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	100
Şekil 3.5 GA-BKA-OMA-PMOMA-ZOA-ZHO-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	101
Şekil 3.6 GWO-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	102
Şekil 3.7 GWO-SCSO-GA-PSO-PMOMA-ZOA-ZHO-COA-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	103
Şekil 3.8 GA-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA-PSO-PMOMA-GWO-ZOA-ZHO-COA-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri.....	104

TABLO LİSTESİ

	<u>Sayfa No</u>
Tablo 1.1 Dijkstra algoritması sonucu	49
Tablo 3.1 CEC2017 fonksiyonlarının sonuç değerleri.....	67
Tablo 3.2 CEC2017 fonksiyon rank analiz değerleri.....	78
Tablo 3.3 CEC2017 fonksiyon toplam rank analiz değerleri.....	85
Tablo 3.4 CEC2022 fonksiyonlarının sonuç değerleri.....	87
Tablo 3.5 CEC2022 fonksiyon rank analiz değerleri.....	92
Tablo 3.6 CEC2022 fonksiyon toplam rank analiz değerleri.....	95
Tablo 3.7 Senaryo 1 ve Senaryo 2 için parametre değerleri.....	98

Doğadan Esinlenmiş Hibrit Bir Optimizasyon Algoritmasının Yol Planlama Problemine Uygulanmasının İstatistiksel Sonuçları

Merve GENÇ

UAV' ler için yol planlama, bir başlangıç noktasından hedefine giderken engellerden kaçınma, enerji-zamana bağlı maliyeti minimize etme ve dinamik çevrede güvenli uçuş sağlama gibi çoklu hedefleri içeren karmaşık bir optimizasyon problemidir. Bu problem genelde mesafe, zaman, enerji, güvenlik gibi çok boyutlu yaklaşımlara ihtiyaç duyar. Hareketsiz veya hareketli ortamlarda ve $2D/3D$ uzaylarda modellenenebilir. Matematiksel model genellikle lineer olmayan yükseklik, açı, hız, çarpışma riski gibi faktörler içeren birkaç kısıta bağlıdır. Bu tip problemler, klasik A^* , Dijkstra gibi deterministik arama yöntemleri ile çözülmeye çalışılsa da, ortamın dinamik doğası ve optimalite gerekliliği nedeniyle bu yöntemler genellikle alt-optimum çözümlere takılır. Bu nedenle Meta-sezgisel optimizasyon algoritmalar türev gerektirmeyen, doğadan esinlenmiş, popülasyon tabanlı veya tek çözüm tabanlı heuristiklerdir. Arama uzayını genel keşif + yerel sömürü stratejisi ile tarar. Global optimuma yaklaşmak ve yerel çözümlerde takılmamak için uygundur. Bu özellikleri nedeniyle UAV yol planlama gibi yüksek boyutlu, çok kriterli ve dinamik problemlerde meta-sezgisel algoritmalar özellikle tercih edilir. Bu çalışmada literatürde en çok çalışılan ve son zamanlarda tanımlanan metaheuristik optimizasyon algoritmaları ele alınarak yol planlama probleminde kullanılmıştır. Daha önce UAV yol planlamada kullanılmayan ZHO, ZOA, WHOA ve COA algoritmaları bu problemi çözmek için uyarlanmış, buldukları sonuçlar istatistiksel olarak analiz edilmiştir. Bu sonuçlar tablo ve grafikler ile görselleştirilerek avantajları açık bir şekilde gösterilmiştir.

Anahtar Kelimeler: Yol planlama, optimizasyon, metaheuristik, ZHO, ZOA, WHOA, COA.

Statistical Results of the Application of a Nature-Inspired Hybrid Optimization Algorithm to the Path Planning Problem

Merve Genç

UAV path planning is a complex optimization problem involving multiple objectives, such as avoiding obstacles while traveling from a starting point to a destination, minimizing energy-time-dependent costs, and ensuring safe flight in a dynamic environment. This problem generally requires multidimensional approaches involving distance, time, energy, and safety. It can be modeled in stationary or moving environments and in $2D/3D$ spaces. The mathematical model is usually dependent on several constraints, including nonlinear factors such as altitude, angle, velocity, and collision risk. Although these types of problems are attempted to be solved with classical deterministic search methods such as A^* and Dijkstra, these methods often get stuck at sub-optimum solutions due to the dynamic nature of the environment and the need for optimality. Therefore, meta-heuristic optimization algorithms are derivative-free, nature-inspired, population-based, or single-solution-based heuristics. They scan the search space with a general exploration + local exploitation strategy. They are suitable for approaching the global optimum and avoiding getting stuck at local solutions. Because of these characteristics, meta-heuristic algorithms are particularly preferred in high-dimensional, multi-criteria, and dynamic problems such as UAV path planning. This study examines the most studied and recently defined metaheuristic optimization algorithms in the literature and applies them to the route planning problem. The ZHO, ZOA, WHOA, and COA algorithms, previously unused in UAV route planning, were adapted to solve this problem, and their results were statistically analyzed. These results are visualized in tables and graphs to clearly demonstrate their advantages.

Keywords: Path planning, optimization, meta-heuristic, ZHO, ZOA, WHOA, COA.

Son yıllarda karşımıza çıkan problemlerin, değişkenlerinin artması ile özellikle zaman ve maliyet değişkenlerinin önem kazanması, problem sahasını genişletmekte olup, kısa bir zaman diliminde çözüme ulaşmayı zorlaştırmıştır. Bu durum problemin çözümünü en az işlem gücüyle gerçekleştirmeye yönlendirmiş, en az maliyetle en yakın sonucu elde etmek için alternatif yollar aramaya sevk etmiştir. Bu alternatif yolların aranması, optimizasyonun keşfedilmesi ve geliştirilmesinde önemli rol oynamıştır.

Yaşamımız boyunca karşılaştığımız problemler hayatımıza bir engel teşkil ettiği için, yaşam koşullarımızı kolaylaştırmak adına ya engeli ortadan kaldırmaya ya da yarattığı problemlere çözüm arayarak en sorunsuz ve ideal bir şekilde sonuca ulaşmaya çalışırız. Elimizde olan sınırlı kaynaklarla elde etmeye çalıştığımız ideal sonuca ulaşma amacı, bizleri ideal olan çözümü bulmaya yönlendirmiştir. Özünde her canlıda görülen bu davranış sadece insanlarla sınırlı olmayıp, bulunduğu ortamın imkanları, sınırları ve kısıtları çerçevesinde en uygun çözüme ulaşma isteği belirli işlemler gerektirir ve bu işlemleri kapsayan tanıma optimizasyon denir. Günlük hayatta bir yerden başka bir yere ulaşım sağlarken en kısa sürede varış noktasına ulaşma, sınırlı parayla en kaliteli ürünü en uygun fiyata satın alma, en az maliyette en etkili sanayi ve sağlık teknolojilerini üretme ve uygulama gibi karşımıza çıkan sorunlar ve çözüm aşamaları, günlük hayatla sınırlandırılmayacak kadar geniş kapsamı olan, finans, mühendislik ve matematik gibi geniş alanları kapsayan optimizasyon problemlerinden sadece birkaçıdır.

1.1 Optimizasyon

İnsanlar problemler karşısında çözüm arayışına girmesi ve izlenen yollara karar vermesi aşamasında her ne kadar hızlı davranırsa da, karmaşık olan matematiksel problemleri ve modelleri çözemeyebilir veya hatalı çözümler elde edebilir. Mümkün çözümler içerisinde insanlar, problemlerine en uygun çözümü ararken doğadaki gözlemledikleri canlıların davranışlarından yola çıkarak çeşitli sezgisel algoritmalar geliştirmişlerdir. Bu sezgisel

algoritmalar çeşitli tekniklerle bilgisayar ortamında modellenerek, karmaşık matematiksel problemleri, daha kısa sürede karar vererek hatasız çözüme ulaştırmayı başarmıştır. Her yeni problem başka bir sezgisel tekniğin keşfedilmesine yol açarak, her tekniğin matematiksel olarak bilgisayar ortamına uyarlanması ile oluşan algoritmalar optimizasyonun gelişmesine ayrıca katkı sağlamıştır.

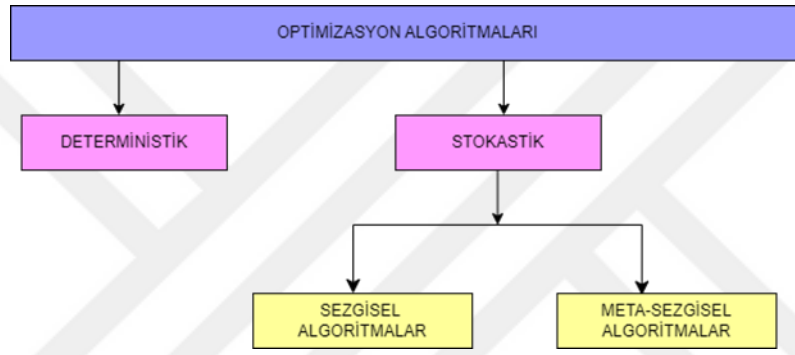


Şekil 1.1 Optimizasyon problemlerinin sınıflandırılması

Optimizasyonun her problemin karakterine uygun bir şekilde uyarlanması için çeşitli algoritmalar geliştirilir. Algoritmalar problemi çözmek için adım adım takip edilmesi gereken etkili komutlar kümesinin tamamıdır. Her algoritmanın bünyesinde aritmetik adımları bulundurması ve matematiksel çözüm isteyen karmaşık bütün problemlerin bir algoritma ile ifade edilmesi ve algoritmaların çalışma sürelerinin diğer çözüm yöntemlerinden daha kısa sürede çözmeye elverişli olması, algoritmaların problem çözümünde ve optimizasyonun tasarlanmasında, matematikteki yeri ve önemini açısından önemli rol oynar. Problemin türüne göre optimizasyon çeşitleri Şekil 1.1 ' de verilmiştir.

Optimizasyon, en iyileme anlamına gelmektedir. Bir problem için, verilen şartlar altında tüm çözümler arasından en iyi çözümü elde etme işidir. Belirli sınırlamaları sağlayacak şekilde, bilinmeyen parametre değerlerinin bulunmasını içeren herhangi bir problem, optimizasyon problemi olarak adlandırılabilir (Murty, K. G., 2003). Teknolojinin de gelişmesiyle birlikte endüstrileşmenin arttığı ve robotların hem sosyal hem de endüstriyel alanlardaki kullanımında önemli bir artış yaşanması optimizasyonun, günümüzde her duruma uyarlanması açısından önem kazanmıştır. Özellikle endüstriyel robotlar, üretim ve imalat süreçlerinde yüksek verimlilik sağlaması, ulaşım açısından avantaj oluşturmaları ve doğada karşılaşılan problemleri çözümlerken zaman ve maliyeti

azaltması ise optimizasyonun gerekliliğini gösteren diğer durumlardır. Tüm bu ihtiyaçlardan yola çıkarak optimizasyon, gün geçtikçe geliştirilerek ve her yeni duruma uyarlanarak en kısa süre ve en az maliyetle, hedeflenen amaç doğrultusunda problemlerin çözümünü elde etmiş, yaklaşık çözümler içerisinde en iyi sonuca ulaşmış, bilim ve teknolojinin ilerlemesine, sürekli kendini yenileyen bir döngü şeklinde katkı sağlamıştır. İşletmelerde yüksek kâr elde etme veya maliyeti en aza indirme, mühendisliklerde ihtiyaca cevap veren en uygun boyutlarda cihaz üretimi, sağlıkta işlevini kaybetmiş organ ve hareket sisteminde hasar gören uzuvlara yardımcı, en az enerji kullanan cihazların tasarımı veya endüstride üretilen bir parçanın çevreye verdiği zararın en aza indirgenmesi birer optimizasyon işlemidir.



Şekil 1.2 Optimizasyon algoritmalarının gruplandırılması

Bazı problemlerin çözümünde anında çözüm sunacak algoritmalar bulunamayabilir, bu tür problemler ise optimizasyon problemleridir. Optimizasyon problemlerinin iki türlü çözüm yöntemi deterministik ve stokastik yöntemlerdir. Deterministik algoritmalar aynı parametreleri kullanarak aynı sonuca ulaşır fakat stokastik algoritmalar rasgele oldukları için aynı parametre ile çalıştırılırsalar dahi her seferinde farklı sonuçlar üretebilir. Deterministik optimizasyon, tüm parametrelerin ve değişkenlerin kesin olarak bilindiği, belirsizlik içermeyen optimizasyon problemlerini ifade eder. Bu tür optimizasyon problemlerinde, amaç fonksiyonu ve kısıtlamaların hepsi deterministik (yani rastgele olmayan, sabit) verilerle tanımlanır. Deterministik optimizasyonda, optimize edilecek olan fonksiyon, tüm verilerin belirli olduğu durumlarda çözülür. Kısıtlamalar da belirli ve değişmezdir. Bu yöntemler, belirli matematiksel algoritmalar kullanarak en iyi çözümü bulmayı hedefler. Deterministik optimizasyon ile çözülebilecek problemlere, bir fabrikanın maliyetleri minimum tutarak hangi ürünlerin ne kadar miktar veya sayıda üretilmesi gerektiğini, tarımda sulama yöntemi ile sınırlı kaynak olan suyun dengeli kullanımı, enerji tasarrufu sağlanarak en kısa ve en ucuz rota bulma problemi, yatırım

yapılan alanlarda hisse senetlerinde riski minimize edip maksimum getiriye elde etme problemlerinin hepsinde veriler ve parametreler bilindiği için deterministik kategorisine girerek, bu kategorideki optimizasyon problemleri genellikle lineer programlama, doğrusal olmayan programlama, tamsayılı programlama gibi yöntemlerle çözülür.

Deterministik optimizasyon, karar vermede belirsizlik olmadığında kullanışlıdır, ancak gerçek dünya senaryolarında nadiren tüm parametreler kesin olarak bilinir. Deterministik optimizasyon yöntemleri, türevsel hesap işlemleri içerirken, stokastik optimizasyon algoritmaları ise istatistiksel yöntemlerle rasgele sayıları kullanarak en yakın çözüme ulaşır. Bir optimizasyon probleminin çözümünde deterministik yöntemler yetersiz kalabilirken, stokastik yöntemler kullanılarak kesin sonuca ulaşılamazsa bile en yakın sonucu garanti eder. Bu durumlarda, stokastik optimizasyon gibi yöntemler dikkate alınabilir.

Stokastik optimizasyon, belirsizlik altında optimizasyon problemlerini ele alır. Bu tür problemlerde, bazı parametreler veya değişkenler rastgele (stokastik) doğaya sahiptir, yani tam olarak bilinmezler ancak olasılık dağılımları ile tanımlanabilirler. Stokastik optimizasyonda, parametreler veya kısıtlamalar kesin değil, olasılık dağılımlarıyla ifade edilir. Belirsizlik kısıtlamalarda da yer alabilir. Kısıtlamaların sağlanma olasılıkları göz önünde bulundurulur. Genellikle beklenen değer veya başka bir risk ölçütü kullanılarak optimize edilir. Belirsizliği senaryolarla modelleyip, her senaryo için optimize eder. Senaryo ağaçları kullanarak olası sonuçları ve bunların olasılıklarını değerlendirir.

Stokastik optimizasyon, gerçek dünya problemlerinde belirsizlikleri yönetmek için güçlü bir araçtır, ancak çözüm süreçleri genellikle deterministik yöntemlere göre daha karmaşık ve hesaplama açısından daha maliyetli olabilir. Stokastik optimizasyon yöntemiyle çözülen problemlerden bazıları, hastalar için tedavi stratejileri belirlenirken, tedavi sonuçlarının ve hastalığın ilerleyişinin belirsizlikleri göz önünde bulundurulur. Örneğin, kanser tedavisinde kullanılacak ilaç dozajlarının ve tedavi türlerinin optimizasyonu stokastik yöntemlerle yapılabilir. Rüzgar veya güneş enerjisi gibi yenilenebilir enerji kaynaklarının üretimi belirsizdir. Bu kaynaklardan enerji elde ederken, enerji talebini karşılamak için optimize edilmiş bir üretim ve dağıtım planı stokastik optimizasyonla çözülebilir. Nakliye rotalarının ve araç kullanımının optimize edilmesi, trafik koşulları, hava durumu ve diğer rastgele faktörler nedeniyle stokastik modeller kullanılarak yapılabilir. Yatırım portföylerinin optimize edilmesi, gelecekteki piyasa hareketlerinin belirsizliği nedeniyle hangi hisse senetlerine, tahvillere veya diğer yatırım araçlarına

yatırım yapılacağını belirlemek için stokastik optimizasyon yöntemleri uygulanabilir. Fabrikalarda üretim miktarlarının, stok seviyelerinin ve üretim planlamasının belirlenmesi, talep ve tedarik zinciri belirsizlikleri nedeniyle stokastik modeller gerektirir. Bu, stok maliyetlerini minimize ederken müşteri taleplerini karşılamayı amaçlar.

Stokastik algoritmalar kendi alanında ikiye ayrılır. Stokastik algoritmalarından ilki olan sezgisel algoritma, problemin yapısına uygun yolları takip ederek kısa sürede yaklaşık çözüm üretirler. İkincisi olan metasezgisel algoritmalar ise genel bir problemin yapısına uygun olmayıp, çözüme varmak için rasgele seçimlerden, keşfetme ve sömürme gibi sezgisel adımlarından yardım alarak en yakın ve ideal sonuca varmayı hedefler. Sezgisel algoritmalar sezgiyi referans alarak en yakın, en iyi, en kolay ve en ucuz şekilde çözüme ulaşmayı hedeflerken, metasezgisel algoritmalar ise sezgi ile birlikte deneyimlerden yola çıkarak daha önce tecrübe edilen yolları ve bilgi birikimini kullanarak en gelişmiş haliyle çözüme ulaşmayı hedefler.

Sezgisel Optimizasyon (Heuristic Optimization), karmaşık optimizasyon problemlerini çözmek için pratik, hızlı ve genellikle yaklaşık çözümler sunan yöntemlerdir. Sezgiseller, tam olarak optimal çözümü garanti etmese de, genellikle makul bir sürede iyi bir çözüm sağlar. Belirli bir problem sınıfı için tasarlanmış, hızlı ve pratik çözümler üretmeyi amaçlayan yöntemlerdir. Sezgilere dayanır ve genellikle optimum çözümü garanti etmez ama iyi bir yaklaşık çözüm sağlar. Sezgisel optimizasyon, özellikle zaman kısıtlamaları altında veya hesaplama gücü sınırlı olduğunda tercih edilir. Ancak, çözümün kalitesi problem yapısına, sezgiselin tasarımına ve uygulama detaylarına bağlı olarak değişebilir. İşte sezgisel optimizasyonun temel özellikleri, probleme özgü olmasıdır dolayısıyla sezgiseller, belirli bir problem sınıfı için tasarlanır. Problem özeldir; yani, belirli bir problem tipi için optimize edilmiştir. Bu, onları o problem türü için etkili yapar, ancak başka problemlere doğrudan uygulanamaz. Uygulanabilirliği açısından hızlı ve pratiktirler. Sezgisel yöntemler, genellikle daha basit ve daha az hesaplama gücü gerektirir. Diğer tam arama veya kesin yöntemlerden daha hızlı çözümler üretir. Çoğu sezgisel yöntem, global optimumu bulmayı garanti etmez. Bunun yerine, iyi bir yaklaşık çözüm veya "yeterince iyi" bir çözüm bulmayı hedefler. Sezgiseller, karmaşık matematiksel teknikler yerine basit kurallara dayanır. Bu, onları uygulamada daha az karmaşık ve daha kolay anlaşılır yapar.

Sezgisel optimizasyonlar, karmaşık ve genellikle çözümü zor olan problemlerde yaygın olarak kullanılır. Sık kullanım alanlarından bazıları, en kısa veya en verimli yol rotasını

bulmak için sezgisel algoritmalar kullanılır. Üretim hatlarında verimliliği artırmak, stok yönetimi, tedarik zinciri optimizasyonu gibi alanlarda maliyetleri düşürmeye ve süreçleri hızlandırmaya yardımcı olur. Risk yönetimi, yatırım stratejilerinin belirlenmesi gibi finansal kararlar için sezgisel yöntemler kullanılır. Ağ tasarımı, bant genişliği, veri iletim yollarının optimize edilmesi gibi konularda da sezgisel optimizasyondan faydalanılır. Hastane personel planlama, hasta randevu sistemlerinin optimizasyonu gibi alanlarda, elektrik şebekelerinde enerji dağıtımının optimize edilmesi, enerji kaynaklarının verimli kullanımında da yararlanır. Oyunların stratejilerini optimize etmek veya karmaşık sistemlerin davranışını simüle etmek için yine sezgisel optimizasyona başvurulur. Bu yüzden sezgisel optimizasyonlar, tam çözüm bulmanın pratik olmadığı durumlarda iyi yaklaşık çözümler sunar ve birçok alanda vazgeçilmez bir araç haline gelmiştir.

Gerçek hayat problemlerini çözmede klasik yöntemler, çözüm sürelerinin uzun zaman alması ve işlem yükünün fazla olması problemin boyutunu arttırdığı için yetersiz kalmaktadır. Bu durum sezgisel araştırma yöntemlerinin keşfedilmesine yol açmış olup, bu sezgisel yöntemleri matematiksel ifadelerle dönüştürmeye ve yeni algoritmaların keşfiyle birlikte hedeflenen en iyi çözüme ulaşmayı amaçlamıştır. Doğadaki tüm canlıların ekosistem içerisinde gerçekleştirdikleri barınma, savunma, avlanma ve yer değiştirme gibi çeşitli eylemlerde bulunarak en kısa sürede optimum çözümü bulmaları için sergiledikleri tüm davranışlar, dikkat çekmeyi başararak araştırma konusu olmuş ve doğadan esinlenerek metasezgisel algoritmaların doğmasına ve tasarlanmasına neden olmuştur.

1.2 Metasezgisel Algoritmalar ve Optimizasyon

Meta-sezgisel optimizasyon algoritmaları doğadaki canlıların beslenme, hareket etme ve birlikte yaşama gibi olaylarından esinlenerek, yani doğayı taklit ederek en iyi sonuca ulaşmaya çalışmaktadırlar. Meta-sezgisel optimizasyon yöntemleri taklit ettikleri doğa olayına göre gruplandırılmaktadır. Işık, fizik, biyoloji, sürü, müzik, spor, kimya, bitki tabanlı çeşitleri mevcuttur (Bingöl ve Yıldırım, 2021).

Metasezgisel Optimizasyon (Metaheuristic Optimization) yöntemleri, karmaşık optimizasyon problemlerini çözmek için daha genel, adaptif ve esnek stratejiler sunar. Metasezgiseller, birçok farklı problem türünde kullanılabilir ve hem yerel optimumları aşarak daha iyi çözümler aramayı hem de geniş arama alanlarını keşfetmeyi hedefler. Metasezgisel optimizasyon, karmaşık ve çok boyutlu problemlerde, özellikle de kesin

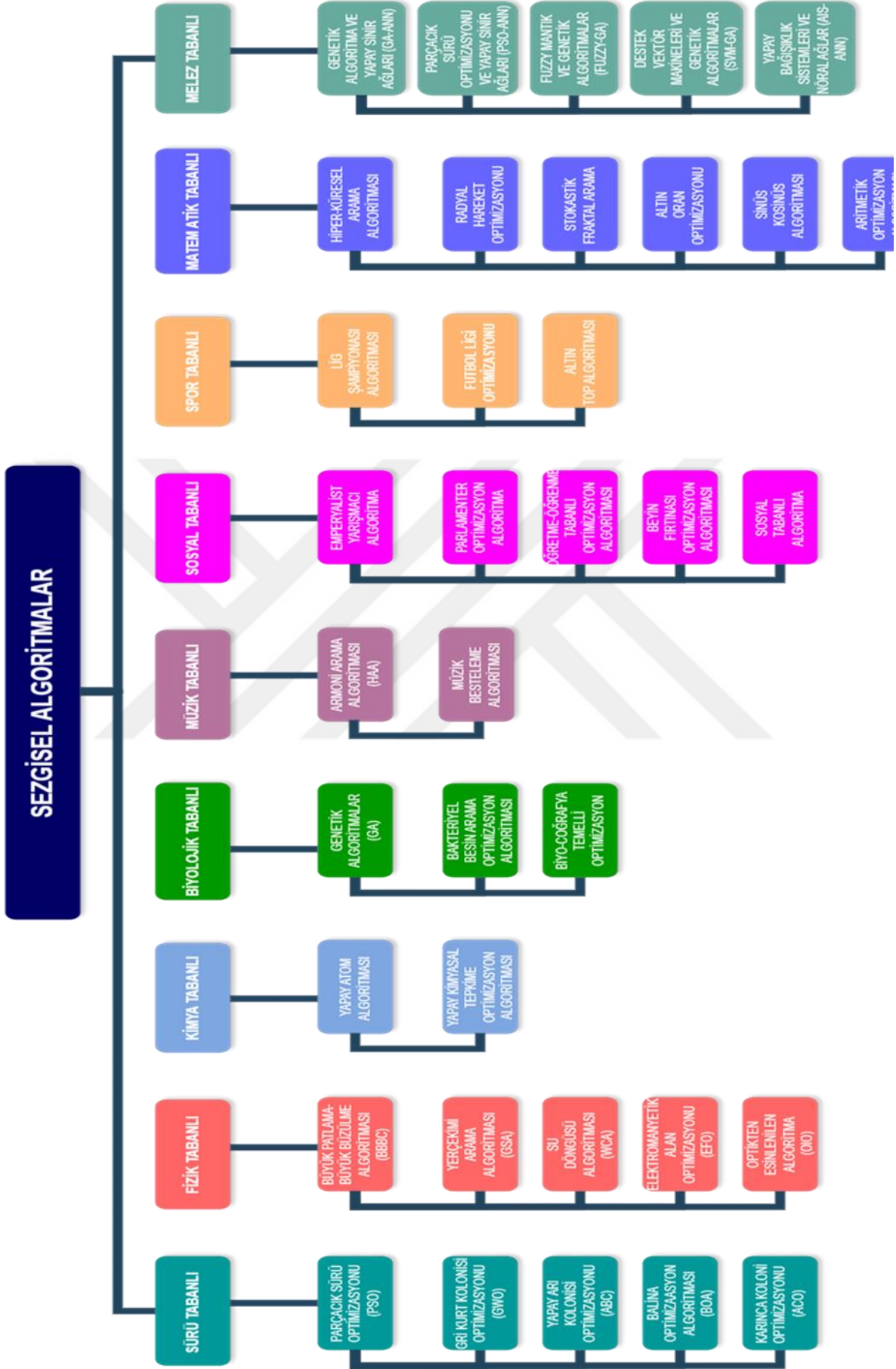
özüm yöntemlerinin uygulanması zor veya mümkün olmadığında, oldukça etkili olabilir. Ancak, bu yöntemler genellikle daha fazla hesaplama gücü ve zaman gerektirir. İşte metasezgisellerin temel özellikleri, metasezgiseller belirli bir probleme özgü değildir. Çeşitli optimizasyon problemlerine uyarlanabilirler. Problemden bağımsız, yani birçok farklı problem türüne uyarlanabilirler. Problemin doğası ve yapısı değişse bile, metasezgisel yöntemler uyum sağlayabilir ve hala etkili çözümler üretebilir. Genellikle daha karmaşık ve hesaplama açısından daha maliyetli olabilir, ancak daha iyi çözümler bulma potansiyeline sahiptir. Yerel optimumlara sıkışma riskini azaltmak için çeşitli yöntemler kullanılır. Bu, daha geniş arama alanlarını keşfetmeyi ve potansiyel olarak global optimuma yaklaşmayı sağlar. Hem yerel optimumları aşarak global optimumu aramaya çalışır hem de bu süreçte çeşitli "kaçış mekanizmaları" kullanır. Genellikle bir popülasyonun (çözümler kümesi) veya tek bir çözümün iteratif olarak iyileştirilmesine dayanır.

Uygulama alanları sezgisel algoritmalarla aynı olmakla birlikte alanları sırasıyla makine öğrenimi model optimizasyon ve seçimi, en kısa yol bulma veya ağ tasarımı, finans alanındaki risk yönetimi, iş çizelgelemesi, ders programı oluşturma, üretim planlaması gibi zaman ve kaynak kısıtlamaları olan problemler için kullanılır. Rotalama ve lojistikte depo yerleşimi ve tedarik zinciri yönetimi gibi lojistik ve taşımacılık alanlarında etkilidir. Proje yönetimi, kaynak planlama zaman odaklı optimizasyon problemleri için uygulanır. Mühendislik alanlarında devre tasarımı, kontrol sistemleri, telekomünikasyon ve ağ optimizasyonu, enerji dağıtım şebekelerinin optimizasyonu, yapay zeka alanında sinir ağlarının eğitimi, parametre ayarlamaları, özellik seçimi, kümeleme algoritmaları başta olmak üzere, sağlık sistemleri, personel çizelgeleme ve hastane kaynak ve hasta yönetimi gibi alanlarda metasezgisel optimizasyon kullanılır.

Sonuç olarak stokastik yöntemleri ele alırsak; Sezgisel (Heuristic) ve Metasezgisel (Metaheuristic) yöntemler, karmaşık optimizasyon problemlerini çözmek için kullanılan stratejilerdir, ancak aralarında bazı temel farklar bulunur. Sezgiseller belirli bir probleme uyarlanmışken, metasezgiseller daha genel ve adaptif çözümler sunar. Metasezgiseller genellikle daha karmaşık algoritmalar içerir ve daha fazla hesaplama gücü gerektirir. Sezgisel yöntemler hızlı iyi çözümler sunarken, metasezgiseller daha kapsamlı bir arama yaparak potansiyel olarak daha iyi sonuçlara ulaşabilir. Bu farklar, bir optimizasyon problemi çözerken hangi yöntemin kullanılacağına karar vermekte önemli rol oynar. Doğanın diğer parçası olan insanlar ise karşılaştığı problemleri çözerken göz önünde

bulundurduğu iki kavram vardır. Biri kısa zamanda en yakın çözüme ulaşmak, diğeri ise zaman kavramını göz ardı ederek her koşulda kesin sonuca varmak. İlk durum sezgisel olarak var olan çözümler içerisinde en iyiyi seçerken, ikinci durum ise kesin sonuca ulaşmak için uzun zaman gerektiren uğraş ve işlemler sonucunda maliyeti artırabilir. Meta-sezgisel algoritmalar, kesin çözümlerin maliyetli olduğu durumlarda kullanılır ve yaklaşık çözümlerin içinden en iyisi seçer. Bu yüzden meta-sezgisel algoritmaların gerçek hayat problemlerinin çözümünde uygulanabilir olması genellikle avantaj sağlar fakat her probleme aynı meta-sezgisel algoritmalar uygulanabilirlik açısından uygun olmayabilir. Bu yüzden problem için en uygun meta-sezgisel algoritmayı seçmek çok önemlidir. Problemin çözümüne uygun meta-sezgisel algoritmanın olmayışı, yeni algoritma arayışına sevk edebilir. Sonucunda ise bir sürü meta-sezgisel algoritma bulunmuş ve kendi içinde sınıflandırılmıştır.

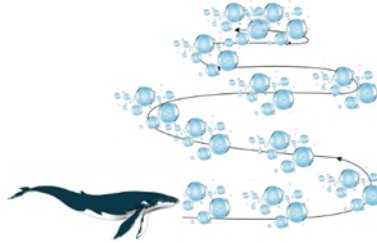
Meta-sezgisel algoritmalar, fiziksel temelli algoritmalar, evrimsel algoritmalar, sürü zekası algoritmaları, biyocoğrafik ilhamlı algoritmalar (Behesti ve Shamsuddin, 2013) ve doğadan ilham alan diğer algoritmalar olarak sınıflandırılabilirler (Blum ve Roli, 2010). Meta-sezgisel algoritmaları doğada gözlemlediğimizde, en göze çarpan özellikleri birlikte iş paylaşımı yaparak, en uygun çözümü bulma, bulunan çözüme en yakın çözümleri ve alternatif potansiyelleri sömürme şeklinde hareket eder. Zor ve karmaşık problemleri çözümlmek için geliştirilen sürü tabanlı algoritmalar başta olmak üzere çeşitli gruplara ayrılmış sezgisel yöntemler Şekil 1.3' te gösterilmiştir.



Sekil 1.3. Sezgisel yöntemlerin sınıflandırılması

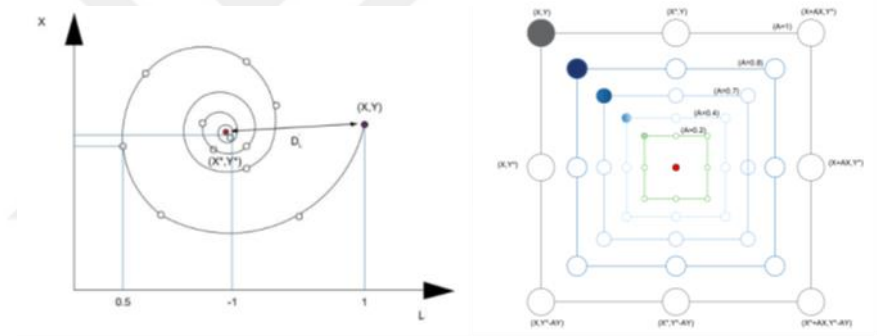
1.2.1 Balina Optimizasyon Algoritması (Whale Optimization Algorithm - WOA)

Kambur balina optimizasyon algoritması (WOA), Mirjalili ve Lewis 2016, tarafından geliştirilmiştir. Sürü zekası tabanlı algoritma olup, kambur balina türünün avlanırken sergilediği davranışları gözlemlenerek düzenlenmiş sezgisel algoritmadır.



Şekil 1.4 Kambur balinanın avlanma yöntemi

Avlanırken balinalar hava kabarcığı yöntemiyle balık sürüsünü yönlendirmeye çalışmakta ve avlama olayını kolaylaştırmaktadırlar (Mirjalili ve Lewis, 2016).



Şekil 1.5 Spiral hareket ve WOA' nın global arama davranışı (Mirjalili ve Lewis, 2016)

Balina önce avın yerini belirler ardından yüksek ses çıkartır ve hedefindeki avın korkmasını sağlar. Hemen sonrasında ise suyun altında nefesini sürekli verip hava kabarcıkları oluşturup, avının kafasını karıştırarak yönünü kaybetmesine sebep olur ve ardından balina daralan dairesel hareketlerle avını tuzağa düşürürerek kısa sürede amacına ulaşmış olur. Bu davranış yöntemi Şekil 4'te görselleştirilmiştir.

WOA, kambur balinaların avlarını baloncuk ağı yöntemiyle kuşatma davranışını matematiksel olarak modellemektedir. Algoritma üç temel aşamadan oluşur: Avı kuşatma (Encircling prey), Baloncuk ağı saldırısı (Bubble-net attacking), Rastgele arama (Exploration).

Her balina, d -boyutlu bir çözüm vektörü ile temsil edilir. $\vec{B}(t) = |b_1(t), b_2(t), \dots, b_d(t)|$ Popülasyondaki en iyi çözüm (av konumu) $\vec{B}^*(t)$ 'dir. Balinalar avın yerini tahmin eder

ve onu çevreler. Optimizasyon sürecinde en iyi çözüm $\vec{B}^*(t)$ olarak kabul edilir. $\vec{\Delta}$, mevcut çözüm ile en iyi çözüm arasındaki mutlak farkı gösteren mesafe vektörüdür. Mesafe vektörü, balinanın en iyi çözüme göre ne kadar uzakta olduğunu belirler ve konum güncellemede kullanılır. $\vec{Y}$, yakınsama davranışını kontrol eden katsayı vektörüdür. $\vec{Y}$ vektörünün büyüklüğü, algoritmanın keşif (exploration) ve sömürü (exploitation) davranışını belirler. $\vec{N}$, rastgelelik içeren ağırlıklandırma katsayısıdır, en iyi çözüm ile mevcut çözüm arasındaki mesafenin ölçeklendirilmesini sağlar ve arama uzayında çeşitliliği artırır. a parametresi iterasyon sayısına bağlı olarak lineer şekilde azaltılır, bu azalma mekanizması sayesinde algoritma başlangıçta keşif, sonlara doğru sömürü davranışı sergiler.

$$\vec{\Delta} = |\vec{N} \cdot \vec{B}^*(t) - B(t)|$$

Balinaların konumu aşağıdaki şekilde güncellenir:

$$\vec{B}(t+1) = \vec{B}^*(t) - \vec{Y} \cdot \vec{\Delta}$$

Bu denklem, balinanın mevcut konumunu en iyi çözüm doğrultusunda ayarlamasını sağlar.

$$a = 2 - \frac{2t}{T}$$

$$\vec{Y} = 2\vec{a} \cdot \vec{r} - \vec{a}$$

$$\vec{N} = 2 \cdot \vec{r}$$

Burada;

$|\vec{Y}| < 1$; Yerel arama (yakınsama)

$|\vec{Y}| \geq 1$; Küresel arama

a , iterasyonla birlikte 2'den 0'a lineer olarak azalan kontrol parametresi

$\vec{r}$, [0,1] aralığında rastgele üretilmiş vektör

t : mevcut iterasyon numarası

T : maksimum iterasyon sayısıdır.

Kambur balinalar avlarına doğru spiral bir yörüngede yaklaşır. Bu davranış logaritmik spiral denklemi ile modellenmiştir:

$$\vec{B}(t+1) = \Delta' e^{bl} \cos(2\pi l) + \vec{B}^*(t)$$

Denklemde;

b , spiral şeklinin sabit parametresi (genellikle 1 alınır),

l , $[-1,1]$ aralığında rastgele üretilen sayıdır,

e^{bl} , spiral genişleme/daralma oranını belirler,

$\cos(2\pi l)$, dairesel hareket bileşenini temsil eder. Bu denklem, çözümün en iyi noktaya spiral biçimde yaklaşmasını sağlar.

Daralan kuşatma modelinde, alternatif olarak balina, en iyi çözüme doğrusal biçimde yaklaşır. Bu modelde $|\vec{Y}| < 1$ olduğunda yakınsama davranışı gerçekleşir.

Olasılıksal seçim mekanizmasında ise balina, spiral model ile daralan kuşatma modeli arasında rastgele seçim yapar. Bu seçim aşağıdaki koşullu ifade ile tanımlanır.

$$\vec{B}(t+1) = \begin{cases} \vec{B}^*(t) - \vec{Y} \cdot \vec{\Delta}, & \beta < 0.5 \\ \Delta' e^{bl} \cos(2\pi l) + \vec{B}^*(t), & \beta \geq 0.5 \end{cases}$$

$\beta \in [0,1]$, rastgele sayı,

$\beta < 0.5$, daralan kuşatma,

$\beta \geq 0.5$, spiral güncelleme gerçeşleşmesini ve bu mekanizma ile algoritmanın deterministik davranmasını engelleyerek çözüm uzayında çeşitlilik sağlar. Sonuç olarak balinalar, en iyi çözüme ya doğrusal daralma ile ya da spiral hareketle yaklaşarak yerel arama gerçeşleştirirler(Çelik ve Karadeniz, 2020).

1.2.2 Çoklu Stratejili Balina Optimizasyon Algoritması (Multi-Strategy Whale Optimization Algorithm-MSWOA)

Çoklu Stratejili Balina Optimizasyon Algoritması (Multi-Strategy Whale Optimization Algorithm – MSWOA), klasik Balina Optimizasyon Algoritması'nın bazı eksiklerini gidermek amacıyla geliştirilmiş bir optimizasyon yöntemidir. Klasik WOA, bazı durumlarda yerel çözümlerde kalabilmekte ve erken yakınsama gösterebilmektedir. MSWOA, bu problemi azaltmak için farklı arama stratejilerini birlikte kullanır (Xing, X., 2023).

MSWOA, birden fazla arama stratejisini birlikte kullanan bir optimizasyon algoritmasıdır. Kaotik başlangıç yöntemi, başlangıç çözümlerinin arama uzayına daha dengeli dağılmasını sağlayarak erken yakınsama riskini azaltır. Lévy uçuşu ise algoritmanın daha geniş alanları araştırmasına yardımcı olarak yerel çözümlerde takılı kalma olasılığını düşürür. Bunun yanında, pozisyon güncelleme ve rastgele arama mekanizmaları çözüm çeşitliliğini koruyarak arama sürecini daha dengeli hale getirir (Xing, X., 2023).

Klasik WOA'ya göre daha fazla hesaplama gerektirmesine rağmen, MSWOA genellikle daha doğru ve kararlı sonuçlar üretmektedir. Özellikle görüntü ve sinyal işleme alanlarında, çok seviyeli eşikleme gibi uygulamalarda başarılı sonuçlar verdiği görülmüştür (Xing, X., 2023). Bu özellikleri sayesinde MSWOA, karmaşık optimizasyon problemleri için etkili bir yöntem olarak değerlendirilmektedir.

Çoklu Stratejili Kambur Balina Algoritması (MS-WOA), temel Kambur Balina Algoritması (WOA) modelinin matematiksel güncelleme mekanizmasını genişleterek keşif-sömürü dengesini iyileştiren hibrit bir yaklaşımdır. Bu versiyon, temel WOA'nın zayıf yönlerini gidermek için ek matematiksel stratejiler ekler. Klasik modelde konum güncellemesi a 'nın 2'den 0'a lineer azaltılmasıyla kontrol edilir (Mirjalili & Lewis, 2016). MS-WOA'da ise bu lineer azalma yerine kaotik haritalar (örneğin lojistik harita);

$$a(t+1) = \mu a(t)(1 - a(t))$$

kullanılarak parametre üretimi doğrusal olmayan ve ergodik bir yapıya dönüştürülür. Ergodik yapı, üretilen parametre değerlerinin zamanla yalnızca belli bir aralığa sıkışmaması, tüm tanım aralığını düzensiz ama kapsayıcı şekilde dolaşması demektir. Ayrıca bazı varyantlarda konum güncellemesi Lévy uçuş terimi ile genişletilerek;

$$B(t+1) = B(t) + a \cdot Levy(\Delta)$$

biçiminde ağır kuyruklu sıçramalar eklenir (Ling vd., 2017). Böylece arama süreci yalnızca $|Y|$ koşuluna bağlı olmaktan çıkar, stokastik derinliği artar ve yerel minimumdan kaçış olasılığı yükselir. Sonuç olarak MS-WOA, temel modelin diferansiyel konum denklemlerini korurken parametre dinamiklerini kaotik ve adaptif yapılarla zenginleştirerek matematiksel olarak daha esnek ve küresel aramaya daha yatkın bir optimizasyon çerçevesi sunmaktadır (Ling vd., 2017).

1.2.3 Yapay Arı Koloni Algoritması (Artificial Bee Colony Algorithm - ABC)



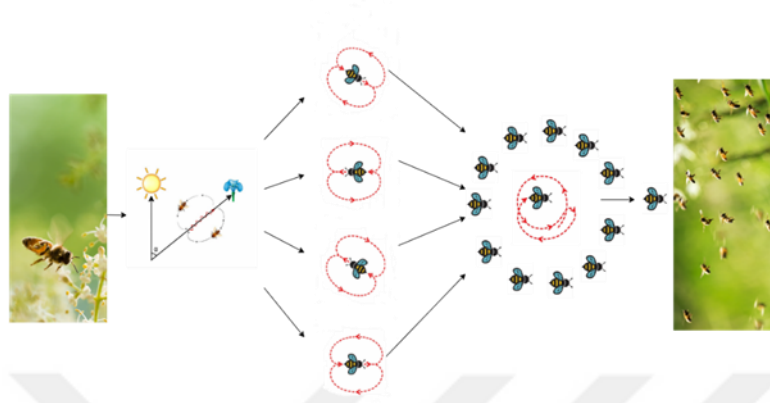
Şekil 1.6 Yapay arı koloni algoritması

Doğadaki bal arıları kendi içlerinde oluşturdukları hiyerarşiye göre iş bölümü yaparak, sezgisel yöntemlerle aralarında iletişim sağlarlar. Arılar koloniler olarak yaşayan sosyal böceklerdir. Bir kolonide Erkek arılar (drones), Kraliçe arı (queen) ve İşçi arılar (workers) olmak üzere üç çeşit arı bulunmaktadır (Karaboğa ve Akay, 2009). Kraliçe arı hayatında bir kez çiftleşir ve arı sayısı azaldığında yeteri kadar sayıya ulaşana kadar üremeye devam eder sonra ara verir. Bir diğer görevi ise kovan içerisine giren yabancı arıları salgıladıkları salgıdan tanıyarak tespit etmektedir. Böylece kovayı dış tehditlere karşı korur. Öldüğünde veya üreme özelliğini kaybettiğinde ise yerine kızlarından biri seçilir (Kartal, B., 2015). Erkek arılar üremeye yardımcı olur ve kraliçe arı ile çiftleştikten sonra ölürler. İşçi arılar dişi olmalarına rağmen üreme dışında kovanın güvenliği, havalandırılması, yabancı maddeler ve ölü arılardan temizlenmesi, besinin tespit edilmesi, toplanması ve saklanmasıyla sorumludur.

Bal arıları yiyecek arama davranışlarını, kuyruk dansından gelen konum bilgisi, koku, başka arıların kaynağa yönelmiş olmaları, hafızada tutulan kaynak ya da koku ile aralarında organize olarak merkezi bir otorite olmadan görev paylaşımı yaparak kaynağa ulaşır ve ihtiyaçları olan besin, nektar, polen veya balı temin edebilirler (Akay, B., 2009). Arılar yiyecek kaynaklarına nektar, bal ya da polen bulmak için gitmektedirler. Gidecekleri yiyecek kaynağının değeri, nektar yoğunluğu, çeşidi, nektarların çıkarılma kolaylığı, yuvaya yakınlığı gibi birçok etkene bağlıdır (Akyol ve Alataş, 2012).

Bal arılarının davranışlarını gözlemleyerek 2005 yılında Derviş Karaboğa' nın ortaya attığı ABC, arıları taklit ederek tasarlanan bir metasezgisel algoritmadır. Bal arılarının besin arama davranışlarını taklit ederek oluşturulan yapay arılar; işçi arılar görevli olarak, gözücü arılar ve kaşif arılar görevsiz olarak aralarında üç gruba ayrılır. Kovandan çıkan kaşif arılar, iyi bir besin arayışına girmek için rastgele uçuşlar yapar. Bu uçuşlar sırasında

çiçekler üzerindeki besinlerin kalitesine göre değerlendirerek kovana geri döner ve kovandaki işçi arılara besinin kalitesi, miktarı, uzaklığı ve konumunu belirtmek için çeşitli titreşim ve danslarla bilgi verir. Sallanma evresi ne kadar uzun sürerse, yiyecek kaynağı o kadar uzaktır (Karaboğa, D., 2005).



Şekil 1.7 Besin kaynağını tarif eden arıların dansı

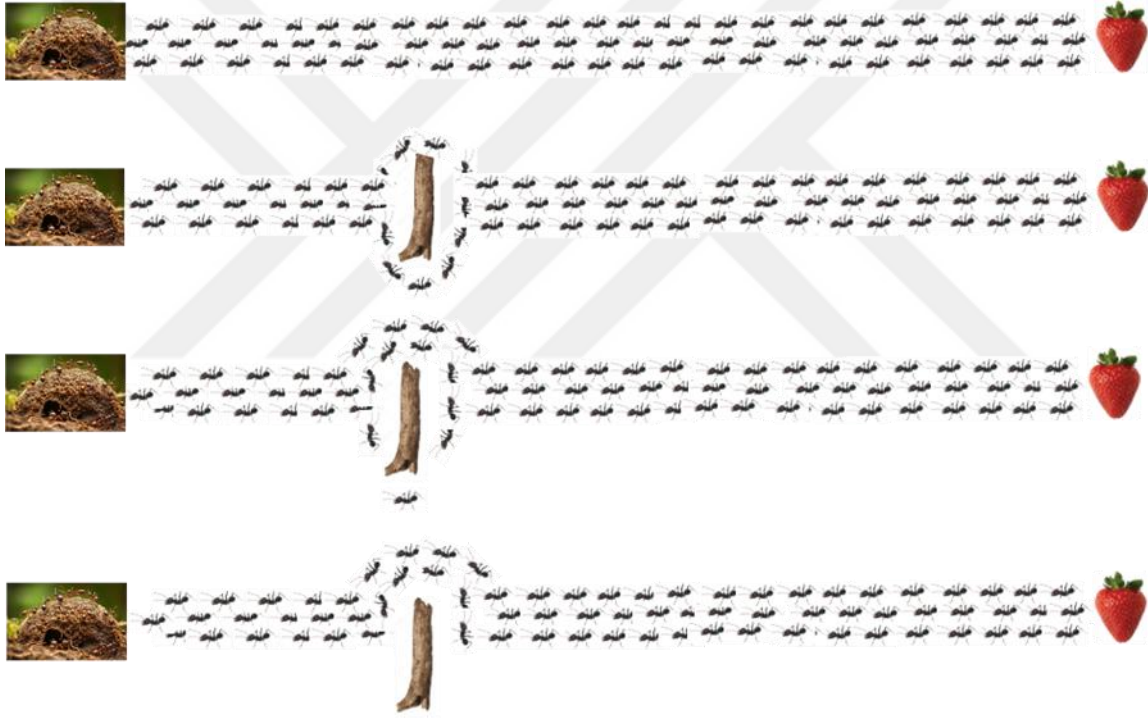
Kovan, güneş ve nektar kaynağı arasında oluşturduğu açı sayesinde kaşif arı nektar kaynağının yönünü (besinin hangi tarafta olduğunu) göstermektedir. Bal arısının kuyruk sallama süresi ise nektar kaynağının kovandan uzaklığını gösteriyor (Frisch, K. V., 1993).

Arı dansını sergilerken çerçeve üzerinde başı yukarı doğru ise besin kaynağı güneş yönünde, başı aşağı doğru ise besin kaynağı güneş yönünün tersi yönündedir. Eğer farklı açılarla dans ediyorsa besin kaynağının, güneşe göre kovanın hangi tarafa kaldığını, kovandaki arılara bilgi olarak aktarır (Dodoloğlu, A., 2017). Şekil 1.7’de arıların bu kusursuz iletişim döngüsü şematize edilerek gösterilmiştir.

1.2.4 Karınca Kolonisi Algoritması (Ant Colony Optimization Algorithm - ACO)

Karıncalar, doğada sürü halinde yaşayan ve karşılaştıkları sorunlara çözüm üretme konusunda gelişmiş özelliklere sahip olan canlılardır. En sık karşılaştıkları problemlerin başında, karıncanın yuvası ile besin arasındaki en kısa mesafeyi keşfederek diğer karınca sürüsünü doğru hedefe yönlendirmeleri gelmektedir. Karıncalar üzerinde ilk araştırmayı Goss ve arkadaşları (1989), laboratuvar ortamında yetiştirilen arjantin karıncalar üzerinde yaptıklarında, karıncaların görme duyusunun olmadığını görmüşlerdir. Yuvadaki besine veya besinden yuvaya ulaşmak için gittikleri güzergâh üzerine feromon adında bir kimyasal madde salgılayarak, bu maddenin oluşturduğu ize veya birden fazla iz varsa feromon miktarına göre en kısa yolu tercih ettikleri gözlenmiştir. Besin ile yuva arasında engelle rast geldiklerinde engel etrafından olası yolları gidip gelerek en kısa yolu

keşfederler. En kısa yol üzerinden besine ulaşan karınca aynı zamanda yuvaya en erken varan karıncadır. Böylece keşfettiği en kısa yol üzerinden sürekli gel-git yapacağı için görmemelerine rağmen yolda daha fazla feromon izi bıraktıkları için, artan feromon miktarına bağlı olarak en kısa yolu kimyasal salgı ile oluşturarak rotayı en kısa mesafe üzerine oluşturdukları gözlenmiştir. Goss ve arkadaşları tarafından 1989’da incelenen gerçek karınca davranışları sonucunda, Dorigo ve arkadaşları 1992 yılında, bu çalışmalar üzerinden yola çıkarak karınca davranışlarını matematiksel dile çevirerek bir modelleme tasarlayarak Karınca Kolonisi Algoritması (Ant Colony Optimization Algorithm - ACO)’nı oluşturmuşlardır (Goss vd., 1989; Dorigo vd., 1999). Şekil 1.8 incelendiğinde, besin kaynağına ulaşmaya çalışan karıncaların en kısa yol keşif aşamaları sırasıyla gösterilmiştir.



Şekil 1.8 Besin kaynağına ulaşmaya çalışan karıncaların en kısa yol keşif aşamaları

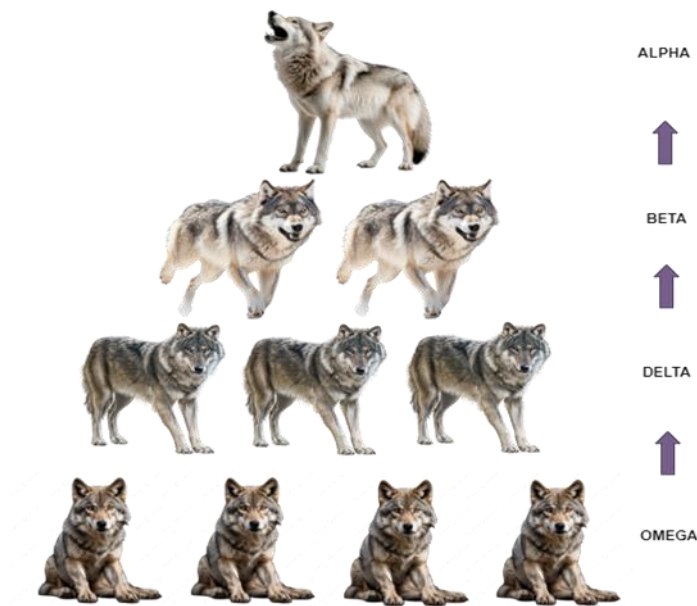
Karıncalar, yuvası ile besin arasındaki en kısa mesafeyi engelsiz şekilde bulabildiği gibi, yollarına bir engel çıkması durumunda da hemen duruma adapte olarak engelin sağına ve soluna olmak üzere eşit olasılıkla dağılarak yeni yol keşfetmeye koyulurlar. Eğer iki seçenekten biri daha kısa yol ise karıncalar bu yola daha çok yoğunlaşarak ve yuva ile besin arasında daha sık gel-git yapacakları için ilerledikleri rota üzerinde daha fazla kimyasal salgı bırakacaklardır. Dolayısıyla diğer uzun yolu tercih eden karıncalar da kısa sürede fark edip, feromon miktarına bağlı olarak yönünü değiştirerek, feromon miktarı

yoğun olduğu rotaya yoğunlaşıp yollarına bu güzergâh üzerinden devam ederek tekrar yeni en kısa mesafeyi keşfederek besin ile yuva arasındaki ulaşımı hızlandıracaklardır.

1.2.5 Gri Kurt Optimizasyonu (Grey Wolf Optimizer-GWO)

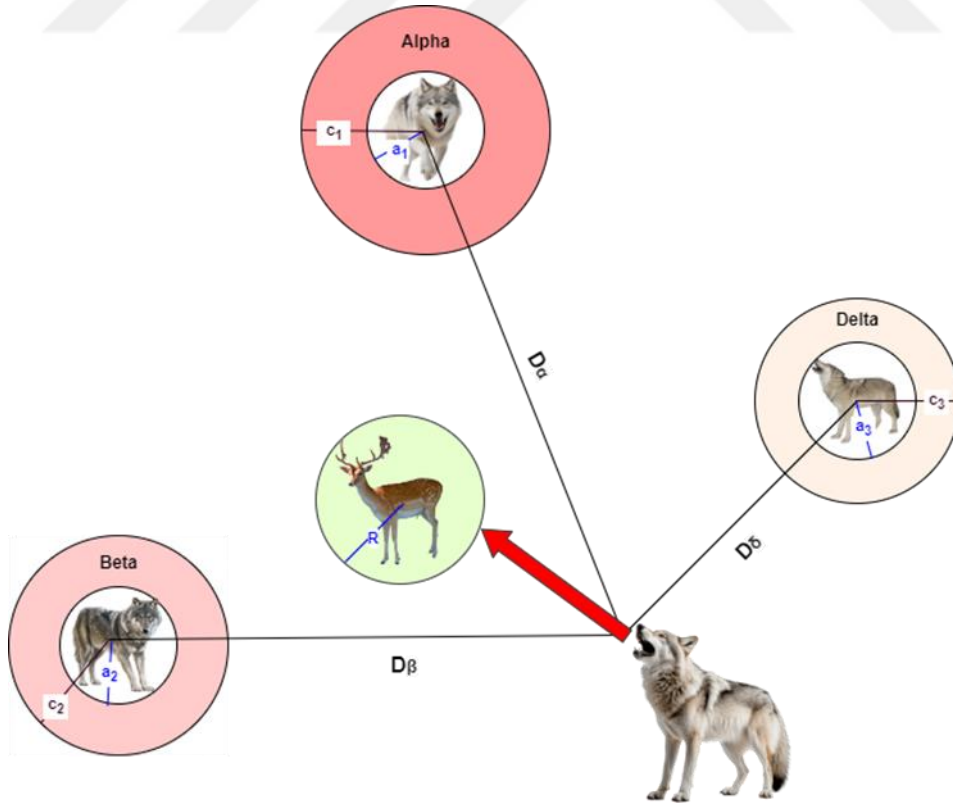
Mirjalili tarafından, gri kurt optimizasyon algoritması (GWO) 2014 yılında tasarlanmış olup kurtların avlanma stratejisi ve hiyerarşisini merkeze alan sürü tabanlı bir algoritmadır (Mirjalili ve Lewis, 2014). Kurtlar sürü halinde dolaşarak, içlerinde sosyal bir hiyerarşinin olduğu ve sayılarının en az 5 en çok 12 ‘ye kadar çıkabilen bir topluluk içerisinde yaşamlarını sürdürmeye çalışırlar. Hiyerarşi Şekil 9. ‘da olduğu gibi aşağıdan yukarı doğru çıkıldıkça liderlik gücünün arttığını gösteren bir yapı şeklinde ilerlemektedir. En alt tabanda bulunan kurtlara omega kurtları denir ve grubun en etkisiz elemanları olup, sürüyle birlikte besin ve güvenlik ihtiyaçlarını karşılamaktadırlar.

Tehlike anında grupta ilk gözden çıkarılan kurtlar omega grubunda olanlardır. Omegaya liderlik eden bir üst grup delta grubudur. Delta grubu kurtları aynı zamanda kendisinden hiyerarşik olarak üstte bulunan kurtlar olan alfa ve betanın yardımcısı konumunda olup, onların kararlarına itaat etmek zorundadır. Beta grubu kurtlar zirvedeki alfa(alpha) kurdunun sağ kolu olup alfa kurdundan sonra sürüdeki en yetkili kurttur ve alfaya yardımcı olur. Hiyerarşide en üst basamağı liderliğiyle kimseyle paylaşmayan alfa kurduna aittir. Sürünün tek lideridir ve yaşamsal fonksiyonları karşılamak için sürü adına karar verme merciidir. Genellikle beslenmek için avlanma, uymak için güvenli yer belirleme gibi faaliyetlerin kararlarını vererek sürünün geri kalanını yönlendirir.



Şekil 1.9 Gri kurt hiyerarşisi

Gri kurtlar avı keşfetme, konumunu belirleme, avın bulunduğu konumu tanıma yeteneğine sahiptirler. Kurtlar arasında hiyerarşi ön planda olmak üzere daima bir iş bölümü vardır. Av aramaya çıkan kurtlar ilk olarak avın yerini keşfederler. Arama sonucunda avı fark eden bu zayıf gruptaki kurtlar uluyarak, sürüdeki diğer güçlü kurtlara konumunu ulumayla bildirmektedir. Konumu tespit edilen ava yaklaşan sürünün diğer kurtları avı çevreleyerek etrafını Şekil 1.10 'da temsil edildiği gibi kuşatırlar. Avlanma genellikle alfa kurdunun yönetiminde gerçekleştiği gibi bazen avlanmaya beta ve delta kurtları da dahil olabilir çünkü bu üç grup kurt avın matematiksel konumu hakkında daha fazla bilgiye sahip oldukları için avı etkisiz hale getirmeleri daha kolay olacaktır. avı kısa sürede etkisiz hale getirmek için sürü içerisinde avı çevreleyen kurtlar arasında sürekli konum güncelleme yapılarak avın bu kuşatmadan kaçması kolaylıkla engellenir ve avlanma gerçekleşmiş olur. Kısaca alfa, beta ve delta kurtları avın tüm kaçış pozisyonlarını tahmin ederek sürekli ava bağlı olarak diğer kurtlarla restgele yerlerini güncelleyerek en iyi hamlelerle avı etkisiz hale getirir. Bu hiyerarşi ile iş bölümü üzerine tasarlanan yapay gri kurt sürüsü optimizasyon algoritması matematiksel atanma modeli, yiyeceğin ilk olarak güçlü kurda atanmasının ardından zayıf olana atanmasıdır. Kurt koloni algoritması bu davranışların taklit edilmesiyle geliştirilmiştir (Liu vd., 2011).



Şekil 1.10 Gri kurtların avlanma stratejileri ve ava göre güncellenen konumları

Grey Wolf Optimizer algoritması, kurtların sosyal hiyerarşisi ve avlanma davranışını matematiksel modellerken sürü zekâsına dayalı; α (alpha): En iyi çözüm, β (beta): İkinci en iyi çözüm, δ (delta): Üçüncü en iyi çözüm, ω (omega): Diğer aday çözümler olmak üzere dört sosyal sınıf üzerinden tasarlanmıştır. Bir optimizasyon problemi genel olarak şu şekilde tanımlanır:

$$\min_{k \in \Omega} f(k)$$

Burada $k = (k_1, k_2, \dots, k_d)$ karar değişkeni vektörünü, Ω çözüm uzayını ve $f(k)$ amaç fonksiyonunu temsil eder. Kurtların avı çevreleme davranışı aşağıdaki eşitliklerle modellenir.

$$\Delta = |C \cdot K_p(t) - K_p|$$

$$K(t+1) = K_p(t) - Y \cdot \Delta$$

$$Y = 2a \cdot r_1 - a$$

$$C = 2r_2$$

$$r = 2 - \frac{2t}{T}$$

Burada, t mevcut iterasyon, T maksimum iterasyon sayısıdır. r_1 ve $r_2 \sim U(0,1)$ rastgele vektörlerdir. a parametresi iterasyon boyunca 2'den 0'a lineer azalır.

GWO'da her birey pozisyonunu α , β ve δ çözümlerine göre günceller.

$$\Delta_\alpha = |C_1 \cdot K_\alpha - K|$$

$$\Delta_\beta = |C_2 \cdot K_\beta - K|$$

$$\Delta_\delta = |C_3 \cdot K_\delta - K|$$

Ara pozisyonları ise şu şekildedir:

$$K_1 = K_\alpha - A_1 \cdot \Delta_\alpha$$

$$K_2 = K_\beta - A_2 \cdot \Delta_\beta$$

$$K_3 = K_\delta - A_3 \cdot \Delta_\delta$$

Son pozisyon ise;

$$K(t+1) = \frac{K_1 + K_2 + K_3}{3}$$

şeklindedir.

1.2.6 Geliştirilmiş Gri Kurt Optimizasyonu (Improved Grey Wolf Optimization- IGWO)

Geliştirilmiş Gri Kurt Optimizasyonu (Improved Grey Wolf Optimizer- IGWO) algoritması, Mirjalili ve arkadaşları tarafından önerilen Grey Wolf Optimizer (GWO) algoritmasının geliştirilmiş bir versiyonudur. GWO algoritması, gri kurtların sürü hâlinde avlanma davranışlarını ve liderlik yapısını temel alarak oluşturulmuş sezgisel bir optimizasyon yöntemidir. Algoritmada en iyi çözümler alfa, beta ve delta kurtları ile temsil edilirken, diğer aday çözümler bu liderleri takip ederek çözüm uzayında ilerlemektedir (Mirjalili vd., 2014).

Klasik GWO algoritması basit yapısı ve az sayıda parametre içermesi nedeniyle avantajlı olsa da, bazı problemlerde erken yakınsama ve yerel minimumlara takılma gibi sınırlamalar gösterebilmektedir. Bu durum, algoritmanın keşif ve sömürü süreçlerini her zaman dengeli bir şekilde yürütememesinden kaynaklanmaktadır. Bu nedenle GWO'nun performansını artırmaya yönelik çeşitli iyileştirilmiş versiyonlar geliştirilmiştir (Mirjalili vd., 2014).

IGWO algoritması, GWO'nun temel yapısını koruyarak arama sürecini daha etkin hâle getirmeyi amaçlamaktadır ve parametrelerin dinamik olarak güncellenmesi veya arama mekanizmasının çeşitlendirilmesi gibi iyileştirmeler uygulanmaktadır. Bu sayede IGWO, çözüm uzayını daha iyi keşfedebilmekte ve küresel en iyi çözüme ulaşma olasılığını artırmaktadır (Gupta ve Deep, 2019).

IGWO algoritması, özellikle mühendislik optimizasyon problemleri, makine öğrenmesi uygulamaları, özellik seçimi ve enerji sistemleri gibi alanlarda yaygın olarak kullanılmaktadır. Daha dengeli yakınsama davranışı ve iyileştirilmiş çözüm kalitesi, bu algoritmanın farklı optimizasyon problemlerinde tercih edilmesini sağlamaktadır (Long vd., 2017).

IGWO, klasik GWO'nun keşif (exploration) ve sömürü (exploitation) dengesini iyileştirmek amacıyla geliştirilmiş varyantlarını ifade eder. Literatürde farklı IGWO yaklaşımları bulunmaktadır (Long vd., 2018; Nadimi-Shahraki vd., 2021). Matematiksel modellemesi ise aşağıdaki adımlar şeklindedir.

Klasik GWO'da a parametresi lineer azalırken, IGWO'da doğrusal olmayan azalma tercih edilir:

$$a(t) = 2 \left(1 - \left(\frac{t}{T} \right)^k \right)$$

Burada $k > 1$ parametresi yakınsama hızını kontrol eder. Bu yapı, erken iterasyonlarda keşfi artırırken son aşamada sömürüyü güçlendirir. Standart ortalama yerine ağırlıklı ortalama kullanılır.

$$K(t + 1) = \varphi_1 K_1 + \varphi_2 K_2 + \varphi_3 K_3$$

$$\varphi_1 + \varphi_2 + \varphi_3 = 1$$

Genellikle $\varphi_1 > \varphi_2 > \varphi_3$ seçilerek α çözümüne daha fazla ağırlık verilir. Başlangıç popülasyonunun çeşitliliğini artırmak için ise kaotik haritalar kullanılır:

$$k(t + 1) = \sigma k_t (1 - k_t)$$

Burada $\sigma = 4$ seçildiğinde sistem kaotik davranış sergiler. Bu yöntem yerel minimuma takılma riskini azaltır. Bazı IGWO varyantlarında parametreler popülasyon çeşitliliğine bağlı olarak güncellenir. Bu üstel azalma daha kontrollü yakınsama sağlar.

$$a(t) = 2e^{-\lambda \frac{t}{T}}$$

IGWO'nun çalışma sistemi, popülasyonu kaotik veya rastgele başlatır, amaç fonksiyonunu hesaplar, a , β ve δ çözümlerini belirledikten sonra adaptif a parametresini günceller. Pozisyonları ağırlıklı lider mekanizması ile günceller ve durdurma kriteri sağlanana kadar tekrarlayarak aynı döngü devam eder. GWO, klasik GWO'nun matematiksel yapısını koruyarak parametre güncelleme mekanizması, ağırlıklı lider etkisi ve kaotik başlatma gibi iyileştirmelerle daha hızlı yakınsama ve daha güçlü global arama yeteneği sağlar. Literatürde özellikle yüksek boyutlu ve doğrusal olmayan optimizasyon problemlerinde klasik GWO'ya kıyasla daha başarılı sonuçlar verdiği rapor edilmiştir (Long vd., 2018; Nadimi-Shahraki vd., 2021).

1.2.7 Parçacık Sürü Optimizasyonu (Particle Swarm Optimization-PSO)

Sürü, aynı cins yaşam formlarının veya bireylerin düzensiz dahi olsa, işbirliği içinde barınma, beslenme, savunma gibi eylemleri koordineli gerçekleştirmek için bir araya gelmiş canlı topluluğudur. Bu topluluğun ortak hareket etme eğilimine dayalı olarak, kolektif olmalarına rağmen kendi kendini organize olan ve genelde hayvan sürülerinden kurt, kuş, karınca, balıklar gibi birden fazla bireyden oluşan sürülerin eğilim gösterdiği ortak davranışlara sürü zekası denir. Sürü içinde bu işbirlikçi davranış nedeniyle, rastgele

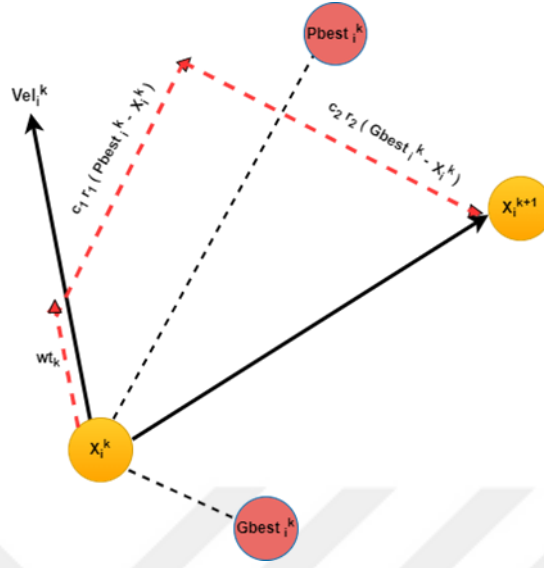
aramadan daha iyi bir arama stratejisi ortaya çıkar. Bu şekilde elde edilen akıllı arama stratejisi, genel olarak sürü zekası olarak adlandırılabilir. (Parsopoulos ve Vrahatis, 2010). Sürüyü oluşturan bireyler aralarında kolayca organize ve etkileşim içerisinde olabilecek ajanlardan oluştuğu için sürü zekâsı bu ajanların davranışlarını taklit etmeye çalışır. Sürü zekâsıyla gerçekleşen bu taklit sayesinde rastgele arama ve davranışlardan öte daha iyi arama ve davranışlar ortaya çıkar. Günlük yaşamda karşımıza çıkan problemleri çözmek açısından basitçe uygulanabilir olması ve işlevselliği nedeniyle sürü zekâsı optimizasyonda önemli rol almaktadır.

Parçacık sürü optimizasyonu (PSO), sürü zekâsını merkeze alan metasezgisel algoritmalarından biridir. Kuş ve balık sürülerinin hareketleri James Kennedy ve Russel Eberhart'a ilham olmuş ve doğrusal olmayan problemlere optimal sonuçlar bulmak için 1995-1996 yıllarında PSO tekniğini ortaya atmışlardır (Alataş, B., 2007).

PSO, sosyal olarak organize edilen sürüleri yöneten belirli kurallar çerçevesinde şekillenen stokastik bir optimizasyon algoritmasıdır. Parçacık sürü optimizasyon algoritmasına göre tanımlanmış ve sürüyü oluşturan parçacıklar, arama uzayında rastgele hız ve konumlara dağılırlar yani parçacıklar hareket ederken hızları ayarlanır ve ayarlanan hızlar önceki konumlara eklenerek parçacıkların hareketi devam ettirilir. Arama uzayında rasgele dağılan parçacıklara stokastik hareket eden bir arama noktası popülasyonu belirlenir. Her bir çözüm parçacık olarak adlandırılırken, her parçacık parametrelerden oluşur ve çok boyutlu uzayda bir noktaya karşılık gelir. Parçacıkların bulunduğu konum sürekli kontrol edilip en iyi konuma yönlendirilmesi ve aramanın bu şekilde gerçekleşmesi sağlanır. Her parçacığın ulaştığı en iyi konum kaydedilir. Kaydedilen bu konumlar popülasyonun diğer parçalarına veya tamamına bilgi olarak ulaştırılır ve parçacıkların çoğunluğu, o ana kadar kaydedilenlerin içinden en iyi konum tespit edilerek, en iyi konuma doğru yönlendirilir. Buradaki kayıt parçacıkların kendinden bir önceki en iyi konumu aklında tutup, takip etmesine yardımcı olur. Bu süreçte parçacıklar arasında gerçekleşen iletişim, algoritmanın yakınsama özelliğini referans alan sosyal ağ tarafından belirlenir. (Kennedy, J., 2010).

PSO algoritmasını oluşturan her bireye parçacık ismi verilmiş olup her parçacığın kendine has rastgele üretilmiş hız ve yön bilgileri mevcuttur. Bu parçacıkların tamamı bir araya gelerek sürüyü oluşturmaktadır. Çözüm uzayını veren parçacıkların pozisyona konum olarak adlandırılır. Konum hıza bağlı değiştiğinden, çok büyük problemlerde bu hıza bağlı olarak çözülebilmektedir. Hız ise, her bir parçacığın bulunduğu konumdan

daha iyi konuma yönelmesi ile sürüyü de en iyi konuma yönlendirme vektörüdür. En iyi konuma gelebilmesi için bir parçacığın, yönelme vektörü olmalıdır.



Şekil 1.11 Parçacıkların arama uzayındaki hareketleri (Fukuyama, Y., 2008)

Böylece sürüyü oluşturan her parçacık, kendinden sonra daha iyi pozisyonu keşfetme dürtüsüyle her iterasyon sonucunda amaç fonksiyonunu en mükemmel çözüme ulaştıracığından parçacığın hızı çok önemlidir. Parçacığın bulunduğu konumun çözüme uygun olup olmadığını karar verebilmek için bir uygunluk fonksiyonu geliştirilmelidir. Parçacığın konumu ile uygunluk fonksiyonu karşılaştırılarak çözüme karar vermek için uygunluk fonksiyonuna en yakın sonuç tercih edilir. Sürü içinden en iyi parçacığın en iyi uygunluk değerine kişisel en iyi (pbest) denir. Parçacık bir sonraki hız ve konuma geçerken daima bir önceki kişisel konum ve hızını karşılaştırıp, uygunluk değerine ulaşmak için her iterasyonda hız ve konumunu güncellemek durumundadır. Parçacıktan sonra sürünün sahip olduğu en iyi hız ve konuma yani pozisyonuna en iyi (gbest) denir. Sürüyü oluşturan tüm parçacıklar en iyi konum ve hıza doğru yönelme eğiliminde olup, sürüdeki her parçacık bir önceki konumu ile bir sonraki konumunu kıyas yaptıktan sonra günceller ve sürünün o an içindeki en iyi global konumunu elde etmeye çalışır ve her iterasyon sonucunda, iterasyon süresi boyunca en iyi kişisel konum ve global konumu elde ederek ulaşabileceği en iyi sonuca ulaşmaya çalışır.

Parçacık sürü optimizasyonunda, aramada referans alınan iki tür öğrenme başlığı vardır. İlki sosyal öğrenme olup, parçacık sosyal öğrenme ile sürünün herhangi bir parçacığının ziyaret ettiği en iyi çözüm olan “gbest” hafızaya kaydedilir. İkinci başlık ise bilişsel

öğrenme tabanlı olan parçacık, kendi kendine son ana kadar uğradığı en iyi çözüm olan “pbest” çözümünü hafızada saklar. (Bansal vd., 2019).

n boyutlu konum $(X_i^k = (x_1^k, x_2^k, x_3^k, \dots, x_n^k))$ ile hız vektörü $(Vel_i^k = (v_1^k, v_2^k, v_3^k, \dots, v_n^k))$ için en son i 'inci parçacık rastgele konum ve hıza sahip olarak başlar.

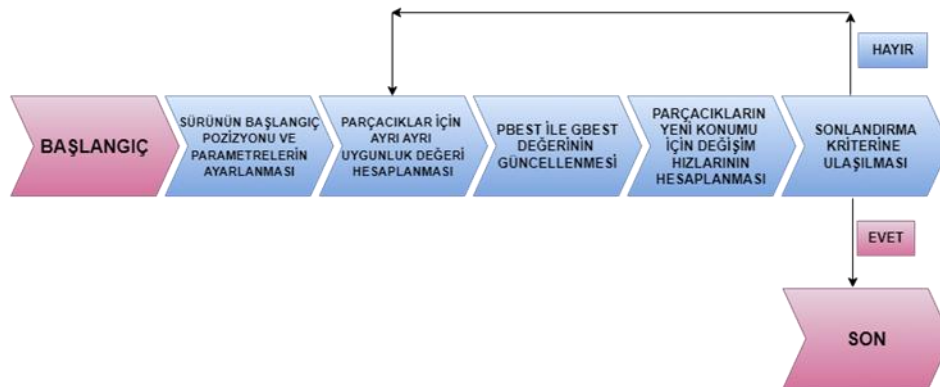
Parçacıkların her biri, bulunduğu konum için amaç fonksiyonunun konum ve değerini bilir. Belli bir parçanın en iyi bulunduğu konumu $Pbest_i^k = (p_1^k, p_2^k, p_3^k, \dots, p_n^k)$ ile gösterilir.

Sürünün bütün elemanlarının en iyi konumu ise $Gbest_i^k = (g_1, g_2, g_3, \dots, g_n)$ olarak gösterilir.

$$Vel_i^{k+1} = wt_k \cdot Vel_i^k + c_1 \cdot r_1 (Pbest_i^k - X_i^k) + c_2 \cdot r_2 (Gbest_i^k - X_i^k)$$

$$X_i^{k+1} = X_i^k + Vel_i^{k+1}$$

Vektörü gösterirken kullandığımız Vel_i^k , i . parçacığının k iterasyonundaki hızını gösterir. X_i^k , i . parçacığının k iterasyonundaki konumunu yani koordinatlarını gösterir. $Pbest_i^k$, k iterasyonundaki i . parçacığın en iyi koordinatını temsil eder. $Gbest_i^k$, ise k iterasyonundaki sürünün global(küresel) en iyi parçacığını ve bu parçacığın en iyi koordinatını temsil eder. c_1 ve c_2 , öğrenme faktörleri, r_1 ve r_2 ise $(0,1)$ aralığında düzgün dağılmış rastgele değişkenlerdir. wt_k ise önceki hızların mevcut hız üzerindeki etkisini kontrol etmek için bir katsayı olan atalet ağırlığıdır (Ahmed ve Glasgow, 2012).



Şekil 1.12 PSO akış diyagramı

Parçacık sürü algoritmasının, Şekil 1.12 'ye göre akış diyagramının adım adım ilerlemesi aşağıdaki gibidir.

1. Sürünün başlangıç hızları ve başlangıç koordinatları rastgele tasarlanır.
2. Sürü içerisindeki tüm parçacıkların uygunluk değerleri ayrı ayrı hesaplanır.
3. Yerel en iyi (Pbest), sürünün her bir parçacığı için ayrı ayrı bulunur.
4. Küresel en iyi (Gbest) ise sürüdeki yerel eniyiler içerisinde seçilir.
5. Tüm parçacıkların hızları, $Vel_i^{k+1} = wt_k . Vel_i^k + c_1 . r_1 (Pbest_i^k - X_i^k) + c_2 . r_2 (Gbest^k - X_i^k)$ denklemi ile güncellenir.
6. Her parçacık, $X_i^{k+1} = X_i^k + Vel_i^{k+1}$ denklemi ile yeni konumuna taşınır.

Sonlandırma veya yakınsama durumunu sağlarsa çözüme ulaşılır, sağlamasa tekrar ikinci adıma dönerek adımlar son maddeye kadar tekrar edilir.

1.2.8 Sinüs Kosinüs Algoritması (Sine Cosine Algorithm - SCA)

Sinüs Kosinüs Algoritması (Sine Cosine Algorithm - SCA), 2016 yılında Mirjalili tarafından geliştirilmiş bir optimizasyon yöntemidir (Mirjalili, A., 2016). Algoritmanın çıkış noktası, çözümlerin konumlarını sinüs ve kosinüs fonksiyonları aracılığıyla güncelleme fikridir. Başlangıçta aday çözümler rastgele belirlenir; ardından her iterasyonda bu çözümler ya mevcut en iyi çözüme yaklaşır ya da arama uzayında rastgele bir yönelime kayar. Böylece hem geniş arama (exploration) hem de çözüm iyileştirme (exploitation) dengesi sağlanmış olur (Mirjalili, A., 2016).

SCA, farklı alanlarda uygulanabilecek esnek bir algoritmadır. Literatürde mühendislik tasarımında yapısal optimizasyon, enerji sistemlerinde kaynak dağıtımı, görüntü işleme, veri sınıflandırma ve robot yol planlama gibi problemlerde kullanıldığı görülmektedir (Rizk Allah, 2022). Algoritmanın öne çıkan avantajları, basit ve anlaşılır yapısı, az sayıda parametre ile çalışabilmesi ve hem sürekli hem de kısıtlı optimizasyon problemlerinde uygulanabilir olmasıdır. Bununla birlikte bazı yüksek boyutlu veya karmaşık problemlerde algoritmanın yerel optimuma takılabileceği veya yakınsama süresinin uzayabileceği belirtilmiş; bu nedenle literatürde hibrit veya geliştirilmiş SCA varyantları önerilmektedir (Rizk Allah, R. M., 2022).

1.2.9 Yer çekimsel Arama Algoritması (A Gravitational Search Algorithm – GSA)

Rashedi ve arkadaşları tarafından 2009 yılında ortaya atılan fiziksel tabanlı olan yerçekimsel arama algoritması (GSA), Newton'un yerçekimi ve hareket prensibine dayanarak geliştirilmiştir. GSA' ya göre her cismin sahip olduğu belirli bir kütlesi vardır ve cisimler yerçekimi kuvveti ile birbirlerini çekerler. Çekme kuvveti doğrultusunda cisimler, kütlesi en ağır olan cisimlere doğru hareket halindedirler. Ağır olan cisimler yavaş hareket etmelerinden kaynaklı olarak diğer cisimleri kendilerine doğru çekerler ve bu çekme işlemi sonucunda tüm kütlesi hafif olan cisimler ağır olan cisimlere doğru çekilerek harekete geçerler. Bu işlem optimizasyona uyarlandığında her cisim bir çözüm olarak kabul edilir ve bu çözümlerin ağırlıkları performanslarını temsil etmektedir. Çözümlerin, ağırlığı en fazla olan cisme doğru yakınsaması algoritma sona erene kadar devam etmektedir ve algoritma sona erdiğinde ağırlığı en fazla olan çözüm global çözümü vermektedir (Rashedi vd., 2009; Can, Ü., 2014).

1.2.10 Yapay Atom Algoritması

Kimya tabanlı sezgisel bir algoritma olan Yapay atom algoritması, 2012 yılında Karcı tarafından kovalent bağ ve iyonik bağın modellenmesiyle ortaya çıkmıştır. Algoritmanın problemini oluşturan yapı atom olup, elektronlar ise bu problemi sağlayan parametreler olarak kabul edilir. İlk basamakta birden fazla atom ile işleme başlanıp oluşan çözümler kümesine atom kümesi adı verilir. Algoritma kovalent bağ ve iyonik bağ işlemi olmak üzere iki basamaktan oluşur. Kovalent bağ basamağında, en az iki atom arasında paylaşılacak elektronlar belirlenir. İyonik bağ basamağında ise değerinin değiştirilmesi gereken elektronlar belirlenir. İyonik bağ basamağında değer değiştirilmesinde amaç etkisi kötü olan elektronlar için yeni değer belirlemektir ve elektronların içinden kovalent bağ kapsamındaki elektronlardan en iyi değere sahip olanı seçilmektedir. Tüm atomlar adına her bir elektronun amaç fonksiyonu üzerindeki değişimler hesaplanarak, en iyi sonucu veren çözüm global en iyi olarak belirlenmektedir (Karcı, A., 2012).

1.2.11 Farksal Gelişim Algoritması (Differential Evolution-DE)

Doğrusal olmayan optimizasyon problemlerinin çözümünde sezgisel ve deterministik yöntemler ile en iyi sonuca ulaşmaya çalışırken, deterministik yöntemlerle modelleme yapmak fazla matematiksel işlem gerektirdiği için, problemin boyutu ve karmaşıklığı arttığında, deterministik yöntem yetersiz kaldığı için sezgisel yöntem, kolaylık ve uygulanabilirlik açısından önem kazanmıştır. Basit ve güçlü çalışan bu algoritma sürekli

optimizasyon problemlerinin çözümünde başarılı olup, temeli genetik algoritma olan ancak genetik algorithmadan bazı kısımlarda ayrılan bir algorithmadır. DE’de değişkenlerin gerçek değerleri kullanılırken, GA’ da gerçek değerli değişkenler kullanılsa da birtakım farklılıklar vardır. DE’de çaprazlama, mutasyon ve seçim operatörleri kullanılmasına rağmen genetik algorithmadan farklı olarak her bir operatör sırasıyla her bireye uygulanmaz. DE ‘de bireyler tek tek düşünülüp aralarından rastgele seçilen bireyler ile yeni birey elde edilir (çaprazlama ve mutasyon). DE’de kullanılan mutasyon algoritma performansını artırarak algorithmayı kullanışlı hale getirir. Seçilen birey yeni bireyle karşılaştırılarak daha iyi olan birey seçilir. (Keskintürk, T., 2006).

Sürekli verilerin ele alındığı problemlerde yaklaşık sonuçlar veren, işleyişi bakımından genetik algorithmaya dayanan popülasyon tabanlı sezgisel optimizasyon çeşidi olan Farksal Gelişim algoritması (DE), Price ve Storn tarafından 1995 yılında ortaya atılmıştır. Farksal gelişim algoritması birçok alanda kolaylık sağlayarak gerçek sayı fonksiyonları için kullanışlı olan en önemli algorithmalardan biridir. Temel yapısı istatistik ve analiz üzerine olan alanlardan mühendislik, finans gibi alanlarda çözülmesi kolay olamayan problemlere yaklaşık bir çözüm bulmak için hızlı, uygulanabilir ve basit yapısı itibarıyla kolay kodlanabilir olması açısından diğer sezgisellerden bir adım öne çıkarak kullanım kolaylığı sağlar (Storn ve Price, 1997; Price vd., 2005).

1.2.12 Genetik Algoritması (Genetic Algorithm –GA)

Birçok alana uyarlanıp kullanılması açısından metasezgisel temelli olan evrimsel algorithmlar adını Darwin’in (1809-1882) evrim teorisinden alır. Alman araştırmacı Ingo Rechenberg (1934-2021) sayesinde literatüre geçen evrim algorithmlarının felsefesi güçlüünün yanında zayıfın elenerek hayatta kalabilmesi için canlıların ortama adapte olmasına dayanır (Rechenberg, I., 1965; Rechenberg, I., 1971)

Genetik algoritma, evrime dayalı bir algoritma türü olup, Michigan Üniversitesinde profesör olan ve ilk defa 1975 yılında Prof. John Holland, öğrencileri ve arkadaşları tarafından geliştirilen algoritma türü olup makine öğrenme alanında en iyileme problemleri için geliştirilmiştir. Genetik algoritma, yapısını oluşturan genetik bilim ve genetik bilimin alt basamaklarını oluşturan çaprazlama, seleksiyon ve mutasyon, evrimsel algorithmanın desteklediği bir optimizasyon tekniğidir. (Tuncer, A., 2007).

Holland, “Adaptation in Natural and Artificial Systems” adlı kitabında, 1975 yılında yaptığı çalışmaları bir araya toplamıştır. Doğal seçim ve genetik evrimden faydalanarak

en iyiyi bulmak için çalışmalar yapmıştır. Bu çalışmaların neticesinde, makine öğrenme problemlerinde çevreye uyum sağlayacak şekilde en uygunu elde etmek için bilgisayar algoritma modelleri tasarlanmıştır. Biyolojik sistemlerde mekanik tek bir yapının öğrenmesinden ise, çoklu mekanik yapıların seleksiyon, çoğalma, mutasyon ve çaprazlama gibi evrelerden geçerek başarılı bir şekilde öğrenmeleri gerçekleştirebilen bireyleri elde edilebileceğini fark etmiştir. 1985 yılında Holland'ın öğrencisi olan David E. Goldberg'in doktora tezinde makine öğrenmesi, nesne tanıma, görüntü işleme ve işlemsel arama alanlarında kullandığı ve gaz boru hatları denetimi üzerine çalıştığı tezi, 1985 National Science Foundation Genç Araştırmacı ödülünü kazandırmış olup evrimsel algoritmaların pratik kullanımının da olabirliğini kanıtlamıştır. Çalışmasında evrimsel algoritmaya dayalı 83 uygulamayı göstererek, dünyada farklı konu ve alanlarda genetik algoritmanın kullanılmakta olduğunu göstermiştir (Akduman, B., 2010).

Evrimsel algoritma çoklu bir veri kümesi içerisinde en özel veriyi bulmak için kullanılan çok ideal bir optimizasyon algoritmasıdır. Dayanağı evrim teorisi olan bu algoritma, güçlünün ve en iyi özelliklere sahip olanın hayatta kalması, zayıf ve yetersiz olanın elenerek en ideal olanın hayatta kalması döngüsüne dayanan çözümler sunar. Evrimsel algoritma, topluluk adı verilen ve rastgele oluşturulan bir sürü çözüm kümesini içinde barındırdığı, gen havuzu ile çalışmayı başlatır. Genlerin kombinasyonlarından oluşan kromozomların her biri bir değişkeni temsil eder. Uygunluk fonksiyonu değişkenlerinin tamamını bünyesinde bulunduran bireyleri oluştururken bu bireyler fonksiyon değişkenlerinin yan yana sıralanan genlerinden oluşur. Bir matris olarak tasarlanan topluluğun bir satırı bireyi oluştururken, bireylerin tümü ise topluluğu oluşturur. Her bireyin genleri birbirinden farklı kodlanır. Kodlama şekillerinden en çok tercih edileni Binary (ikilik sayı sistemi) 'dir. Binary kodlama sisteminde kromozomlar 0 ve 1 genlerinin kombinasyonları şeklindedir.

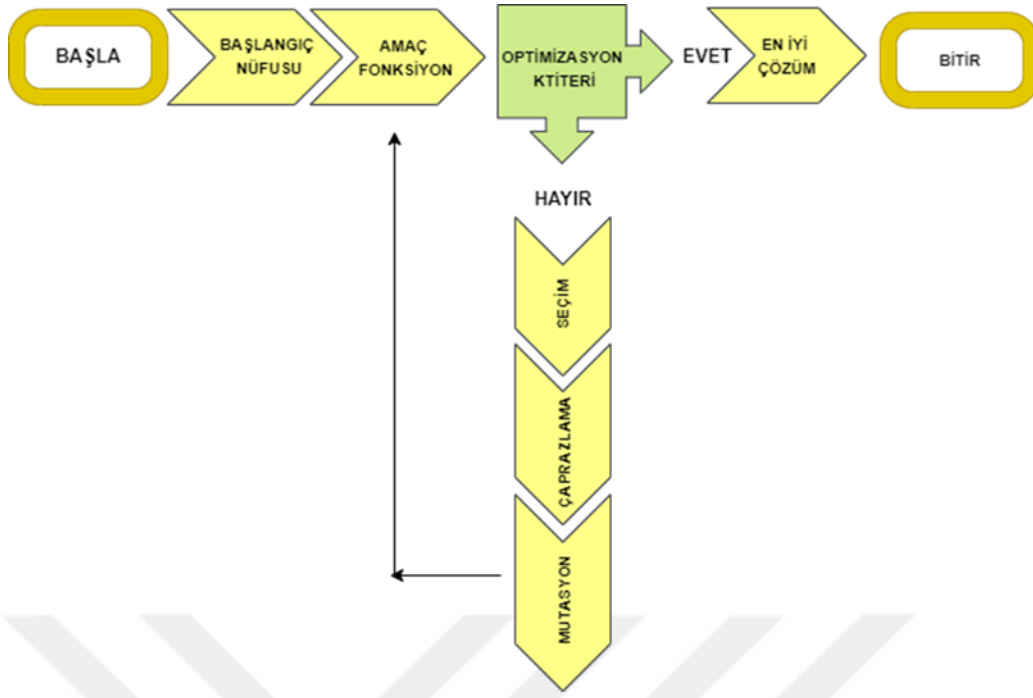
Gen havuzu oluşturulurken ikili sayı sistemi kümesinin elemanları olan 0 ve 1'den yararlanılarak başlangıç topluluğu oluşturulur. Gen havuzunun hacmi, problemin çözümü için aranan uygunluk fonksiyonu değişken sayısı, değişkenlerin değer aralığı, değer aralığındaki eleman sayısına bağlı olup, gen havuzunun her bir satırına topluluk, bu topluluğa ise birey adı verilir. Bu topluluklar her bir değişkenin genler ile kodlanmasıyla kromozomları oluşturur. Bireyin oluşması için değişkenlerin kodlanması çizelgede gösterilmiştir (Akduman, B., 2010).

Değişkenler	X ₁	X ₂	X ₃	X ₄	X ₅	X ₆	X ₇	X ₈	X ₉	X ₁₀	X ₁₁	X ₁₂
Birey 1	1	0	1	0	1	1	0	0	0	1	0	1
Birey 2	1	0	0	0	1	1	1	1	0	1	0	0

Çizelge 1: Değişkenlerin kodlanması ile bireyin elde edilmesi.

Çözümün bulunması için öncelikle bir başlangıç noktasının seçilmesi geleneksel optimizasyon yöntemidir bu yüzden ilk nokta yaklaşık olarak tahmin edilemezse, çözüme ulaşacağımız süre artar ve işlem hacmi genişler. Evrimsel algoritmalarda ise çözüme topluluk sayısı belirlendikten sonra başlanır ve çözümler iyi değilse bir sonraki jenerasyona geçerek adımlar tekrar edilir (Akduman, B., 2010).

Her bir birey için uygunluk fonksiyonu değeri bulunur ve topluluğun devamı ise biyolojik kurallara tabii olarak gerçekleşir. Uygunluk fonksiyonu değerleri yani başarılı olan kromozomlar, belirlenen özellikleri en iyi taşıdığı ölçüde seçilirler, özellikleri taşımayan veya çok az taşıyanlar ise elenirler. Başarılı bir şekilde seçilen kromozomların meydana getirdiği ve ebeveyn bireyler olarak kabul edilenler arasında üreme gerçekleştirilerek, bir sonraki jenerasyonu oluşturacak yeni bireyler oluşur ve neslin devamı için bu döngü kendini sürekli tekrar eder. Evrimsel algortimada meydana gelen bu döngü, ebeveyn bireylerin bazı genlerini yeni oluşturdukları bireyler üzerine kopyalama işlevi gerçekleştirme olayına çaprazlama (üreme) adı verilir. Oluşan yeni bireyler, kendisini oluşturan bireylerin birebir kopyası olmamak için mutasyona uğratılır ve böylece rastgele genler mutasyona uğrayarak farklı yeni bir birey elde edilir. Somut bir örneklendirme ile bireyin gen zincirinin belirli bir kısmında 0 bulunuyorsa bu değer 1, 1 bulunuyorsa bu değer mutasyon ile 0 olur. Algoritmada gerçekleşen değişiklik yani mutasyon oranı, topluluk sayısına göre değişir. (Pohlheim, H., 2006), başarılı olan genler gelecek nesillere taşınırken zayıf olan genler ise elenir ve süreç içerisinde yok olur. İlk başlarda gen havuzunu oluşturan kümede başarısız birçok gen varken, her gelen yeni jenerasyonla gen havuzunun kalitesi zamanla artarak başarılı bireyler çoğalmaya başlar. Sürecin sonunda ise tüm bireyler elde edilmek istenen ideal bireye dönüştüğünde optimizasyon işlevini tamamlamış ve sonuca ulaşmış olur. Sonlandırma şartında dikkat edilmesi gereken hususlar; nesil sayısı, çalışma süresi, bu sürede devamlı aynı sonuçların elde edilmesi olarak sıralanabilir.



Şekil 1.13 Genetik Algoritma akış diyagramı

1.2.13 Yarasa Optimizasyonu (Bat Algorithm - BA)

Çok fazla türü olduğu tahmin edilen yarasalar, yönlerini bulmaya çalışırken yüksek ve kısa ses sinyali yayarak bir nesneye çarpmasını ve yankı yapmasını yorumlayarak nesnenin kendisine ne kadar uzaklıkta olduğunu hesaplayabilen canlılardır. Kuş, yunus ve yarasalar tarafından ses dalgalarının yansımaları yorumlayarak nesnelerin yerini belirlemek için kullanılan bir teknik olan ekolokasyon, biyolojik sonar görevi görerek cismin boyutu ve uzaklığı hakkında çeşitli veriler verir. Yarasalar, ekolokasyon ile ses dalgaları yayarak, ses navigasyonu görevi yapan bir tekniktir. Bu şekilde elde ettiği verilere dayanarak avın hangi yönde olduğunu kolaylıkla tahmin edebilir. Boyutu küçük olan yarasalar, ekolokasyonu etkin bir şekilde kullanırken, yarasaların boyutu büyüdükçe bu durum daha az gözlenmektedir. Yarasaların yaydıkları ses sinyali, yaklaşık 5-20 ms kadar sürer ve genellikle 25-150 kHz aralığında sabit bir frekansa sahiptir. Mikro yarasalar incelendiği zaman yaklaşık olarak saniyede 10-20 arasında ses sinyalleri yayarlar fakat avına yaklaştıkça ses sinyali saniyede 200 yaklaşır. Avını bulana kadar yüksek ses sinyali üreten yarasalar, avın yerini tespit ettikten sonra sessizce avına yaklaşırlar (Chawla ve Duhan, 2015). Yang, yarasaların ekolokasyon yeteneğini ve av bulma konusundaki tipik davranışlarını modelleyerek, 2010 yılında Yarasa Algoritmasını önermiştir (Yang, X.-S., 2010a).

1.2.14 Karga Arama Algoritması

Yeni bir meta-sezgisel yöntem olan ve akıllı sürü davranışlarına dayanan Karga Arama Algoritması, A. Askarzadeh tarafından 2016 yılında literatüre kazandırılmıştır. Karga Arama Algoritması, gelecekte herhangi bir zamanda besin ihtiyacı olduğunda depoladığı ekstra besin maddeleri ve bu besinlere ihtiyaç duyduğunda geri almasından esinlenerek ortaya çıkmış bir meta-sezgisel algoritmadır (Askarzadeh, A., 2016).

Kargalar iyi birer gözlemci olup diğer kuşları gözlemledikleri, bu kuşların besinlerini sakladıkları yerleri izleyerek tespit ettikleri ve besinlerin sahibi ayrıldıktan sonra besinleri hırsızlık yaparak çaldıkları görülmektedir. Karga hırsızlık yaparak çaldığı besini gelip başka kargada aynı yöntemle çalmaması için yer değiştirerek önlem almaktadır. Aslında hırsız karga, diğer hırsızın tahminlerini kestirmeye çalışarak, besinin çalınmasını önlemek için önbelleklerini kullanarak en güvenli rotayı oluştururlar. Kısaca sürü halinde yaşayan kargalar, sakladıkları besinin yerini ezberleyip, başka kargaların gözlemlemesi sonucunda çalmaması için önbelleklerini kullanarak besini olası hırsızlıklara karşı korurlar (Clayton ve Emery, 2005).

Kargaların açgözlü karakteristik yapısı ile sergiledikleri bu davranışlar, optimizasyona uyarlandığında kargaların görevi araştırmadır ve arama alanı ise bulunabileceği her konumu uygun bir çözümü temsil eder. Besin kaynağının kalitesi objektif(uygunluk) fonksiyonu temsil ederken, bu alanda bulunan en iyi olan besin kaynağı ise problemin global (en iyi çözüm) çözümünü temsil eder. Karga arama algoritması, optimizasyona uyarlanarak, kargaların sezgisel davranışları ile problemin çözümünü bulmada kolaylık sağlar.

Kargalar ortamdaki kuşları gözlemleyerek besinleri nereye depoladıklarını öğrenir. Ardından besinin sahibi gittikten sonra harekete geçerek kendisine ait olmayan besinleri çaldıkları gözlenir. Aynı hırsızlığı bir diğer karga kendisine yapmasın diye besinlerin belirli aralıklarla yerlerini değiştirerek besinlerini çalınmasını önlemek amacıyla diğer kargaların tüm davranışlarını tahmin etmeye çalışarak en güvenli yeri tespit ederek belirler ve bunun için güvenli bir yol oluşturur. Popülasyon tabanlı bir metasezgisel algoritma olan ve kargaların sürü şeklinde yaşayıp, sakladıkları besinleri koruma adına konumu tespit ve ezber yeteneklerinin olduğu, açgözlülük ve hırsızlıkla sürekli besin için takip yapan bu kuşların tüm olasılıkları hesaplayarak önbelleklerinin geliştiği ve olası hırsızlıklara karşı besinlerini çalınmaktan koruduğu tespit edilmiştir.

1.2.15 Optik Mikroskop Algoritması (Optical Microscope Algorithm - OMA)

Optik Mikroskop Algoritması (OMA), 2023 yılında Cheng ve Sholeh tarafından geliştirilen bir meta-sezgisel optimizasyon yöntemidir. Algoritmanın çıkış noktası, optik mikroskopların büyütme (magnification) mantığını inceleyerek bu yapıyı arama sürecine uyarlamaktır. Mikroskopta bir görüntü önce normal şekilde görülür, ardından objektif mercek ile alan daraltılır ya da oküler mercek ile görüntü büyütülür. OMA’da da aynı mantık benimsenmiştir. Objektif mercek geniş çaplı keşfi (exploration), oküler mercek ise daha odaklı ve iyileştirici aramayı (exploitation) temsil eder. Bu yaklaşım sayesinde OMA, çok az parametre gerektirmesine rağmen hem farklı test fonksiyonlarında hem de çeşitli mühendislik problemlerinde başarılı sonuçlar verebilmektedir (Cheng ve Sholeh, 2023).

Popülasyon temelli yapısı sayesinde çözüm uzayında etkili bir şekilde arama yapabilen OMA, keşif ve sömürü arasında dengeli bir yaklaşım sunar. Bu yapısı, algoritmanın farklı problemlerde esnek ve kararlı bir performans göstermesine katkı sağlar. Ayrıca PMSOMA gibi geliştirilmiş sürümleri, nüfus çeşitliliğini artırarak yerel minimumlara takılma riskini azaltmaktadır (Guo ve Gu, 2024)

Fizikten esinlenerek geliştirilen OMA, az sayıda parametre gerektirmesi sayesinde kullanımı oldukça kolay olan, sezgisel ve yenilikçi bir algoritmadır. Hem keşif hem de sömürü süreçlerini dengede tutabilen yapısı, gerektiğinde farklı düzenlemelerle daha da güçlendirilebilir. Benchmark testlerinde ve mühendislik uygulamalarında iyi performans göstermesi, çok fazla ayar yapmaya gerek kalmadan kısa sürede etkili sonuçlar elde etmeye olanak tanır.

Orijinal OMA, 50 farklı benchmark fonksiyonunda test edilmiş ve çeşitli metasezgisel algoritmalarla karşılaştırıldığında başarılı sonuçlar vermiştir. Ancak oldukça yeni bir algoritma olduğundan, literatürde çok fazla çalışma bulunmamakta ve temel sürümde zaman zaman yerel minimumlara takılma riski ortaya çıkabilmektedir. Bu sorunları azaltmayı amaçlayan geliştirilmiş sürümlerden biri olan PMSOMA ise daha fazla hesaplama yükü gerektirebilmektedir. Ayrıca OMA’nın az parametreliliğine rağmen, mercek benzetimine dayanan stratejisinin her problem türüyle tam olarak örtüşmediği durumlar da olabilir.

Mühendislik problemlerinde uygulandığında ise, yapısal optimizasyon ya da proje planlaması gibi alanlarda nispeten az sayıda amaç fonksiyonu değerlendirmesiyle tatmin edici sonuçlar elde edilmiştir (Cheng ve Sholeh, 2023).

1.2.16 Gübre Böceği Optimizasyon Algoritması (Dung Beetle Optimizer-DBO)

Gübre Böceği Optimizasyon Algoritması (DBO), doğadaki gübre böceklerinin davranışlarından esinlenerek geliştirilmiş bir algoritmadır. Bu algoritma, böceklerin yuvarlama, beslenme, dans etme ve üreme gibi davranışlarını çözüm uzayında yeni çözümler aramak için kullanır. DBO, hem geniş çaplı keşif hem de en iyi çözümleri sömürme yeteneğine sahiptir. Orijinal çalışmada algoritma, farklı benchmark fonksiyonlar üzerinde test edilmiş ve çoğu popülasyon tabanlı yöntemle karşılaştırıldığında hem hızlı yakınsama hem de çözüm doğruluğu açısından başarılı bulunmuştur (Xue ve Shen, 2023).

Algoritmanın bazı sınırlamaları da vardır. Örneğin, standart DBO'nun nüfus çeşitliliği yeterli olmayabilir ve bu durum yüksek boyutlu problemlerde yerel optimumlara takılma riskini artırabilir (Wang, Y., 2025).

Bu sınırlamaları azaltmak için literatürde farklı iyileştirme yöntemleri önerilmiştir. Örneğin, başlangıç nüfusunu iyileştiren ve arama stratejilerini çeşitlendiren çoklu strateji yöntemleri kullanılarak DBO'nun performansı artırılmıştır. Bu sayede algoritma, klasik versiyona göre daha hızlı ve daha doğru çözümler üretebilmektedir (Zhang ve Liu, 2025).

DBO algoritması ayrıca robot yol planlama, engelden kaçınma ve hiperparametre optimizasyonu gibi mühendislik uygulamalarında da kullanılmıştır. Bu uygulamalar, algoritmanın farklı problemlere uyarlanabilirliğini ve esnek yapısını göstermektedir (Tan vd., 2024).

1.2.17 Kum Kedisi Sürüsü Algoritması (Sand Cat Swarm Optimization-SCSO)

Kum Kedisi Sürüsü Optimizasyon Algoritması (SCSO), kum kedilerinin doğadaki avlanma ve sürü davranışlarından esinlenerek geliştirilmiş bir meta-sezgisel algoritmadır. Algoritmada her bir sanal kum kedisi, çözüm uzayındaki bir aday çözümü temsil eder ve kum kedilerinin düşük frekanslı sesleri algılayarak avını bulma yeteneği matematiksel modellere aktarılır (Seyyedabbasi ve Kiani, 2022).

SCSO'nun arama stratejisi, çözüm uzayında önce geniş çaplı keşif, ardından en iyi çözümleri sömürme üzerine kuruludur. Bu yaklaşım, kum kedilerinin avı tespit etme ve avına saldırma davranışlarının iki aşamalı doğasını yansıtır (He ve Yuhan, 2022).

Ancak, orijinal SCSO algoritması bazı problemlerde yavaş yakınsama ve yerel optimuma takılma gibi sınırlılıklar gösterebilmektedir. Bu nedenle literatürde algoritmanın performansını artırmak için çeşitli iyileştirme yöntemleri geliştirilmiştir. Örneğin, kaotik diziler ve Lévy uçuşu gibi stratejilerle başlangıç nüfusunun çeşitliliği artırılarak arama etkinliği yükseltilmiş ve yakınsama hızı iyileştirilmiştir (Peng vd., 2025).

Ayrıca, çoklu strateji entegrasyonu kullanılarak algoritmanın global arama kapasitesi güçlendirilmiş ve yerel optima takılma riski azaltılmıştır. SCSO algoritması, adaptif lineer filtre tasarımı, robot yol planlama ve derin öğrenme hiperparametre optimizasyonu gibi farklı mühendislik uygulamalarında da başarıyla uygulanmıştır. Bu uygulamalar, algoritmanın esnek yapısını ve çeşitli problemlere uyarlanabilirliğini göstermektedir (Seyyarer vd., 2024; Venkatesan vd., 2025).

1.2.18 Dikenli Kertenkele Optimizasyon Algoritması (Horned Lizard Optimization Algorithm-HLOA)

Dikenli Kertenkele Optimizasyon Algoritması (HLOA), doğadaki dikenli kertenkelelerin savunma ve kaçma davranışlarından esinlenerek geliştirilmiş bir algoritmadır. Burada her kertenkele, çözüm uzayındaki bir aday çözümü temsil ediyor ve kertenkelelerin avcılardan kaçma, renk değiştirme gibi davranışları matematiksel modellere aktarılıyor (Peraza Vázquez vd., 2024).

Algoritmanın temel işlevi, çözüm uzayında hem geniş çaplı keşif hem de yerel sömürü yaparak en iyi çözümü bulmak. HLOA, karmaşık yapılara sahip olmaması sayesinde uygulaması kolay ve parametre sayısı az. Yapılan testlerde, birçok klasik optimizasyon fonksiyonunda iyi sonuçlar verdiği ve diğer algoritmalarla kıyaslandığında güçlü performans sergilediği görüldü (Peraza Vázquez vd., 2024).

Buna rağmen, özellikle yüksek boyutlu problemler için standart HLOA bazen yerel optimumlara takılabiliyor ve yakınsama hızı yavaş olabiliyor. Bu yüzden araştırmacılar algoritmayı geliştirmek için farklı stratejiler önermiş; nüfus çeşitliliğini artırmak, kaçış mekanizmalarını güçlendirmek ve global arama kapasitesini artırmak gibi yöntemler uygulanmış (Emam vd., 2025).

1.2.19 Doğrusal Olmayan Tabanlı Kaotik Harris Şahinleri Optimizasyon Algoritması (Nonlinear Chaotic Harris Hawks Optimization Algorithm-NCHHO)

Harris Hawks Optimization (HHO) ise, Harris şahinlerinin avlanma stratejilerinden esinlenen başka bir meta-sezgisel algoritmadır. HHO, sürü içi koordinasyon ve sürpriz saldırı davranışlarını kullanarak çözüm uzayında keşif ve sömürü dengesini sağlar ve yüksek boyutlu problemlerde iyi küresel arama performansı sunar (Heidari vd., 2019).

HHO, sınıflandırma, mühendislik optimizasyonu, mikro ağ yönetimi ve nesne segmentasyonu gibi farklı uygulamalarda kullanılır. Bununla birlikte klasik HHO, bazı durumlarda yerel optimumlara takılabilir ve yakınsama hızı yavaşlayabilir; bu nedenle kaotik veya çoklu strateji varyantları geliştirilmiştir (Gezici ve Livatyali, 2022).

Doğrusal Olmayan Tabanlı Kaotik Harris Şahinleri Optimizasyon Algoritması (NCHHO), klasik Harris Hawks Optimization (HHO) algoritmasının eksikliklerini gidermek için geliştirilmiş bir algoritmadır. HHO, doğadaki Harris şahinlerinin avlanma davranışlarını taklit ederek çözüm uzayında hem geniş çaplı keşif hem de iyi çözümleri derinlemesine inceleme yapar (Dehkordi vd., 2021).

NCHHO'nun çıkış noktası, klasik HHO'nun bazı durumlarda global optimuma ulaşmada sınırlı kalmasıdır. Bu problemi çözmek için algoritmada kaotik haritalar ve doğrusal olmayan kontrol parametreleri kullanılır; kaotik haritalar arama yeteneğini artırırken doğrusal olmayan parametreler keşif ile sömürü arasındaki dengeyi iyileştirir. Algoritmanın temel işlevi, çözüm uzayında dengeli bir şekilde arama yaparak en iyi çözümleri bulmaktır. Bu sayede NCHHO, HHO'ya kıyasla özellikle yüksek boyutlu problemlerde daha hızlı yakınsama ve daha tutarlı sonuçlar sunar. NCHHO'nun kullanım alanları da oldukça geniştir. Algoritma, özellikle Nesnelerin İnterneti (IoV), rota planlama, kaynak tahsisi ve veri iletişim optimizasyonu gibi karmaşık mühendislik problemlerinde başarılı bir şekilde uygulanmıştır. Bu uygulamalar, algoritmanın esnek yapısını ve farklı problemlere uyarlanabilirliğini göstermektedir (Dehkordi vd., 2021).

1.2.20 Uyarlamalı Spiral Uçuşlu Serçe Arama Algoritması (Adaptive Spiral Flying Sparrow Search Algorithm-ASFSSA)

Uyarlamalı Spiral Uçuşlu Serçe Arama Algoritması (Adaptive Spiral Flying Sparrow Search Algorithm-ASFSSA), klasik SSA'nın erken yakınsama ve yerel optimum problemlerini aşmak için geliştirilmiş hibrit bir optimizasyon yöntemidir. Algoritma,

adaptif stratejiler, kaotik başlatma ve ek optimizasyon mekanizmalarıyla hem global hem de yerel arama yeteneklerini güçlendirir. Bu sayede çözüm doğruluğu artar ve optimuma ulaşma olasılığı yükselir. Dezavantaj olarak, ek stratejiler hesaplama maliyetini artırabilir ve parametre ayarlamalarına duyarlıdır. ASFSSA, mühendislik optimizasyonları, makine öğrenimi, sınıflandırma problemleri ve enerji sistemleri gibi farklı alanlarda başarıyla uygulanabilir, özellikle yüksek boyutlu veri setlerinde özellik seçimi ve hiperparametre optimizasyonunda etkilidir (Ouyang vd., 2021). Genel olarak, ASFSSA, klasik SSA'nın sınırlılıklarını gidermeye yönelik geliştirilmiş, güçlü ve esnek bir optimizasyon yöntemi olarak öne çıkmaktadır.

1.2.21 Kara Kanatlı Yırtıcı Kuşu (Sungur, Doğan vb.) Algoritması (Black Winged Kite Algorithm-BKA)

Kara Kanatlı Yırtıcı Kuş (Black Winged Kite Algorithm - BKA), kara kanatlı doğanların avlanma ve göç davranışlarından ilham alan bir sürü zekâsı temelli optimizasyon yöntemidir (Ramli ve Fauzi, 2024). Algoritma, süzülme (hovering) ve dalma saldırısı (dive-attack) stratejileriyle hem global keşif (exploration) hem de yerel iyileştirme (exploitation) yapabilmektedir. Basit lider seçimi, az parametre ve iyi global arama kapasitesi sayesinde çeşitli test fonksiyonlarında ve mühendislik problemlerinde rekabetçi sonuçlar verir.

BKA'nın sınırlılıkları arasında yüksek boyutlu problemler için yerel optimumlarda takılma eğilimi ve bazı senaryolarda yavaş yakınsama yer alır. Bu sorunlar, Multi Strategy Enhanced BKA (MBKA) gibi geliştirilmiş versiyonlarla, lider mutasyonu, bilgi farklılığı stratejisi ve adaptif göç güncellemeleri gibi mekanizmalarla büyük ölçüde giderilmiştir. BKA ve varyantları, mühendislik tasarımı, yapısal optimizasyon ve otonom hava araçlarının uçuş rotası planlaması gibi alanlarda etkili bir şekilde uygulanabilmektedir (Cui ve An, 2025).

1.2.22 Altın Çakal Optimizasyonu Algoritması (Golden Jackal Optimization Algorithm-GJO)

Altın Çakal Optimizasyonu Algoritması (Golden Jackal Optimization - GJO) algoritması, 2022 yılında Nitish Chopra ve Muhammad Mohsin Ansari tarafından geliştirilmiş doğa temelli bir optimizasyon yöntemidir (Chopra ve Ansari, 2022). Algoritma, çakalların avlanma ve sürü hâlinde av bulma davranışlarından ilham alır; çakalların hem arama hem de avı kuşatma stratejileri, GJO'nun çözüm bulma mekanizmasına temel oluşturur.

Özellikle karmaşık ve doğrusal olmayan problemlerde etkin olan algoritma, arama uzayını geniş bir şekilde tarayarak olası çözümleri belirler ve ardından bu bölgelerde yoğunlaşarak en uygun çözümü bulur (Chopra ve Ansari, 2022; MathWorks, 2023). Literatürde GJO, mühendislik optimizasyonunda yapısal tasarım ve kontrol parametrelerinin belirlenmesinde, görüntü işleme alanında çok seviyeli eşikleme ve segmentasyon problemlerinde, yüksek boyutlu veri setlerinde özellik seçimi çalışmalarında ve enerji sistemlerinde dağıtık jeneratörlerin yerleşimi ve boyutlandırılması problemlerinde başarılı bir şekilde uygulanmıştır (Chopra ve Ansari, 2022; MDPI, 2024; Researcher Manipl, 2023; MathWorks, 2023). Algoritmanın öne çıkan avantajları, az sayıda parametre ile uygulanabilmesi ve farklı problem türlerine kolayca uyarlanabilmesidir. Bununla birlikte, bazı durumlarda temel versiyonu yerel optimuma takılabilir veya çözüm süresi uzayabilir; bu nedenle literatürde çoğunlukla iyileştirilmiş veya hibrit varyantlar tercih edilmektedir.

1.2.23 Pelikan Optimizasyonu Algoritması (Pelican Optimization Algorithm-POA)

Pelikan Optimizasyonu Algoritması (Pelican Optimization Algorithm - POA), 2021 yılında Abualigah ve arkadaşları tarafından geliştirilmiş doğa temelli bir optimizasyon yöntemidir (Abualigah vd., 2021). Algoritmanın ilham kaynağı, pelikanların avlanma ve balık yakalama davranışlarıdır. Pelikanlar hem su yüzeyinde hem de su altında av arar ve avlarını kuşatma yöntemleriyle yakalar; POA da bu stratejileri çözüm arama ve iyileştirme sürecine uyarlayarak çalışır (Abualigah vd., 2021).

POA, özellikle karmaşık ve çok boyutlu problemlerde iyi çözümler bulmak için tasarlanmıştır. Algoritma, arama uzayını keşfederek potansiyel çözümleri belirler ve daha sonra bu çözümler üzerinde yoğunlaşarak en uygun sonucu elde eder (Abualigah vd., 2021; MathWorks, 2022). Böylece hem global hem de lokal optimizasyon dengesini korur.

Literatürde POA, farklı alanlarda uygulanmıştır. Mühendislik optimizasyonunda yapısal tasarım ve kontrol parametrelerinin belirlenmesinde, enerji sistemlerinde jeneratör yerleşimi ve boyutlandırmasında, makine öğrenmesi ve veri madenciliğinde ise özellik seçimi ve sınıflandırma problemlerinde başarılı sonuçlar verdiği görülmektedir. Algoritmanın öne çıkan avantajları, az sayıda parametre ile çalışabilmesi, farklı problem türlerine kolayca uyarlanabilmesi ve basit yapısıyla etkili sonuçlar sunabilmesidir.

Bununla birlikte, bazı durumlarda temel versiyonun yerel optimuma takılabileceği literatürde belirtilmiştir (Abualigah vd., 2021).

1.2.24 Şempanze Optimizasyonu Algoritması (Chimp Optimization Algorithm-ChOA)

Şempanze Optimizasyonu Algoritması (Chimp Optimization Algorithm-ChOA), şempanzelerin sürü hâlinde avlanma davranışlarından ilham alan bir optimizasyon yöntemidir (Khishe ve Mosavi, 2020). Algoritmada birden fazla “şempanze” bireyinden oluşan bir popülasyon bulunur ve bu bireyler farklı roller üstlenirler (örneğin “attacker”, “barrier”, “chaser” ve “driver”). Her birey, arama uzayındaki bir aday çözümü temsil eder ve algoritmanın mantığı, bu bireylerin birlikte çalışarak hem geniş arama (exploration) hem de çözüm iyileştirme (exploitation) yapmasıdır (Khishe vd., 2022).

Başlangıçta bireyler rastgele konumlarda başlatılır, ardından her iterasyonda konumları sürünün en iyi çözümlerine ve birbirlerine dayanarak güncellenir. Bu süreçte şempanzelerin avı kuşatma, izleme ve baskı kurma stratejileri metafor olarak kullanılır. Bu sayede, karmaşık, doğrusal olmayan ve çok boyutlu optimizasyon problemlerinde nispeten kısa sürede etkili çözümler elde edilebilir (Khishe ve Mosavi, 2020; Khishe vd., 2022).

ChOA, literatürde farklı alanlarda uygulanmıştır. Enerji sistemlerinde mikro şebeke yönetimi ve kaynak dağıtımı, yük dağıtımı ve enerji verimliliğinin artırılması gibi problemler bu algoritmayla optimize edilmiştir. Ayrıca sinyal işleme, sınıflandırma, yol planlama, veri analizi ve zaman penceli rota planlama gibi yüksek boyutlu problemlerde de ChOA, meta-sezgisel yapısı sayesinde klasik yöntemlere göre avantaj sağlamaktadır.

Bununla birlikte, ChOA’nın temel versiyonu bazı durumlarda yerel optimuma takılabilir veya yüksek boyutlu problemlerde yakınsama süresi uzayabilir. Bu nedenle araştırmacılar, algoritmanın performansını artırmak için hibrit veya iyileştirilmiş versiyonlarını önermektedir; bu versiyonlar arama sürecini daha esnek ve adaptif hâle getirir.

1.2.25 Cüce Mangus Arama Algoritması (Dwarf Mongoose Optimization Algorithm-DMOA)

Dwarf Mongoose Optimization Algorithm (DMOA), cüce mangustaların birlikte arama ve yiyecek bulma davranışlarından ilham alan bir meta sezgisel optimizasyon yöntemidir. Algoritma, başlangıçta rastgele çözümlerden oluşan bir popülasyon oluşturur; daha sonra

bu bireyler, mangusta grubunun sosyal yapısı ve grup içi roller dikkate alınarak güncellenir. Grup lideri, keşif yapanlar ve takipçiler gibi roller, arama uzayı içinde dengeli bir “keşif ve iyileştirme” süreci sağlar. DMOA’nın bu yapısı, hem yeni çözüm alanlarını keşfetmeyi hem de mevcut umut vaat eden çözümleri iyileştirmeyi mümkün kılar (Mehmood vd. 2022)

Gerçek uygulamalarda DMOA, parametre tahmini, model tanıma gibi istatistiksel modellenmiş problemlerde başarılı şekilde kullanılmıştır. Örneğin, bir çalışmada parametre tahmininde DMOA’nın hızlı yakınsama, yüksek doğruluk ve güvenilirlik sunduğu gösterilmiştir (Mehmood vd. 2022). Ayrıca, son yıllarda DMOA temelli melez yöntemlerin özellikle yüksek boyutlu veri setlerinde özellik seçimi (feature selection) ve genel küresel optimizasyon problemlerinde etkili sonuç verdiği, hem arama alanı genişliği hem de çözüm kalitesi açısından rekabetçi olduğu rapor edilmiştir. (Al Shourbaji vd., 2023)

DMOA’nın en önemli avantajları arasında, yalın bir yapıya sahip olması (az parametre ile çalışabilmesi) ve farklı problem tiplerine kolay uyum gösterebilmesi yer alıyor. Ancak temel DMOA’nın bazı problemlerde çeşitlilik (diversity) kaybı veya yerel en iyi çözümlere takılma riski olabildiği; bu yüzden literatürde genellikle “geliştirilmiş” ya da “hibrit” DMOA versiyonlarının tercih edildiği görülüyor (Moustafa vd., 2023).

1.2.26 Popülasyon Tabanlı Çok Amaçlı Güve Algoritması (Population based Multi Objective Moth Algorithm - PMOMA)

Popülasyon Tabanlı Çok Amaçlı Güve Algoritması (Population based Multi Objective Moth Algorithm - PMOMA), çok amaçlı optimizasyon problemlerini çözmek için geliştirilmiş bir meta-sezgisel algoritmadır. Algoritma, güve sürüsünün davranış modelinden esinlenerek çözüm uzayında etkili bir arama stratejisi sağlar; burada güveler aday çözümleri temsil eder ve popülasyon tabanlı arama süreci, farklı amaçları aynı anda optimize ederken çeşitlilik ve yakınsama arasında denge kurmayı hedefler (Sharifi vd., 2021).

PMOMA’nın temel mantığı, popülasyon içindeki bireylerin Pareto ön cephesine yakın ve çeşitli çözümler üretmesini sağlamaktır. Bu sayede algoritma, çok amaçlı optimizasyon problemlerinde Pareto dağılımının çeşitliliğini ve yakınsamasını artırmayı amaçlar. Popülasyon tabanlı yapısı sayesinde, farklı amaç fonksiyonları arasında optimal dengeye

ulaşmak için arama sürecinde çeşitli çözümleri aynı anda değerlendirebilmek mümkün olur (Sharifi vd., 2021).

Algoritmanın en önemli avantajı, küresel arama (exploration) ve yerel iyileştirme (exploitation) yeteneklerini dengeli bir şekilde sürdürmesidir. Bu özellik, PMOMA'nın gerçek dünya optimizasyon problemlerinde gerçekçi ve çeşitlendirilmiş Pareto ön cephesi çözümleri üretmesini sağlar. Bu nedenle PMOMA, mühendislik, yapay zekâ ve karar verme problemleri gibi çok amaçlı optimizasyon gerektiren alanlar için güçlü bir seçenek olarak değerlendirilebilir (Sharifi vd., 2021).

1.2.27 Zebra Optimizasyon Algoritması (Zebra Optimization Algorithm - ZOA)

Zebra Optimizasyon Algoritması (Zebra Optimization Algorithm - ZOA), doğadaki zebraların beslenme ve savunma davranışlarından esinlenerek geliştirilmiş bir meta-sezgisel optimizasyon yöntemidir (Dehghani vd., 2022).

Algoritmada her zebra, çözüm uzayında bir aday çözümü temsil eder ve sürü davranışlarının matematiksel modellere aktarılmasıyla hem geniş çaplı arama (exploration) hem de yerel iyileştirme (exploitation) yapılabilir. ZOA'nın temel amacı, çözüm uzayında optimum veya yakın optimum çözümler bulmak için rastgele başlayan popülasyonları belirli kurallar çerçevesinde güncelleyerek en uygun çözüm bölgelerine yönlendirmektir. Algoritmanın avantajları arasında basit yapısı, az sayıda parametreye sahip olması ve çeşitli test fonksiyonlarında rekabetçi sonuçlar vermesi yer alır (Dehghani vd., 2022).

Bununla birlikte, ZOA özellikle yüksek boyutlu veya karmaşık problemler için yerel optimumlara takılabilir ve yakınsama hızı yavaşlayabilir; bu nedenle literatürde kaotik haritalar veya çoklu strateji mekanizmalarıyla geliştirilmiş versiyonları da çalışılmıştır (Liu, Y., 2025).

ZOA ve geliştirilmiş varyantları, mühendislik tasarımı, mikro ağ planlama ve video sahteciliği tespiti gibi farklı uygulamalarda etkili bir şekilde kullanılmaktadır (Oke vd., 2025).

Zebra Algoritmasının matematiksel altyapısı, d ' problemin boyut sayısı (karar değişkeni sayısı), Ω çözüm uzayı (tüm mümkün çözümler kümesi), L_j ve U_j , j . boyut için alt ve üst sınırlar, $f(z)$ amaç (uygunluk) fonksiyonu (minimize edilmek istenen fonksiyon) göstermek üzere d -boyutlu sürekli bir arama uzayında tanımlanan $\min_{z \in \Omega} f(z)$ optimizasyon problemini çözmeyi amaçlar. Burada:

$$\Omega = \{z \in R^d | L_j \leq x_j \leq U_j\}$$

kapalı ve sınırlı bir kümedir. Algoritma, her biri bir aday çözümü temsil eden N adet zebra bireyinden oluşan bir popülasyon ile çalışır. Her zebra,

$$\hat{Z}_i = (\hat{Z}_{i1}, \hat{Z}_{i2}, \dots, \hat{Z}_{id})$$

vektörüyle ifade edilir. Başlangıç popülasyonu, her boyut için düzgün dağılımlı rastgele değişkenler kullanılarak;

$$\hat{Z}_{ij}^0 = L_j + r_{ij}(U_j - L_j), \quad r_{ij} \sim U(0,1)$$

eşitliği ile oluşturulur. Böylece başlangıçta arama uzayı homojen biçimde örneklenmiş olur. Başlatmanın ardından her bireyin uygunluk değeri $f(\hat{Z}_i^0)$ hesaplanır ve en iyi uygunluğa sahip birey lider zebra olarak tanımlanır.

$$\hat{Z}^{best}(t) = \min_{1 \leq i \leq N} f(\hat{Z}_i^t)$$

Bu birey sürünün yönünü belirleyen çekim merkezi gibi davranır. Algoritmanın keşif (exploration) aşamasında zebralar lideri takip ederek konumlarını günceller. Bu davranış matematiksel olarak;

$$\hat{Z}_i^{t+1} = X_i^t + r_1(\hat{Z}^{best}(t) - r_2 \hat{Z}_i^t)$$

şeklinde modellenir. Burada $r_1, r_2 \sim U(0,1)$ bağımsız rastgele değişkenlerdir. Bu ifade, bireyin mevcut konumundan lider yönüne doğru stokastik bir hareket yaptığını gösterir. Çarpanların rastgeleliği, arama uzayında çeşitlilik sağlayarak küresel keşfi destekler.

1.2.28 Coati Optimizasyon Algoritması (COA)

Coati Optimizasyon Algoritması (COA), doğadaki coatilerin avlanma ve kaçış davranışlarından ilham alınarak geliştirilmiş bir meta-sezgisel optimizasyon yöntemidir. Algoritma, popülasyon tabanlıdır ve coatilerin davranışlarını matematiksel modellere dönüştürerek çözüm uzayında hem genel keşif (exploration) hem de yerel iyileştirme (exploitation) yapar. COA, rastgele oluşturulan aday çözümleri sürü davranışına benzer şekilde güncelleyerek optimum veya yakın optimum çözümler bulur. Bu yapı, algoritmanın karmaşık ve yüksek boyutlu problemlerde rekabetçi performans göstermesini sağlar (Dehghani vd., 2023).

Algoritmanın kullanım alanları geniştir; COA, enerji yönetimi, mikro ağ optimizasyonu, çok amaçlı tasarım problemleri ve veri işleme uygulamaları gibi alanlarda başarılı şekilde

uygulanabilir. Örneğin, hibrit enerji sistemlerinde maliyet ve performans optimizasyonunda etkili sonuçlar vermiştir (Kumar ve Bhasker, 2025).

Bununla birlikte, COA'nın klasik hâli bazı durumlarda popülasyon homojenliği ve rastgele stratejiye bağlılık nedeniyle yerel optimumlara takılabilir. Bu nedenle literatürde, çoklu strateji adaptif COA gibi geliştirilmiş versiyonlar önerilmiştir; bu varyantlar, Lévy uçuşu ve doğrusal olmayan adım büyüklükleri gibi mekanizmalarla global arama kabiliyetini artırır (Wu vd., 2025).

Coati Optimizasyon Algoritması (COA), d ' boyutlu sürekli bir optimizasyon aşağıdaki şekilde ifade edilir:

$$\min_{c \in \Omega} f(c)$$

Burada $c = (c_1, c_2, \dots, c_d) \in \mathbb{R}^d$ karar değişkeni vektörü $f = \mathbb{R}^d \rightarrow \mathbb{R}$ amaç fonksiyonu ve $\Omega = \{c | l_j \leq c_j \leq U_j\}$ çözüm uzayını temsil eder. COA, bu problemi çözmek için N adet aday çözümden oluşan bir popülasyon kullanır. Popülasyon matrisi;

$$C = \begin{bmatrix} C_1^t \\ C_2^t \\ \vdots \\ C_N^t \end{bmatrix} \text{ şeklinde tanımlanır ve her bir birey } C_i^t = (c_{i1}^t, c_{i2}^t, \dots, c_{id}^t) \text{ biçiminde gösterilir.}$$

Başlatma aşamasında başlangıç popülasyonu rastgele oluşturulur. Her bir karar değişkeni için başlangıç konumu aşağıdaki eşitlik ile belirlenir:

$$c_{ij}^0 = l_j + r_{ij}(U_j - l_j)$$

Burada $r_{ij} \sim U(0,1)$ düzgün olasılık dağılımlı rastgele sayıdır. Bu işlem, başlangıçta çözüm uzayının geniş bir bölümünün örneklenmesini sağlar. Başlangıç uygunluk değerleri $f(C_i^0)$ hesaplanarak en iyi çözüm;

$$C_{best}^t = \arg \min_{C_i^t} f(C_i^t)$$

şeklinde belirlenir. Keşif (exploration) aşamasında bireylerin konumu, en iyi çözüm doğrultusunda ancak rastgelelik içerecek şekilde güncellenir. Bu güncelleme aşağıdaki diferansiyel hareket modeli ile ifade edilir:

$$C_i^{t+1} = C_i^t + \gamma r_1 (C_{best}^t - |C_i^t|)$$

Burada $r_1 \sim U(0,1)$ ve γ keşif katsayısını temsil eder. Mutlak değer işlemi, hareketin yönünü çeşitlendirerek arama uzayında daha geniş bir yayılım sağlar. Bu model, algoritmanın küresel arama kapasitesini artırmayı amaçlar.

Sosyal etkileşim aşamasında bireyler popülasyon ortalamasına doğru yönlendirilir. Popülasyon merkezi:

$$C_{mean}^t = \frac{1}{N} \sum_{i=1}^N C_i^t$$

şeklinde hesaplanır. Konum güncelleme denklemi ise;

$$C_i^{t+1} = C_i^t + \delta r_2 (C_{mean}^t - C_i^t)$$

Şeklinde. Burada ise $r_2 \sim U(0,1)$ ve δ yakınsama katsayısını gösterir. Bu mekanizma, popülasyonun kontrollü biçimde yoğunlaşmasını sağlar.

Sömürü (exploitation) aşamasında yerel arama gerçekleştirilir. Bu aşamada en iyi ve en kötü çözümler arasındaki fark dikkate alınır. En kötü çözüm:

$$C_{worst}^t = \arg \max_{C_i^t} f(C_i^t)$$

şeklinde belirlenir. Yerel güncelleme modeli:

$$C_i^{t+1} = C_{best}^t + \theta r_3 (C_i^t - C_{worst}^t)$$

eşitliği ile verilir. Burada ise $r_3 \sim U(-1,1)$ ve θ yerel arama katsayısıdır. Bu ifade, çözümün en iyi nokta çevresinde hassas şekilde ayarlanmasını sağlar.

$$c_{ij}^{t+1} = \begin{cases} L_j & \text{eğer } c_{ij}^{t+1} < L_j \\ U_j & \text{eğer } c_{ij}^{t+1} > U_j \\ c_{ij}^{t+1} & \text{aksi halde} \end{cases}$$

Algoritma, $t = 1, 2, \dots, T_{max}$ iterasyonları boyunca devam eder ve son çözüm $C_{best}^{T_{max}}$ olarak elde edilir. COA'nın hesaplama karmaşıklığı popülasyon büyüklüğü N , problem boyutu d' ve maksimum iterasyon sayısı T_{max} olmak üzere;

$$O(N \cdot d' \cdot T_{max})$$

mertebedindedir. COA ilk olarak Seyedali Mirjalili ve arkadaşları tarafından önerilmiş olup sürekli optimizasyon problemlerinde test fonksiyonları üzerinde değerlendirilmiştir (Mirjalili vd., 2023).

1.2.29 Walrus Optimizasyon Algoritması (WaOA)

Walrus Optimizasyon Algoritması (WaOA), deniz memelisi olan morse (walrus) davranışlarından esinlenerek geliştirilmiş yeni bir meta-sezgisel optimizasyon yöntemidir. Bu algoritma, morse davranışlarını; beslenme, göç etme, avcılardan kaçma ve düşmanla mücadele etme sergileyerek ve matematiksel modellere dönüştürerek çözüm uzayında etkili bir arama süreci yürütür. WaOA'nın işlevi, üç ana aşama üzerinden ilerler: keşif (exploration) ile geniş alanları tarama, göç etme ile farklı çözüm bölgelerine ulaşma ve sömürü (exploitation) ile iyi bulunan çözümleri daha da iyileştirmedir. Bu yapı sayesinde algoritma, optimizasyon problemlerinde hem yeni çözümler bulur hem de mevcut çözümleri derinlemesine iyileştirebilir. Literatürde yapılan değerlendirmeler, WaOA'nın benchmark fonksiyonları üzerinde güçlü bir keşif-sömürü dengesi kurduğunu ve on tanınmış meta-sezgisel algoritmayla karşılaştırıldığında rekabetçi veya üstün performans gösterdiğini ortaya koymuştur (Trojovský ve Dehghani, 2023).

WaOA'nın kullanım alanları da geniştir; bu algoritma mühendislik tasarımı, kablosuz sensör ağlarının kapsama optimizasyonu ve mikro ağ kontrolü gibi gerçek dünya optimizasyon problemleri için uygulanmıştır ve başarılı sonuçlar elde etmiştir. Bu algoritmanın hâlihazırda bazı sınırlamaları bulunmasına rağmen WaOA, genel olarak dengenin iyi kurulabildiği, pratik uygulamalarda kullanılabilir bir optimizasyon yöntemi olarak değerlendirilmektedir.

1.2.30 Dijkstra Algoritması

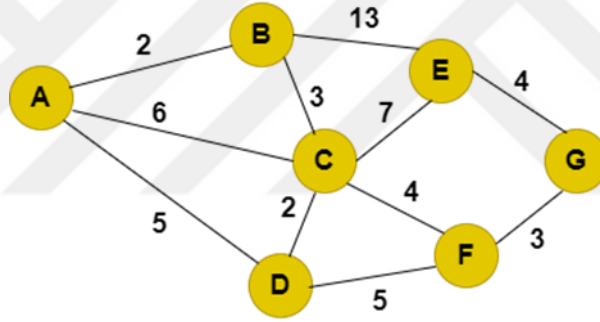
En kısa yol hesaplarında kullanışlı bir diğer yöntemlerden biri olan Dijkstra Algoritması, başlangıç noktası belli olan ve belirli bir hedefe en kısa yol kat ederek ulaşmayı hedefleyen kullanışlı bir algoritmadır. 1959 yılında bir bilgisayar bilimcisi olan Edsger Dijkstra tarafından geliştirilmiş olup, en kısa mesafeyi keşfetmek için, olası çözümler içerisinde bir düğümden başka bir düğüme ilerlerken en iyi çözümü referans alarak bir sonraki düğüme ilerler ve her adımda bir sonraki düğüm için aynı işlemi tekrarlayarak, hedef noktaya ulaşır (Özdemir vd., 2021).

Kullanışlı bir algoritma olması dışında dezavantajı ise bütün ayrıt uzunluklarını negatifliği gözardı ederek pozitif varsayarak ilerlemektedir. Düğümleri ilk olarak geçici etiketleyerek her düğümün başlangıca göre en kısa yolunu tespit etmektedir ve ardından kalıcı olarak düğümleri etiketleyerek en kısa mesafe hesabı yaparak güzergâhlarını hızlı

bir şekilde tayin etmektedir. Dijkstra algoritması bilgisayar ağıları, lojistik alanlarında hedefe en kısa süre ve en uygun şekilde hedefe ulaşmada kullanılır (Özdemir vd., 2021).

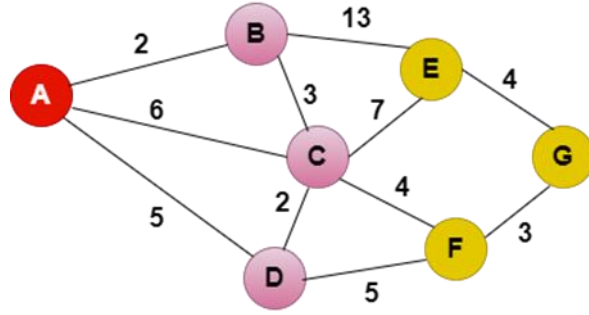
Algoritma aşamalarının basamakları şu şekildedir:

1. Basamak: Geçerli düğümler kümesi içerisinde belirlenen başlangıç noktası, kalıcı düğüm olarak belirlendikten sonra çözüm için, bu düğüme yakın komşu düğümler kümesi, belirlenir.
2. Basamak: Geçerli düğümler kümesinden, erişilebilir düğümler kümesine giden en kısa yollar hesaplanır, hedefe ulaşıldıysa bitirilir. Ulaşılamadıysa bu düğüm saklanıp 3. Basamağa geçilir.
3. Basamak: En kısa yol ile ulaşılan düğüm geçerli kümeye dahil edilir.
4. Basamak: Geçerli kümenin elemanı olan ulaşılabilir düğümler tekrar bulunur ve 2. Basamağa tekrar dönlür.



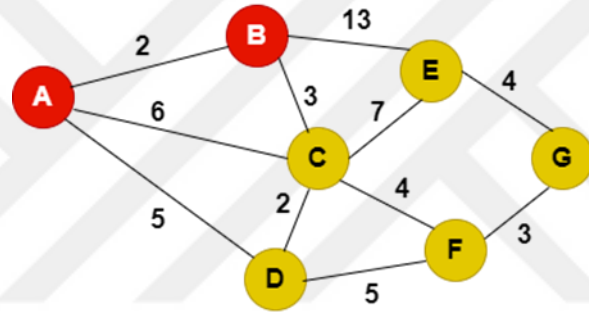
Şekil 1.14 Dijkstra algoritmasının anlatımı için kullanılan çizge

Şekil 1.14 'te verilen ağı kullanarak Dijkstra algoritması ile en kısa yol bulmak için düğümler arası mesafeler; A düğümünden B düğüme olan uzaklık 2 birimken, B düğümünden E düğüme olan uzaklık 13 birim verilmiştir. Bu örnekte amaç A düğümünden G düğüme en kısa yolu bulmaktır.



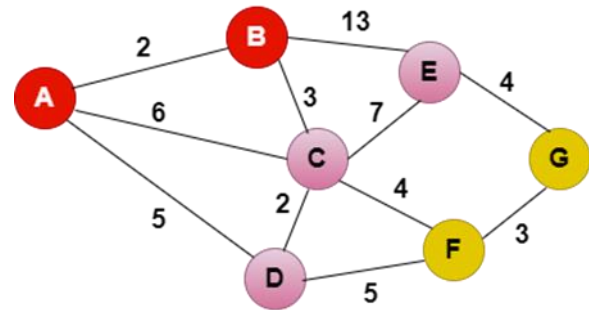
Şekil 1.15 Dijkstra algoritması 1. adım görseli

Şekil 1.15 ‘te görüldüğü üzere 1. Basamakta bahsedildiği gibi A düğümü geçerli düğümler kümesine kalıcı olarak eklenmiştir. A düğümünden erişilebilen komşu düğümler kümesi “B, C ve D” olarak tespit edilmiştir.



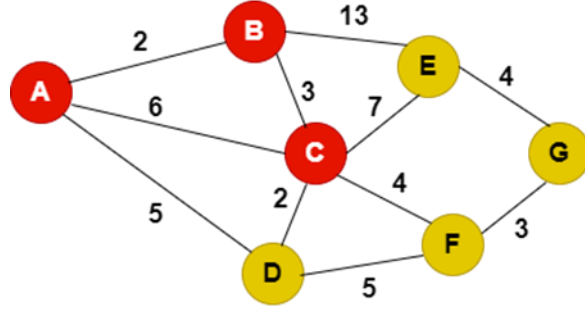
Şekil 1.16 Dijkstra algoritması 2. adım görseli

Şekil 1.16 ’da Adım 1’de tespit edilen erişilebilir düğümler kümesindeki B düğümü A düğümüne en yakın mesafede olduğu için saklanır. B düğümü geçerli düğümler kümesine dahil edilir.



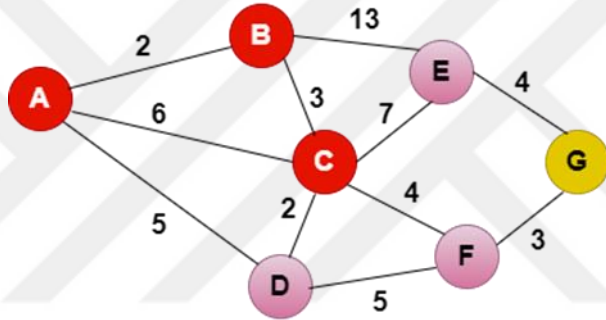
Şekil 1.17 Dijkstra algoritması 3. adım görseli

Geçerli düğümler kümesinde yer alan A ve B düğümlerinden erişilebilir düğümler Şekil 1. 17 ’de verilmiştir. Bunlar “C, D ve E” düğümleridir.



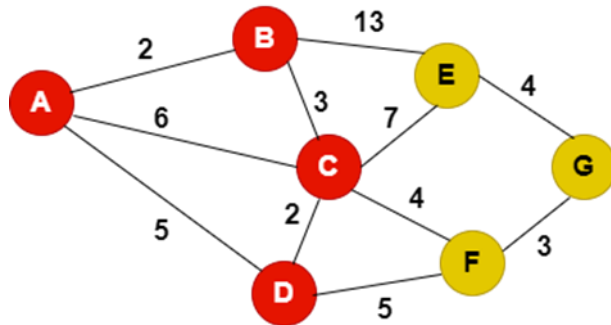
Şekil 1.18 Dijkstra algoritması 4. adım görseli

Şekil 1.18 'de görüldüğü üzere Geçerli düğümler kümesinden erişilebilir düğümler kümesine başlangıçtan olan en kısa yol B-C düğümleri arasındadır. C düğümü geçerli düğümler kümesine dahil edilir.



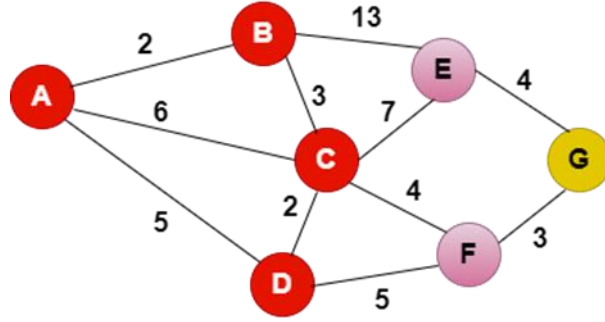
Şekil 1.19 Dijkstra algoritması 5. adım görseli

Şekil 1.19 'da geçerli düğümler kümesi "A, B ve C", erişilebilir düğümler kümesi "D, E ve F" düğümlerinden oluşmaktadır.



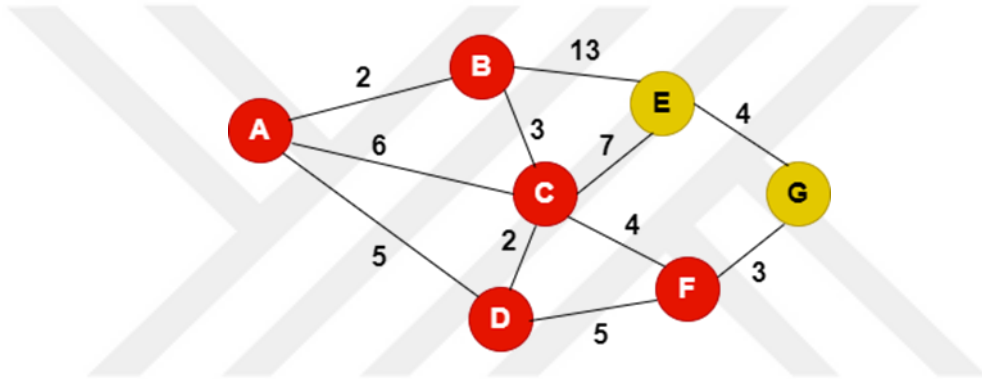
Şekil 1.20 Dijkstra algoritması 6. adım görseli

Şekil 1.20 'de başlangıç A düğümünden erişilebilir düğümler kümesine en kısa yol A-D arasındadır. D düğümü geçerli düğümler kümesine eklenmiştir.



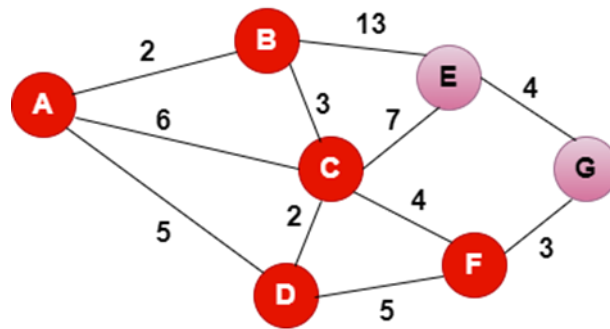
Şekil 1.21 Dijkstra algoritması 7. adım görseli

Şekil 1.21 'de gözlemlenildiği gibi geçerli düğümler kümesi "A, B, C ve D", erişilebilir düğümler kümesi "E ve F" düğümlerinden oluşmaktadır.



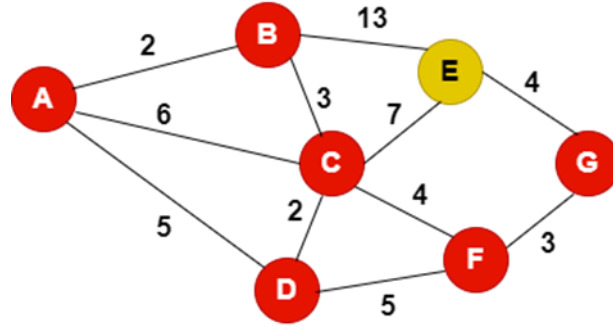
Şekil 1.22 Dijkstra algoritması 8. adım görseli

Şekil 1.22' de F düğümü geçerli düğümler kümesine eklenmiştir.



Şekil 1.23 Dijkstra algoritması 9. adım görseli

Şekil 1.23 'te Geçerli düğümler kümesinden erişilebilir düğümler kümesine en kısa yol F-G düğümü arasındadır.



Şekil 1.24 Dijkstra algoritması 10. adım görseli

Şekil 1.24 'te G düğümü kalıcı düğümler kümesine eklenmiştir. Hedef düğüm olan G'ye ulaşıldığı için algoritma sonlandırılmıştır.

Başlangıç noktası olan A düğümünden hedeflenen G düğümüne giderken saklanan ayrıtlar bağlantı çizgileri ile gösterilmiştir. En kısa yol A-B-C-F-G olup, toplam mesafe 12 birimdir. A-D ayrıtı saklanmış ancak D düğümünden G düğümüne gidilebilecek bir ayrıtı saklanmadığı için bu yol kullanılmamıştır.

Tablo 1.1 Dijkstra algoritması sonucu

Geçerli Düğümler	Erişilebilir Düğümler	Saklanan Ayrıt	Toplam Uzaklık
A	B,C,D	(A,B)	2
A,B	C,D,E	(B,C)	2+3=5
A,B,C	D,E,F	(A,D)	5
A,B,C,D	E,F	(C,F)	5+4=9
A,B,C,D,E	E,G	(F,G)	9+3=12

Tablo 1.1 'de her bir adımda ki geçerli ve erişilebilen düğümler kümesi ile saklanan ayrıt ve başlangıç noktasından saklanan ayrıtın sonuna kadar olan toplam uzaklık bilgileri yer almaktadır.

1.3 Hibrit Algoritmalar

Meta sezgisel algoritmalar, bir problemi kısa sürede optimale yakın olarak çözebilmektedir. Meta sezgisel algoritmaların güçlü yanlarının yanı sıra, zayıf yönleri de vardır. Bu nedenle, meta sezgisel algoritmaların güçlü yönlerini bir araya getiren hibrit algoritmalar geliştirilmiştir (Garip, Z., 2018).

Hibrit algoritmalarından faydalananak, metasezgisel algoritmaların tek başına sağlayamadığı en iyi çözümü, birden fazla metasezgisel algoritmanın iyi yönleri birleştirilerek en ideal çözümü elde etmek için yeni bir hibrit algoritma elde edilir ve bu yeni algoritma tanımlanır. Optimizasyon algoritmalarının problem çözme başarısını arttırmak için eksik yönlerini, diğer algoritmaların güçlü yeteneklerini kullanarak daha iyi bir çözümü daha kısa sürede ve az maliyetle elde eden birçok hibrit algoritma optimize edilerek literatüre kazandırılmıştır. Bunlardan bazıları sırasıyla açıklanmıştır.

1.3.1 Genetik-Simpleks Hibrit Algoritması

Genetik algoritma, 1950 yıllarından beri birçok bilim insanı tarafından araştırılırsa, GA kavramı John Holland ve öğrencileri tarafından toparlanıp, "Adaptation in Natural and Artificial Systems" adlı kitabında detaylı olarak yayınlanmıştır. Öğrencilerinden Goldberg, GA üzerine yaptığı çalışmalar sayesinde, genetik algoritmanın tanınmasında çok önemli katkılarda bulunmuştur. Daha önceki başlıklarda daha detaylı bahsettiğimiz GA, popülasyon tabanlı olup, topluluk içerisinde evrim sürecini taklit ederek elde edilen mevcut en iyi çözümler arasından da daha iyisini seçmeyi hedefleyen, türevden bağımsız olarak gen, kromozom, popülasyon ve kuşak sayısı gibi evrim ilkelerinin bilgisayar ortamına uyarlanmasıyla geliştirilen stokastik optimizasyon algoritmasıdır (Goldberg, D. E., 1989).

GA, başlangıç topluluğunun (popülasyonunun) belirlenmesi, uygunluk değerinin hesaplanması, seçim, çaprazlama, mutasyon gibi genetik operatörlerinin uygulanması, yeni nesil topluluklarının (popülasyonlarının) elde edilmesi ve son olarak durdurma koşulu kontrolü ile adımların birbirini takip ettiği ve en iyi sonuca ulaşana kadar devam ettiği, evrimsel sürecin taklit edildiği döngüden oluşur. Yeni çözümlerin bir önceki nesil çözümlere göre daha iyi olması, genetik operatörlerin seçimi, popülasyon büyüklüğü, kuşak sayısı seçimi, model parametrelerin tahminini etkilkeyen, belirleyici GA ayarlanabilir parametreleridir. Bu ayarlanabilir parametre kombinasyonları içinden seçim yapmak uzun zaman kayıplarına sebep olacağı için daha kısa zaman ve az emek gerektiren iddialı bir algoritmaya ihtiyaç duyulmuştur (Türkşen ve Akgün, 2018).

Nelder – Mead Simplex (NMS) algoritması, 1965 yılında Nelder ve Mead tarafından tanıtılan ve en eski türevsiz optimizasyon yöntemlerinden biri olup, doğrusal olmayan modellerin optimizasyonunda kullanılır. Tek nokta aramalara dayalı, türeve ihtiyaç duymayan stokastik optimizasyon algoritması olan Nelder – Mead Simplex (NMS)

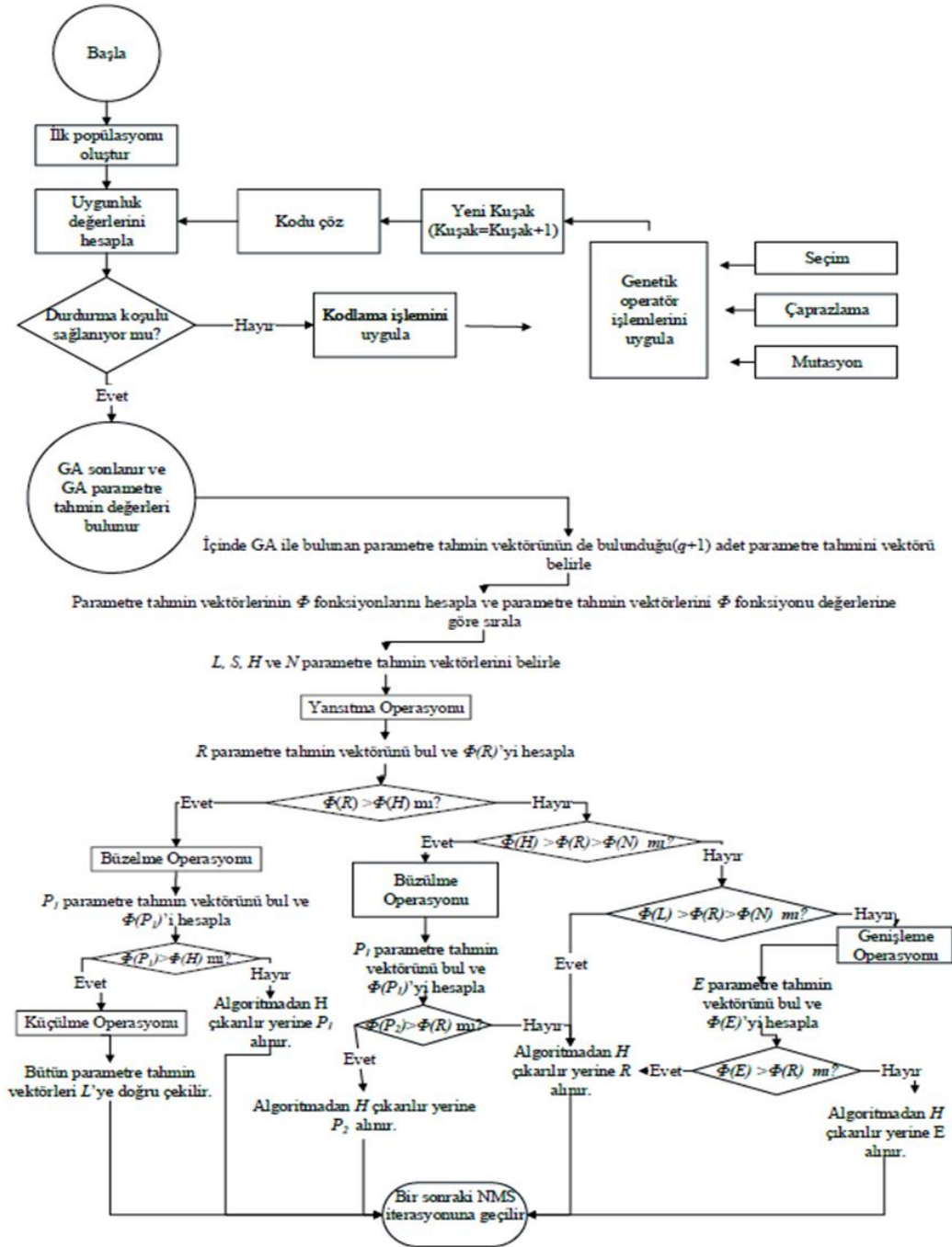
algoritması, Spendley ve arkadaşları tarafından tanıtılan simpleks esaslı arama algoritmasındaki yansıma ve küçülme operasyonlarına, Nelder ve Mead tarafından eklenen genişleme ve büzülme operasyonları geliştirilerek, doğrusal olmayan modellerin optimizasyonu için kullanılır biçime getirilmiştir (Türkşen ve Akgün, 2018).

Yerel bilgidен faydalananarak en yakın optimum noktaya yönelir. Yerel minimum noktasını arayan yöntem, çok boyutlu optimizasyon problemlerinde kullanılabilir. Gradyan yöntemleri gibi türev kullanarak, bir fonksiyon üzerinde hareket etmek zorunda değildir. Simpleksin köşelerinde MF hesaplayarak, maliyeti en yüksek olan, en kötü köşeyi yeni bir köşe ile değiştirir. Bu değiştirme işlemi dört temel uygulama ile gerçekleştirilir: yansıma, genişleme, kısılma ve büzülme. Bu işlem, simpleks yeteri kadar küçülüp yerel minimum noktası bulunana kadar tekrarlanır (Arslan, H., 2020).

Problem çözümünde, Nelder – Mead Simplex Algoritması (NMS) ile Genetik Algoritma (GA)’nın bireysel olarak kullanıldıklarında zayıf kalan yönleri yerine, iki algoritmanın avantajlı yönleri birleştirilerek bir arada kullanılması ile oluşturulan Genetik-Simpleks Hibrit Algoritması, daha iyi sonuçlar elde etmek için tasarlanıp geliştirilmiştir. İlk olarak Genetik Algoritma(GA) ile global arama yapılarak, geniş arama uzayı daraltılır. Ardından Genetik Algoritma(GA) ile bulunan model parametre tahminleri Nelder – Mead Simplex Algoritması (NMS) için model parametrelerin başlangıç değerleri olarak kabul edilerek en iyi çözümün bulunduğu bölgeye indirgenir. Şekil 14 ’te, çalışmada kullanılan Genetik-Simpleks hibrit algoritmasının akış diyagramı verilmiştir (Türkşen ve Akgün, 2018).

q sayıda parametreye sahip bir doğrusal olmayan regresyon modelinde, öncelikli olarak seçilen $q+1$ tane parametre tahmin vektörünün amaç fonksiyon değerlerinin bulunması ile işleme başlanır. Parametre tahminleri, ϕ amaç fonksiyonundaki değerlerine göre sıralanır. Sıralama işleminde, en düşük amaç fonksiyonu değerine sahip parametre tahmin vektörü (L), en düşük ikinci amaç fonksiyonu değerine sahip parametre tahmin vektörü (S), en yüksek amaç fonksiyonu değerine sahip parametre tahmin vektörü (H) ve en yüksek ikinci amaç fonksiyonu değerine sahip parametre tahmin vektörü (N) belirlenir. Sıralama işlemi sonrasında Nelder-Mead Simplex (NMS) algoritmasında sırasıyla yansıma, genişleme, büzülme ve küçülme operasyonları uygulanır. Bu operasyonlara ilişkin NMS ayarlanabilir parametreleri, $\alpha (> 0)$, $\beta (> 1)$, $\gamma (0 < \gamma < 1)$, $\sigma (0 < \sigma < 1)$ ile tanımlıdır. Yapılan çalışmalarda genellikle bu parametre değerleri, $\alpha = 1, \beta = 2, \gamma = 1/2, \sigma = 1/2$ olacak şekilde seçilir(Gurson, A.P., 2000). Her yineleme sonucunda algoritmada bulunan $q + 1$ parametre tahmin vektörü ϕ fonksiyonu

değerine göre yeniden sıralanır ve L, H, S, N parametre tahmin vektörleri güncellenir. Böylece, NMS algoritması, bu operasyonlar ile adım adım ilerleyerek belirlenmiş iyi bir başlangıç çözüm değeri ile amaç fonksiyonu değerini en küçük yapacak parametre tahminine yakınsar. Algoritma, amaç fonksiyon değerlerinin birbirine çok yaklaşması veya belirli bir yinleme sayısına ulaşılması durumlarından birinin gerçekleşmesi yoluyla sonlandırılır (Türkşen ve Akgün, 2018).



Şekil 1.25 Genetik-Simpleks hibrit algoritması akış diyagramı. (Türkşen ve Akgün, 2018).

1.3.2 Geliştirilmiş Hibrit Zebra Algoritması (Zebra Hunt Optimization-ZHO)

Zebra Hunt Optimization (ZHO), literatürde bağımsız bir algoritma olarak yer almasa da, Zebra Optimization Algorithm (ZOA) tabanlı hibrit yaklaşımlar şeklinde kullanılır. ZOA, zebraların sürü hâlinde yiyecek arama ve savunma davranışlarını modelleyen bir popülasyon tabanlı meta-sezgisel optimizasyon yöntemidir (Trojovská vd., 2022).

Algoritma, popülasyondaki zebra bireylerinin hem geniş kapsamlı keşif (exploration) hem de yerel iyileştirme (exploitation) davranışlarıyla optimum veya yakın optimum çözümler bulmasını sağlar; aday çözümler sürü davranışına benzer şekilde güncellenir (Trojovská vd., 2022).

Avlanma (hunt) aşamasında sistemde bir avcı baskısı matematiksel olarak geliştirilerek modellenir. Avcı konumu genellikle lider etrafında rastgele bir sapma ile tanımlanır:

$$\hat{Z}^{pred}(t) = \hat{Z}^{best}(t) + \beta \cdot \varepsilon$$

Burada $\varepsilon \sim N(0,1)$ çok değişkenli normal dağılımı, β ise avcı etkisini belirleyen ölçek parametresidir. Zebralar avcıdan uzaklaşmak için konumlarını,

$$\hat{Z}_i^{t+1} = \hat{Z}_i^t + \hat{S}(\hat{Z}_i^t - \hat{Z}^{pred}(t))$$

eşitliği ile günceller. $\hat{S}$ kaçış katsayısıdır ve genellikle rastgele veya zamana bağlı seçilir. Bu terim, bireyin bulunduğu noktadan avcı yönünün tersine doğru itici bir kuvvet uygular. Böylece yerel arama (exploitation) mekanizması oluşur.

Keşif ve avlanma aşamaları arasında denge kurmak için zamana bağlı bir adaptif parametre tanımlanır:

$$a(t) = 2 \left(1 - \frac{t}{T} \right)$$

burada T maksimum iterasyon sayısıdır. Erken iterasyonlarda $a(t)$ büyük olup keşfi artırır; son iterasyonlarda küçülerek sömürüyü güçlendirir. Genel güncelleme denklemi,

$$\hat{Z}_i^{t+1} = a(t)X_i^{explore} + (1 - a(t))\hat{Z}_i^{hunt}$$

şeklinde yazılabilir. Böylece sistem zamanla küresel aramadan yerel iyileştirmeye geçiş yapar.

Her iterasyon sonunda sınır kontrolü uygulanır. Eğer herhangi bir bileşen sınırları aşarsa,

$$\hat{z}_{ij}^{t+1} = \begin{cases} L_j, & \hat{z}_{ij}^{t+1} < L_j, \\ U_j, & \hat{z}_{ij}^{t+1} > U_j, \end{cases}$$

şeklinde projeksiyon yapılır. Böylece tüm çözümler Ω kümesi içinde kalır. Stokastik yapı açısından algoritma,

$$\hat{Z}^{t+1} = F(\hat{Z}^t, \xi^t)$$

biçiminde yazılabilir; Burada, $\hat{Z}^t$, t anındaki tüm bireylerin konum vektörlerini; ξ^t ise o adımda kullanılan rastgele sayı vektörünü temsil etmektedir (Mirjalili, 2016). Bu yapı, algoritmanın Markov özelliği taşıdığını gösterir; yani bir sonraki durum yalnızca mevcut duruma ve o anki rastgeleliğe bağlıdır ve geçmiş iterasyonlar dikkate alınmaz (Kennedy ve Eberhart, 1995).

Algoritmanın rastgele yapısı nedeniyle bir iterasyonda global optimuma ulaşım garanti edilemez; ancak, yeterince uzun süre çalıştırıldığında ve popülasyon çeşitliliği korunduğunda, algoritma yüksek olasılıkla optimuma yakınsar (Yang, 2014). Bu olasılıksal yakınsamanın gerçekleşebilmesi için, arama uzayının her bölgesi erişilebilir olmalı (ergodiklik) ve popülasyon yeterli çeşitliliğe sahip olmalıdır (Mirjalili, 2016).

Özetle, bu ifade algoritmanın “hafızasız, rastgele ama kontrollü bir süreç” olarak işlediğini ve uygun koşullarda global optimuma yaklaşabildiğini göstermektedir. Bu yaklaşım, özellikle doğadan esinlenen sürü algoritmalarında (örn. Particle Swarm Optimization, PSO) yaygın olarak kullanılan bir matematiksel çerçevedir (Kennedy ve Eberhart, 1995).

Son olarak, hesaplama karmaşıklığı her iterasyonda N birey ve d' boyut için uygunluk değerlendirmesi gerektirdiğinden,

$$O(N \cdot d' \cdot T)$$

mertebedindedir. Böylece Zebra Hunt Algoritması; lider çekimi, avcı itmesi ve adaptif dengeleme mekanizmasıyla çalışan stokastik, popülasyon temelli bir optimizasyon sistemi olarak matematiksel biçimde modellenmiş olur.

ZHO/ZOA, mühendislik tasarım problemleri, yenilenebilir enerji sistemleri optimizasyonu, mikro ağ planlama, kontrol sistemleri ve makine öğrenimi hiperparametre ayarlaması gibi alanlarda uygulanabilir ve hibrit yenilenebilir enerji sistemlerinde maliyet ve performans optimizasyonunda başarılı sonuçlar vermiştir (Abebe Ashetehe vd., 2024).

1.3.3 Yarasa Optimizasyonu (BA) ve Farksal Gelişim Algoritması (Differential Evolution-DE) ile Oluşturulan Hibrit Algoritma (MBADE)

Yarasa algoritması kullanılarak gerçekleştirilen çözümler incelendiğinde, yarasa algoritmasının hem lokal hem de global arama yeteneğinin geliştirilmesi gerektiği görülmektedir. BA algoritmasının yerel arama bölümü incelendiğinde, popülasyondan seçilen bir çözüm etrafında arama yapıldığı görülür. Literatürde, bu arama işlemi için, çoğunlukla o anki çözüm ya da popülasyondaki global en iyi çözüm kullanılır. Yerel arama işleminde, yeni çözüm üretilirken yapılan hesaplama, bireyi oluşturan tüm boyutlarda gerçekleştirilmektedir. Bu durum, BA'da bireylerin birbirine hızla benzemesine ve çeşitliliğin kaybolmasına neden olur. DE algoritmasında yeni çözüm oluşturulurken tek bir çözüm etrafında arama yapmak yerine, popülasyondan seçilen üç veya daha fazla birey arasında belli stratejiler doğrultusunda mutasyon ve çaprazlama gibi işlemler uygulanarak yeni çözüm oluşturulur. Böylece, popülasyon çeşitliliği daha uzun süre korunur ve daha detaylı bir arama gerçekleştirilir. BA ve DE'nin birlikte kullanıldığı hibrit algoritmayı oluşturma fikri de bu sebepten dolayı ortaya çıkmıştır. Önerilen hibrit algoritmanın geliştirilmesi sürecinde, öncelikle BA'nın performansını artırmaya yönelik bazı düzenlemeler yapılarak gelişmiş bir BA algoritması (MBA) elde edilmiştir. Ardından, algoritmanın performansını daha fazla artırmak ve DE'nin güçlü yönlerini BA'ya kazandırmak amacıyla DE ile hibrit çalışan bir algoritma (MBADE) oluşturulmuştur. Standart BA algoritmasının, global arama bölümündeki konum güncelleme denklemi incelendiğinde, popülasyonda bulunan çözümlerin, hızlı bir biçimde sistemdeki en iyi çözüme yakınsama eğiliminde olduğu görülmektedir. Çünkü yeni hız değeri, popülasyondaki global en iyi çözüme bağlı olarak oluşturulur ve yeni konumu oluşturmak için yeni hız değeri ile yapılan toplama işlemi mevcut çözümün tüm boyutlarını etkiler. Bu durum, yarasa algoritmasının global arama yeteneğini zayıflatmaktadır. Bu nedenle, ilk olarak algoritma yeni bir konum belirleme stratejisi kullanılarak geliştirilmiştir. BA algoritmasının performansı incelendiğinde, standart algoritmaya göre performansının ciddi şekilde arttığı görülse de, hala geliştirilmeye ihtiyaç duyduğu tespit edilmiştir. Gelişmiş BA algoritmasında, kullanılan lokal arama formülü sebebiyle algoritmanın ürettiği en iyi çözüme hızlı yakınsama davranışı devam etmektedir. Popülasyondaki bir çözüme hızlı yakınsama, çalışılan problemin türüne ve boyutuna göre kimi zaman avantaj, kimi zaman dezavantaj olabilir. Bu açıdan hızlı yakınsama karakteristiğini de kaybetmeden algoritmanın performansını geliştirme

ihtiyacı ortaya çıkmıştır. BA algoritmasının lokal arama yeteneğini daha fazla desteklemek ve çeşitlendirmek amacıyla, DE algoritmasıyla birlikte kullanıldığı hibrit MBADE algoritması önerilmiştir. DE algoritması, rastgele seçilen bireyler arasında bilgi değişimini farklı stratejilere göre yapan lokal arama yeteneği güçlü bir algoritma olması sebebi ile tercih edilmiştir. DE algoritmasında yeni çözüm üretilirken, BA'dan farklı olarak tek bir çözüm kullanmak yerine popülasyondan seçilen en az üç çözüm kullanılır. Bu yapısı, belli bir çözüme doğru hızlı yakınsamanın azaltılmasına ve popülasyon çeşitliliğinin artmasına neden olur. Önerilen hibrit algoritmada, popülasyondaki bir bireye hangi algoritmanın uygulanacağı performans odaklı bir olasılık değerine göre karar verilir. Algoritmalara ait olasılık, ürettiği uygunluk değerinin başarı durumuna göre güncellenir ve zamanla başarılı olan algoritmanın seçilme olasılığı artar. Aynı iterasyonda, popülasyondaki bireylere farklı algoritmalar uygulanabilir, bu da farklı lokal arama stratejilerinin bir arada kullanılabilmesine imkan verir. Böylece, popülasyon çeşitliliği daha uzun süre korunur ve yakınsama hızı probleme ve boyuta göre adapte edilir (Yıldızdan, G., 2021).

1.3.4 BA ve ABC ile Oluşturulan Hibrit Algoritma (BA_ABC)

BA' nın yapısal olarak arama sürecinin başlangıcında lokal arama yapma eğiliminde olması, aramada kullanılan denklem nedeniyle popülasyondaki birey çeşitliliğinin azalması, arama uzayının ayrıntılı aranmamasına ve lokal minimum tuzaklarına hızlıca düşmesine neden olmaktadır. Özellikle, arama uzayının geniş olduğu büyük boyutlu problemlerde, bu sorunlar daha belirgin hale gelmekte ve ciddi performans düşüşlerine neden olmaktadır. BA' nın global arama yeteneğinin geliştirilmesi konusundaki araştırmalar, global arama konusunda daha başarılı olan bir algoritmadan faydalanma fikrini ortaya çıkarmıştır. Bu doğrultuda, tez çalışması kapsamında, literatürde global arama becerisinin yüksek olmasıyla bilinen ABC algoritmasıyla (Karaboğa ve Baştürk, B., 2008; Abu-Mouti ve El-Hawary, 2012; Babayigit ve Özdemir, 2012) BA' nın beraber kullanıldığı bir hibrit algoritma önerilmiştir. BA' da global arama işlemi sırasında yeni çözüm oluşturulurken mevcut çözümün tüm boyutlarında değişiklik olur. Global aramada kullanılan bu denklemler nedeniyle, iterasyon ilerledikçe popülasyondaki bireyler (çözümler) hızla en iyi bireye ve birbirlerine benzemektedir. Dolayısıyla, popülasyonun çeşitliliği de hızla azalmaktadır. ABC' de ise yeni bir çözümün oluşturulması aşamasında, sadece rastgele seçilen bir boyutta değişiklik olmaktadır. Bu durum, en iyi çözüme doğru hızlı yakınsamayı engeller. Ayrıca, ABC algoritmasında belirli bir limit değeri boyunca

çözüm geliştirilemezse, o çözüm unutulur ve yerine rastgele bir çözüm atanır. Bu yaklaşım, gerektiğinde arama uzayının rastgele noktalarından yeniden aramaya başlamak anlamına gelir. Böylece daha iyi bir global arama ve popülasyon çeşitliliği sağlanmış olur. Bu olumlu yönlerinin yanı sıra, ABC' nin bu arama karakteristiği arama uzayının genişliği arttıkça bir dezavantaj haline gelebilir. Çünkü büyük boyutlu bir arama uzayında, sadece bir boyutta değişiklik yaparak yeni çözüm üretmek, yavaş bir arama yapmaya ve buna bağlı olarak arama uzayının bir kısmının hiç araştırılamamasına neden olabilir. Bu yüzden, ABC' nin global arama becerisinden faydalanırken, BA' nın hızlı yakınsama karakteristiğini de kaybetmeyen bir hibrit algoritmaya ihtiyaç vardır. Yapılan düzenleme sayesinde önceki hız değerinin, yeni hıza etkisi başlangıçta daha fazla olmuştur ve iterasyon ilerledikçe önceki hızın etkisi azalmıştır. Başka bir deyişle, başlangıçta algoritmanın ilerleme hızı (adım miktarı) daha fazladır. İterasyon ilerledikçe azalır. Bu durum başlangıçta global aramanın, daha sonra lokal aramanın önem kazanacağı anlamına gelir. BA'da yapılan bu düzenleme sonrası, algoritmanın global arama yeteneğini ve çeşitliliğini artırmak amacıyla, bu gelişmiş BA algoritmasıyla standart ABC algoritmalarının birlikte kullanıldığı bir hibrit algoritma oluşturulmuştur. Bu hibrit algoritma sayesinde, BA' nın hala devam eden global en iyi çözüme hızlı yakınsama ve zamanla popülasyon çeşitliliğini kaybetme problemlerinin giderilmesi amaçlanmıştır. Bu doğrultuda, BA ve ABC' nin paralel yürütüldüğü hibrit bir algoritma (BA_ABC) geliştirilmiştir. Hibrit algoritmada popülasyon önce ikiye bölünür. Popülasyonun ilk yarısındaki bireylere BA algoritması, ikinci yarısındaki bireylere ise ABC algoritması uygulanarak yeni çözümler üretilir. En iyi çözüm değeri, algoritmalar tarafından daha iyi çözümler bulundukça güncellenir. Her belli sayıda iterasyon tamamlandığında, algoritmaların performansları incelenir. Yani BA ve ABC için tanımlanmış ve global en iyi çözümden, daha iyi çözüm buldukları zaman artırılan sayaçlara bakılır. Geçen süreçte, hangi algoritma daha fazla yeni çözüm üretmişse (yani hangi sayaç daha büyükse) o algoritmaya ait popülasyondaki uygunluk değeri en iyi olan bireylerin bir kısmı (sayısı kadar) diğer algoritmanın aynı sayıda ve uygunluk değeri en kötü bireylerinin yerine yazılır. Böylece popülasyonlar arası bilgi değişimi sağlanır. Her bilgi değişimi sonrasında, daha fazla sayıda yeni çözüm üretmiş olan başarılı algoritmaya ait sayaç artırılır. Bu sayaçlardan herhangi biri maksimum değişim sayısına ulaştığında ise sayaca ait algoritmanın o problem için daha başarılı olduğu kabul edilir ve kalan iterasyonlara tüm popülasyon üzerinde bu algoritma yürütülerek devam edilir (Yıldızdan, G., 2021).

1.3.5 Hibrit Genetik-Karga Algoritma

Genetik Algoritma’da kullandığımız ilk nesli rastgele yaratmaktansa Karga Algoritma’sı kullanarak oluşturduğumuz sonuçları Genetik Algoritmaya ilk nesil olarak verilirse bulunan çözüm optimum çözüme daha yakın bir çözüm olur. Karga Algoritma’sı kullanılırken yiyeceklerin ilk konumlarını rastgele yaratmaktansa Genetik Algoritmadan çıkan çözümler Karga Algoritma’sına yiyeceklerinin ilk konumları olarak verilir. Yani yeni bir problem çözülürken daha önce bu problemle alakalı Karga Algoritma’sı tarafından bulunan çözümlerden faydalanılarak çözüm uzayı daraltılmaya çalışılır. Bu şekilde probleme Hibrit Genetik-Karga Algoritma uygulanmıştır (Khudeer ve Erbay, 2021).

1.3.6 Hibrit Genetik Algoritma (HGA)

Yol planlama problemlerinde, problemin boyutu ve ortamdaki engelsiz noktas sayısı arttıkça, uğrayacağı her nokta için belirli bir süre gerekeceği için problemde çözüm süresi uzamaktadır. Bu yüzden yol planlama problemlerinde problemin boyutu arttığı zaman algoritmaların problemi çözme süresi de artmaktadır. Bir diğer önemli etken ise probleme uygun bir şekilde oluşturulması gereken popülasyon büyüklüğü olup, problemin boyutuna göre tasarlamak önemlidir. Algoritmalar kısa sürede kesin sonuç üretilmezse bile geçerli en iyi çözümü en kısa sürede üretebilmek için metasezgisel yöntemlere başvurmaktadır. Genetik algoritma bu yöntemlerden olup, optimum çözüme ulaşmayı hedefleyen metasezgisel bir yöntemdir. Genetik algoritma, yol planlama problemlerinde en iyi çözümleri elde edebilecek alt yapıya sahip olması, algoritmadaki çaprazlama ve mutasyon gibi operatörlerin etkin kullanılması, problemin çözümü için avantaj sağlar. Genetik algoritmaya, Dijkstra algoritması uyarlanarak daha iyi bir çözüme daha kısa sürede ve hızlı bir şekilde ulaşmayı hedefleyen, sadece genetik algoritmanın metasezgisel olarak işlevini aynı şekilde ilerleteceği yeni bir hibrit algoritma oluşturulmuştur. Genetik algoritmada olduğu gibi, Hibrit genetik algoritmada da başlangıç popülasyonu üretme, uygunluk değeri hesaplama ve mutasyon operatörleri aynı şekildedir fakat hibrit genetik algoritmada farklı olarak mutasyon işleminin ardından çözüme, popülasyondaki tüm bireylere Dijkstra algoritması uygulanarak dahil olmaktadır. Bu işlemle birlikte algoritma hibrit algoritmaya dönüşmektedir. Dijkstra algoritması uygulanması için maksimum hücre sayısı belirlenir. Bu hücreler içerisinde rastgele bir nokta ve ardından rastgele ikinci nokta seçilir. Bu seçilen iki nokta arasında Dijkstra algoritması uygulanır. İkinci seçilen nokta, karar vericinin belirlediği hücre

bölgesinde çözüme ait bir eleman olması gerekir. Buradaki maksimum hücre sayısı bu alanın sahip olduğu büyüklüğü ifade eder. Örnek; karar verici Dijkstra uygulanacak alanı 36 olarak belirlemişse, rasgele seçilen genin $2 \times 2, 4 \times 8, 6 \times 6, 3 \times 7, 3 \times 8, \dots, 1 \times 36$ boyutundaki ulaşılabilir (iki nokta arasında yol olmayabilir) tüm komşuluklarına bakılır. Burada önemli olan seçilen komşuyla oluşturulan alanın 36 birim kareyi aşmamasıdır. Bu komşuluklarda, yine çözümde olan başka bir genin tespit edilmesi gerekmektedir. Rasgele seçilen nokta ile, 36 birimlik komşuluk içinde kalan ve yine çözümün bir parçası olan herhangi bir nokta arasına Dijkstra algoritması uygulanır. Genetik algoritma ile kaliteli çözümler aranırken, aynı zamanda çözümlerin belirli noktalarına Dijkstra Algoritması uygulanıp bir yerel arama yapılarak daha iyi çözümlere hızlı bir yakınsama sağlanmıştır (Boğar, E. 2016).



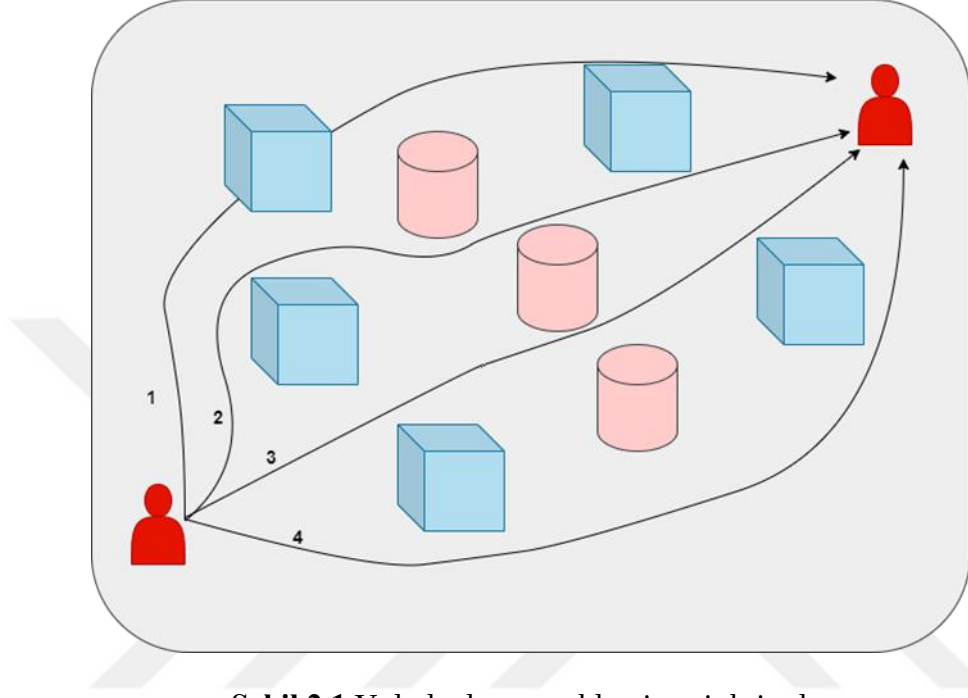
2 YOL PLANLAMA

Yaşamın önemli bir parçası haline gelmiş olan robotik sistemler, zaman içerisinde değişerek ve gelişerek hayatımızı kolaylaştırmak için; ulaşım sistemleri, akıllı sistemlerle karşılanan ev içi ihtiyaçlara cevap veren sistemler, sağlık sistemleri ve laboratuvar alanları, askeri teknoloji sistemleri, havaalanları, alışveriş merkezleri, sanayi sistemleri gibi birçok alanda geliştirilerek insanların kullanımına sunulmuştur. Robotik sistemler, etkin performansları ile mesafe, zaman, maliyet, birim zamanda üretim miktarı, iletişim, güvenlik, düzgünlük, enerji tasarrufu sağlayarak ve hayatımızın vazgeçilmezleri olarak günlük hayatımızın merkezinde önemli roller üstlenmektedir. Tüm bu robotik sistemlerin hareketi üzerine kurulu düzeneklerinin, izleyeceği güzergahının bulunması veya oluşturulması bir optimizasyon problemi olup yol planlama problemi başlığı altında incelenmektedir (Latombe J. C., 1991).

Robot yol planlama sistemleri üzerinde çalışmalar yapan araştırmacılar problemi daha somut hale getirmek için hücre ayrıştırması (Graf vd., 2001), yol haritalama (Zuo, X., 2003) yöntemlerini geliştirerek karmaşık yapıda olan çevre koşullarını basit bir yapıya dönüştürüp hesaplamayı kolaylaştırmıştır.

Yol planlama problemi, bir alanın haritasını çıkartarak başlangıç noktasından ulaşılmak istenen hedefe, önüne çıkan engellere çarpmadan ilerleyerek en kısa zamanda, en kısa mesafeyi uygun yollardan giderek en iyi çözümü bulmayı hedefleyen çözümleri barındıran problemlerdir. Yol planlama problemlerinde dikkat edilecek hususlardan ilki, engeller ve yol arasındaki mesafedir. Engellerle yol çakışmaması gerektiği gibi aralarında uygun bir uzaklık olması gerekir. Engellerle mobil robotlar arasındaki mesafeye de ayrıca dikkat edilmesi gerekmektedir. Engellerle mobil robot arasındaki mesafe kısa olursa engelle çarpması olasıdır ve izlediği güzergâhtan sapması problemin çözümünü tamamen olumsuz etkileyeceği için dikkat edilmesi gerekmektedir. Engellerle robot arasındaki mesafe fazla olursa bu kez de oluşturulan güzergâh uygun bir yol olmaktan uzaklaşacağı için, rota oluşturulamaz ve uygun bir çözüm elde edilemez. Robot için yol gidilebilir

özelliklere sahip olması için engebe, zikzak, çukur olmaması, robotun geçemeyeceği şekilde bozuk bir fiziki yapıya sahip olan yolların seçilmemesi gerekir. Bu koşullar sağlandıktan sonra uygulanan algoritma bulduğu yollardan en kısa olanı seçilir. Böylece hem zaman hem de maliyet açısından diğerlerinden üstün olması gerektiği gibi bozuk bir güzergâh üzerinden olmamalıdır.



Şekil 2.1 Yol planlama problemine ait kriterler

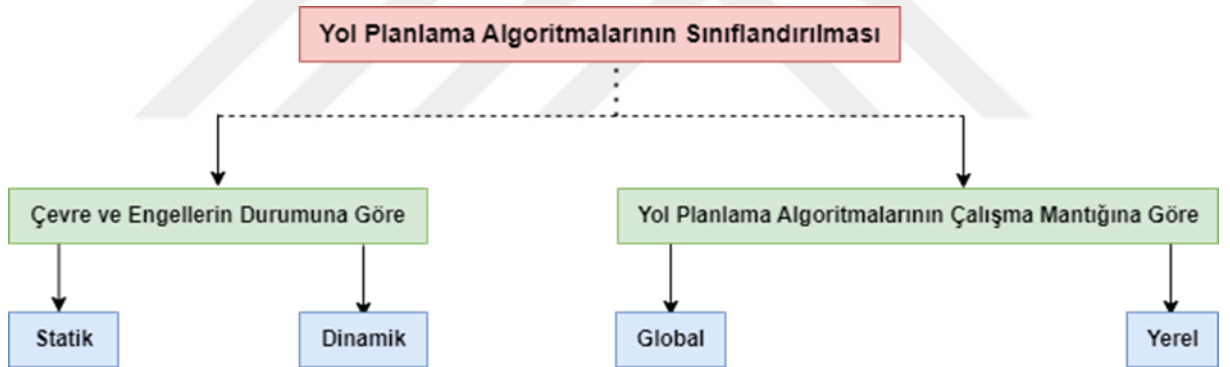
Yol planlama kriterleri göz önünde bulundurulduğunda, Şekil 2.1’de haritalandırılmış düzlem üzerinde oluşturulan güzergahlar incelendiğinde 1.yol engele çarptığı için kullanılmaya elverişli bir güzergah değildir. 2., 3., ve 4. güzergahlardan oluşan yollar herhangi bir engele çarpmadığı, engellerle arasında yeterli mesafede oldukları ve yolları olabildiğince uzatmadan hedefe vardıkları için hepsi birer seçenek dahilindedir. Bu seçenekler içerisinde 2.yol ve 4.yol neredeyse aynı özellik ve uzunluklara sahiptir. Yol planlama algoritmaları bazen aynı uzunlukta ve aynı özelliklerde birden fazla yol seçeneği sunabilir. Bu yol seçeneklerinin her birisi uygundur ve 2. ile 4. güzergahtan herhangi biri kullanılabilir. 2. ve 4. Yol güzergahlarında herhangi bir engele çarpmadıkları halde hedefe daha kısa mesafelerden ulaşılabilme imkânı varken daha uzun bir yollardan oluştuklarından kullanmak optimum değildir ve tercih edilmezler. Haritaya baktığımızda 3. yol herhangi bir engele çarpmadığı, engelle arasına belirli bir mesafe koyduğu ve yolu uzatmadan optimum uzunlukta bir yol olduğu için uygun bir yol olarak görülebilir. Sonuç olarak engelle çarpmadan hedefe ulaşabilen üç farklı yol varken en kısa mesafe üstünlüğü ile optimum çözümü 3 numaralı güzergah tamamlamaktadır. Haritada

da gözlemlendiği üzere bir yol planlama probleminin çözümü için bir sürü parametrenin dikkate alınması ve bu parametrelerin hepsinin aynı anda sağlanmış olması gerekmektedir.

2.1 Yol Planlama Algoritmalarının Sınıflandırılması

Yol planlama problemlerinde mobil robotlar, bulundukları noktadan en uygun yolu tahmin ederek hiçbir engelle çarpmadan hedefe ilerlemek için çeşitli algoritmaları kullanır ve bu algoritmalar arasından en avantajlı olanı belirler. Bazı algoritmalar karmaşık ve yüksek boyutlu alanlarda daha iyi performans sergilerken, bazıları ise basit ve düşük boyutlu alanlarda performansı yüksek olabilmektedir. Yol planlama algoritmaları, yol planlama problemlerinin çözümünde kullanılarak yıllar içerisinde geliştirilmiştir (Garip, vd., 2018).

Algoritmaların çalışma prensiplerine göre birçok çeşidi bulunmaktadır ve yol planlamada kullanılan algoritmalar Şekil 2.2 'deki gibi iki gruba ayrılır (Kunchev vd., 2006; Mnubi, S. A., 2016).



Şekil 2.2 Yol planlama algoritmalarının sınıflandırılması

Robotlar farklı formlarda, ev tipi, sanayileşmiş, insansı, askeri ve medikal olmak üzere farklı ihtiyaçlara göre çeşitlere ayrılır. Bu robotlar sabit ve mobil olmak üzere iki yapıya ayrılır. Robotların çalışma sistemi navigasyon ve yol planlaması üzerinden ilerlediği için bazı teknikler üzerinde durulmaktadır. Küresel Yol Planlama (Çevrimdışı), Bölgesel Yol Planlama (Çevrimiçi), Engeller ile statik ve dinamik ortamlar ve ortam boyutları (2D,3D vb.) robot yol planlama için önemli tekniklerdir.

Yol planlama algoritmaları Şekil 2.3 'te çevre ve çevredeki engellerin durumuna göre 4 farklı sınıfa ayrılırlar (Mnubi, S. A., 2016).

ARAMA YAPILAN ENGELLERİN DURUMU	BİLİNİYOR	BİLİNMIYOR
	A	B
STATİK		
DİNAMİK	C	D

Şekil 2.3 Arama yapılacak çevrenin durumuna göre algoritmaların sınıflandırılması

A bölgesi engellerin statik durumdayken çevre ve engellerin yerinin bilindiği durumdur. Tüm harita tarandıktan sonra mobil robot rota belirler ardından hareket eder.

B bölgesi arama yapılan çevre ve engellerin yerlerinin bilinmeyip ya da kısmen bilinip engellerin statik olduğu durumu ifade eder. Mobil robot elinde bulunan bilgilerle basit bir yol planlayıp harekete geçer. Robot ilerledikçe üzerindeki sensörler vasıtasıyla çevre hakkında yeni bilgiler öğrenir ve bu bilgiler sayesinde yeni yollar planlar. Bu şekilde yerel ve hızlı yol planlaması yaparak ve parça parça hareket ederek hedefine ulaşır.

C bölgesi arama yapılan çevre ve engellerin yerlerinin bilinip engellerin dinamik olduğu durumu ifade eder. Mobil robot harekete geçmeden önce haritayı analiz edip rotasını oluşturur fakat engeller hareket edebildiği için üzerindeki sensörler (lazer, ultrasonik vb.) vasıtasıyla engellerden kaçma manevrası yapar ve engellerin çevresinden dolanarak hedefine ulaşır.

D numaralı bölge arama yapılan çevre ve engellerin yerlerinin bilinmeyip ya da kısmen bilinip engellerin dinamik olduğu durumu ifade eder. Mobil robot 2. bölgedeki gibi hareket eder fakat engeller hareket edebildiği için üzerindeki sensörler (lazer, ultrasonik vb.) vasıtasıyla engellerden kaçma manevrası yapar ve engellerin çevresinden dolanarak hedefine ulaşır.

Yol planlama algoritmalarında çalışma mantığına göre; Global (Küresel – Çevrimdışı) Yol Planlama ve Yerel (Bölgesel – Çevrimiçi) Yol Planlama olmak üzere iki farklı sınıfa ayrılırlar.

2.1.1 Global (Küresel – Çevrimdışı) Yol planlama

Robotun çalıştırılmasında kullanılan global yol planlama algoritmaları, haritada tüm bilgilerin önceden verildiği ve engellerin koordinatları ile hareket edeceği ortamın hakkındaki tüm bilgiye sahip olduğu kabul edilir. Global yol planlama algoritması, önceden sahip olduğu bilgiyle çevrimdışı çalışarak, hareket etmeye başlamadan önce sahip olduğu bilgiler ve haritadan yola çıkarak başlangıç noktası ve hedef noktası arasındaki en uygun olan optimum yolu hesaplar. Bu hesaba göre amaç, oluşturduğu rotayı değiştirmeden, belirli bir hızda takip ederek hiçbir engele çarpmadan hedef noktasına varmaktır. Basit mobil robotlarda rota oluşturulurken kullanılan yol planlama algoritması, çevrimdışı çalıştığı ve çalışma esnasında yeni bir yol keşfedemediği için statik bir yaklaşımdır.

2.1.2 Yerel (Bölgesel – Çevrimiçi) Yol Planlama

Robotun çalıştırılmasında kullanılan bir diğer teknik olan yerel yol planlama algoritması, robotun engellerin koordinatları ve çevre hakkında ya tamamen bilgisinin olmadığı veya çok az bilgiye sahip olduğu durumlarda kullanılır. Net bilinen bilgi başlangıç ve varılacak olan hedef noktasıdır. Robot, yol planlaması yaparken üzerindeki sensörlerden yararlanır veya dışardaki bir bilgi kaynağı tarafından bilgiler edinerek işlevini yerine getirir. Yerel yol planlama algoritmaları çevrimiçi çalışarak, elinde var olan bilgiler ve yolda ilerlerken elde ettiği bilgileri harmanlayarak, her yeni durumda yeni bir rota oluşturarak parça parça ilerleyerek hedef noktasına ulaşır. Global yol planlamaya göre hızlı davranırsa da, optimum yol bulma sayısı konusunda daha zayıftır. Karmaşık mobil robotlarda rota oluşturulurken kullanılan yol planlama algoritması, çevrimiçi çalıştığı ve her önüne çıkan engele çarpmamak için engelleri belirleyerek çalışma esnasında yeni bir yol keşfedip kendini her engelerde güncelleyerek ilerlemeye karar verdiği için dinamik bir yaklaşımdır. Yerel yol planlama algoritmaları global yol planlama algoritmalarına göre daha hızlıdır ve daha kısa bir süreçte cevap verir. Bunun sebebi yerel yol planlama algoritmaları tüm haritayı incelemiden en uygun rotayı bulana kadar hesaplama yapar. Yerel yol planlama algoritmaları çok daha hızlı ve tepkisel olmalarına rağmen global yol planlama algoritmaları kadar optimum yol bulamazlar.

2.2 Yol Planlama Teknikleri

2.2.1 Klasik teknikler

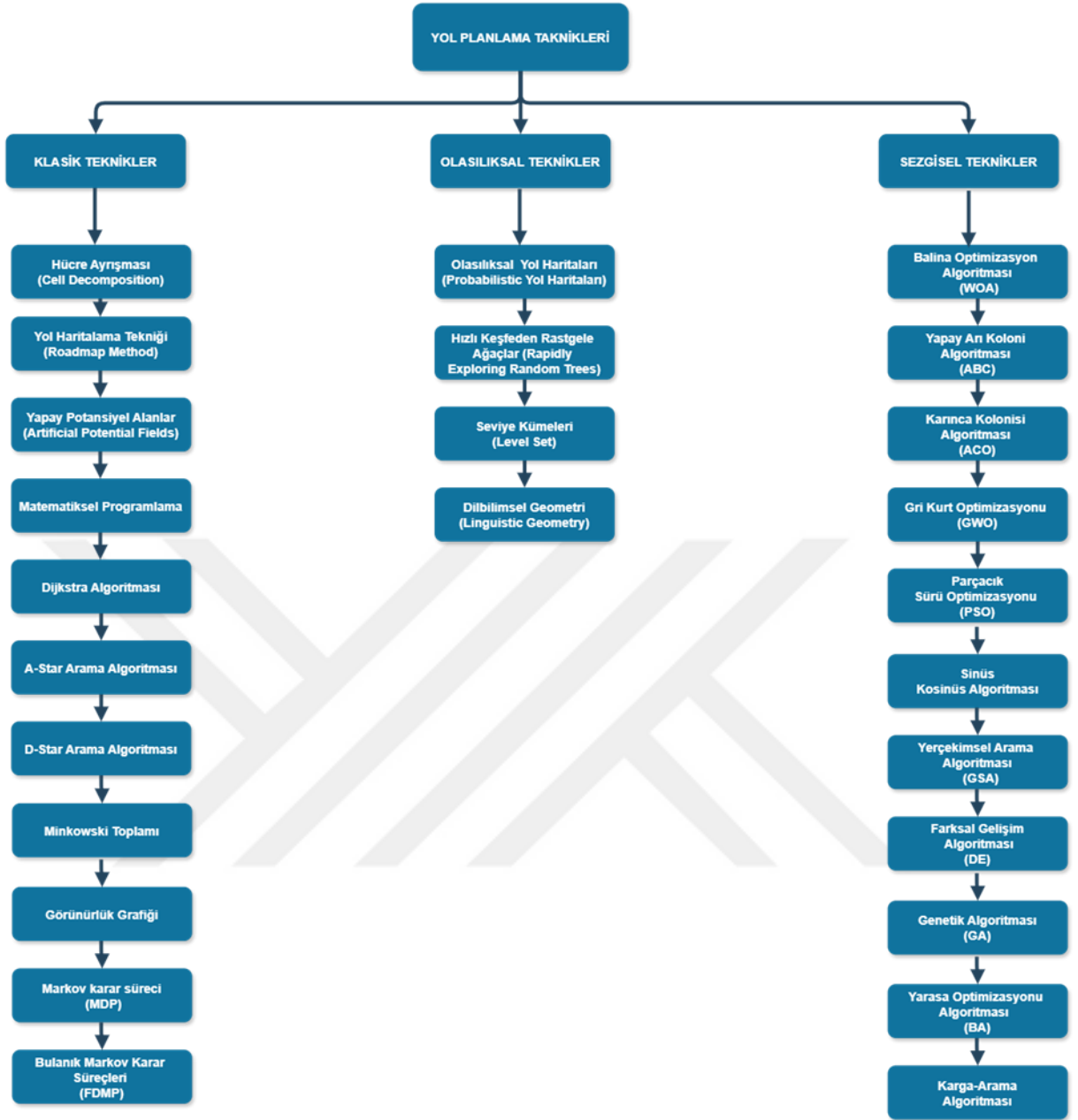
Bilinen en eski yol planlama tekniklerinden olan klasik yol planlama algoritmaları, daha önceden tamamı hakkında bilgi sahibi olduğu haritalarda çalışır. Statik ortam şartlarında ve global yol planlama algoritmaları ile haritada mevcut olan en uygun yolu, klasik yol planlama teknikleri ile bularak işlevini yerine getirmektedir. Bu tekniklerin kullanılmasında haritaların daha basit ve anlaşılır olması zaman açısından önemlidir çünkü çok fazla matematiksel işlem yükü olduğundan karmaşık ve büyük haritalarda hesaplamalar fazla olacağından dolayı süre ideal olandan fazla uzar ve kullanışlı olmaz. Bundan dolayı harita bilgisi olmayan veya eksik olan dinamik ortamlarda kullanıma elverişli değildir (Wahab vd., 2020).

2.2.2 Olasılıksal Teknikler

Yol planlama algoritmalarında tercih edilen diğer önemli teknik olan olasılıksal teknikler, klasik tekniklere benzer gözüktüğü de önemli bir noktada ayrılır. Bu önemli nokta harita üzerinde engellerin bulunmadığı noktaları rastgele belirler, ardından bu noktaları referans alarak yol planlamasını bu noktalar üzerinden gerçekleştirir. Çalışma prensibi klasik teknikler gibi basit olup kısa sürede sonuç vermesi avantajlı olsada sürekli bir noktadan başka bir noktaya rota oluşturması, yolun düz olmamasına ve sürekli zikzak şeklinde ilerlemesine sebep olur.

2.2.3 Sezgisel Teknikler

Yol planlama algoritmalarının en çok kullanıldığı teknik olan sezgisel teknikler, doğadan ilham alarak tasarlanan ve doğal fenomenleri referans alarak kullanan algoritmalar olarak yol planlaması tasarlamaktadır. Sezgisel tekniklerle yol planlamasını gerçekleştiren bu algoritmalar, kesin sonuç garantisi olmamakla birlikte olabilecek en kısa sürede ve en az işlem ile maliyetle, optimum bir sonuca ulaşarak, ideal çözümü sunar.



Şekil 2.4 Yol planlama teknikleri

DENEYSEL ÇALIŞMALAR VE TARTIŞMA

Bu bölümde önerilen yöntemler öncelikle CEC2017 ve CEC 2022 test fonksiyonları ile test edilmesinin ardından sonuçlar elde edildi. Daha sonra önerilen yöntemler ise farklı senaryolarla test edilmiş, elde edilen verilerin performansları karşılaştırılmış ve kıyaslanarak analizleri yapılmıştır.

3.1 CEC 2017 Sonuçları ve Diğer Metasezgizeller ile Karşılaştırılması

Bu bölümde yol planlama problemimiz için 6 sezgisel CEC2017 fonksiyonları ile test etmek için 3 tekrarlı ve 100 iterasyonluk sonuçları tablo ve grafiksel olarak sunulmuştur. Algoritmalarda, boyut 30, popülasyon sayısı ise 20'dir. Grafikler üzerinde sezgisel algoritmaların çözdüğü en kısa yol problemi ve en iyi maliyet değeri için yakınsamalar ve tablolarda; en iyi değer, ortalama, standart sapma ve en kötü değerler gösterilmiştir. En kısa yol problemini çözerken CEC2017 test fonksiyonları ile çalıştıracağımız IGWO, GJO, POA, Chimp, SCA, DMOA algoritmaların en iyi değerleri göz önüne alınarak karşılaştırmalar yapılmıştır. Performansı en iyi olan algoritmaya CEC2017 test fonksiyonlarından tablo ve yakınsama grafiklerinden elde edilen veriler sonucunda karar verilmiştir. Elde edilen yakınsama eğrileri Şekil 2.5'te, toplam rank analizleri ise Tablo 2.1'de verilmiştir.

Tablo 3.1 CEC2017 fonksiyonlarının sonuç değerleri

F	ALGORİTMA	BEST	MEAN	STD	WORST
F1	DMOA	26914756414.924	31062081649.4768	3875489153.4498	34591486767.2108
	İGWO	1610635720.0338	2760171443.5606	1685832022.7803	4695437495.0631
	GJO	11674250269.6431	15584264263.2909	3466237069.0506	18279975213.1007
	POA	27905268663.9401	37149453697.281	8997034032.8441	45877079515.5057
	CHIMP	28601778943.5969	31207758096.2286	2592863745.1418	33787304327.3418
	SCA	28892485706.4473	33893879413.3454	7713437340.2829	42777104084.5106

F2	DMOA	2.66827455017563 7e+41	6.937025117792713 e+44	1.198251672605761 e+45	2.077323090306684 e+45
	İGWO	6.07275337144619 4e+30	7.863744821642685 e+32	9.070528421706611 e+32	1.781580523814849 e+33
	GJO	2.59901328543900 9e+28	1.220773560505644 e+34	1.662469286624179 e+34	3.114160531372343 e+34
	POA	6.62456599634712 3e+35	8.668181699376703 e+35	3.083112401832707 e+35	1.221453074209492 e+36
	CHIMP	2.79890517866126 1e+38	3.727735223548412 e+38	1.212877433817983 e+38	5.099909977557062 e+38
F3	SCA	8.30202109928395 7e+37	2.687071328692975 e+45	4.654142146200076 e+45	8.061211770603638 e+45
	İGWO	139261.12	191559.1785	45723.8741	223981.7608
	GJO	113393.2594	128133.3628	13877.2524	140946.3133
	POA	67631.3946	75094.1639	9667.4742	86015.1496
	CHIMP	136357.7533	173570.9715	32500.2763	196378.801
	SCA	171334.2497	234169.7927	66201.8393	303289.7449
	DMOA	286244.8038	545744.762	230112.4165	724957.3836
F4	İGWO	657.3091	713.7978	56.7005	770.7078
	GJO	2126.9385	3721.9289	1500.5471	5105.6373
	POA	4709.6939	6937.0253	3527.7126	11004.3332
	CHIMP	6049.3153	9564.8907	4191.9744	14204.2051
	SCA	6659.7731	7132.6952	432.3488	7507.6627
	DMOA	8997.768	10061.7853	1156.7259	11293.0186

F5	IGWO	723.4076	741.792	28.0399	774.0656
	GJO	701.1422	770.1241	66.6967	834.2726
	POA	780.0315	807.188	42.4977	856.1633
	CHIMP	828.6204	834.4624	9.4774	845.3975
	SCA	876.1215	898.393	21.3257	918.6265
	DMOA	884.3317	910.5519	25.9764	936.2775
F6	IGWO	620.4093	626.5734	7.6902	635.191
	GJO	642.2591	648.1897	7.1314	656.1025
	POA	666.7477	669.998	3.6641	673.9689
	CHIMP	686.3389	689.8409	3.0388	691.7827
	SCA	674.6895	687.2973	20.7789	711.2801
	DMOA	686.6161	698.2617	10.3634	706.4687
F7	IGWO	994.319	1026.2236	53.4254	1087.9018
	GJO	1023.7125	1077.3443	54.6484	1132.9556
	POA	1236.7385	1296.4127	87.0788	1396.3352
	CHIMP	1369.2218	1385.4941	16.608	1402.4186
	SCA	1439.3678	1461.0378	21.1137	1481.5474
	DMOA	1240.4144	1390.0987	131.6248	1487.767
F8	IGWO	1053.5693	1059.0708	5.1976	1063.8988
	GJO	973.4117	1013.556	36.4384	1044.5407
	POA	1021.5374	1039.7306	15.9997	1051.6106
	CHIMP	1117.9683	1133.1506	15.2772	1148.5209

	SCA	1092.3229	1137.0292	39.7256	1168.2778
	DMOA	1053.5693	1059.0708	5.1976	1063.8988
F9	IGWO	3133.5327	3814.881	697.1945	4526.9087
	GJO	4893.566	6887.0044	2516.0691	9714.0941
	POA	7264.7891	7849.241	607.2474	8476.9686
	CHIMP	11993.5828	13016.3988	1305.4546	14486.7632
	SCA	14236.821	14623.9741	534.9208	15234.3534
	DMOA	20459.047	20745.3013	296.9126	21051.8327
F10	IGWO	9246.0236	9408.4805	179.9515	9601.9067
	GJO	8845.2774	9672.0315	874.5721	10587.6381
	POA	6613.3157	6839.1776	242.818	7095.9848
	CHIMP	8532.4826	9202.2746	580.6812	9564.0917
	SCA	9303.0171	9625.6944	316.6063	9935.8587
	DMOA	9478.6092	9923.7938	430.4745	10337.8705
F11	IGWO	1708.2844	3097.9369	2097.7449	5510.9549
	GJO	2840.1082	4900.003	2251.1256	7302.9732
	POA	3820.3392	5412.3998	1464.5182	6702.2122
	CHIMP	9616.2656	12586.1649	4751.9326	18066.8169
	SCA	5813.5206	6996.8806	1481.325	8658.1717
	DMOA	25895.4697	31821.6816	5653.7226	37156.409
F12	IGWO	107002926.3641	114979266.7546	12791348.2912	129733220.0884
	GJO	438832540.2911	1278228067.5971	905171151.5524	2237273747.8648

	POA	5515891813.0783	6282615883.501	901421017.4855	7275617566.8983
	CHiMP	6249910400.3226	7658225270.3866	1563811104.8012	9341153409.7038
	SCA	4661901885.2478	6313885381.6592	1585576030.797	7823444423.6452
	DMOA	1177647486.3256	2138464756.886	1241152025.3074	3539786514.7621
F13	iGWO	1856795.5107	25805829.9044	37913294.6016	69517560.6093
	GJO	21666789.2293	498327477.6377	468203094.4117	957587853.0987
	POA	10407130.0725	361487303.9419	599450737.1234	1053649322.152
	CHiMP	352737467.0385	9129715429.136	8457695317.2148	17227126553.9453
	SCA	1861360298.6944	2485649424.2164	700722970.1552	3243562981.771
	DMOA	54220486.1227	170744685.6011	140802983.3063	327200793.8638
F14	iGWO	127191.7862	217150.0712	99844.2792	324575.2729
	GJO	87822.782	1930193.8743	2068049.8235	4167085.1783
	POA	16551.933	327449.3277	269246.3152	483726.1151
	CHiMP	303213.0152	4100797.8955	6263726.1223	11330446.4341
	SCA	756293.0886	1787681.2134	1421735.6229	3409499.6284
	DMOA	1344351.4571	1625044.3843	476654.9642	2175401.2934
F15	iGWO	732370.0341	1211174.1339	756773.2248	2083636.7569
	GJO	1335184.7997	9997012.3038	13098162.9112	25065310.7404
	POA	81058.6055	278403.3149	245932.9267	553921.0928
	CHiMP	81241417.0306	89011740.1975	7322574.6151	95784228.8157
	SCA	17826951.8281	120498196.5475	156076824.4589	300106718.4769
	DMOA	7934892.024	15525856.0929	8235754.3513	24282245.1123

F16	İGWO	3353.6731	3576.1727	233.0029	3818.4217
	GJO	3319.6294	3658.112	371.2711	4055.2006
	POA	3361.9934	3788.2868	411.1398	4182.3793
	CHİMP	4853.413	5006.843	260.6866	5307.8389
	SCA	4776.8992	4821.9804	46.0636	4868.9672
	DMOA	4937.1482	5047.0284	154.5675	5223.771
F17	İGWO	2308.2741	2534.7721	228.1717	2764.5814
	GJO	2347.2892	2574.9688	348.1254	2975.7107
	POA	2466.9754	2681.5201	278.3867	2996.1016
	CHİMP	3091.6264	3837.3435	1006.8419	4982.6399
	SCA	2565.1294	3037.7938	411.0789	3311.9032
	DMOA	3410.0462	3535.1042	166.7219	3724.3872
F18	İGWO	2882576.7218	5257941.586	2090819.7578	6819467.0076
	GJO	825983.8122	7533510.994	11215392.7344	20481118.6276
	POA	340914.6216	750053.19	406233.5867	1153318.0979
	CHİMP	5173197.5455	11588123.2442	9490696.9562	22490381.4077
	SCA	27832546.6663	40167505.6535	10953346.2148	48756221.0692
	DMOA	80009565.209	122228003.7707	72199739.4909	205594792.6043
F19	İGWO	735379.8235	1342027.3169	661446.9142	2047216.88
	GJO	1583139.8033	5980353.6957	3820265.7214	8483612.3612
	POA	3749233.0778	5086854.5564	1986083.6224	7368924.3389
	CHİMP	107538501.0439	609793614.0336	440982213.9594	933517066.6611

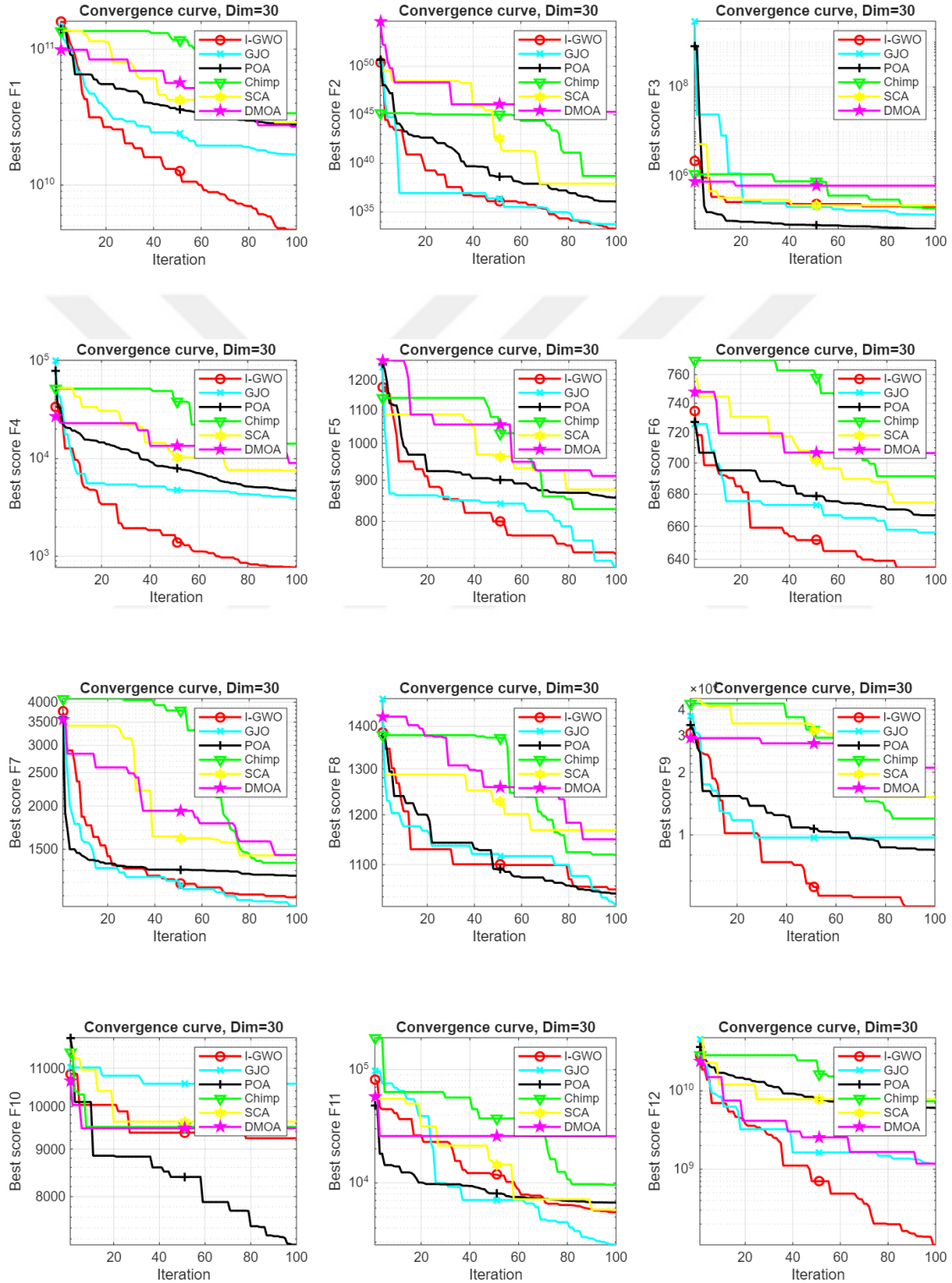
	SCA	104333684.9291	206306211.0047	114414041.8892	330037067.8329
	DMOA	1684860.5736	4292171.5779	3618765.8208	8423708.4532
F20	İGWO	2294.525	2497.5864	244.2838	2768.6728
	GJO	2578.9014	2756.4538	303.497	3106.8916
	POA	2664.3721	2733.4122	73.2214	2810.1989
	CHİMP	3371.5758	3523.7166	134.4405	3626.5103
	SCA	3111.7045	3214.8361	89.7376	3275.1044
	DMOA	3112.2119	3409.2182	317.6536	3744.1194
F21	İGWO	2468.0497	2522.3168	47.0134	2550.7023
	GJO	2531.4475	2563.5487	34.0389	2599.2407
	POA	2597.3862	2625.1719	41.5917	2672.9887
	CHİMP	2631.2578	2682.0256	55.9892	2742.076
	SCA	2682.794	2704.4468	31.041	2740.01
	DMOA	2644.4867	2701.9125	63.2125	2769.645
F22	İGWO	2532.8769	8047.7376	4793.5726	11215.1331
	GJO	6922.0436	8171.6637	1783.8345	10214.5368
	POA	6327.3186	7414.9408	1452.665	9064.6652
	CHİMP	10406.4145	10698.3967	317.9926	11037.2067
	SCA	8656.4618	10357.9285	1508.8916	11533.4879
	DMOA	11238.1155	11456.5655	214.0908	11666.013
F23	İGWO	2853.3224	2867.083	15.7467	2884.2561
	GJO	2928.4782	2978.8721	56.8448	3040.4925

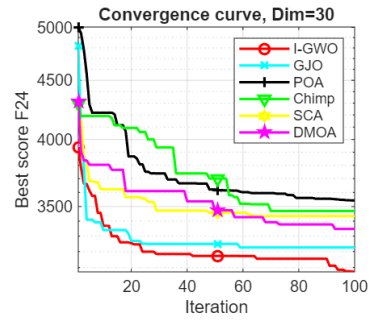
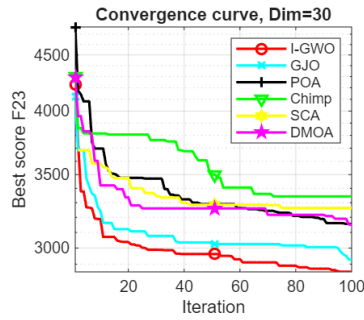
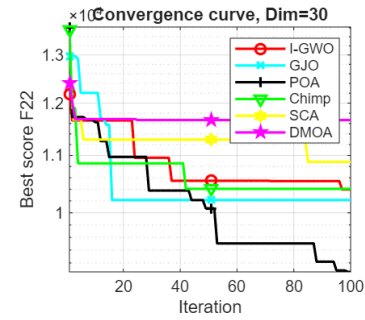
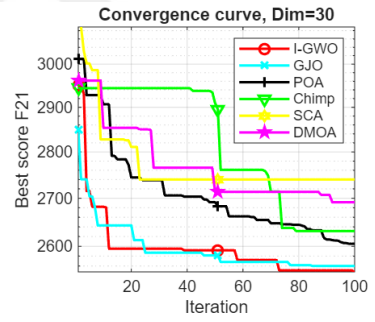
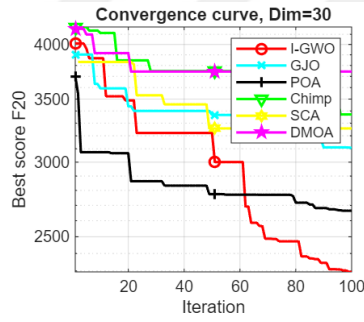
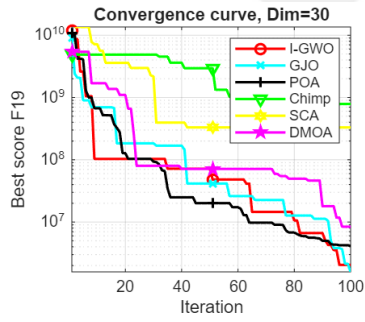
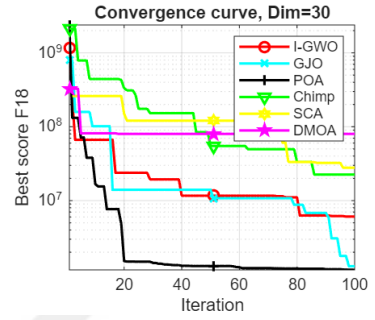
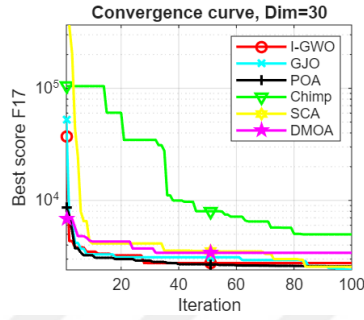
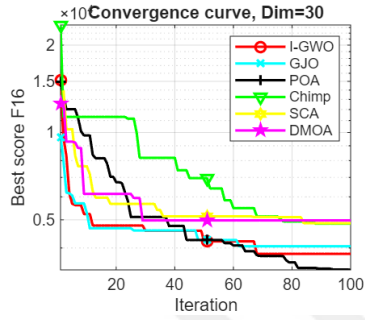
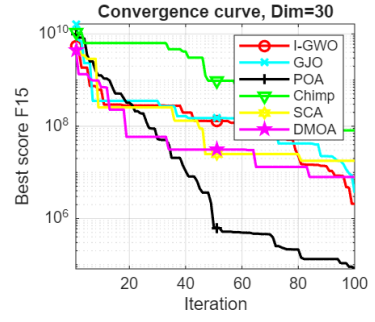
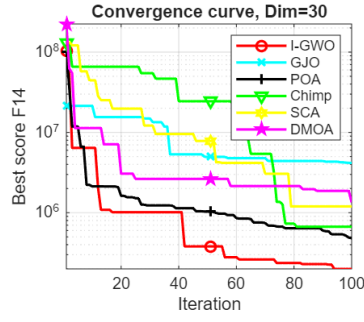
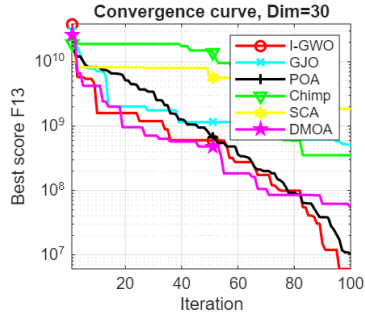
	POA	3155.9069	3325.851	162.5532	3479.8361
	CHIMP	3341.4559	3360.0659	30.6548	3395.447
	SCA	3081.7944	3205.963	107.6089	3272.0841
	DMOA	3078.6448	3108.1826	38.3361	3151.5048
F24	IGWO	3006.1609	3054.2385	41.9513	3083.4073
	GJO	3087.6674	3188.4385	88.1793	3251.4525
	POA	3351.4673	3467.9916	102.3103	3543.1053
	CHIMP	3447.4675	3501.1185	75.1431	3587.0004
	SCA	3286.178	3373.8877	77.8287	3434.7032
	DMOA	3280.533	3340.9291	57.4289	3394.8403
F25	IGWO	2956.882	3000.5873	37.8588	3023.2615
	GJO	3136.3444	3506.0241	560.228	4150.601
	POA	3897.146	4040.8582	207.6548	4278.9389
	CHIMP	4894.4563	5398.2603	567.8324	6013.5764
	SCA	4636.6642	5316.0384	1017.8949	6486.3565
	DMOA	4435.5452	5066.0155	739.7738	5880.3955
F26	IGWO	5484.7595	5800.1322	449.329	6314.612
	GJO	5833.5211	7092.1947	1198.2851	8219.2189
	POA	9569.5157	9966.1934	433.2327	10428.4937
	CHIMP	8215.4894	8794.2868	672.2491	9531.6409
	SCA	7702.193	9070.0149	1204.0072	9969.404
	DMOA	7016.8427	7892.523	778.5715	8506.6062

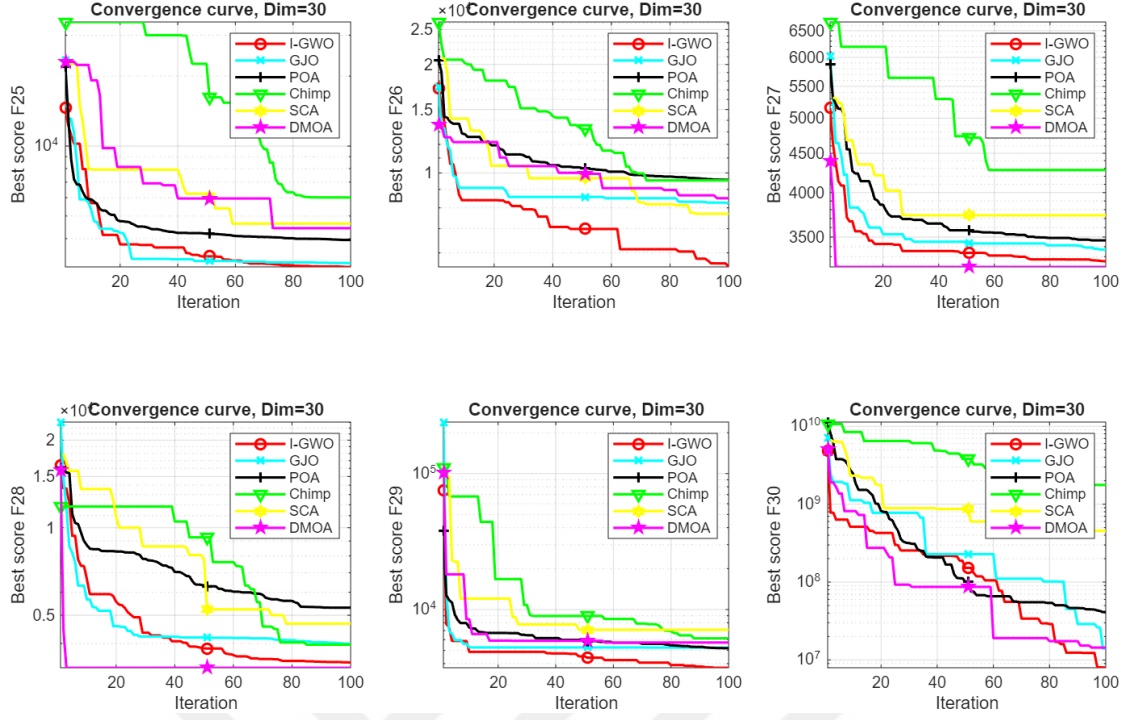
F27	İGWO	3253.5276	3264.9945	18.6681	3286.5356
	GJO	3368.0333	3563.8627	173.3726	3697.78
	POA	3465.1885	3515.0675	60.3724	3582.1839
	CHİMP	4278.5622	4347.9134	73.8038	4425.482
	SCA	3529.7351	3672.1631	123.5654	3750.7323
	DMOA	3200.0073	3200.0073	2.1681e-05	3200.0073
F28	İGWO	3440.1223	3503.586	96.1352	3614.1927
	GJO	3777.965	3970.7185	184.3975	4145.4342
	POA	5120.1076	5394.511	326.2511	5755.2454
	CHİMP	3963.1441	4909.1256	859.7673	5642.9591
	SCA	4678.5374	5637.5171	836.4129	6216.2797
	DMOA	3300.0073	3300.0073	9.605e-06	3300.0073
F29	İGWO	3732.2846	4148.1809	375.1378	4461.0149
	GJO	4897.3289	5033.6589	191.1173	5252.1117
	POA	4749.3704	4986.0038	225.9986	5199.6034
	CHİMP	5814.4471	6092.1006	260.6365	6331.4896
	SCA	5900.3081	6563.3135	606.9461	7091.5432
	DMOA	5230.3829	5688.3203	442.6009	6113.7975
F30	İGWO	5183726.9529	8798350.437	4106633.2892	13263727.222
	GJO	14067086.8395	101126717.9175	133626661.802	254981282.3652
	POA	22300717.9616	31150199.9653	9516032.3127	41215851.5476
	CHİMP	131673474.4272	714710607.2684	914580051.8453	1768795529.8376

SCA 445924963.1273 485195838.9814 61519128.9587 556094830.8242

DMOA 14408879.9926 32740113.9878 26853694.8636 63564339.5045







Şekil 3.1 CEC2017 test fonksiyonlarda karşılaştırma yapılan yöntemlerin yakınsama performansı

3.1.1 CEC2017 Fonksiyon Rank Karşılaştırma

F1

Tablo 3.2 CEC2017 fonksiyon rank analiz değerleri

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	3	2	9
Poa	4	6	6	6	22
Chimp	5	4	2	3	14
Sca	6	5	5	5	21
Dmoa	3	3	4	4	14

F2

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	2	1	1	1	5
Gjo	1	2	2	2	7
Poa	3	3	3	3	12
Chimp	5	4	4	4	17
Sca	4	6	6	6	22
Dmoa	6	5	5	5	21

F3

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	4	4	4	4	16
Gjo	2	2	2	2	8
Poa	1	1	1	1	4
Chimp	3	3	3	3	12
Sca	5	5	5	5	20
Dmoa	6	6	6	6	24

F4

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	3	2	9
Poa	3	3	4	4	14
Chimp	4	5	5	6	20
Sca	5	4	2	3	14
Dmoa	6	6	6	5	23

F5

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	2	1	4	1	8
Gjo	1	2	6	2	11
Poa	3	3	5	4	15
Chimp	4	4	1	3	12
Sca	5	5	2	5	17
Dmoa	6	6	3	6	21

F6

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	2	4	1	8
Gjo	2	3	3	2	10
Poa	3	1	2	3	9
Chimp	5	5	1	4	15
Sca	4	4	6	6	20
Dmoa	6	6	5	5	22

F7

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	3	1	6
Gjo	2	2	4	2	10
Poa	3	3	5	3	14
Chimp	5	4	1	4	14

Sca	6	6	2	5	19
Dmoa	4	5	6	6	21

F8

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	3	5	1	3	12
Gjo	1	1	5	1	8
Poa	2	4	4	2	12
Chimp	5	2	3	4	14
Sca	4	3	6	5	18
Dmoa	6	6	2	6	20

F9

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	4	1	7
Gjo	2	2	6	3	13
Poa	3	3	3	2	11
Chimp	4	4	5	4	17
Sca	5	5	2	5	17
Dmoa	6	6	1	6	19

F10

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	4	3	1	3	11
Gjo	3	5	6	6	20
Poa	1	1	2	1	5
Chimp	2	2	5	2	11
Sca	5	4	3	4	16
Dmoa	6	6	4	5	21

F11

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	3	1	6
Gjo	2	2	4	3	11
Poa	3	3	1	2	9
Chimp	5	5	5	5	20
Sca	4	4	2	4	14
Dmoa	6	6	6	6	24

F12

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	3	2	9

Poa	5	4	2	4	15
Chimp	6	6	5	6	23
Sca	4	5	6	5	20
Dmoa	3	3	4	3	13

F13

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	3	4	3	3	13
Poa	2	3	4	4	13
Chimp	5	6	6	6	23
Sca	6	5	5	5	21
Dmoa	4	2	2	2	10

F14

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	3	1	1	1	6
Gjo	2	5	5	5	17
Poa	1	2	2	2	7
Chimp	4	6	6	6	22
Sca	5	4	4	4	17
Dmoa	6	3	3	3	15

F15

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	2	2	2	2	8
Gjo	3	3	5	4	15
Poa	1	1	1	1	4
Chimp	6	5	3	5	19
Sca	5	6	6	6	23
Dmoa	4	4	4	3	15

F16

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	2	1	3	1	7
Gjo	1	2	5	2	10
Poa	3	3	6	3	15
Chimp	5	5	4	6	20
Sca	4	4	1	4	13
Dmoa	6	6	2	5	19

F17

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	2	1	5
Gjo	2	2	4	2	10
Poa	3	3	3	3	12
Chimp	5	6	6	6	23
Sca	4	4	5	4	17
Dmoa	6	5	1	5	17

F18

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	3	2	2	2	9
Gjo	2	3	5	3	13
Poa	1	1	1	1	4
Chimp	4	4	3	4	15
Sca	5	5	4	5	19
Dmoa	6	6	6	6	24

F19

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	4	4	4	14
Poa	4	3	2	2	11
Chimp	6	6	6	6	24
Sca	5	5	5	5	20
Dmoa	3	2	3	3	11

F20

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	4	1	7
Gjo	2	3	5	3	13
Poa	3	2	1	2	8
Chimp	6	6	3	5	20
Sca	4	4	2	4	14
Dmoa	5	5	6	6	22

F21

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Toplam Rank
İgwo	1	1	4	1	7
Gjo	2	2	2	2	8
Poa	3	3	3	3	12
Chimp	4	4	5	5	18

Sca	6	6	1	4	17
Dmoa	5	5	6	6	22

F22

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	2	6	4	13
Gjo	3	3	5	2	13
Poa	2	1	3	1	7
Chimp	5	5	2	3	15
Sca	4	4	4	5	17
Dmoa	6	6	1	6	19

F23

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	4	2	10
Poa	5	5	6	6	22
Chimp	6	6	2	5	19
Sca	4	4	5	4	17
Dmoa	3	3	3	3	12

F24

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	5	2	11
Poa	5	5	6	5	21
Chimp	6	6	3	6	21
Sca	4	4	4	4	16
Dmoa	3	3	2	3	11

F25

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	2	3	2	9
Poa	3	3	2	3	11
Chimp	6	6	4	5	21
Sca	5	5	6	6	22
Dmoa	4	4	5	4	17

F26

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	2	1	5
Gjo	2	2	5	2	11

Poa	6	6	1	6	19
Chimp	5	4	3	4	16
Sca	4	5	6	5	20
Dmoa	3	3	4	3	13

F27

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	2	2	2	2	8
Gjo	3	4	6	4	17
Poa	4	3	3	3	13
Chimp	6	6	4	6	22
Sca	5	5	5	5	20
Dmoa	1	1	1	1	4

F28

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	2	2	2	2	8
Gjo	3	3	3	3	12
Poa	6	5	4	5	20
Chimp	4	4	6	4	18
Sca	5	6	5	6	22
Dmoa	1	1	1	1	4

F29

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	4	1	7
Gjo	3	3	1	3	10
Poa	2	2	2	2	8
Chimp	5	5	3	5	18
Sca	6	6	6	6	24
Dmoa	4	4	5	4	17

F30

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Toplam Rank
İgwo	1	1	1	1	4
Gjo	2	4	5	4	15
Poa	4	2	2	2	10
Chimp	5	6	6	6	23
Sca	6	5	4	5	20
Dmoa	3	3	3	3	12

Tablo 3.1 incelendiğinde elde edilen verilerin büyüklüğünden dolayı kıyaslanması ve analiz edilmesi zor olacaktır. Bu sebeple her bir algoritma, en iyi değer(best), ortalama

değer(mean), standart sapma(std) ve en kötü değer(worst) kategorilerine ayrılarak en küçük değer 1 olmak üzere 1 ‘den algoritma sayısı referans alınarak 6 ‘ya kadar numaralandırdıktan sonra, Tablo 3.2 de katogorilere göre oluşturulan rank tablosundan yararlanarak Tablo 3.3 toplam rank tablosu oluşturulmuştur.

Tablo 3.3 CEC2017 fonksiyon toplam rank analiz değerleri

Algoritma	İgwo	Gjo	Poa	Chimp	Sca	Dmoa
F1	4	9	22	14	21	14
F2	5	7	12	17	22	21
F3	16	8	4	12	20	24
F4	4	9	14	20	14	23
F5	8	11	15	12	17	21
F6	8	10	9	15	20	22
F7	6	10	14	14	19	21
F8	12	8	12	14	18	20
F9	7	13	11	17	17	19
F10	11	20	5	11	16	21
F11	6	11	9	20	14	24
F12	4	9	15	23	20	13
F13	4	13	13	23	21	10
F14	6	17	7	22	17	15
F15	8	15	4	19	23	15
F16	7	10	15	20	13	19
F17	5	10	12	23	17	17
F18	9	13	4	15	19	24
F19	4	14	11	24	20	11
F20	7	13	8	20	14	22
F21	7	8	12	18	17	22
F22	13	13	7	15	17	19
F23	4	10	22	19	17	12
F24	4	11	21	21	16	11
F25	4	9	11	21	22	17
F26	5	11	19	16	20	13

F27	8	17	13	22	20	4
F28	8	12	20	18	22	4
F29	7	10	8	18	24	17
F30	4	15	10	23	20	12
Toplam	205	346	359	546	557	507

Rank tablosu tüm CEC2017 test fonksiyonları için değerlendirmeye alındığında; ortalama değer bakımından 30 test fonksiyonunun 21’inde IGWO birinci sırada yer alırken 6 en iyi ortalama sonuçla IGWO ’yu POA takip etmektedir. SCA ve DMOA ortalama değerde 30 test fonksiyonunun hiçbirinde en iyi sonuca ulaşamamıştır. Minimum değer bakımından incelediğimiz zaman IGWO, 30 test fonksiyonunun 19’unda en iyi sonuca ulaşarak üstünlüğünü diğerlerinden açık ara farkla göstermiştir. Bu yöntemleri 5 en iyi değerle POA, 4 en iyi değerle GJO ve DMOA ise 2 tanesinde en iyi sonucu vererek takip etmektedir. Chimp ve SCA ise hiçbir fonksiyon için en iyi sonuca ulaşamamıştır. Standart sapmada ise yine IGWO, 30 test fonksiyonundan 22 kez daha az sapma değeri vererek her kategoride IGWO, CEC2017 testi için en iyi performansı göstermiştir. IGWO’nun ardından toplan rank tablosunda en yakın performansı GJO ve ardından POA takip etmektedir. Şekil... incelendiğinde yakınsama eğrileride genel olarak bu sonucu desteklemekte IGWO çözümü en az maliyetle tamamlarken GJO ve POA hemen ardından sırasıyla IGWO’ yu takip etmektedir.

3.2 CEC2022 Sonuçları ve Diğer Metasezgizeller ile Karşılaştırılması

Bu bölümde yol planlama problemimiz için 7 sezgisel CEC2022 fonksiyonları ile test etmek için 3 tekrarlı ve 100 iterasyonluk sonuçları tablo ve grafiksel olarak sunulmuştur. Algoritmalarda alt sınır -100, üst sınır 100, boyut 10 ve popülasyon sayısı ise 20’dir. Grafikler üzerinde sezgisel algoritmaların çözdüğü en kısa yol problemi ve en iyi maliyet değeri için yakınsamalar ve tablolarda; en iyi değer, ortalama, standart sapma, en kötü değer ve zaman gösterilmiştir. En kısa yol problemini çözerken CEC2022 test fonksiyonları ile çalıştıracağımız IGWO, GJO, POA, GWO, Chimp, SCA, DMOA algoritmaların en iyi değerleri göz önüne alınarak karşılaştırmalar yapılmıştır. Performansı en iyi olan algoritmaya CEC2022 test fonksiyonlarından tablo ve yakınsama grafiklerinden elde edilen veriler sonucunda karar verilmiştir. Elde edilen yakınsama eğrileri Şekil 2.6 ’da, toplam rank analizleri ise Tablo 2.4’te verilmiştir.

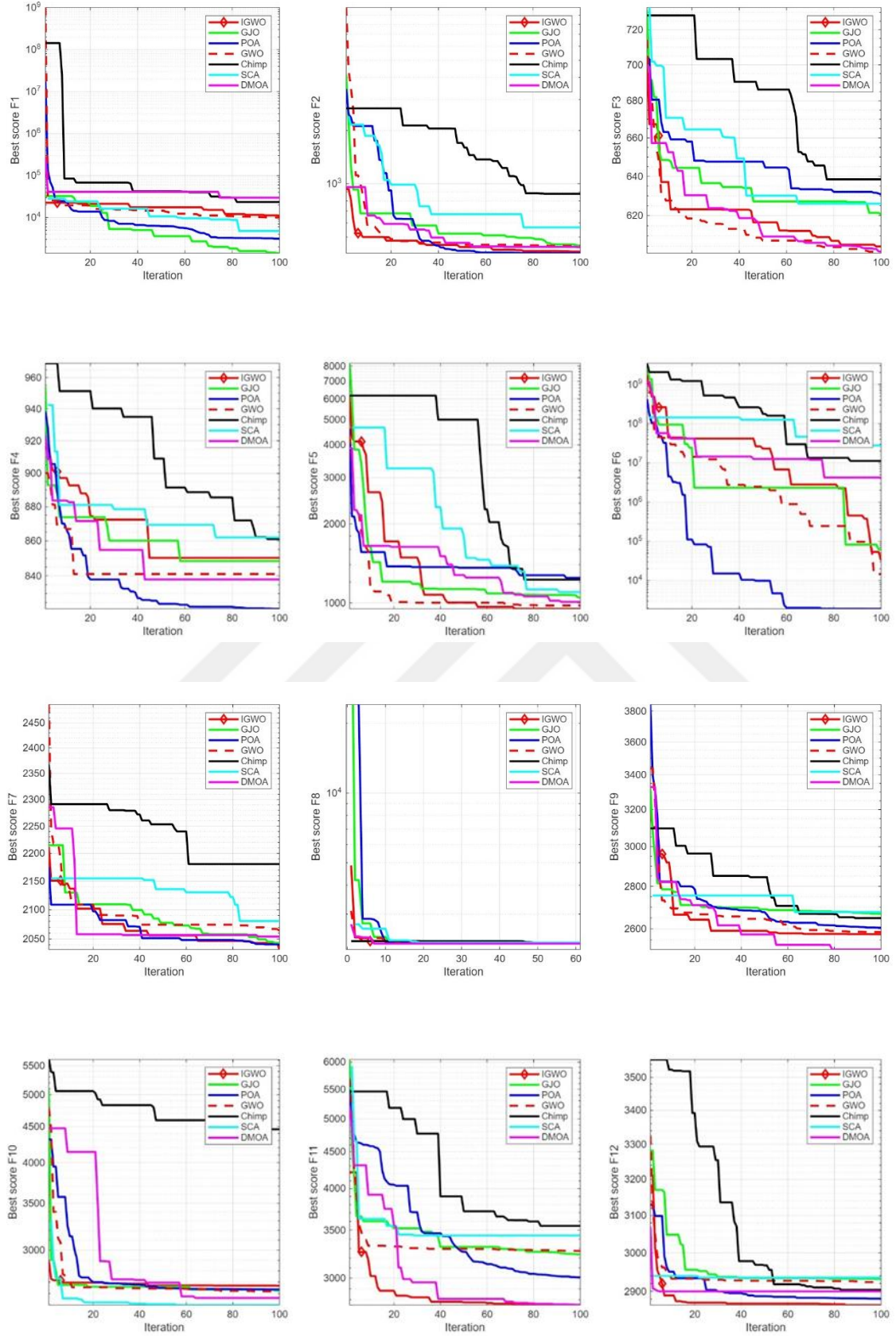
Tablo 3.4 CEC2022 fonksiyonlarının sonuç değerleri

F	ALGORİTMA	BEST	MEAN	STD	WORST	TIME
F1	İGWO	6767.0714	10611.186	6460.4349	15788.43	0.040371
	GJO	1417.8451	2898.2156	1343.6364	4040.5652	0.021084
	POA	2256.9636	3422.7992	1340.7356	4887.8692	0.016259
	GWO	2890.4356	9561.2841	6460.4349	15788.43	0.013402
	CHİMP	21944.8039	22544.0308	800.1487	23452.6872	0.072981
	SCA	4778.165	8043.4587	3446.6643	11646.6099	0.0095286
	DMOA	16964.9687	29390.66	12309.107	41579.7723	0.061809
F2	İGWO	411.7084	412.0913	29.0233	470.1401	0.03586
	GJO	447.282	454.6092	11.7782	468.1955	0.017402
	POA	404.3349	505.9314	92.9068	586.5673	0.011323
	GWO	412.2281	442.3244	29.0233	470.1401	0.014292
	CHİMP	762.2711	801.0425	65.9198	877.1557	0.075041
	SCA	560.8564	568.1239	8.6159	577.6417	0.0086052
	DMOA	408.1433	419.1003	15.824	437.242	0.057839
F3	İGWO	604.8269	607.6966	1.0963	603.3109	0.04143
	GJO	613.3783	617.1096	3.6003	620.5628	0.021719
	POA	627.4293	634.3182	9.1677	644.7235	0.01868
	GWO	601.273	602.0584	1.0963	603.3109	0.014347
	CHİMP	638.3866	650.3773	10.3871	656.6179	0.073913
	SCA	622.7946	625.2504	2.174	626.929	0.01256
	DMOA	602.4028	603.1084	0.94595	604.1833	0.086232

F4	İGWO	850.1995	855.4436	13.3049	847.4787	0.059824
	GJO	843.39	847.3647	3.5663	850.2844	0.037854
	POA	817.9259	822.4845	5.1215	828.0264	0.028996
	GWO	821.8863	836.7862	13.3049	847.4787	0.021111
	CHİMP	852.908	860.375	7.1759	867.2192	0.1425
	SCA	857.1444	867.1612	13.3563	882.3253	0.013072
	DMOA	837.9545	847.0638	8.2725	854.1084	0.14562
F5	İGWO	910.9389	928.4813	59.7704	1024.5445	0.036691
	GJO	1044.4481	1274.1479	393.1025	1728.0524	0.017368
	POA	1032.8488	1162.4683	113.903	1246.5906	0.014265
	GWO	905.9809	969.6658	59.7704	1024.5445	0.010614
	CHİMP	1227.0795	1471.9464	244.8734	1716.8263	0.060345
	SCA	1034.6528	1172.5357	184.7847	1382.497	0.008580 5
	DMOA	947.418	983.336	32.2418	1009.7776	0.048345
F6	İGWO	21329.1964	120876.4312	6532.3774	24506.8472	0.03477
	GJO	40239.7965	532911.3974	834188.813	1496064.466	0.017171
	POA	1904.0386	4356.2972	3053.1754	7775.9855	0.010669
	GWO	12170.5552	17096.9378	6532.3774	24506.8472	0.008721 7
	CHİMP	758543.3742	5684095.8118	5233028.630 5	11178173.4896	0.063137
	SCA	21318031.810 3	28811607.026 8	8090049.047 7	37388885.4435	0.008954 8
	DMOA	160182.3504	1706508.7722	2206294.981 7	4233066.8314	0.053232

F7	İGWO	2031.5411	2049.3975	16.8452	2063.9985	0.033245
	GJO	2043.0226	2055.5845	13.1253	2069.2086	0.018862
	POA	2040.3245	2057.7524	16.9556	2074.1926	0.023509
	GWO	2032.8048	2044.7275	16.8452	2063.9985	0.013642
	CHİMP	2067.5798	2158.7808	82.4352	2227.9893	0.061467
	SCA	2080.6038	2099.0655	16.9122	2113.8096	0.011943
	DMOA	2033.9617	2045.7655	10.479	2053.9721	0.063691
F8	İGWO	2229.0078	2232.7826	3.8235	2232.5013	0.035627
	GJO	2234.3412	2274.8415	63.9232	2348.5329	0.021365
	POA	2225.8327	2229.2652	3.0735	2231.7624	0.022741
	GWO	2225.744	2228.0892	3.8235	2232.5013	0.018148
	CHİMP	2243.6659	2330.3594	75.2203	2378.3183	0.061779
	SCA	2239.6708	2249.7488	10.344	2260.3397	0.014143
	DMOA	2236.8216	2241.5099	4.8936	2246.5858	0.066717
F9	İGWO	2530.6212	2545.6545	28.5525	2635.9098	0.037732
	GJO	2628.3329	2656.6172	24.5737	2672.7255	0.027506
	POA	2531.0526	2588.0331	50.6716	2628.0354	0.0249
	GWO	2585.2504	2618.1882	28.5525	2635.9098	0.022173
	CHİMP	2639.4833	2660.1691	27.9773	2692.0016	0.10585
	SCA	2652.7114	2672.9831	18.6876	2689.5238	0.01831
	DMOA	2507.8037	2516.9153	7.8927	7.8927	0.075339
F10	İGWO	2500.6771	2604.6203	2.3837	2617.7474	0.035569
	GJO	2500.5976	2581.8147	70.3951	2625.3054	0.019442

	POA	2501.2571	2593.223	79.752	2643.3414	0.015846
	GWO	2612.9817	2615.3281	2.3837	2617.7474	0.01077
	CHIMP	2510.8411	3221.5645	1077.9983	4461.9316	0.061178
	SCA	2502.7419	2606.3364	90.0742	2666.1651	0.010815
	DMOA	2501.0369	2523.8596	33.2528	2562.012	0.074539
F11	IGWO	2742.4176	2748.1068	294.4364	3273.5865	0.036124
	GJO	2789.9151	2951.9445	247.9496	3237.3824	0.020499
	POA	2750.5266	2836.8041	148.5904	3008.3806	0.020877
	GWO	2739.8837	2934.8946	294.4364	3273.5865	0.013485
	CHIMP	3547.8254	3712.5011	198.9781	3933.5967	0.062101
	SCA	2820.4955	3137.1479	309.9925	3440.0209	0.012723
	DMOA	2754.3027	2757.2063	4.7872	2762.7317	0.064395
F12	IGWO	2865.2172	2865.7878	26.5062	2923.8095	0.042279
	GJO	2875.5153	2902.2489	28.2106	2931.7347	0.024759
	POA	2881.2554	2906.3787	22.8125	2925.7981	0.025975
	GWO	2870.9981	2898.7356	26.5062	2923.8095	0.018253
	CHIMP	2904.1503	2926.5259	34.9427	2966.7911	0.075972
	SCA	2871.9832	2902.2467	31.6941	2935.1999	0.015781
	DMOA	2900.0023	2900.0023	2.3732e-05	2900.0023	0.073681



Şekil 3.2 CEC2022 test fonksiyonlarda karşılaştırma yapılan yöntemlerin yakınsama performansı

3.2.1 CEC2022 Fonksiyon Rank Karşılaştırma

F1 Tablo 3.5 CEC2022 fonksiyon rank analiz değerleri

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	5	5	5	4	5	24
Gjo	1	1	3	1	4	10
Poa	2	2	2	2	3	11
Gwo	3	4	5	4	2	18
Chimp	7	6	1	5	7	26
Sca	4	3	4	3	1	15
Dmoa	6	7	6	6	6	31

F2

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	3	1	4	3	5	16
Gjo	5	4	2	2	4	17
Poa	1	5	6	5	2	19
Gwo	4	3	4	3	3	17
Chimp	7	7	5	6	7	32
Sca	6	6	1	4	1	18
Dmoa	2	2	3	1	6	14

F3

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	3	3	2	1	5	14
Gjo	4	4	4	3	4	19
Poa	6	6	5	5	3	25
Gwo	1	1	2	1	2	7
Chimp	7	7	6	6	6	32
Sca	5	5	3	4	1	18
Dmoa	2	2	1	2	7	14

F4

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	5	5	5	2	5	22
Gjo	4	4	1	3	4	16
Poa	1	1	2	1	3	8
Gwo	2	2	5	2	2	13
Chimp	6	6	3	5	6	26
Sca	7	7	6	6	1	27
Dmoa	3	3	4	4	7	21

F5

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Zaman	Toplam Rank
İgwo	2	1	2	2	5	12
Gjo	6	6	6	6	4	28
Poa	4	4	3	3	3	17
Gwo	1	2	2	2	2	9
Chimp	7	7	5	5	7	31
Sca	5	5	4	4	1	19
Dmoa	3	3	1	1	6	14

F6

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Zaman	Toplam Rank
İgwo	2	3	2	2	5	14
Gjo	3	4	3	3	4	17
Poa	1	1	1	1	3	7
Gwo	4	2	2	2	1	11
Chimp	6	6	5	5	7	29
Sca	7	7	6	6	2	28
Dmoa	5	5	4	4	6	24

F7

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Zaman	Toplam Rank
İgwo	1	3	3	2	5	14
Gjo	5	4	2	3	3	17
Poa	4	5	5	4	4	22
Gwo	2	1	3	2	2	10
Chimp	6	7	6	6	6	31
Sca	7	6	4	5	1	23
Dmoa	3	2	1	1	7	14

F8

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Zaman	Toplam Rank
İgwo	3	3	2	2	5	15
Gjo	4	6	5	5	3	23
Poa	2	2	1	1	4	10
Gwo	1	1	2	2	2	8
Chimp	7	7	6	6	6	32
Sca	6	5	4	4	1	20
Dmoa	5	4	3	3	7	22

F9

Algoritma	Best Rank	Mean Rank	Std Rank	Worst Rank	Zaman	Toplam Rank
-----------	-----------	-----------	----------	------------	-------	-------------

İgwo	2	2	5	3	5	17
Gjo	5	5	3	4	4	21
Poa	3	3	6	2	3	17
Gwo	4	4	5	3	2	18
Chimp	6	6	4	6	7	29
Sca	7	7	2	5	1	22
Dmoa	1	1	1	1	6	10

F10

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	2	4	1	2	5	14
Gjo	1	2	3	3	4	13
Poa	4	3	4	4	3	18
Gwo	7	6	1	2	1	17
Chimp	6	7	6	6	6	31
Sca	5	5	5	5	2	22
Dmoa	3	1	2	1	7	14

F11

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	2	1	5	4	5	17
Gjo	5	5	4	3	3	20
Poa	3	3	2	2	4	14
Gwo	1	4	5	4	2	16
Chimp	7	7	3	6	6	29
Sca	6	6	6	5	1	24
Dmoa	4	2	1	1	7	15

F12

Algoritma	Best Rank	Mean Rank	Std Rank	Wost Rank	Zaman	Toplam Rank
İgwo	1	1	3	2	5	12
Gjo	4	5	4	4	3	20
Poa	5	6	2	3	4	20
Gwo	2	2	3	2	2	11
Chimp	7	7	6	6	7	33
Sca	3	4	5	5	1	18
Dmoa	6	3	1	1	6	17

Tablo 3.4 incelendiğinde elde edilen verilerin büyüklüğünden dolayı kıyaslanması ve analiz edilmesi zor olacaktır. Bu sebeple her bir algoritma, en iyi değer(best), ortalama değer(mean), standart sapma(std), en kötü değer(worst) ve zaman kategorilerine ayrılarak en küçük değer 1 olmak üzere 1 ‘den algoritma sayısı referans alınarak 7 ‘ye kadar

numaralandırdıktan sonra Tablo 3.5 de kategorilere göre oluşturulan rank tablosundan yararlanarak Tablo 3.6 toplam rank tablosu oluşturulmuştur.

Tablo 3.6 CEC2022 fonksiyon toplam rank analiz değerleri

Algoritma	İgwo	Gjo	Poa	Gwo	Chimp	Sca	Dmoa
F1	24	10	11	18	26	15	31
F2	16	17	19	17	32	18	14
F3	14	19	25	7	32	18	14
F4	22	16	8	13	26	27	21
F5	12	28	17	9	31	19	14
F6	14	17	7	11	29	28	24
F7	14	17	22	10	31	23	14
F8	15	23	10	8	32	20	22
F9	17	21	17	18	29	22	10
F10	14	13	18	17	31	22	14
F11	17	20	14	16	29	24	15
F12	12	20	20	11	33	18	17
Toplam	191	221	188	155	361	254	210

Rank tablosu incelendiğinde tüm CEC2022 test fonksiyonları değerlendirmeye alındığında; ortalama değer bakımından 12 test fonksiyonunun 4'ünde GJO birinci sırada yer alırken 3 en iyi ortalama sonuçla GJO 'yu GWO takip etmektedir. SCA ve DMOA ortalama değerde 12 test fonksiyonunun hiçbirinde en iyi sonuca ulaşamamıştır. Minimum değer bakımından incelediğimiz zaman GWO, 12 test fonksiyonunun 4'ünde en iyi sonuca ulaşarak üstünlüğünü göstermiştir. Bu yöntemleri 3 en iyi değerle POA, 2'şer en iyi değerle GJO ve IGWO takip etmektedir. CHIMP ve SCA ise hiçbir fonksiyon için en iyi sonuca ulaşamamıştır. Standart sapmada ise DMOA, 12 test fonksiyonundan 6 kez, daha az sapma değeri vererek bu kategoride CEC2022 testi için en iyi performansı göstermiştir. Zaman maliyeti açısından 12 test fonksiyonun içerisinde 10 kez en iyi performansı gösteren SCA'yı, 2 kez en iyi zaman değeri gösteren GWO takip etmektedir. Toplam performanslarının bir bütün olarak değerlendirildiği rank tablosunda ise çok az bir farkla GWO'nun performansının en iyi olduğu görülmüştür. Hemen ardından GWO'yu POA ve IGWO yakın değerlerle takip etmektedir. Yakınsama eğrileri ise verilerden elde edilen sonuçları destekler niteliktedir.

3.3 UAV ile Deneysel Çalışmalar

Deneysel çalışmalar, algoritmaların sayıları farklı sayılarda karşılaştırmaları yapılması için statik bir ortam içerisinde, statik engellerle gerçekleştirilmiştir. Her engel belirli bir yarıçapa sahip olup, eliptik şekillerle temsil edilmektedir. Bir UAV vasıtasıyla, konumu sabit bir başlangıç noktasından, yine konumu sabit olan hedef noktasına doğru yol planlamasını yaparak ilerlemektedir. UAV için yol planlaması, genellikle en kısa mesafeyi seçmek amacını taşır ancak bu süreçte bazen ekonomik faktörlerin ihmal edildiği gözlemlenmiştir. Seçilen rotanın en kısa olması önemli olabilir, UAV performansını değerlendirdiğimizde bazen daha uzun rotalar, ekonomik açıdan daha avantajlı olabilmektedir.

Yol planlama probleminde, dronun başlangıç ve hedef noktaları arasında geçilecek bağlantı koordinat noktalarının tespit edilip, o noktalar üzerinden geçerek hedefe varması hedeflenmektedir. Geçilecek bağlantı noktaları belirlenirken iki temel kıstas dikkate alınmıştır. Bu kriterler; rota üzerindeki engellerin dikkate alınması ve rotanın en kısa uzunlukta olacak şekilde belirlenmesidir. Ayrıca dronun engellere çarpıp çarpmaması için en fazla iki dron büyüklüğünde yaklaşmasına imkân verilmiştir. Yol planlama problemine çözüm üretmek için kullanılan algoritmalar ile başlangıç aşamasında aday rotalar kullanılarak rastgele üretilmektedir. Başlangıç ve hedef arasındaki mesafeye ve engel sayısına göre değişen haritalarda sabit bağlantı noktaların kullanılması dezavantaj oluşturabilmektedir. Bu bakımdan çalışmada, rota üretimi sırasında başlangıç ve hedef koordinatları arasında üzerinden geçilecek olan bağlantı koordinat noktalarının sayısı dinamik olarak belirlenmektedir. Örneğin; algoritma başlangıç olarak iki bağlantı koordinat noktası ile algoritmaya başlamakta, iki adet bağlantı koordinat noktası ile hedefe çözüm mümkün olmadığı durumda aradaki koordinat nokta sayısını birer arttırarak çözüm aramaya devam etmektedir. Her bir bağlantı koordinat noktası için rastgele enlem ve boylam bilgisi üretilmektedir.

Bu deneysel çalışmada farklı ortamlarda 2 farklı senaryo dikkate alınmıştır. Senaryolar önerilen algoritmaların başarılarını farklı durumlarda inceleyebilmek için farklı sayıda ve pozisyonlarda engeller kullanılarak, aynı başlangıç ve bitiş noktalarıyla yapılandırılmıştır. Bazı senaryolarda, UAV, en iyi yolu bulma konusunda zorlamak için engeller bilinçli bir şekilde Öklid yolu boyunca kümelenmiştir. İlk senaryo, ilk haritada konumlandırılmış farklı boyutlarda 6 statik engelden oluşmaktadır. Öncelikle herhangi bir engele çarpmadan veya engelli bir alana girmeden rota planlaması yapılması

gerektiğinden, engelli alanların belirlenmesi gerekmektedir. Rota üzerindeki engellerin tespiti için, denklemi bilinen bir doğrunun bir noktaya olan uzaklığı formülü kullanılarak Öklid mesafesi belirlenmiştir.

Engeller sırasıyla $R1 = 30, R2 = 20, R3 = 30, R4 = 20, R5 = 20, R6 = 30$ yarıçaplarına sahiptir. İkinci senaryo, daha yoğun kümelenmelerle ikinci haritada konumlandırılmış farklı boyutlarda 7 statik engelden oluşmaktadır. Bu senaryoda, engeller sırasıyla $R1 = 50, R2 = 40, R3 = 60, R4 = 40, R5 = 50, R6 = 50, R7 = 50$ yarıçapına sahiptir. İki senaryoda engel sayısı yüksekten tutulmuş olup, bir fazla engel farkıyla, dronun izlediği yol ve engellere göre değişimi gözlenmiştir. Ortam haritası boyutu tüm senaryolarda $800 \times 1\,000$ olarak değişmeden kalır. Robotun başlangıç noktadan hedef noktaya Öklid mesafelerinin haritalardaki değerleri aynı olup yaklaşık değeri, 921.954 olarak hesaplanmıştır. Yol planlama problemimiz için 16 sezgisel algoritmanın önce tek tek ardından ise farklı sayılarda kombinasyonları ile 30 tekrarlı ve 500 iterasyonluk uçuşmanın sonuçları grafiksel olarak sunulmuştur. Algoritmalarda popülasyon sayısı 50'dir ve Cauchy Ters Kümülatif Dağılımı (Cauchy Inverse Cumulative Distribution) kullanılmıştır. Grafikler üzerinde sezgisel algoritmaların çözdüğü en kısa yol problemi ve en iyi maliyet değeri gösterilmiştir. En kısa yol problemini çözmede kullanacağımız, ASFSSA, BKA, COA, DBO, GA, GWO, HLOA, MSWOA, NCHHO, OMA, PMOMA, PSO, SCSO, WaOA, ZHO, ZOA algoritmaların en iyi değerleri göz önüne alınarak karşılaştırmalar yapılır.

Bu bölümde, UAV yol planlama çözümü detaylı bir şekilde incelenmektedir. Çalışmada kullanılan çevresel modelleme, genetik algoritmalar ve popülasyona dayalı teknikler ile engellerin temsili gibi konular ele alınmıştır. Algoritmanın ayrıntıları sahte kodlar ve denklemlerle gösterilmiştir. MATLAB modelinden elde edilen uçuş verilerini kullanan UAV 'ye özgü planlama vurgulanmıştır. Başlangıç popülasyonuna ilişkin farklılıklar, algoritmaya ait operatör, araç ve parametreler açıklanmıştır. UAV uygulama adımlarıyla birlikte çalışmanın detayları anlatılmıştır. Geliştirilen paketinin ayrıntıları ve sağladığı faydalar sunulmuştur. Bir çevresel model oluşturulması ve yol planlamasından önce çevresel bilgilerin kullanıma uygun hale getirilmesi hesaplamanın büyük ölçüde azaltılmasına yardımcı olmaktadır.

3.3.1 Ele Alınan Senaryolar ve Simülasyon Parametreleri

Önerilen yaklaşımların performansı ve etkinliği, dron için en iyi yolu geliştirmek için test edilmiştir. Her iki senaryo için robotun başlangıç konumu, hedef konumu ve engel sayısı Tablo 3.7’ de sırasıyla belirtilmektedir.

Tablo 3.7 Senaryo 1 ve Senaryo 2 için parametre değerleri

Parametre	Senaryo 1 (Değer)	Senaryo 2 (Değer)
Robot Başlangıç Noktası (x, y)	(200,100)	(200,100)
Robot Hedef Noktası (x, y)	(800,800)	(800,800)
Engel Sayısı	6	7
İterasyon Sayısı	500	500
Popülasyon sayısı	50	50
Tekrar Sayısı	30	30

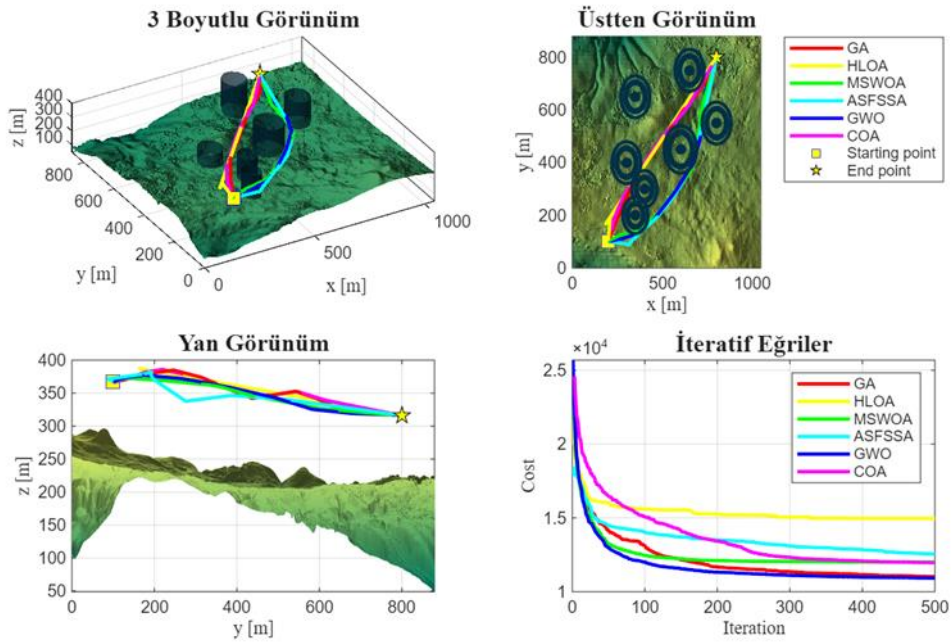
Tüm algoritmaların mevcut iki senaryo için, her bir parametre seti ve 1000 iterasyonla 30’ar kez çalıştırılmıştır. Her iki senaryoda, 1000 iterasyonlu 30 çalıştırma için elde edilen ortalama mesafe, bu mesafelerin standart sapması, en iyi mesafe ve en kötü mesafe değerleri sonucunda izlediği en kısa yolların engel sayısına göre haritalardaki izlediği yollar ve yakınsama eğri grafiklerinin MATLAB görüntüleri elde edilir. Her bir algoritmanın detaylı analizi yapılması için belirli sayıda algoritmaların karşılaştırmaları yapıldıktan sonra gelen sonuçlar analiz edilerek en iyi ve en kötü algoritmalar belirlenmiştir. Daha sonra sonuçlara göre tespit edilen en iyi ve en kötü algoritmalar tekrar aynı yol planlama problemi üzerinden karşılaştırılarak performansları değerlendirilmiştir.

3.3.2 Algoritmaların Karşılaştırıldığı UAV Yol Planlama Sonuçları

Farklı engel sayısına sahip karmaşıklıkta her iki senaryonun, haritalar ve yakınsama eğrileri üzerinden her bir algoritma karşılaştırılmıştır. Farklı karmaşıklıkta haritalar için algoritmaları karşılaştırmak, deneysel değerlendirmemizin ana hedeflerinden biridir. Algoritmaların en kısa yol problem çözümü için haritalardaki izlediği yolun kısıllığı, engellere göre aldığı konum ile problem çözümünün meydana getirdiği problemi çözüm süresi, işlem yoğunluğu vb. gibi durumları kapsayan ve çözümün bize çıkardığı maliyet hakkında genel bir sonuca ulaşmamızı sağlayan yakınsama eğrileri, hangi algoritmanın daha avantajlı olduğu sonucuna ulaştıracaktır. MATLAB ’da hazırlanan kod için 16 algoritma arasından, çalışmalarda genel olarak kullanılan GA algoritması referans olarak seçilip önce farklı sayılarda algoritmalar aralarında karşılaştırılmıştır. GA ile karşılaştırma sonucu, GA’dan iyi, GA’ya benzer ve GA’dan kötü olmak üzere

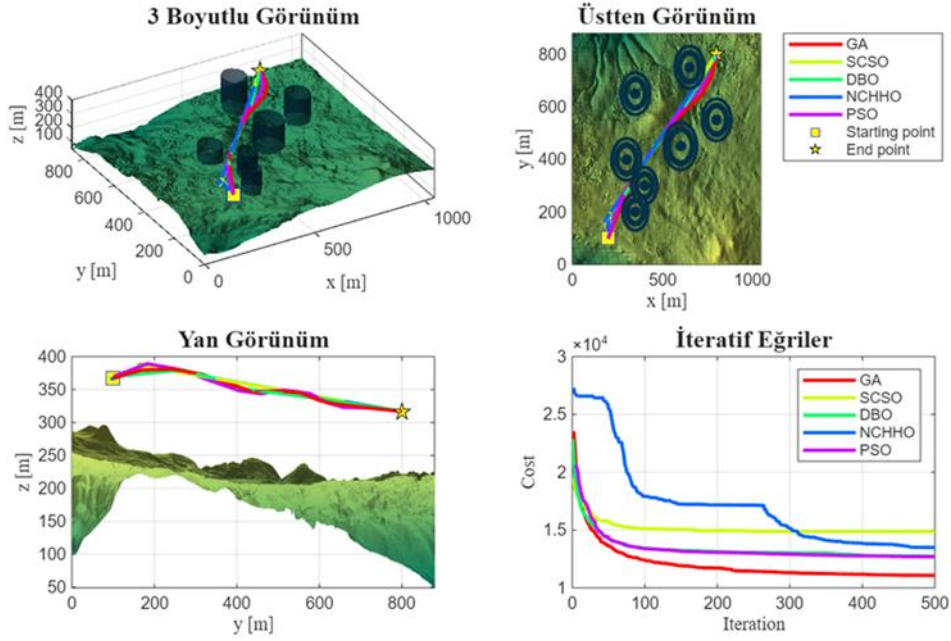
değerlendirme kategorisine giren algoritmalar kendi arasında karşılaştırılarak performansı en iyi ve en kötü algoritmalar belirlenmiştir. Ardından daha sonra test fonksiyonlarının sonuçları ile kıyaslanmak üzere hem test fonksiyonlarında hemde problemin çözümünde ortak olan GWO ile SCSO referans alınarak algoritmaların en kısa yol problemi çözümlerindeki performansları iki farklı gruba ayrılarak ve UAV uçuş sonuçları karşılaştırılmıştır.

3.3.2.1 GA Referans Alınarak Karşılaştırılan Algoritmalar ve Sonuçları



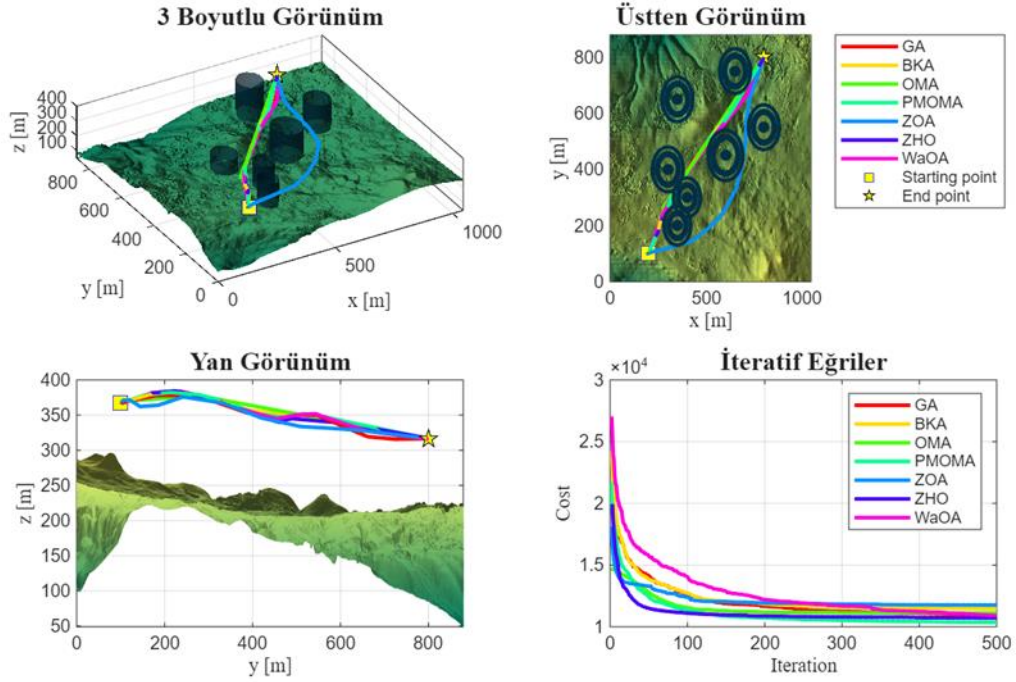
Şekil 3.3 GA-HLOA-MSWOA-ASFSSA-GWO-COA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

Şekil 3.3 ‘te GA-HLOA-MSWOA-ASFSSA-GWO-COA karşılaştırılmasında üç boyutlu ve üstten görünümde COA, GA, HLOA sırasıyla daha kısa yol alırken, MSWOA, GWO, ASFSSA sırasıyla daha uzun bir yol almışlardır. Yandan görünümde ise MSWOA en kısa mesafe alarak hedefe varırken, ASFSSA diğerlerine göre daha uzun yol almıştır. Yakınsama ile ilgili sonuçlara bakıldığında iteratif eğrilerde GWO ’nun maliyetinin daha düşük olduğunu ve HLOA ’nın maliyetinin ise algoritmalar arasında en yüksek olduğunu göstermektedir. GWO’dan sonra en düşük maliyeti GA vererek en kısa mesafede göz önüne alınıp maliyetle birlikte değerlendirilince sonuçlar, GA’nın daha iyi performansı olduğunu desteklemektedir. Sunulan yol problemi sonuçlarında görüldüğü üzere GA ‘nın diğer algoritmadan daha iyi bir performans sergilediği anlaşılmıştır.



Şekil 3.4 GA-SCSO-DBO-NCHHO-PSO karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

Şekil 3.4 ‘te GA-SCSO-DBO-NCHHO-PSO karşılaştırılmasında üç boyutlu görünüm, üstten görünüm ve yan görünümde tüm algoritmalar birbirine çok yakın değerlerde kısa mesafe alarak hedefe varmıştır. Sadece NCHHO kısa mesafede bazı yerlerde kırılmalar yaşadığı için aldığı yol diğerlerine göre uzundur. Yakınsama ile ilgili sonuçlara bakıldığında iteratif eğrilerde GA ’nın maliyetinin daha düşük olduğunu ve NCHHO ’nun maliyetinin ise algoritmalar arasında en yüksek olduğunu göstermektedir. GA’dan sonra en düşük maliyet DBO ve PSO vererek en kısa mesafede göz önüne alınıp maliyetle birlikte değerlendirilince sonuçlar, GA ’nın daha iyi performansı olduğunu desteklemektedir. Sunulan yol problemi sonuçlarında görüldüğü üzere en kötü performansı SCSO ve NCHHO ’nun sergilediği, GA ’nın ise diğer algoritmadan daha iyi bir performans sergilediği anlaşılmıştır.



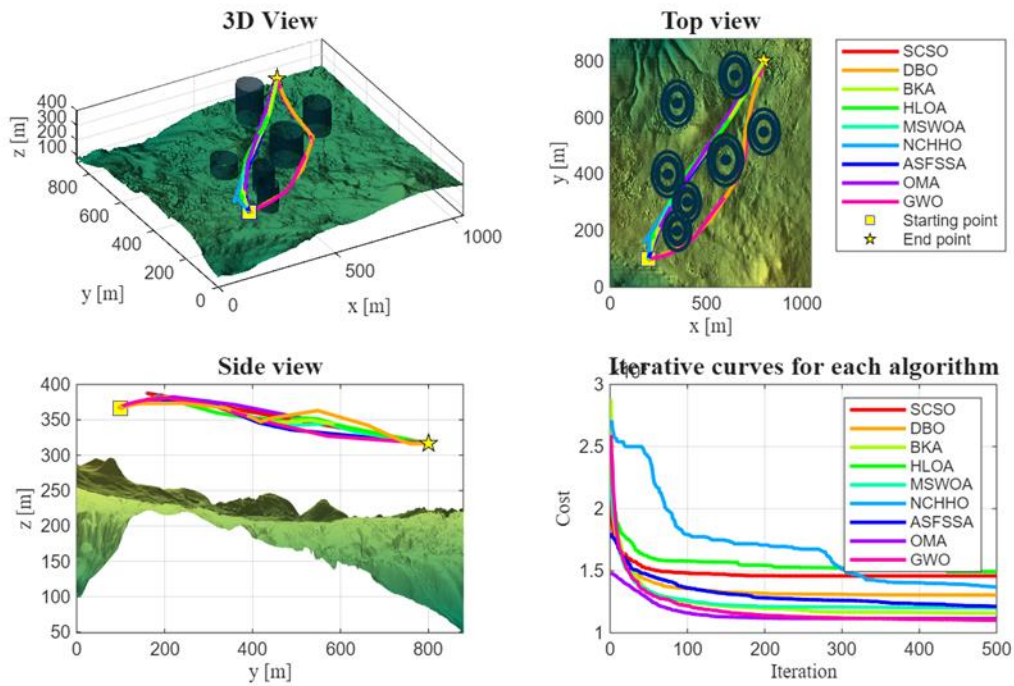
Şekil 3.5 GA-BKA-OMA-PMOMA-ZOA-ZHO-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

Şekil 3.5 'te GA-BKA-OMA-PMOMA-ZOA-ZHO-WaOA karşılaştırılmasında üç boyutlu ve üstten görünümde GA, BKA, OMA, PMOMA, ZHO, WaOA daha kısa yol alırken, ZOA diğer algoritmalarla göre belirgin bir şekilde daha uzun bir yol almışlardır. Yandan görünümde ise tüm algoritmalar benzer şekilde kısa mesafe yol almıştır. İteratif eğrilerde yakınsama ile ilgili sonuçlara bakıldığında PMOMA ve OMA 'nın maliyetinin daha düşük olduğunu ve ZOA'nın maliyetinin ise algoritmalar arasında en yüksek olduğu görülmektedir. ZOA dışında algoritmaların, hem kısa mesafe açısından hemde maliyet açısından birbirine çok yakın değerlere sahip olduğu görülmekte ve bu algoritmaların tamamının iyi bir performansı olduğu desteklemektedir. Sunulan yol problemi sonuçlarında kısa mesafe ve maliyet açısından PMOMA diğer algoritmadan daha iyi bir performans sergilediği anlaşılmıştır.

Deneysel çalışmalarda kullanılan ve kıyas için referans alınan GA sabit kalmak üzere karşılaştırdığımız üç farklı gruptaki tüm algoritmalarından elde edilen çözüm ve eğrilere göre; ilk karşılaştırmada maliyet ve en kısa mesafe göz önünde bulundurduğumuzda en kötü performansı HLOA 'nın ve en iyi performans ise GA 'nın olduğu gözlenmiştir. Kısa mesafelerin birbirine çok yakın olduğu ikinci karşılaştırmada maliyet açısından en kötü performans SCSO ve NCHHO 'nun, en iyi performans ise yine GA 'nın olduğu

gözlemlenmiştir. Son karşılaştırmada ise iteratif eğrilerin birbirine çok yakınsamasından dolayı maliyetlerinin benzer değer aldığı veya aralarında çok az fark olduğu gözlemlenmiştir. Bu yüzden belirgin bir şekilde uzun mesafe kat ederek ZOA en kötü performansı göstermiştir. Maliyet açısından en iyi performansı ise PMOMA göstermiştir. Tüm bu deney sonuçlarından yola çıkarak, GA referans alınarak yapılan karşılaştırmada en kötü performans HLOA 'nın olup en iyi performans ise PMOMA sonucuna ulaşılmıştır.

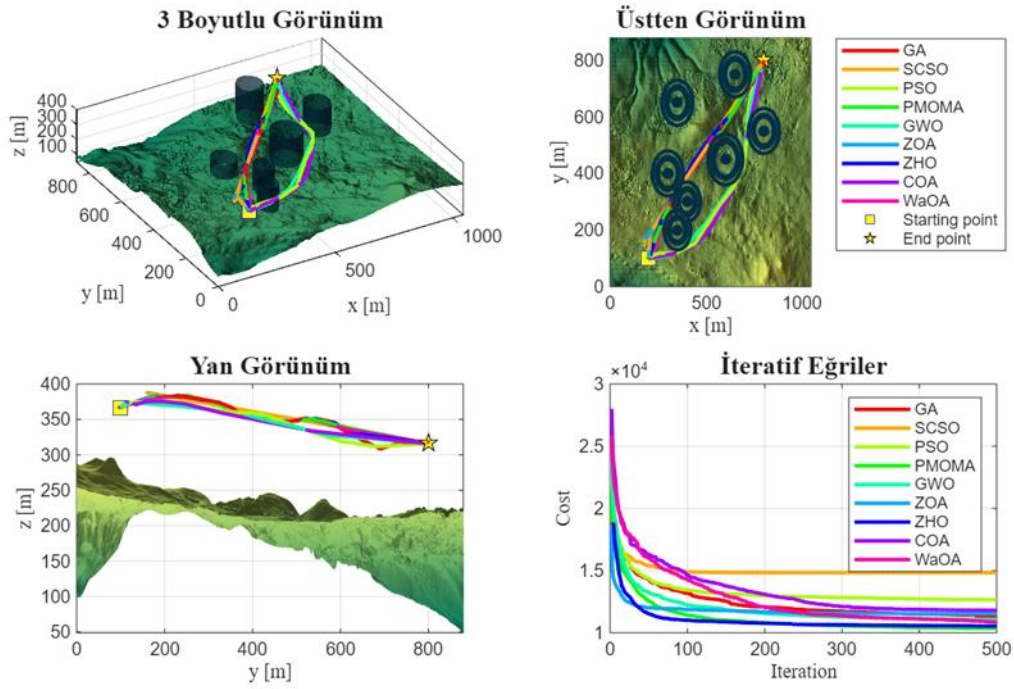
3.3.2.2 GWO ve SCSO Referans Alınarak Karşılaştırılan Algoritmalar ve Sonuçları



Şekil 3.6 GWO-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

GWO-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA karşılaştırılmasında Şekil 3.6 'da, üç boyutlu ve üstten görünümde DBO ve GWO diğerlerinden daha uzun yol alırken SCSO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA daha kısa veya yaklaşık değerlerde kısıklıkta yol almışlardır. Yandan görünümde ise DBO diğerlerine göre daha uzun yol almıştır. Yakınsama ile ilgili sonuçlara bakıldığında iteratif eğrilerde OMA 'nın maliyetinin daha düşük olduğunu ve HLOA 'nın maliyetinin ise algoritmalar arasında en yüksek olduğunu göstermektedir. Sunulan yol problemi sonuçlarında

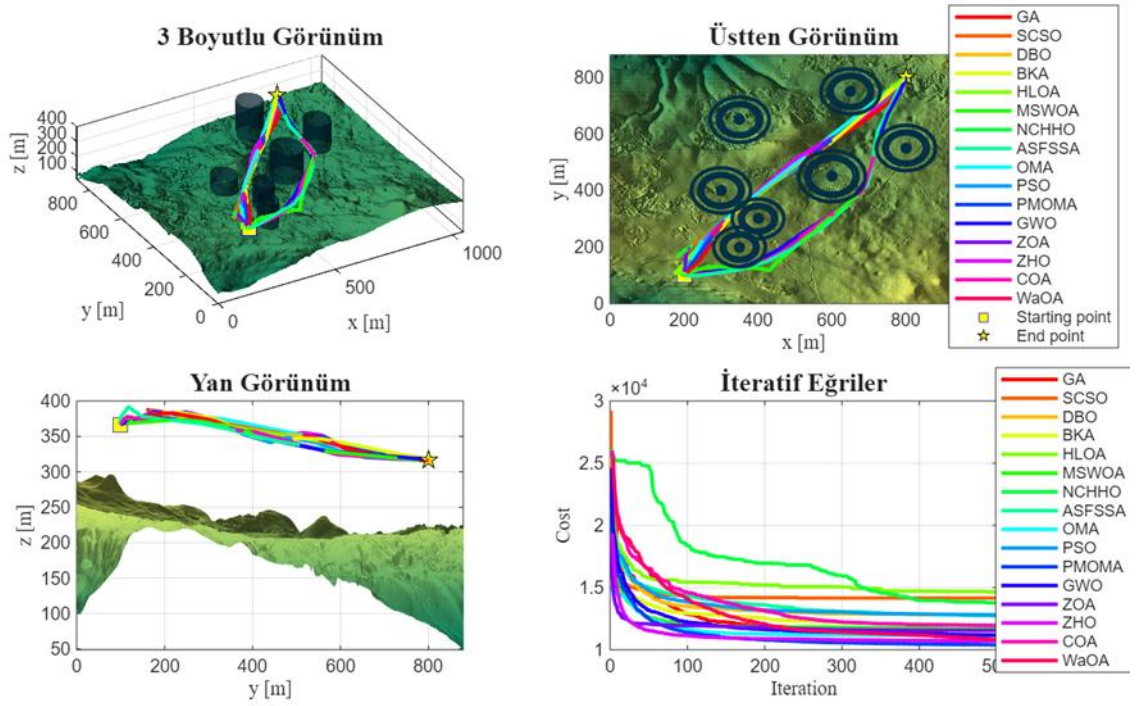
görüldüğü üzere OMA 'nın diğer algoritmadan daha iyi bir performans sergilediği gözlenmiştir.



Şekil 3.7 GWO-SCSO-GA-PSO-PMOMA-ZOA-ZHO-COA-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

Şekil 3.7 'de, GWO-SCSO-GA-PSO-PMOMA-ZOA-ZHO-COA-WaOA karşılaştırılmasında üç boyutlu ve üstten görünümde GWO, PSO, COA diğerlerinden daha uzun yol alırken SCSO, GA, PMOMA, ZOA, ZHO, WaOA daha kısa veya yaklaşık değerlerde kısıklıkta yol almışlardır. Yandan görünümde ise aldıkları mesafeler birbirine benzerdir. Yakınsama ile ilgili sonuçlara bakıldığında iteratif eğrilerde PMOMA 'nın maliyetinin daha düşük olduğunu, ardından maliyet artarak ZHO, WaOA, GWO takip etmektedir. Maliyetinin algoritmalar arasında en yüksek olduğunu ise SCSO göstermektedir. Sunulan yol problemi sonuçlarında görüldüğü üzere sırasıyla PMOMA ve ZHO 'nun diğer algoritmadan daha iyi bir performans sergilediği gözlenmiştir.

En kısa yol problemi çözümünde GWO ve SCSO referans alınarak karşılaştırılan ilk grup GWO-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA, ikinci grup GWO-SCSO-GA-PSO-PMOMA-ZOA-ZHO-COA-WaOA şeklindedir. Problem çözümü için deneysel çalışma sonucunda elde edilen çözümlere göre ilk karşılaştırma grubunda HLOA en kötü, OMA en iyi performans sergilemiştir. İkinci karşılaştırma grubunda ise COA en kötü, PMOMA ve ZHO en iyi performansını sergilemiştir.



Şekil 3.8 GA-SCSO-DBO-BKA-HLOA-MSWOA-NCHHO-ASFSSA-OMA-PSO-PMOMA-GWO-ZOA-ZHO-COA-WaOA karşılaştırılması ile yol planlama problemi için elde edilen çözümler ve yakınsama eğrileri

MATLAB tabanlı olarak geliştirilen uygulama ile çeşitli algoritmik parametreleri değiştirerek yeni haritalar oluşturması, rotaları ve yöntemleri seçmesi ve sonuçları grafikler aracılığıyla incelemesi sonucunda ve bir planlama işlemi sonrasında, grafiksel analizler ve oluşturulan rotalar test edildiğinde ZHO ve PMOMA yol problemimiz için en ideal algoritmalar olduğu sonucuna varıldı.

Bu tez çalışmasında, UAV için insan kontrolü olmadan, durağan engellerin ve durağan bir hedefin bulunduğu bir ortamda bölgesel yol planlaması için senaryoları farklı ortamlar oluşturulmuştur. Senaryo ortamlarında en iyi performans gösterecek olan metasezgisel yöntemler önerilmiştir. Bu senaryolarda sırasıyla, güncel olarakda yol planlama problemlerinde çokça tercih edilen ve tek referans algoritma olarak belirlenen GA, diğerinde ise test fonksiyonlarıyla karşılaştırma yapabilmek için ortak olan GWO ve rastgele tercih edilerek SCSO ikilisi birlikte sabit olmak şartıyla algoritmaları kullanılmaktadır. Bu yöntemlerde, eğer UAV ile hedef arasındaki en kısa mesafenin olduğu doğrultuda bir engel yoksa, UAV bu doğrultuda hareket etmektedir. Eğer bu doğrultuda bir engel varsa robot gideceği noktayı bulmak için tüm algoritmalarını çalıştırmaktadır. Burada kullanılan amaç fonksiyonu engellerden uzaklaşmayı ve hedefe yaklaşmayı sağlayan yapılar içermektedir.

CEC2017 test fonksiyonunda yapılan analizler sonucu en iyi değeri IGWO verirken, CEC2022 fonksiyonlarında en iyi değeri GWO vermiştir. En kısa yol problemleri çözümünde IGWO ve GWO daha kullanışlı olduğu yakınsama eğrileriyle desteklenmektedir.

MATLAB tabanlı olarak geliştirilen uygulama ile çeşitli algoritmik parametreleri değiştirerek yeni haritalar oluşturması, rotaları ve yöntemleri seçmesi ve sonuçları grafikler aracılığıyla incelemesi sonucunda ve bir planlama işlemi sonrasında, grafiksel analizler ve oluşturulan rotalar test edildiğinde ZHO ve PMOMA yol problemimiz için en ideal algoritmalar olduğu belirlenmiştir.

Gelecekteki çalışmalarda, geliştirilen sistemi daha sonraki gelişmelerde kullanılabilir ve geliştirilebilir. Birden fazla hedef noktasıyla uyarlanabilen dinamik ortamlarda birden fazla UAV için yol planlaması yapılabilir. Ulaşım ve askeri savunma sistemlerinin, ticari dağıtım ve faaliyet sistemlerinin mevcut kullanımı da önerilen sisteme dahil edilebilir. Kontrol parametreleri çeşitlendirilerek yeni alanlarda deney tasarımı yapılabilir. Önerilen yaklaşımla daha farklı senaryolarda literatürde yer alan diğer yöntemlerle kıyaslanabilir.

KAYNAKLAR

- Abdollahi Dehkordi, A., Sadiq, A. S., Mirjalili, S., & Ghafoor, K. Z. (2021). Nonlinear-based Chaotic Harris Hawks Optimizer: Algorithm and Internet of Vehicles Application. *Applied Soft Computing*, 109, 107574.
- Abebe Ashetehe, A., Shewarega, F., Bantayirga, B., & Biru, G. (2024). Optimal design of off grid hybrid system using a new zebra optimization algorithm. *Scientific Reports*, 14, 29255. <https://doi.org/10.1038/s41598-024-29255-3>
- Abualigah, L., Diabat, A., & Mirjalili, S. (2021). Pelican Optimization Algorithm: A New Nature-Inspired Metaheuristic for Solving Optimization Problems. *Knowledge-Based Systems*, 227, 107–176. <https://doi.org/10.1016/j.knosys.2021.107176>
- Abu-Mouti, F., & El-Hawary, M. E. (2012). Modified teaching–learning-based optimization for optimal placement of distributed generation in distribution systems. *Electric Power Systems Research*, 92, 64–72.
- Ahmed, H., and Glasgow, J., 2012. *Swarm Intelligence : Concepts , Models and Applications* Technical Report 2012-585. Technical Report, 585, 1–50.
- Akay, B., 2009: Nümerik Optimizasyon Problemlerinde Yapay Arı kolonisi (Artificial Bee Colony) Algoritmasının Performans Analizi, Erciyes Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Doktora Tezi.
- Akduman, B. (2010). Evrimsel algoritmalar ile elektrik dağıtım şebekelerinin restorasyonu.
- Akinola, O., ... (2025). Multi Strategy Horned Lizard Optimization Algorithm for complex optimization and advanced feature selection problems. *Journal of Big Data*.
- Akyol, S. ve Alataş, B., 2012: Güncel Sürü Zekası Optimizasyon Algoritmaları, Nevşehir Üniversitesi Fen Bilimleri Enstitüsü Dergisi, Cilt.1, sy.36-50
- Al Shourbaji, I., Kachare, P., Fadlelseed, S., Jabbari, A., Hussien, A. G., Al Saqqar, F., Abualigah, L., & Alameen, A. (2023). Artificial ecosystem based optimization with dwarf mongoose optimization for feature selection and global optimization problems. *International Journal of Computational Intelligence Systems*, 16, Article 102.
- Alataş, B. (2007). Kaotik Haritalı Parçacık Sürü Optimizasyonu Geliştirme.
- Altıparmak, F., 1996: Genetik Algoritma ile Haberleşme Şebekelerinin
- Aribowo, W., Rahmawati, N., & Setiawan, H. (2024). PID controller optimization using SCSO for DC motor systems. *TELKOMNIKA*, 22(1), 112–121.
- Arslan, H. Genetik, Parçacık Sürü ve Nelder-Mead Simpleks Algoritmalarının Fırlatma Yörüngesi Optimizasyonu Üzerinden Karşılaştırılması.
- Artificial Ecosystem Based Optimization with Dwarf Mongoose Optimization for Feature Selection and Global Optimization Problems (*International Journal of Computational Intelligence Systems*, 2023).

- Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers & Structures*, 169, pp.1–12.
- Babayiğit, A., & Özdemir, M. (2012). A new artificial bee colony algorithm for solving optimization problems. *International Journal of Computer Applications*, 48(14), 38–43.
- Bansal, J. C., Singh, P. K., ve Pal, N.R. (2019). *Evolutionary and Swarm Intelligence Algorithms*. İsviçre: Springer.
- Behesti, Z. and Shamsuddin, S. M. H, “A review of population-based meta-heuristic algorithm”, *International Journal of Advanced in Soft Computing and its Applications*, 5(1): 1-35 (2013).
- Bingöl, H., & Yıldırım, M. (2021). Global optimizasyon için sürü tabanlı bir yaklaşım salp sürü algoritması. *Fırat Üniversitesi Fen Bilimleri Dergisi*, 33(1), 51-59.
- Blum, C., Puchinger, J., Raidl, G. R. and Roli, A., “A brief survey on hybrid metaheuristics”, in *Bioinspired Optimization Methods and their Applications – Proceeding of the 4th International Conference on Bionspired Optimization Methods and their Applications, BIOMA 2010*, 3-16 (2010).
- Boğar, E. (2016). Tek ve çok amaçlı robot yol planlama problemi için hibrit bir optimizasyon yöntemi (Doctoral dissertation, Pamukkale Üniversitesi).
- Bozkurt, F., 2004: Genetik Algoritma Yöntemini Kullanarak Yarı İletken
- Can, Ü. (2014). Yerçekimsel arama algoritmasıyla nicel birliktelik kurallarının keşfi, Yüksek Lisans tezi, Tunceli Üniversitesi, Tunceli.
- Chawla M. ve Duhan M., 2015, Bat algorithm: a survey of the state-of-the-art, *Applied Artificial Intelligence*, 29 (6), 617-34.
- Chen, A., Wang, L., & Zhao, M. (2024). Improved sand cat swarm optimization for complex engineering design. *Artificial Intelligence Review*, 57(4), 1–20.
- Cheng, M. Y., & Nur Sholeh, M. (2023), *Knowledge-Based Systems*, 279, 110939
- Cheng, M. Y., & Sholeh, M.N. (2023). Optical microscope algorithm: A new metaheuristic inspired by microscope magnification for solving engineering optimization problems. *Knowledge-Based Systems*, 279, 110939.
- Chopra, N., & Ansari, M. M. (2022). Golden Jackal Optimization: A Nature-Inspired Metaheuristic. *Engineering Applications of Artificial Intelligence*, 114, 105–126.
- Clayton N, Emery N. Corvide cognition. *Curr Biol* 2005;15:R80–1.
- CORNE, D., DORİGO, M., GLOVER F., 1999, *New Ideas in Optimization*, McGraw-Hill, London UK, 0-07-709506-5.
- Cui, L., & An, Y. (2025). Multi-strategy enhanced Black Winged Kite Algorithm and its applications in engineering design. *Engineering Optimization*, 57(4), 789–805.
- Cui, R., & An, Z. (2025). Multi-strategy Enhanced Black Kite Algorithm for Constrained Engineering Optimization Problems. *Academic Journal of Computing & Information Science*, 8(5), 19 37.

- Çelik, Y., & Karadeniz, A. T. (2020). Whale optimization algorithm for numerical constrained optimization. *Academic Platform Journal of Engineering and Science*, 8(2), 251–257.
- D. E. Goldberg, 1989, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison - Wesley, Boston.
- D. Karaboga, “An idea based on Honey Bee Swarm for Numerical Optimization,” Technical Report TR06, Erciyes University, no. TR06, pp. 10, 2005.
- Dehghani, M., et al. (2020). A Review on Beetle Antennae Search Optimization Algorithm and Its Applications. *Artificial Intelligence Review*, 53, 1–30.
- Dehghani, M., Trojovská, E., & Trojovský, P. (2022). Zebra Optimization Algorithm: A new bio inspired optimization algorithm for solving optimization problems. *IEEE Access*, 10, 49445–49473.
- Dehghani, M., Trojovská, E., & Trojovský, P. (2023). Coati Optimization Algorithm: A new bio inspired metaheuristic algorithm for solving optimization problems. *Knowledge Based Systems*, 259, 110011.
- Dehkordi, A. A., Sadiq, A. S., Mirjalili, S., & Ghafoor, K. Z. (2021). Nonlinear based Chaotic Harris Hawks Optimizer: Algorithm and Internet of Vehicles application. *Applied Soft Computing*, 109, 107574
- Dodoloğlu, A. (2017). Arıcılığın temel esasları. Erzurum: Atatürk Üniversitesi Ziraat Fakültesi Ofset Tesisi.
- Dorigo M., Dicaro G. & Gambardella L. M., (1999). Ant Algorithms For Discrete Optimization. *Artificial Life* 5, 137-172.
- DORIGO, M., STÜTZLE, T., 2004, Ant Colony Optimization, MIT Press, London, 0-262-04219-3.
- E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” *Numer. Math.*, 1959.
- Emam, M. M., Hosney, M. E., Mostafa, R. R., & Houssein, E. H. (2025). Multi-strategy horned lizard optimization algorithm for complex optimization and advanced feature selection problems. *Journal of Big Data*, 12, Article 148.
- Emam, M. M., Hosney, M. E., Mostafa, R. R., Houssein, E. H., & ... (2025). Multi strategy Horned Lizard Optimization Algorithm for complex optimization and advanced feature selection problems. *Journal of Big Data*, 12, Article 148.
- Frisch, K. von. (1993). The dance language and orientation of bees. Cambridge, MA: Harvard University Press.
- Fukuyama, Y. (2008). Fundamentals of particle swarm optimization techniques.
- Fukuyama, Y. (2008). Fundamentals of particle swarm optimization techniques. In K. Y. Lee & M. A. El-Sharkawi (Eds.), *Modern heuristic optimization techniques: Theory and applications to power systems* (pp. 71–87). Hoboken, NJ: John Wiley & Sons.
- Garip, Z. (2018). Mobil Robotların Yol Planması için Meta-Sezgisel Hibrit Algoritmalar Geliştirilmesi ve Uygulanması, (Basılmamış Doktora Tezi), Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya.
- Garip, Z. B., Atali, G., Karayel, D. and Ozkan, S. S. (2018). Path Planning for Multiple Mobile Robots in Static Environment using Hybrid Algorithm, 2nd International

- Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), 1-4.
- Gezici, H., & Livatyali, H. (2022). Chaotic Harris hawks optimization algorithm. *Journal of Computational Design and Engineering*, 9(1), 216–245.
- Goss, S., Aron, S., Deneubourg, J.L., Pasteels, J.M., 1989, Self-organized shortcuts in the Argentine ant, *Naturwissenschaften*, 76, 579-581.
- Graf, B., Hostalet Wandosell, J. M., & Schaeffer, C. (2001). Flexible path planning for nonholonomic mobile robots. In *Proceedings of the Fourth European Workshop on Advanced Mobile Robots (EUROBOT '01)* (pp. 199–206).
- Guo, L., & Gu, W. (2024). PMSOMA: Optical microscope algorithm based on piecewise linear chaotic mapping and sparse adaptive exploration. *Scientific Reports*, 14, 20849.
- Gupta, S., Deep, K. (2019). Improved grey wolf optimizer for global optimization problems. *Engineering with Computers*, 35(1), 235–264.
- Gurson, A. P. (2000). Simplex search behavior in nonlinear optimization (Undergraduate Honors Thesis). Department of Computer Science, College of William & Mary, Williamsburg, VA.
- Hamad ve ark., (2025). Horned Lizard Defense Tactics Optimization Algorithm for Precise Identification of PEMFC Parameters. *Fuel Cells*.
- He, Q., & Yuhan, Z. (2022). Improved Sand Cat Swarm Optimization with multi-strategy cooperative. *Science Discovery*, 10(6), 482–488.
- Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, 97, 849–872. <https://doi.org/10.1016/j.future.2019.02.028>
- Huang, J., Li, X., & Zhou, F. (2024). A multi-strategy enhanced SCSO for global optimization. *Biomimetics*, 9(2), 45–60.
- I. Rechenberg, “Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution,” Ph.D. dissertation, Technical University of Berlin, Department of Process Engineering, 1971.
- I. Rechenberg, Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, 1965.
- J. A. Nelder, R. Mead, 1965, A Simplex Method for Function Minimization, *The Computer Journal*, 7(4), 308-313.
- Jiang, X., Li, S., & Han, Y. (2017). Beetle Antennae Search: A New Simple Metaheuristic Algorithm for Optimization. Preprint, arXiv:1710.10724.
- Karaboğa, D. (2005). An idea based on honey bee swarm for numerical optimization (Technical Report No. TR06). Kayseri: Erciyes University.
- Karaboğa, D. ve Akay, B., 2009 A survey: Algorithms Simulating Bee Swarm Intelligence, *Artificial Intelligence Review*, pp.61-85.
- Karaboğa, D., & Baştürk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, 8(1), 687–697.

- Karcı, A. (2012). A new metaheuristic algorithm based chemical process: Atom Algorithm (p:85). Proc. 1st International Eurasian Conference on Mathematical Sciences and Applications, September 03-07, Pristina, Kosova.
- Kartal, B., 2015: Yapay Arı Kolonisi Algoritması ile Finansal Portföy Optimizasyonu, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü Doktora Tezi.
- Kaynak, A. (2024). Swarm-based optimization algorithms: Trends and applications. *International Journal of Advanced Computational Methods*, 12(3), 45–67.
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1942–1948.
- Kennedy, J. (2010). Particle Swarm Optimization. *Encyclopedia of Machine Learning*. Amerika Birleşik Devletleri: Springer.
- Keskintürk, T., 3–5 Temmuz 2006. "Diferansiyel gelişim algoritması", 214–217, YA/EM 2006 – Yöneylem Arastırması / Endüstri Mühendisliği – XXVI. Ulusal Kongresi – Kocaeli. (diferansiyel gelişim algoritması pdf)
- Khishe, M., & Mosavi, M. R. (2020). Chimp Optimization Algorithm. *Expert Systems with Applications*, 149, 113338.
- Khishe, M., et al. (2022). Improved Chimp Optimization Algorithm for Complex Problems. *Expert Systems with Applications*, 191, 116–128.
- Khudeer, H., & Erbay, H. (2021). Hibrit karga-genetik algoritmasını kullanarak 3 boyutlu kutu paketleme problemi çözme. *Veri Bilimi*, 4(1), 8-22.
- Kumar, S. V., & Bhasker, R. (2025). Coati optimization algorithm for enhanced energy management in AC/DC hybrid microgrids. *Panamerican Mathematical Journal*, 35(2s). <https://doi.org/10.1016/j.pmj.2025.02.001>
- Kunchev, V., Jain, L., Ivancevic, V. and Finn, A. (2006). Path Planning and Obstacle Avoidance for Autonomous Mobile Robots: A Review, *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 537-544.
- Latombe, J. C. (1991). Robot motion planning. Norwell, MA: Kluwer.
- Latombe, J. C., Robot motion planning, Norwell MA: Kluwer, (1991).
- Li, S., & Jiang, X. (2019). An Improved Beetle Antennae Search Algorithm With Dynamic Step Size. *Applied Soft Computing*, 85, 105–118.
- Liu, C., Yan, X., Liu, C., Wu, H., 2011. The Wolf Colony Algorithm and Its Application, *Chinese Journal of Electronics*, 20, 212-216.
- Liu, K., Dai, Y., & Liu, H. (2025). Improvement of Dung Beetle Optimization Algorithm Application to Robot Path Planning. *Applied Sciences*, 15(1), 396.
- Liu, X., Zhang, B., & Chen, Q. (2024). Adaptive SCSO for feature selection in classification problems. *Sensors*, 24(3), 2101.
- Liu, Y. (2025). An improved zebra optimization algorithm. *International Journal of Engineering Research and Management (IJERM)*, 12(03).
- Long, W., Wu, T., Jiao, J., Tang, M., & Xu, M. (2017). Refined grey wolf optimizer for constrained engineering optimization problems. *Computers & Industrial Engineering*, 114, 199–213.

- Long, W., Jiao, J., Liang, X., & Tang, M. (2018). An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. *Engineering Applications of Artificial Intelligence*, 68, 63–80.
- Ling, Y., Zhou, Y., & Luo, Q. (2017). Lévy flight trajectory-based whale optimization algorithm for global optimization. *IEEE Access*, 5, 6168–6186.
- MathWorks. (2022). Pelican Optimization Algorithm (POA). MATLAB Central File Exchange.
- MathWorks. (2023). Golden Jackal Optimization Algorithm. MATLAB Central File Exchange.
- MATLAB Version 3.8 User's Manual, retrieved January 25, 2010,
- MDPI. (2024). Application of GJO in Multi-Level Image Thresholding. *Sensors*, 24(3573), 1–22.
- Mehmood, K., Chaudhary, N. I., Khan, Z. A., Cheema, K. M., Raja, M. A. Z., Milyani, A. H., & Azhari, A. A. (2022). Dwarf mongoose optimization metaheuristics for autoregressive exogenous model identification. *Mathematics*, 10(20), 3821.
- Mirjalili, A. (2016). Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems*, 96, 120–133.
- Mirjalili, S. (2016). SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems*, 96, 120–133.
- Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, 95, 51–67.
- Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, 95, 51–67.
- Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, 69, 46–61.
- Mnubi, S. A., (2016). Motion Planning and Trajectory for Wheeled Mobile Robot, *International Journal of Science and Research*, 5(1), 1064-1068.
- Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, 96, 120–133.
- Mirjalili, S., et al. (2023). *Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems*. *Knowledge-Based Systems*, 275, 110642.
- Moustafa, G., El Rifaie, A. M., Smaili, I. H., Ginidi, A., Shaheen, A. M., Youssef, A. F., & Tolba, M. A. (2023). An enhanced Dwarf Mongoose Optimization Algorithm for solving engineering problems. *Mathematics*, 11(15), 3297.
- Murty, K. G., *Optimization Models For Decision Making*, vol. 1, Internet Edition, Chapter 1: Models for Decision Making, 1-18, 2003.
- Nadimi-Shahraki, M. H., Taghian, S., Mirjalili, S., & Zamani, H. (2021). An improved grey wolf optimizer for solving engineering problems. *Expert Systems with Applications*, 166, 113917.
- Oke, A. O., Adio, M. O., Oladosu, J. B., & Awodoye, O. O. (2025). A comparative analysis of zebra optimization algorithm and chaotic sinusoidal zebra optimization algorithm for video forgery detection system. *LAUTECH Journal of Engineering and Technology*.

- Ouyang, C., Qiu, Y., & Zhu, D. (2021). Adaptive Spiral Flying Sparrow Search Algorithm (ASFSSA): Development and applications. *Scientific Programming*, 2021, Article ID 6505253.
- Özdemir, S., Sacar, Ö., & Özcan, E. (2021). Dijkstra algoritması kullanılarak ipek yolu koridorları arasında en kısa ulaştırma güzergâhının belirlenmesi. *Demiryolu Mühendisliği*, (13), 97-105.
- Parsopoulos, K. E., ve Vrahatis, M. N. (2010). Particle Swarm Optimization and Intelligence: Advances and Applications. Amerika Birleşik Devletleri: Information Science Publishing.
- Peng, X., Li, Y., et al. (2025). Multi-strategy enhanced Sand Cat Swarm Optimization algorithm and its engineering applications. *Scientific Reports*.
- Peraza Vázquez, H., Peña Delgado, A., Merino Treviño, M., Morales Cepeda, A. B., & Sinha, N. (2024). A novel metaheuristic inspired by horned lizard defense tactics. *Artificial Intelligence Review*, 57, 59.
- Pohlheim, H., 2006: GEATbx Genetic and Evolutionary Algorithm Toolbox for MATLAB Version 3.8 User's Manual, retrieved January 25, 2010, from <http://www.geatbx.com>, December.
- Pohlheim, H., 2006: GEATbx Genetic and Evolutionary Algorithm Toolbox for
- Price, K. V., Storn, R. M. and Lampinen, J. A., Differential Evolution: A Practical Approach to Global Optimization, 28 (2005).
- Price, K. V., Storn, R. M., & Lampinen, J. A. (2005). Differential Evolution: A Practical Approach to Global Optimization (1st ed.). Springer.
- Ramli, N., & Fauzi, "Black-winged kite algorithm: a nature-inspired meta heuristic for solving benchmark functions and engineering problems." *Artificial Intelligence Review*, (2024).
- Ramli, N., & Fauzi, M. (2024). Black Winged Kite Algorithm: A swarm intelligence approach inspired by raptor behavior. *Journal of Computational Optimization*, 15(2), 112–130.
- Rashedi, E., Nezamabadi-pour, H., Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Inform. Sciences* 179, 2232–2248.
- Researcher Manipal. (2023). IGJO: An Improved Golden Jackal Optimization Algorithm Using Local Search.
- Rizk Allah, R. M., & Hassanien, A. E. (2023). A comprehensive survey on the sine–cosine optimization algorithm. *Artificial Intelligence Review*, 56(6), 4801–4858.
- Seyyarer, E., Yeşil, T. Y., & Öztunç, A. (2024). Adaptif lineer toplama ağırlıklarının optimizasyonu: Kum Kedisi Sürü ve Benekli Sırtlan algoritmalarının karşılaştırması. *Journal of Computer Science*, 9(2), 151–168.
- Seyyedabbasi, A., & Kiani, F. (2022). Sand cat swarm optimization algorithm. *IAENG International Journal of Computer Science*, 50(2), 1–10.
- Seyyedabbasi, A., & Kiani, F. (2022). Sand Cat Swarm Optimization algorithm.
- Sharifi, M., Mirjalili, S., & Kaveh, A. (2021). Population based Multi Objective Moth Algorithm (PMOMA) for solving multi-objective optimization problems. *PeerJ Computer Science*, 7, e462.

- Storn, R. and Price, K., “Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces”, *J. Glob. Optim.*, 11(4): 41-359 (1997). Sweden, pp. 199-206, (2001).
- Tan, L., Liu, X., Liu, D., et al. (2024). An Improved Dung Beetle Optimizer for Random Forest Optimization. *arXiv preprint*.
- Tang, K. S., Man, K. F., Kwong, S., and He, Q., 1995: Genetic Algorithms
- Torres, L. G., Martins, H. G., Matsunaga, F. M., 2008: Genetic Algorithm to
- Trojovská, E., Dehghani, M., & Trojovský, P. (2022). Zebra Optimization Algorithm: A new bio inspired optimization algorithm for solving optimization problems. *IEEE Access*, 10, 49445–49473.
- Trojovský, P., & Dehghani, M. (2023). A new bio inspired metaheuristic algorithm for solving optimization problems based on walruses behavior. *Scientific Reports*, 13, Article 8775.
- Trojovský, P., & Dehghani, M. (2023). Walrus Optimization Algorithm: A new bio inspired metaheuristic algorithm. *Scientific Reports*, 13, Article 35863.
- Tuncer, A. (2007). Genetik Algoritmalar için Uzak Sanal Laboratuvar. Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik ve Bilgisayar Eğitimi.
- Türkşen, Ö., & Akgün, F. (2018). Genetik-Simpleks hibrit algoritması ile doğrusal olmayan regresyon model parametrelerinin nokta tahmini. *İstatistikçiler Dergisi: İstatistik ve Aktüerya*, 11(2), 81-92.
- Venkatesan, V. K., Sujatha, V., et al. (2025). Metaheuristic-AI enhanced deep learning network optimized with Sand Cat Swarm Optimization for oral cancer diagnosis. *International Journal of Advances in Signal and Image Sciences*.
- W. Spendley, G. R. Hext, F. R. Himsforth, 1962, Sequential Application of Simplex Designs in Optimization and Evolutionary Operation, *Technometrics*, 4(4), 441-461.
- Wahab, M. N. A., Nefti-Meziani, S. and Atyabi, A., (2020). A Comparative Review on Mobile Robot Path Planning: Classical or Meta-Heuristic Methods? , *Annual Reviews in Control*, 50, 233-252.
- Walrus Optimization Based Adaptive Virtual Inertia Control for Frequency Regulation in Islanded Microgrids. (2024). *Electronics*, 14(20), 3980.
- Wang, Y. (2025). Improved Dung Beetle Optimization Algorithm Integrated with Hybrid Strategies. *International Journal of Engineering Research And Management (IJERM)*, 12(03), 85 98.
- Wei, P., Zou, R., Dong, G., Qin, W., Gan, J., & Su, O. (2025). An adaptive traditional Chinese herbal medicine image recognition model via ED-HLOA-optimized DenseNet201. *EAI Endorsed Transactions on AI and Robotics*, 4(1).
- Wu, X., Ding, Y., Wang, L., & Zhang, H. (2025). A Multi Strategy Adaptive Coati Optimization Algorithm for Constrained Optimization Engineering Design Problems. *Biomimetics*, 10(5), 323.
- Xing, X. (2023). Multi-strategy whale optimization algorithm for image and signal processing applications.

- Xing, X. (2023). Multi-strategy whale optimization algorithm for image and signal processing applications.
- Xing, Z. (2023). A Mixed-Strategy-Based Improved Whale Optimization Algorithm for Multilevel Thresholding Image Segmentation. *Electronics*, 12(12), 2698.
- Xue, J., & Shen, B. (2023). Dung beetle optimizer: A new meta heuristic algorithm for global optimization. *The Journal of Supercomputing*, 79, 7305–7336. <https://doi.org/10.1007/s11227-022-04959-6>
- Yang, X.-S. (2010a). A new metaheuristic bat-inspired algorithm. In C. A. Coello Coello, A. H. Aguirre, E. L. Palafox, & R. L. Castro (Eds.), *Nature inspired cooperative strategies for optimization (NICSO 2010)* (Vol. 284, pp. 65–74). Springer, Berlin, Heidelberg.
- Yang, X. S. (2014). *Nature-inspired optimization algorithms*. Elsevier.
- Yıldızdan, G. (2021). Büyük ölçekli sürekli optimizasyon problemleri için yarasal algoritması tabanlı hibrit yöntemlerin geliştirilmesi.
- Yunnan Üniversitesi. (2021). *Adaptive Spiral Flying Sparrow Search Algorithm (ASFSSA): Development and applications*. Yunnan University Press.
- Yunnan Üniversitesi. (2021). Improved Salp Swarm Algorithm for Optimization Problems. *MDPI Symmetry*, 13(6), 1092.
- Zhang, H., & Liu, F. (2025). Multi-Strategy Improved Dung Beetle Optimization Algorithm and Its Applications. *Computational Intelligence and Robotics*, 9(5), 291.
- Zhang, Y., & Li, Q. (2018). BAS: Beetle Antennae Search Algorithm for Optimization Problems. *International Journal of Computational Intelligence Systems*, 11(1), 1–20.
- Zuo, X. B., Cai, Z. X. and Sun, G. R., “Non-smooth environment modeling and global path planning for mobile robots”, *Journal of Central South China of Technology*, vol:10, 3, pp. 248-255, (2003).

İNTİHAL RAPORU İLK SAYFASI

Merve GENÇ Tez Son Teslim Benzerlik

ORJİNALLIK RAPORU

% 8	% 8	% 0	% 2
BENZERLİK ENDEKSİ	İNTERNET KAYNAKLARI	YAYINLAR	ÖĞRENCİ ÖDEVLERİ

BİRENCİL KAYNAKLAR

1	gcris.ktun.edu.tr İnternet Kaynağı	% 3
2	dergipark.org.tr İnternet Kaynağı	% 2
3	acikbilim.yok.gov.tr İnternet Kaynağı	- % 1
4	Submitted to Artvin Āoruh Āniversitesi Öğrenci Ödevi	% 1
5	www.mdpi.com İnternet Kaynağı	% 1
6	acikerisim.pau.edu.tr:8080 İnternet Kaynağı	<% 1
7	Submitted to Karadeniz Teknik University Öğrenci Ödevi	<% 1
8	Mat, Ayse Nagehan. "Levy Ucusuyla Gelistirilen Balina Optimizasyon Algoritmasinin Kumeleme Problemlerine Uygulanmasi", Necmettin Erbakan University (Turkey), 2022 Yayın	<% 1
9	Adişen, Caner. "Farklı IP Sınıflarına Ait Kuru Tip Trafo Kabinlerinde Porozite Değerlerinin	<% 1

Hesaplanması ve Akış Analizleri", Balıkesir
University (Turkey), 2025

Yayın

10	Yıldız, Ali Rıza. "Tasarım Optimizasyon Problemlerinin Çözümü İçin çok Amaçlı Arama Tabanlı Bir Sistemin Gelistirilmesi", Bursa Uludag University (Turkey), 2021 Yayın	<% 1
----	--	-------------

Alınları çıkart

Bibliyografyayı Çıkart

Özetinde

Özetinde

Eşleşmeleri çıkar

< 5 words