T.C.
NECMETTİN ERBAKAN
ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GELİŞTİRİLMİŞ YAPAY ALG ALGORİTMASININ HİYERARŞİK OLMAYAN KÜMELEME PROBLEMLERİNE UYGULANMASI

Sahar RASHEDI

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Ocak-2025
KONYA

TEZ KABUL VE ONAYI

Sahar RASHEDI tarafından hazırlanan "Geliştirilmiş Yapay Alg Algoritmasının Hiyerarşik Olmayan Kümeleme Problemlerine Uygulanması**"** adlı tez çalışması 24/01/2025 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Dr. Öğr. Üyesi Onur İNAN	……..
Danışman Dr. Öğr. Üyesi Murat KARAKOYUN	……..
Üye Dr. Öğr. Üyesi Ayşe Merve ACILAR	……..

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…/20.. gün ve …….. sayılı kararıyla onaylanmıştır.

Prof. Dr. Havvanur UÇBEYİAY
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Sahar RASHEDI

Tarih:

ÖZET

YÜKSEK LİSANS TEZİ

GELİŞTİRİLMİŞ YAPAY ALG ALGORİTMASININ HİYERARŞİK OLMAYAN KÜMELEME PROBLEMLERİNE UYGULANMASI

Sahar RASHEDI

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Murat KARAKOYUN

2025, 35 Sayfa

Jüri
Dr. Öğr. Üyesi Murat KARAKOYUN
Dr. Öğr. Üyesi Ayşe Merve ACILAR
Dr. Öğr. Üyesi Onur İNAN

Optimizasyon, belirli şartlar ve kısıtlar çerçevesinde bir problem için elde edilen en iyi çözüm olarak tanımlanmaktadır. Optimizasyon için geliştirilen algoritmalar, mevcut bilgileri mümkün olduğu kadar en iyi düzeyde kullanmayı hedeflemektedir. Geçmişten günümüze kadar araştırmacılar tarafından birçok optimizasyon algoritması geliştirilmiştir. Geliştirilen bu optimizasyon algoritmaları genel olarak doğada bulunan canlıların sosyal veya bireysel davranışlarına göre tasarlanmıştır. Optimizasyon algoritmaları mühendislik, tıp, endüstri, bankacılık gibi birçok sektörde başarılı bir şekilde kullanılmaktadır. Bununla beraber bilgisayar bilimleri çerçevesinde veri madenciliği alanında optimizasyon algoritmalarının kullanımı yaygınlaşmaya başlamıştır. Veri madenciliği uygulamalarından bir tanesi olan kümeleme birçok alanda sıkça kullanılmaktadır. Kümeleme probleminde en kritik noktalaradan bir tanesi kümelenen verilerin en iyi küme merkezlerinin belirlenmesidir. Bu problemin üstesinden gelmek amacıyla geleneksel birçok yaklaşım ve teknik geliştirilmiş ve kullanılmıştır. Son zamanlarda, kümeleme problemine çözüm bulmak amacıyla optimizasyon algoritmalarına başvurulmaya başlanmıştır.

Bu tez çalışmasında, kümeleme probleminin çözümü için güncel optimizasyon algoritmalarından biri olan yapay alg algoritması kullanılmıştır. Algoritmanın yerel minimuma yakalanma sorununu çözmek ve algoritmanın performansını arttırmak amacıyla Levy uçuşu yaklaşımı kullanılmıştır. Tez çalışmasında, açık kaynak şekilde veri hizmeti sunan UCI veri ambarından alınan 15 adet veri seti (appendicitis, banknote, blobs, circles, diagnosis_II, flame, hear, ionosphere, iris2d, jain, liver, sonar, very-density, vertebral3 ve wine) üzerinde Levy uçuşu destekli yapay alg algoritması ile hiyerarşik olmayan kümeleme yapılmıştır. Önerilen algoritmanın performansını değerlendirmek amacıyla her veri seti için küme merkezleri ve veriler arasındaki toplam karesel uzaklık değerleri hesaplanmıştır. Bu hata değeri algoritma için uygunluk fonksiyonu olarak kullanılmıştır. Yapılan çalışmada önerilen algoritmanın performansı, yapay alg, karınca aslanı optimizasyonu, difrensiyel evrim, güve alev optimizasyonu, parçacık sürü optimizasyonu, ağaç tohumu ve balina optimizasyon algoritmalarının performansı ile karşılaştırılmıştır. Elde edilen deneysel sonuçlar önerilen algoritmanın kümeleme performansının diğer algoritmaların performansından daha iyi olduğunu göstermiştir.

Anahtar Kelimeler: Kümeleme, Levy Uçuşu, Optimizasyon, Yapay Alg Algoritması

ABSTRACT

MS THESIS

APPLICATION OF THE IMPROVED ARTIFICIAL ALGAE ALGORITHM TO NON-HIERARACHICAL CLUSTERING PROBLEMS

Sahar RASHEDI

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER ENGINEERING

Advisor: Asst. Prof. Murat KARAKOYUN

2025, 35 Pages

Jury
Asst. Prof. Murat KARAKOYUN
Asst. Prof. Ayşe Merve ACILAR
Asst. Prof. Onur İNAN

Optimization is defined as the best solution obtained for a problem within certain conditions and constraints. Algorithms developed for optimization aim to use the existing information at the best possible level. Many optimization algorithms have been developed by researchers from past to present. These optimization algorithms are generally designed according to the social or individual behaviors of living beings in nature. Optimization algorithms are successfully used in many sectors such as engineering, medicine, industry, and banking. However, the use of optimization algorithms in the field of data mining within the framework of computer science has begun to become widespread. Clustering, one of the data mining applications, is frequently used in many areas. One of the most critical points in the clustering problem is the determination of the best cluster centers of the clustered data. Many traditional approaches and techniques have been developed and used to overcome this problem. Recently, optimization algorithms have been used to find a solution to the clustering problem.

In this thesis, one of the current optimization algorithms, the artificial algae algorithm, was used to solve the clustering problem. The Levy flight approach was used to solve the problem of the algorithm being caught in the local minimum and to increase the performance of the algorithm. In the thesis study, non-hierarchical clustering was performed with Levy flight supported artificial algae algorithm on 15 data sets (appendicitis, banknote, blobs, circles, diagnosis_II, flame, hear, ionosphere, iris2d, jain, liver, sonar, very-density, vertebral3 and wine) taken from UCI data warehouse which provides data service in open source. In order to evaluate the performance of the proposed algorithm, the total squared distance values between cluster centers and data were calculated for each data set. This error value was used as the fitness function for the algorithms. In the study, the performance of the proposed algorithm was compared with the performance of artificial algae, ant lion optimization, differential evolution, moth flame optimization, particle swarm optimization, tree seed and whale optimization algorithms. The experimental results obtained showed that the clustering performance of the proposed algorithm is better than the performance of other algorithms.

Keywords: Artificial Algae Algorithm, Clustering, Levy Flight, Optimization

ÖNSÖZ

Yüksek lisans sürecim boyunca engin tecrübesiyle bana yol gösteren, çalışmalarımda etkin katkısı bulunan, vaktini ve bilgisini benden esirgemeyen Necmettin Erbakan Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü Öğretim Üyesi Danışman Hocam Sayın Dr. Öğr. Üyesi Murat KARAKOYUN'a ve benden hiçbir desteğini esirgemeyen aileme ve sevgili eşim Muhammad Eshaq RASHEDI'ye teşekkürlerimi sunarım.

Sahar RASHEDI
KONYA-2025

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

YAA : Yapay alg algoritması
YAA_LU : Levy uçuşu tabanlı yapay alg algoritması
SSE : Toplam karesel hata (sum of squared error)
ARI : Ayarlanmış rand indeks
PSO : Parçacık sürü optimizasyonu
BOA : Balina optimizasyonu algoritması
GAO : Güve alev optimizasyonu
KAO : Karınca aslanı optimizasyonu
DE : Diferansiyel evrim
ATA : Ağaç tohum algoritması

1. GİRİŞ

Kümeleme, makine öğrenmesinde, özellikle veri analizi, veri madenciliği ve görüntü işlemede yaygın olarak kullanılan denetimsiz öğrenmede temel yöntemlerden biridir. Bu yöntemin temel amacı, verileri her grup veya kümedeki verilerin diğer kümelerdeki verilere göre birbirine daha benzer olacak şekilde gruplandırılmasıdır. Bu süreç, analistlerin verilerdeki gizli kalıpları, yapıları ve eğilimleri belirlemesine ve bunları pazarlama, tıp, biyoloji vb. çeşitli uygulamalarda kullanmasına yardımcı olur (Karaboga ve Ozturk 2011).

Kümeleme kavramı, verileri her grubun üyelerinin daha fazla benzerliğe sahip olduğu gruplar halinde gruplamak anlamına gelir. Bu benzerlik sayısal veya niteliksel özellikler açısından olabilir ve genellikle öklid mesafesi, kosinüs mesafesi veya diğer benzerlik ölçümleri gibi kriterlerle ölçülür. Kümelemenin amacı, her küme içindeki verilerin birbirine yakın olması ve kümelerin birbirinden olabildiğince farklı olmasıdır (Karaboga ve Ozturk 2011).

Kümeleme, veri madenciliği, istatistiksel veri analizi ve veri karşılaştırması gibi alanlarda kullanılan önemli bir yöntemdir. Bu yöntemde, veriler özelliklerinin benzerliğine göre kategorize edilir. Böylece benzer yapıya sahip veriler aynı gruba yerleştirilir. Öte yandan, veri grupları arasındaki benzerlik mümkün olduğunca minimize edilmelidir. Birçok alanda kümeleme yaygın olarak kullanılan bir tekniktir. Kümelenmiş veriler için en uygun küme merkezlerini bulmak, kümeleme sürecindeki en önemli adımdır. Sezgisel yöntemler, çözüm alanını sınırlamak yerine detaylı arama yaparak daha yüksek kaliteli çözümler ortaya çıkarmaktadır (Yang 2010, Karaboga ve Ozturk 2011).

Kümeleme yöntemi iki genel kategoriye ayrılır: hiyerarşik ve hiyerarşik olmayan kümeleme. Hiyerarşik kümeleme algoritmalarında küme sayısı ilk aşamada belirtilmediğinde, küme sayısını belirlemek için görsel analizler kullanılır. Ağaç diyagramlarıyla temsil edilen hiyerarşik tekniklerin sonuçlarına dendrogram denir. Hiyerarşik kümeleme algoritmalarının aksine, hiyerarşik olmayan kümeleme algoritmaları, küme sayısı kümeleme süreci başlamadan önce önemli ölçüde belirlendiğinde kullanılabilir. Nesnelerin kategorize edildiği küme sayısı belirlendikten sonra, hangi nesnenin hangi kümeye ait olduğuna dair kararlar küme belirleme kriterlerine göre verilir ve ardından atama işlemi gerçekleştirilir (Lorr 1983, Tatlıdil 1992, Everitt ve Dunn 2001).

Hiyerarşik olmayan kümeleme algoritmaları, mevcut nesneleri önceden belirlenmiş sayıda kümeye (k) bölmek için tasarlanmıştır. Küme sayısı keyfi olarak belirlenebilir veya kümeleme algoritmasının bir parçası olarak hesaplanabilir. Hiyerarşik olmayan algoritmalar, hiyerarşik algoritmalara göre daha hızlı olmaları nedeniyle büyük veri kümelerinde uygulanmaya daha uygundur. Hiyerarşik olmayan kümeleme algoritmalarında küme sayısını belirlemek önemli bir konudur. Küme sayısının doğru seçimi, istenilen sonuç ile doğrudan ilişkilidir (Anderberg 1973, Atamer 1992, Johnson ve Wichern 2002).

Hiyerarşik olmayan kümeleme için literatüre kazandırılan ve sıklıkla kullanılan birçok yöntem mevcuttur. K-means algoritması, hiyerarşik olmayan kümeleme algoritmalarının önde gelenlerinden biridir. K-means algoritması hızlı, basit ve genel olarak başarılı bir algoritma olmasına rağmen performans oranı seçilen başlangıç pozisyonuna bağlıdır ve bu nedenle sıklıkla başlangıç pozisyonuna yakın bir yerel minimumda takılıp kalır. Bu sorunun üstesinden gelmek için araştırmacılar çalışmalarında farklı kümeleme algoritmaları geliştirmeye çalışmaktadırlar. Kümeleme problemlerinin çözümünde istatistik, grafik teorisi, maksimizasyon algoritmaları, yapay sinir ağları, evrimsel algoritmalar ve sürü zeka algoritmaları gibi çeşitli teknikler kullanılmaktadır (MacQueen 1967, Karaboga ve Ozturk 2011).

Son zamanlarda bu geleneksel yöntemler ile beraber optimizasyon algoritmaları da kümelemede kullanılmaya başlanmıştır. Yapay alg algoritması (YAA) 2015 yıllında önerilen başarılı bir optimizasyon algoritmasıdır. YAA, doğa esinli bir optimizasyon algoritması olup, yapay alglerin davranışlarından esinlenerek modellenmiştir. Gerçek yaşamdaki bir alg gibi, yapay algler de fotosentez için ışık kaynağına doğru hareket eder. Modellenen algoritmada, problem uzayında bulunan her bir çözüme bir yapay alg kolonisi eşlik eder. Problem boyutu, her bir alg kolonisindeki alg hücresi sayısının eşit olduğu anlamına gelir. Alg kolonisi en uygun çözüme ulaştığında optimum elde edilir (Uymaz vd. 2015).

Bu tez çalışmasında, YAA hiyerarşik olmayan kümeleme problemlerini çözmek için kullanılmıştır. UCI veri ambarından alınan 15 adet veri seti (appendicitis, banknote, blobs, circles, diagnosis_II, flame, heart, ionosphere, iris2d, jain, liver, sonar, vary-density, vertebral3 ve wine) üzerinde en uygun küme kümelerinin belirlenmesi amaçlanmıştır. Küme merkezleri belirlenirken uygunluk fonksiyonu olarak; belirlenen küme merkezleri ve kümelere ait veriler arasındaki toplam karesel uzaklık (hata)'ın minimize edilmesi hedeflenmiştir. YAA'nın kümeleme performansını arttırmak amacıyla

algoritmaya Levy uçuşu uygulanmıştır. Önerilen Levy uçuşu tabanlı yapay alg algoritmasının (YAA_LU) performansı, orijinal YAA, karınca aslanı optimizasyonu (KAO), diferansiyel evrim (DE), güve alev optimizasyonu (GAO), parçacık sürü optimizasyonu (PSO), ağaç tohum algoritması (ATA) ve balina optimizasyonu algoritması (BOA) optimizasyon algoritmalarının performansı ile karşılaştırılmıştır. Performans karşılaştırması için ARI metriği kullanılmıştır. Elde edilen deneysel sonuçlar, önerilen YAA_LU algoritmasının diğer algoritmalara üstünlük sağladığını göstermiştir.

Tez çalışmasının geriye kalan bölümleri şu şekilde organize edilmiştir: Tezin ikinci bölümünde kümeleme alanında, son yıllarda yapılamış ilgili çalışmalar incelenerek literatür taraması apılmıştır. Üçüncü bölümde kullanılan veri setleri, karşılaştırma metrikleri, kullanılan optimizasyon algoritmaları ve önerilen algoritma hakkında bilgilendirme yapılmıştır. Dördüncü bölümde algoritmaların kümeleme problemi üzerinde elde ettikleri deneysel sonuçların karşılaştırmalı analizi yapılmıştır. Beşinci bölümde ise sonuçlar genel anlamda değerlendirilerek ileriki zamanlarda yapılabilecek çalışmalar için önerilerde bulunulmuştur.

2. KAYNAK ARAŞTIRMASI

Kümeleme, veri analizi, veri madenciliği ve makine öğrenmesi gibi alanlarda sıklıkla başvurulan bir yaklaşımdır. Üzerinde çalışılan verinin yapısına ve büyüklüğüne göre bu işlem oldukça zorlayıcı bir hal alabilmektedir. Problemin zorlayıcı ve önemli olması sebebiyle araştırmacılar birçok yöntem ve yaklaşım sunmuşlardır. Literatüre bakıldığında bu alanda yapılan birçok çalışmanın olduğu görülmektedir.

Li ve ark.'ları çalışmalarında, kaya süreksizliklerinin otomatik olarak belirlenmesi ve sınıflandırılması için geliştirilmiş bir yapay arı kolonisi algoritması (ABC) ve bulanık c-ortalamalar kümeleme (FCM) yöntemi üzerine odaklanmışlardır. FCM-ABC yöntemi olarak adlandırdıkları bu yaklaşımda, geleneksel yöntemlerin sınırlamalarını aşarak kaya mühendisliği optimizasyon tasarımı ve güvenlik değerlendirmeleri için yeni bir çözüm sunmayı hedeflemişlerdir. ABC algoritması ile kümeleme yönteminin başlangıç noktaları daha verimli seçilmiş ve yerel minimumlara takılma olasılığı azaltılmıştır. ShapeMetriX 3D Sistemi ile Çin'deki bir altın madeni sahasında yapılan kaya süreksizlikleri ölçülmüş ve yöntem sahada test edilmiştir. FCM-ABC yönteminin geleneksel yöntemlere kıyasla daha yüksek doğruluk, daha hızlı yakınsama ve üstün kümeleme performansı gösterdiğini ifade etmişlerdir. Yöntemin jeoteknik risk analizi ve mühendislik projelerinde kaya kütlesi stabilite değerlendirmesi açısından önemli bir potansiyele sahip olduğu belirtmişlerdir (Li vd. 2025).

Premkumar ve ark.'ları k-means tabanlı gri kurt optimizasyonu (KCGWO) adı verilen yeni bir meta-sezgisel algoritma önererek, geleneksel gri kurt optimizasyonu (GWO) algoritmasının veri kümeleme performansını geliştirmeyi amaçlamışlardır. GWO'nun eksikliklerini gidermek için önerilen KCGWO, k-means algoritması ve ağırlık faktörü kullanarak çözüm çeşitliliğini artırmayı ve erken yakınsamayı önlemeyi hedeflemiştir. Algoritma, on farklı sayısal test fonksiyonu ve sekiz gerçek dünya veri kümesi üzerinde test edilerek diğer optimizasyon algoritmalarıyla karşılaştırılmıştır. KCGWO'nun optimum kümeleme çözümlerini daha az iterasyonla ve daha yüksek doğrulukla bulabildiğini belirtmişlerdir. Bu durum ile de keşif ve sömürü arasındaki dengenin daha iyi sağlandığını ifade etmişlerdir (Premkumar vd. 2024).

Muthukrishnan ve Kannan çalışmalarında, Ad-Hoc Ağları (VANETs) için meta-sezgisel tabanlı kümeleme ve çok atlamalı yönlendirme (EMCMHR-LM) tekniğini sunmuşlardır. VANET'ler, akıllı ulaşım sistemleri, güvenlik ve acil durum hizmetleri gibi alanlarda önemli bir rol oynar. Ancak, yüksek hareketlilik ve değişken topoloji nedeniyle

ağın kararlılığı ve yaşam süresi gibi önemli zorluklar vardır. Bu çalışmada önerilen EMCMHR-LM modeli, bu sorunları gidermek için geliştirilmiş bir kümeleme ve yönlendirme stratejisi sunmuştur. Araçları kümelere ayırmak için slime mold optimizasyonu (SMO) yöntemini kullanmışlardır. Bu şekilde, ağ düğümlerini daha iyi gruplandırarak yönlendirme sürecini iyileştirmeyi hedeflemişlerdir. Önerdikleri yaklaşım ile ağ ömrünü arttırdıklarını, paket teslim oranını ve veri iletim hızını önemli ölçüde arttırdıklarını ifade etmişlerdir (Muthukrishnan ve Kannan 2023).

Prakash ve Pandey kablosuz algılayıcı ağlar (WSN) için PSO tabanlı enerji verimli kümeleme şeması (PSO-EECS) adlı yeni bir meta-sezgisel algoritma önermişlerdir. WSN'lerde en büyük zorluk, düğümlerin sınırlı enerji kaynaklarına sahip olmasıdır. Bu çalışma, kümeleme ve yönlendirme algoritmalarını geliştirerek ağ ömrünü uzatmayı hedeflemiştir. Sensör düğümlerinin küme başı (CH) seçiminde PSO algoritması kullanılarak enerji tüketimi optimize edilmiştir. Önerilen PSO-EECS algoritması, MATLAB ortamında PSO-BSP, TEDRP, PSOGSA, GAPSO–H ve ABC-DE gibi mevcut protokollerle karşılaştırılmıştır. PSO-EECS'nin, diğer yöntemlere kıyasla ağ ömrünü %29,2, kararlılık süresini %95,4, ağın operasyon süresini %34,8 ve veri iletim oranını %27,1 oranında artırdığını belirtmişlerdir (Prakash ve Pandey 2023).

Thieu ve ark.'ları kümeleme probleminde kullanılmak üzere MetaCluster isimli açık kaynaklı bir Python kütüphanesi sunmuşlardır. MetaCluster, meta-sezgisel algoritmalar kullanarak kümeleme problemlerini çözmek için geliştirilmiş olup, 200'den fazla meta-sezgisel algoritmayı destekleyen kapsamlı bir çerçeve sunmaktadır. MetaCluster'ın amacı, veri kümeleme problemlerini optimizasyon sorunları olarak ele alan bir yazılım geliştirmek olarak ifade etmişlerdir. Bu açık kaynak kütüphanenin sundukları içerisinde 200+ metasezgisel algoritma, 48 hazır veri kümesi ve 40+ performans değerlendirme metriği bulunmaktadır. Araştırmacılar, MetaCluster'ın, meta-sezgisel algoritmalarla kümeleme problemlerini çözmek için geliştirilmiş en kapsamlı açık kaynak Python kütüphanelerinden biri olduğunu, kullanıcıların kolayca test senaryoları oluşturmasına, algoritmaları karşılaştırmasına ve model geliştirmesine olanak tanıdığına vurgu yapmışlardır (Van Thieu vd. 2023).

Almodfer ve ark.'ları çalışmalarında, küresel optimizasyon ve veri kümeleme için kuantum mutasyonlu sürüngen arama algoritması (QMRSA) adlı yeni bir yöntem önermişlerdir. Çalışmada QMRSA, aquila optimizasyonu, gri kurt optimizasyonu, sinüs cosinüs algoritması, balina optimizasyonu gibi birçok popüler optimizasyon algoritmasıyla karşılaştırılmıştır. Önerilen algoritmanın matematiksel test

fonksiyonlarında ve gerçek dünya kümeleme problemlerinde üstün performans gösterdiği ifade edilmiştir (Almodfer vd. 2022).

Yadav ve Mahapatra kablosuz sensör ağlarında (WSN) enerji verimli veri iletimi ve ağ ömrünü artırmak için yeni bir hibrit optimizasyon algoritması önermişlerdir. Deniz aslanı optimizasyonu ile parçacık sürü optimizasyonunu hibrit bir şekilde kullanan algoritma, küme başlığı (Cluster Head - CH) seçiminde enerji tüketimini, mesafeyi, gecikmeyi ve hizmet kalitesini (QoS) optimize etmeyi amaçlamıştır. CH'ler, sensör düğümlerinden veri toplayarak ana istasyona (BS) iletir ve doğru seçim, ağın performansında kritik bir rol oynar. Yapılan simülasyon sonuçlarına göre, önerilen yaklaşımın geleneksel algoritmalara (GA, PSO, ALO, GOA) kıyasla daha yüksek ağ ömrü, daha az enerji tüketimi ve daha fazla canlı düğüm oranı sunduğunu belirtmişlerdir. Bu yaklaşımlarının, kablosuz sensör ağlarında daha verimli ve dengeli kümeleme sağlamak için umut vadettiğini ifade etmişlerdir (Yadav ve Mahapatra 2022).

Kaur ve Kumar çalışmalarında, veri kümeleme için su dalgası optimizasyonu (WWO) tabanlı yeni bir meta-sezgisel algoritma önermişlerdir. Geleneksel WWO algoritması, küresel en iyi bilgi eksikliği ve erken yakınsama problemleri nedeniyle karmaşık optimizasyon sorunlarında yetersiz kalabilmektedir. Bu çalışma, geliştirilmiş bir arama mekanizması ve bozunma operatörü ekleyerek WWO'nun bu eksikliklerini gidermeyi amaçlamıştır. PSO tabanlı güncellenmiş arama mekanizması, küresel en iyi çözümü belirleyerek optimizasyon sürecini iyileştirirken, bozunma operatörü erken yakınsamayı önleyerek daha iyi keşif yeteneği sunmaktadır. Algoritmanın başarısı, on üç farklı kümeleme veri seti üzerinde yapılan deneylerle incelenmiştir. Sonuç olarak mevcut kümeleme algoritmalarına kıyasla doğruluk oranında %4, F-skorunda ise %7 iyileşme sağladığını belirtmişlerdir. İstatistiksel analizler ile önerilen yöntemin kümeleme alanındaki etkinliğini ortaya koymuşlardır (Kaur ve Kumar 2022).

Kuo ve ark.'ları kategorik veri kümeleme için metasezgisel tabanlı olasılıksal bulanık k-modlar (PFKM) algoritmasını önermişlerdir. Kategorik verilerin kümeleme sürecinde gürültü ve aykırı değerlerin etkisini azaltmak amacıyla, olasılık tabanlı bir model ile bulanık k-modlar algoritması birleştirilmiştir. Bu yöntemin performansını artırmak için genetik algoritma, parçacık sürü optimizasyonu ve sinüs kosinüs algoritması gibi metasezgisel teknikler kullanılmış ve her biriyle entegre edilerek GA-PFKM, PSO-PFKM ve SCA-PFKM algoritmaları geliştirilmiştir. Sekiz farklı veri kümesi üzerinde yapılan deneysel analizler sonucunda, özellikle PSO-PFKM ve SCA-PFKM'nin geleneksel FKM ve PFKM algoritmalarına kıyasla daha düşük hata oranı (SSE) ve daha

yüksek doğruluk (AC) sağladığını ifade etmişlerdir. Ayrıca, önerilen yöntemlerin aykırı değerlerin etkisini azaltmada ve daha stabil kümeleme çözümleri üretmede başarılı olduğunu belirtmişlerdir (Kuo vd. 2021).

Mageshkumar ve ark.'ları veri kümeleme sürecinin verimliliğini artırmak için karınca aslanı optimizasyonu (ALO) ve karınca kolonisi optimizasyonu (ACO) tabanlı yeni bir hibrit metasezgisel algoritma önermişlerdir. Kümeleme sürecinde intra-cluster mesafesini azaltarak kümelerin homojenliğini artırmayı hedefleyen bu yaklaşım, ALO'nun güçlü keşif yeteneklerini ve ACO'nun en iyi küme merkezlerini belirleme kapasitesini birleştirmiştir. Ayrıca, Cauchy mutasyon operatörü kullanılarak yerel minimuma yakalanma problemini çözmeyi amaçlamışlardır. Elde ettikleri deneysel sonuçlara göre, önerilen ACO-ALO algoritmasının geleneksel K-Means, ACO ve ALO algoritmalarına kıyasla %23 daha düşük intra-cluster mesafesi ve %18 daha iyi mesafe indeksi sağladığını ifade etmişlerdir. Ayrıca önerilen yaklaşımın, daha az iterasyon ile daha hızlı yakınsama sağlayarak kümeleme verimliliğini artırdığını belirtmişlerdir (Mageshkumar vd. 2019).

Nasiri ve Khiyabani çalışmalarında veri kümeleme için balina optimizasyonu algoritması (WOA) tabanlı yeni bir metasezgisel yaklaşım önermişlerdir. K-means gibi geleneksel yöntemlerin yerel minimumlara takılma sorununu aşmak amacıyla tasarlanan bu algoritma, küresel en iyi çözümü takip ederek küme merkezlerini optimize etmektedir. WOA, sekiz farklı veri seti üzerinde test edilmiş ve parçacık sürü optimizasyonu, yapay arı kolonisi, genetik algoritma ve diferansiyel evrim gibi popüler algoritmalardan daha iyi kümeleme doğruluğu ve daha düşük iç küme mesafesi sağladığı gözlemlenmiştir. Sonuçlara dayanarak, WOA'nın hızlı yakınsama, düşük parametre gereksinimi ve daha dengeli keşif/sömürü dengesi sayesinde kümeleme için güçlü bir alternatif sunduğunu ifade etmişlerdir (Nasiri ve Khiyabani 2018).

Literatüre bakıldığında kümeleme probleminin amaç veya uygulamanın belli bir kısmı olarak çalışıldığı bir çok çalışma görülmektedir. Ancak bu çalışmaların tamamının incelenip değerlendirilmesi zaman ve efor açısından neredeyse imkansız bir durumdadır. Bu nedenle çalışmanın literatür taraması kapsamında son yıllarda yapılan önemli bir takım çalışmalar incelenmiştir.

Bu tez çalışmasında, Levy uçuşu tabanlı yapay alg algoritması hiyerarşik olmayan kümeleme problemlerine uygulanmıştır. UCI veri ambarından alınan 15 adet veri seti kullanılarak, önerilen algoritmanın performansı, orijinal yapay alg algoritması ve 6 farklı optimizasyon algoritmasının performasıyla karşılaştırılmıştır. Edilen deneysel sonuçlar

önerilen algoritmanın kümeleme performansının diğer algoritmalardan daha iyi olduğunu göstermiştir.

3. MATERYAL VE YÖNTEM

Bu bölümde tez çalışmasında kullanılan veri setleri hakkında bilgi verilecektir. Algoritmaların performans karşılaştırmasını yapmak için kullanılan metrik incelenecek ve uygunluk fonksiyonu anlatılacaktır. Ayrıca çalışmada kullanılan optimizasyon algoritmaları hakkında genel bilgi verilerek önerilen algoritma detaylandırılacaktır.

3.1. Veri Setleri

Algoritmaların kümeleme performansı, Çizelge 3.1'de verilen farklı özelliklere sahip 15 veri seti (appendicitis, banknote, blobs, circles, diagnosis-II, flame, heart, iris2d, ionosphere, jain, liver, sonar, vary density, vertebral3, wine) ile ölçülmüştür. Bu Veri setleri, UCI veri ambarından alınmıştır (UCI 2024).

Çizelge 3.1. Kümelemede kullanılan veri setleri

Veriseti	Örnek Sayısı	Nitelik Sayısı	Sınıf Sayısı
Appendicitis	106	7	2
Banknote	1372	4	2
Blobs	1500	2	3
Circles	1500	2	2
Diagnosis_II	120	6	3
Flame	240	2	2
Heart	270	13	2
Ionosphere	351	34	2
Iris2D	150	2	3
Jain	373	2	2
Liver	345	6	2
Sonar	208	60	2
Vary-density	150	2	3
Vertebral3	310	6	3
Wine	178	13	3

Çizelge 3.1'e bakıldığında birbirinden farklı veri setlerinin kullanıldığı gözlemlenmektedir. Bu durum algoritmaların performansının farklı veri setlerindeki başarı sonucunu gözlemleme imkanı vermektedir. Kullanılan veri setlerine bakıldığında, en fazla örnek sayısı olarak 1500 veri içeren blobs ve circles göze çarpmaktadır. 106 ile an az örneğe sahip veri seti ise appendicitis'tir. Nitelik sayısı olarak bakıldığında ise veri setlerinin 2 ile 60 nitelik sayısı aralığında dağıldığı görülmektedir. Sınıf sayısı olarak ise tüm veri setleri 2 veya 3 sınıf özelliğine sahiptir.

3.2. ARI Metriği

Rand indeksi, William M. Rand tarafından önerilmiş, istatistikte ve özellikle veri kümelemesinde iki veri kümelemesi arasındaki benzerliği ölçen bir metriktir. Rand indeksinin bir formu, elemanların şans eseri gruplanmasına göre ayarlanmış olabilir, bu ayarlanmış rand indeksidir. Rand indeksi, bir bağlantının bir küme içinde olup olmadığını belirleme doğruluğudur (Rand 1971).

P (beklenen, predicted) ve T (bilinen, true) aynı örneklerden oluşan veri kümeleri olsun. P ve T kümelerinden alınan her eleman çifti için, bu çiftin dahil olduğu küme (veya sınıf) bilgisi olarak 4 farklı durum oluşmaktadır (Rand 1971):

- ✓ a, eleman çiftlerinin P kümesinde de T kümesinde de aynı sınıfta olduğu durumların sayısı.
- ✓ b, eleman çiftlerinin P kümesinde farklı sınıfta ve T kümesinde farklı sınıfta olduğu durumların sayısı.
- ✓ c, eleman çiftlerinin P kümesinde aynı sınıfta ve T kümesinde farklı sınıfta olduğu durumların sayısı.
- ✓ d, eleman çiftlerinin P kümesinde farklı sınıfta ve T kümesinde aynı sınıfta olduğu durumların sayısı.

olmak üzere Rand indeks metriğinin matematiksel formülü aşağıdaki gibidir (Rand 1971, Hubert ve Arabie 1985).

$$RI = \frac{a + b}{a + b + c + d} \tag{3.1}$$

Rand indeksi 0 ile 1 arasında bir değere sahiptir. 0 iki veri kümelemesinin herhangi bir nokta çiftinde uyuşmadığını ve 1 ise veri kümelemelerinin tam olarak aynı olduğunu göstermektedir.

Ayarlanmış Rand indeksi (ARI), rand indeksinin şansa göre düzeltilmiş versiyonudur (Rand 1971, Hubert ve Arabie 1985, Nguyen vd. 2009). Bu tür bir şans düzeltmesi, rastgele bir model tarafından belirlenen kümelemeler arasındaki tüm eşli karşılaştırmaların beklenen benzerliğini kullanarak bir temel oluşturur. Geleneksel olarak, rand indeksi, kümelemeler için permütasyon modeli kullanılarak düzeltilmiştir. Ancak, permütasyon modelinin öncülleri sıklıkla ihlal edilir; birçok kümeleme senaryosunda, ya kümelerin sayısı ya da bu kümelerin boyut dağılımı önemli ölçüde

değişir. Örneğin, k-means'de kümelerin sayısı uygulayıcı tarafından sabitlenir, ancak bu kümelerin boyutları verilerden çıkarılır. ARI'nin varyasyonları, farklı rastgele kümeleme modelleri için hesaplanır (Gates ve Ahn 2017). Rand indeksi 0 ile 1 aralığında değerler almasına karşın ARI negatif değerler de alabilmektedir (Gates ve Ahn 2017). Verilen bilgiler ışığında ARI metriğinin matematiksel olarak hesaplanması Eşitlik 3.2'de sunulmuştur (Qaddoura vd. 2020).

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]} \tag{3.2}$$

Eşitlik 3.2'de kullanılan *RI* Rand indeks (Eşitlik 3.1), *E[RI]* beklenilen Rand indeks ve *max(RI)* maksimum Rand indeks olmak üzere aşağıda verilen matematiksel ifadeler ile hesaplanmaktadır.

$$E[RI] = \frac{\sum_{p=1}^{|P|}\binom{n_p}{2} \quad \sum_{t=1}^{|T|}\binom{n_t}{2}}{\binom{N}{2}} \tag{3.3}$$

$$\max(RI) = \frac{1}{2}\left[\sum_{p=1}^{|P|}\binom{n_p}{2} \quad + \quad \sum_{t=1}^{|T|}\binom{n_t}{2}\right] \tag{3.4}$$

Eşitlikteki n_t ve n_p sırasıyla bilinen kümenin (T) ve tahmin edilen kümenin (P) örnek sayılarıdır.

3.3. Uygunluk Fonksiyonu

Toplam Karesel Hata (Sum of Squared Errors, SSE), istatistik ve makine öğrenimi gibi alanlarda kullanılan bir metriktir. Genellikle regresyon modellerinin performansını değerlendirmek için kullanılır. SSE, bir modelin gerçek değerler ile tahmin ettiği değerler arasındaki farkların karesinin toplamını ifade eder.

Bir regresyon modelinin temel amacı, bağımlı değişkenin gerçek değerlerini mümkün olduğunca iyi tahmin etmektir. Tahminlerle gerçek değerler arasındaki farklar, hataları temsil eder. SSE, bu hataların karelerini alıp toplamını hesaplar. Matematiksel olarak, n adet veri için SSE değeri Eşitlik 3.5 ile hesaplanır.

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad (3.5)$$

Burada;

- y_i, i . veri örneği için gerçek değeri,
- y_i, i. veri örneği için model tarafından yapılan tahmini değeri,
- n, veri örneklerinin toplam sayısıdır.

SSE'nin düşük olması, modelin verilere daha iyi uydurulduğunu ve daha iyi tahminler yaptığını gösterir. SSE'nin değeri sıfıra yaklaştıkça, modelin tahminleri gerçek değerlere daha yakın olur (AKTO vd. 2019, Mat vd. 2021).

Tez çalışmasında küme merkezleri ile bu merkeze ait örneklerin arasındaki SSE değeri ugunluk fonksiyonu olarak kullanılmıştır. Algoritmalar, bu değeri minimize eden küme merkezlerini tespit etmeyi amaçlamışlardır.

3.4. Optimizasyon Algoritmaları

Günümüzde birçok mühendislik ve diğer bilim dallarındaki problemlerin çözümünde yapay zeka tabanlı optimizasyon algoritmaları sıklıkla kullanılmaktadır. Bu algoritmaların en iyi çözümlere ulaşma özelliği, bu alanda birçok araştırmaya yol açmıştır. 20. yüzyılın sonlarından itibaren bu alanda çeşitli algoritmalar geliştirilmiş ve her biri kendi alanında birçok sorunun çözümüne katkıda bulunmuştur. Optimizasyon algoritmaları, bir problemin gerçek anlamda en iyi çözümünü her zaman sunmasa da en iyi çözüme yakın bir çözümü kabul edilebilir süreler içerisinde sunabilir. Bu nedenle, bir problemi çözmek için uygun bir yöntem seçerken tüm karşılaştırmalar ve faktörler göz önünde bulundurulmalı ve ardından optimizasyon işlemine başlanmalıdır. Optimizasyon algoritmalarının amacı, problemin kısıtları ve ihtiyaçları göz önünde bulundurularak kabul edilebilir bir çözüm bulmaktır. Bir problemin çözümünde farklı çözümler bulunabilir. Bu çözümleri karşılaştırmak ve en iyi çözümü seçmek için, uygunluk (maliyet) fonksiyonu olarak da adlandırılan bir amaç fonksiyonu tanımlanır. Bu fonksiyonun seçimi problemin doğasına bağlıdır. Farklı optimizasyon problemleri genel olarak kısıtlı ve kısıtsız optimizasyon olarak ikiye ayrılır ve optimizasyon süreci genellikle problem formülasyonu, modelleme, optimizasyon ve uygulama olmak üzere dört aşamada gerçekleştirilir. Optimizasyon veya keşif algoritmalarının amacı, problem

için uygun kabul edilebilecek bir çözümü makul bir süre içinde sunmaktır. Tez çalışmasında farklı optimizasyon algoritmalarının kümeleme problemi üzerindeki performansı değerlendirilmiştir. Bununla beraber yapay alg algoritmasının performansını arttırmak amacıyla Levy uçuşu stratejisi ile arama özelliği günlendirilmiştir.

3.4.1. Parçacık Sürü Optimizasyonu

Parçacık sürü optimizasyonu (PSO), Eberhart ve Kennedy tarafından 1995 yılında geliştirilen, kuş ve balık sürülerinin sosyal davranışlarından ilham alan, popülasyon tabanlı bir stokastik optimizasyon yöntemidir. PSO algoritması, Genetik Algoritma (GA) gibi evrimsel hesaplama tekniklerine benzerlikler gösterir. Bu sistem, rastgele çözümlerden oluşan bir kümeyle başlar ve nesiller boyunca optimizasyon için arama yapar. Ancak GA'nın aksine, PSO'da çaprazlama ve mutasyon gibi evrimsel operatörler bulunmaz. PSO'da potansiyel çözümler, problem uzayında en iyi çözümü takip ederek uçan parçacıklar olarak adlandırılır. GA'ya kıyasla PSO algoritması daha basittir ve ayarlanması gereken birkaç parametreye sahiptir. Bu algoritma, performans optimizasyonu, yapay sinir ağları eğitimi, bulanık sistem kontrolü ve diğer birçok alanda başarıyla kullanılmıştır. Daha önce de belirtildiği gibi, PSO algoritması kuş sürülerinin davranışlarını simüle eder. PSO'da her bir çözüm, arama uzayında bir "kuş"tur. Biz buna "parçacık" diyoruz. Tüm parçacıklar, optimizasyon için kullanılan uygunluk fonksiyonu tarafından değerlendirilen bir uygunluk değerine ve hareketini yönlendiren bir hıza sahiptir. Parçacıklar, problem uzayında en iyi çözümü takip ederek hareket ederler. PSO algoritması, rastgele parçacıklardan (çözümler) oluşan bir kümeyle başlar ve sonraki nesillerde arama yapar. Her yinelemede, her parçacık iki "en iyi" değerle güncellenir. İlki, parçacığın şimdiye kadar ulaştığı en iyi çözümdür (uygunluk değeri de kaydedilir). Bu değere pbest denir. Diğeri ise, kümedeki herhangi bir parçacık tarafından şimdiye kadar elde edilen en iyi çözümdür. Bu en iyi küresel değere gbest denir. Parçacık, komşularını topolojik olarak bir küme olarak düşündüğünde, en iyi değer yerel en iyi değerdir ve lbest olarak adlandırılır. PSO algoritması, basit yapısı ve etkinliği sayesinde birçok optimizasyon probleminde başarılı sonuçlar vermiştir. Özellikle büyük boyutlu ve karmaşık problemlerde geleneksel yöntemlere göre daha iyi performans gösterdiği görülmüştür (Eberhart ve Kennedy 1995, Cura 2012).

3.4.2. Diferansiyel Evrim Algoritması

Diferansiyel evrim (DE) algoritması, ilk olarak 1995 yılında Rainer Storn ve Kenneth Price tarafından tanıtılan bir evrimsel algoritmadır. Bu araştırmacılar, "differential evolution a practical approach to global optimization" başlıklı makalede, bu algoritmanın türev alınamayan doğrusal olmayan fonksiyonları optimize etmede oldukça başarılı olduğunu ve sürekli uzaylardaki optimizasyon problemleri için güçlü ve hızlı bir yöntem olarak sunulduğunu göstermişlerdir. DE algoritması, GA'nın ana eksikliği olan yerel arama eksikliğini aşmak için geliştirilmiştir. GA ve DE arasındaki temel fark, seçilim operatöründe yatmaktadır. GA'da bir çözümün ebeveyn olarak seçilme olasılığı, uygunluk değerine bağlıdır. Ancak DE'de tüm çözümler eşit seçilme olasılığına sahiptir. Yani seçilme, uygunluk değerine bağlı değildir. Yeni bir çözüm, mutasyon ve çaprazlama operatörleri kullanılarak üretildikten sonra, eski çözümle karşılaştırılır ve daha iyiyse yerini alır. DE'de diğer algoritmalardan farklı olarak, önce mutasyon sonra çaprazlama işlemi uygulanır. Böylece yeni nesil oluşturulur. Mutasyon işleminde belirli bir dağılım kullanılmaz, adım uzunluğu mevcut üyeler arasındaki mesafeler kullanılarak belirlenir. Başlangıç popülasyonu genellikle üyelerin uzayda eşit olarak dağılmasını sağlamak için üniform dağılım kullanılarak oluşturulur. DE algoritmasında her adımda üyeler birbirine yaklaşır ve bu yakınsama sonucu en iyi çözüme ulaşılması hedeflenir. Popülasyonun büyük olması, en iyi çözüme ulaşmayı kolaylaştırır. Algoritmadaki önemli bir parametre, ölçeklendirme faktörüdür. Bu faktör küçük seçilirse mutasyon adımları küçülür ve arama süresi uzar. Büyük seçilirse ise algoritma iyi çözümleri gözden kaçırabilir. Bu nedenle bu faktörün doğru belirlenmesi önemlidir. Çaprazlama işleminde ise 0 ile 1 arasında rastgele bir sayı üretilir. Bu sayı çaprazlama oranından küçükse yeni üyeden bir parça alınır, aksi halde eski üyeden alınır. Bu işlem tüm üyeler için tekrarlanır. Oluşturulan yeni matris, eski matrisle karşılaştırılır ve daha düşük (daha iyi) maliyeti olan matris yeni matris olarak kabul edilir. Bu işlem tüm popülasyon için tekrarlanır. DE, basit yapısı ve etkinliği sayesinde birçok optimizasyon probleminde başarılı sonuçlar vermektedir (Price vd. 2005, Qin vd. 2008).

3.4.3. Balina Optimizasyon Algoritması

Balina optimizasyonu algoritması (BOA), dünyanın en büyük memelilerinden biri olan kambur balinanın davranışlarından esinlenerek modellenmiştir. Bu balinaların en

sevdiği av, kril ve küçük balık gruplarıdır. Kambur balinaların en ilginç özelliklerinden biri, avlanma yöntemleridir. Bu keşifsel davranış, "kabarcık ağ avı" olarak bilinir. Kambur balinaları genellikle su yüzeyine yakın küçük kril veya balık gruplarını avlarlar. Bu avlanma, dairesel veya spiral şekilli kabarcıklar oluşturarak gerçekleştirilir. BOA algoritması, doğadan esinlenerek geliştirilmiş ve çeşitli alanlarda kullanılabilen popülasyon tabanlı bir optimizasyon algoritmasıdır. Balinalar avlanma alanını tespit edebilir ve onları kuşatabilirler. Optimizasyon problemlerinde de en iyi çözümün, şu anki en iyi aday çözüm olduğu varsayılır. En iyi çözüm bulunduğunda, diğer çözümler kendilerini bu en iyi çözüme yaklaştırmaya çalışırlar. Balinaların avı kuşatma ve çevrelemeye dayalı mekanizmaları sayesinde, BOA arama uzayında çözüme ulaşmada oldukça başarılıdır. Esnek ve kullanışlı yapısı sayesinde son zamanlarda popülerleşen bu algoritma, birçok farklı optimizasyon probleminde kullanılmaktadır (Mirjalili ve Lewis 2016, Mostafa Bozorgi ve Yazdani 2019).

3.4.4. Karınca Aslanı Optimizasyonu

Karınca Aslanı Optimizasyonu (KAO), doğadan ilham alarak geliştirilmiş bir optimizasyon algoritmasıdır. Bu algoritma, Karınca Aslanı (Ant Lion) adı verilen bir avcı türünün avlama davranışını model alır. Karınca Aslanı, karıncaları avlamak için yaptığı tuzaklar ile bilinen bir hayvandır ve bu avlanma stratejisi, algoritmanın temelini oluşturur. KAO, doğrudan etkileşimli bir av ve avcı sistemi gibi çalışarak, çözümlere ulaşmada kullanılan etkili bir araçtır. Karınca Aslanı, avlarını yakalamak için bir çukur tuzağı oluşturur. Bu çukurlar, zeminde çeşitli derecelerde eğimli şekilde açılır ve karıncalar bu tuzaklara düşer. Karınca Aslanı, çukurların kenarlarında bekleyerek, tuzağa düşen avlarını yakalar. Karınca Aslanı Optimizasyonu, bu doğal avlanma stratejisini çözüm arama sürecine uyarlayarak çalışır. Aşağıdaki temel bileşenlere dayanmaktadır (Mirjalili 2015, Ali vd. 2017):

Başlangıç Popülasyonu: Algoritma, çözüm uzayında rastgele seçilmiş bir popülasyon (karıncalar) ile başlar. Bu, potansiyel çözümlerin oluşturulması aşamasıdır.

Avlanma Davranışı: Popülasyondaki her birey, çözüm uzayında bir pozisyonda yer alır ve bu pozisyonun uygunluk değeri hesaplanır. Uygunluk değeri, çözümün ne kadar iyi olduğunu belirten bir ölçüdür. Karınca Aslanı, en iyi çözüm ile en kötü çözüm arasındaki farkları dikkate alarak, daha iyi çözümlerin bulunduğu bölgelere doğru hareket eder.

Pozisyon Güncellemeleri: Çözüm uzayındaki her birey, Karınca Aslanı'nın avlanma davranışını taklit ederek, diğer bireylerle etkileşime girer. Aslan, daha iyi çözümleri bulmaya yönelir, karıncalar ise tuzaklar kurarak bu çözüm alanlarında keşif yapar.

Evrimsel Adımlar: Karınca Aslanı, en iyi çözümü bulmak için belirli sayıda adımda evrimsel süreçler uygular. Bu süreç, popülasyondaki çözümlerin sürekli iyileştirilmesini sağlar.

KAO, özellikle çok sayıda yerel minimum bulunan ve karmaşık çözüm uzaylarında başarılı sonuçlar verir. Ayrıca, algoritma oldukça basit bir yapıya sahip olmasına rağmen etkili bir optimizasyon sağlar.

3.4.5. Ağaç Tohumu Algoritması

Ağaç tohum algoritması (ATA), sürekli optimizasyon problemleri için geliştirilmiş, doğadan ilham alınarak oluşturulmuş bir optimizasyon algoritmasıdır. Bu algoritma, ağaç tohumlarının doğadaki yayılma ve büyüme süreçlerinden esinlenerek çözüm uzayında arama yapmayı amaçlar. ATA, özellikle çok sayıda yerel minimuma sahip olan karmaşık problemlerde başarılı sonuçlar elde edebilecek şekilde tasarlanmıştır. ATA'nın en önemli özelliklerinden biri, çözüm uzayını geniş bir şekilde keşfetme ve en iyi çözüme ulaşma yeteneğidir. ATA, biyolojik bir süreçten ilham alarak geliştirilmiş bir doğa ilhamlı algoritmadır. Doğada ağaç tohumları, çevresel etmenler (rüzgar, su, hayvanlar vb.) aracılığıyla yayılır ve büyür. ATA, bu tohumların rastgele yayılması ve zamanla çözüm uzayındaki en verimli bölgelerde kümelenmesi prensibine dayanır. Algoritmada ilk olarak, çözüm uzayında rastgele bir grup ağaç tohumu (birey) seçilir. Bu tohumlar, çözüm uzayındaki potansiyel çözümleri temsil eder. Ardından ağaç tohumları çözüm uzayında hareket eder ve büyür. Bu hareket, her tohumun mevcut konumunun, belirli kurallara göre değiştirilmesiyle gerçekleşir. Bu aşama, çözüm uzayındaki farklı bölgeleri keşfetmeye ve daha iyi çözümler aramaya olanak tanır. Her iterasyon sonunda, en iyi çözüm (en iyi tohum) seçilir ve bu çözüm ile algoritmanın bir sonraki aşamasına geçilir. Bu adım, algoritmanın çözüm uzayında doğru bölgelere odaklanmasını sağlar. Algoritma, belirli bir durma kriterine (örneğin, iterasyon sayısı veya belirli bir çözüm kalitesi) ulaştığında sona erer. Sonuç olarak, en iyi bulunan çözüm, problemi çözmek için en uygun çözüm olarak kabul edilir. ATA, sürekli optimizasyon problemleri için güçlü bir doğa ilhamlı algoritma olarak öne çıkmaktadır. Algoritma, çözüm uzayındaki küresel

optimuma yakın çözümler bulmada etkili olup, karmaşık ve büyük ölçekli problemlerde bile başarılı sonuçlar verebilmektedir. Parametrelerin doğru ayarlanması ve uygun hibrit yöntemlerin kullanılması, ATA'nın performansını daha da iyi bir sürece götürebilir (Kiran 2015, Manoranjani ve Sukumaran 2024).

3.4.6. Güve Alev Optimizasyon Algoritması

Güve alev optimizasyon (GAO) algoritması, doğada gözlemlenen bir fenomene dayanan doğa ilhamlı bir optimizasyon algoritmasıdır. 2015 yılında Mirjalili tarafından önerilmiştir. Özellikle küresel optimizasyon problemlerine çözüm arayan bir algoritma olarak dikkat çekmiştir. Bu algoritma, güve böceklerinin uçuş davranışlarını ve gece ışığına doğru yönelme özelliklerini taklit etmektedir. Güveler, karanlıkta yönlerini aydınlatma kaynaklarına göre ayarlayarak ilerlerler, fakat bu yönelim bazen onları ölümcül tuzaklara da sürükler. GOA, bu biyolojik sürecin matematiksel bir modelini oluşturur ve çözüm uzayında mevcut problem için en iyi çözüme ulaşmayı hedefler.

GAO, güve böceklerinin doğal uçuş davranışını taklit eder. Güveler, ışığa doğru uçarken, ideal olarak bir ışık kaynağının etrafında sabit bir mesafede dönerler. Ancak, doğada, güvelerin genellikle ışığa doğru hareket ederken bu hareketin düz çizgilerden ziyade, spiraller şeklinde olmasını gözlemleyebiliriz. Bu, güvelerin ışık kaynağına yaklaştıkça mesafelerini tekrar yeniden ayarlamalarıyla gerçekleşir. Algoritma, çözüm uzayında rastgele bir popülasyon (güveler) oluşturur. Her bir güve, bir çözümü temsil eder ve algoritma bu çözümleri belirli bir sayıda başlangıç iterasyonu boyunca evrimleştirir. Güveler, çözüm uzayında sabit bir "alev" (en iyi çözüm) etrafında hareket ederler. Güvelerin uçuşu, alevin etrafında dönmeye başladıkça mesafeleri yeniden ayarlanır. Bu, güvelerin alevin etrafında sırasıyla spiral hareketler yapmasını sağlayan bir davranış modelidir. Alev, çözümler arasındaki en iyi çözümü temsil eder ve sürekli olarak güncellenir. Bu, her iterasyonda en iyi çözümün bir hedef fonksiyonu veya performans metriğine göre hesaplanmasıyla yapılır. Algoritma, belirli bir bitirme kriterine ulaştığında (örneğin, maksimum iterasyon sayısına veya belirli bir çözüm kalitesine) sona erer ve son elde edilen çözüm, problemi çözen en iyi çözüm olarak kabul edilir ve sunulur.

Güve alev optimizasyon, doğadaki güve böceklerinin ışığa doğru yönelme davranışını taklit eden, etkili bir küresel optimizasyon algoritmasıdır. Basit yapısı ve küresel en iyiye yakın çözümler bulma yeteneği sayesinde, mühendislik tasarımından makine öğrenmesine kadar birçok alanda başarılı bir şekilde uygulanabilir. Hem hızlı hem

de etkili bir çözüm arayışına girmesi, GAO'yu güçlü bir optimizasyon aracı haline getirmektedir (Mirjalili 2015, Sahoo ve Saha 2022).

3.5.Yapay Alg Algoritması

Yapay alg algoritması (YAA), Uymaz ve arkadaşları tarafından, alglerin yaşam davranışlarını modelleyerek önerilen bir optimizasyon algoritmasıdır. Algoritmadaki yapay algler, alglerin özelliklerinin idealize edilmesiyle problem uzayındaki her bir çözüme karşılık gelir. Gerçek algler gibi, yapay algler de fotosentez yapmak için ışık kaynağına doğru helisel bir şekilde yüzebilir, çevreye uyum sağlayabilir, baskın türleri değiştirebilir ve mitoz bölünmeyle çoğalabilir. Bu nedenle, algoritma "evrimsel süreç", "adaptasyon" ve "helisel hareket" olmak üzere 3 temel bölümden oluşmaktadır. Algoritmada, algler ana cinslerdir. Bu tüm popülasyon, alg kolonilerinden oluşmaktadır. Bir alg kolonisi, birlikte yaşayan bir grup alg hücresidir. Tek bir alg hücresi bölündüğünde iki yeni alg hücresi üretir ve bunlar yan yana yaşarlar. Bu ikisi bölündüğünde, yeni dört hücre birlikte yaşar ve süreç benzer şekilde devam eder. Alg kolonisi tek bir hücre gibi davranır, birlikte hareket eder ve koloni içindeki hücreler uygun olmayan yaşam koşulları altında ölebilir. Şer kuvveti veya bazı uygunsuz koşullar gibi bir dış kuvvet, koloniyi dağıtabilir ve dağılan her bir kısım artık yeni bir koloni haline gelir. Optimum noktada bulunan koloni, optimum alg hücrelerinden oluşan optimum kolonisi olarak adlandırılır (Özkış ve Babalık , Uymaz vd. 2015).

$$\text{Alg kolonisinin popülasyonu} = \begin{bmatrix} x_1^1 & \cdots & x_1^D \\ \vdots & \ddots & \vdots \\ x_N^1 & \cdots & x_N^D \end{bmatrix}$$

$$i\text{-inci alg kolonisi} = \left[x_i^1, x_i^2, \ldots, x_i^D \right]$$

Burada x_i^j , i-inci alg kolonisinin j-inci boyutundaki alg hücresinin bir özelliğini temsil eden bir değişkendir (Özkış ve Babalık , Uymaz vd. 2015).

Evrimsel Süreç

İşlem yeterli besin koşulları altında, alg kolonisi yeterli ışık alırsa, gerçek mitoz bölünmeye benzer şekilde, t zamanında iki yeni alg hücresi üretmek üzere büyür ve çoğalır. Tam tersine, yeterli ışık almayan alg kolonisi bir süre hayatta kalır ancak sonunda

ölür. Alg kolonisi büyüme kinetiği, denklem $\mu = \frac{\mu_{\max} S}{K_s + S}$ monod modeli ile hesaplanmıştır. Burada, μ spesifik büyüme hızı, $\mu_{\max}$ maksimum spesifik büyüme hızı, S besin konsantrasyonu, yani modeldeki t zamanındaki uygunluk değeri $\left(f^t(x_i) \right)$ ve K alg kolonisi substrat yarı doygunluk sabitidir. $\mu_{\max}$, 1 olarak kabul edilmiştir (kütle korunumu ilkesine göre biyokütleye dönüştürülen maksimum miktarın, birim zamanda tüketilen substrat miktarına eşit olması gerektiğinden). K, t zamanında alg kolonisi yarı besin koşullarındaki büyüme hızı olarak hesaplanmıştır (Özkış ve Babalık , Uymaz vd. 2015).

Monod denkleminde t + 1 zamanındaki *i*-inci alg kolonisi boyutu Eşitlik 3.6'da verilmiştir:

$$G_i^{t+1} = \mu_i^t G_i^t \qquad i = 1, 2, ..., N \tag{3.6}$$

Burada G_i^t, t zamanındaki *i*-inci alg kolonisi boyutu, N ise sistemdeki alg kolonisi sayısıdır. İyi çözümler (maliyet etkin ve en uygun çözüm) sağlayan alg kolonisi, elde ettiği besin miktarı arttıkça daha fazla büyür. Evrimsel süreçte en küçük alg kolonisinin her bir alg hücresinin ölümüyle birlikte, en büyük alg kolonisi alg hücresi çoğalır. Bu davranış süreci eşitlik (3.7)-(3.9) ile sunulmuştur:

$$biggest^t = \arg\max\ size(x_i^t), \qquad i = 1, 2, \ldots N \tag{3.7}$$

$$smallest^t = \arg\max\ size(x_i^t), \qquad i = 1, 2, \ldots N \tag{3.8}$$

$$smallest_m^{t+1} = biggest_m^{t+1}, \qquad m = 1, 2, \ldots D \tag{3.9}$$

Burada D problem boyutu, biggest en büyük alg kolonisi ve smallest en küçük alg kolonisi olarak tanımlanmıştır. YAA'da alg kolonileri t zamanında boyutlarına göre sıralanır. Rastgele seçilen herhangi bir boyutta, en küçük alg kolonisi hücresi ölür ve en büyük alg kolonisi hücresi kendini çoğaltır (Özkış ve Babalık , Uymaz vd. 2015).

Adaptasyon

Adaptasyon, bir ortamda yeterince büyüyemeyen alg kolonisi, kendisini ortama uyum sağlamaya çalışır ve sonuç olarak baskın türler değişir. Adaptasyon, yeterince büyüyemeyen bir alg kolonisi, kendisini ortamındaki en büyük alg kolonisine benzetmeye

çalıştığı süreçtir. Bu süreç, algoritmada açlık seviyesindeki değişiklik ile sonuçlanır. Her yapay alg için başlangıç açlık değeri sıfırdır. Alg hücresi yetersiz ışık aldığında açlık değeri *t* zamanı ile birlikte artar. En yüksek açlık değerine sahip yapay algin (eşitlik 3.10), adapte olma süreci eşitlik 3.11'de modellenmiştir.

$$starving^t = \max A_i^t, \qquad i = 1, 2, ..., \mathrm{N} \tag{3.10}$$

$$starving^{t+1} = starving + (biggest^t - starving^t) \times rand \tag{3.11}$$

Burada A_i^t, *t* zamanındaki *i*-inci alg kolonisi açlık değeri, starving^t ise *t* zamanındaki en yüksek açlık değerine sahip alg kolonisidir. Adaptasyon parametresi (Ap), adaptasyon sürecinin *t* zamanında uygulanıp uygulanmayacağını belirler ve *Ap*, [0, 1] aralığında değer alan bir sabittir (Özkış ve Babalık , Uymaz vd. 2015).

Helisel Hareketi

Alg hücreleri ve kolonileri genellikle hayatta kalmak için yeterli ışığın bulunduğu su yüzeyine yakın kalmaya çalışırlar. Yerçekimi ve viskoz sürtünme tarafından kısıtlanan ileri hareketi sağlayan kamçıları ile sıvıda helis şeklinde yüzerler. Alg hücrelerinin hareketleri farklıdır. Büyüyen alg hücresinin sürtünme yüzeyi büyüdükçe, yerel arama yeteneklerini artırarak helis hareketlerinin sıklığı artar. Her alg hücresi enerjisine orantılı olarak hareket edebilir. Bir alg hücresinin *t* zamanındaki enerjisi, o zamandaki besin alım miktarı ile doğru orantılıdır. Bu nedenle, bir alg hücresi yüzeye ne kadar yakınsa, o kadar fazla enerjisi vardır ve sıvı içinde hareket etme şansı daha fazladır. Tam tersine, sürtünme yüzeyi daha azdır; sıvı içindeki hareket mesafesi daha uzundur. Bu nedenle, küresel arama kabiliyetleri daha fazladır. Ancak, enerjisine orantılı olarak daha az hareket edebilirler. Bir alg hücresinin hareketi, gerçek hayatta olduğu gibi helis şeklindedir. YAA'da, hareketi kısıtlayan yerçekimi 0 olarak ve viskoz sürtünme, alg hücresinin boyutuna orantılı olan kesme kuvveti olarak gösterilir. Küresel bir şekle sahiptir ve boyutu modelde hacmi olarak kabul edilir. Bu nedenle, sürtünme yüzeyi yarım kürenin yüzey alanı olarak belirlenir:

$$\tau(x_i) = 2\pi \left(\sqrt[3]{\frac{3G_i}{4\pi}} \right)^2 \tag{3.12}$$

Burada $\tau(x_i)$ sürtünme yüzeyidir. Alg hücresinin helis hareketine yönelik üç boyut rastgele belirlenir. Bunlardan biri eşitlik 3.13'teki doğrusal hareketi, diğer iki boyut ise eşitlik 3.14 ve eşitlik 3.15'deki açısal hareketi sağlar. Eşitlik 3.13 tek boyutlu problemler için kullanılır ve alg hücresi/kolonisi tek yönde hareket eder. İki boyutlu problemlerde alg hareketi sinüsoidaldir ve bu nedenle eşitlik 3.14 ve 3.15 kullanılır. Üç veya daha fazla boyutta, alg hareketi helis şeklindedir ve eşitlik 3.13-3.15 kullanılır. Sürtünme yüzeyi ve ışık kaynağına olan mesafe, hareketin adım boyutunu belirler (Özkış ve Babalık , Uymaz vd. 2015).

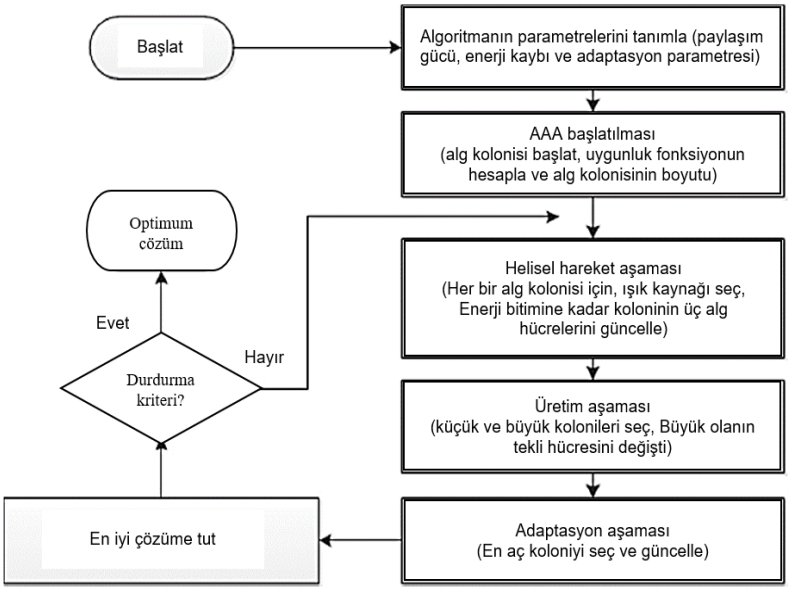
$$x_{im}^{t+1} = x_{im}^{r} + \left(x_{jm}^{t} - x_{im}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right) p \tag{3.13}$$

$$x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)(\Delta - \tau^{t}(\mathrm{x}_i))\cos\alpha \tag{3.14}$$

$$x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)(\Delta - \tau^{t}(\mathrm{x}_i))\sin\beta \tag{3.15}$$

Burada X_{ik}^{t}, X_{il}^{t} ve X_{im}^{t}, *t* zamanındaki *i*-inci alg hücresinin sırasıyla *x, y* ve *z* koordinatları, $\alpha, \beta \in [0, 2\pi]$; $p \in [-1,1]$; Δ kesme kuvveti, $\tau^{t}(x_i)$ ise *i*-inci alg hücresinin *t* zamanındaki sürtünme yüzey alanıdır (Özkış ve Babalık , Uymaz vd. 2015).

YAA'nın genel çalışma mekanizmasını özetleyen akış diyagramı aşağdaki görselde sunulmuştur (Uymaz vd. 2015).



Şekil 3.1. YAA'nın akış diyagramı

3.6. Levy Uçuşu

Fransız matematikçi Paul Lévy, 1937 yılında lévy uçuşunu ortaya atmıştır. Bu, kendisinden yaklaşık bir yüzyıl önce keşfedilen daha geleneksel Brownian hareketinin ötesine geçen istatistiksel bir arama yaklaşımıdır. Her nesil, bir dizi öğe kullanılarak en iyi bilinen konumdan başlar ve yeni bir nesil rastgele mesafelerde oluşturulur. En umut verici nesil seçildikten sonra, belirli bir durdurma koşulu karşılanana kadar prosedür tekrar tekrar devam eder. Genel olarak, hayvanlar besin ararken oldukça rastgele hareket ederler. Bu nedenle, her rastgele hareket, bir sonraki hareketin mevcut konuma ve bir sonraki konuma ilerleme olasılığına bağlı olduğu için son derece önemlidir. Son araştırmalara göre, rastgele hareket modelinde en etkili arama taktiklerinden birinin Levy uçuşu olduğu ifade edilmiştir (Pavlyukevich 2007, Reynolds ve Frye 2007).

Lévy uçuşu modelleme, finans ve fizik gibi alanlarda yaygın olarak uygulanan olasılık teorisinden ve fraktal matematikten türetilen bir kavramdır. Ticarette, lévy uçuşları, daha büyük, daha az sıklıktaki hareketlerle serpiştirilmiş küçük, sık hareketlerle karakterize edilen rastgele bir yürüyüş sürecini tanımlar. Bu davranış, piyasaların genellikle keskin, oynak hareketlerin izlediği nispeten sakin dönemler sergilediği gerçek dünya fiyat dinamiklerini yansıtır. Lévy uçuş modeli, daha küçük olanları yumuşatırken aşırı fiyat değişikliklerini büyüten bir ağırlıklandırma mekanizması sunarak piyasa eğilimlerine daha ayrıntılı bir bakış açısı sağlar (Pavlyukevich 2007, Reynolds ve Frye 2007).

Kararlı gauss dışı rastgele süreçlerin bir türü Levy uçuşudur. Levy kararlı dağılımı kullanılarak rastgele hareket üretilmektedir. Gerçekte, bu dağılım basit bir güç yasası formülüdür. Levy dağılımı, matematiksel olarak eşitlik 3.16'daki gibi tanımlanabilir (Chechkin vd. 2008, Yang 2010, Yang ve Deb 2013):

$$L = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} \exp\left[-\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{3/2}} & eğer \quad 0 < \mu < s < \infty \\ 0 & eğer \quad s \le 0 \end{cases} \tag{3.16}$$

Ölçek parametresi $\gamma > 0$ dağılım ölçeğini düzenlerken, μ konum veya yer değiştirme parametresidir. Levy dağılımı genellikle fourier dönüşümü ile en iyi şekilde tanımlanır.

$$F(k) = \exp\left[-\alpha\left|k\right|^{\beta}\right], \qquad 0 < \beta \le 2 \tag{3.17}$$

Çarpıklık veya ölçek faktörü olarak bilinen α, [-1, 1] aralığında düşen bir değerdir. Kararlılık indeksi $\beta \in (0, 2]$ için başka bir ad da Levy indeksidir. Genel β için, birkaç özel durum dışında, integralin analitik formu bilinmemektedir. Dağılımı büyük ölçüde β ve α parametreleri belirlerken, γ ve μ parametrelerinin etkisi azdır. Özellikle kuyruk bölgesinde, β değerini ayarlayarak farklı olasılık dağılımı şekilleri elde edilebilir [53,52].

Parametre β daha küçük olduğunda, saçılma daha büyük sıçramalar yapar çünkü daha uzun bir kuyruk oluşturacaktır. Eğrinin çarpıklık yönünü gösteren, çarpıklık parametresi α'nın işaretleridir. Sağa doğru yön pozitif sayılar, sola doğru yön ise negatif değerler ile temsil edilir. $\alpha = 0$ olduğunda dağılımda simetri vardır. Dağılımın zirveleri, son iki parametre olan γ genişliği ve μ değişimi ile belirlenir. Dağılım, β parametresinin değerine bağlı olarak değişir. Daha küçük sayılar daha uzağa sıçramasına, daha büyük değerler ise daha kısa sıçramasına neden olur (Lee ve Yao 2001, Chechkin vd. 2008, Al-Temeemy vd. 2010, Yang 2010, Yang ve Deb 2013).

3.7. Levy Uçuşunun Yapay Alg Algoritmasına Uygulanması

YAA, optimizasyon problemlerini çözmek için esnek ve uyarlanabilir bir çerçeve sunar. Keşif ve sömürüyü etkin bir şekilde dengelemek için biyolojik olarak ilham alınan mekanizmalarını kullanır. Algoritma, çözümün uygunluğunu temsil eden enerji seviyesi ile alglerin enerji birikimi ve dağılımını modeller. Helisel hareket gibi anahtar işlemler, yerel keşfi artırmak için alglerin spiral hareketini taklit eder. Ayrıca, enerji metabolizması, erken yakınsamayı önlemek için sömürüyü düzenler. Dahası, açlık ve üreme mekanizmalarının dahil edilmesi, popülasyon içinde çeşitliliği sağlar ve küresel arama yeteneklerini destekler.

YAA'nın yerel keşfe ve enerjiye dayalı güncellemelere bağımlılığı, özellikle çok modlu veya karmaşık optimizasyon manzaralarında, umut verici performansına rağmen arama sürecinde durgunluğa veya erken yakınsamaya yol açabilir. Bu, çözüm uzayının uzak bölgelerini etkin bir şekilde keşfetmede potansiyel bir sınırlama olduğunu gösterir.

YAA'ya Levy uçuşunun dahil edilmesi, arama uzayında büyük, stokastik sıçramalar için bir mekanizma getirerek bu zayıflığı gidermek için kritik bir gelişme sağlar. Ağır kuyruklu bir adım büyüklüğü dağılımı ile karakterize edilen Levy uçuşu, algoritmanın yerel optimum noktalarından kaçmasına ve keşfedilmemiş bölgelere arama alanını genişletmesine olanak tanır. Böylece YAA'nın helisel hareket gibi mevcut yerel keşif mekanizmalarını tamamlar. Bu yetenek, çözüm manzarasının engebeli veya çok

modlu olduğu yüksek boyutlu ve karmaşık problemler için özellikle hayati önem taşır. Helisel hareket ayrıntılı, yerel keşfe odaklanırken, Levy uçuşu küresel keşfi kolaylaştırır ve daha dengeli ve kapsamlı bir arama süreci sağlar. Levy uçuşuna olan ihtiyaç, YAA'nın yerel ve küresel keşif arasında sık geçişler gerektiren optimizasyon görevlerini ele alma konusundaki içsel sınırlamalarından kaynaklanmaktadır. Bu boşluğu kapatarak, Levy uçuşu yalnızca algoritmanın sağlamlığını ve uyarlanabilirliğini artırmakla kalmaz, aynı zamanda dinamik ve belirsiz ortamlarda küresel optimum noktayı verimli bir şekilde tanımlama yeteneğini de geliştirir.

Tez çalışmasında Levy uçuşu YAA'nın popülasyonunda bulunan herhangi bir çözüme uygulanabilmektedir. Burada belirlenen en çok başarısız olma değişkeni (LFMax) önemlidir. Her bir çözüm için ayrı ayrı oluşturulan bu değişken, bir çözümün üst üste kaç başarısız iterasyondan sonra Levy uçuşuna tabi tutulacağını temsil etmektedir. Her iterasyon adımında daha iyiye gitmeyen çözümler için başarısızlık sayacı 1 arttırılır. Bir çözüm için başarısızlık sayısı maksimum başarısız olma seviyesine ulaştığında bu çözüme Levy uçuşu uygulanarak konum güncellemesi yapılır. Bu durumda bu çözüm için sayaç sıfırlanmaktadır. Ayrıca iterasyon sonucunda çözümler daha iyi bir konuma gidiyorsa sayaç gene sıfırlanmaktadır. Bu şekilde üst üste belli bir sayıda daha iyi konuma gidemeyen çözümler takıldığı noktadan kurtarılarak olası daha iyi bir konuma yönlendirilir.

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu çalışmada, Levy uçuşu tabanlı yapay alg algoritmasının hiyerarşik olmayan kümeleme problemlerine uygulanması için UCI deposundan elde edilen 15 veri seti (appendicitis, banknote, blobs, circles, diagnosis_II, flame, hear, ıonosphere, iris2d, jain, liver, sonar, very-density, vertebral3 ve wine) kullanılmıştır. Uygunluk fonksiyonu olarak toplam karesel hata (SSE) değeri kullanılmıştır. Algoritmaların kümeleme performansını karşılaştırmak amacıyla ARI metriği kullanılmıştır.

Deneyler 11th gen Intel® Core™ i7-1165G7 @ 2.80 GHz işlemci, 8 GB RAM ve Windows 10 (64-bit) Professional işletim sistemi özelliklerine sahip bir bilgisayarda gerçekleştirilmiştir.

4.1. Önerilen Algoritmanın Parametre Ayarlanması

Algoritmaların kümeleme performansını ölçmek amacıyla 30 tekrar ile veri setlerine uygulanmıştır. Tüm algoritmalar için ortak olan parametreler: popülasyon boyutu (*N*) 40, maksimum değerlendirme sayısı (*maxFes*) 10000 ve çalıştırma sayısı (*r*) 30 olarak kullanılmıştır. Bununla beraber algoritmaların kendine ait özel parametreleri orijinallerinde verilen değerler olarak kullanılmıştır. Önerilen algoritmanın (YAA_LU) kendine has paremetreleri ise çizelge 4.1'de verilmiştir.

Çizelge 4. 1. Önerilen algoritmanın parametreleri

Parametre	Parametre açıklanması	Parametre değerleri
K	Kesme kuvveti	2
Le	Enerji kaybı	0.3
Ap	Adaptasyon	0.5
Bata	Levy dağılımı	1.5
LFMax	Levy uçuşu uygulanması için üst üste beklenen maksimum başarısızlık sayısı	10

Bu parametre değerleri ile algoritmaların çalıştırılması sonucu elde edilen değerler SSE ve ARI metriği ile detaylandırılmıştır.

4.2. SSE Sonuçları

Uygunluk fonksiyonu olarak kullanılan SSE değeri, algoritmaların her veri setine 30 defa uygulanması ile elde edilmiştir. Algoritmalara ait SSE değerlerinin sunulduğu çizelge 4.2'de; 30 çalışmanın ortalaması (Ort.) ve standart sapması (Std.) ile gösterilmiştir. Ayrıca, sıralama (S) sütunu, her algoritmanın performans başarı sıralamasını göstermek için kullanılmıştır. SSE değerinin, dolayısıyla küme merkezi ile örnekler arasındaki farklılığın minimum olması istenmektedir. Daha düşük bir SSE değeri daha iyi performans anlamına gelmektedir.

Çizelge 4. 2. SSE metriği sonuçları

Veri Seti	Metrik	YAA_LU	YAA	KAO	DE	GAO	PSO	ATA	BOA
Appendicitis	Ort	**17.44**	17.70	19.00	26.76	18.49	42.95	17.84	21.6
	Std	**0.01**	0.18	3.11	2.36	2.16	6.13	0.30	3.5
	S	**1**	2	5	7	4	8	3	6
Banknote	Ort	**477.94**	**477.94**	478.38	670.38	**477.94**	839.02	**477.94**	489.5
	Std	**0.00**	**0.00**	0.67	66.28	**0.00**	137.80	**0.00**	15.0
	S	**1**	**1**	5	7	**1**	8	**1**	6
Blobs	Ort	**409.50**	409.51	**409.50**	473.44	409.75	577.29	409.52	413.4
	Std	**0.00**	0.01	**0.00**	35.32	409.75	51.72	0.02	18.8
	S	**1**	3	**1**	7	5	8	4	6
Circles	Ort	**607.50**	**607.50**	**607.50**	634.70	607.51	693.40	**607.50**	608.3
	Std	**0.00**	**0.00**	**0.00**	15.12	0.05	30.70	**0.00**	2.1
	S	**1**	**1**	**1**	7	5	8	**1**	6
Diagnosis_II	Ort	**67.78**	68.61	69.77	81.42	69.60	114.56	71.08	73.7
	Std	**0.12**	0.64	2.21	4.52	2.58	6.27	1.34	4.5
	S	**1**	2	4	7	3	8	5	6
Flame	Ort	**7.97**	**7.97**	**7.97**	7.99	**7.97**	9.65	**7.97**	8.0
	Std	**0.00**	**0.00**	**0.00**	0.04	**0.00**	1.00	**0.00**	0.0
	S	**1**	**1**	**1**	6	**1**	8	**1**	7
Heart	Ort	**245.96**	250.10	272.54	315.51	270.84	427.82	261.59	292.7
	Std	**0.27**	2.27	142.27	16.95	23.19	32.73	3.39	23.5
	S	**1**	2	5	7	4	8	3	6
Ionosphere	Ort	**2496.06**	2519.96	**2496.06**	3997.00	2777.60	4607.93	2954.86	2840.2
	Std	**16.47**	25.51	**0.00**	97.23	295.39	152.79	92.44	209.7
	S	**1**	3	**1**	7	4	8	6	5
Iris2D	Ort	**1.41**	**1.41**	**1.41**	1.84	**1.41**	4.30	1.42	1.44
	Std	**0.00**	**0.00**	**0.00**	0.36	**0.00**	1.18	0.00	0.1
	S	**1**	**1**	**1**	7	**1**	8	5	6
Jain	Ort	**18.20**	**18.20**	**18.20**	18.36	**18.20**	22.61	**18.20**	**18.20**
	Std	**0.00**	**0.00**	**0.00**	0.29	**0.00**	2.70	**0.00**	**0.0**
	S	**1**	**1**	**1**	7	**1**	8	**1**	**1**
Liver	Ort	**22.92**	23.09	25.26	35.36	23.20	81.32	23.05	29.1
	Std	**0.00**	0.19	3.32	8.85	0.88	20.23	0.10	3.9
	S	**1**	3	5	7	4	8	2	6
Sonar	Ort	**553.58**	587.94	604.11	1160.89	758.43	1565.89	849.66	655.1
	Std	**15.99**	28.79	42.01	78.45	73.97	82.74	31.79	50.4
	S	**1**	2	3	7	5	8	6	4
Vary-density	Ort	**2.48**	**2.48**	2.56	3.29	**2.48**	4.86	**2.48**	2.7
	Std	**0.00**	**0.00**	0.32	0.52	**0.00**	0.78	**0.00**	0.5
	S	**1**	**1**	5	7	**1**	8	**1**	6
Vertebral3	Ort	**15.23**	17.49	19.25	31.37	18.38	69.88	19.04	25.0
	Std	**0.24**	0.69	4.10	4.50	2.26	19.66	0.80	25.0
	S	**1**	2	5	7	3	8	4	6
Wine	Ort	**29.96**	38.75	46.52	65.80	48.83	124.32	56.65	52.9
	Std	**1.05**	2.83	8.40	6.48	8.96	13.89	3.41	7.8
	S	**1**	2	3	7	4	8	6	5
Ort. Sıralama		**1.00**	1.80	3.07	6.93	3.07	8.00	3.27	5.47

Çizelge 4.2'deki deneysel sonuçlar, algoritmaların uygunluk fonksiyonu üzerinde elde ettiği başarıları göstermektedir. Önerilen algoritma ile beraber 7 adet optimizasyon algoritmasının 30 çalıştırma sonucunda elde edilen ortalama başarıları, standart sapması ve başarı sıralaması sunulmuştur. Diğer algoritmalara kıyasla orijinal YAA algoritması daha başarılı sonuçlar vermiştir. YAA'nın performansını arttırmak amacıyla uygulanan

Levy uçuşu ile beraber önerilen algoritma daha da başarılı sonuçlar elde etmiştir. Her veri seti için elde edilen sonuçlara göre algoritma bazında başarı sıralamalarına bakıldığında 1,00 değeri ile önerilen algoritma ilk sırada yer almaktadır. Önerilen algoritmayı orijinal YAA algoritması 1,80 başarı sıralaması değeri ile takip etmektedir. PSO ise hiçbir veri setinde istenilen başarıyı sağlayamamış ve 8,00 sıralama değeri ile son sırada yer almaktadır. Bununla beraber diğer algoritmalar bazı veri setlerinde başarılı olurken bazılarında başarız olarak farklı sıralamaları yakalamışlardır.

Sonuçlar genel olarak değerlendirildiğinde orijinalinde başarılı olan YAA algoritmasının yapılan Levy uçuşu katkısı ile daha başarılı bir performans gösterdiği görülmektedir.

4.3. ARI Metriği Sonuçları

Algoritmaların ARI metriği kullanılarak elde ettiği sonuçlar çizelge 4.3'te sunulmuştur. Buradaki değerler 30 bağımsız çalıştırma sonucunda elde edilen sonuçlardır. 30 çalıştırmanın ortalama, standart sapma değerleri ve ortalama değerine göre başarı sıraları sunulmuştur. Negatif değerler de alabilen ARI metriğinin büyük olması istenilen bir durumdur. Daha büyük metrik değerleri daha başarılı sonuçlar anlamına gelmektedir.

Çizelge 4. 3. ARI metriği sonuçları

Veri Seti	Metrik	YAA_LU	YAA	KAO	DE	GAO	PSO	ATA	BOA
Appendicitis	Ort	0.48	0.47	0.35	-0.07	0.47	0.02	**0.50**	0.33
	Std	0.03	0.01	0.26	0.13	0.17	0.19	**0.04**	0.31
	S	2	3	5	8	3	7	**1**	6
Banknote	Ort	**0.59**	**0.59**	**0.59**	-0.01	**0.59**	0.04	**0.59**	0.55
	Std	**0.00**	**0.00**	**0.01**	0.39	**0.00**	0.27	**0.00**	0.13
	S	**1**	**1**	**1**	8	**1**	7	**1**	6
Blobs	Ort	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.78	**1.00**	0.97
	Std	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.38	**0.00**	0.19
	S	**1**	**1**	**1**	**1**	**1**	8	**1**	7
Circles	Ort	**-0.12**	**-0.12**	**-0.12**	-0.14	**-0.12**	-0.20	**-0.12**	**-0.12**
	Std	**0.00**	**0.00**	**0.00**	0.04	**0.00**	0.08	**0.00**	**0.00**
	S	**1**	**1**	**1**	4	**1**	5	**1**	**1**
Diagnosis_II	Ort	0.23	0.15	**0.24**	0.18	**0.24**	0.19	0.20	0.23
	Std	0.16	0.20	**0.16**	0.17	**0.17**	0.15	0.19	0.20
	S	3	8	**1**	7	**1**	6	5	3
Flame	Ort	**0.04**	**0.04**	0.03	-0.06	0.02	-0.03	**0.04**	0.00
	Std	**0.00**	**0.00**	0.04	0.08	0.06	0.22	**0.00**	0.08
	S	**1**	**1**	4	8	5	7	**1**	6
Heart	Ort	**0.36**	0.35	0.13	-0.13	0.19	-0.07	0.28	0.11
	Std	**0.09**	0.10	0.21	0.21	0.24	0.16	0.16	0.18
	S	**1**	2	5	8	4	7	3	6
Ionosphere	Ort	**0.39**	0.38	0.31	0.03	0.21	0.09	0.35	0.35
	Std	**0.01**	0.01	0.10	0.12	0.23	0.16	0.04	0.11
	S	**1**	2	5	8	6	7	3	4
Iris2D	Ort	**0.94**	**0.94**	**0.94**	0.91	**0.94**	0.80	**0.94**	**0.94**
	Std	**0.00**	**0.00**	**0.00**	0.04	**0.00**	0.15	**0.00**	**0.00**
	S	**1**	**1**	**1**	7	**1**	8	**1**	**1**
Jain	Ort	**0.72**	**0.72**	**0.72**	0.71	**0.72**	0.70	**0.72**	0.71
	Std	**0.00**	**0.00**	**0.00**	0.01	**0.00**	0.11	**0.00**	0.01
	S	**1**	**1**	**1**	6	**1**	8	**1**	6
Liver	Ort	**0.01**	0.00	0.00	**0.01**	0.00	**0.01**	0.00	**0.01**
	Std	**0.01**	0.00	0.01	**0.01**	0.01	**0.01**	0.01	**0.01**
	S	**1**	5	5	**1**	5	**1**	5	**1**
Sonar	Ort	**0.03**	**0.03**	0.01	0.01	0.00	0.01	0.02	**0.03**
	Std	**0.04**	**0.02**	0.02	0.01	0.01	0.09	0.02	**0.03**
	S	**1**	**1**	5	5	8	5	4	**1**
Vary-density	Ort	**0.93**	**0.93**	0.88	0.63	**0.93**	0.43	**0.93**	0.86
	Std	**0.00**	**0.00**	0.18	0.31	**0.00**	0.28	**0.00**	0.20
	S	**1**	**1**	5	7	**1**	8	**1**	6
Vertebral3	Ort	0.12	**0.13**	-0.02	-0.02	-0.04	0.01	0.02	-0.03
	Std	0.08	**0.04**	0.07	0.04	0.06	0.05	0.10	0.03
	S	2	**1**	5	5	8	4	3	7
Wine	Ort	0.58	**0.85**	0.20	0.12	0.26	0.14	0.37	0.21
	Std	0.19	**0.04**	0.21	0.16	0.18	0.16	0.12	0.23
	S	2	**1**	6	8	4	7	3	5
Ort. Sıralama		**1.33**	2.00	3.40	6.07	3.33	6.33	2.27	4.40

Çizelge 4.3'teki sonuçlara bakıldığında, önerilen algoritmanın bu metrikte de en başarılı değerleri ürettiği görülmektedir. Bununla beraber appendicitis, diagnosis_II, vertebral3 ve wine veri setlerinde bu metrikte ilk sıralamayı elde edemediği dikkat çekmektedir. Bu 4 veri seti dışında kalan 11 veri setinde ise en başarılı sonuçları elde ettiği görülmektedir. Önerilen algoritma 1,33 başarı sıralaması ile tüm algoritmaları

geride bırakırken, orijinal YAA algoritması 2,00 başarı sıralaması ile ikinci sırada yer almıştır. SSE metrik sonuçlarında olduğu gibi PSO algoritması son sırada yer almasına karşın başarı sıralaması değerini yükseltmiştir. Diğer algoritmalar da veri setleri üzerindeki başarı sıralamasına göre farklı dereceleri elde etmişlerdir.

5. SONUÇ VE ÖNERİLER

Bu tez çalışmasında, optimizasyon algoritmaları kullanılarak hiyerarşik olmayan kümeleme problemine çözüm aranmıştır. 7 farklı optimizasyon (YAA, KAO, DE, GAO, PSO, ATA ve BOA) algoritması 15 farklı veri seti (appendicitis, banknote, blobs, circles, diagnosis_II, flame, hear, ionosphere, iris2d, jain, liver, sonar, very-density, vertebral3 ve wine) üzerinde test edilmiştir. Algoritmalar veri setlerindeki örnekleri en iyi temsil edecek küme merkezlerini belirlemek için çalıştırılmıştır. Algoritmaların performansını karşılaştırmak amacıyla ARI metriği kullanılmıştır. Kullanılan bu optimizasyon algoritmaları içerisinde YAA en başarı algoritma olmuştur. Ancak bu algoritmanın da eksik kaldığı durumlar olmuş ve iyileştirilmeye ihtiyaç duymuştur. Algoritmanın performansını arttırmak amacıyla Levy uçuşu stratejisi ilave edilerek (YAA_LU) kümeleme problemine uygulanmıştır. Yapılan bu ekleme ile önerilen algoritma hem orijinal YAA'yı hem de diğer 6 optimizasyon algoritmasını performans olarak geride bırakmıştır. Elde edilen deneysel sonuçlar sonuçlar algoritmanın başarılı olmasına rağmen halen geliştirilebilecek yönlerinin olduğunu ortaya koymuştur.

Gelecekteki çalışmalar için, önerilen yöntemin daha farklı yapıdaki (ikili, kesikli, kombinasyonel) problemlere uygulanarak performansı değerlendirilebilir. Ayrıca mevcut diğer optimizasyon algoritmaları ile hibrit bir şekilde kullanılabilir veya başka arama stratejileri ile desteklenebilir.

KAYNAKLAR

AKTO, İ., O. İNAN ve M. KARAKOYUN, 2019, Grey Wolf Optimizer (GWO) algorithm to solve the partitional clustering problem, *International Journal of Intelligent Systems and Applications in Engineering*, 7(4), 201-206.

Al-Temeemy, A. A., J. Spencer ve J. Ralph, 2010, Levy flights for improved ladar scanning, 2010 IEEE International Conference on Imaging Systems and Techniques, 225-228.

Ali, E., S. Abd Elazim ve A. Abdelaziz, 2017, Ant Lion Optimization Algorithm for optimal location and sizing of renewable distributed generations, Renewable Energy, 101(1311-1324.

Almodfer, R., M. Mudhsh, S. Chelloug, M. Shehab, L. Abualigah ve M. Abd Elaziz, 2022, Quantum mutation reptile search algorithm for global optimization and data clustering, Hum-Centr Comput Inf Sci, 12

Anderberg, M. R., 1973, The broad view of cluster analysis, Cluster analysis for applications, 1(1), 1-9.

Atamer, B., 1992, Kümeleme Analizi (Cluster Analysis) ve Kümeleme Analizinin İlaç Sektörüne Uygulanması, Marmara Universitesi (Turkey).

Chechkin, A. V., R. Metzler, J. Klafter ve V. Y. Gonchar, 2008, Introduction to the theory of Lévy flights, Anomalous transport: Foundations and applications, 129-162.

Cura, T., 2012, A particle swarm optimization approach to clustering, Expert Systems with Applications, 39(1), 1582-1588.

Eberhart, R. ve J. Kennedy, 1995, A new optimizer using particle swarm theory, MHS'95. Proceedings of the sixth international symposium on micro machine and human science, 39-43.

Everitt, B. ve G. Dunn, 2001, Applied multivariate data analysis, Wiley Online Library.

Gates, A. J. ve Y.-Y. Ahn, 2017, The impact of random models on clustering similarity, Journal of Machine Learning Research, 18(87), 1-28.

Hubert, L. ve P. Arabie, 1985, Comparing partitions, Journal of classification, 2(193-218.

Johnson, R. A. ve D. W. Wichern, 2002, Applied multivariate statistical analysis.

Karaboga, D. ve C. Ozturk, 2011, A novel clustering approach: Artificial Bee Colony (ABC) algorithm, Applied soft computing, 11(1), 652-657.

Kaur, A. ve Y. Kumar, 2022, A new metaheuristic algorithm based on water wave optimization for data clustering, Evolutionary Intelligence, 15(1), 759-783.

Kiran, M. S., 2015, TSA: Tree-seed algorithm for continuous optimization, Expert Systems with Applications, 42(19), 6686-6698.

Kuo, R.-J., Y. Zheng ve T. P. Q. Nguyen, 2021, Metaheuristic-based possibilistic fuzzy k-modes algorithms for categorical data clustering, Information Sciences, 557(1-15.

Lee, C.-Y. ve X. Yao, 2001, Evolutionary algorithms with adaptive lévy mutations, Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546), 568-575.

Li, P., T. Chen, Y. Liu, M. Cai, L. Sun, P. Wang, Y. Wang ve X. Zhang, 2025, Automatic Identification of Rock Discontinuity Sets by a Fuzzy C-Means Clustering Method Based on Artificial Bee Colony Algorithm, Applied Sciences, 15(3), 1497.

Lorr, M., 1983, Cluster analysis for social scientists, London, Jossey-Bass Publishers.

MacQueen, J., 1967, Some methods for classification and analysis of multivariate observations, Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, 281-297.

Mageshkumar, C., S. Karthik ve V. Arunachalam, 2019, Hybrid metaheuristic algorithm for improving the efficiency of data clustering, Cluster Computing, 22(435-442.

Manoranjani, M. ve S. Sukumaran, 2024, Enhanced Tree-Seed Continuous Optimization Algorithm for Optimal Multipath Routing in Wireless Sensor Networks, 2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS), 1566-1573.

Mat, A. N., O. İnan ve M. Karakoyun, 2021, An application of the whale optimization algorithm with Levy flight strategy for clustering of medical datasets, An International Journal of Optimization and Control: Theories & Applications (IJOCTA), 11(2), 216-226.

Mirjalili, S., 2015, The ant lion optimizer, Advances in engineering software, 83(80-98.

Mirjalili, S., 2015, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-based systems, 89(228-249.

Mirjalili, S. ve A. Lewis, 2016, The whale optimization algorithm, Advances in engineering software, 95(51-67.

Mostafa Bozorgi, S. ve S. Yazdani, 2019, IWOA: An improved whale optimization algorithm for optimization problems, Journal of Computational Design and Engineering, 6(3), 243-259.

Muthukrishnan, P. ve P. M. Kannan, 2023, Metaheuristics-based Clustering with Routing Technique for Lifetime Maximization in Vehicular Networks, Computers, Materials & Continua, 75(1).

Nasiri, J. ve F. M. Khiyabani, 2018, A whale optimization algorithm (WOA) approach for clustering, Cogent Mathematics & Statistics, 5(1), 1483565.

Nguyen, V., J. Epps ve J. Bailey, 2009, Information theoretic measures for clusterings comparison: is a correction for chance necessary?, International Conference on Machine Learning 2009, 1073-1080.

Özkış, A. ve A. Babalık, Çok amaçlı yapay alg algoritması ile kısıtlı mühendislik tasarım problemlerinin çözülmesi, Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 29(2), 183-193.

Pavlyukevich, I., 2007, Lévy flights, non-local search and simulated annealing, journal of computational physics, 226(2), 1830-1844.

Prakash, V. ve S. Pandey, 2023, Metaheuristic algorithm for energy efficient clustering scheme in wireless sensor networks, Microprocessors and Microsystems, 101(104898.

Premkumar, M., G. Sinha, M. D. Ramasamy, S. Sahu, C. B. Subramanyam, R. Sowmya, L. Abualigah ve B. Derebew, 2024, Augmented weighted K-means grey wolf optimizer: An enhanced metaheuristic algorithm for data clustering problems, Scientific reports, 14(1), 5434.

Price, K. V., R. M. Storn ve J. A. Lampinen, 2005, The differential evolution algorithm, Differential evolution: a practical approach to global optimization, 37-134.

Qaddoura, R., H. Faris, I. Aljarah ve P. A. Castillo, 2020, Evocluster: An open-source nature-inspired optimization clustering framework in python, Applications of Evolutionary Computation: 23rd European Conference, EvoApplications 2020, Held as Part of EvoStar 2020, Seville, Spain, April 15–17, 2020, Proceedings 23, 20-36.

Qin, A. K., V. L. Huang ve P. N. Suganthan, 2008, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE transactions on Evolutionary Computation, 13(2), 398-417.

Rand, W. M., 1971, Objective criteria for the evaluation of clustering methods, Journal of the American Statistical association, 66(336), 846-850.

Reynolds, A. M. ve M. A. Frye, 2007, Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search, PloS one, 2(4), e354.

Sahoo, S. K. ve A. K. Saha, 2022, A hybrid moth flame optimization algorithm for global optimization, Journal of Bionic Engineering, 19(5), 1522-1543.

Tatlıdil, H., 1992, Uygulamalı çok değişkenli istatistiksel analiz, Engin Yayınları.

UCI, 2024, UC Irvine Machine Learning Repository, https://archive.ics.uci.edu/, [Ziyaret Tarihi: 2024].

Uymaz, S. A., G. Tezel ve E. Yel, 2015, Artificial algae algorithm (AAA) for nonlinear global optimization, Applied soft computing, 31(153-171.

Van Thieu, N., D. Oliva ve M. Pérez-Cisneros, 2023, MetaCluster: An open-source Python library for metaheuristic-based clustering problems, SoftwareX, 24(101597.

Yadav, R. K. ve R. P. Mahapatra, 2022, Hybrid metaheuristic algorithm for optimal cluster head selection in wireless sensor network, Pervasive and Mobile Computing, 79(101504.

Yang, X.-S., 2010,A new metaheuristic bat-inspired algorithm, Nature inspired cooperative strategies for optimization (NICSO 2010), Springer, 65-74.

Yang, X.-S. ve S. Deb, 2013, Multiobjective cuckoo search for design optimization, Computers & Operations Research, 40(6), 1616-1624.

Yang, X., 2010, Engineering Optimization: An Introduction with Metaheuristic Applications, John Wiley & Sons.