ONDOKUZ MAYIS ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Dron Ağları İçin Yer İstasyonu Tasarımı

ZEYNEP SILA KAYMAK

CAN YILMAZ

CENKAY KUCUR

Danışman

Doç. Dr. Sercan DEMİRCİ

HAZİRAN 2024

T E Ş E K K Ü R

Proje boyunca yardımlarını esirgemeyen ve her konuda bilgisini paylaşan danışman hocamız Doç. Dr. Sercan DEMİRCİ'ye ve Arş.Gör.Dr. Doğan YILDIZ'a teşekkür ederiz.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

1T1

1T1

TABLO LİSTESİ

BÖLÜM: I

GİRİŞ

1

GİRİŞ

Günümüzde teknolojinin hızla ilerlemesi, birçok sektörde çeşitli yenilikleri beraberinde getirmektedir. Bu yeniliklerden biri de İnsansız Hava Araçları (İHA) veya popüler adıyla Dronelar olarak bilinen uzaktan kumandalı hava araçlarıdır. Başlangıçta askeri operasyonlarda hizmet veren insansız hava araçları (İHA'lar), kısa bir süre içinde çeşitli sektörlerde de uygulama alanı bulmuştur. İHA'lar, başlangıçta stratejik keşif ve gözetleme amaçlarıyla askeri alanda kullanılmış daha sonra tarım, güvenlik, medya ve çevre izleme gibi birçok alanda yaygın bir şekilde kullanılmıştır. Bu noktada, İHA'ların verimli bir şekilde kullanılabilmesi için uzaktan kumanda edilmesi ve kontrol edilmesi amacıyla özel yer istasyonlarına ihtiyaç duyulmuştur.

Figure 2: Örnek Drone Görseli

İHA'ların etkili bir şekilde kontrol edilmesi ve verimli bir iletişim kurabilmesi için Mobil Veri Dağıtım Sistemi (MBDS) mimarisi, özellikle uzaktan kumandalı hava araçlarıyla ilgili bir dizi kritik görevi yerine getirmek üzere tasarlanmıştır. MBDS, İHA'lar ile yer istasyonları arasında güvenli ve hızlı veri transferini sağlamak adına özel olarak geliştirilen bir iletişim altyapısı sunar. Bu mimari, belirli iletişim protokollerini benimseyerek İHA'larla yer istasyonları arasında düzenli bir veri akışını destekler.

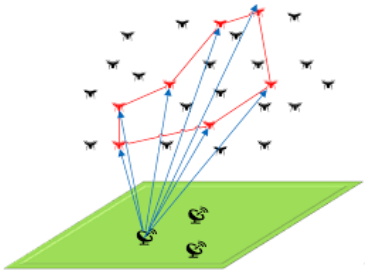

Figure 3: Yer İstasyonu Bağlantı Görseli

Temel olarak ele alacağımız problemimiz ise kapsama alanı içindeki İHA'mızın yer istasyonu veya yer istasyonlarıyla en hızlı, kısa şekilde bağlantı kurmasıdır. Birden fazla İHA yer istasyonunun kapsama alanında olabilir, bu durumda iniş önceliği kazanmalıdır. Bağlantı kurduğu yer istasyonuna iniş önceliği kazanması için bazı şartları sağlaması gerekmektedir. Örneğin bu şartlar; kalan pil süresi, saniyede gönderebileceği paket sayısı, istasyona olan uzaklığı ve İHA yer istasyonu ile bağlantı kurarken yol kaybı modeline de bakılmalıdır. Bu model sinyal gücünün mesafe ile ters orantılı olarak azaldığını varsayar. İHA yer istasyonundan uzaklaştıkça sinyal gücü kaybedilir ve bu da iletişim kalitesini etkiler. Ayrıca çevresel etkenler ve diğer engeller de göz önüne alınarak bir düzeltme faktörü eklenir. Gerekli parametreler de dahil edildiğinde iniş önceliğini kazanıyor ise inişini gerçekleştirir.

BÖLÜM: II

A M A Ç

2

AMAÇ

2.1 *Problem Tanımı*

Bu raporun amacı, drone istasyon atama problemini üç farklı senaryoda ele alarak çözümlemek ve optimizasyon algoritmalarıyla bu atamaları gerçekleştirmektir. Projemiz, sabit dron sabit istasyon, hareketli dron sabit istasyon ve hareketli dron hareketli istasyon senaryolarında drone'ların enerji tüketimini minimize etmeyi amaçlamaktadır. Bu çalışmada kullanılan algoritmalar, her bir senaryo için uygun metodolojiye dayalı olarak seçilip uygulanacak, elde edilen sonuçlar kapsamlı bir performans karşılaştırmasıyla değerlendirilecektir. Sonuç olarak, drone istasyon atama probleminde optimal çözümler bulmak ve bu algoritmaların pratik uygulamalarını değerlendirmek hedeflenmektedir.

2.1.1 *Problem çözümü*

Bu projemizde, drone'ların en iyi istasyonlara yerleşmesi için enerji optimizasyonu hedeflendi. Bunun için PSO (Particle Swarm Optimization), GWO (Grey Wolf Optimizer), ACO (Ant Colony Optimization), Genetik Algoritma, ABC (Artificial Bee Colony) ve Çekirge

Algoritması (Grasshopper Optimization Algorithm), Diferansiyel Gelişim Algoritması (Differential Evolution Algorithm) olmak üzere yedi farklı optimizasyon algoritması kullanıldı. Bu algoritmalar birbirleriyle karşılatırıldı ve her algoirtma 10000 iterasyonla çalıştırılarak en optimal çözümün elde edilmesi amaçlandı.

BÖLÜM: III

LİTERATÜR ÖZETİ

3

LİTERATÜR ÖZETİ

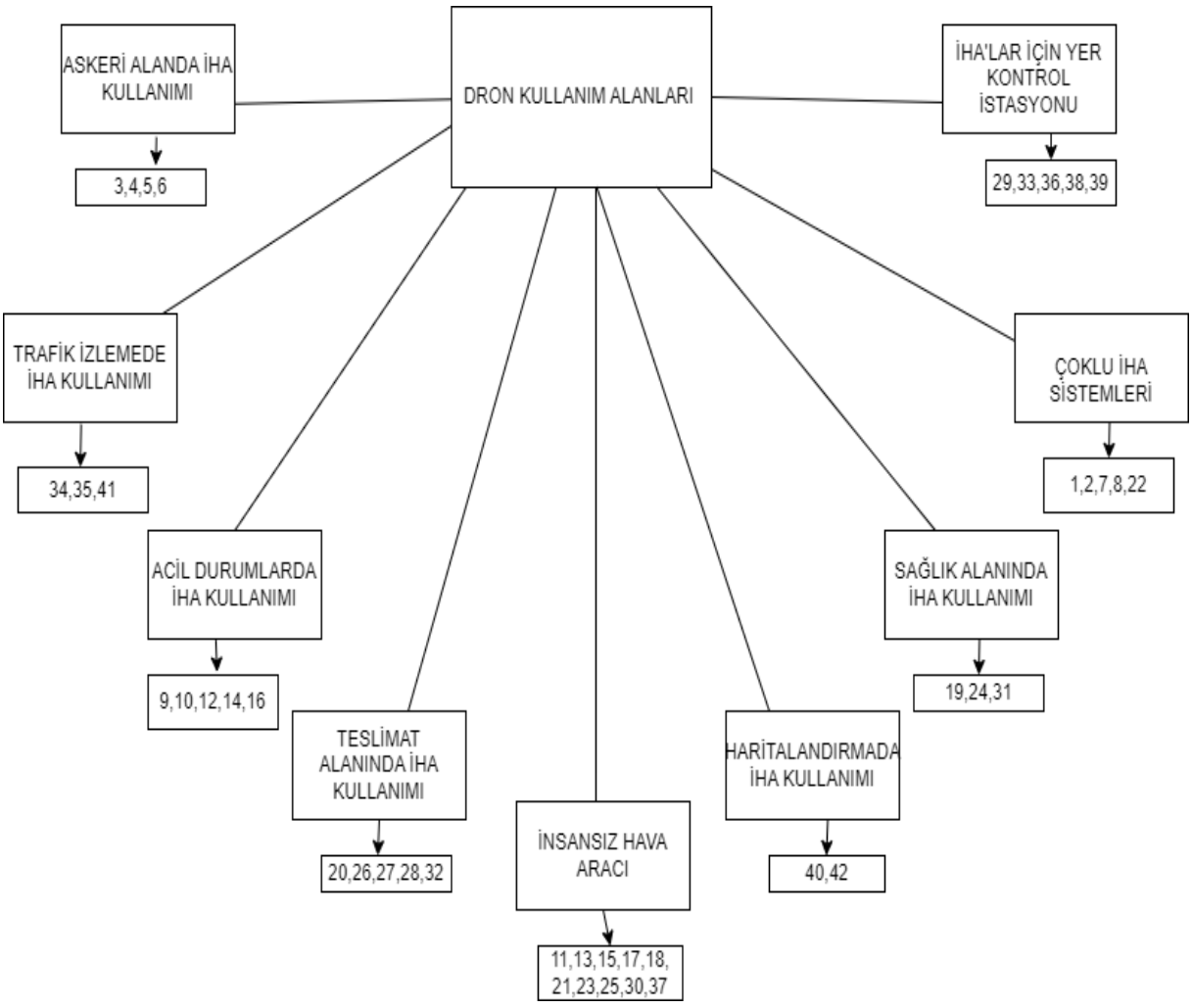

Figure 4: Literatür Diyagramı

Savunma sanayisine olan ilgi, İnsansız Hava Araçları (İHA) ve bu araçların kontrolünü sağlayan yer istasyonlarına olan talebi büyük ölçüde artırmıştır. Bu kapsamda, son yıllarda İHA'larla ilgili birçok çalışma yapılmış ve bu çalışmalar teknolojideki hızlı ilerlemeleri yansıtmaktadır.

İHA'lar, belirli rotalar üzerinde otonom veya uzaktan kumanda ile uçabilen ve belirli görevleri yerine getirebilen hava araçlarıdır [1]. Tarihsel olarak, İHA'lar farklı amaçlarla kullanılmıştır. Savaş dönemlerinde uçak ve pilot kayıplarının artması, İHA teknolojisinin gelişimini hızlandırmıştır [2]. Örneğin, İHA'lar ilk olarak patlayıcı içeren insansız balonlarla Venedik şehrini bombalamada kullanılmıştır ve bu askeri kullanımlar, sonraki çalışmalara ilham kaynağı olmuştur [3].

Quadrotor kullanılarak askeri gözlem amaçlı bir İHA simülasyonu geliştirilmiştir [4]. Bu çalışmada, Haar-Like özellikleri ve Adaboost nesne tespiti için kullanılmış, genişletilmiş Kalman filtresi ise İHA'nın konumunu belirlemek amacıyla tercih edilmiştir. Araştırmanın sonuçları, tüm durumlar için ortalamada sadece 0,24 hata olduğunu göstermektedir.

FANET ağlarının askeri uygulamalarına yönelik bir diğer çalışma [5], radyo-elektronik savaşa karşı direnç sağlamak için GPS düzeltme kanallarını entegre eden ve OSI modelinin farklı katmanlarında ağ yönetimini iyileştirmeyi amaçlayan bir model sunmaktadır. Çalışma, MU-MIMO teknolojileri kullanarak konumlandırma bilgilerini öne çıkarma ve hizmet bilgilerinin hacmini azaltma stratejilerini içermektedir.

İHA'ların stratejik kullanımı, sivil kayıpları minimize etme, önemli hedefleri etkisiz hale getirme ve örgütlerin saldırı yeteneklerini sınırlama kapasitesine odaklanmaktadır [6].

İHA'lar askeri alan dışında da birçok farklı alanda kullanılmaktadır: arama-kurtarma, afet yardımı, trafik izleme, haritalandırma, teslimat ve yangın söndürme [7].

Doğal afetlerde, geleneksel iletişim altyapısının çökmesi durumunda, acil kablosuz iletişim ağlarının hızlı bir şekilde kurulması önemlidir [8]. İHA'lar, bu tür durumlarda hızlı zarar değerlendirmesi ve kurtarma tedbirleri geliştirmek için kullanılabilirler [9]. Arama-kurtarma robotları, afet bölgelerine hızlı ulaşım ve bilgi toplama süreçlerinde önemli bir rol oynayabilirler [10].

İHA tabanlı iletişim ve ağ kurma konsepti akademik alanda büyük ilgi görmektedir [11]. Felaket bölgelerinde hayatta kalanları en yakın kurtarma kampına yönlendirmek amacıyla bir acil durum Wi-Fi ağı geliştirilmiştir [8]. Raspberry PI kartı, İHA üzerinde entegre edilerek afet bölgelerinde hızlı Wi-Fi altyapısı kurma amacıyla kullanılmıştır.

Piyasada yaygın olarak kullanılan üç ana İHA türü vardır: döner kanatlı İHA, sabit kanatlı İHA ve dikey kalkış ve iniş (VTOL) sabit kanatlı İHA [12]. Döner kanatlı İHA'ların havada asılı kalma yeteneği ve küçük rotor çapları, güvenlik ve tercih edilme nedenleridir [13]. Acil durum iletişimi için döner kanatlı İHA'lar kablosuz baz istasyonu olarak kullanılabilir [14]. İnsansız hava sistemleri, ticari ve kişisel kullanım potansiyeli açısından geniş bir yelpazeye sahiptir. Örneğin, acil durumlarda otomatik harici defibrilatörlerin kalp krizi geçiren kişilere ulaştırılması gibi çeşitli uygulamalar, bu teknolojinin etkinliğini göstermektedir [15]. Çoğu insansız hava sistemi, çevresel sürdürülebilirlik ve büyük çevresel olayları izlemek için kullanılmaktadır [16].

Nesnelerin İnterneti (IoT), ekonomik ve sağlık çözümleri için giderek daha fazla önem kazanmaktadır [17], [18]. Sağlık sistemlerinde, hastaların verilerinin iletilmesi ve uzaktan sağlık takibi gibi alanlarda kullanılmaktadır [19]. İHA'lar aracılığıyla ilaç ve kan teslimatı da önemli bir kullanım alanıdır [20]. Otonom ilaç dağıtım sistemleri, acil durumlarda hızlı ve etkili teslimat sağlamaktadır [21].

Şehir içi alanlarda İHA'ların teslimat için kullanımı, karadan teslimat trafiği sorunlarını çözmek amacıyla düşünülmektedir [20]. İHA'ların bir kalkışta birden fazla müşteriye hizmet verme hedefi doğrultusunda yapılan araştırmalar, enerji tüketimini en aza indirme ve görev tahsis modelleri geliştirilmiştir [22].

İHA'lar gözlemleme ve veri aktarımında kritik bir rol oynamaktadır. Günümüzde, çoklu İHA sistemleriyle birlikte karmaşık görevler de tamamlanabilmektedir [9]. Birden fazla İHA içeren ağlar, ölçeklenebilirlik ve kapsama alanı açısından avantajlıdır [23].

İHA'lar farklı yörünge türlerini izleyebilme ve havada sabit kalabilme yeteneklerine sahiptir. Bu durum, İHA'nın kapsama alanını etkilemektedir [24, 25].

Akıllı şehirlerde İHA'lar geniş bir uygulama yelpazesine sahiptir ancak bazı iş ve teknik zorluklar bulunmaktadır [26]. İHA'lar izleme gibi kötü niyetli amaçlarla kullanılabilir, bu da etik ve gizlilik sorunlarına yol açabilir. Ayrıca, maliyet ve arıza olasılıkları da dikkate alınmalıdır.

Artan araç sayısı, trafik izlenebilirliğini güçleştirerek yeni teknolojilere olan talebi artırmıştır. İHA'lar trafik izleme için etkili bir çözüm sunmaktadır [27]. İHA'lar devriye gezerek gözetimde de kullanılabilir [28]. Kavşaklarda araç sayısı toplamak ve bu bilgileri bulut sunucusunda depolamak gibi amaçlarla İHA'lar kullanılmaktadır [29]. Yüksek takip doğruluğu sunan sistemler, hava koşullarının etkisi ve genel kullanılabilirliği göz önünde bulundurarak değerlendirilmelidir [30].

Yer Kontrol İstasyonları, İHA'nın uçuş ve rotalama kabiliyetlerini denetlemek, veri toplamak ve işlemek amacıyla kullanılır [30]. Yer istasyonundan İHA'nın irtifası, hızı, pil durumu ve konumu gibi veriler izlenebilir [31]. Yer kontrol istasyonları, operatör performansını artırma potansiyeline sahiptir [32].

Haritalama alanında İHA'lar, perspektif resimlerdeki hataları düzeltme ve ortofoto veya foto-planlar üretme gibi çeşitli uygulamalarda kullanılmaktadır [33]. Bitki türlerinin tanımlanması ve tür bolluğunun tahmin edilmesi gibi çalışmalar da yapılmıştır [34].

BÖLÜM: IV

YÖNTEM

4

YÖNTEM

Bu projede, drone'ların en uygun istasyonlara yerleşmesini sağlamak amacıyla yedi farklı optimizasyon algoritmasını kullanıldı. Kullanılan algoritmalar şunlardır: PSO (Particle Swarm Optimization), GWO (Grey Wolf Optimizer), ACO (Ant Colony Optimization), Genetik Algoritma, ABC (Artificial Bee Colony) ve Çekirge Algoritması (Grasshopper Optimization Algorithm), DEA(Differential Evolution Algorithm) . Her bir algoritma, drone'ların ve istasyonların konumlarını optimize ederek enerji tüketimini minimize etmeyi hedeflenmiştir.

Meta-sezgisel algoritmaların temel avantajlarından biri belirli bir probleme özgü olmamaları ve farklı türde optimizasyon problemlerine kolayca uyarlanabilmeleridir. Bu esneklik, işletme, mühendislik, ekonomi ve bilim gibi alanlarda geniş bir kullanım alanı bulmalarını sağlar.

4.1 *Yapay Arı Koloni Algoritması (ABC)*

ABC algoritması, bal arılarının yiyecek arama davranışlarından esinlenerek geliştirilmiştir. Bu algoritma, çok boyutlu ve çok modlu optimizasyon problemlerini çözmek için kullanılır. İşçi arılar, potansiyel çözüm alanları olan yiyecek kaynaklarını keşfederler ve bu kaynakların kalitesini getirirler. Gözcü arılar, işçi arıların getirdiği kaynakları değerlendirir ve en iyi kaynakları seçmek için olasılık hesaplamaları yaparlar. Algoritmanın işlem adımları aşağıda özetlenmiştir:

1. Başlangıç: Başlangıç yiyecek kaynakları alanları rastgele oluşturulur.
2. İşçi Arı Aşaması: İşçi arılar, yiyecek kaynaklarına giderek bilgileri getirirler ve yeni kaynaklar oluştururlar.
3. Gözcü Arı Aşaması: Gözcü arılar, işçi arıların getirdiği kaynakların kalitesini değerlendirir ve seçim olasılıklarını hesaplarlar.
4. Yiyecek Kaynağı Seçimi: En iyi yiyecek kaynağı seçilir ve hafızada tutulur.
5. Limit ve Yeniden Başlama: Belirlenen limit kontrol edilene kadar işlem adımları tekrarlanır.

ABC algoritması, optimizasyon problemlerinde iyi performans sergileyen bir meta-sezgisel algoritmadır.

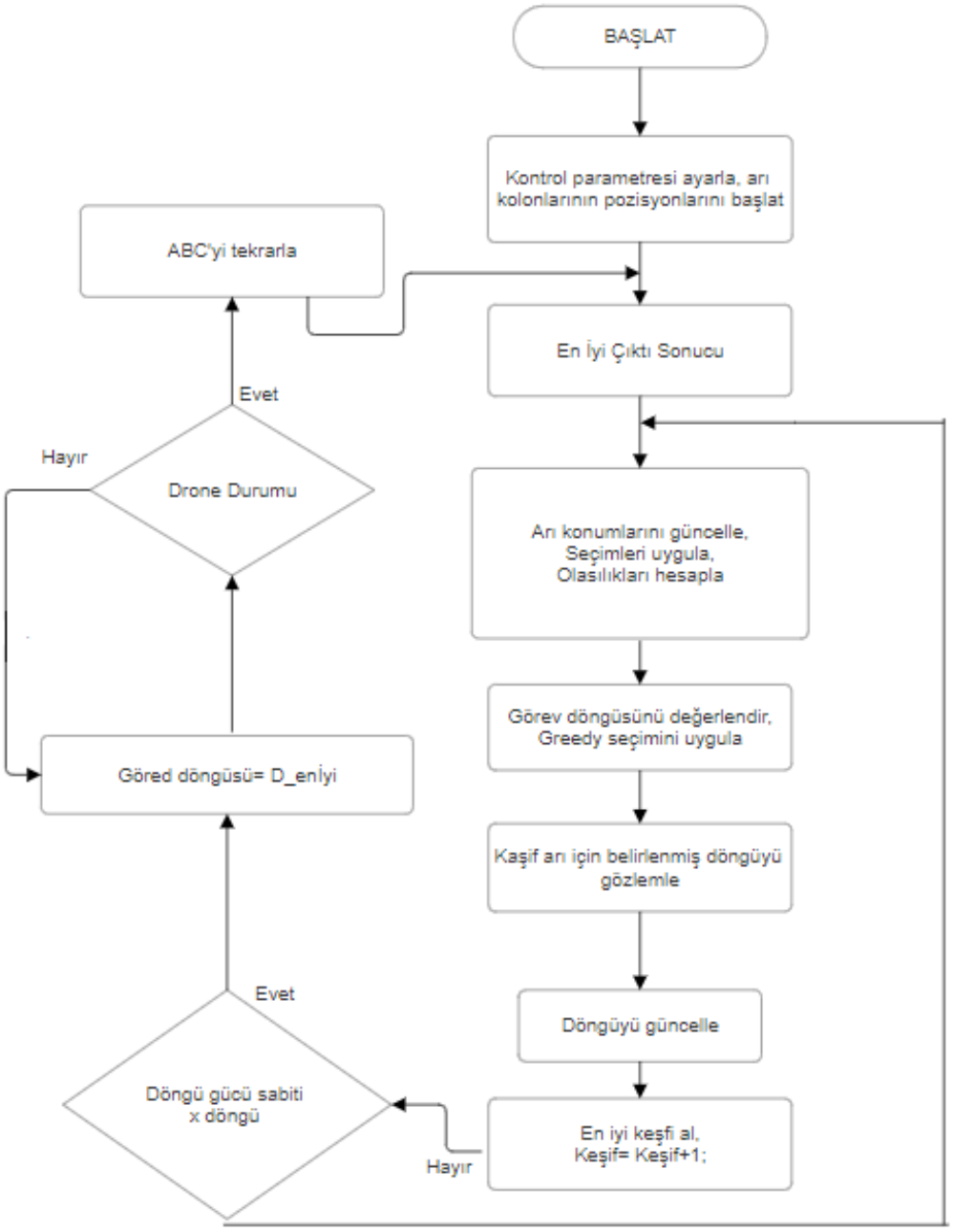

Figure 5: ABC Diyagramı

Algorithm 1: Arı Kolonisi Optimizasyonu

1 Başlangıçta, rastgele çözümler oluşturulur ve bunların fitness değerleri hesaplanır. **while** *sonlandırma kriteri sağlanana kadar* **do**

2 **end**

3 **İşçi Arı Aşaması: for** *her işçi arı için* **do**

4 **end**

5 Komşulukta yeni bir çözüm keşfet. **if** *yeni çözümün fitness değeri daha iyi ise* **then**

6 **end**

7 Çözümü güncelle ve deneme sayacını sıfırla. **else**

8 **end**

9 Deneme sayacını artır.

10 **Gözcü Arı Aşaması: for** *her gözcü arı için* **do**

11 **end**

12 Fitness değerine göre bir çözüm seç. Komşulukta yeni bir çözüm keşfet. **if** *yeni çözümün fitness değeri daha iyi ise* **then**

13 **end**

14 Çözümü güncelle ve deneme sayacını sıfırla. **else**

15 **end**

16 Deneme sayacını artır.

17 **Keşif Arı Aşaması: for** *her işçi arı için* **do**

18 **end**

19 deneme sayacı limiti aşılırsa Mevcut çözümü terk et ve yeni bir rastgele çözüm oluştur. Deneme sayacını sıfırla.

20 Şu ana kadar bulunan en iyi çözümü güncelle. **return** En iyi çözümü ve onun fitness değerini.

Algoritma 1, Arı Kolonisi Optimizasyonu (ABC) algoritmasını uygulayan bir sınıf içermektedir. ABC algoritması, arıların bal arıları gibi davranarak potansiyel çözümleri keşfetmeye ve en iyi çözüme yaklaşmaya çalışan bir doğal optimizasyon algoritmasıdır.

işçi arı sayısı (num-employed-bees), gözlemci arı sayısı (num-onlooker-bees), maksimum iterasyon sayısı (max-iterations), limit (limit), istasyonlar (stations) ve dronelar (drones).

ABC algoritmasının temel aşamalarını içerir:

Her bir işçi arı, rastgele bir çözüm üretir (generate-random-solution) ve bu çözümün uygunluğunu değerlendirir (evaluate-fitness). Sonrasında, çözümü bir miktar değiştirerek (explore-neighbor) yeni bir çözüm üretir ve bu yeni çözümün uygunluğunu tekrar değerlendirir. Eğer yeni çözüm mevcut çözümden daha iyi ise, işçi arı bu çözümü kabul eder.

Gözlemci arılar, işçi arıların ürettiği çözümleri gözlemleyerek en iyi çözümleri seçmeye çalışır. Her bir gözlemci arı, seçilen bir işçi arısının çözümünü alır, bu çözümü değiştirir ve yeni çözümün uygunluğunu değerlendirir. İşçi arıya benzer şekilde, yeni çözüm mevcut çözümden daha iyi ise gözlemci arı bu çözümü kabul eder.

Her bir işçi arı için belirli bir deneme limiti (limit) belirlenir. Eğer bir işçi arı belirlenen limiti aşarsa, rastgele bir çözüm üretilerek bu işçi arının çözümü yenilenir.

Her iterasyonda, işçi ve gözlemci arıların ürettiği çözümler arasından en iyi çözümü (best-solution) ve bu çözümün uygunluğunu (best-fitness) güncelleyen bir mekanizma bulunmaktadır. Her iterasyon sonunda, o iterasyondaki en iyi uygunluğu (best-fitness) best-fitness-per-iteration listesine kaydedilir.

Bu süreçler, ABC algoritmasının temel mantığını oluşturur ve belirli bir problem alanında çözüm aramak için kullanılır. Kodun kullanımı için gerekli olan parametreler ve fonksiyonlar sağlanarak, belirtilen parametrelerdeki maksimum iterasyon sayısına ulaşana kadar en iyi çözüm ve uygunluk değerini döndürür.

4.2 *Parçacık Sürü Optimizasyonu (PSO)*

PSO, parçacıkların sürü davranışını taklit ederek optimal çözümler bulmaya çalışan bir algoritmadır. Parçacıklar, kendi en iyi konumlarına ve global en iyi konuma doğru hareket ederler. PSO'nun temel işlem adımları şu şekildedir:

1. Başlangıç: Parçacıklar rastgele konum ve hızlarla başlatılır.
2. Güncelleme: Parçacıkların konumları ve hızları, en iyi yerel ve global konumlara göre güncellenir.
3. Uygunluk Değerlendirmesi: Parçacıkların yeni konumları ile uygunluk değerleri hesaplanır.
4. Global En İyi Güncelleme: En iyi global konum güncellenir.
5. Iterasyon: Belirlenen iterasyon sayısına veya durma kriterine kadar 2-4 adımları tekrarlanır.

PSO, özellikle çok boyutlu ve sürekli optimizasyon problemlerinde etkilidir.

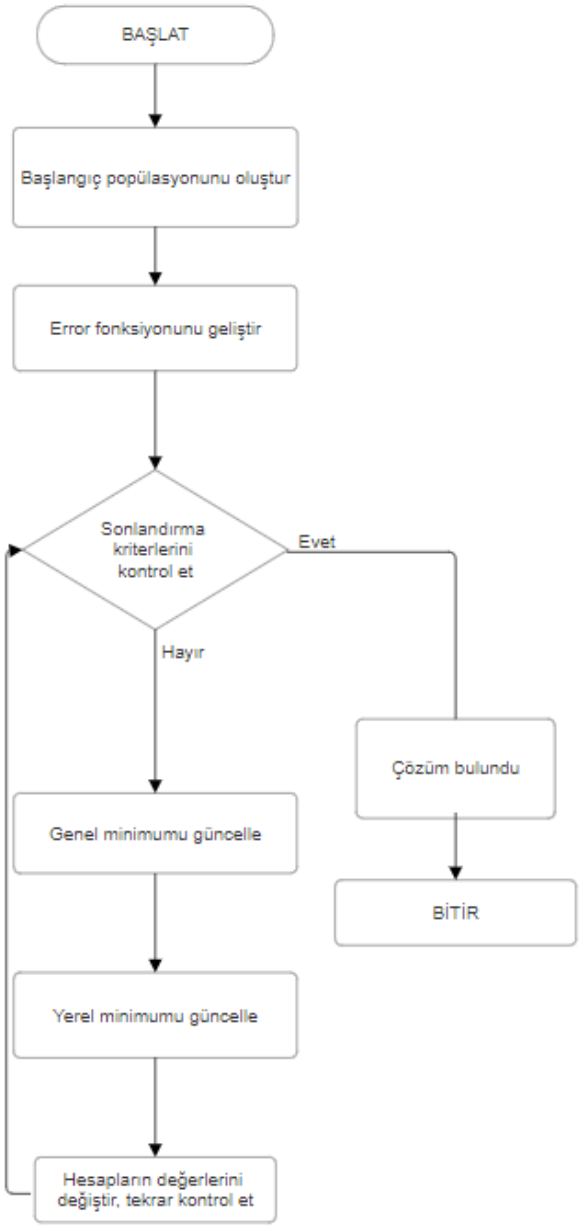

Figure 6: PSO Diyagramı

Algoritma 2, Parçacık Sürüsü Optimizasyonu (PSO) yöntemiyle drone istasyon atama problemini çözmek amacıyla geliştirilmiştir. Bu algoritmada, drone'ların istasyonlara atanması süreci optimize edilmiştir.

PSO sınıfı, num-particles, max-iterations, inertia-weight, cognitive-weight, ve social-weight parametreleri ile başlatılmaktadır. optimize fonksiyonu, drone'ların istasyonlara atanmasını

Algorithm 2: Parçacık Sürü Optimizasyonu

Girdi: Dronelar, İstasyonlar, İterasyon Sayısı **Çıktı:** En iyi drone-istasyon atamaları Başlat: Parçacıkları rastgele konum ve hızlarla Global en iyi konum ve uygunluk değerini başlat (gBest) Her parçacığın en iyi konumunu ve uygunluk değerini başlat (pBest)

for *iterasyon = 1 to iterasyon sayısı* **do**

end

parçacık p Hesapla: p'nin uygunluk değeri **if** *p'nin uygunluk değeri pBest'ten daha iyi ise* **then**

end

Güncelle: pBest = p'nin mevcut konumu **if** *p'nin uygunluk değeri gBest'ten daha iyi ise* **then**

end

Güncelle: gBest = p'nin mevcut konumu

Başlat: Atanmış istasyonlar kümesi (assignedStations) Başlat: Atanmış dronlar kümesi (assignedDrones)

foreach *drone d ve istasyon s çifti* **do**

end

Hesapla: Atalet, bilişsel ve sosyal terimler Güncelle: Yeni istasyon indeksi

if *newstationindex atanmamış ise* **then**

end

Güncelle: Parçacığın konumu Ekle: newstationindex'i assignedStations kümesine Ekle: drone'u assignedDrones kümesine **else**

end

Seç: Rastgele bir istasyon Güncelle: Parçacığın konumu Ekle: newstationindex'i assignedStations kümesine

foreach *atanmamış drone d* **do**

end

Uygula: Ceza işlemi (isteğe bağlı)

Hesapla: Ortalama uygunluk değeri Sakla: averagefitness değerini fitnessvalues listesine

optimize etmek için kullanılmaktadır. Başlatma zamanı kaydedilmiştir ve en iyi global pozisyon ile uygunluk değeri tutulmaktadır.

Parçacıklar, istasyonların sayısı ve drone'ların maksimum batarya seviyeleri arasında rastgele konum ve hızlarla başlatılmıştır. particles listesi, her bir parçacığın konum bilgilerini tutmaktadır.

Algoritma, belirtilen maksimum iterasyon sayısı kadar çalıştırılmıştır. Her iterasyonda, her bir parçacık için uygunluk değeri hesaplanmış ve en iyi global pozisyon güncellenmiştir. iteration-fitness-values listesi, her iterasyondaki uygunluk değerlerini saklamak için kullanılmıştır. Parçacıkların konumları ve hızları, atama kuralları ve ağırlıklar göz önünde bulundurularak güncellenmiştir.

evaluate-fitness fonksiyonu, bir parçacığın mevcut konumuna göre uygunluk değerini hesaplamak için kullanılmıştır. Uygunluk değeri, drone ile istasyon arasındaki mesafe ve drone'un batarya seviyesi göz önünde bulundurularak hesaplanmıştır. Mesafe ne kadar kısa ve batarya seviyesi ne kadar yüksekse uygunluk değeri o kadar düşük olmaktadır. Eğer bir drone, hiçbir istasyona atanmazsa uygunluk değerine ceza eklenmiştir.

İterasyonlar tamamlandığında, en iyi global pozisyon ve uygunluk değeri döndürülmüştür. fitness-values listesi, her iterasyondaki ortalama uygunluk değerlerini saklamakta ve algoritmanın performansını değerlendirmek için kullanılmaktadır. Algoritmanın çalışma süresi hesaplanmış ve döndürülmüştür.

Bu algoritmada, PSO'nun temel prensipleri uygulanarak drone'ların istasyonlara optimal şekilde atanması sağlanmıştır. Başlangıçta rastgele konumlandırılan parçacıklar, her iterasyonda konum ve hızlarını güncelleyerek daha iyi uygunluk değerlerine ulaşmaya çalışmaktadır. Sonuç olarak, en iyi drone-istasyon atamaları belirlenmiş ve algoritmanın performansı uygunluk değerleri ve çalışma süresi ile değerlendirilmiştir.

4.3 *Genetik Algoritma (GA)*

GA, doğal seleksiyon ve genetik operatörler (seçim, çaprazlama, mutasyon) kullanarak popülasyonu evrimleştirerek optimal çözümler bulur. Seçim operatörü, en düşük enerji tüketimi sağlayan bireyleri seçerken, çaprazlama ve mutasyon operatörleri yeni konumlar oluşturur ve popülasyonu geliştirir. GA'nın işlem adımları şu şekildedir:

1. Başlangıç: Rastgele popülasyon oluşturulur.
2. Seçim: Ebeveynler seçilerek çaprazlama ve mutasyon için hazırlanır.
3. Çaprazlama ve Mutasyon: Ebeveynler çaprazlanır ve mutasyona uğrar.
4. Yeni Popülasyon Oluşturma: Yeni bireyler oluşturulur ve uygunluk değerleri hesaplanır.
5. Elitizm ve Seçim: En iyi bireyler yeni popülasyona aktarılır ve yeni iterasyon hazırlanır.
6. Iterasyon: Belirlenen iterasyon sayısına veya durma kriterine kadar 2-5 adımları tekrarlanır.

GA, genellikle kombinatoryal optimizasyon problemleri için tercih edilir.

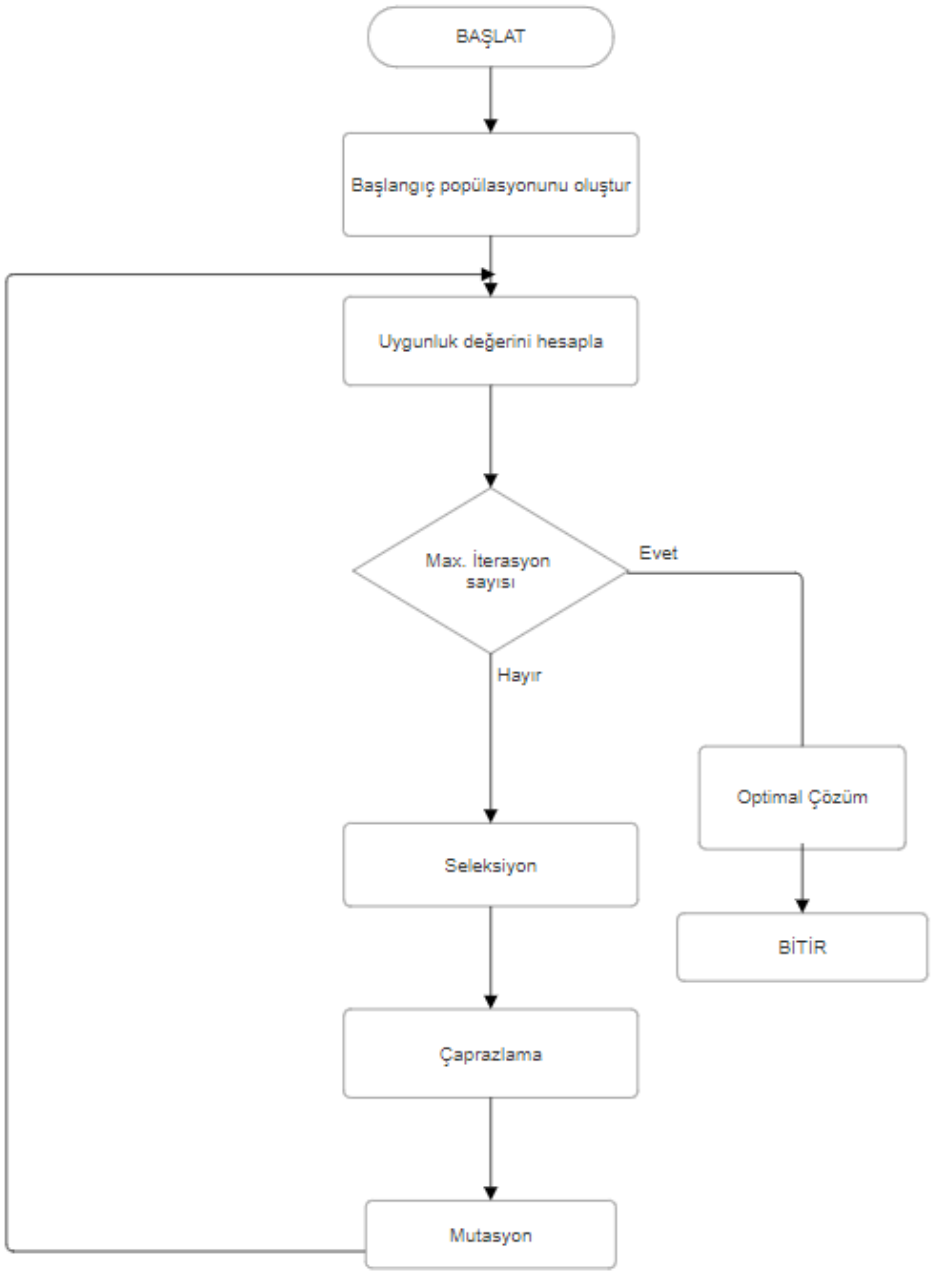

Figure 7: Genetik Diyagramı

Algorithm 3: Genetik Algoritma

1 [1] Dronlar, İstasyonlar, Max Generasyon Sayısı En iyi çözüm ve uygunluk değeri

2 Rastgele popülasyonu başlat Başlangıç popülasyonunun uygunluk değerlerini hesapla

3 **for** *generasyon = 1 to maxgenerasyon* **do**

4 **end**

5 Popülasyondaki her bireyin uygunluk değerini hesapla

6 Fitness'e göre ebeveyn seçimi yap (rulet tekerleği seçimi)

7 Yeni popülasyonu başlat **while** *yeni popülasyon doluncaya kadar* **do**

8 **end**

9 İki ebeveynden rastgele seçim yap Çaprazlama işlemi yap (olasılıkla $aprazlamaoran$ß) Mutasyon işlemi yap (olasılıkla $mutasyonoran$ß) Oluşan çocukları yeni popülasyona ekle

10 Eski popülasyonu yeni popülasyonla değiştir

11 Mevcut popülasyondaki en iyi çözümü bul

12 En iyi uygunluk değerini geçmişi sakla

13 **return** En iyi çözüm, en iyi uygunluk değeri, uygunluk geçmişi

Genetik algoritma (GA), doğal seçilim ve genetik operatörler kullanarak problemleri optimize etmek için güçlü bir tekniktir. Algoritma 3'te GA, drone yönlendirme problemini çözmek için uygulanmıştır.

GA, belirli parametrelerle başlatılır: popülasyon büyüklüğü, mutasyon oranı, çaprazlama oranı ve maksimum nesil sayısı. Her bir drone için başlangıçta rastgele bir istasyon atanır ve batarya seviyesi belirlenir.

İlk adımda, initialize-population fonksiyonu kullanılarak rastgele bir başlangıç popülasyonu oluşturulur. Her bir bireyde, droneların sayısı kadar bir istasyon endeksi ve bir batarya seviyesi bulunur.

Her bireyin uygunluğu, evaluate-fitness fonksiyonu ile değerlendirilir. Her drone için atanmış istasyona olan uzaklık, batarya seviyesi ve hız dikkate alınarak bir fitness değeri hesaplanır. Atanmamış bir drone için ceza verilir.

select-parents fonksiyonu, bireylerin uygunluklarına göre seçim yaparak ebeveynleri belirler. Ardından crossover fonksiyonu ile çaprazlama işlemi gerçekleştirilir. Ebeveynlerden yeni çocuklar oluşturmak için genler belirli bir noktada kesilir ve değiştirilir.

Oluşturulan çocuklar, mutate fonksiyonu ile mutasyona uğrar. Belirli bir mutasyon oranına göre, drone'ların atanmış istasyonları ve batarya seviyeleri rastgele değiştirilir.

Her iterasyonda, yeni bir popülasyon oluşturulur. En iyi bireyler seçilerek bir sonraki nesil için kullanılır. Bu adım, popülasyonun devamlılığını ve çeşitliliğini sağlar.

Her nesilde, en iyi çözüm min fonksiyonu kullanılarak belirlenir. En iyi çözüm, en düşük fitness değerine sahip olan birey olarak seçilir ve kaydedilir.

Belirlenen maksimum-nesil sayısına ulaşılana kadar bu adımlar iteratif olarak tekrarlanır. Her iterasyonun sonunda en iyi bireyin fitness değeri fitness-history listesine eklenir.

Bu genetik algoritma örneği, drone yönlendirme problemini optimize etmek için etkili bir yöntem sağlar. Parametreler ve genetik operatörler, problem alanına ve gereksinimlere uygun olarak ayarlanabilir ve uygulanabilir. Genetik algoritma, karmaşık optimizasyon problemlerinde kullanılabilecek geniş bir optimizasyon tekniklerinden biridir.

4.4 *Gri Kurt Optimizasyonu (GWO)*

Gri kurt optimizasyonu, lider ve takipçi kurtların avlarını çevreleme ve yakalama stratejilerini taklit ederek drone'ların optimal konumlarını belirlemek için uygulandı. Bu algoritmada drone'lar, alfa, beta ve delta kurtları olarak sınıflandırıldı. Alfa drone, en iyi konumu temsil ederken beta ve delta drone'lar ikinci ve üçüncü en iyi konumları temsil etti. Diğer drone'lar (omega) ise bu lider drone'ların konumlarına göre kendilerini güncelleyerek en iyi konumu bulmaya çalıştı. Algoritmanın adımları şu şekildedir:

1. Başlangıç: Gri kurtlar rastgele konumlarla başlatılır.
2. Lider Kurtların Güncellenmesi: Alfa, beta ve delta lider kurtların konumları güncellenir.
3. Takipçi Kurtların Güncellenmesi: Diğer kurtların konumları lider kurtlara göre güncellenir.
4. Uygunluk Değerlendirmesi: Her kurt için uygunluk değerleri hesaplanır.
5. En İyi Kurt Seçimi: En iyi kurtun konumu ve uygunluk değeri hafızada tutulur.
6. Iterasyon: Belirlenen iterasyon sayısına veya durma kriterine kadar 2-5 adımları tekrarlanır.

GWO, özellikle sürekli optimizasyon problemlerinde başarılı olmuştur.

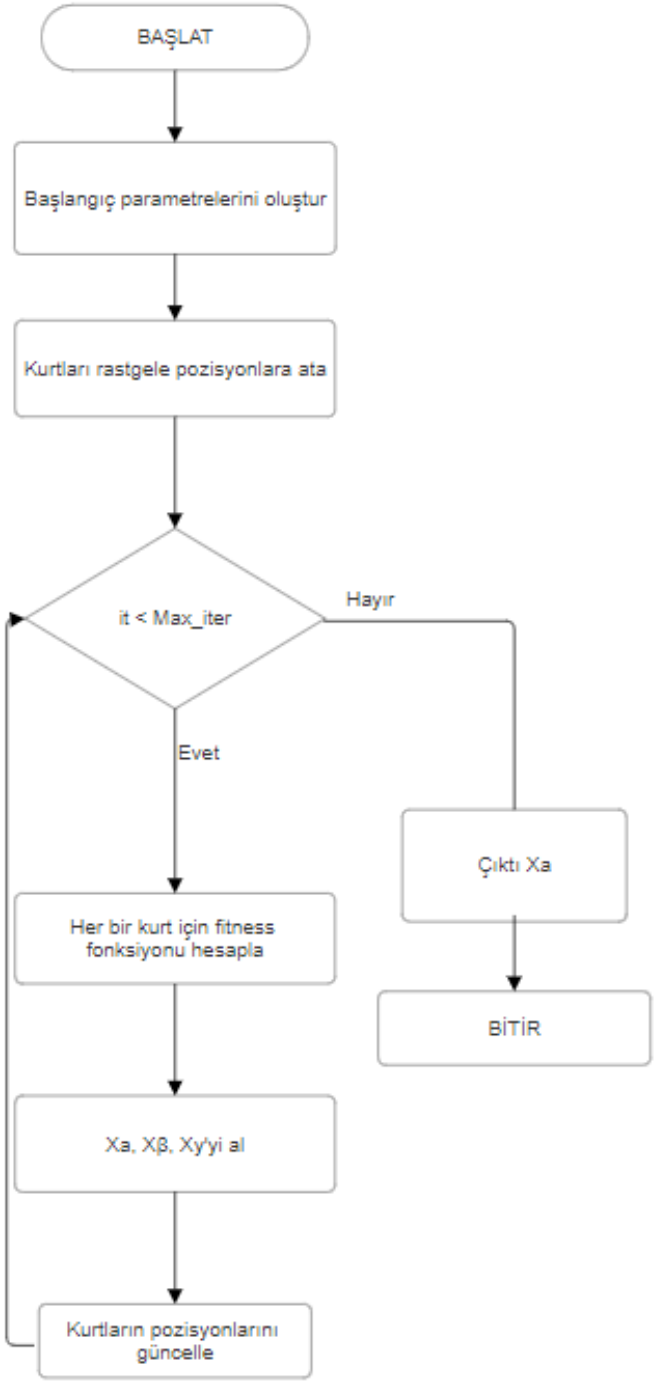

Figure 8: GWO Diyagramı

Algoritma 4, doğal gri kurt sürüsünün sosyal davranışlarını taklit ederek optimizasyon problemlerini çözmeye çalışır. Kodun ana işlevi, verilen istasyon ve drone yapılandırmalarıyla en iyi çözümü bulmaktır.

Algorithm 4: Gri Kurt Optimizasyonu

Input: Kurt sayısı numwolves, Maksimum iterasyon sayısı maxiterations, Parametreler: a, A, C, b, l, İstasyonlar, Drone'lar

Output: En iyi global konum, En iyi global uygunluk

```
kurtları rastgele başlat
eniyiglobalkonum ← None, eniyiglobaluygunluk ← ∞
for iterasyon ← 1 to maxiterations do
    foreach kurt in kurtlar do
        kurtun uygunluğunu değerlendir
        if uygunluk < eniyiglobaluygunluk then
            eniyiglobaluygunluk ← uygunluk, eniyiglobalkonum ← kurt
        end
        alfa, beta, delta pozisyonlarını kurtlardan al
        foreach drone i kurtta do
            if istasyonindex veya drone i zaten atanmışsa then
                istasyon veya drone i boşta işaretle
            end
            else
                drone i için yeni pozisyon ve batarya hesapla
                sınırları kontrol et
                if yeni istasyonindex veya drone i zaten atanmışsa then
                    istasyon veya drone i boşta işaretle
                end
                else
                    kurtu güncelle, istasyon ve drone i ata
                end
            end
        end
    end
end
```

GreyWolfOptimizer sınıfı, gri kurt optimizasyonunu gerçekleştiren ana sınıftır. num-wolves: Sürüdeki kurt sayısı. max-iterations: Optimizasyonun maksimum iterasyon sayısı. a, A, C, b, l: Algoritmanın parametreleri. Bu değerler, kurtların sosyal hiyerarşisini ve davranışlarını kontrol eder.

optimize metodu, gri kurt optimizasyon algoritmasını çalıştıran ana fonksiyondur. wolves: Başlangıçta rastgele oluşturulan kurtların pozisyonları ve batarya seviyeleri. iteration döngüsü, maksimum iterasyon sayısına kadar devam eder. Her iterasyonda, her bir kurt için fitness değeri hesaplanır ve en iyi global çözüm güncellenir. get-alpha-beta-delta metoduyla alfa, beta ve delta kurtları belirlenir. Her kurt için yeni pozisyon ve batarya seviyeleri hesaplanır. Yeni pozisyonlar, kurtların sosyal hiyerarşisine ve etkileşimine göre güncellenir. evaluate-fitness metodu, bir kurt sürüsünün verilen bir çözümünün fitness değerini hesaplar. Her kurtun atanmış istasyon ve drone sayısını kontrol eder. Birden fazla kurt aynı istasyona veya aynı anda çalışıyorsa ceza uygulanır. Uzaklık ve batarya seviyelerine dayalı bir fitness hesaplaması yapılır. Her iterasyon sonunda en iyi global pozisyon ve fitness değeri döndürülür.

Bu kod, gri kurt optimizasyonunun temel prensiplerini içerir ve verilen bir optimizasyon problemi için çok sayıda kurt kullanarak en iyi çözümü bulmaya çalışır. iterate metodu ise başlangıç çözümüyle başlamak için kullanılabilir.

4.5 *Karınca Kolonisi Optimizasyonu (ACO)*

Karınca kolonisi optimizasyonu, drone'ların en verimli rotalarını ve istasyon konumlarını belirlemek için feromon izlerini kullanarak gerçekleştirildi. Bu algoritmada her karınca, bir drone'un konumunu temsil eder ve feromon izleri, daha iyi enerji tüketimi sağlayan yolları işaret eder. İterasyonlar boyunca feromon izleri güncellenir ve drone'ların enerji tüketimi optimize edilir. ACO'nun adımları şu şekildedir:

1. Başlangıç: Karınca popülasyonu ve feromon izleri başlatılır.
2. Yol Seçimi: Karıncalar, feromon izleri ve yol uzunluklarına göre yollar seçerler.
3. Yol Güncellemesi: Her karınca yol üzerinde feromon bırakır veya var olan feromonu günceller.
4. Global En İyi Güncelleme: En iyi yol ve feromon güncellenir.
5. Iterasyon: Belirlenen iterasyon sayısına veya durma kriterine kadar 2-4 adımları tekrarlanır.

ACO, özellikle kombinatoryal optimizasyon problemlerinde kullanışlıdır.

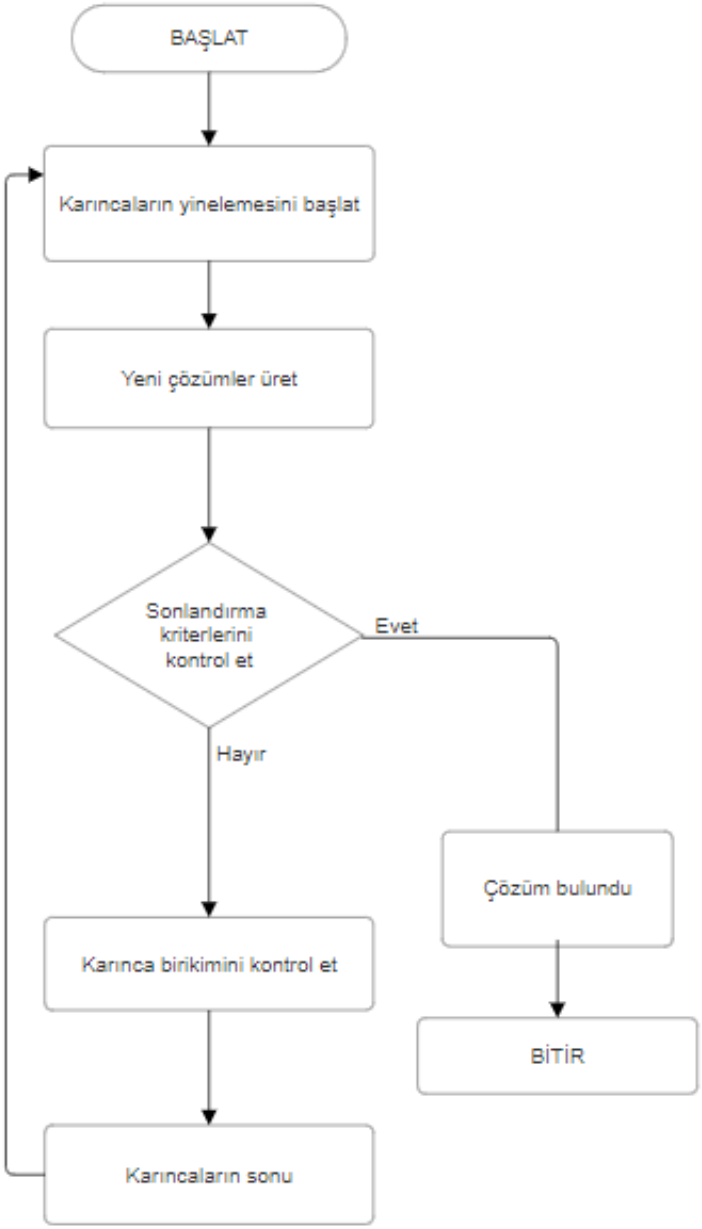

Figure 9: ACO Diyagramı

Algoritma 5, Karınca Kolonisi Optimizasyonu (Ant Colony Optimization, ACO) algoritmasını temsil eden bir sınıf içermektedir. ACO, karıncaların yiyecek kaynaklarını bulma davranışından esinlenerek, olası çözümleri keşfetmek ve iyileştirmek için kullanılan bir meta-sezgisel optimizasyon tekniğidir.

Algorithm 5: Karınca Kolonisi Optimizasyonu

[1] Parametreleri başlat:

numants, *numiterations*, *evaporationrate*, α, β, *pheromonedeposit*, *initialpheromone* Başlangıç feromon matrisi **T** ile başlangıç değerlerini ayarlayın Karıncalar için başlangıç çözümlerini oluşturun *bestsolution* ve *bestfitness* değerlerini başlangıç çözümünün değerlerine ayarlayın Uygunluk geçmişi listesi *fitnesshistory* ve ortalama uygunluk geçmişi listesi *averagefitnesshistory* başlatın

for $t = 1$ *ila numiterations* **do**

end

Tüm karıncalar için çözümleri generate edin using constructsolutions() Feromon matrisini **T** güncelleyin using updatepheromones() En iyi çözümü *iterationbestsolution* ve bu çözümün uygunluğunu *iterationbestfitness* bulun *fitnesshistory* ve *averagefitnesshistory* güncelleyin

if *iterationbestfitness* $<$ *bestfitness* **then**

end

bestfitness ve *bestsolution* güncelleyin **dönüş** *bestsolution*, *bestfitness*, *fitnesshistory*, *averagefitnesshistory*

constructsolutionsstations, drones **for** *her bir karınca için* **do**

end

T baz alınarak olasılıkları kullanarak çözüm oluşturun Çözüm listesini dönün

updatepheromonessolutions **T** içindeki feromonları buharlaştırın Karıncalar tarafından bulunan çözümlere göre feromonları yatırın

calculateprobabilitiesdrone, stations, availablestations Sonraki istasyonu seçmek için olasılıkları hesaplayın

calculatefitnesssolution, stations Toplam mesafeye dayalı bir çözümün uygunluğunu hesaplayın

Sınıf, ACO algoritmasını uygulayan AntColonyOptimization adında bir yapı taşır. Başlangıçta algoritmanın çalışması için gerekli olan parametrelerle (karınca sayısı, iterasyon sayısı, feromon buharlaşma oranı, alpha ve beta gibi etkileşim parametreleri, feromon depozito miktarı, başlangıç feromon seviyesi) başlatılır.

Algoritmanın işleyişi optimize metodu üzerinden gerçekleşir. Her iterasyonda, çözümler oluşturulur (construct-solutions metodu), feromon seviyeleri güncellenir (update-pheromones metodu), en iyi çözüm bulunur ve uygunluk değerleri kaydedilir. Oluşturulan çözümler arasından en iyi çözümü seçmek için uygunluk fonksiyonu (calculate-fitness) kullanılır. Ayrıca her iterasyonda ortalama uygunluk değeri de hesaplanarak geçmiş performans izlenir.

Çözümler oluşturulurken, her bir karıncanın istasyonları ziyaret sırasında karar verme yeteneği (olasılıkları hesaplamak ve istasyon seçmek için calculate-probabilities ve select-station metodları) simüle edilir. Bu seçimler, feromon seviyeleri ve istasyonlar arası uzaklıkların etkileşimi ile belirlenir. Feromon seviyeleri ise iyi çözümleri hatırlamak ve olası çözüm alanlarını keşfetmek için güncellenir.

Sonuç olarak, ACO algoritması bu şekilde uygulanarak, verilen istasyon ve drone konumları için en kısa yol problemini çözmeye çalışır. Algoritma, karıncaların toplu zeka ve iz bırakma mekanizmalarını taklit ederek, potansiyel çözüm alanlarını araştırır ve iyileştirir

4.6 *Çekirge Optimizasyon Algoritması (GOA)*

Çekirge optimizasyon algoritması, drone'ların en verimli rotalarını ve istasyon konumlarını belirlemek için çekirgelerin sürü davranışlarını taklit eder. Bu algoritmada her çekirge, bir drone'un konumunu temsil eder ve komşularıyla etkileşerek daha iyi enerji tüketimi sağlayan yolları bulmaya çalışır. İterasyonlar boyunca çekirgelerin konumları güncellenir ve drone'ların enerji tüketimi optimize edilir. GOA'nın adımları şu şekildedir:

1. Başlangıç: Çekirge popülasyonu rastgele konumlarla başlatılır.
2. Konum Güncellemesi: Çekirgeler, komşularıyla etkileşerek konumlarını güncellerler.
3. Uygunluk Değerlendirmesi: Her çekirgenin yeni konumu ile uygunluk değeri hesaplanır.
4. En İyi Çekirge Seçimi: En iyi çekirgenin konumu ve uygunluk değeri hafızada tutulur.
5. Iterasyon: Belirlenen iterasyon sayısına veya durma kriterine kadar 2-4 adımları tekrarlanır.

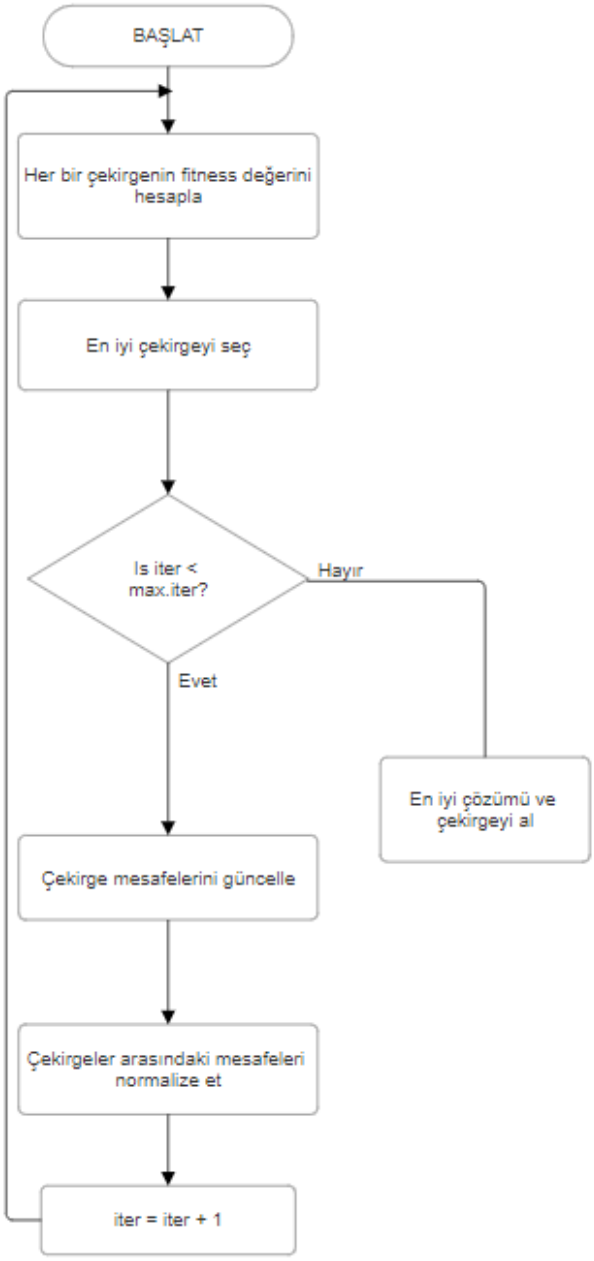

Figure 10: GOA Diyagramı

Algoritma 6, bir optimizasyon algoritması olan Çekirge Optimizasyon Algoritması (GOA) üzerine kurulu bir simülasyonu içerir. Drone ve Station sınıfları, her bir drone'ın konumunu, maksimum batarya seviyesini ve maksimum hızını tanımlar. GOA sınıfı ise algoritmanın parametrelerini ve optimizasyon işlemini yönetir.

Algorithm 6: Çekirge Optimizasyon Algoritması

Input: Popülasyon boyutu populationsize, Maksimum iterasyon sayısı maxiterations, Drone'lar, İstasyonlar

Output: En iyi global konum, En iyi global uygunluk

```
Başlangıç:
çekirge popülasyonunu rastgele başlat
eniyiglobalkonum ← None, eniyiglobaluygunluk ← ∞
for iterasyon ← 1 to maxiterations do
    foreach çekirge do
        çekirgenin uygunluğunu değerlendir
        if uygunluk < eniyiglobaluygunluk then
            eniyiglobaluygunluk ← uygunluk, eniyiglobalkonum ← çekirge
        end
        Alfa, beta, delta pozisyonlarını çekirgelerden al
        foreach drone i için çekirgede do
            if istasyonindex veya drone i zaten atanmışsa then
                istasyon veya drone i boşta işaretle
            end
            else
                drone i için yeni pozisyon ve batarya hesapla
                sınırları kontrol et
                if yeni istasyonindex veya drone i zaten atanmışsa then
                    istasyon veya drone i boşta işaretle
                end
                else
                    Çekirge güncelle, istasyon ve drone i ata
                end
            end
        end
    end
end
```

GOA sınıfının optimize metodu verilen drone ve istasyon listeleri üzerinde optimizasyon işlemini gerçekleştirir. İterasyon sayısı max-iterations ve populasyon büyüklüğü population-size olarak tanımlanmıştır.

Her iterasyonda population-size kadar rastgele çekirge oluşturulur. Her bir çekirge, her bir drone için bir istasyon seçimi (station-index) ve bir batarya seviyesi (battery-level) içerir.

Her bir çekirge için evaluate-fitness metodu çağrılır. Bu metod, çekirgenin mevcut durumuyla her bir drone için fitness değerini hesaplar. Fitness değeri, dronenin seçilen istasyona olan uzaklığı ve mevcut batarya seviyesine göre belirlenir. Ulaşılabilir olmayan istasyonlar için yüksek bir fitness değeri döndürülür.

Her iterasyonda, en iyi global fitness değeri güncellenir ve en iyi çekirge pozisyonu kaydedilir.

Her bir çekirge için yeni pozisyon hesaplanırken, üç terim kullanılır: inertia-term (rastgele bir değer), cognitive-term (en iyi global pozisyon ile çekirge arasındaki farkın rastgele bir kısmı) ve social-term (en iyi global pozisyon ile çekirge arasındaki farkın rastgele bir kısmı). Bu terimler, çekirgenin mevcut pozisyonunu güncellemek için kullanılır.

Her iterasyonun sonunda, en iyi fitness değeri, iterasyon boyunca elde edilen fitness değerleri ve ortalama fitness değerleri (fitness-history, all-fitness-values, avg-fitness-history) kaydedilir.

Iterasyonlar tamamlandığında, en iyi global pozisyon, en iyi fitness değeri, fitness geçmişi ve atanamayan dronların listesi (unassigned-drones) döndürülür.

Bu şekilde, GOA algoritması kullanılarak dronların optimal olarak istasyonlara atanması ve enerji seviyelerinin optimize edilmesi sağlanır. Algoritmanın her iterasyonunda çekirgeler rastgele hareket ederek, en iyi çözümü bulmaya çalışırlar.

4.7 *Diferansiyel Gelişim Algoritması (DEA)*

Diferansiyel gelişim algoritması (DGA), doğal bir seçilim ve çaprazlama mekanizması kullanarak optimize edilmesi gereken bir hedef fonksiyonunu eniyilemeye çalışan bir evrimsel hesaplama tekniğidir. Bu algoritma, bireyler arasındaki farklılıkları vurgulayan ve bu farklılıkları yeni çözüm adayları oluşturmak için kullanarak genetik çeşitliliği teşvik eder. Başlangıçta rastgele seçilen bir popülasyon üzerinde işlem yaparak, bireylerin birbirlerinden farklılıklarını temsil eden vektörler arasında çaprazlama ve mutasyon operatörleri uygulanır. Bu süreç, neslin her bir üyesinin potansiyel olarak daha iyi çözümler üretmesini sağlar ve zamanla hedef fonksiyonu optimize eden bir sonuç elde edilir. DEA algoritmasının adımları şu şekildedir:

1. Başlatma: İlk popülasyon rastgele drone-istasyon atamalarıyla başlatılır.

2. İterasyonlar: Belirlenen iterasyon sayısına kadar iterasyonlar yapılır.

3. Mutasyon ve Çaprazlama: Her birey için mutasyon operatörü uygulanarak yeni bir çözüm adayı oluşturulur. Daha sonra çaprazlama ile bir sonraki nesil için yeni bir birey oluşturulur.

4. Uygunluk Değerinin Hesaplanması: Her yeni birey için uygunluk değeri, drone-istasyon atama probleminin kriterlerine göre hesaplanır.

5. Popülasyon Güncelleme: Elitizm veya seçim yöntemleriyle yeni bireylerin popülasyona dahil edilmesi sağlanır. En iyi çözümler korunur ve kötü çözümler eğer gerekiyorsa elenir.

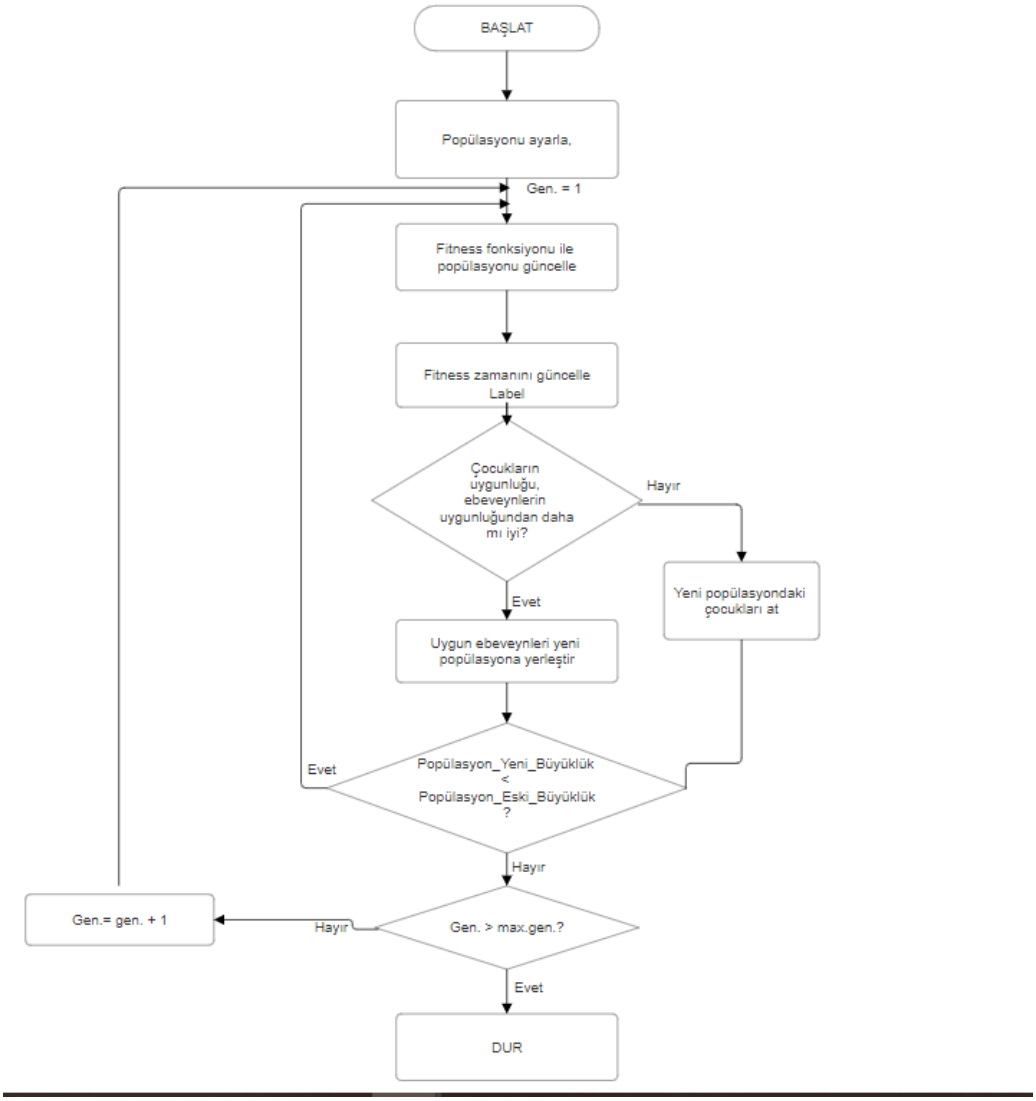

Figure 11: DEA Diyagramı

Algoritma 7, diferansiyel evrim algoritmasını kullanarak drone'ların belirli istasyonlara en uygun şekilde atanmasını sağlamak amacıyla yazılmıştır. DEA sınıfı, drone ve istasyonların özelliklerine göre en uygun atamayı bulmaya çalışır.

Öncelikle, DEA sınıfı, parçacık sayısı, maksimum iterasyon sayısı, ölçeklendirme faktörü ve çaprazlama oranı gibi parametrelerle başlatılır. Bu parametreler, algoritmanın nasıl çalışacağını belirler. optimize metodu, algoritmanın ana döngüsünü içerir ve drone'ların istasyonlara atanmasını optimize eder. Bu metodun başında, başlangıç zamanı kaydedilir ve en iyi küresel konum ile en iyi küresel fitness değeri tanımlanır. Rastgele parçacıklar

Algorithm 7: Diferansiyel Evrim Algoritması (DEA)

1 [1] Parametreleri başlat: *numparticles*, *maxiterations*, *scalingfactor*, *crossoverrate* Başlangıç partiküllerini oluştur *bestglobalposition* ve *bestglobalfitness* değerlerini başlangıç konumlarına ayarla Uygunluk geçmişi listesi *fitnessvalues* başlat Iterasyon = 0 olarak başlat

2 **while** *Iterasyon* $<$ *maxiterations* **do**

3 **end**

4 her bir partikül *i trialvector* oluştur Tekil atanma veya atanmamış durumu sağla Çaprazlama yap Yeni partikülün uygunluğunu değerlendir **if** *Yeni partikülün uygunluğu daha iyi ise* **then**

5 **end**

6 Partikülü güncelle **if** *Yeni uygunluk en iyi global uygunluktan daha iyi ise* **then**

7 **end**

8 En iyi global uygunluğu ve konumu güncelle Ortalama uygunluğu hesapla ve *fitnessvalues* listesine ekle Iterasyonu bir artır

9 Bitiş zamanını hesapla Dönen değerler: *bestglobalposition*, *bestglobalfitness*, *fitnessvalues*, *elapsedtime*

10 generatetrialvectorpartiküller, geçerliindex, olçekfaktörü, istasyonlar, dronlar İndeksleri karıştır a, b, c seç Deneme vektörü oluştur Geçerli istasyon indeksi veya atanmamış durumu sağla Pil seviyesini sınırla Deneme vektörünü döndür

11 evaluatefitnesspartikül, dronlar, istasyonlar Toplam uygunluğu hesapla Döndür: toplam uygunluk

oluşturulur ve her bir parçacık, drone'ların rastgele istasyonlara atanmasını ve batarya seviyelerini içerir.

İterasyon döngüsünde, her bir parçacık için bir deneme vektörü oluşturulur. Deneme vektöründe, her bir drone'un atanacağı istasyon ve batarya seviyesi belirlenir. Bu vektörlerin geçerliliği sağlanır; örneğin, bir drone birden fazla istasyona atanmaz. Çaprazlama işlemi gerçekleştirilir ve mevcut çözüm ile deneme vektörü karıştırılarak yeni bir çözüm oluşturulur. Yeni çözümün fitness değeri, mevcut çözümle karşılaştırılır ve daha iyi ise mevcut çözüm güncellenir. Ayrıca, en iyi küresel çözüm de güncellenir. Her iterasyonda ortalama fitness değeri hesaplanır ve kaydedilir.

Deneme vektörü oluşturma işlemi, üç rastgele parçacık seçilerek gerçekleştirilir. Her bir drone için yeni bir istasyon indeksi ve batarya seviyesi hesaplanır. Rastgele bir olasılıkla, drone'ların bazıları istasyonsuz (unassigned) olabilir. İstasyon indekslerinin ve batarya seviyelerinin geçerliliği sağlanır.

Fitness değerlendirme işleminde, her bir drone'un atanma durumu kontrol edilir. Drone'un istasyona olan mesafesi ve batarya seviyesi hesaplanarak fitness değeri belirlenir. Drone bir istasyona atanmamışsa veya geçersiz bir istasyona atanmışsa, fitness değeri yüksek tutulur. Toplam fitness değeri hesaplanır.

Sonuç olarak, en iyi küresel konum (drone'ların istasyonlara en uygun atanması) ve buna karşılık gelen fitness değeri döndürülür. Her iterasyon için ortalama fitness değerleri bir liste olarak döndürülür. Algoritmanın çalışma süresi hesaplanır ve döndürülür. Ayrıca, istasyonlara atanamayan drone'ların listesi belirlenir. Bu algoritma, drone'ların en verimli şekilde istasyonlara atanmasını sağlamak için diferansiyel evrim algoritmasını kullanır. Drone'ların istasyonlara olan mesafesi ve batarya seviyeleri gibi faktörleri dikkate alarak en uygun çözümü bulur.

4.8 *Greedy Algoritması*

Algorithm 8: Greedy Algoritma ile Drone İstasyon Atama Problemi

Data: Dronelerin iş/zaman oranları, İstasyon konumu, Görevler

Result: Optimal dronelerin görevlere atanması

foreach *Drone d* **do**
    Hesapla: *d*'nin istasyona olan uzaklığı;
    Hesapla: *d*'nin görevi tamamlama süresi;
    Hesapla: *d*'nin iş/zaman oranı;
end
Sırala: Droneleri iş/zaman oranına göre azalan sıraya koy;
foreach *Görev t* **do**
    Ata: En yüksek iş/zaman oranına sahip droneyi göreve *t*;
    Güncelle: Atanan dronenin iş/zaman oranını;
    Güncelle: Atanan dronenin pil seviyesini;
    Güncelle: Atanan dronenin uçuş süresini;
end
return *Optimal dronelerin görevlere atanması*

Greedy algoritma, her adımda en iyi görünen seçimi yaparak problemi çözer. Bu algoritma, droneları iş/zaman oranına göre sıraya koyarak en uygun göreve atar. Bu projede de greedy algoritma mantığı kullanılmıştır.

4.9 *Senaryo Tanımları*

4.9.1 *Senaryo 1: Sabit Drone, Sabit İstasyon*

Bu senaryoda, drone'ların ve istasyonların konumları sabit tutuldu. Algoritmalar, bu sabit konumlar arasında en düşük enerji tüketimini sağlayan yerleşim düzenini belirlemek için kullanıldı.

4.9.2 *Senaryo 2: Sabit İstasyon, Hareketli Drone*

İkinci senaryoda, istasyonlar sabitken drone'lar belirli bir rota üzerinde hareket etti. Algoritmalar, hareketli drone'ların enerji tüketimini minimize etmek için optimal yolları ve konumları hesaplandı.

4.9.3 *Senaryo 3: Hareketli İstasyon, Hareketli Drone*

Üçüncü senaryoda hem drone'lar hem de istasyonlar hareketli seçildi. Algoritmalar, bu dinamik ortamda en düşük enerji tüketimini sağlayacak yerleşim ve hareket düzenlerini belirledi.

4.10 *Optimizasyon Süreci*

-Her bir algoritma, belirlenen senaryolara uygulanarak enerji tüketimini minimize etmeyi hedefledi. -Algoritmalar, belirli iterasyon sayıları ile çalıştırılarak en iyi sonuçlar elde edildi. -Elde edilen sonuçlar karşılaştırılarak, her bir algoritmanın performansı değerlendirildi ve en düşük enerji tüketimini sağlayan konumlar belirlendi.

4.11 *Performans Değerlendirmesi ve Karşılaştırma*

-Her bir algoritmanın sonuçları yüzlerce ve binlerce iterasyon sonucunda analiz edildi. -Algoritmaların performansları, enerji tüketimi açısından karşılaştırıldı ve en verimli algoritmalar belirlendi.

BÖLÜM: V

DEGERLENDIRME VE BULGULAR

5

DEGERLENDIRME VE BULGULAR

Algoritma	Ortalama	Standart Sapma
ABC	112.66	0.38
ACO	1112.13	2.27e(-13)
DEA	110.26	1.01
GA	38.69	6.10
GOA	22.34	2.84
GWO	1015.34	0.38
PSO	20.95	2.61

Table 1: Senaryo 1 için algoritma en iyi uygunluk değeri ortalamaları ve standart sapma değerlerleri

Algoritma	**Ortalama**	**Standart Sapma**
ABC	125.67	0.40
ACO	1112.13	2.27e(-13)
DEA	121.51	0.61
GA	43.48	5.89
GOA	30.18	3.57
GWO	1017.37	0.18
PSO	27.48	3.10

Table 2: Senaryo 2 için algoritma en iyi uygunluk değeri ortalamaları ve standart sapma değerlerleri

Algoritma	Ortalama	Standart Sapma
ABC	122.56	0.31
ACO	598.51	1.13e(-13)
DEA	119.28	0.94
GA	49.17	4.96
GOA	27.69	1.85
GWO	1018.32	0.51
PSO	25.15	1.59

Table 3: Senaryo 3 için algoritma en iyi uygunluk değeri ortalamaları ve standart sapma değerlerleri

Tablo1, Tablo2 ve Tablo3'te 10000 iterasyon ve 100 kez programın çalıştırılması sonucunda elde edilen değerlerin ortalamaları alınmış ve bunların standart sapmaları hesaplanmıştır.

Tablo 1'de sunulan veriler, sabit istasyon ve sabit drone senaryosunda farklı optimizasyon algoritmalarının performansını göstermektedir. Projenin amacı, en iyi uygunluk değerini minimize etmektir. Bu amaç doğrultusunda, yedi farklı algoritma (ABC, ACO, DEA, GA, GOA, GWO, PSO) çalıştırılmış ve elde edilen ortalama uygunluk değerleri ile standart sapmalar hesaplanmıştır.

Algoritmalar arasında karşılaştırma yapıldığında, PSO algoritmasının en düşük ortalama uygunluk değeri olan 20.95 ile en iyi performansı gösterdiği gözlemlenmiştir. PSO algoritmasının standart sapması ise 2.61 olarak tespit edilmiştir. Bu durum, PSO'nun uygunluk değerinde düşük ortalamaya sahip olduğunu ve sonuçlarının göreceli olarak tutarlı olduğunu göstermektedir.

GOA algoritması, 22.34 ortalama uygunluk değeri ile PSO'dan sonra en iyi performansı sergilemiştir. Bununla birlikte, GOA'nın standart sapması 2.84'tür, bu da sonuçlarının PSO'ya kıyasla biraz daha değişken olduğunu işaret etmektedir.

GA algoritmasının ortalama uygunluk değeri 38.69 olup, standart sapması 6.10'dur. Bu sonuç, GA algoritmasının uygunluk değeri açısından diğer algoritmalara göre daha az başarılı olduğunu ve sonuçlarının daha değişken olduğunu göstermektedir.

DEA ve ABC algoritmaları sırasıyla 110.26 ve 112.66 ortalama uygunluk değerlerine sahiptir. DEA'nın standart sapması 1.01, ABC'nin ise 0.38 olarak belirlenmiştir. Bu iki algoritma, uygunluk değerleri açısından benzer performans sergilemekte olup, ABC algoritmasının daha tutarlı sonuçlar verdiği görülmektedir.

ACO ve GWO algoritmaları, sırasıyla 1112.13 ve 1015.34 ortalama uygunluk değerleri ile en yüksek değerleri elde etmişlerdir. ACO'nun standart sapması 2.27e-13 gibi oldukça düşük bir değer olup, bu sonuçların son derece tutarlı olduğunu göstermektedir. GWO'nun standart sapması ise 0.38'dir, bu da sonuçlarının oldukça tutarlı olduğunu göstermektedir.

Sonuç olarak, sabit istasyon ve sabit drone senaryosunda en iyi performansı PSO algoritması göstermiştir. Bu algoritma, en düşük ortalama uygunluk değerini sunarak projenin amacına en uygun sonucu sağlamıştır. Diğer algoritmalar arasında, GOA ve GA nispeten daha düşük uygunluk değerleri sunarken, ACO ve GWO algoritmaları en yüksek uygunluk değerlerini vermiştir.

Tablo 2'de, sabit istasyon ve hareketli drone senaryosu için çeşitli optimizasyon algoritmalarının en iyi uygunluk değeri ortalamaları ve standart sapma değerleri gösterilmektedir. Bu veriler, algoritmaların performanslarının karşılaştırılmasında önemli bir rol oynamaktadır. Çalışmanın amacı, en iyi uygunluk değerini minimize etmektir.

PSO algoritması, 27.48 ortalama uygunluk değeri ve 3.10 standart sapma ile en düşük uygunluk değerine ulaşmıştır. Bu sonuç, PSO'nun en iyi performansı gösterdiğini ve en uygun sonuçları sağladığını göstermektedir.

GOA , 30.18 ortalama uygunluk değeri ve 3.57 standart sapma ile ikinci en iyi sonucu elde etmiştir. Bu durum, PSO'ya yakın bir performans sergilediğini, ancak biraz daha yüksek bir uygunluk değerine sahip olduğunu göstermektedir.

GA , 43.48 ortalama uygunluk değeri ve 5.89 standart sapma ile üçüncü en iyi performansı sergilemiştir. Bu algoritmanın uygunluk değeri, PSO ve GOA'ya kıyasla daha yüksek olmasına rağmen, yine de nispeten iyi bir performans göstermiştir.

DEA , 121.51 ortalama uygunluk değeri ve 0.61 standart sapma ile dördüncü sırada yer almaktadır. Standart sapmasının düşük olması, sonuçların tutarlılığını gösterse de, ortalama uygunluk değerinin yüksek olması performansının yeterince iyi olmadığını ortaya koymaktadır.

ABC, 125.67 ortalama uygunluk değeri ve 0.40 standart sapma ile beşinci sırada yer almaktadır. Uygunluk değerinin yüksek olması, bu algoritmanın diğerlerine kıyasla daha az etkili olduğunu göstermektedir.

GWO algoritması, 1017.37 ortalama uygunluk değeri ve 0.18 standart sapma ile altıncı sırada yer almaktadır. Standart sapmasının düşük olmasına rağmen, ortalama uygunluk değerinin oldukça yüksek olması, performansının düşük olduğunu göstermektedir.

ACO algoritması, 1112.13 ortalama uygunluk değeri ve 2.27e(-13) standart sapma ile en yüksek uygunluk değerine sahiptir. Bu sonuç, ACO'nun en düşük performansa sahip algoritma olduğunu göstermektedir.

Bu senaryoda en iyi uygunluk değerini minimize etme amacı doğrultusunda, PSO algoritmasının en iyi performansı sergilediği, bunu sırasıyla GOA ve GA algoritmalarının takip ettiği görülmektedir. DEA, ABC, GWO ve ACO algoritmaları ise daha yüksek uygunluk değerleriyle daha düşük performans göstermiştir. Bu sonuçlar, sabit istasyon ve hareketli drone senaryosunda PSO algoritmasının tercih edilmesi gerektiğini önermektedir.

Tablo 3, drone istasyon atamalarını optimize etmeye yönelik çeşitli algoritmaların performanslarını karşılaştırmaktadır. Bu çalışmada, hareketli istasyon ve hareketli drone senaryosu için en iyi uygunluk değeri minimize edilmeye çalışılmış ve bu doğrultuda farklı algoritmalar kullanılmıştır.

PSO Algoritması, en düşük ortalama uygunluk değeri olan 25.15 ile en iyi performansı göstermiştir. Bu sonuç, PSO algoritmasının drone ve istasyon atamalarını optimize etmede oldukça etkili olduğunu göstermektedir. Ayrıca, standart sapma değeri 1.59 ile sonuçların tutarlılığı da dikkat çekicidir.

GOA , 27.69 ortalama uygunluk değeri ile PSO'yu takip etmektedir. Bu algoritmanın standart sapma değeri 1.85, sonuçların nispeten tutarlı olduğunu göstermektedir.

GA , ortalama 49.17 uygunluk değeri ile orta düzeyde bir performans sergilemiştir. Ancak, 4.96 gibi yüksek bir standart sapma değeri, bu algoritmanın sonuçlarının diğerlerine göre daha değişken olduğunu göstermektedir.

ABC ve DEA Algoritmaları, sırasıyla 122.56 ve 119.28 ortalama uygunluk değerleri ile benzer performanslar sergilemişlerdir. Bu algoritmaların standart sapma değerleri ise sırasıyla 0.31 ve 0.94 olup, DEA algoritmasının sonuçlarının ABC'ye göre daha değişken olduğunu göstermektedir.

ACO Algoritması, 598.51 gibi yüksek bir ortalama uygunluk değeri ile diğer algoritmalara kıyasla daha düşük bir performans sergilemiştir. Ancak, 1.13e(-13) gibi son derece düşük bir standart sapma değeri, sonuçların aşırı derecede tutarlı olduğunu göstermektedir.

GWO Algoritması, 1018.32 ortalama uygunluk değeri ile en yüksek değere sahiptir ve bu da en düşük performansı gösterdiğini işaret etmektedir. Standart sapma değeri ise 0.51 olup, sonuçların nispeten tutarlı olduğunu göstermektedir.

Sonuç olarak, PSO algoritması en düşük ortalama uygunluk değeri ile en iyi performansı göstermiştir ve bu da drone istasyon atamalarının optimize edilmesinde bu algoritmanın en uygun seçenek olduğunu göstermektedir. Diğer algoritmaların performansları da belirli senaryolar için değerlendirilebilir ancak bu çalışma özelinde PSO algoritması en iyi sonucu vermiştir.

5.1 *BULGULAR VE ÇIKTILAR*

Bu bölümde projenin çalışması sonucu elde edilen grafikler ve çıktılar verilmiştir. ABC, ACO, DEA, GA, GOA, GWO, PSO algortimaları 10000 iterasyonla çalıştırılmış her algoirtma ve her senaryo için En İyi Uygunluk Değeri Grafiği ve En İyi Çözüm Grafiği elde edilmiştir. Ayrıca her algoritmanın optimizasyon sonucu çıktı görüntüleri elde edilmiştir. Bu görüntüler Elde Edilen En İyi Uygunluk Değeri, Ortalama Mesafe, Ortalama Batarya Kullanımı, Geçen Süre, Kapanma Hızı gibi değerleri içermektedir.

Elde Edilen En İyi Uygunluk Değeri: Algoritma tarafından bulunan en iyi çözümün uygunluk değeridir. Bu değer, optimize edilmek istenen hedef fonksiyonun ne kadar iyi optimize edildiğini gösterir. Projemizde minimize edilmeye çalışan fonksiyonun performans metriğidir

Ortalama Mesafe: Algoritma tarafından belirlenen en iyi çözümle drone istasyonları arasındaki ortalama mesafedir. Bu, dronların operasyon sırasında ortalama olarak ne kadar yol aldığını veya ne kadar yakın olduklarını gösterir.

Ortalama Batarya Kullanımı: Algoritma tarafından belirlenen en iyi çözümle dronların ortalama batarya kullanımıdır. Bu, dronların görev sırasında ortalama olarak batarya kapasitelerinin ne kadarını kullandığını gösterir.

Geçen Süre: Algoritmanın çalıştırılması için harcanan toplam süredir. Bu süre, algoritmanın ne kadar hızlı veya yavaş çalıştığını, optimizasyon işleminin ne kadar zaman aldığını gösterir.

Kapanma Hızı: Algoritmanın optimizasyon problemini çözerken ne kadar hızlı yakınsadığını gösteren bir metriktir. Daha yüksek bir kapanma hızı, algoritmanın daha hızlı konverjans sağladığını gösterir.

Bu metrikler, algoritmaların performanslarını ve elde edilen çözümlerin kalitesini değerlendirmek için kullanılır. Raporda çok fazla yer kaplamaması amacıyla her senaryo için yalnızca bir çıktı görüntüsüne yer verilmiştir.

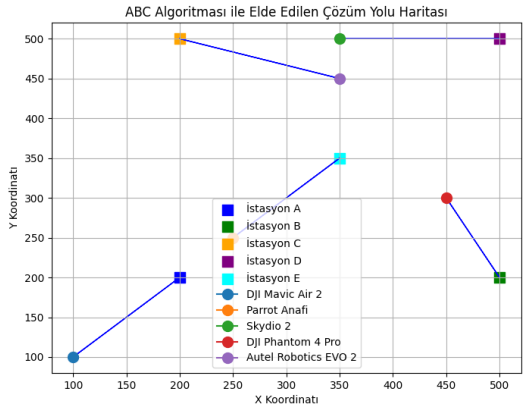

Figure 12: Senaryo1: ABC En İyi Çözüm Grafiği

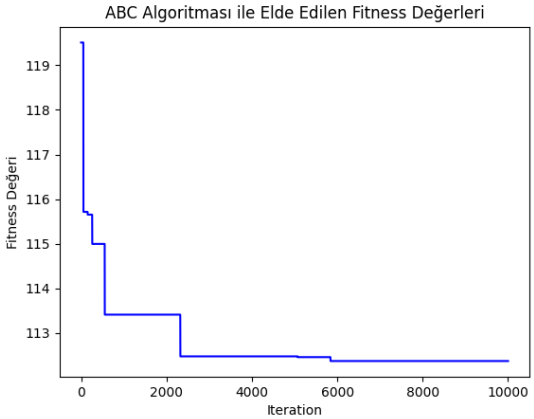

Figure 13: Senaryo1: ABC Uygunluk Değeri Grafiği

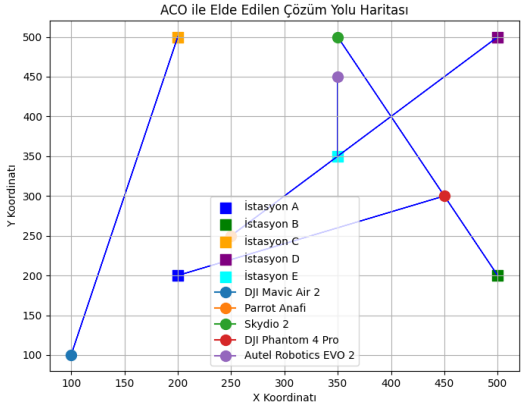

Figure 14: Senaryo1: ACO En İyi Çözüm Grafiği

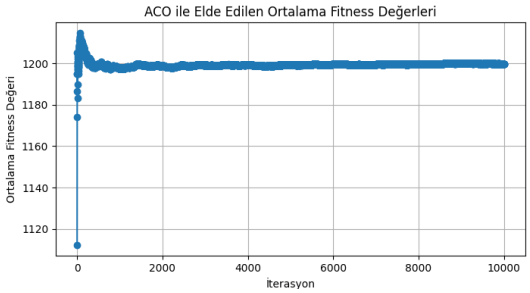

Figure 15: Senaryo1: ACO Uygunluk Değeri Grafiği

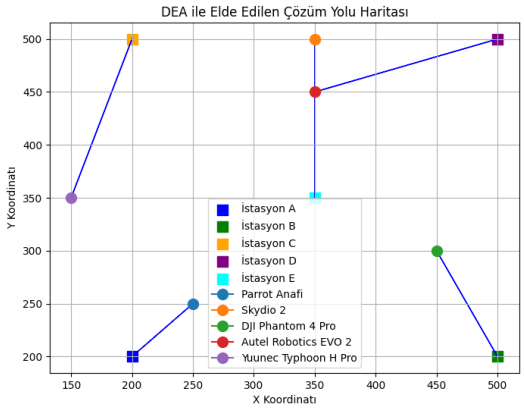

Figure 16: Senaryo1: DEA En İyi Çözüm Grafiği

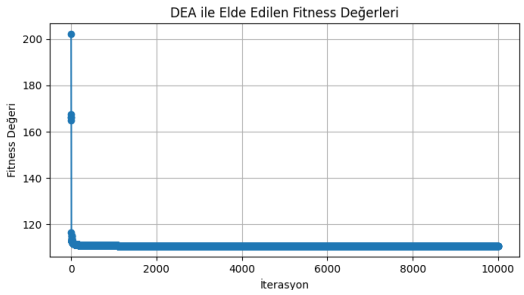

Figure 17: Senaryo1: DEA Uygunluk Değeri Grafiği

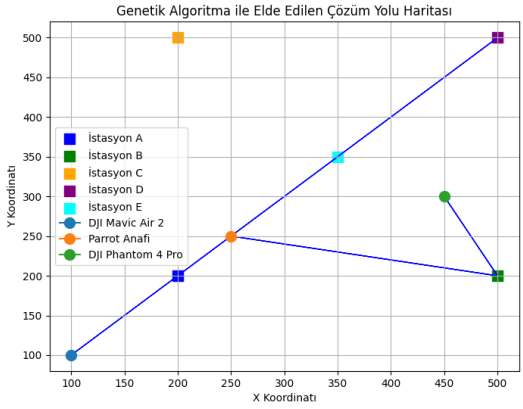

Figure 18: Senaryo1: GA En İyi Çözüm Grafiği

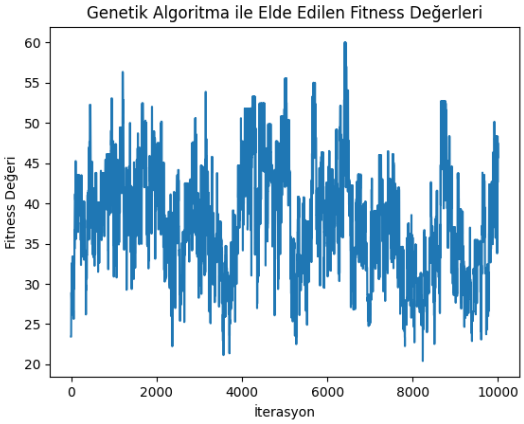

Figure 19: Senaryo1: GA Uygunluk Değeri Grafiği

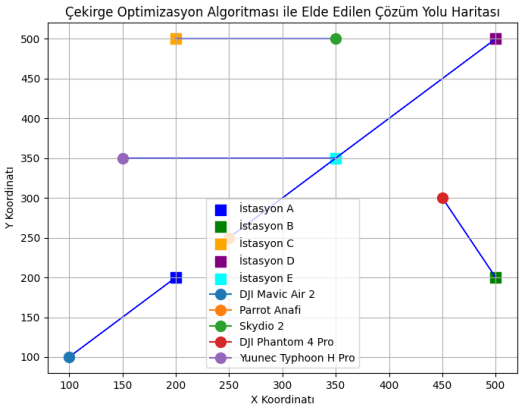

Figure 20: Senaryo1: GOA En İyi Çözüm Grafiği

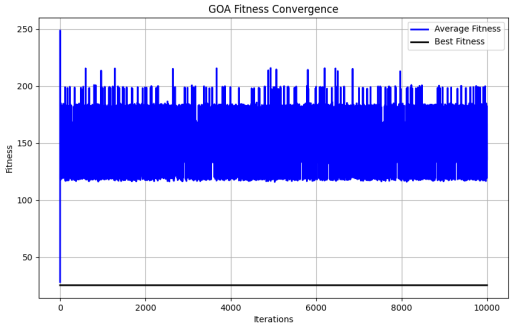

Figure 21: Senaryo1: GOA Uygunluk Değeri Grafiği

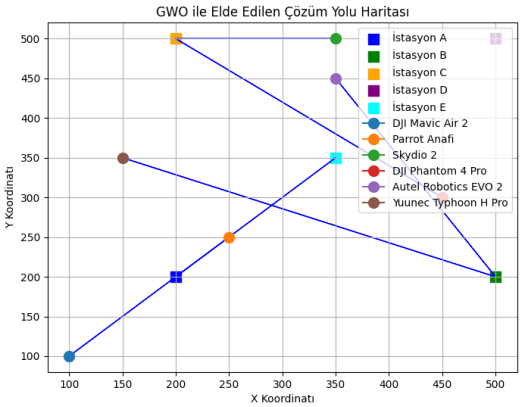

Figure 22: Senaryo1: GWO En İyi Çözüm Grafiği

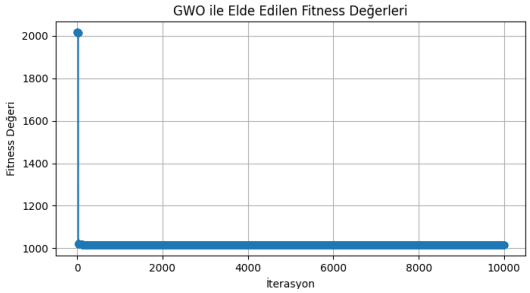

Figure 23: Senaryo1: GWO Uygunluk Değeri Grafiği

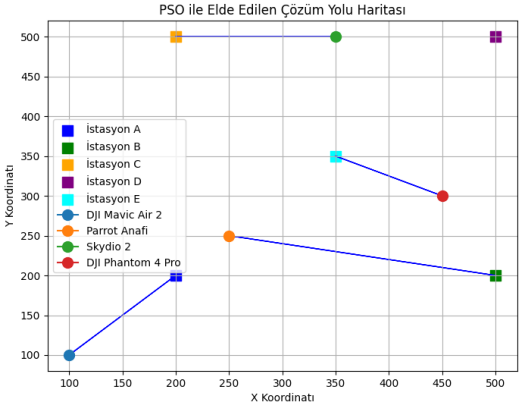

Figure 24: Senaryo1: PSO En İyi Çözüm Grafiği

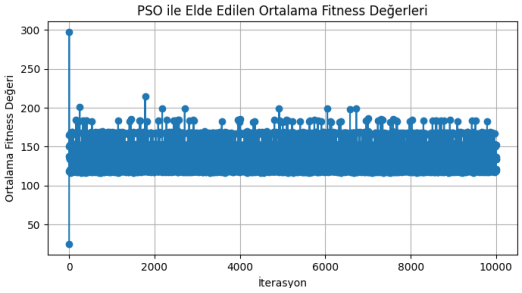

Figure 25: Senaryo1: PSO Uygunluk Değeri Grafiği

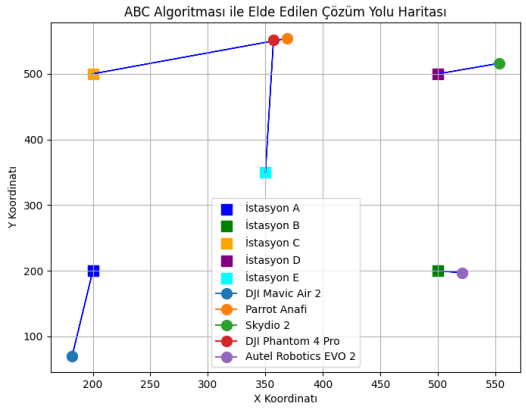

Figure 26: Senaryo2: ABC En İyi Çözüm Grafiği

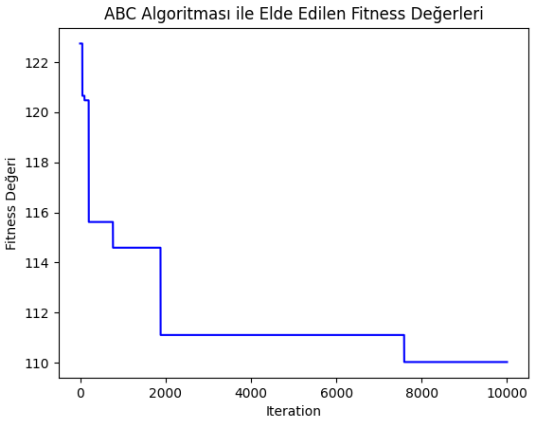

Figure 27: Senaryo2: ABC Uygunluk Değeri Grafiği

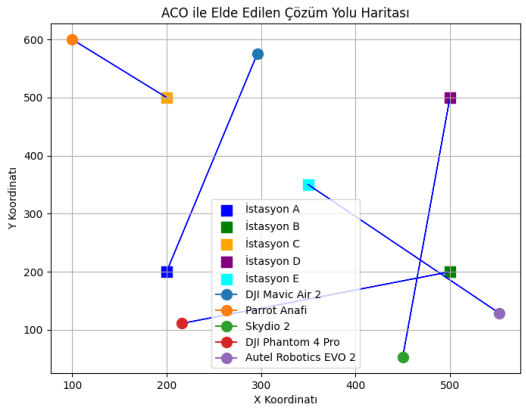

Figure 28: Senaryo2: ACO En İyi Çözüm Grafiği

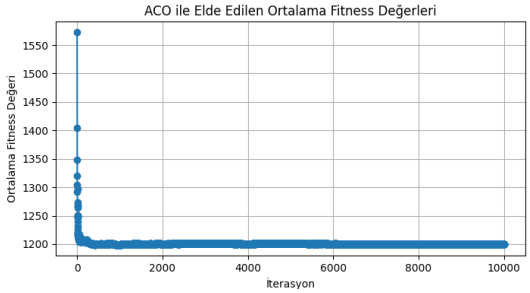

Figure 29: Senaryo2: ACO Uygunluk Değeri Grafiği

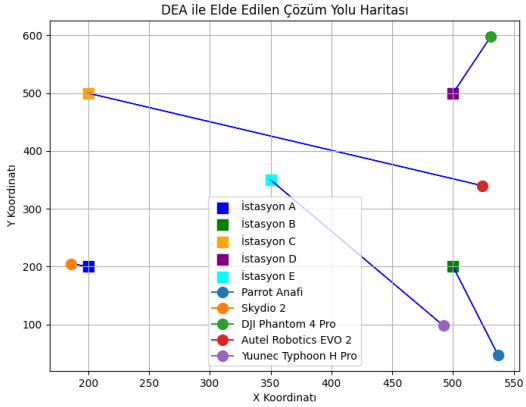

Figure 30: Senaryo2: DEA En İyi Çözüm Grafiği

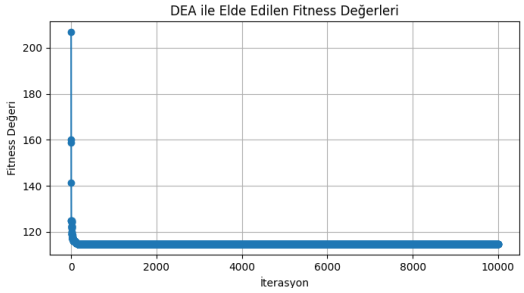

Figure 31: Senaryo2: DEA Uygunluk Değeri Grafiği

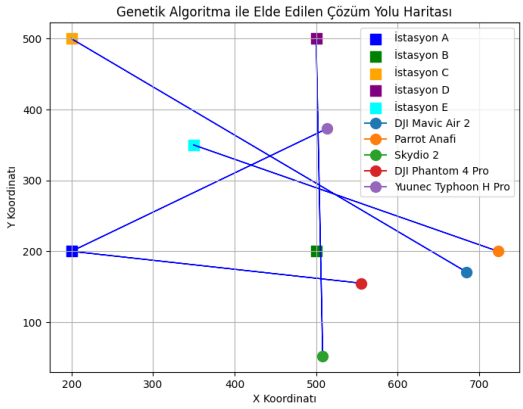

Figure 32: Senaryo2: GA En İyi Çözüm Grafiği

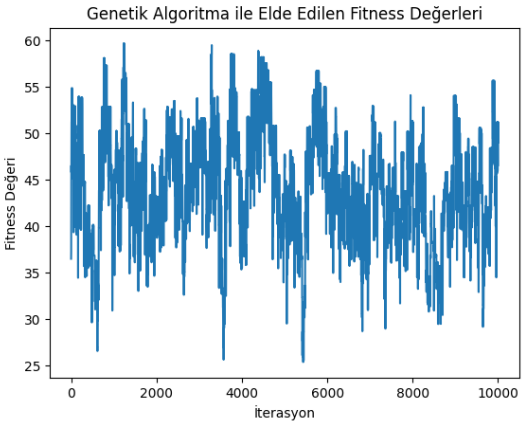

Figure 33: Senaryo2: GA Uygunluk Değeri Grafiği

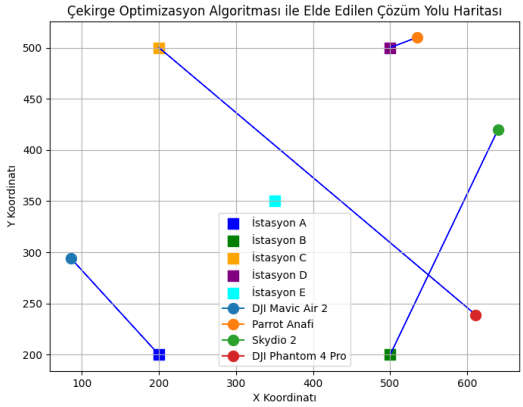

Figure 34: Senaryo2: GOA En İyi Çözüm Grafiği

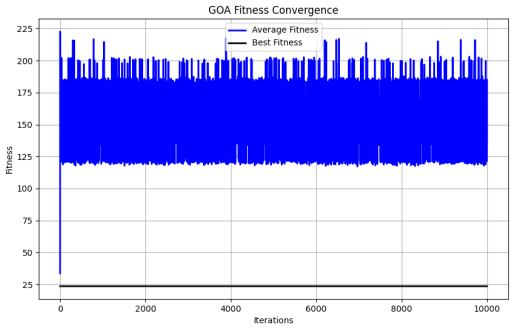

Figure 35: Senaryo2: GOA Uygunluk Değeri Grafiği

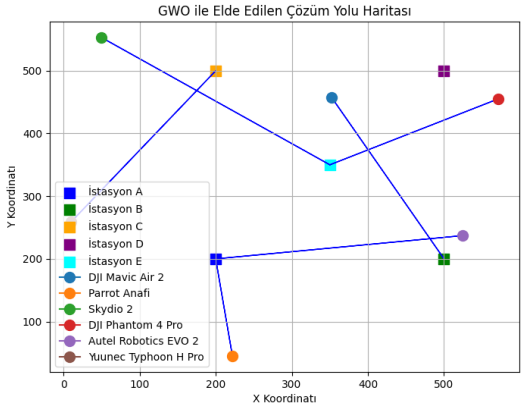

Figure 36: Senaryo2: GWO En İyi Çözüm Grafiği

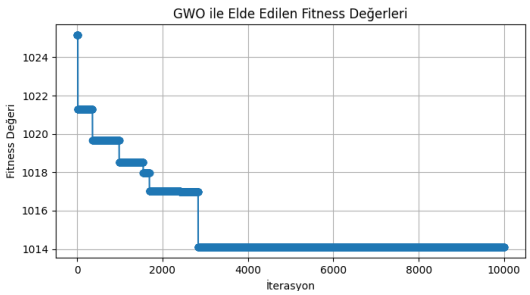

Figure 37: Senaryo2: GWO Uygunluk Değeri Grafiği

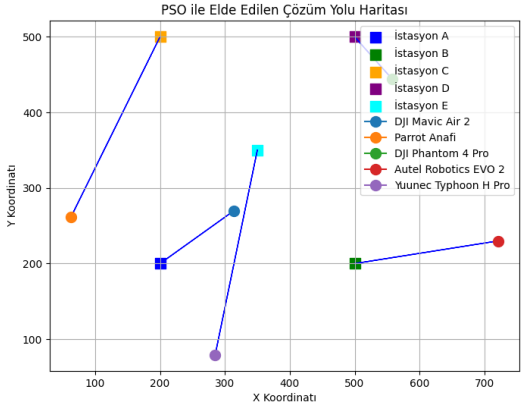

Figure 38: Senaryo2: PSO En İyi Çözüm Grafiği

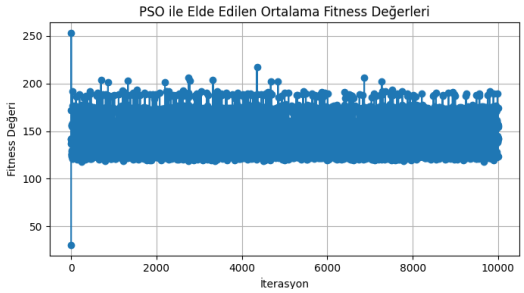

Figure 39: Senaryo2: PSO Uygunluk Değeri Grafiği

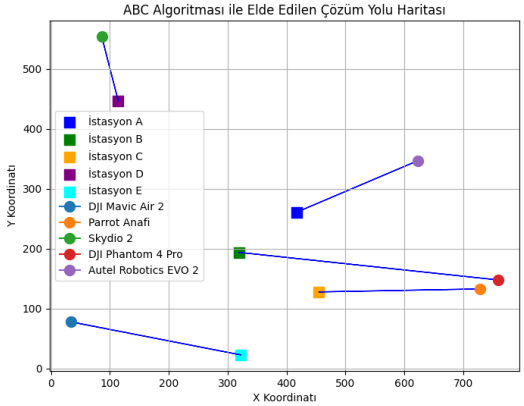

Figure 40: Senaryo3: ABC En İyi Çözüm Grafiği

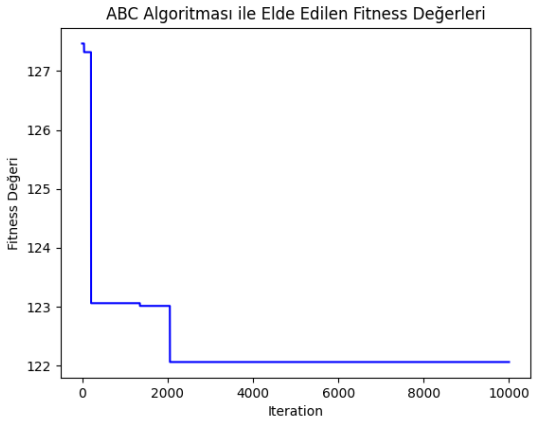

Figure 41: Senaryo3: ABC Uygunluk Değeri Grafiği

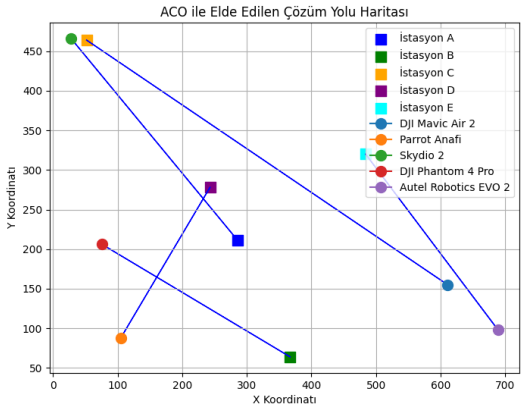

Figure 42: Senaryo3: ACO En İyi Çözüm Grafiği

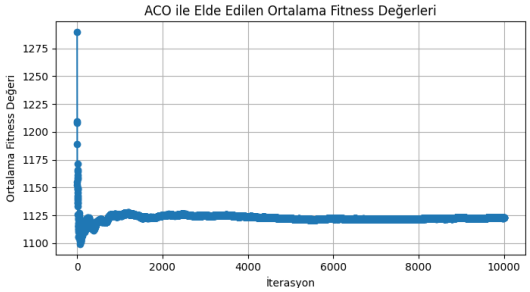

Figure 43: Senaryo3: ACO Uygunluk Değeri Grafiği

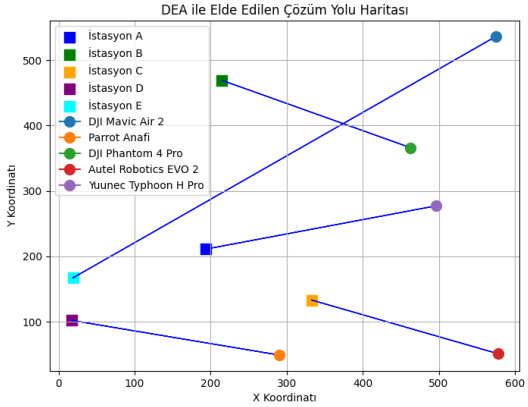

Figure 44: Senaryo3: DEA En İyi Çözüm Grafiği

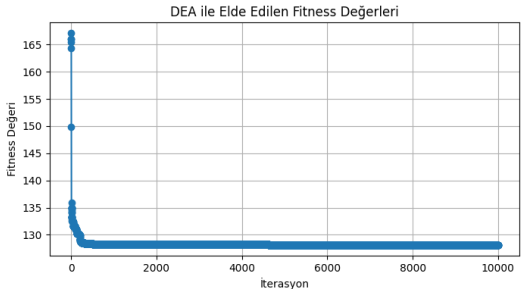

Figure 45: Senaryo3: DEA Uygunluk Değeri Grafiği

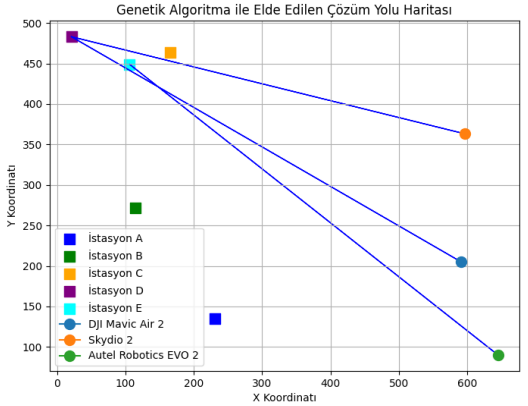

Figure 46: Senaryo3: GA En İyi Çözüm Grafiği

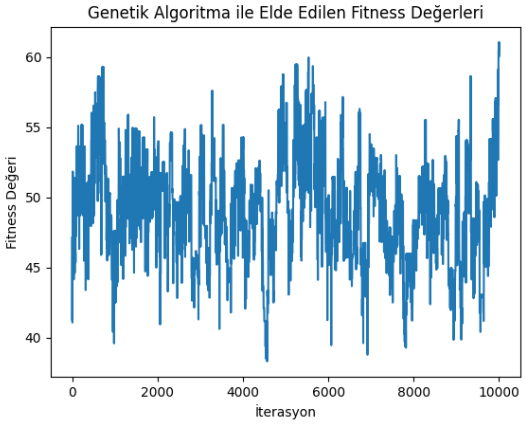

Figure 47: Senaryo3: GA Uygunluk Değeri Grafiği

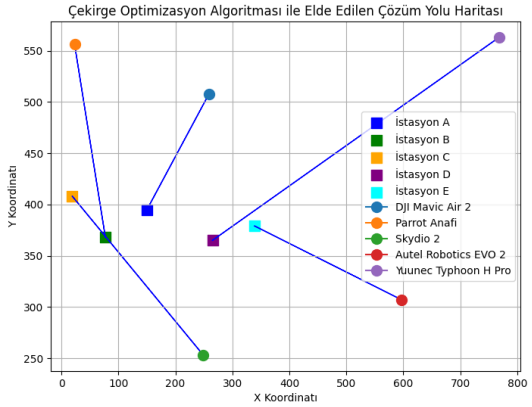

Figure 48: Senaryo3: GOA En İyi Çözüm Grafiği

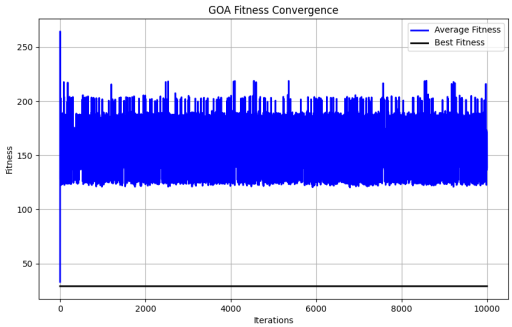

Figure 49: Senaryo3: GOA Uygunluk Değeri Grafiği

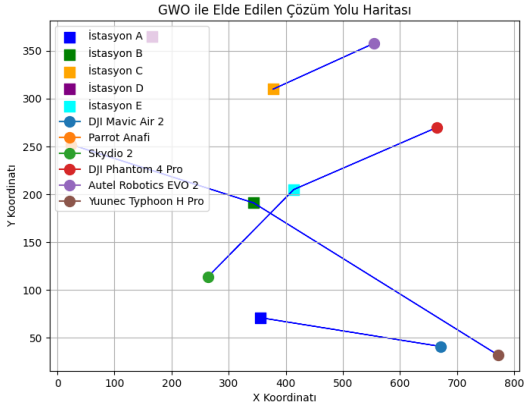

Figure 50: Senaryo3: GWO En İyi Çözüm Grafiği

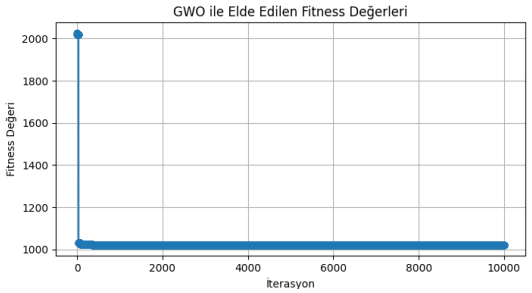

Figure 51: Senaryo3: GWO Uygunluk Değeri Grafiği

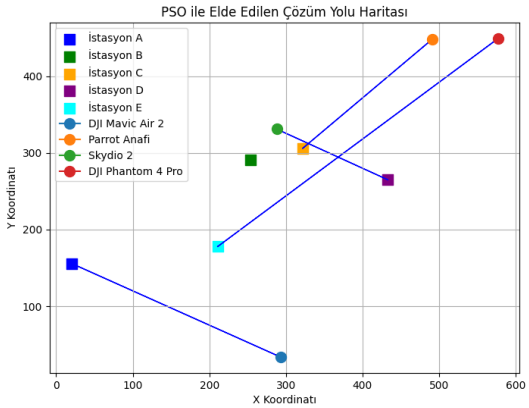

Figure 52: Senaryo3: PSO En İyi Çözüm Grafiği

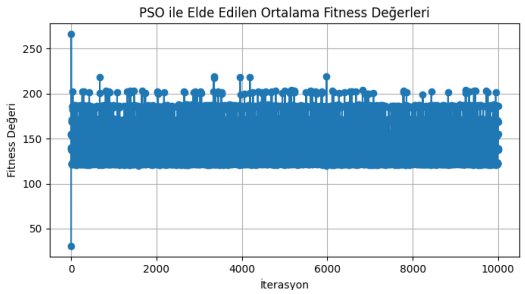

Figure 53: Senaryo3: PSO Uygunluk Değeri Grafiği

Figure 54: Senaryo1: Çıktı

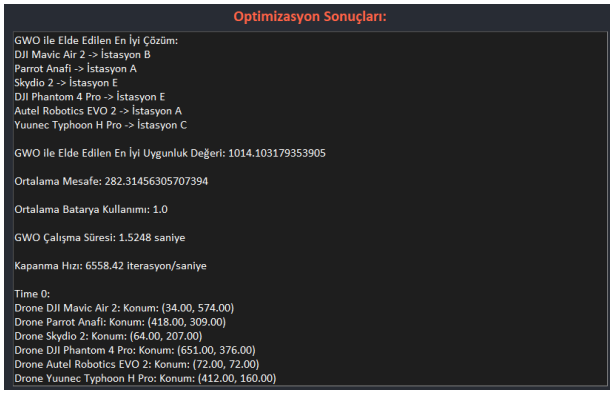

Figure 55: Senaryo2: Çıktı

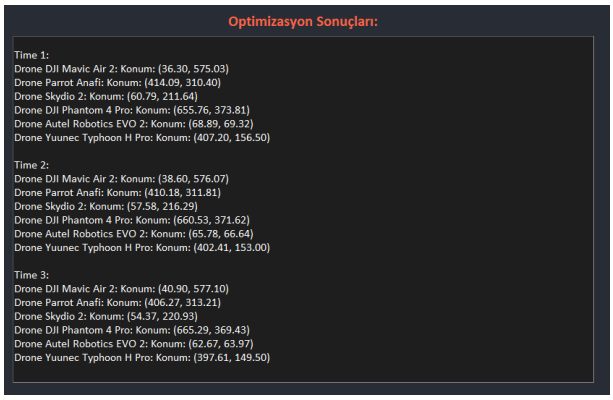

Figure 56: Senaryo2: Çıktı

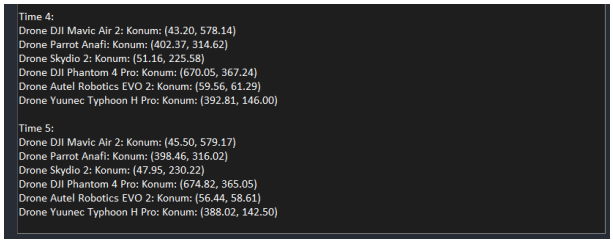

Figure 57: Senaryo2: Çıktı

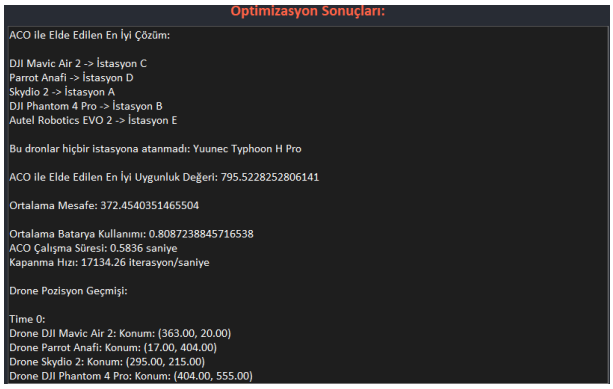

Figure 58: Senaryo3: Çıktı

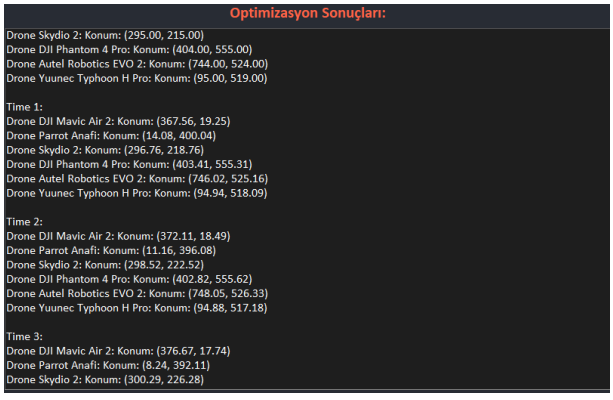

Figure 59: Senaryo3: Çıktı

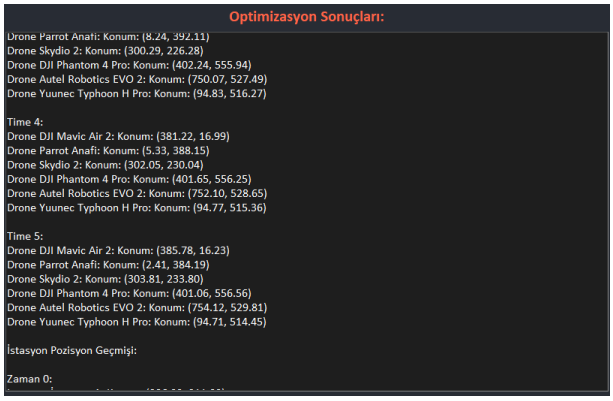

Figure 60: Senaryo3: Çıktı

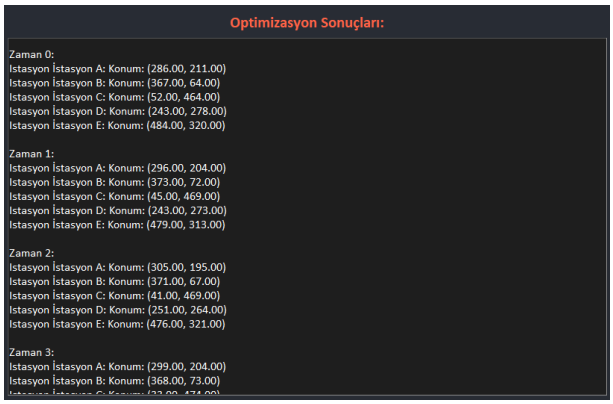

Figure 61: Senaryo3: Çıktı

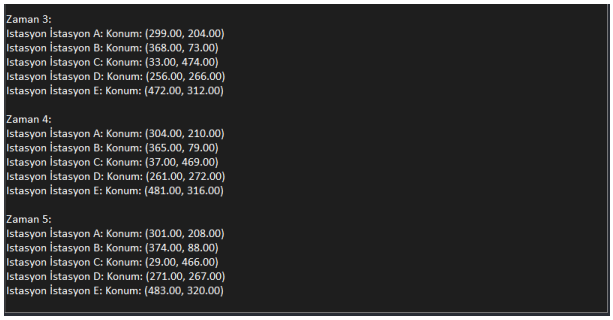

Figure 62: Senaryo3: Çıktı

BÖLÜM: VI

SONUÇ

6

SONUC

Projenin amacı, sabit istasyon ve sabit drone, sabit istasyon ve hareketli drone, hareketli istasyon ve hareketli drone senaryolarında optimizasyon algoritmalarının performanslarını karşılaştırarak en iyi uygunluk değerini minimize etmektir. Bu doğrultuda, yedi farklı algoritma (PSO, GOA, GA, DEA, ABC, ACO, GWO) kullanılmış ve elde edilen ortalama uygunluk değerleri ile standart sapmaları değerlendirilmiştir.

Tablo 1'de sunulan verilere göre, PSO algoritması sabit istasyon ve sabit drone senaryosunda en düşük ortalama uygunluk değeri olan 20.95 ile en iyi performansı sergilemiştir. Ayrıca, 2.61 standart sapma değeri ile sonuçlarının göreceli olarak tutarlı olduğu gözlemlenmiştir.

GOA algoritması, 22.34 ortalama uygunluk değeri ile PSO'dan sonra ikinci sırada gelmiştir. Ancak, 2.84 standart sapma değeri ile sonuçlarının PSO'ya kıyasla biraz daha değişken olduğu belirlenmiştir.

GA algoritması ise 38.69 ortalama uygunluk değeri ve 6.10 standart sapma ile daha düşük bir performans sergilemiştir. Bu durum, GA'nın diğer algoritmalara göre uygunluk değeri açısından daha az başarılı olduğunu ve sonuçlarının daha değişken olduğunu göstermektedir.

DEA ve ABC algoritmaları sırasıyla 110.26 ve 112.66 ortalama uygunluk değerlerine sahiptir. DEA'nın 1.01, ABC'nin ise 0.38 standart sapma değeri ile ABC algoritmasının daha tutarlı sonuçlar verdiği gözlemlenmiştir.

ACO ve GWO algoritmaları, sırasıyla 1112.13 ve 1015.34 ortalama uygunluk değerleri ile en yüksek değerleri elde etmişlerdir. ACO'nun 2.27e-13 gibi düşük bir standart sapma değeri ile sonuçlarının son derece tutarlı olduğu görülmüştür. GWO'nun ise 0.38 standart sapma değeri ile sonuçlarının oldukça tutarlı olduğu belirlenmiştir.

Tablo 2'de ise sabit istasyon ve hareketli drone senaryosu için PSO algoritmasının 27.48 ortalama uygunluk değeri ve 3.10 standart sapma ile en düşük uygunluk değerine sahip olduğu görülmüştür. GOA, 30.18 ortalama uygunluk değeri ve 3.57 standart sapma ile ikinci en iyi performansı sergilemiştir. GA algoritması ise 43.48 ortalama uygunluk değeri ve 5.89 standart sapma ile diğer senaryolara göre daha yüksek bir uygunluk değeri sağlamıştır, ancak sonuçlarının daha değişken olduğu belirlenmiştir.

DEA ve ABC algoritmaları, sırasıyla 121.51 ve 125.67 ortalama uygunluk değerleri ile daha yüksek performans sergilemişlerdir. DEA'nın 0.61, ABC'nin ise 0.40 standart sapma değeri ile DEA'nın daha tutarlı sonuçlar verdiği tespit edilmiştir.

GWO ve ACO algoritmaları ise 1017.37 ve 1112.13 ortalama uygunluk değerleri ile en düşük performansı sergilemişlerdir. ACO'nun 2.27e(-13) standart sapma değeri ile sonuçlarının aşırı derecede tutarlı olduğu görülmüştür.

Tablo 3'te ise hareketli istasyon ve hareketli drone senaryosunda PSO algoritmasının en iyi performansı gösterdiği, 25.15 ortalama uygunluk değeri ve 1.59 standart sapma ile belirtilmiştir. GOA, 27.69 ortalama uygunluk değeri ve 1.85 standart sapma ile ikinci sırada gelmiştir. GA, 49.17 ortalama uygunluk değeri ve 4.96 standart sapma ile nispeten daha düşük bir performans sergilemiştir.

DEA ve ABC algoritmaları, 119.28 ve 122.56 ortalama uygunluk değerleri ile benzer performans göstermişlerdir. DEA'nın 0.94, ABC'nin ise 0.31 standart sapma değeri ile DEA'nın daha değişken sonuçlar verdiği belirlenmiştir.

GWO ve ACO algoritmaları ise sırasıyla 1018.32 ve 598.51 ortalama uygunluk değerleri ile en düşük performansı sergilemişlerdir. ACO'nun 1.13e(-13) standart sapma değeri ile sonuçlarının aşırı derecede tutarlı olduğu görülmüştür.

Sonuç olarak, üç farklı senaryoda da PSO algoritmasının en düşük ortalama uygunluk değeri ile en iyi performansı gösterdiği ve drone istasyon atamalarının optimize edilmesinde tercih edilmesi gerektiği sonucuna varılmıştır. Diğer algoritmaların performansları belirli senaryolar için değerlendirilebilir ancak bu çalışma kapsamında PSO algoritması en kapsamlı ve en etkili sonuçları sağlamıştır.

KAYNAKÇA

[1] Sung-Chan Choi, Jong-Hong Park, and Jaeho Kim. “A networking framework for multiple-heterogeneous unmanned vehicles in FANETs”. In: *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE. 2019, pp. 13–15.

[2] Adnan Nadeem et al. “A review and classification of flying ad-hoc network (FANET) routing strategies”. In: *Journal of Basic and Applied Scientific Research* 8.3 (2018), pp. 1–8.

[3] Gheorghe Udeanu, Alexandra Dobrescu, and Mihaela Oltean. “Unmanned aerial vehicle in military operations”. In: *Sci. Res. Educ. Air Force* 18.1 (2016), pp. 199–206.

[4] M Anwar Ma’Sum et al. “Simulation of intelligent unmanned aerial vehicle (uav) for military surveillance”. In: *2013 international conference on advanced computer science and information systems (ICACSIS)*. IEEE. 2013, pp. 161–166.

[5] Sudhir Kumar Chaturvedi et al. “Comparative review study of military and civilian unmanned aerial vehicles (UAVs)”. In: *INCAS bulletin* 11.3 (2019), pp. 181–182.

[6] Giordano Bruno Antoniazzi Ronconi, Thaís Jessinski Batista, and Victor Merola. “The utilization of Unmanned Aerial Vehicles (UAV) for military action in foreign airspace”. In: *UFRGSMUN: UFRGS Model United Nations Journal* 2 (2014), pp. 137–180.

[7] Bieliakov Robert and Fesenko Oleksii. “FANET MANAGEMENT PROCESS SIMULATION”. In: *The XII International Scientific and Practical Conference" Youth, education and*

science through today's challenges", December 04-06, 2023, Bordeaux, France. 454 p. P. 395.

[8] Zhuohui Yao et al. "Resource allocation for 5G-UAV-based emergency wireless communications". In: *IEEE Journal on Selected Areas in Communications* 39.11 (2021), pp. 3395–3410.

[9] Kirtan Gopal Panda et al. "Design and deployment of UAV-aided post-disaster emergency network". In: *IEEE Access* 7 (2019), pp. 102985–102999.

[10] Yogianandh Naidoo, Riaan Stopforth, and Glen Bright. "Development of an UAV for search & rescue applications". In: *IEEE Africon'11*. IEEE. 2011, pp. 1–6.

[11] Yu Lin, Tianyu Wang, and Shaowei Wang. "UAV-assisted emergency communications: An extended multi-armed bandit perspective". In: *IEEE Communications Letters* 23.5 (2019), pp. 938–941.

[12] Yong Zeng, Rui Zhang, and Teng Joon Lim. "Throughput maximization for UAV-enabled mobile relaying systems". In: *IEEE Transactions on communications* 64.12 (2016), pp. 4983–4996.

[13] Milan Erdelj et al. "Help from the sky: Leveraging UAVs for disaster management". In: *IEEE Pervasive Computing* 16.1 (2017), pp. 24–32.

[14] David Greer, Phillip McKerrow, and Jo Abrantes. "Robots in urban search and rescue operations". In: *Australasian Conference on Robotics and Automation, Auckland*. Citeseer. 2002, pp. 27–29.

[15] Matthew Dunbabin and Lino Marques. "Robots for environmental monitoring: Significant advancements and applications". In: *IEEE Robotics & Automation Magazine* 19.1 (2012), pp. 24–39.

[16] Rajesh Gupta et al. “VAHAK: A blockchain-based outdoor delivery scheme using UAV for healthcare 4.0 services”. In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE. 2020, pp. 255–260.

[17] Min Chen et al. “Cognitive internet of vehicles”. In: *Computer Communications* 120 (2018), pp. 58–70.

[18] Yin Zhang et al. “SOVCAN: Safety-oriented vehicular controller area network”. In: *IEEE Communications Magazine* 55.8 (2017), pp. 94–99.

[19] Ibrar Yaqoob et al. “Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges”. In: *IEEE wireless communications* 24.3 (2017), pp. 10–16.

[20] Hanlin Zhang et al. “Scheduling methods for unmanned aerial vehicle based delivery systems”. In: *2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*. IEEE. 2014, pp. 6C1–1.

[21] Khin Thida San et al. “UAV delivery monitoring system”. In: *MATEC Web of Conferences*. Vol. 151. EDP Sciences. 2018, p. 04011.

[22] Geoffrey Ling and Nicole Draghic. “Aerial drones for blood delivery”. In: *Transfusion* 59.S2 (2019), pp. 1608–1611.

[23] Alain Ajami et al. “Path planning and ground control station simulator for uav”. In: *2013 IEEE Aerospace Conference*. IEEE. 2013, pp. 1–13.

[24] Sana Ullah et al. “UAV-enabled healthcare architecture: Issues and challenges”. In: *Future Generation Computer Systems* 97 (2019), pp. 425–432.

[25] Tao Sheng et al. “Unmanned Aerial Vehicle Mediated Drug Delivery for First Aid”. In: *Advanced Materials* 35.10 (2023), p. 2208648.

[26] Ayşegül GÜVEN and Sadık KARA. “BİR MODEL UÇAKLA YER İSTASYONU ARASINDA ORTAM BİLGİLERİNİN İLETİMİ VE UÇUŞ SİMÜLASYONU”. In: ().

[27] Mahmud Hossain, Md Arafat Hossain, and Farhana Akter Sunny. “A UAV-based traffic monitoring system for smart cities”. In: *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*. IEEE. 2019, pp. 1–6.

[28] Qiuchen Gu et al. “A vehicle-UAV operation scheme for instant delivery”. In: *Computers & Industrial Engineering* 149 (2020), p. 106809.

[29] Haoran Niu, Nuria Gonzalez-Prelcic, and Robert W Heath. “A UAV-based traffic monitoring system-invited paper”. In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. IEEE. 2018, pp. 1–5.

[30] Ye Hong, Jiancheng Fang, and Ye Tao. “Ground control station development for autonomous UAV”. In: *Intelligent Robotics and Applications: First International Conference, ICIRA 2008 Wuhan, China, October 15-17, 2008 Proceedings, Part II 1*. Springer. 2008, pp. 36–44.

[31] Ivan Maza et al. “Multimodal interface technologies for UAV ground control stations: a comparative analysis”. In: *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*. Springer. 2010, pp. 371–391.

[32] Eva Husson, Olle Hagner, and Frauke Ecke. “Unmanned aircraft systems help to map aquatic vegetation”. In: *Applied vegetation science* 17.3 (2014), pp. 567–577.

[33] KV Najiya and M Archana. “UAV video processing for traffic surveillence with enhanced vehicle detection”. In: *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*. IEEE. 2018, pp. 662–668.

[34] İZCİ Volkan and ULVİ Ali. “Yer Kontrol Noktalarının Harita Üretimine Etkileri”. In: *International Geoinformatics Student Symposium (IGSS)*. Vol. 1. 1. 2021, pp. 41–47.

[35] Omar Sami Oubbati et al. “Leveraging communicating UAVs for emergency vehicle guidance in urban areas”. In: *IEEE Transactions on Emerging Topics in Computing* 9.2 (2019), pp. 1070–1082.

[36] Wenbo Jin et al. “Research on application and deployment of UAV in emergency response”. In: *2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC)*. IEEE. 2020, pp. 277–280.

[37] Milan Erdelj and Enrico Natalizio. “UAV-assisted disaster management: Applications and open issues”. In: *2016 international conference on computing, networking and communications (ICNC)*. IEEE. 2016, pp. 1–5.

[38] J Vijitha Ananthi and P Subha Hency Jose. “Implementation of IoT and UAV Based WBAN for healthcare applications”. In: *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*. IEEE. 2021, pp. 37–42.

[39] Hazim Shakhatreh et al. “On the continuous coverage problem for a swarm of UAVs”. In: *2016 IEEE 37th Sarnoff Symposium*. IEEE. 2016, pp. 130–135.

[40] Farhan Mohammed et al. “UAVs for smart cities: Opportunities and challenges”. In: *2014 international conference on unmanned aircraft systems (ICUAS)*. IEEE. 2014, pp. 267–273.

[41] Hacı Murat YILMAZ et al. “İnsansız hava aracı ile ortofoto üretimi ve Aksaray Üniversitesi kampüsü örneği”. In: *Geomatik* 3.2 (2018), pp. 129–136.

[42] Zhenyu Xiao et al. “Unmanned aerial vehicle base station (UAV-BS) deployment with millimeter-wave beamforming”. In: *IEEE Internet of Things Journal* 7.2 (2019), pp. 1336–1349.