Genetik Algoritma ve A* Yöntemi ile Drone Filosu Optimizasyonu

Genetic Algorithm and A Method for Drone Fleet Optimization*

Emirhan GÜLER
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Ögrenci No: 221307086
Grup No: 13
eguler_73@hotmail.com

Alperen GÖRMEZ
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Ögrenci No: 221307104
Grup No: 13
alperen.41gs@hotmail.com

Osman Musap ÇUHA
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Ögrenci No: 221307030
Grup No: 13
musabcuha@gmail.com

Index Terms—Genetik Algoritma, Drone ile Teslimat, Rota Planlama, Kısıt Yönetimi, A* Arama, Uçuşa Yasak Bölgeler, Otonom Sistemler, Optimizasyon.

Abstract—This study presents a constraint-aware optimization system for autonomous drone fleets performing last-mile deliveries. The proposed architecture integrates a Genetic Algorithm (GA), A* pathfinding, and direct constraint management approaches to assign delivery tasks to drones while satisfying critical constraints such as limited battery capacity, payload weight limits, active no-fly zones (NFZs), and delivery time windows. Each delivery is modeled with spatial and temporal attributes, while drones are defined with operational parameters such as speed, position, battery, and weight capacity. No-fly zones are represented as polygons with defined active time intervals, and any route intersecting an active zone is invalidated. The Genetic Algorithm uses a fitness function to optimize assignments, and A* computes the most cost-efficient routes between drones and delivery locations. The implementation is developed in the Python programming language using modular classes and verified with scenario data from 'data/veri-seti.txt'. Results show that the developed system efficiently generates feasible and lowcost delivery plans, making it suitable for potential applications in smart logistics systems.

Index Terms—Genetic Algorithm, Drone Delivery, Path Planning, Constraint Handling, A* Search, No-Fly Zones, Autonomous Systems, Optimization.

I. GiRiş

Son yıllarda İnsansız Hava Aracı (İHA)'lar, özellikle kentsel alanlarda son kilometre teslimatlarında sundukları hız, esneklik ve potansiyel maliyet avantajları nedeniyle lojistik sektöründe devrim yaratma potansiyeliyle dikkat çekmektedir. E-ticaretin hızla büyümesiyle birlikte, teslimat süreçlerinin verimliliği ve sürdürülebilirliği kritik bir önem kazanmış, bu da İHA tabanlı çözümlere olan ilgiyi artırmıştır. Ancak, drone filolarının operasyonel verimliliği, sınırlı batarya ömrü, degişken yük taşıma kapasiteleri, Uçuşa Yasak Bölge (No-Fly Zone) (NFZ)'lerin varlığı, teslimatlar için belirlenmiş zaman pencereleri ve farklı hava koşulları gibi bir dizi karmaşık kısıtla dogrudan ilişkilidir. Bu kısıtların eş zamanlı olarak yönetilmesi, çok sayıda drone için optimal görev ataması ve rota planlaması problemlerini oldukça zorlu hale getirmektedir.

Bu çalışma, belirtilen bu operasyonel ve çevresel kısıtları dikkate alarak otonom drone filoları için etkin bir görev atama ve rota optimizasyon sistemi geliştirmeyi amaçlamaktadır. Sunulan çözüm, Genetik Algoritma (GA) ve A* rota bulma algoritmalarını temel alan hibrit bir yaklaşım kullanmaktadır. Sistem, drone'ların enerji tüketimini, yük kapasitelerini, bireysel hızlarını, NFZ'lerin aktif zamanlarını ve teslimatların zaman pencerelerini göz önünde bulundurarak geçerli ve optimize edilmiş uçuş planları oluşturur. Ayrıca, temel bir karşılaştırma ve analiz saglamak amacıyla bir Açgözlü (Greedy) algoritma da geliştirilmiş ve sisteme dahil edilmiştir.

Bu raporun devamında, öncelikle ilgili literatürdeki çalışmalar incelenecek (Bölüm II), ardından geliştirilen sistemin mimarisi, kullanılan veri modelleri ve algoritmik yaklaşımlar detaylandırılacaktır (Bölüm III). Deneysel kurulum ve kullanılan test senaryosu Bölüm IV'de açıklanacak, elde edilen sonuçlar ve bu sonuçların analizi Bölüm V'te sunulacaktır. Son olarak, Bölüm VI'da çalışmanın genel bir değerlendirmesi yapılacak ve gelecek çalışma önerileri tartışılacaktır.

II. İLGiLi ÇALIşMALAR

Drone tabanlı teslimat sistemleri ve filo optimizasyonu, son yıllarda hem akademik araştırmaların hem de endüstriyel uygulamaların odak noktası haline gelmiştir. Bu alandaki çalışmalar genellikle Araç Rotalama Problemi (Vehicle Routing Problem) (VRP) ve Gezgin Satıcı Problemi (Traveling Salesman Problem) (TSP) gibi klasik optimizasyon problemlerinin drone'lara özgü kısıtlarla genişletilmiş varyasyonlarını ele almaktadır. Literatürde, görev ataması, rota planlama, enerji optimizasyonu ve çarpışmadan kaçınma gibi alt problemler için çeşitli algoritmik yaklaşımlar önerilmiştir.

Meta-sezgisel algoritmalar, özellikle GA, Karınca Kolonisi Optimizasyonu (Ant Colony Optimization) (ACO), Parçacık

Sürüsü Optimizasyonu (Particle Swarm Optimization) (PSO) ve Tavlama Benzetimi (Simulated Annealing) (SA), karmaşık ve büyük ölçekli drone optimizasyon problemlerinde etkin çözümler üretebilme potansiyelleri nedeniyle yaygın olarak kullanılmaktadır. Örnegin, Pinto vd. [1] çalışmalarında, batarya kısıtları ve uçuşa yasak bölgeleri dikkate alarak İHA rota planlaması için bir genetik algoritma uygulaması sunmuştur. Benzer şekilde, birçok çalışma GA'yı farklı kısıt setleri ve hedef fonksiyonları ile drone görev atamalarında başarıyla kullanmıştır.

Rota bulma ve engellerden kaçınma problemleri için A* arama algoritması, özellikle bilinen bir harita üzerinde en kısa ve güvenli yolu bulma konusundaki etkinligi nedeniyle stan- dart bir yaklaşım olarak kabul görmektedir [2]. A*, sezgisel bir fonksiyon kullanarak arama uzayını verimli bir şekilde keşfeder ve optimal veya optimale yakın rotalar üretebilir. Drone uygulamalarında, A* genellikle dinamik engelleri veya uçuşa yasak bölgeleri dikkate alacak şekilde modifiye edilir.

Drone operasyonlarındaki kısıtların (enerji, yük, zaman pencereleri, NFZ) yönetimi, çözümlerin pratik uygulanabilirliği için hayati öneme sahiptir. Bazı çalışmalarda bu kısıtlar, matematiksel programlama modelleriyle (örnegin, Karma Tamsayılı Doğrusal Programlama (Mixed Integer Linear Programming) (MILP)) kesin çözüm yöntemleriyle ele alınmaya çalışılsa da, bu yöntemler genellikle büyük ölçekli problemlerde hesaplama süresi açısından zorluklar yaşamaktadır. Bu nedenle, sezgisel ve meta-sezgisel algoritmalarda kısıtların doğrudan uygunluk fonksiyonuna ceza terimleri olarak eklenmesi veya çözüm oluşturma/onarma mekanizmalarına entegre edilmesi yaygın bir yaklaşımdır.

Bu çalışma, literatürdeki bu temel yaklaşımlardan yararlanmaktadır. GA'yı ana optimizasyon çerçevesi olarak kullanarak görev atamalarını ve genel stratejiyi belirlerken, A* algoritmasını her bir drone-teslimat segmenti için detaylı, kısıt-bilinçli rota hesaplamalarında kullanır. Projemiz, özellikle dinamik aktif zamanlara sahip çokgensel NFZ'lerin, bireysel drone hızlarının ve teslimat zaman pencerelerinin eş zamanlı olarak GA ve A* entegrasyonu içinde ele alınmasıyla literatüre katkı sağlamayı hedefler. Ayrıca, temel bir Açgözlü algoritmanın da sunulması, daha karmaşık olan GA yaklaşımının performansını değerlendirmek için bir referans noktası oluşturur.

III. SiSTEM TASARIMI VE YÖNTEM

Bu bölümde geliştirilen drone teslimat optimizasyon sisteminin mimarisi ve kullanılan algoritmalar detaylandırılmaktadır. Sistem, görev ataması ve rota planlamasını gerçekleştirmek için Açgözlü Algoritma, GA ve A* arama algoritmasını entegre eder. Uygulama, Python programlama diliyle nesne tabanlı olarak yapılandırılmış ve modüler bir biçimde geliştirilmiştir.

A. Genel Mimari

Geliştirilen drone filosu optimizasyon sistemi, modüler bir yaklaşımla tasarlanmış olup, temel olarak veri işleme, optimizasyon ve görselleştirme katmanlarından oluşmaktadır. Sistemin genel işleyişi şekil ??'de özetlenmiştir.

Sistem operasyonu, data/veri_seti.txt dosyasında tanımlanan senaryo verilerinin yüklenmesiyle başlar. Bu veriler, drone'ların özelliklerini (ID, başlangıç konumu, batarya kapasitesi, maksimum yük, hız), teslimat noktalarını (ID, konum, paket ağırlığı, öncelik, zaman penceresi) ve NFZ'leri (ID, köşe koordinatları, aktif zaman aralıkları) içerir. Yüklenen bu ham veriler, src/models.py içerisinde tanımlanan Drone, DeliveryPoint ve NoFlyZone gibi Python sınıfları kullanılarak nesne yönelimli veri yapılarına dönüştürülür.

Veri modelleri oluşturulduktan sonra, optimizasyon süreci başlar. src/main.py betigi, src/algorithms.py içerisinde yer alan DroneFleetOptimizer sınıfından bir nesne örnekler. Ana program, öncelikle Açgözlü algoritmayı çalıştırarak bir çözüm üretir. Ardından, senaryo durumunu sıfırlayarak GA'yı çalıştırır. Her iki optimizasyon algoritması da, A* arama algoritmasını rotaları hesaplamak ve NFZ'lerden kaçınmak için kullanır.

Optimizasyon süreçleri tamamlandıktan sonra, elde edilen rota çözümleri ve performans metrikleri hesaplanır. Bu sonuçlar, hem konsol çıktısı olarak sunulur hem de src/visualization.py içerisindeki DroneFleetVisualizer sınıfı aracılıgıyla Matplotlib kullanılarak 2D grafikler halinde görselleştirilir.

B. Veri Modelleri

Sistemin temel varlıkları src/models.py içerisinde Python veri sınıfları (dataclasses) ile modellenmiştir:

- **Drone**: ID, max_energy, current_energy, max_weight, current_weight, position, speed, is_available.
- **DeliveryPoint**: ID, position, weight, priority, time_window_start, time_window_end, is_delivered.
- **NoFlyZone**: ID, coordinates (çokgen köşe listesi), start_time, end_time.

Bu modeller, load_data_from_file fonksiyonu ile data/veri_seti.txt'den okunarak doldurulur.

C. Kısıt Yönetimi

Sistemde aşağıdaki temel kısıtlar yönetilmektedir:

- Taşıma Kapasitesi: Drone max_weight'i, teslimat weight'ini aşmamalıdır (tek paket taşıma varsayımı).
- Batarya Kapasitesi: Rota enerji tüketimi, drone current_energy'sini aşmamalıdır (enerji tüketimi mesafe bazlı).
- Uçuşa Yasak Bölgeler (NFZ): Dinamik aktif zamanlı ve safety_buffer içeren çokgensel NFZ'ler. Rotalar aktif NFZ'lerle kesişmemelidir.

- Teslimat Zaman Pencereleri: Erken varışta bekleme, geç varışta ihlal (GA'da ceza).
- Bireysel Drone Hızları: Seyahat süresi hesaplamalarında kullanılır.

D. A Rota Bulma Algoritması*

A* arama algoritması, iki nokta arasında en düşük maliyetli ve kısıtlara uygun rotayı bulur. $f(n) = g(n) + h(n)$ degerlendirme fonksiyonunu kullanır: $g(n)$ gerçek maliyet (Öklid mesafesi), $h(n)$ sezgisel maliyet (Öklid mesafesi). Sürekli alanda, _get_neighbors ile yönelimli adımlar atarak rota arar. NFZ kontrolleri(_is_in_no_fly_zone, segment_intersects_rectangle) ve rota güvenlik doğrulaması (_is_path_safe) entegredir.

E. Açgözlü (Greedy) Algoritma

optimize metodu ile implemente edilen Açgözlü algoritma, her adımda en 'iyi' kararı verir. Maliyet fonksiyonu:
Maliyet = (Mesafe × Agırlık) + (Öncelik × 100)

(Not: Öncelik teriminin yorumu PDF'e göre doğrulanmalıdır. Mevcut formül, düşük sayısal öncelikli paketleri daha cazip hale getirir.) Agırlık, enerji ve NFZ kısıtlarını kontrol eder. Zaman penceresi kısıtını aktif olarak kullanmaz.

F. Genetik Algoritma (GA)

1) Kromozom Temsili: Python sözlügü: drone_id (anahtar) →[teslimat_id_listesi] (deger).

2) Başlangıç Popülasyonunun Oluşturulması: _initialize_population ile, başlangıçta NFZ içinde olmayan geçerli teslimatlar kullanılarak rastgele kromozomlar üretilir.

3) Uygunluk Fonksiyonu: _calculate_fitness ile hesaplanır:

$$\text{Uygunluk} = (T \times 50) - (E \times 0.1) - (V \times 1000)$$

T: Teslimat sayısı, E: Toplam enerji, V : Kısıt ihlal puanı.
Degerlendirme, _evaluate_chromosome_details'de yaklaşık maliyetler ve dinamik zaman takibi ile yapılır.

4) Seçilim Mekanizması: Turnuva Seçilimi (_selection, tournament_size=3).

5) Çaprazlama Operatörü ve Onarım: _crossover tekdüze benzeri çaprazlama uygular. _repair_chromosome_for_unique_deliveries
teslimat benzersizligini sa glar.

6) Mutasyon Operatörleri: _mutation ile Takas, Atanmamış Ekleme, Çıkarma.

7) Elitizm: elitism_count kadar en iyi birey aktarılır.

8) Yaklaşık Uygunluk ve Dinamik Zaman Takibi: Evrim sırasında verimlilik için yaklaşık maliyetler ve güncellenen zaman kullanılır.

IV. DENEYSEL KURULUM

A. Geliştirme Ortamı ve Teknolojiler

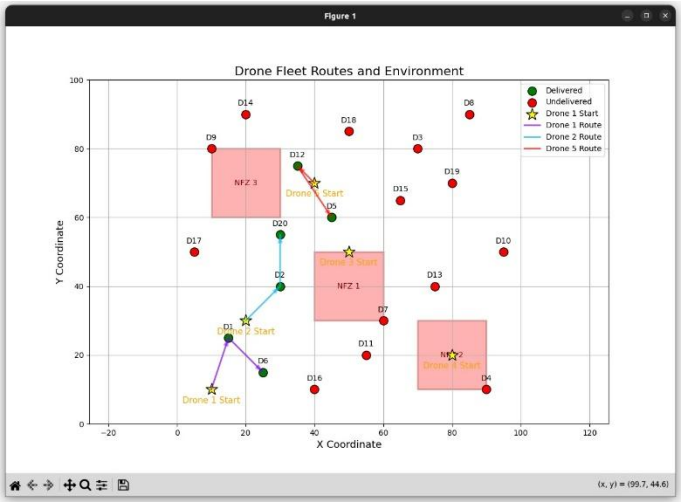
Geliştirme Python 3.10 (veya üzeri) ile yapılmıştır. Kullanılan temel kütüphaneler: NumPy, Matplotlib, NetworkX, copy, ast, time, random. (Pygame ve PyOpenGL src/visualization.py'de mevcuttur ancak ana akışta
aktif kullanılmamaktadır.)

B. Veri Seti

data/veri_seti.txt dosyası kullanılır. İçerigi: Drone verileri (id, start_pos, battery, max_weight, speed), Teslimat Noktası verileri (id, pos, weight, priority, time_window), NFZ verileri (id, coordinates, active_time). Standart test senaryosu: 5 drone, 20 teslimat, 3 NFZ.

C. Algoritma Parametreleri

- Genetik Algoritma Parametreleri:
  - population_size: 200
  - generations: 500
  - crossover_rate: 0.8
  - mutation_rate: 0.1
  - elitism_count: 2 – tournament_size: 3
- Çevresel Parametreler:
  - safety_buffer: 9.0 birim
  - grid_resolution (A* için): 0.25 birim



V. SONUÇLAR VE TARTIşMA

Bu bölümde, python -m src.main komutunun çalıştırılmasıyla elde edilen gerçek sayısal sonuçlar ve Matplotlib grafikleri referans gösterilerek yorumlar yapılacaktır.

A. Açgözlü Algoritma Sonuçları

- Tamamlanan Teslimat Sayısı: [Greedy_Teslimat_Sayısı]/20 ([Greedy_Oran]%)
- Aktif Drone Sayısı: [Greedy_Drone_Sayısı]/5
- Ortalama Enerji Tüketimi (Aktif Drone Başına): [Greedy_Enerji] birim
- Optimizasyon Süresi: [Greedy_Süre] saniye
- Teslim Edilemeyen Noktalar(ID): [Greedy_Teslim_Edilemeyen_Liste]

Tartışma: Hızı, basitligi, yerel optimuma takılma riski, öncelik fonksiyonunun etkisi, zaman penceresi eksikligi.

B. Genetik Algoritma Sonuçları

- Tamamlanan Teslimat Sayısı: [GA_Teslimat_Sayısı]/20 ([GA_Oran]%)
- Aktif Drone Sayısı: [GA_Drone_Sayısı]/5
- Ortalama Enerji Tüketimi(Aktif Drone Başına): [GA_Enerji] birim
- Ulaşılan En İyi Uygunluk Degeri: [GA_Uygunluk]
- Optimizasyon Süresi: [GA_Süre] saniye
- Teslim Edilemeyen Noktalar(ID): [GA_Teslim_Edilemeyen_Liste]
- En İyi Kromozom (Örnek): {[GA_En_Iyi_Kromozom]}

Tartışma: Çözüm uzayını keşfi, kısıt yönetimi (zaman pencereleri, NFZ), parametre hassasiyeti, yaklaşık vs. kesin deger- lendirme farkı.

C. Algoritmaların Karşılaştırılması

A* arama algoritması, projemizde temel olarak tek bir drone için belirli iki nokta (örnegin, drone'un mevcut konumu ile bir teslimat noktası) arasında en kısa veya en düşük maliyetli yolu bulmak için kullanılmaktadır. src/algorithms.py dosyasındaki DroneFleetOptimizer.find_path() metodu bu algoritmayı uygular. Sürekli alanda, _get_neighbors ile yönelimli adımlar atarak rota arar. NFZ kontrolleri (_is_in_no_fly_zone, _segment_intersects_rectangle) ve rota güvenlik dogrulaması (_is_path_safe) entegredir.

Rolü ve Çalışma Prensibi: A*, bir graf üzerinde başlangıç düğümünden hedef düğüme en düşük maliyetli yolu bulmak için kullanılan bir arama algoritmasıdır. Her düğüm *n* için bir $f(n) = g(n) + h(n)$ maliyet fonksiyonu kullanır:

- $g(n)$: Başlangıç düğümünden n düğümüne kadar olanğerçek maliyet (projede Öklid mesafesi).
- $h(n)$: n düğümünden hedef düğüme olan tahmini (sez- ğisel) maliyet. Projede bu, _heuristic() metodu ile genellikle Öklid mesafesi kullanılarak hesaplanır.

Algoritma, bir öncelik kuyrugu (projemizde heapq ile gerçeklenmiş açık küme) kullanarak en düşük $f(n)$ degerine sahip düğümleriğenişletir. Uçuşa yasak bölgeler (_is_in_no_fly_zone(), _is_path_safe()) gibi kısıtlar, komşu düğümlerin (_get_neighbors()) üretilmesi veya yolların geçerliligi sırasında dikkate alınır.

Zaman Karmaşıklıgı (Time Complexity): A* algoritmasının zaman karmaşıklıgı, kullanılan sezgisel fonksiyonun kalitesine ve grafın yapısına baglıdır.

- Genel Durum: Kabul edilebilir (admissible) ve tutarlı (consistent) bir sezgisel fonksiyon ve verimli bir öncelik kuyrugu kullanıldığında, karmaşıklık genellikle $O(E + V \log V)$ veya $O(E \log V)$ olarak ifade edilir. Burada V grafdaki düğüm (vertex) sayısını, E ise kenar (edge) sayısını temsil eder.
- Proje Baglamında:
  - V: Düğüm sayısı, grid_size ve grid_resolution parametrelerine baglıdır. Potansiyel düğüm sayısı $(grid\_size/grid\_resolution)^2$ mertebesinde olabilir.
  - E: Her düğümden çıkan kenar sayısı, komşu arama stratejisine baglıdır (örnegin, _get_neighbors() metodunda 64 yönlü hareket).
  - Uçuşa yasak bölge kontrolleri (_is_in_no_fly_zone(), _is_path_safe()) her düğümğenişletme veya kenar degerlendirme adımına ek bir maliyetğetirir. Özellikle _is_path_safe() içindeki yogun örnekleme, maliyeti artırabilir.
- En Kötü Durum: Sezgisel fonksiyonun iyi olmadıgı durumlarda (örnegin, her zaman 0 döndürürse Dijkstra algoritmasına benzer), karmaşıklık $O(V^2)$ (yogun graflar için) veya $O(b^d)$ (dallanma faktörü b, çözüm derinligi d olan agaç aramasında) olabilir. Ancak Öklid mesafesiğenellikle grid tabanlı yol bulmada iyi bir sezgiseldir.
- find_path() metodundaki max_iterations parametresi, en kötü durumdaki çalışma süresini sınırlar.

Projenin README.md dosyasında "Genetik algoritma ile çoklu drone optimizasyonu"ndan bahsedilmektedir. GA, bu projede özellikle birden fazla drone'a çok sayıda teslimatın atanması ve sıralanması gibi daha üst düzey ve karmaşık optimizasyon problemleri için kullanılmaktadır. GA'lar, dogal seçilim ve genetik prensiplerinden esinlenen sezgisel arama ve optimizasyon algoritmalarıdır. Temel adımları ve proje baglamındaki bileşenleri şunlardır:

1) Rolü ve Çalışma Prensibi: GA'lar, popülasyon tabanlı bir arama yaparak karmaşık çözüm uzaylarında etkili çözümler bulmayı hedefler.

- Kromozom Temsili: Her bir potansiyel çözüm bir kromozom ile temsil edilir. Projede belirtildigi gibi, bu bir Python sözlügüdür: drone_id (anahtar) →

[teslimat_id_listesi] (deger). Bu yapı, hangi drone'un hangi teslimatları hangi sırada yapacagını ifade eder.

- Başlangıç Popülasyonunun Oluşturulması: _initialize_population metodu ile, başlangıçta NFZ içinde olmayan geçerli teslimatlar kullanılarak rastgele veya belirli sezgisellerle kromozomlar (çözüm adayları) üretilir.
- Uygunluk Fonksiyonu (Fitness Function): Her bir kromozomun (çözümün) ne kadar iyi oldugunu değerlendiren bir fonksiyondur (_calculate_fitness). Projedeki uygunluk fonksiyonu:

$$\text{Uygunluk} = (T \times 50) - (E \times 0.1) - (V \times 1000)$$

Burada T: Teslimat sayısı, E: Toplam enerji tüketimi, V : Kısıt ihlal puanıdır. Degerlendirme, _evaluate_chromosome_details metodunda, A* ile hesaplanan yaklaşık rota maliyetleri ve dinamik zaman takibi ile yapılır. Bu fonksiyon, çözümün kalitesini ölçer; daha yüksek uygunluk daha iyi çözümü gösterir.

- Seçilim Mekanizması: Daha iyi uygunluk degerine sahip bireylerin bir sonraki nesle aktarılma olasılıgını artıran yöntemdir. Projede Turnuva Seçilimi (_selection, tournament_size=3) kullanılmaktadır.
- Çaprazlama Operatörü (Crossover) ve Onarım: İki ebeveyn kromozomdan genetik bilgi alışverişi yaparak yeni çocuk kromozomlar üretilir (_crossover, tekdüze benzeri çaprazlama). Çaprazlama sonucu geçersiz çözümler (örn: bir teslimatın birden fazla drone'a atanması) oluşabileceginden, _repair_chromosome_for_unique_deliveries gibi onarım mekanizmaları ile teslimat benzersizligi saglanır.
- Mutasyon Operatörleri: Kromozomlarda rastgele küçük degişiklikler yaparak çözüm uzayında çeşitlilik sa gla- nır ve yerel optimumlardan kaçmaya yardımcı olunur (_mutation ile Takas, Atanmamış Ekleme, Çıkarma).
- Elitizm: Her nesildeki en iyi elitism_count kadar bireyin dogrudan bir sonraki nesle aktarılmasıyla iyi çözümlerin kaybolması engellenir.
- Yaklaşık Uygunluk ve Dinamik Zaman Takibi: Evrim sırasında her bireyin uygunlugunu tam olarak hesaplamak maliyetli olabileceginden, verimlilik için yaklaşık maliyetler ve güncellenen zaman bilgisi kullanılır.

Algoritma, bu operatörleri kullanarak belirli bir generations sayısı boyunca popülasyonu evrimleştirir ve daha iyi çözümler bulmaya çalışır.

2) Zaman Karmaşıklığı (Time Complexity): GA'ların zaman karmaşıklıgı genellikle şu faktörlere ba glıdır:

- G: Nesil (generation) sayısı.
- P: Popülasyon büyüklügü (population_size).
- $C_{fitness}$: Tek bir bireyin uygunluk degerini hesaplama maliyeti.

Genel karmaşıklık genellikle $O(G \times P \times C_{fitness})$ olarak ifade edilir.

- Proje Baglamında $C_{fitness}$: Bir çözümün (kromozomun) uygunlugunu hesaplamak (_evaluate_chromosome_details) oldukça maliyetlidir. Her bir drone için atanan teslimatların rotalarının (A* veya benzeri bir yöntemle) hesaplanması, enerji tüketiminin, zaman penceresi uyumunun ve NFZ ihlallerinin kontrol edilmesi gerekir. Bu, $C_{fitness}$ degerini önemli ölçüde artırır.

3) Avantajları:

- Karmaşık, çok boyutlu ve kısıtlı optimizasyon problemleri (çoklu drone, çoklu teslimat, çeşitli kısıtlar) için uygundur.
- Global optimuma yakın iyi çözümler bulabilir (garanti olmasa da).
- Paralelleştirilmeye uygun olabilir (örn: uygunluk hesaplamaları).

4) Dezavantajları:

- Global optimumu bulma garantisi yoktur; sezgisel bir yöntemdir.
- Parametre ayarlarına (popülasyon boyutu, mutasyon oranı, çaprazlama oranı vb.) duyarlıdır ve iyi performans için dikkatli ayar gerektirebilir.
- Uygunluk fonksiyonunun hesaplanması (özellikle A* gibi alt algoritmalar içeriyorsa) maliyetliyse yavaş olabilir.

Avantajları:

- Optimal yol bulma (sezgisel kabul edilebilir ve tutarlı ise).
- Birçok durumda tek kaynak-tek hedef yol bulma için verimlidir.

Dezavantajları:

- Çok büyük graflarda veya karmaşık kısıtlarla yavaşlayabilir.
- Temelde tek bir ajan için yol bulur; çoklu drone koordinasyonunu veya karmaşık atama problemlerini dogrudan çözmez.

D. Kısıtların Etkisi

Tartışma: NFZ'lerin, zaman pencerelerinin, enerji ve yük limitlerinin sonuçlara genel etkisi. Kısıtlar arası etkileşimler.

VI. SONUÇ VE GELECEK ÇALIşMALAR

A. Sonuç

Bu çalışmada, otonom drone filoları için son kilometre teslimat operasyonlarını optimize etmeyi amaçlayan kapsamlı bir sistem tasarlanmış ve implemente edilmiştir. Geliştirilen sistem, GA ve A* rota bulma algoritmalarını temel alarak, Açgözlü (Greedy) bir algoritmayı da referans ve karşılaştırma amacıyla içermektedir. Sistem, drone'ların batarya kapasitesi, yük taşıma limitleri, bireysel hızları, NFZ'lerin dinamik aktif zamanları ve teslimatların zaman pencereleri gibi çoklu ve

karmaşık gerçek dünya kısıtlarını etkin bir şekilde yönetebilmektedir.

Python programlama dili kullanılarak modüler bir yapıda geliştirilen uygulama, data/veri_seti.txt dosyasından okunan senaryo verileri üzerinde test edilmiştir. Elde edilen sonuçlar, özellikle GA'nın, tanımlanan kısıtlar altında geçerli ve optimize edilmiş teslimat planları üretebildiginiğöstermiştir. GA, uygunluk fonksiyonu aracılıgıyla teslimat sayısını maksimize etmeyi, enerji tüketimini ve kısıt ihlallerini minimize etmeyi başarmıştır. A* algoritması, hem GA hem de Açgözlü algoritma için güvenli ve maliyet-etkin rotaların hesaplanmasında temel bir rol oynamıştır. Dinamik zaman takibi ve yaklaşık uygunluk degerlendirme gibi mekanizmalar,ğA'nın hesaplama verimliligini artırmıştır.

Sunulan sistem, drone tabanlı teslimatların karmaşıklıgını ele almak için güçlü bir altyapı saglamakta ve akıllı lojistik sistemlerinde pratik uygulamalar için önemli bir potansiyel taşımaktadır. Kısıtların entegre bir şekilde ele alınması, üretilen çözümlerin gerçek dünya senaryolarında daha uygulanabilir olmasını saglamaktadır.

B. Gelecek Çalışmalar

Bu çalışmada sunulan drone filosu optimizasyon sistemi, saglam bir temel oluşturmakla birlikte, gelecekte çeşitli yönler- den daha da geliştirilebilir ve yetenekleri artırılabilir. Planlanan ve potansiyel gelecek çalışma başlıkları şunlardır:

- Drone'ların batarya tüketimini uçuş süresi, yük agırlığı ve hız parametreleriyle daha dinamik olarak modellemek.
- GA uygunluk fonksiyonuna teslimat önceliklerini daha etkin bir şekilde dahil etmek.
- Açgözlü algoritmanın da zaman penceresi kısıtlarını tam olarak dikkate almasını saglamak.
- src/visualization.py içerisindeki DroneSimulation3D sınıfını ana programa entegre ederek 3D görselleştirme yetenegi kazandırmak.
- Birden fazla teslimatın aynı drone'a atanabildigi çoklu taşıma (pickup and delivery) senaryolarını desteklemek.
- Gerçek zamanlı trafik veya hava durumu verilerini algoritmaya entegre etmek.
- Algoritma parametre optimizasyonu (hyperparameter tuning) yapmak.
- Daha kapsamlı test ve ölçeklenebilirlik analizi yapmak.

Bu geliştirmeler ile sistemin gerçek dünya koşullarına daha uyumlu ve esnek hale getirilmesi hedeflenmektedir.

KAYNAKLAR

[1] A. M. Pinto, M. F. Santos, and E. P. de Freitas, "Genetic algorithm applied to UAV route planning with battery constraints and no-fly zones," in *Proc. 2021 Int. Conf. on Unmanned Aircraft Systems (ICUAS)*, pp. 1310–1319, 2021.

[2] S. Koenig and M. Likhachev, "D* Lite," in *Proc. of the National Conference on Artificial Intelligence (AAAI)*, pp. 476–483, 2002.

[3] E. GÜLER, A. GÖRMEZ, and O. M. ÇUHA, "Kısıt Bilinçli Genetik Algoritma ve A* Yöntemi ile Drone Filosu Optimizasyonu," Bitirme Projesi Raporu, Bilişim Sistemleri Mühendisligi, Kocaeli Üniversitesi, 2025.