Düzce Üniversitesi Bilim ve Teknoloji Dergisi

Araştırma Makalesi

Güncel Metasezgisel Algoritmalarının Performansları Üzerine Karşılaştırılmalı Bir Çalışma

Sibel ARSLAN [a,*]

[a] *Yazılım Mühendisliği Bölümü, Teknoloji Fakültesi, Sivas Cumhuriyet Üniversitesi, Sivas, TÜRKİYE*
** Sorumlu yazarın e-posta adresi: sibelarslan@cumhuriyet.edu.tr*
DOI: 10.29130/dubited.1150453

ÖZ

Günümüzde, metasezgiseller optimizasyon problemlerinin çözümünde çok önemli bir rol oynamaktadır. Bu çalışmada sürü zekasından ve doğadaki canlıların yiyecek arama davranışlarından esinlenerek geliştirilen üç yeni metasezgisel (Afrika Akbabaları Optimizasyon Algoritması (African Vulture Optimization Algorithm, AVOA), Geliştirilmiş Gri Kurt Optimizasyon Algoritması (Improved Gray Wolf Optimization Algorithm, I-GWO) ve Deniz Avcıları Algoritması (Marine Predators Algorithm, MPA)), kıyaslamalarda en çok kullanılan metasezgisellerden biri olan Parçacık Sürü Optimizasyon Algoritması (Particle Swarm Optimization, PSO) ile kıyaslanmıştır. Deneysel çalışmalara göre, AVOA ve MPA'nın diğer algoritmalara göre daha başarılı sonuçlara sahip olduğu görülmektedir. Sonuçların istatiksel anlamlılığı, Friedman ve Wilcoxon işaretli sıralar testleri ile değerlendirilerek bu iki algoritmanın üstünlüğü kanıtlanmıştır.

Anahtar Kelimeler: *Afrika Akbabaları Optimizasyon Algoritması, Geliştirilmiş Gri Kurt Optimizasyon Algoritması, Deniz Avcıları Algoritması, Parçacık Sürü Optimizasyon Algoritması*

Comparison of Current Metaheuristic Algorithms with Different Performance Criteria

ABSTRACT

Nowadays, metaheuristics play a very important role in solving optimization problems. In this study, Particle Swarm Optimization Algorithm (PSO), one of the most commonly used metaheuristics, was compared in three new metaheuristic (African Vulture Optimization Algorithm-AVOA, Improved Gray Wolf Optimization Algorithm- I-GWO and Marine Predators Algorithm-MPA) comparisons inspired by swarm intelligence and foraging behavior of creatures in nature. According to the experimental studies, AVOA and MPA achieve more successful results than other algorithms. The statistical significance of the results was evaluated using the Friedman Wilcoxon signed-rank test, and the significant superiority of these two algorithms was proven.

Keywords: *African Vulture Optimization Algorithm, Improved Gray Wolf Optimization Algorithm, Marine Predators Algorithm, Particle Swarm Optimization*

Geliş: 28/07/2022, Düzeltme: 22/10/2022, Kabul: 29/11/2022

I. GİRİŞ

Birçok mühendislik problemlerinin karmaşıklığı ve matematiksel yöntemlerin optimal çözümü sağlayamaması nedeniyle bu problemleri çözmek için metasezgisel algoritmaların kullanımı önemli ölçüde artmıştır. Araştırmacılar doğa olaylarından ve canlıların davranışlarından esinlenerek farklı algoritmalar önermişlerdir. Bu algoritmalar sezgisel ve metasezgisel algoritmalar olmak üzere iki ana gruba ayrılır. Sezgisel algoritmalar, yerel optimum noktalarına takılabildiği ve belirli optimizasyon problemlerini çözdükleri için daha az ilgi görmüştür [1-3]. Metasezgisel algoritmalar, farklı zorlu ve karmaşık optimizasyon problemlerine optimal çözümler ya da makul olarak kabul edilebilir çözümler bulabilmektedir [4]. Bununla birlikte optimal çözüm bulmayı garanti etmez. Bu algoritmalar çözümleri bulabilmek için arama ve sömürü fazı olmak üzere iki temel fazdan yararlanır. Arama fazı, algoritma başlangıcında yeni çözümler üretmek için kullanılır. İterasyonlar boyunca üretilen bu çözümlerin uygunluk değerlerinin iyileştirilmesi için kademeli olarak sömürü fazına geçilir. Çalışmalar, önerilen metasezgisel algoritmaların doğadan esin kaynağı ve arama mekanizmalarındaki farklılıklara rağmen, arama ve sömürü fazlarını dengelemeye amaçladığını göstermektedir [5,6].

Bu çalışmada son zamanlarda önerilen üç farklı metasezgiselin (Afrika Akbabaları Optimizasyon Algoritması (African Vultures Optimizer Algorithm, AVOA), Geliştirilmiş Gri Kurt Optimizasyon Algoritması (Improved Grey Wolf Optimizer, I-GWO) ve Deniz Avcıları Algoritması (Marine Predators Algorithm, MPA) temel bir metasezgisel olan Parçacık Sürü Optimizasyon algoritmasıyla (Particle Swarm Optimization, PSO) karşılaştırılmıştır. Çalışmanın temel katkıları aşağıda sıralanmıştır.

- Literatür taramasına göre bu üç metasezgisel benzetim sonuçları yakınsama hızları gibi farklı değerlendirme kriterleri ilk kez karşılaştırılmıştır.
- Algoritmaların benzetim sonuçlarına göre en başarılı düşünülen iki metasezgiselin diğerleriyle başarısının kıyaslanması için tek kuyruk Wilcoxon istatiksel test yapılmıştır.
- Deneysel çalışmalara göre AVOA ve MPA'nın diğer metasezgisellere üstünlük sağladığı değerlendirilmiştir.

Çalışmanın geri kalanında karşılaştırılacak metasezgiseller hakkında detaylı bilgi ve literatür taramasına Bölüm 2'de yer verilmiştir. Bölüm 3'te kıyaslama yapılan test fonksiyonları, parametreler, benzetim sonuçları ve istatiksel test sonuçlarının verildiği deneysel çalışmalar sunulmuştur. Bölüm 4'te çalışma özetlenerek genel değerlendirme yapılmıştır.

II. KARŞILAŞTIRILACAK METASEZGİSEL ALGORİTMALAR

Bu bölümde kıyaslama yapılacak metasezgisel algoritmalar hakkında detaylı bilgiye yer verilmiştir.

A. PARÇACIK SÜRÜ OPTİMİZASYONU

Parçacık sürüsü optimizasyonu algoritması, bir sürü içindeki kuşların sosyal davranışlarına dayanan bir metasezgiseldir. Kennedy ve Eberhart [7] tarafından 1995 yılında önerilen bu algoritma çeşitli optimizasyon problemlerini çözmek için sıklıkla kullanılan güçlü bir araç haline gelmiştir. PSO, her bir parçacığı bir çözümü temsil eden rastgele bir parçacık sürüsünün oluşturulmasına dayanır. Her çözümün değerlendirilmesine bir uygunluk değeri ile karar verilir. Algoritma, en kötü uygunluk değerlerine sahip parçacıkların/çözümlerin en iyi olanlara durdurma kriteri/kriterleri sağlanıncaya kadar çekilmesine dayanır. Erken yakınsamayı önlemek için, tüm parçacıklar şimdiye kadar elde ettikleri en iyi uygunluk değerinin hafızasını içerir ve bu bilgi aynı zamanda daha iyi bir genel çözüm arayışında parçacıkların

konumunu güncellemek için kullanılır. Algoritmanın sözde kodu Tablo 1'de paylaşılmıştır. Daha detaylı bilgi için [7]'ye bakılabilir.

***Tablo 1**. Standart PSO sözde kodu*

Algoritma 1: Standart PSO	
Adım 1.	Başlangıç popülasyonundaki parçacıkların hızlarını ve konumlarını rastgele oluştur.
Adım 2.	Her bir parçacığın uygunluk değerini hesapla.
Adım 3.	Her bir parçacık için yerel en iyiyi (***pbest***) değerini bul.
Adım 4.	Tüm parçacıklar için global en iyiyi (***gbest***) değerini bul.
Adım 5.	Parçacıkların hız ve konum değerlerini ***pbest*** ve ***gbest***'e göre güncelle.
Adım 6.	Durdurma kriteri sağlanmıyorsa Adım 2' ye git.

PSO ve varyantları, uygulanması kolay olduğu için literatürde stabilite analizi [8], öznitelik seçimi [9], ısı ve güç dağıtımı [10], yörünge planlama [11] ve büyük ölçekli optimizasyon [12] gibi birçok problemi çözmek için kullanılmıştır. Cleghorn ve Stapelberg fark vektörlerine dayanan yeni teoremler üreterek dört popüler PSO varyantı için birinci ve ikinci dereceden kararlılık kriterleri elde ettiler [8]. Böylece PSO varyantlarında kontrol katsayıları arasındaki ilişkide kısıtlama olmaksızın kararlılık kriterleri türetilmiştir. Hu ve ark. büyük ölçekli verilerin işlenmesinde öznitelik seçiminin yüksek karmaşıklığı için MS-destekli DBPSO olarak adlandırdıkları, çoklu vekil destekli ikili parçacık sürü optimizasyonu önerdi [9]. Algoritma PSO'nun yakınsama yeteneğini hızlandırmış ve büyük ölçekli öznitelik seçim problemlerinin çalışma süresini önemli derece azaltmıştır. Chen ve Li, valf noktası etkileri ile çok yakıtlı birleşik ısı ve güç ekonomik dağıtım problemini (Multi-fuel combined heat and power economic dispatch, MF-CHPED) çözmek için yeni bir toplu bilgi tabanlı PSO (collective information-based particle swarm optimization, CIB PSO) algoritması önerdiler [10]. Algoritmada, en iyi çözümlerin doğrusal kombinasyonu tarafından oluşturulan toplu bilgi tabanlı en iyi konumları kullanarak parçacık arama ve toplu bilgi tabanlı elit adı verilen iki yeni strateji geliştirildi. Bu stratejilerle PSO'nun arama verimliliği arttırılarak MF-CHPED problemleri başarıyla çözülmüştür. Fernandes ve ark. mobil robotik araçların statik ve dinamik ortamlardaki yörünge planlama görevi için yeni bir kuantum davranışlı parçacık sürü optimizasyonu (Quantum-behaved particle swarm optimization, QPSO) algoritması önerdi [11]. Algoritma ile çeşitli tepe noktalarına sahip bu problem için yerel minimumlardan etkili bir şekilde kaçınarak etkili bir şekilde yakınsama amaçlanmıştır. Deney sonuçlarına göre, yörünge planlamasında algoritma klasik bazı algoritmalarla karşılaştırıldığında, otonom robotik araçlar için güvenlik ve enerji tasarrufuna sahip optimal rotalar elde edebilmiştir. Wang ve ark. geleneksel PSO'nun arama yeteneğini geliştirmek ve erken yakınsamadan kaçınmak için seviye tabanlı popülasyon yapısı ve seviye sayısının kontrolünü sağlayan bir takviyeli öğrenme stratejisi önermişlerdir [12]. Yazarlar bu stratejiyi, takviyeli öğrenme seviyeli tabanlı parçacık sürüsü optimizasyon algoritması (Reinforcement Learning Level based Particle Swarm Optimization algorithm, RLLPSO) olarak adlandırdılar. Deneysel sonuçlar algoritmanın, büyük ölçekli optimizasyon problemlerinde birçok geleneksel algoritmaya karşı üstünlük sağladığını göstermiştir.

B. AFRİKA AKBABALARI OPTİMİZASYON ALGORİTMASI

Abdollahzadeh ve ark. [13] Afrika akbabalarının yer bulma ve yiyecek arama davranışlarından esinlenerek Afrika Akbabaları Optimizasyon algoritmasını önermiştir. Doğal yaşamda akbabalar iki temel gruba ayrılır. Her bir akbaba bir çözümü temsil eder. Algoritmada ilk olarak akbabaları gruplara ayırmak için tüm çözümlerin uygunluk değeri hesaplanır. En iyi değere sahip ilk akbaba birinci grubun ilk ve en iyi akbabası; en iyi değere sahip ikinci akbaba ise ikinci grubun en iyi ve ilk akbabası olarak kabul edilir. Popülasyondaki diğer akbabalar bu en iyi iki akbabadan birini hareket ettirmek veya değiştirmek için kullanılır. Akbabalar yiyecek bulma ve grup halinde yaşayabilmek için iki temel gruba ayrılır. Her grubun yiyecek arama ve yeme yeteneği farklıdır. Akbabaların yemek yeme eğilimleri ve saatlerce yiyecek aramaları onların aşırı/gereksiz yemek yemelerini engellemektedir. Algoritmada popülasyondaki en kötü çözümün en zayıf ve en aç akbaba olduğunu varsayarak, akbabalar en kötü akbabadan uzak durmaya ve en iyi akbabaya yakınlaşmaya çalışırlar. AVOA'da gruptaki en iyi

akbabanın belirlenmesi, akbabaların açlık oranının hesaplanması, arama ve sömürü aşamaları olmak üzere dört temel faz izlenir.

B. 1. Birinci Faz: Herhangi Bir Gruptaki En İyi Akbabayı Belirlemek

Başlangıç popülasyonu oluşturulduktan sonra tüm çözümlerin uygunluk değeri hesaplanır ve en iyi çözüm birinci grubun en iyi akbabası, ikinci en iyi çözüm de ikinci grubun en iyi akbabası olarak seçilir. Diğer çözümler, birinci ve ikinci gruplar için en iyi çözümlere doğru ilerler. Her iterasyonda, tüm popülasyon uygunluk değerlerine göre yeniden konumlandırılır.

B. 2. İkinci Faz: Akbabaların Açlık Oranının Hesaplanması

Akbabalar genellikle yiyecek arama eğilimindedir ve tok olduklarında yüksek enerjiye sahiptirler. Dolayısıyla akbabalar tok olduğunda yiyecek aramak için daha uzun mesafelere gidebilirler, ancak açlarsa uzun süre uçmak ve daha güçlü akbabanın yanında yiyecek aramak için yeterli enerjileri bulunmamaktadır. Bu aşamada, azalma eğiliminde olan açlık oranı Eşt. 1 ve 2 ile modellenmeye çalışılmıştır.

$$t = h \times \left(\sin^{w} \left(\frac{\pi}{2} \times \frac{\text{iter}}{\text{maks\_iter}} \right) + \cos \left(\frac{\pi}{2} \times \frac{\text{iter}}{\text{maks\_iter}} \right) - 1 \right) \quad (1)$$

$$F = (2rand_1 + 1) \times z \times \left(1 - \frac{\text{iter}}{\text{maks\_iter}} \right) + t \quad (2)$$

Burada F akbabaların açlık oranını gösterir, iter mevcut iterasyon sayısını, maks_iter maksimum iterasyon sayısını belirtir. z, her iterasyonda değiştiren -1 ile 1 arasında rastgele bir sayıdır. h -2 ile 2 arasında; $rand_1$ 0 ile 1 arasında rastgele bir sayıdır. z değeri 0'ın altında değere sahipse akbabanın aç kaldığı, 0'dan yüksek değere sahipse akbabanın doyduğu anlamına gelir. $|F|$ değeri 1'den büyük eşit ise, akbabalar farklı alanlarda yiyecek arar ve AVOA keşif fazına girer. Diğer durumda AVOA sömürü aşamasına girer ve akbabalar, komşu çözümlerin yakınında yiyecek arar.

B. 3. Üçüncü Faz: Arama

Bu fazda akbabalar, iki farklı stratejiyi seçerek farklı rastgele alanları arar. Bu stratejileri seçmek için p_1 adı verilen bir parametre kullanılır. Parametre, 0 ile 1 arasında bir değere sahip olmalıdır ve arama işleminden önce değerlendirilmelidir. Stratejilerin seçilmesinde Eşt. 3'ü kullanılır:

$$P_i(t+1) = \begin{cases} R(i) - |X \times R(i) - P(i)| \times F & p_1 \geq rand_{P_1} \\ R(i) - F + rand_2 \times \left((u_b - l_b) \times rand_3 + l_b \right) & p_1 < rand_{P_1} \end{cases} \quad (3)$$

Burada $rand_{p1}$ arama fazında stratejilerden herhangi birini seçmek için 0 ile 1 arasında rastgele bir sayıdır. $R(i)$ en iyi akbabalardan biridir, X akbabaların yiyeceği diğerlerinden korumak için hareket ettiği mesafeyi, $rand_2$ ve $rand_3$ [0, 1] aralığında rastgele sayıları temsil eder. u_b ve l_b arama uzayının üst ve alt sınırlarını ifade eder. Eğer $rand_3$ değeri 1'e yakınsarsa, çeşitlilik ve farklı uzay alanlarını arama yeteneği artar.

B. 4. Dördüncü Faz: Sömürü

$|F|$ değeri ise 1'den küçük olduğu durumda algoritma sömürü fazına geçer. Öncelikle $|F|$ değerinin 0,5'den büyük olup olmadığına göre faz iki ayrı seçime ayrılır. Eğer bu değer 0,5'den küçükse sömürü değilse yiyecek için akbabalar arasında rekabet olduğu varsayılmıştır. Her bir seçim için rastgele üretilen değerler ile iki farklı strateji seçilir. Bu stratejilerin nasıl seçileceği p_2 ve p_3 parametreleri ile belirlenir. Rekabet aşamasında akbabalar yiyecek arama için yeterli enerjiye sahip olup yiyecek kaynakları

üzerinde çatışmaya girebilirler. Zayıf akbabalar güçlü olanlardan yiyecek almaya çalışırlar ya da spiral düzende uçarlar. Bu davranış p_2 parametresine bağlı olarak Eşt. 4 ile modellenmiştir.

$$P_i(t+1) = \begin{cases} |X \times R(i) - P(i)| \times (F + rand_4) - \big(R(i) - P(i)\big) \; eğer \; p_2 \geq rand_{P_2} \\ R(i) - R(i) \times \left(\frac{P(i)}{2\pi}\right)\big(rand_5 \times \cos\big(P(i)\big) + rand_6 \times \sin\big(P(i)\big)\big) \; eğer \; p_2 < rand_{P_2} \end{cases} \quad (4)$$

Burada $P(i)$, akbaba ile iki gruptaki en iyi akbabalardan biri arasındaki mesafenin elde edildiği akbabanın mevcut vektör konumudur. $R(i)$, mevcut iterasyondaki en iyi iki akbabadan birinin konum vektörünü temsil eder. $rand_4$, $rand_5$ ve $rand_6$, 0 ile 1 arasında rastgele üretilen bir sayıdır.

Bu fazın ikinci aşamasında, iki akbabanın hareketi, yiyecek kaynağı üzerinde birkaç tür akbaba toplanır ve yiyecek bulmak için akbabalar arasında saldırgan çekişmeler yaşanır. Eğer $|F|$ sayı 0,5'ten küçükse, fazın bu aşaması yürütülür. Bu aşamanın başlangıcında, 0 ile 1 arasında rastgele bir sayı olan $rand_{P_3}$ üretilir. $rand_{P_3}$, p_3 parametresinden büyük veya eşitse, bu fazda, besin kaynağı üzerinde birkaç tür akbaba toplanmaktadır. Diğer durumda, akbabalar arasında kuşatma savaşı gerçekleşir. Bu aşamanın matematiksel modellenmesi Eşt. 5'de gösterilmiştir.

$$P(i+1) = \begin{cases} Eşt.7 \; eğer \;\; p_3 \geq rand_{P_3} \\ Eşt.8 \; eğer \;\; p_3 < rand_{P_3} \end{cases} \quad (5)$$

Dolayısıyla akbabaların yiyecek kaynağı üzerinde toplanma hareketi Eşt. 7 Eşt. 6'ya bağlı olarak aşağıdaki gibi ifade edilebilir.

$$\begin{aligned} A_1 &= \text{EnİyiAkbaba}_1(i) - \frac{\text{EnİyiAkbaba}_1(i) \times P(i)}{\text{EniyiAkbaba}_1(i) - P(i)^2} \times F \\ A_2 &= \text{EnİyiAkbaba}_2(i) - \frac{\text{EnİyiAkbaba}_2(i) \times P(i)}{\text{EnİyiAkbaba}_2(i) - P(i)^2} \times F \end{aligned} \quad (6)$$

$$P(i+1) = \frac{A_1 + A_2}{2} \quad (7)$$

Burada, güncel iterasyonda $\text{EnİyiAkbaba}_1(i)$ birinci grubun en iyi akbabası, $\text{EnİyiAkbaba}_2(i)$ ikinci grubun en iyi akbabası, F akbabaların açlık oranını, $P(i)$ akbabanın şimdiki konumunu ve $P(i+1)$ akbabanın bir sonraki iterasyondaki konumunu belirtir.

$|F| < 0{,}5$ olduğu durumda grupların lider akbabaları aç kalır ve gruptaki diğer akbabalarla başa çıkmak için yeterli enerjiye sahip olmaz. Bununla birlikte, diğer akbabalar da yiyecek arayışlarında saldırganlaşırlar. Lider akbabalar doğru farklı yönlerde hareket ederler. Eşt. 8 bu hareketi modellemek için kullanılır.

$$P(i+1) = R(i) - |R(i) - P(i)| \times F \times \text{LF}(d) \quad (8)$$

Burada d problem boyutunu, $|R(i) - P(i)|$ akbabanın iki grubun en iyi akbabalarından birine olan mesafesini temsil eder. AVOA'nın etkinliği arttırmak için Levy uçuşuna (Levy Flight, LF) da yer verilmiştir. Bu uçuşun modellenmesi Eşt. 9'daki gibidir.

$$LF(x) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma(\frac{(1+\beta)}{2}) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \quad (9)$$

Burada u ve v 0 ve 1 arasında rastgele bir sayı ve β varsayılan değeri 1,5 olarak tanımlanan sabit bir sayıdır. AVOA algoritmasının sözde kodu Tablo 2'de verilmiştir.

Tablo 2. *AVOA sözde kodu*

Algoritma 2: AVOA

Adım 1. Girdiler: $\boldsymbol{N}$ popülasyon büyüklüğü ve maksimum iterasyon sayısı $\boldsymbol{T}$
Adım 2. Çıktılar: Akbabanın konumu ve uygunluk değeri
Adım 3. Başlangıç popülasyonunu $\boldsymbol{P_i(i = 1, 2, \dots, N)}$ rastgele oluştur.
Adım 4. **döngü** (durdurma kriteri sağlanıncaya kadar) **yap**
Adım 5. Akbabaların uygunluk değerlerini hesapla
Adım 6. $\boldsymbol{P_{EniyiAkbaba_1}}$ 'i akbabanın konumu olarak ayarla (İlk gruptaki en iyi akbaba)
Adım 7. $\boldsymbol{P_{EniyiAkbaba_2}}$ 'i akbabanın konumu olarak ayarla (İkinci gruptaki en iyi akbaba)
Adım 8. **döngü** (Her akbaba ($\boldsymbol{P_i}$) için)
Adım 9. $\boldsymbol{R(i)}$'yi seç
Adım 10. $\boldsymbol{F}$'i güncelle
Adım 11. **eğer** ($\boldsymbol{|F| \geq 1}$) **ise**
Adım 12. **eğer** ($\boldsymbol{P_1 \geq rand_{P1}}$) **ise**
Adım 13. Eşt. 3'ü kullanarak akbabanın konumunu güncelle
Adım 14. **değilse**
Adım 15. Eşt. 3'ü kullanarak (ikinci bölüm) akbabanın konumunu güncelle
Adım 16. **eğer** ($\boldsymbol{|F| < 1}$) **ise**
Adım 17. **eğer** ($\boldsymbol{|F| \geq 0,5}$) **ise**
Adım 18. **eğer** ($\boldsymbol{P_1 \geq rand_{P2}}$) **ise**
Adım 19. Eşt. 4'ü kullanarak akbabanın konumunu güncelle
Adım 20. **değilse**
Adım 21. Eşt. 4'ü kullanarak (ikinci bölüm) akbabanın konumunu güncelle
Adım 22. **değilse**
Adım 23. **eğer** ($\boldsymbol{P_1 \geq rand_{P2}}$)
Adım 24. Eşt. 5'i kullanarak akbabanın konumunu güncelle
Adım 25. **değilse**
Adım 26. Eşt. 5'i kullanarak (ikinci bölüm) akbabanın konumunu güncelle
Adım 27. Durdurma kriteri sağlanmıyorsa Adım 4' e git.
Adım 28. **Dön** $\boldsymbol{P_{EniyiAkbaba_1}}$

AVOA, 2021 yılında önerilen yeni bir algoritma olmasına rağmen literatürde parametre tanımlama [14-16], sistem yapılandırma [17] ve tasarımı [18] problemlerinin çözümünde kullanılmıştır. Bagal ve ark. Katı Oksit Yakıt Pili (Solid Oxide Fuel Cell, SOFC) modellerinde ampirik, gerçek gerilim ve akım profilleri arasındaki hatayı minimize etmek için Modifiye Edilmiş AVOA (Modified African Vulture Optimizer, MAVO) algoritmasını önerdiler [14]. Wang ve ark. ise aynı problemi AVOA'nın bir başka varyantını (Adaptive African Vulture Optimization, AAVO) önererek çözdüler [15]. Chen ve Zhang bir başka model olan Proton değişim membranlı yakıt hücresinin (Proton Exchange Membrane Fuel Cell, PEMFC) parametreleri tanımlamak için AVOA'nın geliştirilmiş bir varyantını (Improved version of African Vulture Optimizer, IAVOA önerdiler [16]. Her üç çalışmadaki deneysel sonuçlar algoritmaların yakınsamada ve doğrulukta oldukça başarılı sonuçlar ürettiğini göstermektedir. Alanazi ve ark. fotovoltaik sistem dizisinin çıkış gücünü yeniden yapılandırmak için AVOA'yı kullandılar [17]. Çalışmadaki sonuçlar çıkış gücünün başarıyla maksimize edildiğini ifade etmektedir. Wang ve ark. Hibrit Yenilenebilir Enerji Sistemlerinde (Hybrid Renewable Energy System, HRES) maliyeti minimize edecek sistem elementlerini önerdikleri geliştirilmiş AVOA ile tasarlamayı amaçladılar [18]. Çalışmada algoritmanın etkinliği birçok çalışmayla kıyaslanarak ifade edilmiştir.

C. GELİŞTİRİLMİŞ GRİ KURT OPTİMİZASYON ALGORİTMASI

Geliştirilmiş Gri Kurt Optimizasyon Algoritması algoritmasının bir varyantı olduğu için öncelikle kısaca GWO açıklanarak, daha sonra I-GWO algoritması tanıtılmıştır.

C. 1. Gri Kurt Optimizasyon Algoritması

Gri kurt optimizasyon algoritması, 2014 yılında Mirjalili tarafından önerilen gri kurtların avlanma stratejisini ve sosyal liderliğini taklit eden bir metasezgiseldir [19]. Gri kurtlar doğal yaşamlarında kendi içerisinde alfa (α), beta (β), delta (δ) ve omega (ω) kurtları olmak üzere dört farklı grup hiyerarşisine sahiptir. Alfa kurtları bu sosyal hiyerarşide en üst basamaktaki en iyi kurtlardır. Sırasıyla diğer en iyi kurtlar beta, delta ve omegadır. GWO'nun avı çevreleme, avlama, avına saldırma ve av arama-keşif olmak üzere üç temel aşaması bulunmaktadır. Avlanma stratejilerinde alfa, beta ve delta kurtlarının avın yeri hakkında daha iyi bilgi sağladığı varsayılmaktadır. Bu nedenle, ilk üç en iyi çözüm (alfa, beta, delta), GWO algoritmasında kurtların konumlarını güncellemek için kullanılır.

Algoritma, başlangıç popülasyonunun rastgele çözümler üretilmesi ile başlar. Aday çözümler iyilik değerlerine göre alfa, beta veya delta gruplarına dahil edilirler. Her bir iterasyonda aday çözümler avın olası konumlarını tahmin ederek ava olan mesafesini günceller. Çözümler öncesinde belirlenen koşullarla avdan uzaklaşma veya yaklaşma eğilimi gösterirler. Sonlandırma koşulu/koşullarının gerçekleşmesiyle algoritma sonlandırılır. Algoritmanın detayları için [19] incelenebilir.

C. 2. Geliştirilmiş Gri Kurt Optimizasyon Algoritması

GWO algoritması optimal çözümü bulmak için α, β, δ ve ω kurtlarını arama uzayının farklı alanlarına yönlendirir. Her kurt için popülasyonun üç lider kurdunun yardımıyla yeni bir konum elde edilir. Bu durum, GWO'nun yavaş yakınsama göstermesine, popülasyonun çeşitliliğini çok erken kaybetmesine ve kurtların yerel optimuma takılmasına neden olmaktadır. Algoritmanın bu dezavantajlarını kaldırmak için Nadimi-Shahraki ve ark. Geliştirilmiş GWO (Improved GWO, I-GWO) varyantını önerdiler [20]. Varyant ile seçme ve güncelleme adımında yeni arama stratejisi içermektedir. I-GWO başlangıç, ava doğru hareket, seçme ve güncelleme olmak üzere üç temel fazdan oluşur.

C.2.1. Başlangıç Fazı

Bu fazda, N kurt, sınırları $[l_i, u_j]$ aralığı ile tanımlanmış bir uzayında Eşt. 10'a göre rastgele dağıtılır.

$$X_{ij} = l_j + \text{rand}_j[0,1] \times \left(u_j - l_j\right), i \in [1, N], j \in [1, D] \quad (10)$$

Algoritmada i. kurdun t. iterasyondaki konumu, D problemin boyutu olmak üzere $\mathrm{X_i(t)} = \{\mathrm{x_{i1}}, \mathrm{x_{i2}}, \dots, \mathrm{x_{iD}}\}$ vektörü olarak temsil edilir. Popülasyonu, N satırı ve D sütunu olan bir Pop matrisinde tutulur. $\mathrm{X_i(t)}$'nin uygunluk değeri, uygunluk fonksiyonu $\mathrm{f(X_i(t))}$ tarafından hesaplanır.

C.2.2. Hareket Fazı

I-GWO, boyut öğrenme tabanlı avcılık (Dimension Learning-based Hunting, DLH) adlı ek bir hareket stratejisi içerir. DLH'da bireysel kurt avının komşular tarafından öğrenildiği kabul edilir. Stratejinin temel işleyişi, kurt $\mathrm{X_i(t)}$'nin yeni konumunun her boyutu bireysel kurdun farklı komşuları ve Pop'tan rastgele seçilen bir kurt tarafından öğrenildiği Eşt. 13 ile hesaplanır. Daha sonra $\mathrm{X_{i-GWO}(t+1)}$'nin yanı sıra strateji, $\mathrm{X_{i-DLH}(t+1)}$ olarak adlandırılan $\mathrm{X_i(t)}$'nin yeni konumu için başka bir aday üretir. Bunu yapmak için $\mathrm{X_i(t)}$'nin mevcut konumu ile $\mathrm{X_{i-GWO}(t+1)}$ aday konumu arasındaki Öklid mesafesi Eşt. 11'deki gibi hesaplanarak $R_i(t)$ yarıçapı elde edilir.

$$R_i(t) = \|X_i(t) - X_{i-GWO}(t+1)\| \quad (11)$$

Daha sonra, $\mathrm{N_i(t)}$ ile gösterilen $\mathrm{X_i(t)}$'nin komşuları $R_i(t)$ yarıçapına göre Eşt. 12 ile elde edilir. Burada $\mathrm{D_i}$, $\mathrm{X_i(t)}$ ve $\mathrm{X_j(t)}$ arasındaki Öklid mesafesidir.

$$N_i(t) = \left\{X_j(t) \mid D_i\left(X_i(t), X_j(t)\right) \leqslant R_i(t), X_j(t) \in \mathrm{Pop}\right\} \quad (12)$$

$X_i(t)$'nin komşularının elde edilmesinin ardından, çoklu komşu öğrenmesi Eşt. 13 ile elde edilir. $X_{i-DLH,d}(t+1)$'nin hesaplanmasında kullanılan $X_{n,d}(t)$ $N_i(t)$'nin d. boyutunda rastgele seçilen komşusunu ve $X_{r,d}(t)$ Pop'tan rastgele seçilen bir kurdu tanımlar.

$$X_{i-DLH,d}(t+1) = X_{i,d}(t) + \text{rand} \times \left(X_{n,d}(t) - X_{r,d}(t)\right) \quad (13)$$

C.2.3. Seçme ve Güncelleme Fazı

Bu fazda ilk olarak $X_{i-GWO}(t+1)$ ve $X_{i-DLH}(t+1)$ adaylarının uygunluk değerleri Eşt. 14 ile karşılaştırılarak daha iyi aday seçilir.

$$X_i(t+1) = \begin{cases} X_{i-GWO}(t+1), & \text{eğer } f(X_{i-GWO}) < f(X_{i-DLH}) \\ X_{i-DLH}(t+1) & \text{diğer durumlarda} \end{cases} \quad (14)$$

$X_i(t+1)$'in yeni konumunu güncellemek için seçilen adayın uygunluk değeri $X_i(t)$'den küçükse $X_i(t)$ seçilen aday tarafından güncellenir. Aksi takdirde, $X_i(t)$ Pop'ta değişmeden kalır. Algoritma bu işleyişi tüm bireyler için gerçekleştirdikten sonra, iterasyon 1 artırılır ve arama, maksimum iterasyon sayısına ulaşana kadar devam eder. I-GWO algoritmasının sözde kodu Tablo 3'te gösterilmektedir.

Tablo 3. *I-GWO sözde kodu*

Algoritma 3: Geliştirilmiş GWO (I-GWO)	
Adım 1.	**Girişler:** $\boldsymbol{N}$, $\boldsymbol{D}$, $\boldsymbol{maks\_iter}$
Adım 2.	**Çıkış :** Küresel optimum
Adım 3.	**Başla**
Adım 4.	Başlangıç popülasyonundaki $\boldsymbol{N}$ kurt arama uzayında rastgele dağıt ve uygunluk değeri hesapla
Adım 5.	**döngü** $\boldsymbol{iter = 2, \dots, maks\_iter}$
Adım 6.	$\mathbf{X}_{\boldsymbol{\alpha}}$, $\mathbf{X}_{\boldsymbol{\beta}}$ ve $\boldsymbol{X_\delta}$'yı bul
Adım 7.	**döngü** $\boldsymbol{i = 1, \dots, N}$
Adım 8.	$\boldsymbol{X_{i-GWO}(t+1)}$'i hesapla
Adım 9.	$\boldsymbol{R_i(t)}$'yi Eşt. 11'e göre hesapla
Adım 10.	$\mathbf{X_i(t)}$'nin komşularını $\boldsymbol{R_i}$'ye göre Eşt. 12'yi kullanarak hesapla
Adım 11.	**döngü** $\boldsymbol{d = 1, \dots, D}$
Adım 12.	Eşt. 13'e göre $\boldsymbol{X_{i-DLH,d}(t+1)}$'i hesapla
Adım 13.	**döngü bitir**
Adım 14.	Eşt. 14'e göre $\mathbf{X_{i-GWO}(t+1)}$ ve $\mathbf{X_{i-DLH}(t+1)}$ arasında aç gözlü seçim yap.
Adım 15.	$\boldsymbol{Pop}$'u güncelle
Adım 16.	**döngü bitir**
Adım 17.	**döngü bitir**
Adım 18.	Küresel optimumu döndür.
Adım 19.	**bitir**

Literatürde, Hua ve ark. GWO'nun bu varyantını temel alınarak PID kontrolör tasarımı yaptılar [21]. Çalışma neticesinde tasarlanan kontrolörün insan midesine ilaç salınımı için kullanılan robotların dar alanda hareket kabiliyetini arttırarak hızını optimize edebildiği gözlemlenmiştir.

D. DENİZ AVCILARI ALGORİTMASI

Deniz Avcıları Algoritması (Marine Predators Algorithm, MPA), 2020 yılında Faramarzi ve ark. tarafından önerilen deniz avcılarının avlara saldırı davranışından ilham alarak geliştirdikleri popülasyon tabanlı bir sezgisel algoritmadır [22]. Avcılar, avlarını hedeflerken Brownian ve Levy olarak adlandırılan iki temel hareket kullanır. İki hareket arasında bir denge sağlamak için avcılar, avdan

mevcut konumlarına göre hız oranını hesaplar. İlk olarak, MPA, arama uzayına rastgele yayılan başlangıç çözümlerini Eşt. 15'e göre üretir.

$$X_0 = X_{min} + B(X_{maks} - X_{min}) \tag{15}$$

Burada X_{min}ve X_{maks} arama uzayları için minimum ve maksimum sınırlardır. B, [0,1] aralığında rastgele üretilmiş bir sayıdır. Algoritma Elit (Elite) ve Av (Prey) matrisleri kavramlarını ortaya atar. Elit matris, en iyi avcıyı temsil eder ve avın konum verilerini kullanarak avın aranmasını ve algılanmasını içerir. Elit matrisi Eşt. 16'daki gibi tanımlanır.

$$\text{Elit} = \begin{bmatrix} X_{1,1}^I & \cdots & X_{1,d}^I \\ \vdots & \ddots & \vdots \\ X_{n,1}^I & \cdots & X_{n,d}^I \end{bmatrix} \tag{16}$$

Burada $\overrightarrow{X^I}$, n popülasyon büyüklüğü ve d problem boyutu olmak üzere n kez tekrarlanarak Elit matrisini oluşturmak için kullanılan en iyi avcı olarak tanımlanır. MPA'da popülasyon bireyleri hem avcılardan hem de avdan oluşmaktadır. Matrisin bu şekilde oluşturulmasının temel nedeni, avcının av ararken diğer avcılar için av da olabilmesidir. Diğer bir deyişle bireyler avcı ya da av olabilir. Her iterasyon sonunda en iyi avcı değişebileceğinden Elit matrisi güncellenir. Avcıların konumlarını güncellemek için kullanılan Av matrisi Eşt. 17'deki gibi tanımlanır.

$$\text{Av} = \begin{bmatrix} X_{1,1} & \cdots & X_{1,d} \\ \vdots & \ddots & \vdots \\ X_{n,1} & \cdots & X_{n,d} \end{bmatrix} \tag{17}$$

Burada $X_{i,j}$ $i.$ avın $j.$ boyutunu temsil eder. MPA, aşağıda belirtilen av ve avcı hızına dayalı üç orana göre adımlarını gerçekleştirir:

1. Avın avcıdan daha hızlı hareket ettiği durumda yüksek hız oranı
2. Hem avcı hem de av hemen hemen aynı hızda hareket ettiği durumda eşit hız oranı
3. Avcının avdan daha hızlı olduğu durumda düşük hız oranı

Bu adımlar aşağıda detaylandırılmaktadır.

D. 1. Birinci Adım

Bir avcının avdan daha hızlı hareket etmesi durumunda, mevcut iterasyon maksimum iterasyon sayının üçte birinin altındayken Eşt. 18'e göre konum güncellenir.

$$\text{eğer } t_{\text{current}} < \frac{1}{3} * t_{maks} \text{ ise } \overrightarrow{AdımSayısı_i} = \vec{R}_B \otimes \left(\overrightarrow{Elit_i} - \vec{R}_B \otimes \overrightarrow{Av_i}\right) i = 1,2,\dots,n \tag{18}$$

$$\overrightarrow{Av_i} = \overrightarrow{Av_i} + P.\vec{R} \otimes \overrightarrow{AdımSayısı_i} \tag{19}$$

Burada $\otimes$ eleman bazında çarpmayı ifade eder. R_B, Brownian hareketini temsil eder ve bir rastgele sayı vektörü içerir. Avın hareketinin simülasyonu $\vec{R}_B \otimes \overrightarrow{Av_i}$ tarafından tanımlanır. Ayrıca, P ve R, sırasıyla 0,5'e eşit sabit bir sayı ve [0, 1] arasında rastgele sayılardan oluşan bir vektördür. Bu adım, adım boyutunun veya hareketin hızının büyük olduğu durumlarda yüksek arama yeteneği için kullanılır.

D. 2. İkinci Adım

İkinci adımda hem aramanın hem de sömürmenin önemi eşit ağırlıktadır. Bu adımda avcı ve avın hızı eşit olduğun durumda arama, sömürüye dönüşme eğilimindedir. Böylece bireylerin yarısı arama, diğer yarısı sömürü için kullanılır. Av sömürü ve avcılar ve arama rollerine sahiptir. Her birey kendine özgü hareketi gerçekleştirir. Avlar lévy hareketini, avcılar Brownian hareketini benimserler. Bu adım için

denklemler aşağıdaki gibi tanımlanırken, iterasyon sayısı maksimum iterasyon sayısının üçte biri ile üçte ikisi arasındadır.

$$\text{eğer } \frac{1}{3} * t_{maks} < t_{güncel} < \frac{2}{3} * t_{maks} \text{ ise } \overrightarrow{AdımSayısı_i} = \vec{R}_L \otimes \left(\overrightarrow{Elit_i} - \vec{R}_L \otimes \overrightarrow{Av_i}\right) \quad i = 1,2,\dots,n/2 \tag{20}$$

$$\overrightarrow{Av_i} = \overrightarrow{Av_i} + P.\vec{R} \otimes \overrightarrow{AdımSayısı_i} \tag{21}$$

Burada R_L, Lévy hareketini temsil eder ve bir rasgele sayı vektörü içerir. Avın hareketinin matematiksel modellenmesi $\vec{R}_L \otimes \overrightarrow{Av_i}$ ile sunulmaktadır. Adım boyutu daha iyi sömürüye yol açan küçük adımlarla ilişkilendirilir. Popülasyonun diğer yarısı için aşağıdaki eşitlikler kullanılır.

$$\text{eğer } \frac{1}{3} * t_{maks} < t_{\text{güncel}} < \frac{2}{3} * t_{maks} \text{ ise } \overrightarrow{AdımSayısı_i} = \vec{R}_B \otimes \left(\vec{R}_B \otimes \overrightarrow{Elit_i} - \overrightarrow{Av_i}\right) \quad i = \frac{n}{2},\dots,n. \tag{22}$$

$$\overrightarrow{Av_i} = \overrightarrow{Av_i} + P.\vec{R} \otimes \overrightarrow{AdımSayısı_i} \tag{23}$$

$$CF = \left(1 - \frac{\text{iter}}{\text{maks\_iter}}\right)^{\left(2 \times \frac{\text{iter}}{\text{maks\_iter}}\right)} \tag{24}$$

Burada CF, avcının hareketini kontrol etme faktörüdür. R_B ile çarpılan elit, avcının Brownian hareketini matematiksel modellememek için kullanılır ve av pozisyonu avcının hareketine göre güncellenir.

D. 3. Üçüncü Adım

Bu adımda avcının hızının avdan yüksek olduğu durumlarda kullanılır ve yüksek sömürü kabiliyetine karşılık gelir. İterasyon sayısı, maksimum iterasyon sayısının üçte ikisinden daha yüksek olduğunda, konumu güncellemenin en iyi yolu lévy eşitliğidir. Bu aşamada aşağıdaki eşitlikler kullanılır.

$$\text{eğer } t_{güncel} > \frac{2}{3} * t_{maks} \text{ ise } \overrightarrow{AdımSayısı_i} = \vec{R}_L \otimes \left(\vec{R}_L \otimes \overrightarrow{Elit_i} - \overrightarrow{Av_i}\right) \quad i = 1,\dots,n. \tag{25}$$

$$\overrightarrow{Av_i} = \overrightarrow{Elit_i} + P.CF \otimes \overrightarrow{AdımSayısı_i} \tag{26}$$

Yukarıda belirtilen adımlara ek olarak, MPA, deniz avcılarını daha iyi modellemek için iki farklı çevresel etkiyi de kullanır. Bu etkilerden ilki balık toplama cihazları (Fish Aggregating Devices, FADs) veya girdap dönüşümü olarak adlandırılır ve Eşt. 27'de bu dönüşüm modellenmiştir.

$$\overrightarrow{Av_i} == \begin{cases} \overrightarrow{\text{Av}_i} + CF\left[\vec{X}_{min} + \vec{R} \otimes \left(\vec{X}_{maks} - \vec{X}_{min}\right)\right] \otimes \vec{U} & \text{eğer } r \leq \text{ FADs} \\ \overrightarrow{\text{Av}_i} + [\text{ FADs } (1-r) + r]\left(\overrightarrow{\text{Prey}}_{r1} - \overrightarrow{\text{Prey}}_{r2}\right) & \text{eğer } r > \text{ FADs} \end{cases} \tag{27}$$

Eşt. 27'deki $FADs$, algoritma için olasılıksal bir etki olarak tanımlanır ve 0,2'ye eşittir. $\vec{U}$ ikili bir vektördür. Bu ikili vektördeki her dizi, 0 ile 1 arasında rastgele bir sayı üreterek oluşturulur. Oluşturulan sayı 0,2'den küçükse, dizi 0 değerini alır, aksi takdirde 1 değerini alır. r, 0 ile 1 arasında rastgele bir sayıdır. $\vec{X}_{min}$ ve $\vec{X}_{maks}$ sırasıyla boyutların alt ve üst sınırlarıdır. Son olarak, $r1$ ve $r2$, av matrisinin rastgele indislerini temsil eder.

Deniz hafızası adı verilen diğer çevresel tepki, avcının av aramada başarılı olduğu en iyi yerleri hatırlama kabiliyeti ile ilgilidir. Bu kabiliyet MPA'da bellek tasarrufu olarak kullanılır. Avın konumu güncelledikten ve FAD'lerin etkisini sağladıktan sonra, Elit'i yükseltmek için matris değerlendirilecektir. Mevcut iterasyonda, her çözümün uygunluk değeri, önceki iterasyondaki değeri ile karşılaştırılıp en iyisi seçilecektir. Bu aç gözlü seçimle, her iterasyon boyunca çözümün kalitesi artırılması hedeflenmektedir. Bununla birlikte, avcıların, avların bol olduğu bölgelerinin önceki

konumlarını başarılı bir arama ile hatırlamalarına yardımcı olmaktadır. MPA'nın sözde kodu Tablo 4'te verilmiştir.

Tablo 4. *MPA sözde kodu*

Algoritma 4: MPA

Adım 1. Başlangıç $\boldsymbol{Adım}$, $\boldsymbol{P}$, $\overrightarrow{\boldsymbol{Av_i}}$.
Adım 2. **döngü** $\mathbf{t} < \boldsymbol{t_{maks}}$ **yap**
Adım 3. Her bir $\overrightarrow{\boldsymbol{Av_i}}$ için $\boldsymbol{f}(\overrightarrow{\boldsymbol{Av_i}})$ değerini hesapla.
Adım 4. Elit matrisini oluştur.
Adım 5. Hafızaya al.
Adım 6. $\boldsymbol{CF}$'yi Eşt. 24'e göre güncelle
Adım 7. **döngü** each $\overrightarrow{\boldsymbol{Av_i}}$ **yap**
Adım 8. **eğer** $\left(\boldsymbol{t_{current}} < \frac{\mathbf{1}}{\mathbf{3}} * \boldsymbol{t_{max}}\right)$ **ise**
Adım 9. $\overrightarrow{\boldsymbol{Prey_i}}$'yi Eşt. 18 ve 19'a göre konumlandır.
Adım 10. **değilse**
Adım 11. **eğer** $\left(\frac{\mathbf{1}}{\mathbf{3}} * \boldsymbol{t_{max}} < \boldsymbol{t_{\text{current}}} < \frac{\mathbf{2}}{\mathbf{3}} * \boldsymbol{t_{max}}\right)$ **ise**
Adım 12. **eğer** $\left(\boldsymbol{i} < \frac{\mathbf{1}}{\mathbf{2}} * \boldsymbol{n}\right)$ **ise**
Adım 13. $\overrightarrow{\boldsymbol{Av_i}}$'yi Eşt. 20 ve 21'e göre konumlandır.
Adım 14. **değilse**
Adım 15. $\overrightarrow{\boldsymbol{Av_i}}$'yi Eşt. 22 ve 23'e göre konumlandır.
Adım 16. **eğer bitir**
Adım 17. **değilse**
Adım 18. $\overrightarrow{\boldsymbol{Av_i}}$'yi Eşt. 25 ve 26'ya göre konumlandır.
Adım 19. **eğer bitir**
Adım 20. **eğer bitir**
Adım 21. Her bir $\overrightarrow{\boldsymbol{Av_i}}$ için $\boldsymbol{f}(\overrightarrow{\boldsymbol{Av_i}})$ değerini hesapla.
Adım 22. En iyi $\overrightarrow{\boldsymbol{Av_i}}$'nin konumunu ve uygunluk değerini güncelle
Adım 23. En iyi $\overrightarrow{\boldsymbol{Av_i}}$ hafızaya al
Adım 24. $\boldsymbol{FAD}$ etkisini Eşt. 27'ye göre uygula
Adım 25. **döngü bitir**
Adım 26. $\boldsymbol{t_{güncel}} + +$
Adım 27. **döngü bitir**

MPA algoritmasının bir çok varyantı ile farklı optimizasyon problemleri çözülmüştür. Elaziz ve ark. quantum teorisini MPA'ya uyarlayarak resim segmentasyonu problemi için en uygun eşik seviyelerini bulmaya çalıştılar [23]. Xing ve He bu probleminin çok amaçlı versiyonu için güçlendirilmiş MPA'yı önerdiler [24]. Her iki algoritma literatürde bilinen metasezgisellere karşı başarılı sonuçlar elde etmiştir. Saqid ve ark. MPA algoritmasının nonlineer varyantı ile bir çok test probleminde ve NOMA-VLC-B5G sistemlerinde birden fazla kullanıcı için adil güç tahsisi problemini başarıyla çözdüler [25]. Hassan ve ark. MPA'nın bir diğer varyantını tek ve çok amaçlı birleşik ekonomik emisyon dağıtımı problemini çözmek için önerdiler [26]. Yazarlar, çok amaçlı problem için pareto yaklaşımı önerilen varyant ve bulanık yöntem ile kullanarak klasik algoritmalara karşı üstünlük sağlamıştır. Houssein ve ark. Deep-MPA olarak adlandırdıkları MPA'nın başka bir varyantını, klasik MPA'nın dezavantajlarının üstesinden gelmek için önerdiler [27]. Varyant ile bölgesel optimumlardan kaçınma, arama ve sömürme arasındaki dengenin sağlanması, yakınsama performansının arttırılması gibi iyileşmeler gerçekleştirildi.

III. DENEYSEL ÇALIŞMALAR

Bu bölümde algoritmaların karşılaştırılmasında kullanılan test fonksiyonları, parametreler yer verilmiştir. Sonrasında benzetim sonuçları ve sonuçların anlamlılığını değerlendiren istatiksel test sonuçları yer almıştır.

A. TEST FONKSİYONLARI

Bu bölümde, algoritmalar birçok çalışmada kullanılan 23 farklı test fonksiyonu ile kıyaslanmıştır [28-30]. Fonksiyonlar üç temel gruba ayrılabilir: tek modlu, çok modlu, farklı boyutlu çok modlu. Her bir grup sırasıyla Tablo 5, Tablo 6 ve Tablo 7'de sunulmuştur. Tablolarda verilen aralık fonksiyonların arama uzayının sınırlarını f_{min} optimum değerini temsil etmektedir.

Tablo 5. *Tek modlu test fonksiyonları*

Fonksiyon	Boyut	Aralık	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30, 50, 100	[-100,100]	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30, 50,1 00	[-10,10]	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30, 50, 100	[-100,100]	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30, 50, 100	[-100,100]	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30, 50, 100	[-30,30]	0
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30, 50, 100	[-100,100]	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}(0,1)$	30, 50, 100	[-1.28,1.28]	0

Tablo 6. *Çok modlu test fonksiyonları*

Fonksiyon	Boyut	Aralık	f_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30, 50, 100	[-500,500]	-418.98
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	30, 50, 100	[-5.12,5.12]	0
$F_{10}(x) = -20\exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	30, 50, 100	[-32,32]	0
$F_{11}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30, 50, 100	[-600,600]	0
$F_{12}(x) = \frac{\pi}{n} \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\}$ $+ \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30, 50, 100	[-50,50]	0

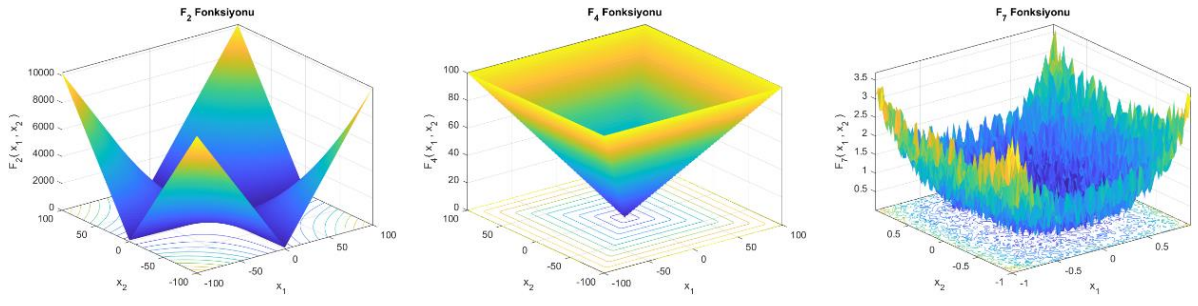
Tablo 6(devam). *Çok modlu test fonksiyonları*

$F_{13}(x) = 0.1\left\{\begin{array}{c}\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)] + (x_n-1)^2[1+ \\ \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i,5,100,4)\end{array}\right.$	30, 50, 100	[-50,50]	0

Tablo 7. *Farklı boyutlu çok modlu test fonksiyonları*

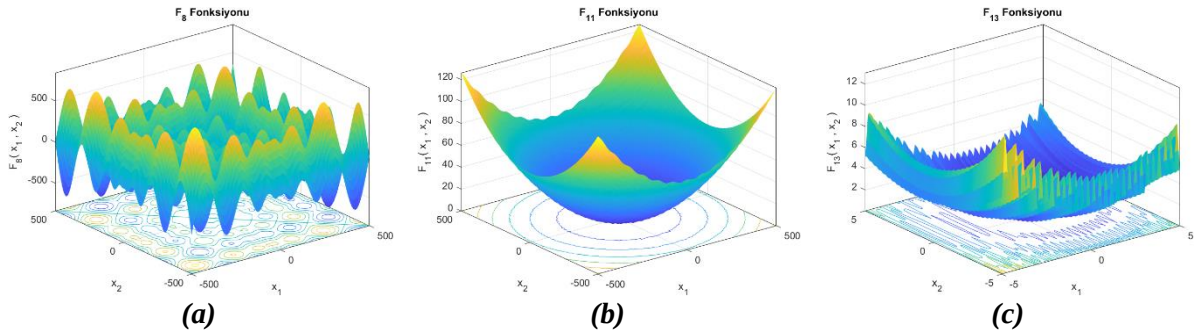
Fonksiyon	Boyut	Aralık	f_{min}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[-65,65]	1
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	[-5,5]	0.0003
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316
$F_{17}(x_1,x_2) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1-\frac{1}{8\pi}\right)\cos x_1 + 10$	2	[-5,5]	0.398
$F_{18}(x_1,x_2) = [1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)] \times [30+(2x_1-3x_2)^2 \times (18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	[-2,2]	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j-p_{ij})^2\right)$	3	[1,3]	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j-p_{ij})^2\right)$	6	[0,1]	-3.32
$F_{21}(x) = -\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T + c_i]^{-1}$	4	[0,10]	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T + c_i]^{-1}$	4	[0,10]	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T + c_i]^{-1}$	4	[0,10]	-10.5363

Tablo 7'de ifade edilen fonksiyonların birçoğunun boyutu eşitliklerinden de anlaşıldığı üzere değiştirilemeyeceğinden boyut değerleri sabit bırakılmıştır. Tablolardaki bazı fonksiyonların grafikleri Şekil 1 (F_2, F_4, F_7), Şekil 2 (F_8, F_{11}, F_{13}), ve Şekil 3 (F_{14}, F_{18}, F_{21})'de çizdirilmiştir.

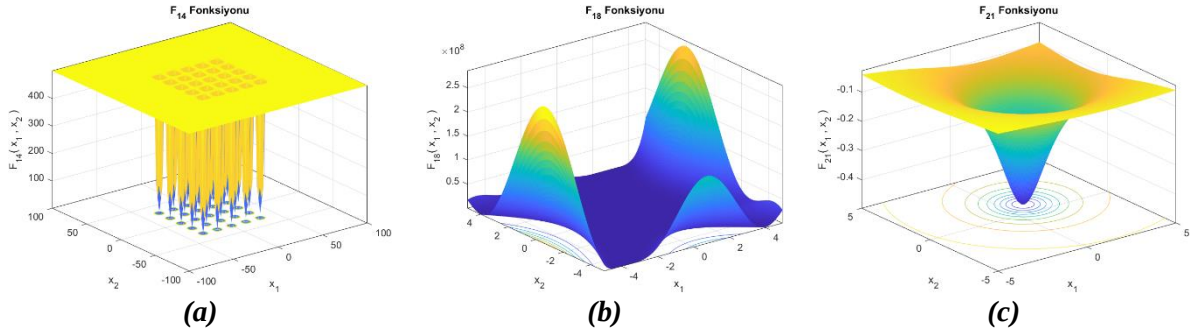


(a) ***(b)*** ***(c)***

Şekil 1. *Bazı tek modlu fonksiyonlar*



Şekil 2. *Bazı çok modlu fonksiyonlar*



Şekil 3. *Bazı farklı modlu fonksiyonlar*

B. PARAMETRELER

Çalışmada kullanılan parametreler Tablo 8'de sunulmuştur. p_1, p_2 ve p_3 AVOA'da temel fazlarda arama ve sömürü aşamalarında stratejilerin nasıl seçilmesine dair kullanılan parametrelerdir. Burada L_1 en iyi akbabayı seçmek için olasılık parametresi, L_2 en iyi ikinci akbabayı seçmek için olasılık parametresidir. w ise arama ve sömürü fazlarını sonlandırılıp/ sonlandırılmayacağını belirleyen bir parametredir. I-GWO'da tanımlanan a başlangıçtaki değeri 2 olarak belirlenen ve iterasyonlar boyunca lineer azalan bir parametredir. MPA'da ise iki temel kontrol parametresi bulunmaktadır. $FADs$, Eşt. 27'de tanımlanan etki değer; P, Eşt. 19'da ifade edilen sabittir. PSO için tanımlanan c_1, c_2, w_{min} ve w_{max} parametreleri hız ve konum değerleri için ağırlık katsayılarıdır. Benzer çalışmalarda iterasyon sayısı ve popülasyon büyüklüğü sırasıyla 500 ve 30 alındığı için bu çalışmada da aynı değerler kullanılmıştır [16, 22]. Adil bir karşılaştırma olması için bir iterasyonda algoritmaların çağırdığı uygunluk fonksiyonları karşılaştırılmıştır. P popülasyon büyüklüğü olmak üzere AVOA, I-GWO ve PSO'nun çağırdıkları uygunluk fonksiyonu sayısı eşit ve P iken, MPA'nın çağırdığı uygunluk fonksiyonu sayısı $2P$'dir. Bu nedenle tüm algoritmaların toplam uygunluk fonksiyonu çağırma sayısı $500P$ olması için sadece MPA'nın iterasyon sayısı 500 ve popülasyon büyüklüğü 15 alınmıştır. Bunun haricinde diğer tüm algoritmalarda iterasyon sayısı 500, popülasyon büyüklüğü 30 alınmıştır.

Tablo 8. *Parametreler tek modlu test fonksiyonları benzetim sonuçları*

Kontrol Parametreleri	**Algoritmalar**			
	AVOA	**I-GWO**	**MPA**	**PSO**
p_1	0,6	-	-	-
p_2	0,4	-	-	-
p_3	0,6	-	-	-
L_1	0,8	-	-	-
L_2	0,2	-	-	-

w	2,5	-	-	-
a	-	2	-	-
$FADs$	-	-	0,2	-
P	-	-	0,5	-
c_1	-	-	-	2
c_2	-	-	-	2
w_{min}	-	-	-	0,2
w_{max}	-	-	-	0,9
Popülasyon Büyüklüğü	30	30	15	30
Maksimum İterasyon Sayısı	500	500	500	500

C. BENZETİM SONUÇLARI

Çalışmada tüm algoritmalar 30 kez bağımsız koşturulmuştur. Algoritmalar MATLAB 2022b platformu kullanılarak yazılmıştır. Kullanılan bilgisayar, Core i7-8565U CPU, 1.8 Ghz hıza ve 8GB RAM'e sahiptir. Benzetim sonuçları sırasıyla tek modlu fonksiyonlar için Tablo 9'da, çok modlu fonksiyonlar için Tablo 10'da ve farklı boyutlu çok modlu fonksiyonlar için Tablo 11'de paylaşılmıştır. Her fonksiyon için koşmaların ortalama, standart sapma, en iyi ve en kötü sonuçları tablolarda verilmiştir. Ayrıca tablolarda ortalamalarda en iyi değere sahip algoritmalar kalınlaştırılmıştır.

Tablo 9'da görüldüğü üzere bütün tek modlu fonksiyonlar için AVOA diğer algoritmalardan daha başarılı sonuçlar elde etmiştir. Bununla birlikte problemin boyutu arttıkça algoritmaların hata değeri de artmaktadır. F_1 ve F_2fonksiyonları için PSO'nun hata değerleri hariç diğer algoritmalar minumum değere oldukça yakın sonuçlar üretmişlerdir. PSO sadece F_1, F_6 ve F_7 fonksiyonlarında 30 boyutludaki hata değerleri sıfıra yakındır. Ancak diğer boyutlarda ve fonksiyonlarda doğru tahmin yapamadığını söylenebilir. I-GWO özellikle 100 boyutluda F_3, F_5 ve F_6 değerlerinde diğer boyutlardaki başarısını yakalayamamıştır. MPA'nın ise en başarısız olduğu fonksiyon F_5'dir. Bunun dışında MPA F_3 fonksiyonu için 100 boyutluda da I-GWO'dan ve PSO'dan daha başarılı sonuç üretse de optimum çözümü elde edememiştir. Tablodan tek modlu fonksiyonlar için başarı sıralamasının AVOA'nın en başarılı olduğu, I-GWO ve MPA'nın yakın başarılara sahip olduğu ve PSO'nun sıralamanın sonunda olduğu söylenebilir.

Çok modlu fonksiyonlarda özellikle F_8'in tahmininde algoritmaların optimum çözümü bulamadığı Tablo 10'dan çıkarılabilir. Tablo 9'daki başarı sıralamasında olduğu gibi AVOA algoritmalar arasında en başarılıdır. Burada özellikle F_9 ve F_{13} fonksiyonları için I-GWO'nun optimum çözümü üretemediği görülebilir. Benzer durum gene F_{13} fonksiyonu için MPA'da da geçerlidir. PSO genel olarak 50 ve 100 boyutta başarı sağlayamamıştır. Bu tablonun başarı sıralaması AVOA, MPA, I-GWO ve PSO olarak verilebilir.

Çalışmamızda Tablo 7'de gösterilen toplamda 9 farklı boyutlu fonksiyon için başarı kıyaslamasına Tablo 11'den incelenebilir. Buradaki her fonksiyonun optimum değerleri farklıdır. F_{14} fonksiyonunda ortalamada AVOA diğer algoritmalardan daha yüksek bir değer ile optimum sonucu elde edememiştir. F_{15} fonksiyonunda en başarılı algoritma tüm koşmalarda optimum değeri yakalayan MPA'dır. Bu fonksiyon için başarı sıralaması yüksekten düşüğe PSO, AVOA ve I-GWO'dur. F_{16}, F_{17}, F_{18} ve F_{19} fonksiyonları için tüm algoritmalar bütün koşmalarda optimum değeri üretebilmişlerdir. F_{20}'de I-GWO ve MPA, F_{21}'de AVOA ve MPA optimum değerleri üretebilmiştir. F_{22} ve F_{23}'de PSO diğer algoritmaların gerisinde kalmıştır.

Algoritmaların yakınsamaları hakkında yorum yapılabilmesi için her bir gruptan ikişer fonksiyonun yakınsama grafikleri çizilmiştir. Tek modlu için F_3 ve F_5 çok modlu için F_8 ve F_{13}; farklı boyutlu çok modlu için F_{15} ve F_{20} fonksiyonlarının yakınsama grafikleri çizilmiştir. Ortalamalarda farklı algoritmaların başarılı olması nedeniyle bahsedilen fonksiyonlar grafik çiziminde tercih edilmiştir. Bu grafikler sırasıyla Şekil 4, Şekil 5 ve Şekil 6'da paylaşılmıştır.

Şekil 1 ve Şekil 2'deki her fonksiyon 30, 50 ve 100 boyutlarında çalıştırıldığından sırasıyla bu boyutlardaki yakınsamaları için 3 farklı grafik çizilmiştir. Şekil 4'deki tek modlu her iki fonksiyon için AVOA'nın yakınsama hızının diğer algoritmalardan çok daha fazla olduğu görülebilir. F_3 fonksiyonunda tüm boyutlarda AVOA hızlı yakınsarken 30 ve 50 boyutları için MPA ve I-GWO'nin yakınsama hızı; 100 boyutu için I-GWO ve PSO yakınsama hızı birbirine oldukça yakındır. F_5 fonksiyonunda I-GWO ve MPA ilk 100 iterasyona ulaşmadan bulabildikleri minumum amaç değerine ulaşarak diğer iterasyonlarda kendilerini geliştirememişlerdir. PSO 30 boyutu hariç tüm iterasyonlarda iyileşme sağlasa da diğer algoritmaların gerisinde kalmıştır.

Tablo 9. *Tek modlu test fonksiyonları benzetim sonuçları*

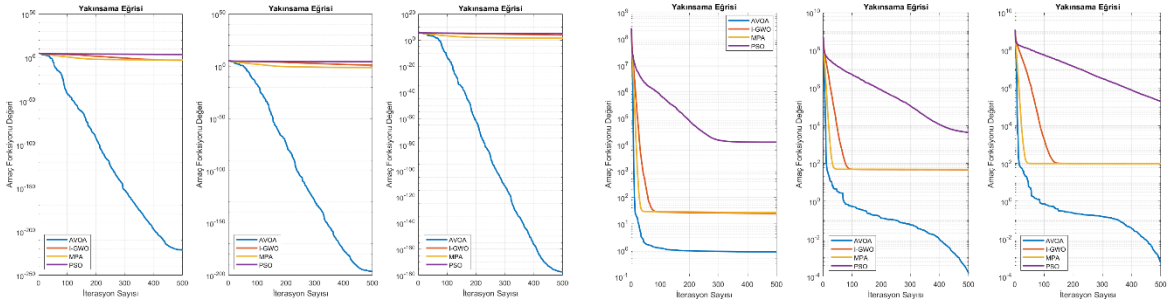
Fonksiyonlar	Algoritmalar	AVOA			I-GWO			MPA			PSO		
	Boyut	**30**	**50**	**100**	**30**	**50**	**100**	**30**	**50**	**100**	**30**	**50**	**100**
F_1	**Ortalama**	**2,55E-293**	**1,69E-289**	**2,35E-289**	3,45E-28	3,98E-20	2,57E-12	1,11E-23	9,04E-22	3,27E-20	4,76E-03	7,97E+00	3,17E+03
	Standart Sapma	0,00E+00	0,00E+00	0,00E+00	7,61E-28	3,81E-20	2,25E-12	1,32E-23	2,02E-21	3,22E-20	4,94E-03	4,94E+00	6,20E+03
	En İyi	0,00E+00	0,00E+00	0,00E+00	4,95E-30	1,19E-21	4,56E-13	5,01E-27	8,51E-24	6,58E-22	1,75E-04	1,90E+00	4,81E+02
	En Kötü	7,66E-292	5,07E-288	7,05E-288	3,88E-27	1,38E-19	1,03E-11	5,31E-23	1,10E-20	1,35E-19	1,94E-02	2,37E+01	2,09E+04
F_2	**Ortalama**	**2,42E-144**	**5,58E-146**	**2,37E-141**	6,31E-18	4,52E-13	1,34E-08	1,21E-13	1,26E-12	7,26E-12	2,36E+00	1,22E+01	6,35E+01
	Standart Sapma	1,32E-143	3,05E-145	9,02E-141	4,07E-18	2,34E-13	5,78E-09	1,44E-13	1,39E-12	5,71E-12	4,29E+00	1,42E+01	2,34E+01
	En İyi	1,59E-176	2,23E-175	1,51E-176	5,20E-19	1,26E-13	6,86E-09	1,42E-14	5,57E-14	3,77E-13	2,98E-03	6,31E-01	3,38E+01
	En Kötü	7,25E-143	1,67E-144	3,73E-140	1,91E-17	1,09E-12	3,05E-08	6,40E-13	6,43E-12	2,01E-11	1,00E+01	6,07E+01	1,22E+02
F_3	**Ortalama**	**1,37E-221**	**6,69E-197**	**4,45E-178**	5,54E-04	1,43E+01	6,61E+03	1,16E-03	5,33E-02	3,27E+01	2,85E+03	2,07E+04	1,01E+05
	Standart Sapma	0,00E+00	0,00E+00	0,00E+00	9,88E-04	1,69E+01	2,97E+03	4,66E-03	6,43E-02	6,75E+01	2,75E+03	6,25E+03	1,83E+04
	En İyi	4,24E-283	8,51E-278	2,09E-270	1,05E-05	9,64E-01	1,53E+03	2,63E-09	5,31E-04	2,91E-01	5,72E+02	1,02E+04	6,49E+04
	En Kötü	4,12E-220	2,01E-195	1,33E-176	5,10E-03	6,80E+01	1,39E+04	2,57E-02	2,87E-01	3,30E+02	1,08E+04	3,73E+04	1,40E+05
F_4	**Ortalama**	**3,41E-146**	**1,01E-144**	**5,39E-136**	1,77E-05	1,46E-02	4,33E+00	2,95E-09	1,80E-08	1,39E-07	7,14E+00	1,88E+01	4,23E+01
	Standart Sapma	1,80E-145	5,27E-144	2,95E-135	1,85E-05	1,18E-02	2,27E+00	2,23E-09	1,25E-08	8,44E-08	1,50E+00	2,43E+00	3,66E+00
	En İyi	3,19E-178	6,78E-169	2,20E-171	3,03E-06	2,04E-03	8,55E-01	8,05E-10	4,51E-09	2,99E-08	3,64E+00	1,47E+01	3,56E+01
	En Kötü	9,84E-145	2,89E-143	1,62E-134	7,81E-05	5,94E-02	9,72E+00	1,10E-08	5,52E-08	3,19E-07	9,95E+00	2,52E+01	5,37E+01
F_5	**Ortalama**	**8,60E-01**	**1,47E-04**	**5,63E-04**	2,43E+01	4,55E+01	9,69E+01	2,63E+01	4,73E+01	9,77E+01	1,23E+04	4,33E+03	1,95E+05
	Standart Sapma	4,71E+00	2,53E-04	7,55E-04	3,56E-01	1,31E+00	1,45E+00	4,64E-01	6,74E-01	6,50E-01	3,10E+04	1,64E+04	7,23E+04
	En İyi	1,72E-06	7,33E-06	1,03E-05	2,37E+01	4,41E+01	9,45E+01	2,53E+01	4,61E+01	9,60E+01	2,35E+01	5,44E+02	5,50E+04
	En Kötü	2,58E+01	1,42E-03	3,09E-03	2,51E+01	4,86E+01	9,85E+01	2,74E+01	4,86E+01	9,84E+01	9,01E+04	9,10E+04	3,55E+05
F_6	**Ortalama**	**4,32E-07**	**1,10E-05**	**3,46E-04**	7,27E-05	1,37E+00	7,88E+00	1,31E-01	1,29E+00	7,64E+00	6,01E-03	6,56E+00	2,84E+03
	Standart Sapma	2,07E-07	6,83E-06	7,71E-04	2,20E-05	4,54E-01	9,21E-01	1,49E-01	3,93E-01	9,31E-01	7,27E-03	4,72E+00	4,02E+03
	En İyi	8,20E-08	1,47E-06	8,46E-06	3,42E-05	2,44E-01	6,36E+00	2,65E-07	4,09E-01	5,37E+00	4,21E-04	1,57E+00	3,72E+02
	En Kötü	8,80E-07	3,33E-05	4,34E-03	1,26E-04	2,02E+00	9,90E+00	5,06E-01	1,96E+00	9,42E+00	3,87E-02	2,21E+01	1,12E+04
F_7	**Ortalama**	**1,75E-04**	**2,02E-04**	**1,67E-04**	3,02E-03	5,79E-03	1,35E-02	1,57E-03	1,95E-03	2,45E-03	4,67E-02	5,37E-01	6,98E+00
	Standart Sapma	1,72E-04	2,40E-04	1,75E-04	1,38E-03	1,80E-03	4,26E-03	8,63E-04	1,15E-03	1,35E-03	1,95E-02	1,53E+00	9,02E+00
	En İyi	4,47E-07	1,43E-05	2,69E-06	1,17E-03	2,60E-03	5,25E-03	2,44E-04	3,15E-04	4,55E-04	1,31E-02	1,14E-01	9,06E-01
	En Kötü	7,28E-04	1,17E-03	8,70E-04	6,65E-03	9,74E-03	2,32E-02	4,14E-03	4,98E-03	5,77E-03	9,45E-02	8,20E+00	3,67E+01

Tablo 10. *Çok modlu test fonksiyonları benzetim sonuçları*

Fonksiyonlar	Algoritmalar	AVOA			I-GWO			MPA			PSO		
	Boyut	**30**	**50**	**100**	**30**	**50**	**100**	**30**	**50**	**100**	**30**	**50**	**100**
F_8	**Ortalama**	-1,24E+04	-2,02E+04	-4,08E+04	-7,68E+03	-1,07E+04	-1,55E+04	-8,34E+03	-1,28E+04	-2,31E+04	-8,63E+03	-1,29E+04	-2,19E+04
	Standart Sapma	2,74E+02	1,13E+03	1,98E+03	1,60E+03	3,23E+03	5,97E+03	4,85E+02	9,32E+02	1,15E+03	5,21E+02	1,03E+03	1,46E+03
	En İyi	-1,26E+04	-2,09E+04	-4,19E+04	-1,03E+04	-1,58E+04	-2,67E+04	-9,50E+03	-1,48E+04	-2,56E+04	-9,59E+03	-1,47E+04	-2,41E+04
	En Kötü	-1,15E+04	-1,71E+04	-3,42E+04	-5,32E+03	-6,53E+03	-9,41E+03	-6,96E+03	-1,14E+04	-2,08E+04	-7,69E+03	-1,10E+04	-1,81E+04
F_9	**Ortalama**	**0,00E+00**	**0,00E+00**	**0,00E+00**	3,08E+01	5,08E+01	1,25E+02	**0,00E+00**	**0,00E+00**	**0,00E+00**	5,65E+01	1,35E+02	4,01E+02
	Standart Sapma	0,00E+00	0,00E+00	0,00E+00	3,30E+01	1,72E+01	4,30E+01	0,00E+00	0,00E+00	0,00E+00	1,99E+01	2,99E+01	5,39E+01
	En İyi	0,00E+00	0,00E+00	0,00E+00	7,03E+00	1,85E+01	4,12E+01	0,00E+00	0,00E+00	0,00E+00	3,30E+01	7,70E+01	2,83E+02
	En Kötü	0,00E+00	0,00E+00	0,00E+00	1,44E+02	8,38E+01	2,12E+02	0,00E+00	0,00E+00	0,00E+00	1,31E+02	1,94E+02	4,94E+02
F_{10}	**Ortalama**	4,44E-16	4,44E-16	4,44E-16	5,69E-14	3,38E-11	1,52E-07	1,13E-12	4,15E-12	1,70E-11	8,33E-01	2,83E+00	7,19E+00
	Standart Sapma	0,00E+00	0,00E+00	0,00E+00	9,89E-15	1,86E-11	5,71E-08	9,31E-13	2,53E-12	1,29E-11	7,40E-01	1,91E+00	2,48E+00
	En İyi	4,44E-16	4,44E-16	4,44E-16	3,95E-14	5,24E-12	7,31E-08	6,44E-14	4,66E-13	3,31E-12	6,92E-03	1,50E+00	4,68E+00
	En Kötü	4,44E-16	4,44E-16	4,44E-16	7,51E-14	1,04E-10	2,93E-07	4,13E-12	1,14E-11	6,14E-11	2,12E+00	1,26E+01	1,20E+01
F_{11}	**Ortalama**	**0,00E+00**	**0,00E+00**	**0,00E+00**	4,95E-03	2,59E-03	1,78E-03	**0,00E+00**	**0,00E+00**	**0,00E+00**	2,47E-02	4,03E+00	2,66E+01
	Standart Sapma	0,00E+00	0,00E+00	0,00E+00	9,87E-03	5,64E-03	4,71E-03	0,00E+00	0,00E+00	0,00E+00	1,20E-02	1,64E+01	3,69E+01
	En İyi	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	2,88E-13	0,00E+00	0,00E+00	0,00E+00	2,86E-03	8,79E-01	4,71E+00
	En Kötü	0,00E+00	0,00E+00	0,00E+00	3,98E-02	1,99E-02	1,68E-02	0,00E+00	0,00E+00	0,00E+00	5,22E-02	9,11E+01	1,05E+02
F_{12}	**Ortalama**	**3,76E-08**	**1,63E-07**	**6,36E-07**	4,39E-03	3,40E-02	2,03E-01	4,10E-03	2,57E-02	9,90E-02	1,74E-01	3,06E+00	1,35E+03
	Standart Sapma	4,91E-08	1,23E-07	5,64E-07	1,89E-02	2,43E-02	5,13E-02	3,92E-03	9,57E-03	2,05E-02	2,57E-01	1,10E+00	2,49E+03
	En İyi	7,65E-09	1,22E-08	5,11E-08	3,72E-06	1,06E-02	1,20E-01	2,82E-05	6,52E-03	6,07E-02	2,02E-05	1,16E+00	7,40E+01
	En Kötü	2,73E-07	6,85E-07	2,74E-06	1,04E-01	1,15E-01	3,39E-01	1,70E-02	4,11E-02	1,41E-01	9,58E-01	5,41E+00	1,32E+04
F_{13}	**Ortalama**	**4,75E-08**	**8,25E-08**	**1,62E-07**	1,22E-01	1,11E+00	6,07E+00	2,60E-01	1,59E+00	9,22E+00	1,72E-01	4,45E+01	9,25E+04
	Standart Sapma	4,22E-08	8,96E-08	3,27E-07	1,15E-01	2,90E-01	6,75E-01	2,67E-01	7,07E-01	7,42E-01	2,43E-01	1,78E+01	7,16E+04
	En İyi	3,84E-09	3,54E-09	8,27E-09	7,81E-05	5,78E-01	4,78E+00	9,78E-03	5,44E-01	5,76E+00	3,37E-04	1,30E+01	2,60E+04
	En Kötü	1,77E-07	3,68E-07	1,80E-06	3,60E-01	1,82E+00	7,55E+00	1,45E+00	3,58E+00	9,77E+00	9,04E-01	8,29E+01	3,65E+05

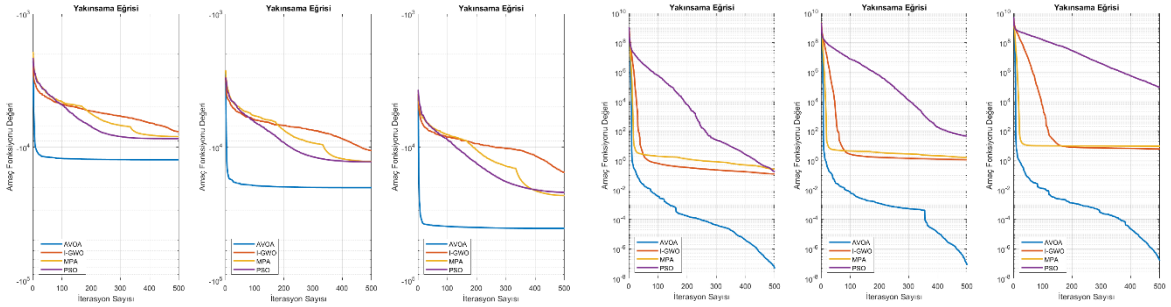
Tablo 11. *Farklı boyutlu çok modlu test fonksiyonları benzetim sonuçları*

Fonksiyonlar	Algoritmalar	AVOA	I_GWO	MPA	PSO
F_{14}	**Ortalama**	1,36E+00	**9,98E-01**	**9,98E-01**	**9,98E-01**
	Standart Sapma	7,59E-01	2,10E-16	1,27E-16	0,00E+00
	En İyi	9,98E-01	9,98E-01	9,98E-01	9,98E-01
	En Kötü	2,98E+00	9,98E-01	9,98E-01	9,98E-01
F_{15}	**Ortalama**	4,04E-04	9,77E-04	**3,07E-04**	3,25E-03
	Standart Sapma	1,27E-04	3,66E-03	1,75E-15	6,89E-03
	En İyi	3,08E-04	3,07E-04	3,07E-04	3,08E-04
	En Kötü	7,41E-04	2,04E-02	3,07E-04	2,26E-02
F_{16}	**Ortalama**	**-1,03E+00**	**-1,03E+00**	**-1,03E+00**	**-1,03E+00**
	Standart Sapma	4,40E-16	6,58E-16	4,79E-16	6,52E-16
	En İyi	-1,03E+00	-1,03E+00	-1,03E+00	-1,03E+00
	En Kötü	-1,03E+00	-1,03E+00	-1,03E+00	-1,03E+00
F_{17}	**Ortalama**	**3,98E-01**	**3,98E-01**	**3,98E-01**	**3,98E-01**
	Standart Sapma	3,24E-16	0,00E+00	2,07E-14	0,00E+00
	En İyi	3,98E-01	3,98E-01	3,98E-01	3,98E-01
	En Kötü	3,98E-01	3,98E-01	3,98E-01	3,98E-01
F_{18}	**Ortalama**	**3,00E+00**	**3,00E+00**	**3,00E+00**	**3,00E+00**
	Standart Sapma	5,44E-06	1,96E-15	2,19E-15	1,66E-15
	En İyi	3,00E+00	3,00E+00	3,00E+00	3,00E+00
	En Kötü	3,00E+00	3,00E+00	3,00E+00	3,00E+00
F_{19}	**Ortalama**	**-3,86E+00**	**-3,86E+00**	**-3,86E+00**	**-3,86E+00**
	Standart Sapma	1,11E-12	2,54E-15	2,43E-15	2,65E-15
	En İyi	-3,86E+00	-3,86E+00	-3,86E+00	-3,86E+00
	En Kötü	-3,86E+00	-3,86E+00	-3,86E+00	-3,86E+00
F_{20}	**Ortalama**	-3,27E+00	**-3,32E+00**	**-3,32E+00**	-3,27E+00
	Standart Sapma	6,04E-02	2,15E-02	2,17E-02	7,26E-02
	En İyi	-3,32E+00	-3,32E+00	-3,32E+00	-3,32E+00
	En Kötü	-3,20E+00	-3,20E+00	-3,20E+00	-3,14E+00
F_{21}	**Ortalama**	**-1,02E+01**	-9,92E+00	**-1,02E+01**	-6,40E+00
	Standart Sapma	3,44E-13	9,65E-01	3,61E-11	3,64E+00
	En İyi	-1,02E+01	-1,02E+01	-1,02E+01	-1,02E+01
	En Kötü	-1,02E+01	-5,22E+00	-1,02E+01	-2,63E+00
F_{22}	**Ortalama**	**-1,04E+01**	**-1,04E+01**	**-1,04E+01**	-8,27E+00
	Standart Sapma	6,91E-13	9,71E-05	1,34E-12	3,36E+00
	En İyi	-1,04E+01	-1,04E+01	-1,04E+01	-1,04E+01
	En Kötü	-1,04E+01	-1,04E+01	-1,04E+01	-2,75E+00
F_{23}	**Ortalama**	**-1,05E+01**	**-1,05E+01**	**-1,05E+01**	-8,15E+00
	Standart Sapma	4,16E-13	8,64E-09	1,46E-11	3,72E+00
	En İyi	-1,05E+01	-1,05E+01	-1,05E+01	-1,05E+01
	En Kötü	-1,05E+01	-1,05E+01	-1,05E+01	-2,42E+00



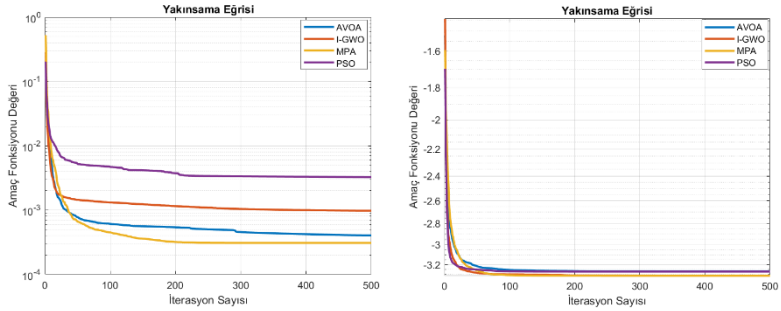
(a) ***(b)***

Şekil 4. *Bazı tek modlu fonksiyonlar için yakınsama grafikleri (**(a)** F_3, **(b)** F_5)*



(a) ***(b)***

Şekil 5. *Bazı çok modlu fonksiyonlar için yakınsama grafikleri (**(a)** F_8, **(b)** F_{13})*



(a) ***(b)***

Şekil 6. *Bazı farklı modlu fonksiyonlar için yakınsama grafikleri (**(a)** F_{15}, **(b)** F_{20})*

Şekil 5'de F_8 fonksiyonunda AVOA neredeyse ilk 10 iterasyon sonrasında gelişim sağlayamamıştır. MPA süreksiz bir yakınsama eğrisine sahiptir. Bu fonksiyon için PSO'da AVOA ile benzer olarak 30 ve 60 boyutları için 300 iterasyondan sonra iyileştirme sağlayamadığı yorumu yapılabilir. F_{13}'de ise PSO tüm iterasyonlarda iyileştirme sağlamasına rağmen, 50 ve 100 boyutları için diğer algoritmaların gerisinde kalmıştır. Burada I-GWO ve MPA her boyut için farklı olmakla birlikte tüm iterasyonlarında iyileştirme sağlayamadılar. AVOA hem yakınsama hızı ve bulduğu amaç fonksiyonu değeri açısından diğer algoritmalardan daha başarılıdır.

Şekil 6'da çalışılan fonksiyonlar kendi tanımlandığı tek bir boyut için çalıştırıldığından her fonksiyon için grafik sayısı tektir. F_{15}'de algoritmaların başarı sıralaması yüksekten düşüğe doğru MPA, AVOA, I-GWO ve PSO'dur. Algoritmaların tamamı yaklaşık 100. iterasyondan sonra gelişim sağlayamamıştır. F_{20} fonksiyonu için de benzer bir durum söz konusudur. Burada algoritmalar 100. iterasyonun çok gerisinde bulabildikleri en iyi değerleri ulaşmışlardır. I-GWO ve MPA; AVOA ve PSO'nun amaç fonksiyonu değerleri birbirlerine oldukça yakındır. Bununla birlikte I-GWO ve MPA'nın daha başarılı olduğu grafikten çıkarılabilir.

D. İSTATİKSEL TEST SONUÇLARI

D. 1. Friedman Testi

Sıralamalı iki yönlü varyans analizi olarak da adlandırılan Friedman testi, parametrik olmayan bir istatistik testidir. Bu testin kullanmasının amacı, çalışmada deneylerini yapılan farklı algoritmaların sonuçlarının tek bir anakütleden mi geldiğini yoksa ayrı anakütlelerden mi geldiğini incelemektir. Diğer bir ifadeyle algoritmaların sonuçları arasında anlamlı fark olup olmadığını test etmektir. Friedman testini yapabilmek için deneylerde kullanılan toplam 23 fonksiyonun tüm koşmalar için sonuçları birleştirildi. 4 farklı algoritma için her bir sütun bir algoritmanın sonucunu ifade etmek üzere toplamda $23 * 30 = 690$ satır, 4 sütun büyüklüğünde bir örneklem elde ettik. Friedman testi, sütun etkilerinin hepsinin aynı olduğu sıfır hipotezini, hepsinin aynı olmadığı alternatifine karşı test ederek p değeri hesaplar. Test sonucunda p değerini 5.09E-70 olarak hesaplandı. Bu değer, sonuçların başarısının sadece rastgelelikten değil, algoritmanın farklı olmasından da kaynaklandığını göstermektedir. Bu nedenle ikinci bir test olan Wilcoxon Tek Kuyruklu İşaretli Sıralar Testi'ni de gerçekleştirilmiş ve diğer alt bölümde sunulmuştur.

D. 2. Wilcoxon Tek Kuyruklu İşaretli Sıralar Testi

Benzetim sonuçları ve yakınsama grafikleri birlikte değerlendirildiğinde AVOA ve MPA'nın diğer algoritmalara nazaran daha başarılı olduğu söylenebilir. Bu nedenle bu algoritmalarla diğerleri arasında anlamlı bir fark olup olmadığı 0,05 anlamlılık düzeyinde diğer boyutlarda benzer başarılara sahip olduğundan F_1 ve F_{13} arasındaki tüm fonksiyonlar için 30 boyutlu değerleri kullanılarak test yapılmıştır. Wilcoxon tek kuyruklu işaretli sıralar testi yapılarak değerlendirilmiştir. Tüm koşmaların ortalama değerleri üzerinden AVOA ve MPA için gerçekleştirilen istatiksel test sonuçları Tablo 12'de paylaşılmıştır. Tabloda belirtilen h'ın 1'e eşit olması H_0 hipotezinin reddedileceği anlamına gelir. Diğer bir deyişle algoritmanın ortalamalarının kıyaslanan algoritmanın ortalamalarından daha küçük olduğu anlamına gelir.

Tablo 12'deki istatiksel test sonuçları değerlendirildiğinde AVOA'nın 16 fonksiyonda I-GWO'dan, 14 fonksiyonda MPA'dan ve 15 fonksiyonda PSO'dan daha düşük ortalama değerleri ile daha başarılı olduğu görülmektedir. MPA kıyaslamalarında ise 5 fonksiyonda AVOA'dan, 9 fonksiyonda I-GWO'dan, 12 fonksiyonda PSO'dan daha iyi sonuçlara sahiptir. F_9, F_{11}, F_{16} ve F_{22} fonksiyonları için AVOA ve MPA algoritmaları birbirlerine üstünlük sağlayamamıştır.

IV. SONUÇ

Bu çalışmada son yıllarda önerilmiş AVOA, I-GWO ve MPA'nın temel bir metasezgisel olan PSO ile karşılaştırılması amaçlanmıştır. Her bir metasezgisel ile ilgili açıklayıcı bilgiler okuyuculara sunulmuştur. Bildiğimiz kadarıyla literatürde bu metasezgiseller ilk kez kıyaslanmaktadır. Deneylerde, 23 farklı test fonksiyonu kullanılmıştır. Algoritmalar farklı performans kriterleri ile karşılaştırıldığında AVOA ve MPA'nın diğerlerine kıyasla daha üstün başarı gösterdiği anlaşılmıştır. Başarının anlamlı olup olmadığı Wilcoxon işaretli sıralar testi yapılarak analiz edilmiştir. Tüm deneysel çalışmalar birlikte yorumlandığında bu dört metasezgisel algoritmanın başarısı sırasıyla AVOA, MPA, I-GWO ve PSO sonucuna varılabilir.

Tablo 12. *Wilcoxon istatiksel test sonuçları*

Fonksiyonlar	AVOA-I-GWO		AVOA-MPA		AVOA-PSO		MPA-AVOA		MPA-I-GWO		MPA-PSO	
	p	h	p	h	p	h	p	h	p	h	p	h
F_1	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	1,00E+00	0	9,13E-07	**1**
F_2	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	1,00E+00	0	9,13E-07	**1**
F_3	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	1,01E-01	0	9,13E-07	**1**
F_4	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	9,13E-07	**1**	9,13E-07	**1**
F_5	1,01E-06	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	1,00E+00	0	2,74E-06	**1**
F_6	9,13E-07	**1**	1,01E-06	**1**	9,13E-07	**1**	1,00E+00	0	1,00E+00	0	1,00E+00	0
F_7	9,13E-07	**1**	1,12E-06	**1**	9,13E-07	**1**	1,00E+00	0	1,67E-06	**1**	9,13E-07	**1**
F_8	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	2,79E-02	**1**	9,91E-01	0
F_9	9,13E-07	**1**	1,00E+00	0	9,13E-07	**1**	1,00E+00	0	9,13E-07	**1**	9,13E-07	**1**
F_{10}	8,33E-07	**1**	9,12E-07	**1**	9,13E-07	**1**	1,00E+00	0	1,00E+00	0	9,13E-07	**1**
F_{11}	3,91E-03	**1**	1,00E+00	0	9,13E-07	**1**	1,00E+00	0	3,91E-03	**1**	9,13E-07	**1**
F_{12}	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	9,99E-01	0	8,25E-04	**1**
F_{13}	9,13E-07	**1**	9,13E-07	**1**	9,13E-07	**1**	1,00E+00	0	9,98E-01	0	9,81E-01	0
F_{14}	1,00E+00	0	1,00E+00	0	1,00E+00	0	1,95E-03	**1**	1,00E+00	0	1,00E+00	0
F_{15}	1,00E+00	0	1,00E+00	0	6,21E-04	**1**	9,13E-07	**1**	9,13E-07	**1**	9,12E-07	**1**
F_{16}	1,00E+00	0	1,00E+00	0	1,00E+00	0	1,00E+00	0	1,00E+00	0	1,00E+00	0
F_{17}	1,00E+00	0	3,91E-03	**1**	1,00E+00	0	1,00E+00	0	1,00E+00	0	1,00E+00	0
F_{18}	1,00E+00	0	1,00E+00	0	1,00E+00	0	9,13E-07	**1**	9,92E-01	0	1,00E+00	0
F_{19}	1,00E+00	0	1,00E+00	0	1,00E+00	0	9,59E-06	**1**	1,00E+00	0	1,00E+00	0
F_{20}	9,98E-01	0	1,00E+00	0	8,71E-01	0	3,45E-04	**1**	2,69E-01	0	4,92E-01	0
F_{21}	9,13E-07	**1**	7,05E-05	**1**	4,46E-03	**1**	1,00E+00	0	3,33E-06	**1**	4,49E-03	**1**
F_{22}	9,13E-07	**1**	3,41E-01	0	3,71E-01	0	6,67E-01	0	9,13E-07	**1**	3,33E-01	0
F_{23}	1,24E-06	**1**	8,96E-03	**1**	4,92E-01	0	9,92E-01	0	1,24E-06	**1**	3,88E-01	0

V. KAYNAKLAR

[1] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Computers & Industrial Engineering*, vol. 137, no. 5, 2019.

[2] F. S. Gharehchopogh, H. Shayanfar, and H. Gholizadeh, "A comprehensive survey on symbiotic organisms search algorithms," *Artificial Intelligence Review*, vol. 53, no. 56, pp. 1–48, 2020.

[3] K. Hussain, M. N. M. Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artificial Intelligence Review*, vol. 52, no. 4, pp. 2191–2233, 2019.

[4] V. Stojanovic, S. He, and B. Zhang, "State and parameter joint estimation of linear stochastic systems in presence of faults and non-Gaussian noises," *International Journal of Robust and Nonlinear Control*, vol. 30, no. 16, pp. 6683–6700, 2020.

[5] B. Abdollahzadeh, and F. S. Gharehchopogh, "A multi-objective optimization algorithm for feature selection problems," *Engineering with Computers*, pp. 1–19, 2021.

[6] F. S. Gharehchopogh, I. Maleki, and Z. A. Dizaji, "Chaotic vortex search algorithm: Metaheuristic algorithm for feature selection," *Evolutionary Intelligence*, pp. 1–32, 2021.

[7] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, pp. 1942–1948.

[8] C. W. Cleghorn, and B. Stapelberg, "Particle swarm optimization: stability analysis using n-informers under arbitrary coefficient distributions," *Swarm and Evolutionary Computation*, vol. 71, 2022.

[9] P. Hu, J.S. Pan, S. C. Chu, and C. Sun, "Multi-surrogate assisted binary particle swarm optimization algorithm and its application for feature selection," *Applied Soft Computing*, vol. 121, 2022.

[10] X. Chen, and K. Li, "Collective information-based particle swarm optimization for multi-fuel CHP economic dispatch problem," *Knowledge-Based Systems*, vol. 248, 2022.

[11] P. B. Fernandes, R. C. L. Oliveira, and J. F. Neto, "Trajectory planning of autonomous mobile robots applying a particle swarm optimization algorithm with peaks of diversity," *Applied Soft Computing*, vol. 116, 2022.

[12] F. Wang, X. Wang, and S. Sun, "A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization," *Information Sciences*, vol. 602, pp. 298-312, 2022.

[13] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems," *Computers & Industrial Engineering*, vol. 158, 2021.

[14] H. A. Bagal, Y. N. Soltanabad, M. Dadjuo, K. Wakil, M. Zare, and A. S. Mohammed, "SOFC model parameter identification by means of Modified African Vulture Optimization algorithm," *Energy Reports*, vol. 7, pp. 7251-7260, 2021.

[15] Y. Wang, S. Li, H. Sun, C. Huang, and N. Youssefi, "The utilization of adaptive African vulture optimizer for optimal parameter identification of SOFC," *Energy Reports*, vol. 8, pp. 551-560, 2022.

[16] Y. Chen, and G. Zhang, "New parameters identification of Proton exchange membrane fuel cell stacks based on an improved version of African vulture optimization algorithm," *Energy Reports*, vol. 8, pp. 3030-3040, 2022.

[17] M. Alanazi, A. Fathy, D. Yousri, and H. Rezk, "Optimal reconfiguration of shaded PV based system using African vultures optimization approach," *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 12159-12185, 2022.

[18] Y. Wang, J. Wang, L. Yang, B. Ma, G. Sun, and N. Youssefi, "Optimal designing of a hybrid renewable energy system connected to an unreliable grid based on enhanced African vulture optimizer," *ISA Transactions*, 2022.

[19] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46-61, 2014.

[20] M. H. Nadimi-Shahraki, S. Taghian, and S. Mirjalili, "An improved grey wolf optimizer for solving engineering problems," *Expert Systems with Applications*, vol. 166, 2021.

[21] D. Hua, X. Liu, S. Sun, Z. Li, Z. Li and W. Li, "Precise locomotion controller design for a novel magnetorheological fluid robot based on improved gray wolf optimization algorithm," *Smart Materials and Structures*, vol. 30, no. 2, 2021.

[22] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine Predators Algorithm: A nature-inspired metaheuristic," *Expert Systems with Applications*, vol. 152, 2020.

[23] M. Abd Elaziz, D. Mohammadi, D. Oliva, and K. Salimifard, "Quantum marine predators algorithm for addressing multilevel image segmentation," *Applied Soft Computing*, vol. 110, 2021.

[24] Z. Xing, and Y. He, "Many-objective multilevel thresholding image segmentation for infrared images of power equipment with boost marine predators algorithm," *Applied Soft Computing*, vol. 113, 2021.

[25] A. S. Sadiq, A. A. Dehkordi, S. Mirjalili, and Q. V. Pham, "Nonlinear marine predator algorithm: A cost-effective optimizer for fair power allocation in NOMA-VLC-B5G networks," *Expert Systems with Applications*, vol. 203, 2022.

[26] M. H. Hassan, D. Yousri, S. Kamel, and C. Rahmann, "A modified marine predators algorithm for solving single-and multi-objective combined economic emission dispatch problems," *Computers & Industrial Engineering*, vol. 164, 2022.

[27] E. H. Houssein, I. E. Ibrahim, M. Kharrich, and S. Kamel, "An improved marine predators algorithm for the optimal design of hybrid renewable energy systems," *Engineering Applications of Artificial Intelligence*, vol. 110, 2022.

[28] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82-102, 1999.

[29] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: theory and application," *Advances in Engineering Software*, vol. 105, pp. 30-47, 2017.

[30] F. MiarNaeimi, G. Azizyan, and M. Rashki, "Horse herd optimization algorithm: a nature-inspired algorithm for high-dimensional optimization problems," *Knowledge-Based Systems*, vol. 213, 2021.