Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi
Afyon Kocatepe University – Journal of Science and Engineering
https://dergipark.org.tr/tr/pub/akufemubid

Araştırma Makalesi / Research Article
DOI: https://doi.org/10.35414/akufemubid.1387447
AKÜ FEMÜBİD **24** (2024) 035902 (726-748)
AKU J. Sci. Eng. **24** (2024) 035902 (726-748)

TOPSIS Yaklaşımı ile Metasezgisel Optimizasyon Algoritmalarının Performans Değerlendirmesi

Makale Bilgisi / Article Info
Alındı/Received: 07.11.2023
Kabul/Accepted: 07.05.2024
Yayımlandı/Published: 27.06.2024

Performance Evaluation of Meta-Heuristic Optimization Algorithms with The TOPSIS Approach

Şehmus FİDAN[1*], Metin ZALOĞLU[2], Emre ERKAN[3]

[1] *Batman Üniversitesi, Teknik Bilimler MYO/Elektronik Programı, Türkiye*
[2] *Batman Üniversitesi, Elektrik Elektronik Mühendisliği / Lisansüstü Eğitim Enstitüsü, Türkiye*
[3] *Batman Üniversitesi, Teknik Bilimler MYO/Elektronik-Haberleşme Programı, Türkiye*

Öz

Bir sistemin sadece giriş/çıkış verilerinin kullanılarak matematiksel bir model elde etmek için doğadan ilham alan metasezgisel algoritmalar kullanılabilir. Bunu gerçekleştirmek için yapay ekosistem (YEA), çiçek tozlaşma (ÇTA), güve-alev (GAA), karınca aslanı algoritması (KAA), halat çekme (HÇA), atom arama (AAA), beyin fırtınası (BFA), su döngüsü (SDA), mercan resifleri (MRA) ve yaşam seçimi tabanlı algoritma (YSTA) gibi çeşitli metasezgisel optimizasyon algoritmaları ele alınmış ve önerilen transfer fonksiyonunun parametrelerini optimize etmek için kullanılmıştır. Ayrıca zaman, maksimum fonksiyon, erken durdurma ve maksimum generasyon sınırlılıkları altında performanslar karşılaştırılmıştır. Ancak bu durumda MAE, MAPE, R^2 gibi performans metriklerinin yanında transfer fonksiyonlarına özgü yükselme zamanı, oturma zamanı, aşım miktarı gibi metrikler de ortaya çıkmaktadır. Çok sayıdaki metrik hangi algoritmanın en iyi olduğunu belirlemeyi zorlaştırmaktadır. Bu zorluğun üzerinden gelmek için bu çalışmada Topsis (Technique for Order Preference by Similarity) olarak anılan çok kriterli bir karar verme yaklaşımının kullanımını önerilmiştir. Çoklu kriter için algoritmanın çözüm zamanı, performans (R^2) ve yükselme zamanı dikkate alınmıştır. Yapılan çalışma neticesinde en iyi algoritma sıralamasını belirlemek oldukça kolay ve pratik bir şekilde gerçekleştirilmiştir.

Anahtar Kelimeler *Metasezgisel Algoritma; Sistem Tanımlama; Transfer Fonksiyonu; Topsis.*

Abstract

Meta-heuristic algorithms inspired by nature can be utilized to derive a mathematical model of a system based on input/output data. To achieve this, various meta-heuristic optimization algorithms such as artificial ecosystem optimization (AEO), flower pollination algorithm (FPA), ant lion optimizer (ALO), moth-flame optimization (MFO), tug of war optimization (TWO), atomic search optimization (ASO), brain storm optimization (BSO), water cycle algorithm (WCA), coral reefs optimization (CRO), and life choice-based optimization (LCO) have been considered and employed to optimize the parameters of the proposed transfer function. Additionally, their performances have been compared under constraints such as time, maximum function evaluations, early stopping, and maximum generations. However, in this context, alongside performance metrics such as MAE, MAPE, and R^2, metrics specific to transfer functions like rise time, settling time, and overshoot also emerge. The multitude of metrics makes it challenging to determine which algorithm performs best. To overcome this difficulty, the use of a multi-criteria decision-making approach known as Topsis (Technique for Order Preference by Similarity) is proposed in this study. The algorithm's solution time, performance (R^2), and rise time have been considered for multiple criteria. As a result of the study, determining the best algorithm ranking has been accomplished in a straightforward and practical manner.

Keywords: *Meta-heuristic Algorithm; System Identification; Transfer Function; Topsis.*

1. Giriş

Karmaşık ve çok boyutlu optimizasyon problemlerinin çözümünde doğadan ilham alarak geliştirilen metasezgisel (MS) algoritmalar giderek popüler hale gelmektedir. Bu algoritmalar, global çözüme yaklaşmaya çalışırken deterministik olmayan yöntemler kullanırlar (İzci vd. 2022). Birçok farklı MS algoritması bulunmakla birlikte yapay arı koloni algoritması (Karaboğa ve Baştürk 2007), karınca kolonisi optimizasyonu (Dorigo vd. 2006), parçacık sürü optimizasyonu (Kennedy ve Eberhart 1995) vb. algoritmalar örnek olarak verilebilir. Bu algoritmaların her biri, benzemeye çalıştığı doğal sistemlerden farklı özelliklere sahiptir. Bu özellikleri ile MS algoritmaları farklı optimizasyon problemlerine uyarlanabilirler (Yang 2020). MS algoritmaları bir sistemin matematiksel modelini elde etmek için kullanılabilir. Bu yaklaşım, özellikle matematiksel modelin elde edilmesinin zor olduğu karmaşık sistemlerde faydalı olabilir (Ding vd. 2015). Bir sistemden ölçülen giriş/çıkış verilerinin kullanılmasıyla matematiksel model üretmenin önemli ve etkili yöntemlerinden biri sistem tanımlamadır. Sistem

*Sorumlu Yazar/Corresponding Author: Şehmus FİDAN e-posta/e-mail: sehmus.fidan@gmail.com

tanımlama yöntemleri gerçek sistemlerin giriş/çıkış verilerini kullanarak tahmini matematiksel model elde etmek için kullanılır ve bu yöntemlerin birçok farklı alan için önemli uygulamaları vardır. Parametrik yöntemler sistemin matematiksel modele sahip olması durumunda, parametrik olmayan yöntemler ise modele bağlı olmadan sistemin davranışlarını tahmin etmek için kullanılır. Her iki yöntemde giriş/çıkış verilerini kullanarak sistemin matematiksel modelini oluşturmak, sistemin davranışını, özelliklerini ve performansını analiz etmek için kullanılır (Ji vd. 2020). MS algoritmalar, bir sistemin gerçek davranışına daha yakın bir matematiksel model elde etmek için kullanılabilir. Bu algoritmalar, sistem davranışının matematiksel olarak ifade edilmesi için önerilen transfer fonksiyonlarının en uygun parametrelerini belirlemek üzere iteratif bir süreç uygularlar (Mohammadi vd. 2022). Evrimsel, sürü, fizik, insan, biyolojik, sistem ve matematik temelli olarak 6 kategoriye ayrılan MS algoritmalar; optimizasyon, makine öğrenimi, veri madenciliği ve sistem tasarımı alanlarında sıklıkla kullanılmaktadır.

Yapay ekosistem tabanlı optimizasyon algoritması (YEA), canlı organizmaların üretim, tüketim ve ayrışma davranışlarını örnek alır. El-Dabah vd. (2021), PV'lerin üç diyotlu modelinin bilinmeyen parametrelerini belirlemek için YEA tabanlı bir çözüm önermişlerdir. Omotoso vd. (2022), akıllı mikro şebekelerin entegrasyonu ve optimal boyutlandırma problemini çözmek için YEA önermişlerdir. Dağıtım ağlarının işletilmesi ve olası güç kayıplarını azaltma üzerine çalışmaları olan Nguyen (2023) YEA yöntemine dayanan yeniden ağ yapılandırma yöntemini sunmuştur. İzci vd. (2022), bir düşürücü tip DA-DA (doğru akım) dönüştürücünün çıkış voltajının kontrolünde en uygun PID (proportional integral ve derivative-PID) denetleyici parametrelerini belirlemek için nelder-meat ve yapay ekosistem algoritması (YEA-NM) tabanlı bir algoritma sunmuştur.

Çiçek tozlaşma algoritması (ÇTA), çiçeklerin yayılma yönteminden esinlenerek geliştirilen bir MS optimizasyon algoritmasıdır (Yang 2012). ÇTA, esnek kiriş sisteminde (Fadzli vd. 2022), IIR filtrenin (Infinite Impulse Response) modelinin belirlenmesinde (S. Singh vd. 2016), sıcaklık kontrol sistemini tanımlamada (Sompracha ve Rukkaphan 2021) ve da motor model parametrelerinin belirlenmesinde kullanılmıştır (Puangdownreong vd. 2017). Ayrıca fırçasız da motorun (FDA) motorun kısmı kesirli model parametrelerinin elde edilmesinde (Khluabwannarat vd. 2018), kısmı kesirli kaotik harita modelinin oluşturulmasında (Yousri vd. 2020) ve hiper-kaotik bir sistemin model parametrelerinin tahmininde (Chen vd. 2019) kullanılmıştır. Karınca aslanı algoritması (KAA), karınca aslanlarının doğadaki avlanma mekanizmasını simüle eder (Mirjalili 2015b). KAA algoritması hidrolik türbin tahrik sisteminde (Tian vd. 2018), IIR filtresinin parametrelerinin belirlenmesinde (Nair vd. 2017) ve PV hücre modelinin elde edilmesinde (Wu vd. 2019) kullanılmıştır.

Güve-alev optimizasyonu (GAA), yapay ışık kaynakları etrafında güvelerin uçuşundan ilham alan bir MS optimizasyon algoritmasıdır. GAA algoritması, arama aracısı olarak güvelerle arama uzayını araştırır ve keşif tamamlanana kadar her adımda en iyi çözümü günceller (Mirjalili 2015a). GAA algoritmaları lineer olmayan sistem tanımlamada (Canayaz 2019), iletim hattı parametrelerinin tanımlanmasında (Shaikh vd. 2023), LCL filtrenin modelinin belirlenmesinde (Long vd. 2022), tek fazlı inverter filtresinin hesaplanmasında (Wu vd. 2017) ve kalman filtresinin sistem parametrelerinin tahmin edilmesinde (Janjanam vd. 2022) kullanılmıştır.

Halat çekme algoritmasında (HÇA), her aday bir ip çekme yarışmasında takım olarak kabul edilir ve çözüm kalitelerine göre takımların yeni konumlarını belirlemek için Newton yasaları kullanılmaktadır. Kaveh ve Bakhshpoori (2021) açıklıkları ve kirişleri olan mazgallı kirişlerin tasarımını optimize etmek için bu algoritmayı kullanmışlardır. Atom arama algoritması (AAA), Lennard-Jones ve bağ uzunluğu potansiyellerini kullanarak atom etkileşimlerini modeller ve çok çeşitli optimizasyon problemlerine uygulanabilir (Zhao vd. 2019). AAA, proton değişim membran yakıt hücre (PEMFC- proton exchange membrane fuel cells) modelinin parametrelerinin belirlenmesinde (Mossa vd. 2021) ve lineer-periyodik sistem tanımlama (Yin vd. 2020) gibi problemlere uygulanmıştır. Beyin fırtınası optimizasyonu (BFA) algoritması, insana özgü olan beyin fırtınasını taklit ederek çok amaçlı optimizasyon problemlerini çözmek için geliştirilmiştir (Shi 2011). BFA, PV modelinin parametrelerinin hesaplanmasında (Yan vd. 2019) ve lineer olmayan modellerde (NARMAX) çalışılmıştır (Pal vd. 2016).

Eskandar vd. (2012), doğal çevrede gözlemlenen idealize hidrolojik döngüden esinlenerek geliştirilen MS algoritması olan su döngüsü algoritmasını (SDA) tanıtmışlardır. SDA tek ve çift diyotlu PV sistem model parametrelerinin belirlenmesinde (Kler vd. 2017) ve hibrit PV/Rüzgâr sistem modelinin elde edilmesinde (Nazir vd. 2022) kullanılmıştır. Mercan resifleri algoritması (MRA), mercanların bulunduğu resifte çoğalmak için diğer mercanlarla olan mücadeleyi modellemektedir (Salcedo-Sanz vd. 2014). MRA algoritması manyetik tahrik

sisteminin model parametrelerinin belirlenmesinde (Y. Yang vd. 2017) ve adaptif IIR tabanlı sistem tanımlama amaçlı olarak çalışılmıştır (Y. Yang vd. 2018). Yaşam seçim tabanlı algoritma (YSTA), insanların hedeflerine ulaşmak için diğer üyelerin öğrenme yeteneğine dayanan algoritmalardan biridir (Khatri vd. 2020). Çok kriterli karar verme, farklı amaçları ve kısıtlamaları dengelemek, karmaşık seçimlerde daha bütünsel ve bilinçli kararlar almak için kullanılan bir süreç olarak ifade edilebilir. Bu süreç, çeşitli amaçları ve kısıtlamaları dengelemek, karar vericilere daha bütünsel bir perspektif sunmak ve sonuç olarak daha bilinçli kararlar almak amacıyla kullanılır. Bu noktada, çeşitli çok kriterli karar verme yöntemleri devreye girmektedir. Çok kriterli karar verme yöntemleri arasında AHP, Topsis, Vikor, Electre, Promethee gibi yöntemler bulunur (Paul vd. 2021). Topsis yöntemi AHP yönteminin karmaşıklığına göre daha basit ve anlaşılır bir yapıdadır. Electre yöntemi ile kıyaslandığında ideal ve ideal olmayan çözümlere olan benzerlik üzerinden bir sıralama sunmaktadır. Vikor yönteminde ise çok daha karmaşık denklemler bazen karmaşıklığa sebep olabilmektedir (Ceballos vd. 2016). Bu makalede kolayca uygulanabilmesi sebebiyle Topsis yöntemine odaklanılmış ve sistem tanımlama amaçlı olarak üretilen modellerin en iyisinin hangisinin olduğuna karar vermek için kullanılmıştır. Topsis, gri kurt algoritması (GWO) ile birleştirilerek aşındırıcı jet işleme sürecinin parametrik optimizasyonu için kullanılmıştır (Kalita vd. 2022). Singh vd. (2020), dağıtılmış enerji kaynaklarının şebekeye optimal entegrasyonunu sağlamak için çok kriterli karar verme yöntemi kullanmıştır. Topsis yöntemi ayrıca pareto optimal çözümü bulmada (Farag vd. 2020), 6 farklı MS algoritmasının en iyisini belirlemede (Shadkam vd. 2021), işletmelerde MS ile ortaya çıkan çözümlerin en iyisini bulmada (Crispim ve Pinho 2009), sürdürülebilir tedarik zinciri ağı tasarımının karar verme sürecinde (Guo vd. 2022) kullanılmıştır. Sistem tanımlama problemlerinin çözümü için önerilen transfer fonksiyonlarının zamana ve performansa dayalı farklı birçok performans parametresi bulunmaktadır. Yaygın olarak kullanılan Wilcoxon istatiksel testi tek bir parametre (R^2) kullanarak karşılaştırma yapmak için yetersizdir. Bu sebeple, sistem tanımlama amaçlı olarak kullanılan transfer fonksiyonlarının performansını karşılaştırmada ilk defa çok kriterli karar verme yöntemi olan Topsis önerilmiştir.

Bu çalışmanın özgün katkıları şu şekilde özetlenebilir:

1- MS algoritmaları giriş/çıkış verilerini kullanarak bir transfer fonksiyonu üretmek için kullanılmıştır. Böylece MS algoritmaların sistem tanımlama problemlerine uygulanabilirliği gösterilmiştir.

2-MS algoritmaları için kararlılık ve güvenilirlik analizleri kapsamında maksimum generasyon, durdurma sınırlılığı, fonksiyon hesaplama ve zaman sınırlılıkları dikkate alınarak performansları incelenmiştir.

3-10 farklı MS algoritması ve 4 farklı sınırlılık altında ortaya çıkan 40 farklı durumda en iyi algoritmayı bulmak için Topsis yöntemi uygulanmıştır.

4-Elde edilen transfer fonksiyonunun geçici durum cevapları üzerinden performansları incelenmiştir. Ayrıca çeşitli istatiksel performans göstergeleri (MAPE, MAE, MSE, R^2) sunularak sonuçları karşılaştırılmıştır.

2. Materyal ve Metot

Alt bölümlerde Topsis yöntemi ve metasezgisel algoritmaların detaylarından bahsedilmiştir. Ele alınan metasezgisel algoritmaların seçiminde güncel yöntemler olmasına dikkat edilmiştir. Ayrıca karşılaştırma yapabilmek için sistem tabanlı (yapay ekosistem, su döngüsü), evrim tabanlı (çiçek tozlaşma, mercan resifleri), sürü tabanlı (karınca aslanı, güve-alev), fizik tabanlı (halat çekme, atom arama) ve insan davranışı tabanlı (beyin fırtınası, yaşam seçim) algoritmalara odaklanılmıştır. Bu şekilde, çeşitli algoritmaların özellikleri özetlenerek karşılaştırma yapılabilecek bir çerçeve oluşturulmuştur.

2.1 Topsis

Topsis yöntemi, bir dizi alternatif arasından en iyi olanı seçmek için kullanılan çok kriterli karar verme yöntemidir (Fan vd., 2020). Bu yöntemde öncelikle bir karar verme matrisinin oluşturulması gereklidir. D karar matrisi:

$$D = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1\mathrm{n}} \\ x_{21} & x_{22} & \cdots & x_{2\mathrm{n}} \\ \vdots & \vdots & \vdots & \vdots \\ x_{\mathrm{m1}} & x_{\mathrm{m2}} & \cdots & x_{\mathrm{mn}} \end{bmatrix} \quad (1)$$

D matrisinde x_{mn} *m* alternatif sayısını ve *n* kriter sayısını temsil eder. D matrisinin normalizasyonu Denklem 2 ile gerçekleştirilir.

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \quad (2)$$

Ardından ağırlıklandırılmış normalize karar matrisinin oluşturulması için Denklem 3 kullanılır.

$$v_{ij} = w_j \cdot r_{ij} \quad (3)$$

v_{ij}, hesaplandıktan sonra ideal (A^+) ve negatif-ideal (A^-) çözümler Denklem 4-5'e göre hesaplanır.

$$A^+ = \{\max_i v_{ij} \mid j \in J_+\} \cup \{\min_i v_{ij} \mid j \in J_-\} \quad (4)$$

$$A^- = \{\min_i v_{ij} \mid j \in J_+\} \cup \{\max_i v_{ij} \mid j \in J_-\} \quad (5)$$

Burada, J_+ olumlu etkisi olan kriterlerle ilişkilendirilir. J_- ise olumsuz etkileri olan kriterler ile ilişkilidir. Bu aşamadan sonra ideal ve negatif ideal çözümlerin hesaplanması (Denklem 6-7) gereklidir.

$$S_i^+ = \sqrt{\sum_{j=1}^{n} \left(v_{ij} - A_j^+\right)^2} \quad (6)$$

$$S_i^- = \sqrt{\sum_{j=1}^{n} \left(v_{ij} - A_j^-\right)^2} \quad (7)$$

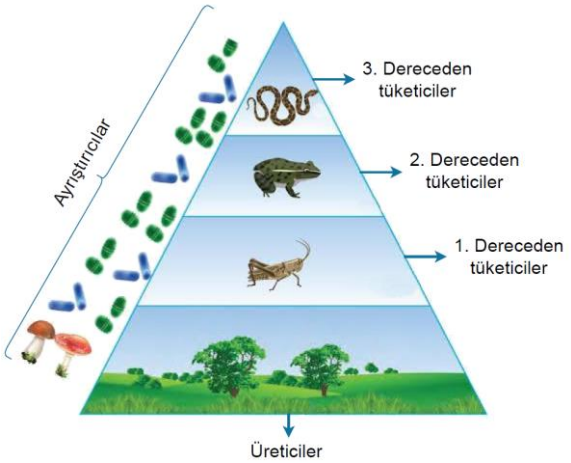
Son olarak, her bir alternatif çözümün ideal çözüme göreceli yakınlığı Denklem 8'e göre hesaplanır.

$$C_i = \frac{S_i^-}{S_i^+ + S_i^-} \quad (8)$$

2.2 Metasezgisel Algoritmalar

2.2.1 Yapay Ekosistem Tabanlı Optimizasyon

Yapay ekosistem tabanlı optimizasyon algoritması (YEA), ekosistemin enerji akışından ilham alarak önerilmiştir. YEA algoritması, canlı organizmaların üretim, tüketim ve ayrışma gibi üç benzersiz davranışını taklit etmektedir. Üreticiler, bitkiler gibi kendi yiyeceklerini üreten organizmalardır. Tüketiciler, kendi yiyeceklerini üretemeyen canlılardır; bu sebeple besin elde etmek için üretici veya tüketici türündeki canlılardan beslenirler. Tüketicileri etobur, otobur veya omnivor olarak sınıflandırmak mümkündür. Otobur tüketiciler, sadece bitkileri tüketirler. Etobur tüketiciler, sadece hayvanları tüketirler. Omnivor tüketiciler ise hem bitkileri hem de diğer hayvanları tüketirler. Ayrıştırıcılar ise ölü bitkiler ve hayvan atıkları gibi materyallerle beslenirler (İzci vd. 2022). YEA'ya dayalı bir ekosistemin temsili Şekil 1'de gösterilmektedir.



Şekil 1. Ekosistem besin piramidi (Prakash vd. 2020)

YEA'da üretici, gıda üretmek için üreticinin eylemini taklit eder. En düşük uygunluk değerine sahip üretici, bu bireyin ve en iyi bireyin (ayrıştırıcı) arama limitlerine göre güncellenir. Bu işlemin sonucu olarak, popülasyondaki diğer bireylerde konumlarını günceller. Üretim operatörü kullanılarak, en iyi olan (x_n) ile rastgele oluşturulmuş bir birey (x_{rand}) arasında yeni bir birey (üretici) üretilir. Bu ifade matematiksel olarak Denklem 9-11 arasındaki gibi ifade edilmektedir.

$$x_1(t+1) = (1-a) \cdot x_n(t) + a \cdot x_{rand}(t) \quad (9)$$

$$a = (1 - t/max_{iter}) \cdot r_1 \quad (10)$$

$$x_{rand} = r \cdot (U - L) + L \quad (11)$$

Üreticiler tarafından üretim işlemi gerçekleştirildikten sonra, tüketicilerin tüketim işlemi başlar. Bu süreçte her tüketici, daha düşük enerji seviyesine bağlı başka bir tüketiciden veya üreticiden enerjisini sağlayabilir. Levy flight adlı matematiksel bir operatör sayesinde birçok hayvanın gıda arama mekanizmasını taklit edilebilir. Bu durumda tüketim aşaması Denklem 12 ve 13'te gösterilmektedir.

$$C = \frac{1}{2}\frac{v_1}{|v_2|} \quad (12)$$

$$v_1 \sim N(0,1), v_2 \sim N(0,1) \quad (13)$$

Denklem 13'te, n (0,1) ortalaması 0 olan ve standart sapması 1 olan normal bir dağılımı temsil eder. Bu tüketim faktörü, her bir tüketicinin farklı av stratejileri kullanarak yiyecek temin etmesine yardımcı olacaktır. Tüketici rastgele otçul olarak seçilirse, o zaman sadece üreticiler yiyecektir. Bu durumda tüketicinin davranışı matematiksel olarak Denklem 14'te sunulmuştur.

$$x_i(t+1) = x_i(t) + C \cdot \left(x_i(t) - x_1(t)\right), i \in [2,..,n] \quad (14)$$

Bir tüketici, rastgele bir etobur olarak seçilirse daha yüksek enerji seviyesine sahip başka bir tüketiciyi yer. Bu durumda davranış Denklem 15 ile modellenebilir.

$$x_i(t+1) = x_i(t) + C \cdot \left(x_i(t) - x_j(t)\right), i \in [3,..,n] \quad (15)$$
$$j = randi([2i-1])$$

Ekosistem içinde hepçil (omnivor) seçilirse, yüksek enerji seviyesine sahip bir tüketici veya üreticiyi yiyebilir. Bu durumda matematiksel model Denklem 16'da gösterilmektedir.

$$x_i(t+1) = x_i(t) + C \cdot \left(r_2 \cdot \left(x_i(t) - x_1(t)\right)\right) + (1 - r_2)\left(x_i(t) - x_j(t)\right), i \in [3,..,n] \quad (16)$$
$$j = randi([2i-1])$$

Denklem 16'da r_2 değeri [0,1] aralığında rastgele bir sayıdır. Birey pozisyonlarının güncellenmesinde sırasıyla rastgele seçim veya en başarısız bireylerin tüketim operatörüne bakılır. Bu sayede global noktanın

araştırılması mümkün olmaktadır. Ayrıştırma sürecinde kullanılan h ve e ağırlık katsayıları Denklem 17-20'de verilmiştir. Bu parametreler *i.* bireyin x_i pozisyon bilgisini güncelleme için kullanılmaktadır.

$$x_i(t+1) = x_n(t) + D \cdot \left(e \cdot x_n(t) - h \cdot x_i(t)\right), i \in [1,..,n] \quad (17)$$

$$D = 3 \cdot u, u \sim N(0,1) \quad (18)$$

$$e = r_3 \cdot randi([1\ 2]) - 1 \quad (19)$$

$$h = 2 \cdot r_3 - 1 \quad (20)$$

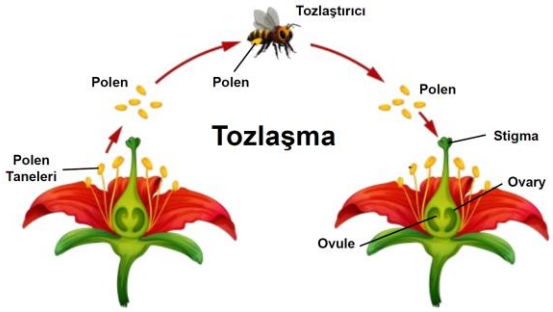
Şekil 2'de YEA'ya ilişkin sözde kodu verilmiştir. Bu kodda öncelikle popülasyon büyüklüğü, karar değişken sayısı ve maksimum yineleme sayısı belirlendikten sonra, rastgele başlatılan popülasyon içinde her bireyin uygunluğu hesaplanır. Ardından, belirli bir yineleme sınırına kadar bireylerin konumları rastgele güncellenir. Bu güncellemeler, rastgele seçilen denklemler aracılığıyla yapılır. Her yineleme sonrasında uygunluk fonksiyonları hesaplanır ve en iyi çözüm (X_{best}) güncellenir. Yineleme sınırına ulaşıldığında veya en iyi çözüm belirlendiğinde algoritma sona erer.

YEA Sözde Kodu
Popülasyon büyüklüğünü(n), karar değişken sayısını ve maksimum yineleme sayısını (tmax) belirle .
Ekosistem popülasyonunu rastgele başlat..
En iyi çözüm için Fiti uygunluk değerini ve X_{best} değerini hesapla.
Geçerli yinelemeyi t=1 olarak ayarla.
While *t < tmax ise*
X_1 bireysel çözümünü güncelle.
  if *rand<1/3 ise Denklem (6) yı kullanarak çözümü güncelle*
    else if *1/3<rand<2/3 aralığında ise Denklem(7) yi kullanarak çözümü güncelle*
    else *1/3<rand<2/3 aralığında değil ise Denklem(8) i kullanarak çözümü güncelle*
    end if
  end if
Her birey için uygunluk fonksiyonunu hesaplayın ve şimdiye kadar elde edilen en iyi X_{best} çözümünü güncelleyin
Denklem (9) u kullanarak bireyin konumunu güncelleyin
Her birey için uygunluk fonksiyonunu hesaplayın ve şimdiye kadar elde edilen en iyi X_{best} çözümünü güncelle
end while *t = t_{max} ise döngüyü bitir.*
t = t_{max} değil ise eşitlik oluşana kadar X_{best} e dön.

Şekil 2. YEA sözde kodu

2.2.2 Çiçek Tozlaşma Algoritması (ÇTA)

Çiçek tozlaşma algoritması, çiçekli bitkilerin üremesinden ilham alan bir optimizasyon yöntemi olarak ifade edilmektedir (Yang 2012). Optimum üreme sağlamak için biyotik ve abiyotik olmak üzere iki önemli tozlaşma şekli oluşur. Biyotik formda sinek, arı (polinatör) gibi canlılar polenleri taşır, abiyotik formda herhangi bir polinatöre ihtiyaç duyulmaksızın rüzgâr ve su gibi dinamiklerle yayılmasını sağlayan tozlaşma gerçekleşir. Şekil 3 çiçek tozlaşma stratejisini göstermektedir.



Şekil 3. Çiçek tozlaşma stratejisi (Yang 2012)

Çiçek tozlaşma algoritmasının matematiksel modeli Denklem 21 ve 22'de verilmiştir.

$$x^{t+1} = x_i^t + \gamma L(\lambda)(g_* - x_i^t) \quad (21)$$

$$L \sim \frac{\lambda \Gamma(\lambda) sin\left(\frac{\pi\lambda}{2}\right)}{\pi} \frac{1}{s^{1+\lambda}}, (s \gg s_0 > 0) \quad (22)$$

Burada, *L,* Levy dağılımının matematiksel ifadesini, *Γ (λ),* standart gama fonksiyonunu sembolize eder. Ayrıca *s*, adım büyüklüğünü oluşturur. Çiçek tozlaşmasında yeni neslin hesaplanmasında Denklem 23 kullanılmaktadır.

$$x^{t+1} = x_i^t + \epsilon\left(x_j^t - x_k^t\right) \quad (23)$$

ÇTA'da başlangıçta rastgele bir çiçek popülasyonu oluşturulur ve her çiçeğin olasılık anahtarına bağlı olarak genetik operatörler uygulanır. Bu operatörler, çiçeklerin birbirleriyle bilgi paylaşmasını simüle eder. İteratif olarak en iyi çözümü güncelleyen algoritma, genetik çeşitlilik ve işbirliğini birleştirerek karmaşık optimizasyon problemlerine uygun çözümler üretir. Şekil 4'te çiçek tozlaşma algoritmasının sözde kodu sunulmuştur.

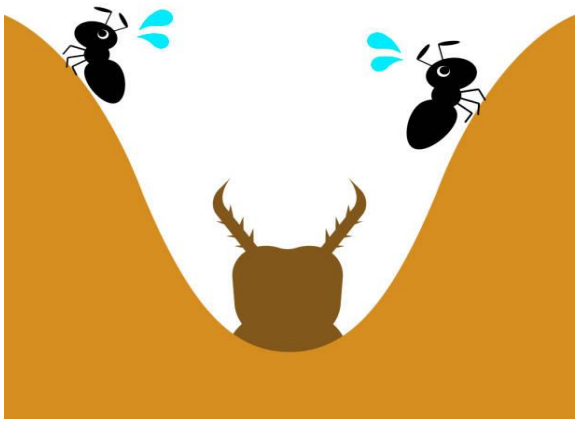
ÇTA Sözde Kodu
Rastgele başlangıç popülasyonunu üret. (n: Polen sayısı)
*Başlangıç popülasyonu için en iyi çözümü hesapla g**
Olasılık anahtarı p ∈ [0, 1] oluştur.
while *(t<Maksimum İterasyon)*
  for *i=1:n*
    if rand < p (üniform dağılım)
      Levy dağılımı L (Parametre sayısı kadar)
      Biyotik üreme $x_i^{t+1} = x_i^t + \gamma L(\lambda)(g_* - x_i^t)$
    Else
      for *ε (Üniform dağılıma göre [0,1])*
        *Rastgele **j** ve **k** çözümlerini seç*
        Abiyotik üreme $x_i^{t+1} = x_i^t + \varepsilon\left(x_j^t - x_k^t\right)$
      end if
yeni çözümü al ve kontrol et
yeni çözüm daha iyiyse popülasyonu güncelle
  end for
  *En iyi çözümü seç g**
end while

Şekil 4. ÇTA sözde kodu

2.2.3 Karınca Aslanı Algoritması (KAA)

Karınca aslan algoritmasında (KAA), karınca aslanları larva evresinde kumlu topraklarda küçük bir çukur kazarak kendilerine bir tuzak oluştururlar. Karınca aslanı, tuzak

çukurunun dibine gömüldükten sonra, bir avcı olarak bekleyişe geçer. Karıncaların veya diğer küçük böceklerin tuzaklara düşmesi sonrasında, çıkışı engellemek için kum fırlatır ve böceği çukurun dibine çeker (Mirjalili 2015b). Bu avlanma mekanizması doğadan ilham alınmış ve bir optimizasyon algoritması geliştirilmiştir. Bu avlanma stratejisi Şekil 5'te gösterilmektedir.



Şekil 5. Karınca aslanı avlanma stratejisi (Mirjalili 2015b)

Karınca aslanının avlanma mekanizmasına ait matematiksel model Denklem 24'te gösterilmiştir.

$$X(t) = \begin{bmatrix} 0 \\ cumsum(2r(t_1) - 1) \\ cumsum(2r(t_2) - 1) \\ \dots \\ cumsum(2r(t_n) - 1) \end{bmatrix} \quad (24)$$

Burada, *n*, maksimum iterasyon sayısı, *t*, rastgele yürüyüş adımı *cumsum*, kümülatif toplam ve *r(t),* bir stokastik fonksiyonu ifade eder. Denklem 25'te matematiksel ifadesi sunulmuştur.

$$r(t) = \begin{cases} 1, & if\ rand > 0.5 \\ 0, & if\ rand \leq 0.5 \end{cases} \quad (25)$$

Rastgele yürüyüşe başlayan karıncaları arama uzayında tutmak için kullanılan matematiksel eşitlik Denklem 26'daki gibi ifade edilmektedir.

$$X_i^t = (X_i^t - a_i)(d_i^t - c_i^t)(b_i - a_i)^{-1} + c_i^t \quad (26)$$

Burada, *i,* değişken sayısını *t,* iterasyon sayısını *a,* minimum rastgele yürüyüşünü, *b,* maksimum rastgele yürüyüşünü *c*, *d,* her bir iterasyonda güncellenen karınca aslanı pozisyonlarının sırasıyla minimum ve maksimum değerlerini göstermektedir. Karınca tuzağa girdiğinde karınca aslanı onları tuzağın dibine çekmek için kum fırlatmaya başlar (Yüzgeç ve Kılıç, 2018). Bu davranışa ait matematiksel ifade Denklem 27-30 arasında gösterilmektedir.

$$c_i^t = Antlion_i^t + c^t \quad (27)$$

$$d_i^t = Antlion_i^t + d^t \quad (28)$$

$$c^t = c^t . I^{-1} \quad (29)$$

$$d^t = d^t . I^{-1} \quad (30)$$

Şekil 6'da verilen KAA'a ait sözde kodu, karınca aslanlarını belirli bir başlangıç pozisyonuna yerleştirir ve maliyet değerlerini hesaplar. Elit bir karınca aslanına ait pozisyon saklanır. Belirlenen maksimum iterasyona kadar, her bir karınca aslanı için seçim yapılır. Seçilen karınca aslanı etrafında rastgele yürüyüş modeli oluşturulur, bu model kullanılarak karıncanın yeni pozisyonu belirlenir. Eğer karınca arama uzayı dışında ise, pozisyonu rastgele bir konuma atanır. Ardından, karınca aslanlarının maliyet değerleri hesaplanır ve her bir karınca aslanı için, eğer karıncanın maliyeti fonksiyon değeri daha iyi ise, karınca aslanı yerini ve pozisyonunu günceller. Elit karınca aslanı da güncellenir ve iterasyon nesiller boyunca devam eder. Bu şekilde, KAA algoritması, karınca aslanlarının etkileşimli bir şekilde pozisyonlarını güncelleyerek optimizasyon problemlerini çözer.

```
KAA Sözde Kodu
Karınca aslanları başlangıç pozisyonu belirle
Karınca aslanları maliyet değerleri hesapla
Elit karınca aslanı ve pozisyonu sakla
while (iterasyon < maksimum iterasyon)
    for (her bir karınca aslanı için)
        Karınca aslanı seçimi
        Elit karınca aslanı etrafında rastgele yürüyüş modeli
        Seçilen karınca aslanı etrafında karıncanın rastgele yürüyüş
        modelinin çıkarılması
        Rastgele yürüyüş modelinin normalize edilmesi
        Karıncanın pozisyonun belirleme
        if Karınca arama uzayı dışında ise,
            Rastgele arama uzayı içerisine at
        end if
        end for
            Karınca maliyet değerleri hesapla
            for (her bir karınca aslanı)
        İf karıncanın maliyeti daha iyi ise, karınca aslanı karıncayı
        yer ve pozisyonunu karıncanınki ile güncelle.
        end if
    end for
    Elit karınca aslanı güncelle
end while
```

Şekil 6. KAA sözde kodu

2.2.4 Güve-Alev Algoritması (GAA)

GAA algoritması, güvelerin ışık kaynaklarının etrafında yaptığı hareketleri modellemektedir. Güveler, ışık kaynağına doğru uçarak uzaklıklarını en aza indirmeye çalışırlar. GAA algoritması bu fikri temel alır ve popülasyon içerisinde bulunan güvelerin pozisyonlarını ayarlayarak en iyi çözüme ulaşmayı hedefler (Mirjalili 2015a). Güve-alev algoritmasına ilişkin Denklemler 31-35 arasında verilmiştir.

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & \cdots & m_{1,d} \\ m_{2,1} & m_{2,2} & \cdots & \cdots & m_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{n,1} & m_{n,2} & \cdots & \cdots & m_{n,d} \end{bmatrix} \quad (31)$$

$$OM = \begin{bmatrix} OM_1 \\ OM_2 \\ \vdots \\ OM_n \end{bmatrix} \quad (32)$$

$$F = \begin{bmatrix} F_{1,1} & F_{1,2} & \cdots & \cdots & F_{1,d} \\ F_{2,1} & F_{2,2} & \cdots & \cdots & m_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_{n,1} & F_{n,2} & \cdots & \cdots & F_{n,d} \end{bmatrix} \quad (33)$$

$$OF = \begin{bmatrix} OF_1 \\ OF_2 \\ \vdots \\ OF_n \end{bmatrix} \quad (34)$$

$$MFO = (I, P, T) \quad (35)$$

Burada *I* güvelerin ilk rastgele konumlarını, P güvelerin arama uzayındaki hareketini ve *T* arama sürecini bitirmeyi ifade eder. Denklem 36-37 rastgele dağılımı uygulamak için kullanılan fonksiyonu temsil eder.

$$M(i,j) = (ub(i) - lb(j)) * rand() + lb(i) \quad (36)$$

$$S(M_i, F_j) = D_i \cdot e^{bt} \cdot cos(2\pi t) + F_j \quad (37)$$

Burada D_i, *i* 'inci güve ile *j* 'inci alev arasındaki boşluğu (yani, $D_i = |F_j - M_i|$), b, logaritmik spiralin şeklini tanımlayan bir sabiti ve *t*, [-1,1] arasında rastgele bir sayıyı belirtir.

Şekil 7'de verilen GAA sözde kodu, güve M_i'nin başlangıç konumunu rastgele belirler ve uygunluk fonksiyonunu hesaplar. Ardından, belirli bir maksimum iterasyona kadar M_i'nin konumunu günceller ve Denklem 36'yı kullanarak alev sayısını hesaplar. İlk iterasyonda, sıralama işlemleri F ve OF matrislerini hesaplamak için kullanılır. Her iterasyonda, rastgele seçilen r ve t değerleri ile Denklem 36 ve 37 kullanılarak D ve $M(i,j)$ değerleri güncellenir. En iyi çözüm, iterasyonlar tamamlandığında yazdırılır.

```
GAA Sözde Kodu
Güve Alevi için parametreleri ayarla
Güve Mi konumunu rastgele başlat
for i = 1'den n'ye kadar
      fi için uygunluk fonksiyonunu hesaplayın
end for
while iterasyon < Max_itere et.
      Mi'nin konumunu güncelle
      Denklemi (36) kullanarak alev sayısını hesaplayın.
      fi için uygunluk fonksiyonunu değerlendirin
      if yineleme ==1 ise
            F=sıralama(M) ve OF=sıralama (OM)
      else
            F=sıralama(Mt-1, Mt) ve OF=sıralama (Mt-1, Mt)
      end if
      for i = 1'den n'ye kadar
            for j = 1'den d'ye itere et.
                  r ve t değerlerini güncelle.
Denklemi (36) ile D'nin değerini karşılık gelen güveye göre hesapla.
Denklem (37)'i ile M(i,j) saygısını karşılık gelen güveye güncelle.
            end for
      end for
end while
En iyi çözümü yazdırın
```

Şekil 7. GAA sözde kodu

2.2.5 Halat Çekme Optimizasyonu (HÇA)

Halat çekme algoritması (HÇA), iki rakip takımın halatın zıt uçlarını çekmesini modellemektedir. Oyunun temel kuralı, takımların rakibi belirli bir mesafeye kadar kendi tarafına çekmesini içerir. İki takım arasında oynanmakla birlikte takımların eşit sayıda oyuncusu olur (Kaveh ve Bakhshpoori 2021). Bu yarışmanın gözlemlenmesi ile HÇA algoritması önerilmiş ve çeşitli problemlerin çözümünde başarı ile uygulanmıştır. Şekil 8'de çekişmede yarışan takımlardan biri gösterilmektedir.

Şekil 8. Halat çekmede yarışan bir takım (Anonim)

HÇA algoritmasında bileşke kuvvet Denklem 38'deki gibi hesaplanabilir.

$$F_r = F_p - W_i \mu_k \quad (38)$$

Sonuç olarak, Newton'un ikinci yasasına göre, *i* nesnesi, *j* nesnesine doğru ivmelenir. Bu durumda Matematiksel ifade Denklem 39'daki gibidir.

$$a = \frac{F_r}{\left(W_i / g\right)} \quad (39)$$

i nesnesi sıfır hızdan başladığı için, yeni konumu Denklem 40'da belirlenir:

$$X_i^{yeni} = \frac{1}{2} a t^2 + X_i^{eski} \quad (40)$$

$$x_{ij}^0 = x_{j,min} + rand\left(x_{j,max} - x_{j,min}\right) j = 1,2,\ldots,n \quad (41)$$

Denklem 41'de x_{ij}^0, *i*'inci aday çözümün, *j*'inci değişkeninin başlangıç değeridir. $x_{j,max}$ve $x_{j,min}$ sırasıyla *j*'inci değişken için izin verilen maksimum ve minimum değerlerdir. *rand* [0, 1] aralığındaki tekdüze bir dağılımdan rastgele bir sayıyı göstermektedir. Şekil 9'da HÇA'nın sözde kodu verilmiştir. Başlangıçta belirlenen parametrelerle bir popülasyon oluşturup, bu popülasyon içindeki aday çözümleri bir lige yerleştirerek başlar. Amaç

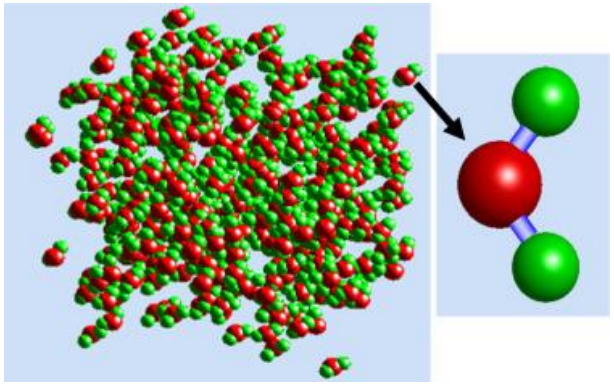
fonksiyonları değerlendirilir ve ligdeki takımlar arasında ağırlık tabanlı bir rekabet gerçekleşir. Lig içinde takımlar arasındaki yer değiştirme işlemleri sonucunda elde edilen toplam yer değiştirmeler ve son konumlar belirlenir. Yan kısıtlamalar kullanılarak değişkenler yeniden oluşturulur. Bu adımlar sonlandırma koşulu karşılanana kadar tekrarlanır, böylece algoritma popülasyonu optimize eder ve problem çözümünü geliştirir.

```
HÇA Sözde Kodu
Parametreleri ayarla
    N rastgele aday çözümden oluşan bir popülasyonu seç.
    Tüm rastgele aday çözümlerini kaydederek ligi başlat.
    while (sonlandırma koşulu karşılanmadı)
        Aday çözümler için amaç fonksiyonu değerlerini değerlendir.
        Yeni çözümleri sıralayın ve ligi güncelle.
        Wi ligindeki takımların ağırlıkları (Xi)'e tanımla.
            for i takımı için
            for j takımı için
                if (Wi<Wj)
                i takımını j takımına doğru hareket ettir.
                end if
            end for
            Takımının toplam yer değiştirmesini belirle.
            Takımının son konumunu belirle.
Değişkenleri yeniden oluşturmak için yan kısıtlama işlemini uygula
        end for
    end while
end
```

Şekil 9. HÇA sözde kodu

2.2.6 Atom Arama Optimizasyonu (AAA)

AAA, moleküler düzeyde atomların dinamik davranışından ilham alır ve arama uzayındaki her bir çözümü, o çözümü temsil eden atomun konumu ve kütlesi ile belirler. Daha iyi bir çözüm, daha ağır bir atomla temsil edilirken, daha kötü bir çözüm daha hafif bir atomla temsil edilir. Bu şekilde, AAA doğal seçilim benzeri bir evrim süreci uygular ve daha iyi sonuçlar için ağır atomları tercih eder. Atomlar kovalent bağlarla birleşerek molekülleri oluşturur. Atomların kütlesi ve hacmi vardır ve birbirleriyle etkileşim içindedirler. Bu etkileşim kuvvetleri, atomlar arasındaki mesafelere bağlı olarak çekme veya itme şeklinde gerçekleşir (Zhao vd. 2019). Şekil 10'da atom arama süreci gösterilmektedir.



Şekil 10. Atom arama süreci (Zhao vd. 2019)

Atom arama algoritması temelde Newton'un ikinci hareket yasasını kullanır. Denklem 42'de bu algoritmanın temel aldığı yasanın genel bir formu bulunmaktadır. Bu denklemde F_i, atom *i*'nin, G_i üzerindeki etkileşiminin bileşke kuvvetidir. Atom *i*, m_i üzerindeki kısıtlayıcı kuvvetin bileşke kuvvetidir, a_i, *i* atomunun ivmesidir.

$$F_i + G_i = m_i a_i, \tag{42}$$

Şekil 11'de sunulan AAA sözde kodu, bir dizi *X* atomunu ve onların hızlarını rastgele başlatır ve başlangıçta Fit_{Best} değerini sonsuz olarak atar. Durdurma kriteri karşılanana kadar, her bir *X* atomu için uygunluk değeri hesaplanır ve bu değer Fit_{Best}'ten küçükse, Fit_{Best} ve X_{Best} güncellenir. Ardından, kütleyi ilgili denklemler kullanarak hesaplar ve her atom için belirlenen komşularını seçer. Etkileşim ve kısıtlama kuvvetleri hesaplanarak ivme elde edilir. Hız ve konum güncellemeleri yapılır. En sonunda, şu ana kadar bulunan en iyi çözüm X_{Best} bulunur.

```
AAA Sözde Kodu
Bir dizi X atomunu (çözümler) ve bunların v hızlarını rastgele başlat.
Fit_Best=Inf atama yap
While Durdurma kriteri karşılanmadığında
  Her X_i atomu için şunu yap
            Uygunluk değerini hesaplay
            if Fit_i < Fit_Best ise
                Fit_Best = Fit_i;
                X_Best = X_i;
            end if
      Kütleyi ilgili denklemleri kullanarak hesapla.
      K komşusunu belirle.
      Sırasıyla etkileşim kuvveti F_i ve kısıtlama kuvveti G_i'yi hesapla.
      İvmeyi hesapla.
      Hızı güncelle.
      Konumu güncelleyin;
  end for
end while
Şu ana kadarki en iyi çözümü bul X_Best
```

Şekil 11. AAA sözde kodu

2.2.7 Beyin Fırtınası Optimizasyonu (BFA)

Beyin fırtınası optimizasyonu algoritması (BFA), insanların beyin fırtınası sürecindeki kolektif davranışını taklit eden bir algoritmadır. BFA, arama alanını azaltmak için kullanılan çeşitli çözümler önermektedir. Tüm çözümler sonunda birkaç küme oluşturacak şekilde gruplandırılır ve bu kümeler bir sorunun yerel optimumlarını temsil eder. BFA, en iyi uygunluk değerlerine sahip çözümleri içeren bir alanın bilgisini tutar ve bu bilgi kümeler arasında yayılır. Algoritma keşif yapmak için ilk önce karar uzayında dolaşacak ve iterasyonlar ilerledikçe keşif ve sömürü arasında bir denge durumu oluşturacaktır (Shi 2011). Denklem 43 BFA algoritmasına ilişkin genel ifadeyi göstermektedir.

$$X' = X_s + \xi(t).N(0,1) \quad (43)$$

Denklem 43'te X', yeni bir bireyi temsil ederken X_s seçim sonucu olur. *N (0, 1)* standart normal dağılımı temsil eder. $\xi(t)$ Denklem 44'te gösterildiği gibi adım boyutunu ayarlamak için bir işlevdir.

$$\xi(t) = logsig\left(\frac{0.5.T - t}{c}\right).U(0,1) \quad (44)$$

Denklem 44'te, aktarım işlevi olduğu yerde *T* ve *t değerleri,* sırasıyla maksimum yineleme sayısı ve geçerli yineleme sayısıdır. *c, logsig*'in eğimini kontrol eden bir katsayıdır. *U(0, 1)*, 0'dan 1'e düzgün dağılım temsil eder.

BFA Sözde Kodu
Giriş n, T, K, P_{rep}, $P_{clus/1}$, $P_{cen/1}$,$P_{cen/2}$ parametrelerini ayarla.
Bir popülasyon oluşturmak için rastgele n adet birey oluştur.
While *t < T*
  Popülasyonu K şeklinde kümelemek için K-araç sayısını belirle.
  Popülasyonu uygunluk değerini hesapla.
  Her kategoride en iyi olanı merkez olarak seç.
  if *U(0, 1) < P_{rep}*
    Rastgele bir kategorinin merkezini rastgele oluşturulmuş bir bireyle değiştir.
  for *i=1:n için yap*
    if *U(0, 1) < $P_{clus/1}$ ise*
      Rastgele bir küme seç.
      if *U(0, 1) < Pcen/1 ise*
        Merkezi X_s olarak seçin;
      else
        Bir bireyi X_s olarak rastgele seç.
    else
      İki kümeyi rastgele seç.
      if *U(0, 1) < P_c ise*
        iki merkezi birleştir
      else
        Her kategoriden rastgele bir birey seç.
        X_s olarak iki rastgele bireyin birleşimini yap.
  Yeni birey oluştur.
  Nüfusu güncelle;
  En iyi bireyi kaydet
  t = t + 1;

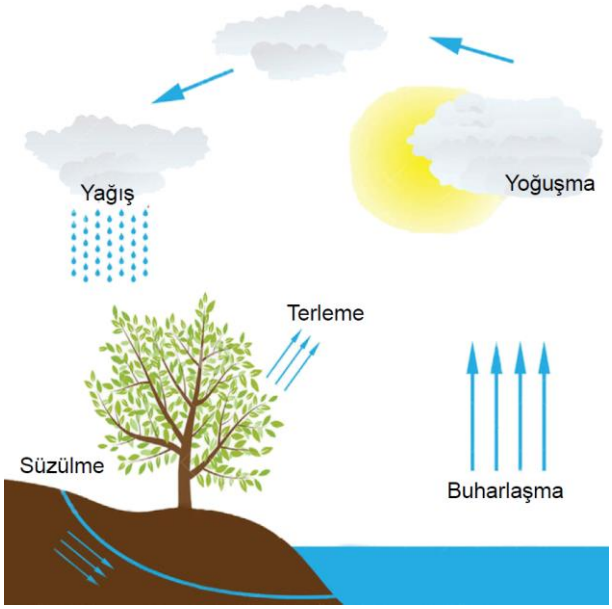
Şekil 12. BFA sözde kodu

Şekil 12'de sözde kodu verilen BFA şu adımları içerir: Belirli parametreleri ayarla ve başlangıçta n adet birey içeren bir popülasyon oluştur. Belirlenen iterasyon sayısına *(T)* ulaşana kadar popülasyonu belirli bir sayıda *(K)* kümeye ayırır, uygunluk değerlerini hesaplar ve her kategoride en iyi olanları merkez olarak seçer. Eğer belirli bir olasılık (P_{rep}) altında ise, rastgele bir kategorinin merkezini başka bir rastgele birey ile değiştir. Her birey için, belirlenen olasılıklara göre kümelenme ve merkez değiştirme işlemlerini uygular. Yeni bireyler oluşturarak popülasyonu günceller.

2.2.8 Su Döngüsü Algoritması (SDA)

Bu algoritma, gerçek dünyada nehirlerin oluşumundan esinlenerek tasarlanmıştır. Nehirler, yüksek rakımlardaki kar ve buzulların erimesiyle oluşur ve denize doğru akarlar. Suyun buharlaşması ve atmosfere karışması sonucunda oluşan bulutlar ise yağmur ya da diğer şekillerde yeryüzüne geri döner. SDA'da, rastgele oluşturulan bir popülasyon yağmur damlalarını temsil eder. Her bir yağmur damlası, bir dizi tasarım veya karar değişkenini içeren bir bireydir. En iyi birey, belirli bir problemin en uygun çözümünü temsil eden "deniz" olarak seçilir. (Eskandar vd. 2012). Şekil 13 su döngüsünü temsili olarak göstermektedir. Denklem 45'te yağmur damlacıklarının temsili olarak matris formu bulunmaktadır.

$$N_{var} = [x_1, x_2, x_3, \dots, x_N,] \quad (45)$$



Şekil 13. Su döngüsü (Eskandar vd. 2012)

Algoritmanın başlayabilmesi için $X = N_{pop}xN_{var}$ boyutunda bir yağmur damlası popülasyonu oluşturulur. Bu nedenle, rastgele üretilen X matrisi Denklem 46'da verilmiştir:

$$X = \begin{bmatrix} YD_1 \\ YD_2 \\ YD_3 \\ . \\ . \\ YD_{Npop} \end{bmatrix} = \begin{bmatrix} X_1^1 & X_2^1 & X_3^1 & . & X_{Nvar}^1 \\ X_1^2 & X_2^2 & X_3^2 & . & X_{Nvar}^2 \\ . & . & . & . & . \\ X_1^{Npop} & X_2^{Npop} & X_3^{Npop} & . & X_{Nvar}^{Npop} \end{bmatrix} \quad (46)$$

Karar değişkeni değerlerinin (X_1, X_2, …, X_{var}) her biri başlangıç noktası olarak gösterilebilir. Bir damlanın maliyeti en düşük değeri temsil eder. Buna göre, verilen denklemde N_{sr}, nehir sayısını ve denizin toplam değerini ifade eder. Geri kalan değerler ise yağmur damlalarının

nehirlerde veya doğrudan denizde biriktirebileceği değerlerden oluşur (Gür, 2020, s.17). Sonuç olarak su zerreciklerin birleşmesi suyu büyütür. Buharlaşma ve erozyon ise suyun küçülmesine sebep olur. Denklem 47 maliyet fonksiyonunu gösterir.

$$C_i = f\left(x_1^i, x_2^i, x_3^i, \dots, x_{Npop}^i\right) \quad (47)$$

Burada $i = 1,2,3,\dots,N_{pop}$ olarak ifade edilir. Bu durumda yeni eşitlikler Denklem 48 ve49'da verilmiştir.

$$N_{sr} = Nehir\ Sayısı + 1 \quad (48)$$

$$N_{yd} = N_{pop} - N_{sr} \quad (49)$$

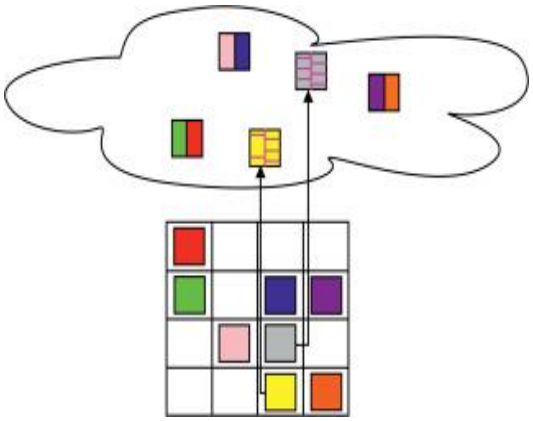
Akışkanlık hızına bağlı olarak damlaların nehirlere ve denize akışının modeli Denklem 50'de verilmiştir. NSn, belirli nehirlere veya denize akan akarsuların sayısını oluşturur.

$$NS_n = yuvarla\left\{\left|\frac{Maliyet_n}{\sum_{i=1}^{N_{ST}} Maliyet_i}\right| XN_{yd}\right\} \quad (50)$$

Şekil 14'te genel olarak SDA'nın sözde kodu verilmiştir. Süreç öncelikle başlangıç parametrelerinin seçilmesi ile başlar. Rastgele bir başlangıç popülasyonu oluşturulur ve yağmur damlaları, dereler, nehirler ve deniz gibi akışlar simüle edilir. Her yağmur damlasının değeri (maliyeti) hesaplanır ve akış yoğunluğu belirlenir. Nehirler ve denizler arasındaki akışlar hesaplanarak en yüksek yokuşa doğru olan deniz'e akarlar. Her akışın konumu, en iyi çözümü temsil eden bir akış ile değiştirilir. Eğer bir nehir, denizden daha iyi bir çözüm bulursa, nehrin konumu denizle değiştirilir. Buharlaşma durumu kontrol edilir ve bu durum sağlanıyorsa yağmurlama işlemi yapılır. Kullanıcı tanımlı bir parametre olan d_{max} değeri azaltılır ve yakınsama kriterleri kontrol edilir. Durdurma kriteri karşılanırsa, algoritma durur; aksi takdirde, başlangıçtaki popülasyon oluşturulma aşamasına geri dönülür. Bu adımlar, belirli bir yakınsama durumuna ulaşılana kadar tekrarlanır.

SDA Sözde Kodu
WCA'nın başlangıç parametreleri seçilir.
Rastgele başlangıç popülasyonu oluşturulur.
ilk akışlar (yağmur damlaları), dereler, nehirler ve deniz oluşturulur.
Her bir yağmur damlasının değeri(maliyeti)hesaplanır.
Nehirler ve deniz için akış yoğunluğunu belirlenir.
Akış değeri hesaplanır..
Nehirler, en yokuş aşağı yer olan Deniz (Mutlak Hedef)'e akar.
Nehir konumlarını en iyi çözümü veren bir akımla değiştirilir.
Eğer bir nehir denizden daha iyi bir çözüm bulursa, nehrin konumu denizle değiştirilir.
Buharlaşma durumu kontrol edilir
Buharlaşma koşulu sağlanmışsa, yağmurlama işlemi yapılır.
Kullanıcı tanımlı parametre olan d max değeri azaltılır.
Yakınsama kriterleri kontrol edilir.
Durdurma kriteri karşılanırsa, algoritma durur, aksi takdirde 5. adıma dönülür.

Şekil 14. SDA sözde kodu



Şekil 15. Mercan resifleri algoritmasının temel süreci (Yang vd. 2018)

2.2.9 Mercan Resifleri Algoritması (MRA)

Mercan resifleri algoritması (MRA), belirli bir doğal ekosistemin davranışının yapay simülasyonunu deneyen biyolojik esinli bir evrimsel algoritmadır. MRA algoritması, farklı çözümleri kodlayan bireyler, popülasyonu oluşturmak ile başlar ve bu çözümler kare bir ızgarada (resif) bulunur. Algoritma, mercan üreme sürecini (cinsel ve eşeysiz üreme operatörleri uygulanır) ve uzay için bir mücadelenin meydana geldiği mercan resifleri oluşum sürecini simüle eder (Salcedo vd. 2014). Şekil 15'te mercan resiflerinde mercanın çoğalma sürecini göstermektedir.

Mercan resifleri algoritması, fraktal geometriyi kullanarak doğal mercan resiflerinin karmaşık yapısını modeller. Algoritma, mercan resiflerinin büyümesini ve gelişmesini simüle etmek için Denklem 51'i kullanılır.

$$dN/dt = rN(1 - N/K) \quad (51)$$

Denklem, klasik lojistik denkleminden türetilmiştir ve mercan kolonilerinin zaman içinde nasıl büyüdüğünü göstermektedir.

$$dD/dt = -mD \quad (52)$$

Denklem 52, ölü mercan kolonilerinin zaman içinde nasıl arttığını gösterir. Bu denklem ayrıca ölü koloni sayısının *(D)* ölüm hızı *(m)* ile orantılı olduğunu gösterir.

$$dS/dt = cD \quad (53)$$

Burada *S,* mercan resiflerinde biriken tortu miktarını, *c* ise mercan kolonilerinin tortulaşma oranını ifade etmektedir. Şekil 16'da MRA'nın sözde kodu sunulmuştur.

```
MRA Sözde Kodu
Başlangıç parametrelerini N, M, Fa, Fb, Fd, Pdi γ, ke, n, k, Ngen ayarla
Tüm çözüm programları için çözüm alanı araştır
Mercan resiflerini konumlarını belirle
Mercan popülasyonlarını hesapla Ω;
for c = 1 için 2; c < n; c≠ Ngen
    Yayın yumurtlaması (dış cinsel üreme);
    Kuluçkaya yatma (içsel cinsel üreme);
    Larva ortamını belirle
    Eşeysiz üreme değerini hesapla.
    Larva ortamı;
    if (c≠ Ngen) ise
        Yağma mekanizması kullanarak hesaplama yap.
    end
end
```

Şekil 16. MRA sözde kodu

2.2.10 Yaşam Seçim Tabanlı Algoritma (YSTA)

YSTA algoritması, insanın yaşam döngüsünü ve çalışma etiğini inceleyerek geliştirilen bir algoritmadır. İnsanların doğadan ilham alarak yeni şeyler öğrendiği ve diğer türlerden öğrenme kabiliyetine sahip oldukları düşünülerek hayatta kalma odaklı çalışmalar yürüttükleri esasına dayanarak ortaya konmuştur (Khatri vd. 2020). İnsan her zaman bir şeylerden ilham alır, ya kendisi gibi birini, bir ünlüyü ya da arkadaşlarını takip eder. Kişi, hedefi gerçekleştirmek için en iyi kişilerden faydalı bir şeyler almaya çalışır ve üstün kişinin verimliliğini gözlemleyerek bir desen türetir. LCBO algoritması, insan düşünme süreçlerinden ve karar verme şekillerinden etkilenmiştir. Aynı zamanda Jaya optimizasyon tekniğinden de ilham almıştır. Jaya 'da sadece en iyi ve en kötü ajanlar mevcut ajanı etkilerken, LCBO'nun önerilen optimizasyonu belirli bir isteğe bağlı dalı içerir ve en iyi arama ajanları da mevcut ajanı etkiler. Bu da arama alanının daha iyi bir şekilde kullanılmasını sağlar. Denklem 54'te $U(x)$, seçim x'in toplam faydasını, w_i, *i.* seçeneğin önemini gösteren ağırlık, $u_i(x_i)$, *i.* seçeneğin faydası ve *n,* olası seçeneklerin sayısını göstermektedir.

$$U(x) = \sum_{\{i=1\}}^{n} w_i \, u_i(x_i) \tag{54}$$

Denklem 55 ise bir seçimin olası sonuçlarını hesaplar. Bu denklemde $P(y|x)$, seçim *x*'i yaptıktan sonra *y* sonucunu elde etme olasılığı $p_i(x_i)$ ise seçim x_i yaptıktan sonra *y* sonucunu elde etme olasılığını ifade etmektedir.

$$P(y|x) = \sum_{\{i=1\}}^{n} p_i(x_i) \tag{55}$$

YSTA, belirli bir insan popülasyonu ve r_1 değeri ile başlar. Uygunluk değerlerini hesaplar ve popülasyonu uygunluk değerlerine göre sıralar. Belirlenen bir değişim sayısına ulaşana kadar şu adımları tekrarlar: Her birey için rastgele bir değer oluşturur ve bu değere göre mevcut arama aracısını günceller. Eğer oluşturulan rastgele değer belirli bir aralığın dışında ise, Denklem 54 ve 55'i kullanarak güncelleme yapar. Yeni arama aracısının uygunluk değerini hesaplar ve eğer bu değer önceki uygunluk değerinden daha iyi ise arama aracısının konumunu ve uygunluk değerini günceller. Güncellenen popülasyonu sıralar ve güncel değişimi artırarak iterasyonu devam ettirir. Sonunda, başlangıçtaki popülasyonun en iyisini seçerek algoritmayı tamamlar. Şekil 17 YSTA'ya ait sözde kodu göstermektedir.

```
YSTA Sözde Kodu
İnsan popülasyonunu ve r1'i değerini belirle
Uygunluk değerlerini hesaplayıp ve popülasyonu sırala
While (güncel değişim <değişim sayısı)
for i=1:n için yap
    z = Rand()
    if (z> 0,875)
     Mevcut arama aracısını güncelle.
     else (z <0.70)
     Denklem 54 ve 55 kullanarak güncelleme yap.
     else
     Mevcut arama aracısını güncelleyin.
     end if
     Arama aracısının uygunluk değerini hesapla
     if (yeni uygunluk değeri önceki uygunluk değerinden daha iyiyse)
     Arama aracısı konumunu ve uygunluk değerini güncelle
     end if
end if
    X = sıralama(X)
Güncel değişim = Güncel değişim + 1
end while
Xi e geri dön
```

Şekil 17. YSTA sözde kodu

2.3.Sistem Tanımlama

Kontrol mühendisliğinin alt bir alanı olarak ifade edilen sistem tanımlama, temel olarak bir sistemin giriş/çıkış verilerine bağlı olarak matematiksel model elde etme sürecidir (Çelikel ve Gündoğdu 2020, Fidan vd. 2022, Zaloğlu vd. 2023). Sistemlerin matematiksel modellerini elde etmek için kullanılan çeşitli yöntemler bulunmakla birlikte en yaygın kullanılanlardan biri siyah kutu modellemedir. Siyah kutu sistem tanımlama modelinde, sisteme ait tanımlanmış bir matematiksel model yoktur. Bu yöntemde sistemden toplanan giriş/çıkış verileri kullanılarak model elde edilir. Kara kutu sistem tanımlama Denklem 56'da sunulmuştur.

$$\begin{aligned} y(t) = f[&y_1(t-1), \dots, y_1(t-n_{y_1}), \dots, y_r(t-1), \dots, y_r(t-n_{y_r}), \\ &u_1(t-1), \dots, u_1(t-n_{u_1}), \dots, u_m(t-1), \dots, u_m(t-n_{u_m}), \\ &e_1(t-1), \dots, e_1(t-n_{e_1}), \dots, e_r(t-1), \dots, e_r(t-n_{e_r})] \end{aligned} \tag{56}$$

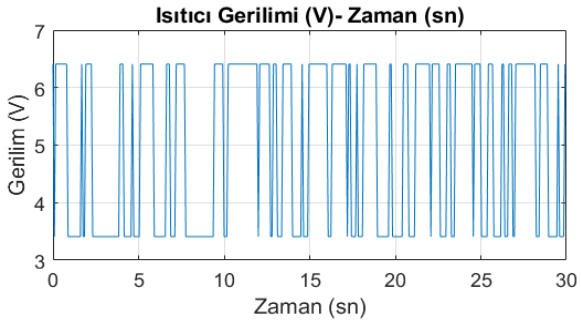
Denklem 56'da $\boldsymbol{y(t)}$ çıkış sinyalini, *u(t)* giriş sinyalini, *e(t)* ise hata sinyalini ifade etmektedir.

2.4. Ön Analiz

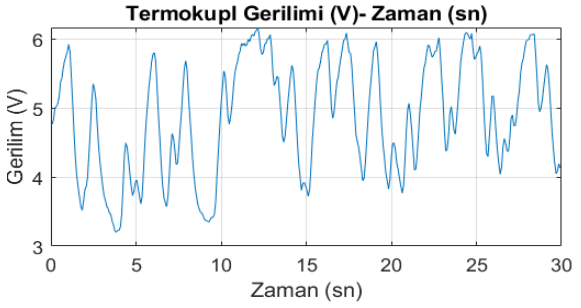
Sistem tanımlama algoritmalarının çalıştırıldığı PC 2.5 GHz hıza sahip, 6 çekirdeği olan (Intel Core i5 CPU) işlemci, 6 Gb RAM ve 1 TB özelliğindedir.

Şekil 18. Hava ısıtma deney seti görünümü

Algoritmaların geliştirilmesi için Python 3.8 altında Pandas 1.4.2, Mealpy 2.5.0 ve Control 0.9.2 kütüphaneleri kullanılmıştır. Veri seti için laboratuvar ölçeğinde hava ısıtma deney setinden toplanan veriler kullanılmıştır (Kumbasar vd. 2011). Bu setin görünümü Şekil 18'de verilmiştir. Hava akışı ve referans sıcaklık değerlerinin ölçümüyle elde edilen 1000 adet veri, 0.08 saniyelik bir zaman aralığında toplanmıştır. Deney düzeneğinde ısıtıcıya uygulanan gerilim ve hava sıcaklık ölçüm sensörü (termokupl) gerilimi sırasıyla Şekil 19-a ve Şekil 19-b'de gösterilmiştir.



a) Isıtıcı Gerilimi



b) Termokupl Gerilimi

Şekil 19. Giriş ve çıkış gerilim sinyalleri

3.BULGULAR

Bu çalışmada önerilen algoritmalar, rastgele başlangıç değerlerine bağlı olarak çalıştığı için çözümler her seferinde aynı başarıyla hesaplanamayabilir. Bu sebeple algoritmalar bağımsız olarak 100'er defa çalıştırılmış ve performanslar gösterilmiştir. Ayrıca alt bölümlerde çeşitli sınırlılıklar ele alınarak performans üzerine etkisi de sunulmuştur. Çizelge 1'de algoritma isimleri, döngü sayıları, popülasyon sayısı ve algoritmalara ait diğer parametreler sunulmuştur. Bu algoritmalar çalıştırılırken farklı sınırlılık değerleri için iterasyon sayısı değişmektedir. Bu sebeple Çizelge 1'de maksimum iterasyon sayısı sunulmuştur. Çizelge 1'de diğer parametrelerin seçiminde ise mealpy kütüphanesi içinde bulunan MS algoritmalarının her birine ait ön tanımlı değerler olduğu gibi kullanılmıştır. Topsis yönteminde, normalize edilmiş karar matrisinde her bir kriterin ağırlığını belirlemek için normalizasyon işlemi uygulanır. Karar matrisindeki değerler, bir ölçeklendirme yöntemi (örneğin, 0 ile 1 arasına normalleştirme) kullanılarak normalize edilir. Bu adım, farklı ölçeklerdeki kriterlerin karşılaştırılabilir hale getirilmesini sağlar. Bu makale çalışmasında kriterlerin seçimi ve normalizasyonunda performans (R^2), çözüm zamanı ve yükselme zamanı parametreleri dikkate alınmıştır. Bu parametrelerin değerleri sırasıyla 0.5, 0.25 ve 0.25 olarak belirlenmiştir. Bu değerlerin seçiminde sonuca doğrudan etki ettiği için araştırmacılar tarafından R^2 en önemli değer olarak belirlenmiştir. Çözüm zamanı ve yükselme zamanın etkisinin benzer olacağı düşünülmüştür.

Çizelge 1. Algoritmalara ilişkin parametreler

Algoritma İsimleri	Döngü Sayısı	Popülasyon Sayısı	Diğer Parametreler
AEO	10000	100	-
ALO	10000	100	-
ASO	10000	100	$\alpha = 10, \beta = 0.4$
BSO	10000	100	$m_c = 5, p_1 = 0.5,\ p_2 = 0.5,$ $p_3 = 0.5, p_4 = 0.5$
CRO	10000	100	$p_o = 0.4, F_b = 0.4,$ $F_a = 0.1,$ $F_d = 0.1\ GCR = 0.1,$ $\gamma_{max} = 0.2, n_{trials} = 3$
LCO	10000	100	$r_1 = 2.35$
FPA	10000	100	$p_s = 0.8, levy_m = 0.1$
MFO	10000	100	-
TWO	10000	100	-
WCA	10000	100	$N_{sr} = 4, w_c = 2.0,$ $d_{max} = 1e-6$

3.1 Güvenilirlik Analizi ve Sınırlılıklar

Performansa odaklanan birçok MS algoritması için durdurma kriterleri önemlidir. Doğru seçilmiş durdurma kriterleri sistem modelinin elde edilmesinde mümkün olan en kısa sürede sonuca gitmeyi sağlayabilir. Özellikle sınırlı kaynaklara sahip gömülü donanımlar için kritik zaman sınır değerinin ne olduğu iyi belirlenmelidir (Ravber vd. 2022). Ancak zaman değerini kısıtlı tutmak en uygun çözümü bulmanın önünde bir engel olabilir. Bu sebeple alt bölümlerde zaman sınırlılığı, maksimum generasyon, maksimum fonksiyon hesaplama ve erken durdurma sınırlılıkları ele alınmıştır. Sunulan algoritmalar belirtilen sınırlılıklar altında incelenmiş ve veri seti için en uygun olan algoritmalar belirlenmiştir. Rastgele başlangıç değerlerine bağlı olarak çözüm üreten MS algoritmalar bir çözümde oldukça iyi sonuç üretebilirken diğer bir çözümde başarısız olabilmektedir. Bu belirsizlik önerilen algoritmaların güvenilirliğinin testini gerektirir ve iki Denklem (57-58) ile kontrol edilecektir. Bu ifadeler:

$$\frac{\sum_{i=0}^{n} R^2}{n} > 0.8 \tag{57}$$

$$f(x) = \begin{cases} P(0.7\ x < 0.8) \leq 0.05, & x = 1 \\ P(0.7\ x < 0.8) > 0.05, & x = 0 \end{cases} \tag{58}$$

Önerilen denklem 57 ve 58'e göre güvenilirliğin tanımı öncelikle ortalama R^2 değerinin 0.8'in üzerinde olmasına bağlıdır. İkinci aşamada ise tüm çözümler içinde %5'ten daha fazla sayıda değer 0.8'in altına inemeyeceği varsayımına dayanmaktadır. Bu iki varsayım kod geliştirme sürecinde bir ihtiyaç olarak ortaya çıkmıştır. Bu tanımlama yapılmadığı takdirde parametre tahmininde önemli tahmin hatalarının olduğu belirlenmiştir.

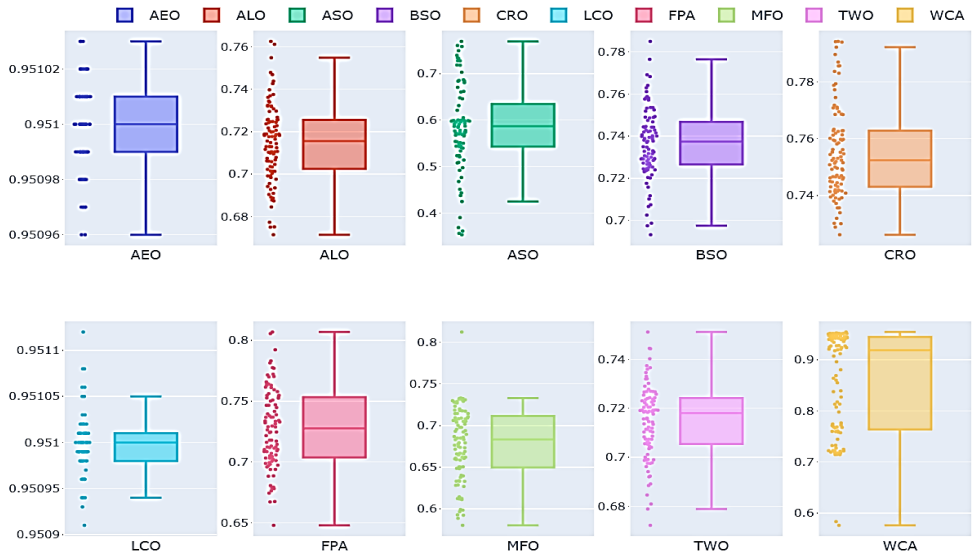
3.1.1. Zaman Sınırlılığı Senaryosu

Metasezgisel algoritmalar bazen kısıtlı bir performans artışı için önemli bir zaman kaynağı harcarlar. Bununla birlikte performanstan bir miktar ödün vererek önemli bir zaman kazanmak mümkündür. Bu durumda kısıtlanmış bir zamanda önerilen algoritmaların çözüm performanslarını incelemek gereklidir. Bu sebeple algoritmaların zaman sınırlılığı altında (45 sn) bağımsız olarak 100 defa çalışması sağlanmıştır. 45 sn değerinin seçiminde kullanılan PC'nin hızı ve araştırma ekibinin problem çözümünde edindiği deneyimler dikkate alınmıştır. Elde edilen R^2 değerleri Şekil 20'de box-plot grafiği olarak sunulmuştur. Bu sonuçlara göre, YEA ve YSTA algoritması 100 çözümde ortalama olarak en iyi performansa (0.951) ulaşmıştır. AAA algoritması ise en düşük performansa (0.59) erişmiştir.

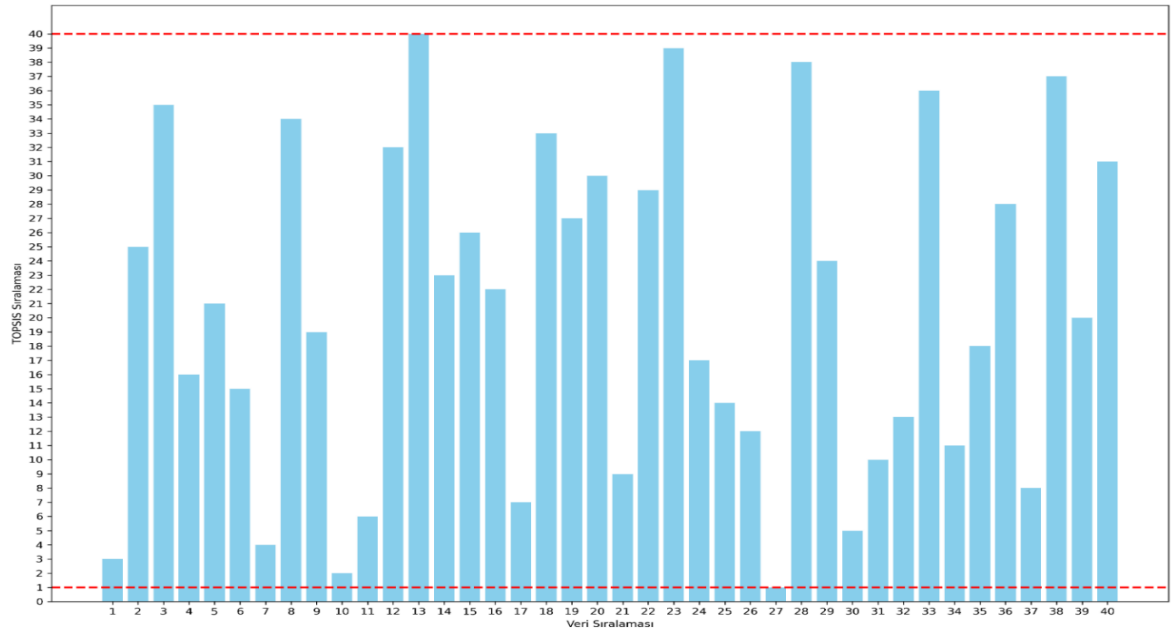
Şekil 20. Zaman sınırlığı altında (TB-45) R^2 performansları

3.1.2. Maksimum Generasyon Sınırlılığı

Zaman kritik olmayan sistemler için maksimum generasyon (MG=30) sınırlandırma kriterini kullanmak mümkündür. Zaman sınırlılığına benzer şekilde, MG değeri PC'nin hızı dikkate alınarak 30 seçilmiştir. Şekil 21'de sunulan box-plot grafiği ve algoritmaların R^2 performans değerleri incelendiğinde YEA ve YSTA algoritmaları 0.951 performans değerlerine ulaşmıştır. AAA algoritması en düşük R^2 ortalama değerlerine ulaşmıştır. YEA ve YSTA algoritmaları diğer algoritmalara kıyasla daha etkilidir. Ancak, diğer algoritmaların da farklı sınırlamalar altında iyi performans gösterme potansiyeli vardır. Performansı etkileyen diğer bir etkende başlangıç sınırların iyi seçilmesidir. Ancak optimal başlangıç değer seçimi başka bir çalışmanın konusudur.



Şekil 21. Maksimum generasyon (MG-30) sınırlılığında algoritmalarının R^2 performansı



Şekil 24. Topsis yöntemi ile yeniden sıralama işlemi

3.2. Çoklu karar verme (Topsis) sonuçları

Çizelge 2, 10 farklı MS algoritması ve 4 farklı sınırlılık için R^2, çözüm ve yükselme zamanlarını vermektedir. Bu değerlerden R^2'nin maksimum olması, yükselme ve çözüm zamanının ise minimum olması istenmektedir. Bu çizelge aslında Topsis yöntemi ile sıralama öncesi MS algoritmalarına ait verileri göstermek için ele alınmıştır. Bu sayede Topsis ile sıralamadan sonra hangi satırın sıralamasının nasıl değiştiği belirlenebilecektir. Şekil 24, Topsis yöntemi kullanılarak gerçekleştirilen çok kriterli karar verme analizinin sonuçlarını göstermektedir. Grafikte, her bir çubuk ideal çözüme göre göreceli yakınlığı temsil etmektedir. Kısa bir çubuk, ideal çözüme daha yakın olduğunu ve dolayısıyla tercih edilmesi gerektiğini gösterir ve karar vericiler için önerilen seçenektir. Grafikteki yatay kesikli çizgiler, en yüksek ve en düşük sıralamaları vurgulamak için eklenmiştir. Bu durumda Topsis sıralamasının gösterildiği *y* ekseninde 1. Sırada YSTA algoritması maksimum generasyon (MG) sınırlılığı altında R^2 (0.95096), çözüm zamanı (42.69 sn) ve yükselme zamanı (0.73) değerleri elde edilmiştir.

3.3. Performans İndekslerinin Karşılaştırılması

Sistem tanımlama amaçlı olarak kullanılan metasezgisel algoritmaların performans indeksleri ve elde edilen transfer fonksiyonları bu bölümde çizelgeler olarak sunulmuştur. Çizelgelerde performans indeksi olarak MAPE, MAE, MSE, R^2 ve zaman (süre) sunulmuştur. MAPE (Ortalama Mutlak Yüzde Hata), tahmin hatalarının yüzdesel ortalamasını ölçer. MAE (Ortalama Mutlak Hata),

tahmin ve gerçek değerler arasındaki mutlak farkların ortalamasını ifade eder. MSE (Ortalama Kare Hatası), farkların karelerinin ortalamasını ölçer. R^2 (Performans Katsayısı), bir regresyon modelinin uyumunu ifade eder; 1'e ne kadar yakınsa, modelin daha iyi uyum sağladığını gösterir. Bu bölümde algoritmalar bağımsız olarak bir kere çalıştırılmıştır. Yani optimize edilmiş parametreler bulunmuş ve algoritma çalışmayı durdurmuştur. Çizelge 3 zaman sınırlılığı altında performans indekslerini sunmaktadır. Bağımsız olarak bir defa çalışma sonucunda YEA (0.94768), YSTA (0.95092) ve SDA (0.95029) algoritmalarının performansları oldukça yeterlidir. Ancak SDA algoritmasının Şekil 24 dikkate alındığında bağımsız olarak bir defa çalışma durumunda performansı iyi olsa bile çoklu çalıştığında performansın düşük olma ihtimali bulunmaktadır.

Çizelge 2. 10 farklı MS algoritması ve 4 farklı sınırlılık için R^2, çözüm ve yükselme zamanı

Sıralama	Algoritma	Sınırlılık	R^2	Çözüm Zamanı	Yükselme Zamanı
1	YEA	TB	0.94768	45.81526	0.730082
2	KAA	TB	0.7045	45.13911	1.221899
3	AAA	TB	0.56406	46.2597	1.256448
4	BFA	TB	0.71945	45.6263	1.064155
5	MRA	TB	0.72777	45.32079	1.229626
6	ÇTA	TB	0.74248	46.23655	1.090785
7	YSTA	TB	0.95092	46.27365	0.731788
8	GAA	TB	0.7036	45.18044	1.658491
9	HÇA	TB	0.73306	46.4729	1.18741
10	SDA	TB	0.95029	45.82632	0.69
11	YEA	FE	0.95071	55.59337	0.711341
12	KAA	FE	0.70206	67.29051	1.408223
13	AAA	FE	0.39823	71.13545	2.191563
14	BFA	FE	0.74588	56.55411	1.347705
15	MRA	FE	0.70641	57.07968	1.2123
16	ÇTA	FE	0.7401	57.0222	1.278904
17	YSTA	FE	0.95098	55.68399	0.731269
18	GAA	FE	0.72973	56.83546	1.585394
19	HÇA	FE	0.7142	63.56779	1.240024
20	SDA	FE	0.95107	261.8676	0.730464
21	YEA	MG	0.95103	83.92427	0.649956
22	KAA	MG	0.69284	51.15	1.225706
23	AAA	MG	0.57223	54.77643	2.144938
24	BFA	MG	0.75325	42.59495	1.278513
25	MRA	MG	0.74851	24.1026	1.196992
26	ÇTA	MG	0.75624	42.6317	1.037952
27	YSTA	MG	0.95096	42.69103	0.730219
28	GAA	MG	0.59915	42.81121	2.141183
29	HÇA	MG	0.70734	49.31555	1.21763
30	SDA	MG	0.9236	43.02428	0.713384
31	YEA	ES	0.951	131.0782	0.730118
32	KAA	ES	0.74627	14.76414	1.184306
33	AAA	ES	0.49428	7.073467	1.386964
34	BFA	ES	0.81835	7.000684	1.188327
35	MRA	ES	0.67456	6.720242	0.969139
36	ÇTA	ES	0.81274	12.83776	1.809733
37	YSTA	ES	0.95092	60.12135	0.667294
38	GAA	ES	0.61392	9.734757	2.175592
39	HÇA	ES	0.70382	9.007025	1.162184
40	SDA	ES	0.951	265.0669	0.73044

MG sınırlılığında algoritma maksimum generasyon sayısına (30) ulaştığında durdurulur. Çizelge 4'te bu sınırlılık altında performans indeksleri sunulmuştur. YEA ve YSTA benzer performanslara sahip olmakla birlikte, YEA daha uzun çözüm süresine (83.92 sn) sahiptir. Performansı iyi olan algoritmalar içinde YSTA daha kısa sürede çözüm yaptığı için daha başarılıdır. Bu noktada YEA çözüm süresi uzun olsa bile zaman kısıtlılığı altında da benzer bir performans göstermiştir. Önerilen bir diğer sınırlılık olan fonksiyon hesaplama (FE) sınırlandırma kriteri, veri setinin doğasına bağlı olarak 4000 olarak belirlenmiştir. Çizelge 5'te, FE kriteri ile optimize edilmiş

transfer fonksiyonları ve performans göstergeleri sunulmaktadır. Algoritmalar, diğer kısıtlamalar altında olduğu gibi birbirinden farklı performans göstermiştir. YEA, YSTA ve SDA algoritmaları benzer ve iyi performanslar göstermiştir. Ancak SDA algoritmasının çözüm süresi (261.87 sn), en yavaş algoritma olduğunu göstermektedir. Bu iki yöntemin optimize ettiği transfer fonksiyonlarının katsayıları birbirlerine oldukça benzemektedir. SDA algoritmasının hesapladığı transfer fonksiyonu katsayıları YEA ve YSTA algoritmalarının hesapladığına benzemektedir. Ancak SDA algoritmasının başlangıç değerlerine bağlı olarak daha hassas olduğu ortadır. Elde edilen transfer fonksiyonlarının payda kısmı $s^2 + 6.1s + 12.9$ değerlerine yakındır.

Çizelge 3. Zaman sınırlılıkları altında performans göstergeleri ve transfer fonksiyonu

Algoritma	MAPE	MAE	MSE	R^2	Zaman	Transfer Fonksiyonu
YEA	0.0283	0.1371	0.0294	0.9477	45.82	$T_f(s) = \frac{-1.50s + 12.65}{s^2 + 6.14s + 12.95}$
KAA	0.06798	0.33375	0.16516	0.7045	46.14	$T_f(s) = \frac{-10.81s + 97.34}{s^2 + 60.67s + 100}$
AAA	0.08282	0.39716	0.24365	0.56406	45.26	$T_f(s) = \frac{-5.03s + 74.96}{s^2 + 84.30s + 83.42}$
BFA	0.06721	0.32456	0.15681	0.71945	45.62	$T_f(s) = \frac{-7.21s + 57.30}{s^2 + 33.17s + 58.99}$
MRA	0.06597	0.31821	0.15216	0.72777	45.32	$T_f(s) = \frac{-9.97s + 73.91}{s^2 + 43.19s + 75.97}$
ÇTA	0.06345	0.30803	0.14393	0.74248	46.23	$T_f(s) = \frac{-4.90s + 35.35}{s^2 + 27.67s + 36.32}$
YSTA	0.02748	0.13216	0.02743	0.95092	46.27	$T_f(s) = \frac{-1.51s + 12.66}{s^2 + 6.16s + 12.97}$
GAA	0.06811	0.32467	0.16567	0.7036	45.18	$T_f(s) = \frac{-13.82s + 100}{s^2 + 62.52s + 100}$
HÇA	0.06519	0.3122	0.1492	0.73306	46.47	$T_f(s) = \frac{-11.08s + 97.59}{s^2 + 59.79s + 99.22}$
SDA	0.02775	0.13349	0.02779	0.95029	45.89	$T_f(s) = \frac{-3.81s + 22.63}{s^2 + 9.61s + 23.12}$

Çizelge 4. Maksimum generasyon sınırlılıkları altında performans göstergeleri ve transfer fonksiyonu

Algoritma	MAPE	MAE	MSE	R^2	Zaman	Transfer Fonksiyonu
YEA	0.0274	0.1319	0.0274	0.951	83.92	$T_f(s) = \frac{-1.49s + 12.59}{s^2 + 6.11s + 12.90}$
KAA	0.06947	0.33863	0.17168	0.69284	51.15	$T_f(s) = \frac{-11.27s + 97.26}{s^2 + 60.41s + 100}$
AAA	0.08194	0.39324	0.23909	0.57223	54.78	$T_f(s) = \frac{-10.03s + 73.01}{s^2 + 43.02s + 72.29}$
BFA	0.06299	0.30406	0.13791	0.75325	42.59	$T_f(s) = \frac{-8.23s + 65.16}{s^2 + 38.07s + 66.84}$
MRA	0.06374	0.30694	0.14056	0.74851	24.10	$T_f(s) = \frac{-8.12s + 48.88}{s^2 + 26.57s + 50.73}$
ÇTA	0.06127	0.29784	0.13624	0.75624	42.63	$T_f(s) = \frac{-6.22s + 44.19}{s^2 + 28.53s + 46.17}$
YSTA	0.02747	0.13212	0.02741	0.95096	42.69	$T_f(s) = \frac{-1.50s + 12.60}{s^2 + 6.12s + 12.90}$
GAA	0.07765	0.38444	0.22404	0.59915	42.81	$T_f(s) = \frac{-10.15s + 78.31}{s^2 + 38.59s + 86.02}$
HÇA	0.06818	0.33059	0.16357	0.70734	49.32	$T_f(s) = \frac{-8.57s + 85.21}{s^2 + 49.71s + 83.37}$
SDA	0.03598	0.16954	0.0427	0.9236	43.02	$T_f(s) = \frac{-7.30s + 47.53}{s^2 + 27.49s + 48.61}$

Erken durdurma sınırlık kriteri global çözümün azalmaması durumunda algoritmanın durdurulmasını sağlar. Bu yöntem aslında çözüm için algoritmalara yeterince zaman sunmaktadır. Çizelge 5'te 10 farklı algoritma için tahmin edilen transfer fonksiyonlarının *MAE, MAPE, MSE* ve R^2 değerleri verilmiştir. Durdurma kriter değeri 3 olarak seçildiği zaman YEA algoritmasının performansı yeterli olmakla birlikte çözüm zamanı (131.08 sn) artmıştır. Performanslar dikkate alındığında YEA, YSTA ve SDA yine en yüksek değerlere ulaşmıştır. Ancak zaman sınırlılığı kriteri de dikkate alındığında YSTA algoritması performansıyla öne çıkmaktadır. AAA ve BFA ise nispeten başarısız performanslar göstermişlerdir. Şekil 14 ve Çizelge 5, 10 farklı algoritma ve 4 farklı sınırlılık için algoritmaların performansını sunmaktadır. Çizelge 5'te YEA ve YSTA algoritmasının başarılı olduğu açıktır ancak Topsis sonuçlarına göre YSTA algoritması 1. Sırada gözükmektedir. Topsis burada 3 farklı kriter dikkate alınarak kazananı belirlediği için öne çıkmaktadır. Çizelge 5'te ise sadece R^2 değerlendirmesi yapılmaktadır.

Çizelge 5. Fonksiyon hesaplama sınırlılıkları altında performans göstergeleri ve transfer fonksiyonu

Algoritma	MAPE	MAE	MSE	R^2	Zaman	Transfer Fonksiyonu
YEA	0.02755	0.13246	0.02755	0.95071	55.59	$T_f(s) = \frac{-1.49s + 12.61}{s^2 + 6.12s + 12.91}$
KAA	0.06928	0.33421	0.16652	0.70206	67.29	$T_f(s) = \frac{-13.31s + 97.59}{s^2 + 60.55s + 100}$
AAA	0.1001	0.47568	0.33634	0.39823	71.14	$T_f(s) = \frac{-19.08s + 100}{s^2 + 100s + 100}$
BFA	0.06411	0.30883	0.14203	0.74588	56.55	$T_f(s) = \frac{-8.48s + 67.20}{s^2 + 39.33s + 68.92}$
MRA	0.06847	0.33061	0.16409	0.70641	57.08	$T_f(s) = \frac{-9.33s + 80.51}{s^2 + 46.26s + 82.73}$
ÇTA	0.06534	0.31326	0.14526	0.7401	57.02	$T_f(s) = \frac{-9.43s + 61.27}{s^2 + 37.17s + 61.74}$
YSTA	0.02746	0.13206	0.0274	0.95098	55.68	$T_f(s) = \frac{-1.49s + 12.60}{s^2 + 6.11s + 12.91}$
GAA	0.06526	0.31034	0.15106	0.72973	56.84	$T_f(s) = \frac{-10.21s + 100}{s^2 + 80.48s + 100}$
HÇA	0.06798	0.32737	0.15974	0.7142	63.57	$T_f(s) = \frac{-13.99s + 95.31}{s^2 + 59.27s + 97.75}$
SDA	0.02743	0.13192	0.02735	0.95107	261.87	$T_f(s) = \frac{-1.50s + 12.62}{s^2 + 6.13s + 12.93}$

Çizelge 6. Erken durdurma sınırlılıkları altında performans göstergeleri ve transfer fonksiyonu

Algoritma	MAPE	MAE	MSE	R^2	Zaman	Transfer Fonksiyonu
YEA	0.02745	0.13203	0.02739	0.951	131.08	$T_f(s) = \frac{-1.49s + 12.60}{s^2 + 6.12s + 12.90}$
KAA	0.06397	0.30631	0.14181	0.74627	14.76	$T_f(s) = \frac{-4.72s + 51.72}{s^2 + 26.47s + 52.84}$
AAA	0.08961	0.42803	0.28265	0.49428	7.07	$T_f(s) = \frac{-17.04s + 42.09}{s^2 + 29.95s + 42.79}$
BFA	0.05357	0.25709	0.10153	0.81835	7.00	$T_f(s) = \frac{-8.52s + 88.20}{s^2 + 47.01s + 90.66}$
MRA	0.07147	0.34896	0.1819	0.67456	6.72	$T_f(s) = \frac{-7.14s + 50.75}{s^2 + 37.18s + 54.08}$
ÇTA	0.05411	0.26344	0.10466	0.81274	12.84	$T_f(s) = \frac{-18.78s + 79.65}{s^2 + 47.91s + 80.45}$
YSTA	0.02748	0.13215	0.02743	0.95092	60.12	$T_f(s) = \frac{-1.49s + 12.60}{s^2 + 6.12s + 12.91}$
GAA	0.07683	0.3693	0.21579	0.61392	9.74	$T_f(s) = \frac{-0{,}71s + 98{,}04}{s^2 + 100s + 93.53}$
HÇA	0.06961	0.33004	0.16554	0.70382	9.01	$T_f(s) = \frac{-8.42s + 37.39}{s^2 + 15.60s + 38.20}$
SDA	0.02745	0.13202	0.02738	0.951	265.07	$T_f(s) = \frac{-1.50s + 12.70}{s^2 + 6.18s + 13.01}$

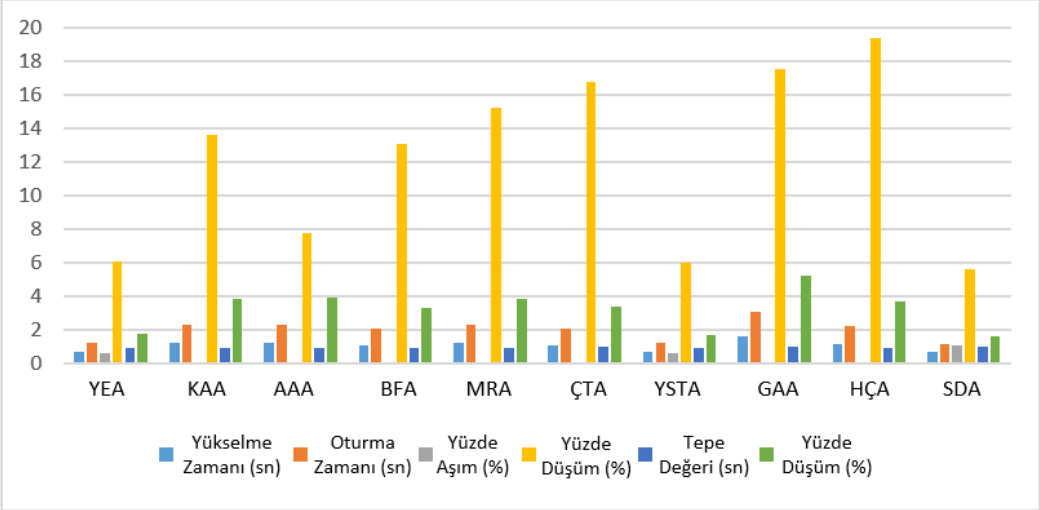
Sunulan çizelgelere ek olarak aykırı değerlerden de bahsetmek gerekir. Çizelge 7'de aykırı değer sayıları gösterilmektedir. İlgili bölümde her algoritmanın belirtilen sınırlılıklar altında başarısını görmek için bağımsız olarak 100 defa çalıştırıldığından bahsedilmişti. Bu çizelge incelenecek olursa YEA algoritması tüm sınırlılıklar altında en az 0.8 ve üstü R^2 performans değerine ulaşmıştır. YSTA algoritması 2 aykırı değere sahiptir. SDA algoritması ise toplamda 30 adet aykırı performans değerine ulaşmıştır.

Çizelge 7. Aykırı Değer Tablosu

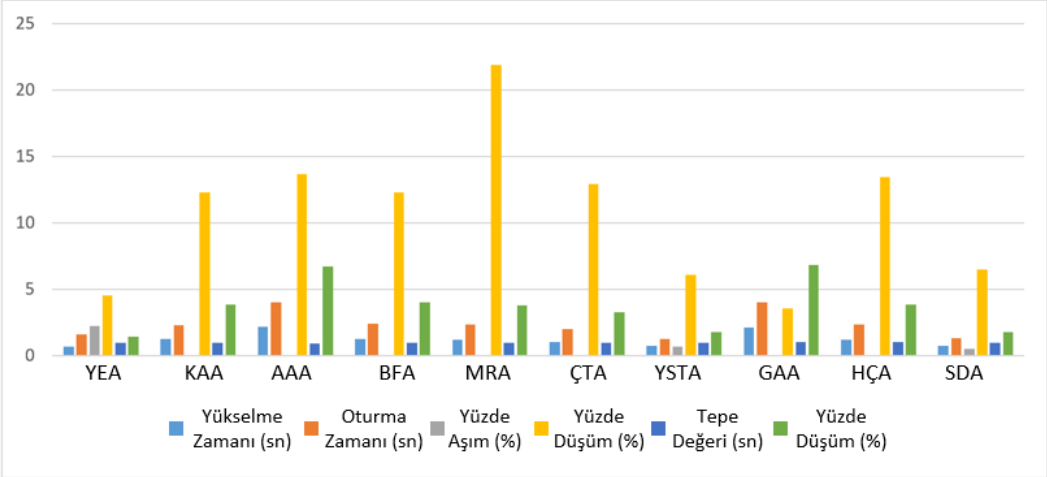
Algoritmalar	TB	MG	FE	ES
YEA	0	0	0	0
KAA	98	98	98	100
AAA	100	98	98	100
BFA	97	98	98	98
MRA	97	95	95	98
YSTA	1	0	0	1
ÇTA	89	90	89	99
GAA	100	100	100	100
HÇA	99	100	100	100
SDA	26	1	1	2

3.4. Algoritmaların Geçici Durum Cevapları

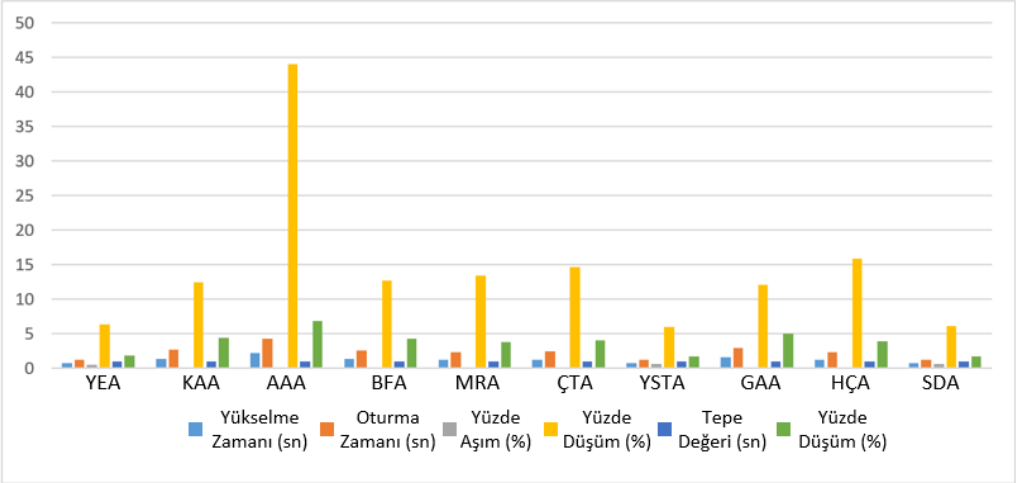
Şekil 25, 26, 27 ve 28'de farklı algoritmalar kullanılarak elde edilen transfer fonksiyonlarının geçici ve kalıcı zaman cevapları karşılaştırılmaktadır. Bu amaçla yükselme zamanı, oturma zamanı, aşım miktarı, düşüm miktarı, tepe değeri ve tepe zamanı gibi geçici durum değerleri ele alınmıştır. Yükselme zamanı, sistem tepkisinin sıfırdan maksimum değere ulaşma süresini ifade etmektedir. Oturma zamanı ise sistem tepkisinin ilk kez sabit değere ulaşma süresidir. Aşım miktarı, sistemin maksimum değeri geçme miktarını belirtir. Düşüm miktarı, sabit değere ulaştıktan sonraki hızlı değişimi ölçer. Tepe değeri, sistem cevabının maksimum değerini ifade eder. Tepe zamanı, maksimum değere ulaşma zamanını belirtir. Bu değerler, kontrol sistemi tasarımında önemli bir rol oynamaktadır. Şekil 25-28'de sunulan verileri kullanarak, farklı algoritmaların cevaplarının karşılaştırılması mümkündür. Bu nedenle, bu tür performans ölçütleri, kontrol sistemleri tasarımı ve iyileştirmesi için önemli bir araçtır. Şekil 25'te dikkat edilecek olursa YEA, YSTA ve SDA minimum aşım değerine (yaklaşık %6) sahiptir. Zaman sınırlılığı altında bu algoritmaların zaman cevabının benzer olduğunu göstermektedir. Şüphesiz ki kontrol sistemlerinde aşım değerinin fazla olması tercih edilmeyen bir durumdur.



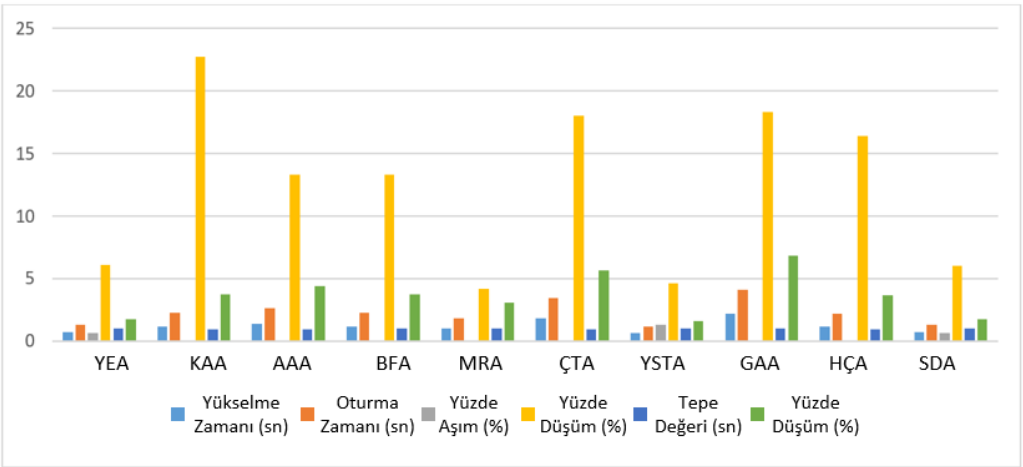
Şekil 25. MS algoritmalarının zaman sınırlılığında geçici durum cevapları



Şekil 26. MS algoritmalarının maksimum generasyon sınırlılığında geçici durum cevapları



Şekil 27. MS algoritmalarının fonksiyon hesaplama sınırlılığında geçici durum cevapları



Şekil 28. Algoritmaların erken durdurma sınırlılığında geçici duruma cevapları

Şekil 26'da maksimum generasyon sınırlılığı altında zaman cevapları sunulmuştur. Bu şekil incelendiğinde YEA, YSTA ve SDA öne çıkmaktadır. Bazı algoritmaların aşım değerleri 0 çıkmıştır. Bu değer zaman cevabında hiç aşım olmadığı manasına gelmektedir. Yükselme ve oturma zamanları açısından incelenecek olursa YEA, YSTA ve SDA kabul edilebilir bir aralıkta kalmıştır. Yani diğer algoritmalara kıyasla istenen minimum değerlere yakındırlar. Şekil 27 fonksiyon hesaplama sınırlılığı altında zaman cevaplarını göstermektedir. Burada dikkat edilirse özellikle AAA çok uzun aşım değerine sahiptir. Elde edilen transfer fonksiyonları açısından daha önce de belirtilen

YEA, YSTA ve SDA birbirlerine yakın geçici durum değerlerine sahiptir. Erken durdurma kriterleri dikkate alındığında elde edilen zaman cevapları Şekil 28'de sunulmuştur. Burada YEA, YSTA ve SDA'ya ek olarak MRA'nın da zaman cevapları diğer algoritmalara göre daha kısadır. Ancak geçici zaman kriterleri açısından Çizelge 7'de sunulan aykırı değerler dikkate alındığında halen en yüksek skorlu algoritma YEA algoritmasıdır. Topsis yönteminde ise YSTA algoritması öne çıkmaktadır. Bu noktada Topsis çok kriterli bir karar verme yöntemi olduğu için önerdiği algoritma daha öne çıkmaktadır.

4.Tartışma ve Sonuçlar

Bu çalışmada, 10 farklı metasezgisel algoritma tanıtılmış ve sistem tanımlama problemlerine uygulanmıştır. Makale çalışmasında zaman sınırlılığı kriteri dikkate alındığında yapay ekosistem ve yaşam seçim tabanlı algoritmalar öne çıkmıştır. Yapılan bir diğer analizde ise transfer fonksiyonlarının geçici durum cevapları incelenmiştir. Yapılan karşılaştırmalarda YEA ve YSTA algoritmalarının sistem tanımlama problemlerine kolaylıkla uygulanabildiği ve diğer MS algoritmalarla karşılaştırıldığında yüksek performans değerleriyle ön plana çıktığı görülmektedir. Bu makalede önerilen Topsis çok kriterli karar verme mekanizması ise R^2, çözüm ve yükselme zamanlarını birlikte değerlendirmektedir. Bu 3 kriter birlikte değerlendirildiğinde ise YSTA algoritması kazanan algoritma olarak öne çıkmıştır. Sonuç olarak bu çalışmayla MS algoritmalarının sistem tanımlama problemlerine uygulanabileceği görülmüştür. Ancak en iyi MS algoritmasının belirlenmesinde yaşanan sorunlar için Topsis yönteminin kullanımının uygun olduğu görülmüştür.

Etik Standartlar Bildirgesi

Makalenin yazarları tüm etik standartlara uyduklarını beyan ederler.

Yazarlık Katkı Beyanı

Yazar-1: Kaynaklar, Fikir Sahibi, Araştırma, Yazma
Yazar-2: Araştırma, Deneyleme, Doğrulama, Metodoloji
Yazar-3: Deney tasarımı, Görselleştirme

Çıkar Çatışması Beyanı

Yazarların bu makalenin içeriğiyle ilgili olarak beyan edecekleri hiçbir çıkar çatışması yoktur.

Verilerin Kullanılabilirliği

Bu çalışma sırasında oluşturulan veya analiz edilen tüm veriler, yayınlanan bu makaleye dâhil edilmiştir.

5. Kaynaklar

Canayaz, M., 2019, Training Anfis System with Moth-Flame Optimization Algorithm. *International Journal of Intelligent Systems and Applications in Engineering*, ***7*(3)**, Article 3. https://doi.org/10.18201/ijisae.2019355375

Çelikel, R., & Gundogdu, A., 2020, System identification-based MPPT algorithm for PV systems under variable atmosphere conditions using current sensorless approach. *International Transactions on Electrical Energy Systems*, ***30*(8)**, e12433.

Chen, Y., Pi, D., & Wang, B., 2019, Enhanced global flower pollination algorithm for parameter identification of chaotic and hyper-chaotic system. *Nonlinear Dynamics*, ***97*(2)**, 1343-1358. https://doi.org/10.1007/s11071-019-05052-z

Crispim, J. A., & Pinho de Sousa, J., 2009, Partner selection in virtual enterprises: A multi-criteria decision support approach. *International Journal of Production Research*, ***47*(17)**, 4791-4812. https://doi.org/10.1080/00207540902847348

Ding, S., Shi, Z., Chen, K., & Azar, A. T., 2015, Mathematical Modelling and Analysis of Soft Computing. *Mathematical Problems in Engineering*, *2015*, e578321. https://doi.org/10.1155/2015/578321

Dorigo, M., Birattari, M., & Stutzle, T., 2006, Ant colony optimization. *IEEE Computational Intelligence Magazine*, ***1*(4)**, 28-39. https://doi.org/10.1109/MCI.2006.329691

El-Dabah, M. A., & El-Sehiemy…, R. A., 2021, *Parameter estimation of triple diode photovoltaic model using an artificial ecosystem based optimizer*. *Int Trans Electr Energ Syst.* ***31(11)****:e13043. https://doi.org/10.1002/2050-7038.13043*

Eskandar, H., Sadollah, A., Bahreininejad, A., Hamdi, M., 2012, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems. *Computers & Structures*, ***110-111***, 151-166. https://doi.org/10.1016/j.compstruc.2012.07.010

Fadzli, A. A. M., Hadi, M. S., Eek, R. T. P., Talib, M. H. Ab., Yatim, H. M., & Darus, I. Z. M., 2022, PID Controller Based on Flower Pollination Algorithm of Flexible Beam System. *Recent Trends in Mechatronics Towards Industry 4.0*. Springer, 173-183 https://doi.org/10.1007/978-981-33-4597-3_17

Fan, S., Zhang, J., Blanco-Davis, E., Yang, Z., Yan, X., 2020, Maritime accident prevention strategy formulation from a human factor perspective using Bayesian NeHÇArks and TOPSIS. *Ocean Engineering*, ***210***, 107544. https://doi.org/10.1016/j.oceaneng.2020.107544

Farag, M. A., El-Shorbagy, M. A., Mousa, A. A., El-Desoky, I. M., 2020, A New Hybrid Metaheuristic Algorithm for Multi objective Optimization Problems. *International Journal of Computational Intelligence Systems*, ***13*(1)**, 920-940. https://doi.org/10.2991/ijcis.d.200618.001

Fidan, Ş., Sevim, D., Erkan, E. 2022, System Identification and Control of High Voltage Boost Converter. *2022 Global Energy Conference (GEC)*, 25-31. https://doi.org/10.1109/GEC55014.2022.9986621

Guo, Y., Shi, Q., Guo, C., 2022, A Performance-Oriented Optimization Framework Combining Meta-Heuristics and Entropy-Weighted TOPSIS for Multi-Objective Sustainable Supply Chain NeHÇArk Design. *Electronics*, ***11*****(19)**, Article 19. https://doi.org/10.3390/electronics11193134

Izci, D., 2022, A novel modified arithmetic optimization algorithm for power system stabilizer design. *Sigma Journal of Engineering and Natural Sciences*, ***40*****(3)**, 3

.Izci, D., Hekimoğlu, B., Ekinci, S., 2022, A new artificial ecosystem-based optimization integrated with Nelder-Mead method for PID controller design of buck converter. *Alexandria Engineering Journal*, ***61*****(3)**, 2030-2044. https://doi.org/10.1016/j.aej.2021.07.037

Janjanam, L., Saha, S. K., Kar, R., Mandal, D., 2022, Wiener model-based system identification using moth flame optimised Kalman filter algorithm. *Signal, Image and Video Processing*, ***16*****(5)**, 1425-1433. https://doi.org/10.1007/s11760-021-02096-w

Ji, Y., Jiang, X., Wan, L., 2020, Hierarchical least squares parameter estimation algorithm for HÇA-input Hammerstein finite impulse response systems. *Journal of the Franklin Institute*, ***357*****(8)**, 5019-5032. https://doi.org/10.1016/j.jfranklin.2020.03.027

Kalita, K., Pal, S., Haldar, S., Chakraborty, S., 2022, A Hybrid TOPSIS-PR-GWO Approach for Multi-objective Process Parameter Optimization. *Process Integration and Optimization for Sustainability*, ***6*****(4)**, 1011-1026. https://doi.org/10.1007/s41660-022-00256-0

Karaboga, D., & Basturk, B., 2007, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, ***39*****(3)**, 459-471. https://doi.org/10.1007/s10898-007-9149-x

Kaveh, A. and Bakhshpoori T., 2021, Tug of War Optimization, *Advances in Metaheuristic Algorithms for Optimal Design of Structures*. Springer International Publishing, 467-503. https://doi.org/10.1007/978-3-030-59392-6_15

Kennedy, J., & Eberhart, R., 1995, Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural NeHÇArks*, ***4***, 1942-1948 c.4. https://doi.org/10.1109/ICNN.1995.488968

Khatri, A., Gaba, A., Rana, K. P. S., Kumar, V., 2020. A novel life choice-based optimizer. *Soft Computing*, ***24*****(12)**, 9121-9141. https://doi.org/10.1007/s00500-019-04443-z

Khluabwannarat, P., Nawikavatan, A., Puangdownreong, D., 2018, *Fractional-Order Model Parameter Identification of BLDC Motor by Flower Pollination Algorithm*. *13*.

Kler, D., Sharma, P., Banerjee, A., Rana, K. P. S., Kumar, V., 2017,. PV cell and module efficient parameters estimation using Evaporation Rate based Water Cycle Algorithm. *Swarm and Evolutionary Computation*, ***35***, 93-110. https://doi.org/10.1016/j.swevo.2017.02.005

Kumbasar, T., Eksin, I., Guzelkaya, M., & Yesil, E., 2011, Adaptive fuzzy model based inverse controller design using BB-BC optimization algorithm. *Expert Systems with Applications*, ***38*****(10)**, 12356-12364. https://doi.org/10.1016/j.eswa.2011.04.015

Long, B., Yang, W., Hu, Q., Guerrero, J. M., Garcia, C., Rodriguez, J., & Chong, K. T., 2022, Moth–Flame-Optimization-Based Parameter Estimation for FCS-MPC-Controlled Grid-Connected Converter With LCL Filter. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, ***10*****(4)**, 4102-4114. https://doi.org/10.1109/JESTPE.2022.3140228

Mirjalili, S., 2015a, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, ***89***, 228-249. https://doi.org/10.1016/j.knosys.2015.07.006

Mirjalili, S., 2015b, The Ant Lion Optimizer. *Advances in Engineering Software*, ***83***, 80-98. https://doi.org/10.1016/j.advengsoft.2015.01.010

Mohammadi, A., Sheikholeslam, F., Mirjalili, S., 2022, Inclined planes system optimization: Theory, literature review, and state-of-the-art versions for IIR system identification. *Expert Systems with Applications*, ***200***, 117127. https://doi.org/10.1016/j.eswa.2022.117127

Mossa, M. A., Kamel, O. M., Sultan, H. M., Diab, A. A. Z., 2021, Parameter estimation of PEMFC model based on Harris Hawks' optimization and atom search optimization algorithms. *Neural Computing and Applications*, ***33*****(11)**, 5555-5570. https://doi.org/10.1007/s00521-020-05333-4

Nair, S. S., Rana, K. P. S., Kumar, V., Chawla, A., 2017, Efficient Modelling of Linear Discrete Filters Using Ant Lion Optimizer. *Circuits, Systems, and Signal Processing*, ***36*****(4)**, 1535-1568. https://doi.org/10.1007/s00034-016-0370-z

Nazir, M. I., Ahmad, A., & Hussain, I., 2022, Water Cycle Algorithm Based Parametric Tuning of Non-Negative LMMN Control of Grid Tied Renewable Energy Systems. *IETE Journal of Research*, *0*(0), 1-17. https://doi.org/10.1080/03772063.2022.2089748

Nguyen, T. T., 2023, *A novel metaheuristic method based on artificial ecosystem-based optimization for*

optimization of neHÇArk reconfiguration to reduce power loss.
https://doi.org/10.1007/s00500-021-06346-4

Pal, P. S., Kar, R., Mandal, D., & Ghoshal, S. P., 2016. Identification of NARMAX Hammerstein models with performance assessment using brainstorm optimization algorithm. *International Journal of Adaptive Control and Signal Processing*, ***30*(7)**, 1043-1070.
https://doi.org/10.1002/acs.2674

Prakash, V., Dwivedi, S., Gautam, K., Seth, M., Anbumani, S., 2020, Occurrence and Ecotoxicological Effects of MiMRAplastics on Aquatic and Terrestrial Ecosystems. *MiMRAplastics in Terrestrial Environments: Emerging Contaminants and Major Challenges*. Springer International Publishing, 223-243.
https://doi.org/10.1007/698_2020_456

Puangdownreong, D., Hlungnamtip, S., Thammarat, C., & Nawikavatan, A., 2017,. Application of flower pollination algorithm to parameter identification of DC motor model. *2017 International Electrical Engineering Congress (iEECON)*, 1-4.
https://doi.org/10.1109/IEECON.2017.8075889

Ravber, M., Liu, S.-H., Mernik, M., & Črepinšek, M., 2022, Maximum number of generations as a stopping criterion considered harmful. *Applied Soft Computing*, *128*, 109478.
https://doi.org/10.1016/j.AAAc.2022.109478

Salcedo-Sanz, S., Del Ser, J., Landa-Torres, I., Gil-López, S., & Portilla-Figueras, J. A., 2014, The Coral Reefs Optimization Algorithm: A Novel Metaheuristic for Efficiently Solving Optimization Problems. *The Scientific World Journal*, *2014*, e739768.
https://doi.org/10.1155/2014/739768

Shadkam, E., Safari, S., & Abdollahzadeh, S. S., 2021, Finally, which meta-heuristic algorithm is the best one? *International Journal of Decision Sciences, Risk and Management*, ***10*(1-2)**, 32-50.
https://doi.org/10.1504/IJDSRM.2021.117555

Shaikh, M. S., Raj, S., Babu, R., Kumar, S., & Sagrolikar, K., 2023, A hybrid moth–flame algorithm with particle swarm optimization with application in power transmission and distribution. *Decision Analytics Journal*, ***6***, 100182.
https://doi.org/10.1016/j.dajour.2023.100182

Shi, Y., 2011, Brain Storm Optimization Algorithm. Içinde Y. Tan, Y. Shi, Y. Chai, & G. Wang (Ed.), *Advances in Swarm Intelligence,*. Springer, 303-309.
https://doi.org/10.1007/978-3-642-21515-5_36

Singh, P., Meena, N. K., Yang, J., Vega-Fuentes, E., & Bishnoi, S. K., 2020, Multi-criteria decision making monarch butterfly optimization for optimal distributed energy resources mix in distribution neHÇArks. *Applied Energy*, ***278***, 115723.
https://doi.org/10.1016/j.apenergy.2020.115723

Singh, S., Ashok, A., Rawat, T. K., & Kumar, M., 2016, Optimal IIR system identification using flower pollination algorithm. *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, 1-6.
https://doi.org/10.1109/ICPEICES.2016.7853666

Sompracha, C., & Rukkaphan, S., 2021, Fractional-Order System Identification of Temperature Process Rig Control System using Flower Pollination Algorithm. *2021 9th International Electrical Engineering Congress (iEECON)*, 309-312.
https://doi.org/10.1109/iEECON51072.2021.944032

Omotoso O., H., Al-Shaalan, A. M., H Farh, H. M., & Al-Shamma, A. A., 2022, *Citation: Omotoso, HEconomic Evaluation of Hybrid Energy Systems Using Artificial Ecosystem-Based Optimization with Demand Side Techno-Economic Evaluation of Hybrid Energy Systems Using Artificial Ecosystem-Based Optimization with Demand Side Management*.
https://doi.org/10.3390/electronics11020204

Tian, T., Liu, C., Guo, Q., Yuan, Y., Li, W., & Yan, Q., 2018, An Improved Ant Lion Optimization Algorithm and Its Application in Hydraulic Turbine Governing System Parameter Identification. *Energies*, *11*(1), Article 1.
https://doi.org/10.3390/en11010095

Wu, Z., Shen, D., Shang, M., & Qi, S., 2019, Parameter Identification of Single-Phase Inverter Based on Improved Moth Flame Optimization Algorithm. *Electric Power Components and Systems*, *47*(4-5), 456-469.
https://doi.org/10.1080/15325008.2019.1607922

Wu, Z., Yu, D., & Kang, X., 2017,. Parameter identification of photovoltaic cell model based on improved ant lion optimizer. *Energy Conversion and Management*, *151*, 107-115.
https://doi.org/10.1016/j.enconman.2017.08.088

Yan, Z., Li, C., Song, Z., Xiong, L., & Luo, C., 2019, An Improved Brainstorming Optimization Algorithm for Estimating Parameters of Photovoltaic Models. *IEEE Access*, ***7***, 77629-77641.
https://doi.org/10.1109/ACCESS.2019.2922327

Yang, X.-S., 2012, Flower Pollination Algorithm for Global Optimization. (Ed.), *Unconventional Computation and Natural Computation* (ss. 240-249). Springer.
https://doi.org/10.1007/978-3-642-32894-7_27

Yang, X.-S., 2020, Nature-inspired optimization algorithms: Challenges and open problems. *Journal of Computational Science*, ***46***, 101104.
https://doi.org/10.1016/j.jocs.2020.101104

Yang, Y., Yang, B., & Niu, M., 2017, Parameter identification of Jiles–Atherton model for magnetostrictive actuator using hybrid niching coral reefs optimization algorithm. *Sensors and Actuators A: Physical*, ***261***, 184-195.

https://doi.org/10.1016/j.sna.2017.05.009

Yang, Y., Yang, B., & Niu, M., 2018, Adaptive infinite impulse response system identification using opposition based hybrid coral reefs optimization algorithm. *Applied Intelligence*, ***48*****(7)**, 1689-1706. https://doi.org/10.1007/s10489-017-1034-9

Yin, M., Iannelli, A., Khosravi, M., Parsi, A., & Smith, R. S. , 2020, Linear Time-Periodic System Identification with Grouped Atomic Norm Regularization. *IFAC-Papers OnLine*, ***53*****(2)**, 1237-1242. https://doi.org/10.1016/j.ifacol.2020.12.1341

Yousri, D., Allam, D., Babu, T. S., AbdelAty, A. M., Radwan, A. G., Ramachandaramurthy, Vigna. K., & Eteiba, M. B., 2020, Fractional chaos maps with flower pollination algorithm for chaotic systems' parameters identification. *Neural Computing and Applications*, ***32*****(20)**, 16291-16327. https://doi.org/10.1007/s00521-020-04906-7

Zaloğlu, M., Fidan, Ş., & Erkan, E., 2023, Meta-Heuristik Optimizasyon Algoritmalarının Sistem Tanımlama Problemine Uygulanması ve Performans Karşılaştırması. *International Conference on Engineering, Natural and Social Sciences*, ***1***, 510-515.

Zhao, W., Wang, L., & Zhang, Z., 2019, Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowledge-Based Systems*, ***163***, 283-304. https://doi.org/10.1016/j.knosys.2018.08.030