TURKISH JOURNAL OF ENGINEERING RESEARCH AND EDUCATION

Türk Mühendislik Araştırma ve Eğitimi Dergisi
Turkish Journal of Engineering Research and Education

3(2), 112-122, (2024)

ARAŞTIRMA MAKALESİ

Yılan Optimizasyon Algoritması ile CEC 2019 Problem Seti ve Mühendislik Problemlerinin Çözümü

Solving of CEC 2019 Problem Set and Engineering Problems with the Snake Optimization Algorithm

Merve Arslan[1,*] , Gürcan Yavuz[2]
[1] Kütahya Dumlupınar Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği Bölümü, Kütahya, Türkiye.
[2] Kütahya Dumlupınar Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Kütahya, Türkiye.

Geliş / Received: 11.09.2024
Kabul / Accepted: 28.10.2024
***Sorumlu Yazar:** **Merve Arslan** merve.arslan@dpu.edu.tr

ÖZ: Optimizasyon problemlerinin çözümünde Meta-sezgisel algoritmaların kullanımı yaygındır. Bu çalışmada, CEC-2019 problem setinin çözümünde Yılan Optimizasyonu (Snake Optimization, SO) Algoritması kullanılmaktadır. Elde edilen sonuçlar literatürde sık kullanılan, güncel ve başarılı sonuçlar elde edilmiş diğer optimizasyon algoritmalarından Koati Optimizasyon Algoritması (Coati Optimization Algorithm, COA), Bernstein Arama DE Algoritması (Bernstein-Search Differential Evolution, BSDE), Karşıt-Karşılıklı Öğrenme DE (Oppositional-Mutual Learning Differential Evolution, OMLDE), Ağırlıklı Diferansiyel Evrim (Weighted Differential Evolution, WDE), Parçacık Sürü Optimizasyonu (Particle Swarm Optimization, PSO), Başarı Geçmişi Uyarlaması DE (Success-History Adaptation Differential Evolution, SHADE), İsteğe Bağlı Harici Arşivle Uyarlanabilir DE (Adaptive Differential Evolution with Optional External Archive, JADE), Kovaryans Matrisi Uyarlama Evrim Stratejisi (Covariance Matrix Adaptation Evolution Strategy, CMAES) ve Bernstein Operatörü ve Kırılmış Karşıt Karşılıklı Öğrenme Diferansiyel Evrim (Bernstein Operator and Refracted Oppositional Mutual Learning Differential Evolution, BROMLDE) ile karşılaştırılmaktadır. İlklendirme aşamasında, çözüm uzayını daha iyi keşfetmesi adına ve optimum çözüme yakınsamasının hızlandırılması açısından Latin hiperküp tekniği kullanılmaktadır. Geliştirilen yılan algoritmasının performansı CEC 2019 problem çözümünde değerlendirilmektedir. Sonuçları analiz etmek için Friedman testi kullanılmaktadır. Elde edilen sonuçlara göre Yılan algoritmasının 10 problemden 6'sında rakiplerinden daha iyi sonuçlar verdiği görülmektedir. Ek olarak Yılan Optimizasyonu ile iki adet mühendislik problemi çözümü yapılmaktadır. Mühendislik problemi çözüm sonuçları literatürde bulunan diğer optimizasyon algoritmaları sonuçları ile kıyaslanmaktadır. Sonuç olarak Yılan Algoritması diğer optimizasyon algoritmaları ile karşılaştırıldığında rekabetçi sonuçlar vermektedir.

Anahtar Kelimeler: Yılan Optimizasyon Algoritması, Latin Hiperküp Örnekleme, CEC 2019, Dişli Sistemi Tasarımı Problemi, Borulu Kolon Tasarımı

ABSTRACT: Meta-heuristic algorithms are commonly used in solving optimization problems. In this study, the Snake Optimization (SO) Algorithm is employed to solve the CEC-2019 problem set. The obtained results are compared with other optimization algorithms that are frequently used in the literature and have achieved recent successful outcomes, including the Coati Optimization Algorithm (COA), Bernstein-Search Differential Evolution (BSDE), Oppositional-Mutual Learning Differential Evolution (OMLDE), Weighted Differential Evolution (WDE), Particle Swarm Optimization (PSO), Success-History Adaptation Differential Evolution (SHADE), Adaptive Differential Evolution with Optional External Archive (JADE), Covariance Matrix Adaptation Evolution Strategy (CMAES), and Bernstein Operator and Refracted Oppositional Mutual Learning Differential Evolution (BROMLDE). In the initialization phase, the Latin hypercube technique is used to explore the solution space more effectively and accelerate convergence to the optimal solution. The performance of the developed Snake Algorithm is evaluated in solving the CEC 2019 problem set. The Friedman test is used to analyze the results. According to the results, the Snake Algorithm outperforms its competitors in 6 out of 10 problems. Additionally, two engineering problems are solved using the Snake Optimization algorithm.

Merve Arslan, Orcid: 0000-0002-2867-6198 / Gürcan Yavuz, Orcid: 0000-0002-2540-1930

The results of the engineering problem solutions are compared with the results of other optimization algorithms in the literature. In conclusion, the Snake Algorithm provides competitive results when compared to other optimization algorithms.

Keywords: Snake Optimization Algorithm, Latin Hypercube Sampling, CEC 2019, Gear Train Design Problem, Tubular Column Design Problem

1. GİRİŞ

Bir problemin en uygun çözümünü bulmak için veya en iyiye yakın çözümü elde edebilmek için optimizasyon algoritmalarına başvurulmaktadır. Tam bu noktada probleme ait uygunluk fonksiyonunu optimize etmek başlıca amaç olmaktadır. Genel gerçek dünya problemlerinin veya mühendislik problemlerinin çözümü için literatürde pek çok optimizasyon algoritması mevcuttur. Bazı temel optimizasyon algoritmalarına Genetik Algoritma (Genetic Algorithms, GA) [1], PSO [2] ve Diferansiyel Evrim Algoritması (Differential Evolution, DE) [3] örnek olarak verilebilmektedir. Genel itibariyle optimizasyon algoritmaları kullanım şekilleri veya ortaya çıkış sebepleri gibi durumlardan dolayı alt başlıklara ayrılabilmektedir. Bu algoritmalar temelde matematiksel veya sezgisel olarak iki alt başlıkta toplanmaktadır [4].

Bu kapsamda optimizasyon algoritmalarının bir dalını da Meta-sezgisel algoritmalar oluşturmaktadır. Sezgisel algoritmaların yöntemleri bir araya getirilerek verimli bir şekilde kullanımı sonucunda Meta-sezgisel adı verilen algoritmalar geliştirilmiştir. Meta-sezgisel algoritmalar üst seviye yöntemlerdir. Etkili arama işlemleri sayesinde optimum çözüme hızlı bir şekilde varılmaktadır. Eğer çözülmek istenen problemin çözüm uzayı büyükse meta-sezgisel algoritmaların kullanımı daha avantajlıdır. Şöyle ki çözüm uzayı geniş olan bir probleme matematiksel algoritma uygulandığı durumda algoritma tüm çözüm uzayını taraması gerektiğinden maliyet fazla olabilmektedir [4].

Geçmişten günümüze kadar kabul görmüş birçok meta-sezgisel algoritma bulunmaktadır. Bu algoritmalara ilk olarak Genetik algoritma örnek olarak verilebilmektedir. Genetik algoritma 1960'yıllarda John Holland ve ekibi tarafından geliştirilmiştir. John Holland çaprazlama ve birleştirme kavramları üzerine çalışan ilk kişidir [1].

Diğer bir meta-sezgisel optimizasyon algoritması Karınca Kolonisi Optimizasyonu (Ant Colony Optimization, ACO)'dur. Marco Dorigo tarafından 1991 yılında ortaya atılmıştır. Algoritma oluşturulurken temelde doğadaki karınca kolonilerinin yiyecek arama ve izleme durumlarından esinlenilmiştir. Doğal yaşamda karıncalar besin kaynağını bulmak amacıyla feromonlar yani kimyasal izler bırakmaktadırlar. Burada, kolonideki diğer karıncaların bu feromonları takip ederek en fazla feromon bırakılan yolun takip edilmesi amaçlanmaktadır. Karınca Kolonisi Algoritmasında karıncalar kendilerine göre bir yol oluşturmaktadırlar. Bu oluşturulan yollar çözümü temsil etmektedir. Feromon miktarı ve uzaklık parametreleri yol oluşumunda etkili olmaktadır [5].

Bir diğer optimizasyon algoritması olan PSO, grup şeklinde davranan canlıların hareketlerinden yola çıkılarak oluşturulmuş bir optimizasyon algoritmasıdır. Örnek olarak kuş sürüsü verilebilmektedir. Algoritmada her bir parçacık çözüm uzayındaki bir veriyi temsil etmektedir. Parçacıkların konumları problemdeki bir çözümü ifade etmektedir [2]. Farklı bir optimizasyon algoritması da Storn ve Price tarafından geliştirilmiş olan DE' dir. Algoritmada bireyler oluşturulurken mutasyon, çaprazlama, seçim gibi işlemler yapılmaktadır. Popülasyon tabanlı bir algoritmadır [3].

Başka bir optimizasyon algoritması olan Benzetim Tavlama (Simulated Annealing, SA), Kirkpatrick, Gelatt ve Vecchi tarafından geliştirilmiştir. Algoritmada esinlenilen fikir fizikte maddenin yavaş bir şekilde soğumasıdır. Şöyle ki metal bir madde yüksek derece bir ısıda eritilmektedir. Akabinde yavaş yavaş soğuması sağlanmaktadır. Tavlama kavramı buradan gelmektedir. Bu düşünceden yola çıkarak algoritma çözüm uzayında dolaşır ve sonuç olarak da global optimumu bulmayı hedeflemektedir. Başlangıç noktası genellikle rastgele olarak belirlenmektedir.

Algoritma bir sıcaklık parametresi kullanmaktadır [6].

Farklı bir optimizasyon stratejisi olan Ateşböceği Algoritması (Firefly Algorithm, FA), doğada bulunan ateşböceklerinin yanma-sönme durumlarından esinlenilerek oluşturulmuş bir optimizasyon algoritmasıdır. Cambridge Üniversitesinde Yang tarafından 2009 yılında geliştirilmiştir. Algoritmada hedeflenen amaç fonksiyonu ateşböceklerinin parlaklıklarına göre oluşturulmuştur. Ateşböceklerinin hepsi tek cinsiyetlidir. Parlaklık arttıkça ateşböceğinin çekiciliği de artmaktadır. Buna karşın mesafe arttıkça çekicilik azalmaktadır [7].

SO, yılanların yiyecek arama ve çiftleşme davranışlarından yola çıkılarak matematiksel formüllerin oluşturulduğu bir meta-sezgisel optimizasyon algoritmasıdır. 2022 yılında gerçekleştirilen çalışma, CEC-2017 problem seti kullanılarak test edilmiştir. Ek olarak Hız Düşürücü Tasarımı, Kaynaklı Kiriş Tasarımı, Basınçlı Kap Tasarımı ve Germe-Sıkıştırma Yayı Tasarımı olmak üzere 4 adet mühendislik probleminin çözümü yapılmıştır. Sonuçlar literatürde sık kullanılan optimizasyon algoritmaları ile kıyaslandığında başarılı sonuçlar verdiği görülmektedir [8].

Literatürde bulunan bu ve benzeri algoritmaları test etmek amacıyla genellikle CEC problem setleri kullanılmaktadır. Bunlardan bir tanesi 2019 yılındaki yarışmaya ait CEC-2019 problem setidir. Örneğin, 2019 yılında DISH Algoritması kullanılarak CEC-2019 problem setinin çözümü yapılmıştır [9]. Başka bir örnek, 2023 yılında yeni bir şempanze sinüs kosinüs algoritması CEC-2019 ölçü seti kullanılarak test edilmiştir [10]. Hibrit Gri Kurt-Sinüs Kosinüs algoritmasının çözümü de CEC-2019 kullanılarak test edilmektedir [11].

Sezgisel algoritmalar, optimizasyon problemlerinin çözümünde sıkça tercih edilerek etkili sonuçlar üretmektedir. Fakat bazı durumlarda algoritma yetersiz kalabilmektedir. Bu sebeple algoritmanın belli adımlarına çeşitli yöntemler entegre edilerek performans artışı sağlanabilmektedir. Örneğin 2019'da yapılan bir çalışmada Ağaç tohum algoritmasını (Tree-Seed Algorithm, TSA) geliştirmek adına rastgele seçim yerine kaotik haritalardan elde edilen sayılar kullanılmaktadır. Bu sayede performans artışı sağlanmaktadır [12].

Meta sezgisel bir algoritmanın temel bileşenleri, arama alanı, değerlendirme fonksiyonu, konum güncelleme formülü, yeni çözümleri kabul etme koşulları ve sonlandırma kriterleri gibi unsurlar, algoritmanın verimliliği açısından kritik bir öneme sahiptir. Bu bileşenler, algoritmanın performansını ve çözüm kalitesini doğrudan etkileyen unsurlar olarak işlev görmektedir. Bu yapıların tasarımı, işleyişi ve geliştirilmesine yönelik çalışmalar mevcuttur. Örnek olarak bir çalışmada, Aritmetik optimizasyon algoritmasına Levy rastgele adımlarının (Lévy random steps, LRS) entegre edilerek aday çözümlerin, algoritmanın arama alanının henüz keşfedilmemiş bölgelerini zaman zaman keşfetmesi amaçlanmaktadır [13].

Bir diğer örnek çalışma, Güve alevi optimizasyonu (Moth Flame Optimization, MFO) algoritmasının yakınsama hızını ve küresel arama yeteneğini artırmak amacıyla IMFO (Improved Moth Flame Optimization) algoritmasının geliştirilmesi çalışmasıdır. Güvelerin konumlarını güncellemek için uygunluğa bağlı bir ağırlık faktörü kullanarak keşif (exploration) ve sömürü (exploitation) arasında hibrit bir faz oluşturur. Bu hibrit yaklaşım, algoritmanın performansını artırarak daha etkili bir optimizasyon süreci sağlamaktadır [14].

Karga arama algoritmasında (Crow Search Algorithm, CSA), arama yeteneklerini geliştirmek amacıyla uyarlanabilir bir turnuva tabanlı seçim mekanizması kullanılarak yeni bir çalışma gerçekleştirilmektedir [15].

Kril sürüsü (Krill Herd, KH) algoritmasında performansı artırmak amacıyla rastgele sayı başlatma yöntemi yerine düşük tutarsızlık dizileri kullanılmaktadır. Bu çalışmada, Faure, Sobol ve Van der Corput dizileri ile başlatmanın etkisi incelenmektedir [16].

Literatürde sayısız optimizasyon algoritması mevcuttur. Fakat herhangi bir optimizasyon algoritması tüm problemlerde her zaman en iyi sonucu vermez. Nitekim NFL (No Free Lunch) teoremine göre bir optimizasyon algoritması bir problemde diğerlerinden daha iyi performans gösteriyorsa, başka bir problem kümesinde daha

kötü performans göstermektedir [17]. Bu sebeple optimizasyon çalışmalarında odaklanılması gereken asıl nokta problem bazında yeni algoritmaların geliştirilmesi ve var olan algoritmaların performansını artırmaya yönelik teknikler tasarlanmasıdır.

Bu çalışmada, literatürde bulunan ve başarılı sonuçlar veren bir meta-sezgisel optimizasyon algoritması olan SO kullanılarak CEC-2019 [18] ölçü setinin çözümü gerçekleştirilmiştir. Friedman testi ile analizi yapılmıştır. Friedman testi, iki veya daha fazla yöntemi karşılaştırmak amacıyla kullanılan, parametrik olmayan bir istatistiksel testtir [19]. Literatürde yer alan çeşitli optimizasyon algoritmaları ile karşılaştırıldığında başarılı sonuçlar verdiği görülmüştür. Yılan optimizasyon algoritmasını geliştirmek adına ilklendirme adımında rastgele sayı seçimi yerine Latin Hiperküp Örnekleme (Latin Hypercube Sampling, LHS) yöntemi kullanılarak popülasyon oluşturulmuştur. Bu, algoritmanın çözüm uzayını daha etkili bir biçimde araştırmasına ve daha iyi çözümler bulmasına yardımcı olmaktadır. Ek olarak iki adet mühendislik problemi Yılan algoritması kullanılarak çözülmüştür.

Makalenin geri kalanı şu şekildedir: Yılan algoritmasının çalışma sistemi, CEC 2019 Problem seti ve Latin hiperküp örnekleme yöntemi 2. başlık altında verilmiştir. Deneyler sonucunda elde edilen sonuçlara 3. başlık altında yer verilmiştir. Son olarak 4. başlık sonuç şeklinde sonlandırılmıştır.

2. MATERYAL VE METOT

2.1 Yılan Algoritması

Yılan Optimizasyonu, 2022 yılında Hashim ve Hussien tarafından geliştirilmiş bir meta-sezgisel optimizasyon algoritmasıdır [8]. Yılanların çiftleşme davranışlarından esinlenerek ortaya çıkmıştır. Şöyle ki yılanlarda çiftleşme sıcaklığın düşük olması ve yiyeceğin var olması durumunda gerçekleşmektedir. Yiyecek olmaması halinde yılanlar yiyecek arayacaklardır. Arama işleyişi "Keşif (Exploration)" ve "Kullanım (Exploitation)" olmak üzere iki bölümden oluşmaktadır. Keşif, yerin soğuk olması ve yiyeceğin olmaması durumunda yılanın yiyecek aramasını temsil etmektedir. Kullanım ise yiyecek var fakat sıcaklığın yüksek olması halinde yılanlar sadece var olan yiyecekleri yemektedirler. Öte yandan eğer yiyecek var ve alan soğuk ise çiftleşme gerçekleşecektir. Böyle bir durumda da iki mod vardır: Birincisi her erkeğin dişiyi elde edebilmek için mücadele edeceği Dövüş Modu, ikincisi de çiftleşme modudur.

2.1.1 Başlatma

Yılan optimizasyon algoritmasının başlatılması rastgele bir popülasyon oluşturularak yapılmaktadır. Başlangıç popülasyonu oluşturabilmek için kullanılacak matematiksel model Denklem 1'de verilmiştir.

$$X_i = X_L + r \times (X_U - X_L) \quad (1)$$

Verilen Eşitlik 1 'de X_i *i.* bireyin konumunu verirken, r [0-1] aralığında rastgele bir sayıdır. X_L, problemin alt sınırını temsil ederken X_U, üst sınırı göstermektedir.

2.1.2 Sürüyü İki Gruba Ayırma

Bu aşamada, erkek ve dişi sayısının eşit olduğu varsayılmaktadır. Sürüyü erkek ve dişi olarak ikiye bölmek için kullanılacak matematiksel modeller Denklem 2 ve Denklem 3' de verilmiştir.

$$N_m \approx N/2 \quad (2)$$

$$N_f = N - N_m \quad (3)$$

Burada N ile birey sayısı temsil edilmektedir. N_m erkek birey sayısını ifade ederken N_f kadın birey sayısını göstermektedir.

2.1.3 Grubu Değerlendirme ve Sıcaklık-Yiyecek Miktarını Belirleme

Grubu değerlendirme adımında en iyi erkek ($f_{best,m}$) ve en iyi kadın ($f_{best,f}$) bulunmaktadır. Ek olarak yiyecek konumu (f_{food}) da bulunmaktadır. Sıcaklık ($Temp$) hesaplama için gereken formül Denklem 4' de verilmiştir.

$$Temp = exp\left(\frac{-t}{T}\right) \quad (4)$$

Verilen eşitlikte T maksimum iterasyonu temsil ederken, t mevcut iterasyonu belirtmektedir. Yiyecek miktarı (Q) hesaplama için gereken matematiksel model ise Denklem 5' de verilmiştir.

$$Q = c_1 * exp\ (\frac{t-T}{T}) \quad (5)$$

Eşitlikte ifade edilen c_1 sabittir. Değeri 0.5'tir.

2.1.4 Keşif ve Sömürü

Keşif aşamasında yiyecek yoktur. Eğer Q<eşik (0.5) ise yılanlar yiyecek arayacaklardır. Keşif aşaması için gereken matematiksel hesaplama Denklem 6' da verilmiştir.

$$X_{i,m}(t+1) = X_{rand,m}(t) \pm c_2 \times A_m \times ((X_U - X_L) \times rand + X_L \quad (6)$$

Verilen eşitlikte $X_{i,m}$ i.erkek pozisyonunu, $X_{rand,m}$ rastgele erkek pozisyonunu temsil etmektedir. Ek olarak *rand* [0-1] aralığında rastgele bir sayıdır ve c_2 sabittir, 0.05dir. A_m erkeğin yiyecek bulabilme kabiliyetidir ve Denklem 7' deki gibi hesaplanmaktadır.

$$A_m = \exp\ (\frac{f_{rand,m}}{f_{i,m}}) \quad (7)$$

Denklemde verilen $f_{rand,m}$, $X_{rand,m}$ 'in uygunluk(fitness) değeridir. Bunun yanında $f_{i,m}$ ise i. erkek grubundaki bireyin uygunluğudur. Aynı ifadeler dişi yılan için de geçerlidir. Şöyle ki dişi yılan için hesaplama formülü Denklem 8 ve 9' da verilmiştir.

$$X_{i,f} = X_{rand,f}(t+1) \pm c_2 \times A_f \times ((X_U - X_L) \times rand + X_L \quad (8)$$

$$A_f = \exp\ (\frac{f_{rand,f}}{f_{i,f}}) \quad (9)$$

Sömürü aşamasına gelindiğinde, eğer Q> eşik ve sıcaklık> eşik (0.6) ise yılanlar sadece yiyecek için harekete geçecektir ve matematiksel modeli Denklem 10' da verilmiştir.

$$X_{i,j}(t+1) = X_{food} \pm c_3 \times Temp \times rand \times (X_{food} - X_{i,j}(t)) \quad (10)$$

Eşitlikte verilen $X_{i,j}$ bireyin konumudur. X_{food} en iyi bireylerin konumunu temsil ederken c_3 sabittir, 2'dir.

Eğer sıcaklık < eşik (0.6) olursa yılanlar dövüş veya çiftleşme moduna geçecektir. Dövüş modu için gereken matematiksel model Denklem 11 ve Denklem 12' de verilmiştir.

$$X_{i,m}(t+1) = X_{i,m}(t) + c_3 \times FM \times rand \times (Q \times X_{best,f} - X_{i,m}(t)) \quad (11)$$

$$X_{i,f}(t+1) = X_{i,f}(t+1) + c_3 \times FF \times rand \times (Q \times X_{best,m} - X_{i,f}(t+1)) \quad (12)$$

Eşitliklerde verilen $X_{i,m}$ i. erkeğin konumunu, $X_{i,f}$ i. dişinin konumunu, $X_{best,f}$ en iyi dişinin konumunu, $X_{best,m}$ en iyi erkeğin konumunu, FM erkeğin dövüş kabiliyetini, FF kadının dövüş kabiliyetini temsil etmektedir. Dövüş yeteneklerini hesaplamak için gereken formüller Denklem 13 ve Denklem 14' de verilmiştir.

$$FM = exp\ (\frac{-f_{best,f}}{f_i}) \quad (13)$$

$$FF = exp\ (\frac{-f_{best,m}}{f_i}) \quad (14)$$

Eşitlikte verilen f_i ajanın uygunluğudur. Çiftleşme modunun hesaplamaları Denklem 15 ve Denklem 16' da verilmiştir.

$$X_{i,m}(t+1) = X_{i,m}(t) + c_3 \times M_m \times rand \times (Q \times X_{i,f}(t) - X_{i,m}(t)) \quad (15)$$

$$X_{i,f}(t+1) = X_{i,f}(t) + c_3 \times M_f \times rand \times (Q \times X_{i,m}(t) - X_{i,f}(t)) \quad (16)$$

Verilen eşitliklerde M_m, M_f sırasıyla erkek ve dişi çiftleşme yeteneklerini temsil etmektedir ve Denklem 17 ve Denklem 18'deki gibi hesaplanmaktadır.

$$M_m = \exp\ (\frac{-f_{i,f}}{f_{i,m}}) \quad (17)$$

$$M_f = \exp\ (\frac{-f_{i,m}}{f_{i,f}}) \quad (18)$$

Son olarak yumurtanın çatlaması halinde en kötü erkek ve dişi seçilip değiştirilmektedir. Hesaplama için gerekli matematiksel model Denklem 19 ve Denklem 20' de verilmiştir.

$$X_{worst,m} = X_L + rand \times (X_U - X_L) \quad (19)$$

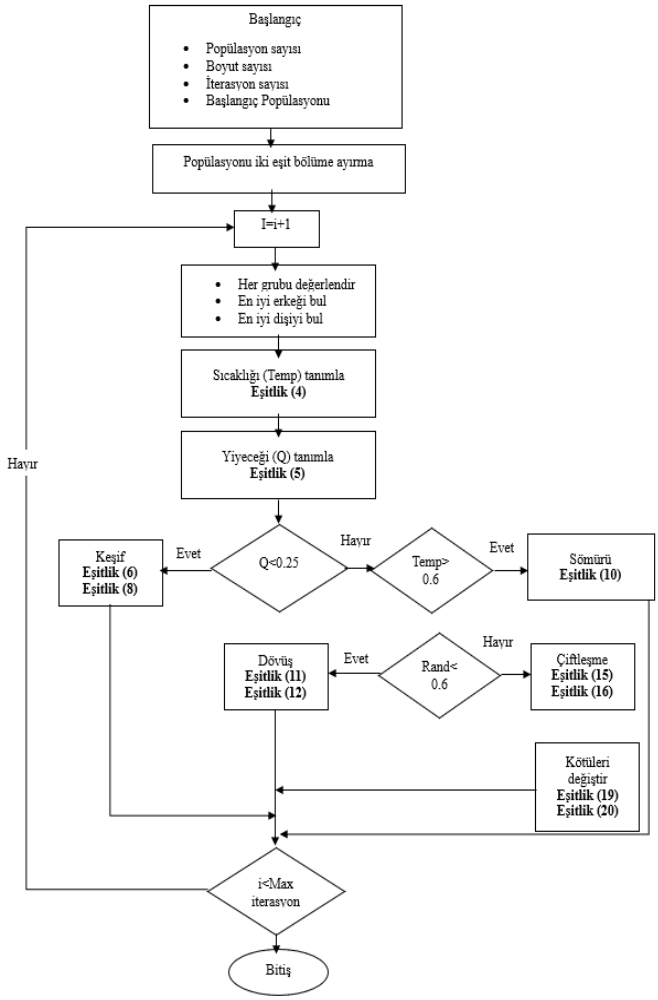
$$X_{worst,f} = X_L + rand \times (X_U - X_L) \quad (20)$$

Verilen eşitliklerde $X_{worst,m}$, $X_{worst,f}$ sırasıyla erkek grubundaki en kötü bireyi ve dişi grubundaki en

kötü bireyi temsil etmektedir. Algoritmaya ait akış diyagramı Şekil 1'de verilmiştir.

2.2 CEC 2019

CEC, optimizasyon algoritmalarını test etmek amacıyla başvurulan bir problem setidir. Bu çalışma kapsamında kullandığımız CEC 2019, 100 basamak problemi olarak da tanımlanmaktadır. Söz konusu set, farklı boyut ve kısıtlara sahip 10 adet problemden oluşmaktadır. Problemler çeşitli matematiksel fonksiyonları içermektedir [18]. Tablo 1'de CEC 2019 fonksiyonları ve özelliklerinin yer aldığı tablo verilmiştir.



Şekil 1: Yılan algoritması akış şeması [8].

Tablo 1: CEC 2019 fonksiyonları.

Fonksiyon No	Fonksiyonlar	Boyut	Aralık
1	Storn's Chebyshev Polynomial Fitting Problem	9	[-8192,8192]
2	Inverse Hilbert Matrix Problem	16	[-16384,16384]
3	Lennard-Jones Minimum Energy Cluster	18	[-4,4]
4	Rastrigin's Function	10	[-100,100]
5	Griewangk's Function	10	[-100,100]
6	Weierstrass Function	10	[-100,100]
7	Modified Schwefel's Function	10	[-100100]
8	Expanded Schaffer's F6 Function	10	[-100,100]
9	Happy Cat Function	10	[-100,100]
10	Ackley Function	10	[-100,100]

2.3 Latin Hiperküp Örnekleme (Latin Hypercube Sampling-LHS)

Latin Hiperküp 1979 yılında önerilmiş bir örnekleme yöntemidir [20]. Söz konusu yöntem örnekleme alanını belli kısımlara bölerek her parçadan örnek almayı amaçlamaktadır. Bu sayede alınan örnekler ile homojen bir temsil sağlanmaktadır. Seçilen noktalar sonrasında eşleştirilerek kombinasyonlar oluşturulmaktadır. Oluşturulan bu noktalar optimizasyon algoritmalarının başlangıç popülasyonu olarak

kullanılmaktadır. Bu yöntem optimizasyon algoritmalarında kritik öneme sahip aşamalardan ilki olan başlangıç popülasyon ilklendirme adımında kullanılması algoritmanın performansında etkilidir. Algoritmanın çözüm uzayını daha iyi keşfetmesinde ve optimum çözüme yakınsamasında faydalı olabilmektedir.

3. BULGULAR

Deneysel çalışmalar kapsamında orijinal yılan algoritması ve LHS tekniği kullanılarak geliştirilmiş yılan algoritması ile CEC 2019 problem seti çözülmüştür. Akabinde literatürdeki diğer optimizasyon algoritmaları ile karşılaştırılması yapılmıştır. Deneysel çalışmalar kapsamında iki adet mühendislik probleminin çözümü de gerçekleştirilmektedir. Yapılan bütün testler Windows 11 Home işletim sistemine sahip, 11th Gen Intel(R) Core(TM) i5-11300H işlemcili, 16 GB hafızalı bilgisayarda yapılmıştır. Deneylerde adil bir karşılaştırma yapılabilmesi adına parametreler Wu ve diğerlerinin çalışmasına göre ayarlanmaktadır [21]. Kullanılan parametre değerleri Tablo 2'de verilmektedir.

Tablo 2: Parametre tablosu.

Parametre	Değeri
Popülasyon boyutu	100
MaxFES	10000
Çalıştırma sayısı	30

Deneysel çalışmalar kapsamında kullanılan algoritma, Tablo 2'de verilen popülasyon boyutu 100, Maksimum fonksiyon değerlendirme sayısı (Max FES) 10000 ve 30 çalıştırma kullanılarak test edilmiştir.

3.1 CEC 2019 Sonuç Karşılaştırması

Yılan algoritması kullanılarak çözülen CEC 2019 setinden elde edilen sonuçlar, Wu ve diğerlerinin çalışmasında yer alan BSDE [22], OMLDE [23], WDE [24], PSO [25], SHADE [26], JADE [27], CMAES [28], BROMLDE [21] yöntemlerinin sonuç değerleri ile karşılaştırılmıştır. Yılan algoritmasının ilklendirme adımında LHS tekniği kullanılarak algoritmanın performansı değerlendirilmiştir. Kullanılan parametre değerleri söz konusu çalışmadakiyle aynıdır. Karşılaştırma tablosu Tablo 3 ve Tablo 4' de verilmiştir. Tabloya Wu ve diğerlerinin önerdikleri yöntem sonuçları ve karşılaştırdıkları algoritma sonuçları da eklenmiştir. Ek olarak CEC 2019 problem seti COA algoritmasıyla da çözülmüştür ve elde edilen sonuçlar da tabloya eklenmiştir. Deney sonuçlarından elde edilen değerler Friedman testi kullanılarak analiz edilmiştir. Friedman testi sonuçları Tablo 5' de verilmiştir.

Tablo 3: CEC 2019 test sonuçları (F1-F5)

	F1		F2		F3		F4		F5	
Metrik	Ortalama	Std	Ortalama	Std	Ortalama	Std	Ortalama	Std	Ortalama	Std
SO (LHS)	**-9.8921E-03**	1.9434E-01	**7.3959E-03**	6.0533E-02	**-3.9488E-01**	5.4195E-01	7.6887E+00	3.8638E+01	9.8294E+00	4.3974E+01
SO	1.7082E-04	4.3557E-02	6.4056E-01	1.5515E+01	2.7029E-01	1.9588E+0	**6.5932E+0**	5.3889E+01	1.0453E+01	4.5216E+01
COA	8.8494E+05	1.3201E+02	5.4097E+02	6.3275E+02	8.2211E+0	3.6464E+0	1.1009E+02	4.1901E+01	1.3993E+02	6.4205E+01
BSDE	2.3365E+10	1.3624E+10	8.6463E+0	3.8929E+0	1.2702E+0	9.0900E−06	2.7441E+02	7.4444E+0	1.7300E+0	1.3887E−01
OMLDE	1.3686E+09	2.1840E+09	1.3452E+02	1.2929E+02	1.2703E+0	2.0974E−04	4.0491E+03	2.1316E+03	2.5679E+0	3.1217E−01
WDE	6.0428E+10	3.4700E+10	1.0931E+03	4.4152E+02	1.2703E+0	1.3621E−04	2.5733E+03	7.7972E+02	2.5137E+0	1.8449E−01
PSO	2.2371E+10	1.7275E+10	2.7232E+0	5.3829E+0	1.2703E+0	7.5574E−04	1.2030E+03	6.6790E+02	1.6473E+0	4.3138E−01
SHADE	8.8385E+09	4.8209E+09	2.6338E+0	4.5771E+0	1.2702E+0	6.5941E−06	8.4969E+0	1.3287E+0	1.6712E+0	1.0437E−01
JADE	3.1163E+10	1.5759E+10	**1.9322E+0**	2.7313E+0	1.2702E+0	1.9165E−05	8.4365E+0	2.5848E+0	1.5414E+0	1.1315E−01
CMAES	8.6816E+10	1.6158E+11	7.5171E+04	1.2393E+04	1.2705E+0	3.1064E−03	1.9674E+03	1.0480E+04	**1.0218E+0**	1.1879E−01
BROMLDE	1.1022E+09	1.6019E+09	2.5057E+0	1.0313E+0	1.2702E+0	4.5257E−06	2.1898E+02	1.6873E+02	1.5682E+0	8.3493E−02

Tablo 4: CEC 2019 test sonuçları (F6-F10)

	F6		F7		F8		F9		F10	
Metrik	Ortalama	Std	Ortalama	Std	Ortalama	Std	Ortalama	Std	Ortalama	Std
SO (LHS)	**-2.5529E-01**	3.0807E+01	1.0901E+01	6.0694E+01	1.1699E+01	4.8099E+01	**-1.0689E+01**	3.2667E+01	**-2.0822E+01**	6.1447E+01
SO	3.8577E-01	3.3825E+01	**8.7866E+0**	4.9881E+01	4.4729E+0	5.4267E+01	-1.6148E+01	3.0985E+01	6.4077E+0	7.6726E+01
COA	1.1332E+01	2.7426E+0	2.0345E+03	9.8438E+02	**4.3501E+0**	2.1119E-01	3.3848E+0	1.3150E+0	2.0981E+01	2.9052E-01
BSDE	8.6195E+0	6.7997E−01	2.9658E+02	1.0557E+02	5.4627E+0	3.6571E−01	5.0168E+0	1.8876E+0	2.0021E+0	7.7741E−01
OMLDE	1.1608E+0	6.7551E−01	9.4097E+02	1.8477E+02	6.5042E+0	2.7377E−01	3.3530E+02	1.8130E+02	2.0499E+0	1.8716E−01
WDE	9.4992E+0	8.8662E−01	5.4830E+02	1.2762E+02	5.9984E+0	2.1249E−01	5.5962E+02	2.1148E+02	2.0311E+0	6.6710E−02
PSO	8.5524E+0	1.1533E+0	3.2512E+02	2.8856E+02	5.4877E+0	6.8809E−01	3.4069E+0	4.0954E+0	2.0041E+0	5.4341E−02
SHADE	1.0043E+0	7.6363E−01	5.8042E+02	1.4751E+02	5.9982E+0	3.0162E−01	3.3978E+0	2.6188E−01	2.0314E+0	1.8459E−01
JADE	9.0537E+0	6.5637E−01	4.1793E+02	9.2663E+0	5.6098E+0	4.7670E−01	3.0946E+0	3.1284E−01	2.0059E+0	5.6194E−01
CMAES	1.3562E+0	6.6821E−01	1.0757E+03	2.1772E+02	6.4208E+0	1.6190E+0	2.5553E+0	9.5374E−02	2.1022E+0	8.8160E−02
BROMLDE	8.5392E+0	4.8619E−01	2.8929E+02	9.4926E+0	5.3501E+0	2.4341E−01	4.5696E+0	6.2230E+0	**1.9994E+0**	7.1465E−01

Tablo 5: Friedman Testi.

No	SO (LHS)	SO	COA	BSDE	OMLDE	WDE	PSO	SHADE	JADE	CMAES	BROMLDE
F1	2	1	3	8	5	10	7	6	9	11	4
F2	1	2	9	7	8	10	6	5	3	11	4
F3	1	2	11	4,5	8	8	8	4,5	4,5	10	4,5
F4	2	1	5	7	11	10	8	4	3	9	6
F5	9	10	11	6	8	7	4	5	2	1	3
F6	1	2	11	8	4	10	7	3	9	5	6
F7	2	1	11	4	9	7	5	8	6	10	3
F8	11	2	1	4	10	8	5	7	6	9	3
F9	2	1	5	9	10	11	7	6	4	3	8
F10	1	10	11	3	8	6	4	7	5	9	2
Ort. sıralama	3,2	3,2	7,8	6,05	8,1	8,7	6,1	5,55	5,15	7,8	4,35

3.2 Mühendislik Problemleri

Yılan algoritmasının performansını değerlendirmek adına dişli sistemi tasarımı [29] ve borulu kolon tasarımı [30] olmak üzere iki adet mühendislik problemi çözülmüştür. Güncel optimizasyon algoritmalarından kelebek optimizasyon algoritması (BOA) [31] ve COOT optimizasyon algoritması [32] ile karşılaştırmalar yapılmıştır. Problemlere ait parametre değerleri Tablo 6'da verilmiştir. Tabloda gösterilen 'N', 'I', 'D' ifadeleri sırasıyla popülasyon sayısı, iterasyon sayısı ve parametre miktarını belirtmektedir.

Tablo 6: Mühendislik problemleri için parametre değerleri.

Problem	N	I	D
Dişli sistem tasarımı	30	100	4
Borulu kolon tasarımı	30	100	2

3.2.1 Dişli Sistem Tasarımı Problemi

Dişli tasarımı problemi, mühendislikte sıklıkla karşılaşılan kısıtsız bir problemdir. Amaç, dişli çarklarının oranını en aza indirmektir. Problemde bulunan dişli sayılarını $\{n_A, n_B, n_C, n_D\} = \{X_1, X_2, X_3, X_4\}$şeklinde ifade ettiğimizde probleme ait amaç fonksiyonu Denklem 21' de verilmiştir.

$$f(X) = \left(\frac{1}{6.931} - \frac{X_3 X_2}{X_1 X_4}\right)^2 \quad (21)$$

Denklemde verilen karar değişkenleri $12 \leq X_i \leq 60$, $i=1,\ldots,4$ aralığında integer değerler olmalıdır. Dişli tasarım problemi Yılan algoritması (SO), Kelebek optimizasyon algoritması (BOA) ve COOT optimizasyon algoritması olmak üzere üç adet optimizasyon algoritması ile çözülmüştür. Karar değişkenleri, ortalama ve standart sapma sonuçlarının karşılaştırılması Tablo 7'de verilmiştir. Tablo incelendiğinde Yılan algoritmasının

sonucunun diğer iki algoritmaya kıyasla daha iyi olduğunu söylemek mümkündür.

3.2.2 *Borulu Kolon Tasarımı Problemi*

Borulu kolon tasarımı problemi, Tablo 6'da görüldüğü üzere iki adet karar değişkenine sahiptir. Problemdeki amaç, toplam maliyeti minimum yapmaktadır. Probleme ait matematiksel model Denklem 22' de verilmiştir. Problem kısıtları Denklem 23-Denklem 28 aralığında verilmiştir.

$$f(X) = 9{,}82X_1X_2 + 2X_1 \quad (22)$$

$$g_1(X) = \frac{P}{\pi X_1 X_2 \sigma_y} - 1 \leq 0 \quad (23)$$

$$g_2(X) = \frac{8PL^3}{\pi^3 E X_1 X_2 (X_1^2 + X_2^2)} - 1 \leq 0 \quad (24)$$

$$g_3(X) = \frac{2}{X_1} - 1 \leq 0 \quad (25)$$

$$g_4(X) = \frac{X_1}{14} - 1 \leq 0 \quad (26)$$

$$g_5(X) = \frac{0.2}{X_2} - 1 \leq 0 \quad (27)$$

$$g_6(X) = \frac{X_1}{0.8} - 1 \leq 0 \quad (28)$$

Verilen denklemlerde P=2500 kgf, σ_y=500 kgf/cm^2, E=0.85x10^6 kgf/cm^2, olarak karar değişkenleri $2<X_1<14$, $0.2<X_2<0.8$ sağlamalıdır. Borulu kolon tasarımı problemine ait sonuç tablosu Tablo 8'de verilmiştir. Tablo incelendiğinde Yılan algoritmasının minimum sonuç verdiği görülmektedir.

Tablo 7: Dişli sistemi tasarımı problemi optimum sonuçlar.

Algoritma	X1	X2	X3	X4	Ortalama	Standart Sapma
SO	42	18	18	56	1.3672E-10	4.1858E-10
BOA	40	12	20	42	2.6526E-04	1.2530E-03
COOT	42	26	12	51	4.2771E-03	1.0837E-02

Tablo 8: Borulu kolon tasarımı problemi optimum sonuçlar.

Algoritma	X1	X2	Ortalama	Standart Sapma
SO	5.452182	0.2916263	2.649E+01	1.8007E-02
BOA	5.444396	0.293256	2.6747E+01	1.3814E-01
COOT	5.452676	0.291615	2.9468E+01	2.7668E+00

Tablo 3 ve Tablo 4' de sunulan CEC 2019 sonuçlarına dayanarak, geliştirilen Yılan algoritmasının diğer güncel ve etkili optimizasyon algoritmalarıyla karşılaştırıldığında belirgin bir başarı sergilediği ortaya çıkmaktadır. Özellikle Yılan algoritması test edilen 10 fonksiyondan 6'sında üstün performans göstermiştir. Bu durum, algoritmanın belirli problem türlerine yönelik etkinliğini ve esnekliğini ortaya koymaktadır. Çalışmanın ilk aşamasında Latin hiperküp tekniğinin kullanılması, çözüm uzayının daha etkili bir şekilde araştırılmasını sağlamıştır. Dolayısıyla, bu yaklaşımın Yılan algoritmasının başarısında önemli bir rol oynadığı söylenebilir. Tablo 7 ve Tablo 8' de Yılan algoritmasıyla çözülmüş iki adet mühendislik problemi çözümü görülmektedir. Bu, algoritmanın uygulama alanlarının çeşitliliğini ve pratikteki etkinliğini göstermektedir. Mühendislik problemlerinin optimizasyonu genellikle karmaşık ve zorlu bir süreçtir. Bu nedenle Yılan algoritmasının bu alanda sağladığı rekabetçi sonuçlar potansiyel kullanıcılar için önemli bir buluş niteliği taşımaktadır.

4. SONUÇ

Bu çalışmada güncel meta-sezgisel optimizasyon algoritmalarından olan Yılan algoritması kullanılarak CEC 2019 ölçü setinin çözümü gerçekleştirilmiştir. Yapılan çalışmada, optimizasyon algoritmalarının performansını ve verimliliğini artırmak adına geleneksel başlatma yöntemi dışında kullanılan yöntemlerden biri olan LHS tekniği algoritmaya entegre edilmiştir. LHS ile elde edilen sonuçların algoritmada kullanılması ile performansta artış sağladığı gösterilmektedir. Şöyle ki iyi bir şekilde ilklendirilmiş popülasyon daha hızlı ve verimli bir şekilde yakınsamayı sağlamaktadır. Elde edilen sonuçlar literatürdeki diğer başarılı meta-sezgisel algoritmalar ile ortalama ve standart sapma bazında karşılaştırılmıştır. Geliştirilen Yılan algoritması 10 problemden 6'sında

(F1, F2, F3, F6, F9, F10) başarı göstermiştir. Ek olarak Yılan algoritması, BOA ve COOT algoritmaları kullanılarak iki adet mühendislik probleminin çözümü yapılmıştır. Sonuçlar karşılaştırıldığında Yılan optimizasyon algoritması BOA ve COOT optimizasyon algoritmasını geride bırakarak başarılı sonuçlar vermiştir. Gelecekteki çalışmalarda Yılan algoritmasının diğer kritik aşamalarına yeni yöntemler eklenerek ve güncellemeler yapılarak başarısı daha da artırabilir. Özellikle, algoritmanın diğer meta-sezgisel yöntemlerle entegrasyonu, daha karmaşık optimizasyon problemlerine yanıt vermesi açısından önemlidir. Çeşitli gerçek zaman problemlerinin çözümünde kullanılabilir.

Yazar Katkıları: Her iki yazar, literatür taraması, araştırma fikrini geliştirme, makale yazımı ve düzenlenmesi süreçlerine ortak katkı sağlamıştır.

Çıkar Çatışması: Bu çalışmanın yazarları, herhangi bir kurum/kuruluş ya da kişi ile çıkar çatışması bulunmadığını beyan etmiştir.

5. KAYNAKLAR

[1] J. H. Holland, "Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence. Ann Arbor, MI: University of Michigan Press, 1975.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of the IEEE International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948.

[3] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous space," Journal of Global Optimization, vol. 11, pp. 341–359, 1997.

[4] Y. Çelik, İ. Yıldız, and A. T. Karadeniz, "A brief review of metaheuristic algorithms improved in the last three years," Avrupa Bilim ve Teknoloji Dergisi, pp. 463–477, 2019.

[5] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive feedback as a search strategy," pp. 91–107, 1991.

[6] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671–680, 1983.

[7] X.-S. Yang, Nature-inspired metaheuristic algorithms*:* Second edition, 2nd ed. Frome, England: Luniver Press, 2010.

[8] F. A. Hashim and A. G. Hussien, "Snake optimizer: A novel meta-heuristic optimization algorithm," Knowledge Based System, vol. 242, no. 108320, p. 108320, 2022.

[9] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, and A. Zamuda, "DISH algorithm solving the CEC 2019 100-digit challenge," in 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 2019, pp. 1-6.

[10] Vigya, S. Raj, C. K. Shiva, B. Vedik, S. Mahapatra, and V. Mukherjee, "A novel chaotic chimp sine cosine algorithm Part-I: For solving optimization problem," Chaos Solitons Fractals, vol. 173, no. 113672, p. 113672, 2023.

[11] Y. Duan and X. Yu, "A collaboration-based hybrid GWO-SCA optimizer for engineering optimization problems," Expert System with Application, vol. 213, no. 119017, p. 119017, 2023.

[12] B. Durmuş, "Kaotik harita temelli ağaç tohum algoritması," Süleyman demirel üniversitesi fen bilimleri enstitüsü dergisi, c. 23, sy. 2, ss. 601–610, 2019.

[13] S. Barua and A. Merabet, "Lévy arithmetic algorithm: An enhanced metaheuristic algorithm and its application to engineering optimization," *Expert Syst. Appl.*, vol. 241, no. 122335, p. 122335, 2024.

[14] D. Pelusi, R. Mascella, L. Tallini, J. Nayak, B. Naik, and Y. Deng, "An Improved Moth-Flame Optimization algorithm with hybrid search phase," *Knowl. Based Syst.*, vol. 191, no. 105277, p. 105277, 2020.

[15] T. Thaher, A. Sheta, M. Awad, and M. Aldasht, "Enhanced variants of crow search algorithm boosted with cooperative based island model for global optimization," *Expert Syst. Appl.*, vol. 238, no. 121712, p. 121712, 2024.

[16] O. J. Agushaka and A. E.-S. Ezugwu, "Influence of initializing krill herd algorithm with low-discrepancy sequences," *IEEE Access*, vol. 8, pp. 210886–210909, 2020.

[17] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," IEEE Transactions Evolutionary Computation, vol. 1, no. 1, pp. 67–82, 1997.

[18] K. V. Price, N. H. Awad, M. Z. Ali, and P. N. Suganthan, "Problem definitions and evaluation criteria for the 100-digit challenge

special session and competition on single objective numerical optimization", Technical report. Singapore: Nanyang Technological University, 2018.

[19] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," Ann. Math. Stat., vol. 11, no. 1, pp. 86–92, 1940.

[20] M. D. Mckay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," Technometrics, vol. 42, no. 1, pp. 55–61, 2000.

[21] F. Wu, J. Zhang, S. Li, D. Lv, and M. Li, "An enhanced differential evolution algorithm with Bernstein operator and refracted oppositional-mutual learning strategy," Entropy (Basel), vol. 24, no. 9, p. 1205, 2022.

[22] P. Civicioglu and E. Besdok, "Bernstain-search differential evolution algorithm for numerical function optimization," Expert System with Applications., vol. 138, no. 112831, p. 112831, 2019.

[23] Y. Xu *et al.*, "An enhanced differential evolution algorithm with a new oppositional-mutual learning strategy," Neurocomputing, vol. 435, pp. 162–175, 2021.

[24] P. Civicioglu, E. Besdok, M. A. Gunen, and U. H. Atasever, "Weighted differential evolution algorithm for numerical function optimization: a comparative study with cuckoo search, artificial bee colony, adaptive differential evolution, and backtracking search optimization algorithms," Neural Computing and Applications, vol. 32, no. 8, pp. 3923–3937, 2020.

[25] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39-43.

[26] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for Differential Evolution," in 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 2013, pp. 71-78.

[27] J. Q. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive", IEEE Transactions Evolutionary Computation, vol. 13, pp. 945–958, 2009.

[28] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," Evol. Comput., vol. 9, no. 2, pp. 159–195, 2001.

[29] E. Sandgren, "Nonlinear integer and discrete programming inmechanical design," in Proceeding of the ASME design technology conference, 1998.

[30] S. S. Rao, Engineering Optimization: Theory and Practice. Hoboken, NJ, USA: Wiley, 2009.

[31] S. Arora, S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," Soft Comput., vol. 23, no. 3, pp. 715–734, 2019.

[32] I. Naruei, F.Keynia, "A new optimization method based on COOT bird natural life model," Expert Syst. Appl. Vol. 183, no. 115352, p.115352, 2021.