Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Araştırma Makalesi

Güncel metasezgisel optimizasyon algoritmalarının CEC2020 test fonksiyonları ile karşılaştırılması

Comparison of current metaheuristic optimization algorithms with CEC2020 test functions

Elif VAROL ALTAY[1*], Osman ALTAY[2]

[1]Kırklareli Üniversitesi, Yazılım Mühendisliği Bölümü, Kırklareli, elifvarol@klu.edu.tr, ORCID: 0000-0001-8087-2754

[2]Manisa Celal Bayar Üniversitesi Yazılım Mühendisliği Bölümü, Manisa, osman.altay@cbu.edu.tr, ORCID: 0000-0003-3989-2432

MAKALE BİLGİLERİ

Makale Geçmişi:

Geliş 8 Ekim 2021
Revizyon 15 Aralık 2021
Kabul 27 Aralık 2021
Online 31 Aralık 2021

Anahtar Kelimeler:

Metasezgisel yöntemler, kalite test fonksiyonları, global optimizasyon

ÖZ

Son yıllarda karmaşık, çok modlu, yüksek boyutlu ve doğrusal olmayan arama ve optimizasyon problemleri için birçok metasezgisel optimizasyon algoritması önerilmiştir. Doğada yer alan canlıların sürü davranışları, bitkilerin davranış biçimleri, insanların sosyal davranışları, matematiksel, fiziksel, kimyasal, biyolojik yasalar ve kurallardan ilham alan çok sayıda metasezgisel optimizasyon algoritması bulunmaktadır. Bu algoritmalar bazı problemlerde başarı ile sonuç üretirken bazı problemlerde yeterince başarılı sonuç üretememektedir. Önerilen bu algoritmaların performansları problemin yapısına göre değişiklik göstermektedir. Araştırmacılar da bundan dolayı her geçen gün yeni yöntemler önermektedir. Bu çalışmada son zamanlarda ortaya çıkan Cıvık Mantar Optimizasyon Algoritması, Balina Optimizasyon Algoritması, Gri Kurt Optimizasyonu, Harris Şahin Optimizasyonu ve Arşimet Optimizasyon Algoritması tanıtılmış ve bu yöntemlerin performansları 10 adet unimodal, multimodal, hibrit ve composition fonksiyonlarını içeren CEC2020 test fonksiyonlarında karşılaştırılmıştır.

ARTICLE INFO

Article history:

Received 8 October 2021
Received in revised form 15 December 2021
Accepted 27 December 2021
Available online 31 December 2021

Keywords:

Metaheuristic methods, benchmark functions, global optimization

ABSTRACT

In recent years, many metaheuristic optimization algorithms have been proposed for complex, multimodal, high-dimensional, and nonlinear search and optimization problems. There are many metaheuristic optimization algorithms inspired by the swarm behavior of living things in nature, the behavior of plants, the social behavior of humans, mathematical, physical, chemical, biological laws, and rules. While these algorithms produce successful results in certain problems, they cannot produce sufficiently successful results in some problems. Therefore, researchers propose new methods every day. In this study, the recently emerged Slime Mould Optimization Algorithm, Whale Optimization Algorithm, Grey Wolf Optimization, Harris Hawk Optimization, and Archimedes Optimization Algorithm are introduced, and the performances of these methods are 10 unimodal, compared to the CEC2020 test functions, which include multimodal, hybrid and composition functions.

Doi: 10.24012/dumf.1051338

* Sorumlu Yazar

Giriş

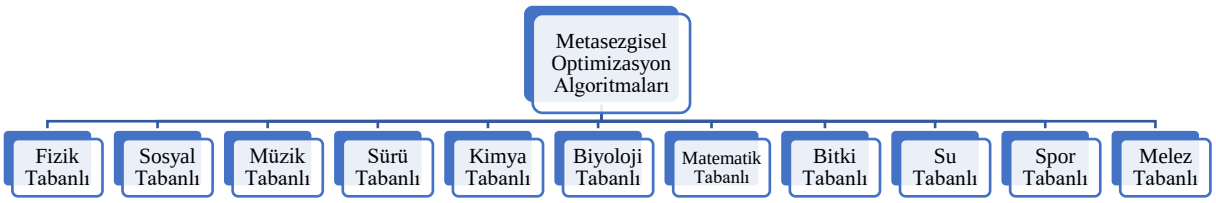
Optimizasyon, belirli bir problemin verilen şartlar altında tüm mevcut çözümleri arasından en uygun çözümü bulma sürecidir [1]. Karmaşık sistemler için matematiksel bir model oluşturmak zordur. Model kurulsa bile çok maliyetli olması ve çok zaman almasından dolayı pek tercih edilmemektedir. Son zamanlarda doğada bulunan olaylardan esinlenen matematiksel araçlar kullanılarak çok sayıda karmaşık doğrusal olmayan optimizasyon problemi çözülmüştür. Bu gibi durumlarda klasik algoritmalar genellikle istenen sonuçları vermeyebilir ve bu nedenle alternatif yöntemler kullanılmalıdır. Metasezgisel optimizasyon yöntemleri; birçok farklı optimizasyon problemini çözmek için yaygın olarak kullanılan, matematiksel modellerin oluşturulamadığı büyük ölçekli arama ve optimizasyon problemleri için kabul edilebilir sürede optimuma yakın çözümler üretmesiyle literatürde iyi bilinen bir küresel optimizasyon yaklaşımıdır. Bu yaklaşım; doğadaki olayları, mekanizmaları veya türlerin sosyal davranışlarını taklit ederek verilen problem için en uygun sonucu bulmayı amaçlamaktadır [2]. Metasezgisel optimizasyon algoritmaları fizik tabanlı, sosyal tabanlı, müzik tabanlı, sürü tabanlı, kimya tabanlı, biyoloji tabanlı, matematik tabanlı, bitki tabanlı, su tabanlı, spor tabanlı ve melez tabanlı olmak üzere 11 farklı kategoride değerlendirilmektedir [3]. Bu kategorilendirme işlemi Şekil 1'de gösterilmektedir. Bunlardan melez kategorisinde aynı kategori ya da farklı kategorideki yöntemlerin üstün özellikleri kullanılarak çözüm arayışı sağlanmaktadır. Bu kategorilerin içerisinde literatürde yüzlerce algoritma bulunmaktadır.

Fizik tabanlı algoritmalar, doğadaki fizik olaylarından esinlenerek ortaya çıkmıştır. En bilinenleri Elektromanyetizma Algoritması [4], Merkezi Kuvvet Optimizasyon Algoritması [5], Yapay Fizik Optimizasyon Algoritması [6], Büyük Çöküş Algoritması [7], Galaksi Tabanlı Algoritmalar [8], Su Döngüsü Algoritması [9], Ücretlendirilmiş Sistem Arama Algoritması [10]'dır. Sosyal tabanlı algoritmalar, insan davranışlarından, insan öğrenme mekanizmasından ve halkın sosyal durumu ile ilişkili pek çok özellikten esinlenerek ortaya çıkmış algoritmalardır. Emperyalist Yarışmacı Algoritması [11], Öğretme Öğrenme Tabanlı Optimizasyon Algoritması [12], Parlamento Optimizasyon Algoritması [13], Sosyal Duygusal Optimizasyon Algoritması [14], Beyin Fırtınası Optimizasyonu [15], Grup Liderleri Optimizasyon Algoritması [16], Sosyal Taklit Algoritma [17], Sosyal Tabanlı Algoritma [18] sosyal tabanlı algoritmalardan bazılarıdır. Bu yöntemlerin performanslarının karşılaştırıldığı çalışmalar da bulunmaktadır [19].

Sürü zekâsı tabanlı optimizasyon algoritmaları kuş, balık, arı gibi sürü halinde yaşayan hayvan topluluklarının hareketlerinden esinlenerek ortaya çıkmıştır. Bu algoritmalardan en bilinenleri Parçacık Sürü Optimizasyonu [20], Karınca Kolonisi Algoritması [21], Yapay Arı Kolonisi [22]'dir. Kimya tabanlı algoritmalar, doğadaki kimyasal olaylardan esinlenerek ortaya çıkmış algoritmalardır. Yapay Atom Algoritması [23], Yapay Kimyasal Tepkime Optimizasyon Algoritması [24] kimya tabanlı algoritmalardır. Biyoloji tabanlı algoritmalar doğadaki canlıların hareketlerinden esinlenerek ortaya çıkmış algoritmalardır. Genetik Algoritma [25], Bakteriyel Besin Arama Optimizasyon Algoritması [26], Diferansiyel Gelişim Algoritması [27] en bilinen biyoloji tabanlı algoritmalardır. Matematik tabanlı algoritmalar, meta-sezgisel ve matematik programlama tekniklerinin birleştirilmesi ile oluşturulmuş algoritmalardır. Bu algoritmaların amacı meta-sezgisel algoritmalardaki yerel arama işlemlerinin matematiksel teknikler aracılığıyla daha verimli hale gelmesini sağlamaktır. Temel Optimizasyon Algoritması [28], Sinüs-Kosinüs Algoritması [29] matematik tabanlı yöntemlerdir.

Bitki tabanlı algoritmalar, bitki istihbaratından esinlenerek ortaya çıkmış algoritmalardır. Fidan Yetiştirme Algoritması [30], Koşucu Kök Algoritması [31], Yol Planlama Algoritması [32], Köklü Ağaç Optimizasyonu [33] bitki tabanlı yöntemlerdir. Su tabanlı algoritmalar, suların akıllı hareketleri esnasında ilham kaynağı olarak önerilmiştir. Su Akışı Algoritması [34], Su Döngüsü Algoritması [35], Su Buharlaştırma Optimizasyonu [36], Simüle Edilmiş Yağmur Damlası Algoritması [37] su tabanlı yöntemlerdir. Spor tabanlı algoritmalar ise Lig Şampiyonası Algoritması [38], Futbol Ligi Optimizasyonu [39], Futbol Oyunu Optimizasyonu [40], Futbol Ligi Yarışması [41], Altın Top Algoritması [42]'dır. Müzik tabanlı algoritmalar Harmoni Arama Algoritması [43], Melodi Arama Algoritması [44], Müzik Besteleme Algoritması [45]'dır. Bu yöntemlerin performanslarının karşılaştırıldığı çalışmalar da bulunmaktadır [46, 47].

Farklı metasezgisel algoritmaların problemin çözümüne yaklaşımları farklı olmasına rağmen hepsinin arama uzayında iki aşaması vardır. Bunlar keşif aşaması ve sömürü aşamasıdır. Keşif aşaması, çözüm uzayını olabildiğince geniş, rastgele ve global olarak arama sürecini ifade ederken, sömürü aşaması algoritmanın keşif aşamasıyla elde edilen alanda daha doğru arama yapabilme yeteneğini ifade etmektedir ve kesinliği artarken rastgeleliği azalmaktadır. Algoritmanın keşif yeteneği baskın olduğunda, çözüm uzayını daha rastgele arayabilir ve hızlı bir şekilde yakınsamak için daha farklılaştırılmış çözüm kümeleri üretebilir. Algoritmanın sömürü yeteneği baskın olduğunda, çözüm kümelerinin kalitesini ve kesinliğini artırmak için daha yerel olarak arama yapmaktadır. Ancak, keşif yeteneği iyileştirilmesi ile sömürü yeteneği ters orantılı olarak düşüş yaşayabilmektedir. Bu iki yeteneğin dengesinin farklı problemlerle aynı olmaması zor bir problemdir. Bu nedenle, tüm optimizasyon problemleri için verimli olan iki aşama arasında uygun bir denge elde etmek zordur.



Şekil 1. Metasezgisel optimizasyon algoritmalarının kategorilendirilmesi

Tüm optimizasyon problemlerini verimli bir şekilde çözmek için kullanılabilecek en iyi optimizasyon algoritması yoktur. Bu no free lunch teoremi ile mantıksal olarak kanıtlanmıştır [48]. Bu teorem çok sayıda araştırmacıyı yeni bir algoritma tasarlamaya motive etmiştir. Ancak son zamanlarda çok fazla sayıda yöntem önerilmiştir. Önerilen bu yöntemlerden hangilerinin hangi alanlarda iyi olduğu ile alakalı çalışmalar pek bulunmamaktadır. Bu çalışma ile son zamanlarda önerilen ve popüler olan beş yöntem CEC2020 test fonksiyonlarından 10 tanesi seçilerek karşılaştırılmıştır. Bu seçilen fonksiyonların türleri farklıdır. Bunun nedeni de optimizasyon algoritmalarının farklı yeteneklerinin birbirleri ile farklı türdeki problemlerde karşılaştırılarak daha iyi karşılaştırma sonuçları elde edebilmektir.

Bu çalışmanın ikinci bölümünde güncel metasezgisel yöntemlerden Cıvık Mantar Optimizasyon (CMO) Algoritması, Balina Optimizasyon Algoritması (BOA), Gri Kurt Optimizasyonu (GKO), Harris Şahin Optimizasyonu (HŞO) ve Arşimet Optimizasyon Algoritması (AOA) çalışma prensipleri açıklanmıştır ve sözde kodları verilmiştir. Üçüncü bölümde CEC2020 test fonksiyonlarından seçilen 10 farklı fonksiyon açıklanmıştır ve CMO, BOA, GKO, HŞO ve AOA yöntemlerinin performansları CEC2020 test fonksiyonları kullanılarak karşılaştırılmıştır. Dördüncü bölümde ise sonuçlar kısmı yer almaktadır.

Güncel Metasezgisel Optimizasyon Algoritmaları

Bu bölümde son zamanlarda ortaya çıkmış ve popüler olan metasezgisel optimizasyon yöntemlerinden CMO, BOA, GKO, HŞO ve AOA'nın esin kaynakları, çalışma prensipleri ayrıntılı bir şekilde açıklanmıştır ve sözde kodları verilmiştir.

Cıvık Mantar Optimizasyon (CMO) Algoritması

Li ve arkadaşları [49] gıdaları bağlamak için en uygun yolu elde etmede cıvık mantarın davranışlarından esinlenerek yeni bir optimizasyon algoritması önermişlerdir. İlk olarak CMO yiyecek ararken, yiyeceğe ulaşmak için havadaki kokuyu kullanmaktadır. Bu davranış matematiksel olarak Denklem 1'deki gibi tanımlanmaktadır.

$$\overrightarrow{X(t+1)} = \begin{cases} \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(\overrightarrow{W} \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r < p \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r \geq p \end{cases} \quad (1)$$

$$a = \text{arctanh}\left(-\left(\frac{t}{\max\_t}\right) + 1\right) \quad (2)$$

$\overrightarrow{vb}$, $[a, -a]$ arasında rastgele üretilmektedir. $\overrightarrow{vc}$, 1'den 0'a lineer bir şekilde azalmaktadır. t mevcut iterasyon sayısını, $\max\_t$ maksimum iterasyon sayısını temsil etmektedir. $\overrightarrow{X_b}$, şimdiye kadar bulunan en yüksek koku konsantrasyonuna sahip konumu içeren vektörü, $\overrightarrow{X(t+1)}$ mevcut cıvık mantarın aldığı bir sonraki pozisyonu, $\overrightarrow{X(t)}$ cıvık mantarın mevcut konumunu $\overrightarrow{X_A(t)}$ ve $\overrightarrow{X_B(t)}$ popülasyondan rastgele seçilen iki bireyin konumunu içeren iki vektörü temsil etmektedir. r 0 ile 1 arasında rastgele bir sayıdır. $\overrightarrow{W}$, cıvık mantarın ağırlığını tanımlamaktadır ve Denklem (3)'teki gibi hesaplanmaktadır.

$$\overrightarrow{W(SmellIndex(i))} = \begin{cases} 1 + r \cdot log\left(\frac{bF - S(i)}{bF - wF} + 1\right), condition \\ 1 - r \cdot log\left(\frac{bF - S(i)}{bF - wF} + 1\right), \ others \end{cases} \quad (3)$$

$$SmellIndex = sort(S) \quad (4)$$

bF ve wF sırasıyla mevcut iterasyon içindeki en iyi ve en kötü uygunluk değerini temsil ederken, $SmellIndex$ sıralanan uygunluk değerlerinin sırasını ifade etmektedir. $S(i)$, popülasyonun ilk yarısının sıralarını göstermektedir. p parametresi Denklem (5)'teki gibi modellenmektedir.

$$p = \tanh|S(i) - DF| \quad (5)$$

$i \in 1, 2, 3, \ldots, n$ $S(i)$ $\vec{X}$'in uygunluğunu temsil ederken DF tüm iterasyonlarda elde edilen en iyi uygunluk değerini temsil etmektedir.

İkinci aşama olan yiyecekleri sarma aşamasında, arama sırasında cıvık mantarının venöz doku yapısının kasılma modunu matematiksel olarak simüle etmektedir. Kap ile temas eden gıdanın konsantrasyonu biyo-osilatör tarafından üretilen dalga boyu ile doğru orantılıdır. Böylece dalga boyu ne kadar yüksek olursa, sitoplazma o kadar hızlı ve kapta o kadar kalın olmaktadır. Ağırlık, gıda konsantrasyonuna göre değişmektedir. Ağırlık düşük olduğunda, cıvık mantar diğer alanları keşfetme eğiliminde olmaktadır. Cıvık mantarın yerini güncellemek için matematiksel formül ve denklem Denklem (6)'da verilmiştir.

$$\overrightarrow{X^*} = \begin{cases} rand \cdot (UB - LB) + LB, rand < z \\ \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(W \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r < p \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r \geq p \end{cases} \quad (6)$$

UB ve LB, problemin arama uzayının üst ve alt sınırlarıdır. z, CMO'nun başka bir besin kaynağı arayacağını veya mevcut en iyi kaynak çevresinde arama yapacağını belirlemek için kullanılan bir olasılıktır. W, $\overrightarrow{vb}$ ve $\overrightarrow{vc}$ venöz genişlik değişimini taklit etmek için kullanılmaktadır. CMO'nun sözde kodları Algoritma 1'de gösterilmiştir.

Algoritma 1. CMO'nun sözde kodu

Cıvık mantarın başlangıç popülasyonunun ayarlanması
$X_i(i = 1, 2, \dots, n)$
while $(t < maksimum\ iterasyon\ sayısı)$
  Her bir arama ajanının uygunluk değerini hesapla
  $bestFitness$, X_b parametrelerini güncelle
  W (bireyin konumu) değerini hesapla
  for her bir arama bölümü
    p,vb, vc değerlerini güncelle
    Cıvık mantarın konumunu Denklem (6)'ya göre güncelle
  end for
  $t = t + 1$
end while
return $bestFitness$, X_b

Balina Optimizasyon Algoritması (BOA)

Mirjalili ve Lewis [50] kambur balinaların sosyal davranışlarını taklit eden ve doğadan ilham alan yeni bir metasezgisel yöntem önermişlerdir. Bu balinalar hareket ederek avını sarmal bir şekilde çevrelemektedir ve saldırırken küçülen bir daire içinde avına doğru hareket etmektedir. Bu davranışa kabarcık yeni yiyecek arama denir. Bu avlanma mekanizması BOA içinde, bir sarmal model ile %50 olasılıkla küçülen, çevreleyen bir av arasında bir değiş tokuş yaparak optimizasyon süreci içinde yeni bir çözüm üreterek taklit edilmektedir. Çemberleme mekanizması aşağıdaki denklemlerle hesaplanmaktadır.

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A}.\vec{D} \quad (7)$$

$$\vec{D} = \left|\vec{C}.\overrightarrow{X^*}(t) - \vec{\mathrm{X}}(t)\right| \quad (8)$$

$$\vec{A} = 2.\vec{a}.\vec{r} - \vec{a} \quad (9)$$

$$\vec{a} = 2 - 2\frac{t}{\max\_t} \quad (10)$$

$$\vec{C} = 2.\vec{r} \quad (11)$$

t mevcut iterasyon sayısını, max $\_t$ maksimum iterasyon sayısını temsil etmektedir. $\vec{A}$ ve $\vec{C}$ katsayı vektörleridir, $\overrightarrow{X^*}$ şimdiye kadar elde edilen en iyi çözümün konum vektörüdür. $\vec{\mathrm{X}}$ konum vektörüdür, $\vec{r}$ 0 ile 1 arasında üretilen rastgele bir sayı, $\vec{a}$ 2'den 0'a doğru lineer bir şekilde azalan bir sayıdır ve mesafe kontrol parametresidir. Kambur balinaların sarmal şeklindeki hareketini taklit etmek için balina ve avın konumu arasında Denklem (12)'deki gibi bir spiral denklem oluşturulmaktadır.

$$\vec{X}(t+1) = \overrightarrow{D'}.e^{bl}.\cos(2\pi l) + \overrightarrow{X^*}(t) \quad (12)$$

$$\overrightarrow{D'} = \left|\overrightarrow{X^*}(t) - \vec{\mathrm{X}}(t)\right| \quad (13)$$

$\overrightarrow{D'}$ i. balinanın ava olan mesafesini gösterir, b logaritmik spiralin şeklini tanımlamak için bir sabittir, l [-1, 1] aralığında rastgele bir sayıdır. Şimdiye kadarki en iyi çözüm, yerel bir minimum problem olabilir, bu nedenle optimizasyon süreci içinde tamamen buna odaklanmak, bahsedilen herhangi bir faydalı arama sürecini boşa harcayabilir. Bu nedenle, başka bir konum için balina araması, mevcut balinayı daha iyi bir çözüm bulmaya doğru hareket ettirmek için popülasyondan rastgele bir balina seçerek avı arama alanı içinde bulunabilir. Spesifik olarak eğer $A < 1$ ise mevcut balina, popülasyondan rastgele seçilen bir balinaya göre yönlendirilmektedir. Bu keşif aşamasının matematiksel modeli aşağıdaki denklemler gibidir.

$$\vec{D} = \left|\vec{C}.\overrightarrow{X_{rand}} - \vec{\mathrm{X}}\right| \quad (14)$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A}.\vec{D} \quad (15)$$

Burada $\overrightarrow{X_{rand}}$ popülasyondan rastgele seçilen bir konum vektörüdür. BOA'nın sözde kodları Algoritma 2'de gösterilmiştir.

Algoritma 2. BOA'nın sözde kodları

Balinaların başlangıç popülasyonunun ayarlanması
$X_i(i = 1, 2, \dots, n)$
Her bir arama ajanının uygunluk değerini hesapla
X^*= en iyi arama ajanı
while $(t < maksimum\ iterasyon\ sayısı)$
  for her bir arama ajanı
   a, A ve C, l ve p parametrelerini güncelle
    if $(p < 0.5)$
      if $(|A| < 1)$
        Denklem (8)'e göre arama ajanının konumunu güncelle
      else if $(|A| \geq 1)$
        Rastgele bir arama ajanı seç X_{rand}
        Denklem (15)'e göre arama ajanının konumunu güncelle
      end if
    else if $(p \geq 0.5)$
      Denklem (12)'ye göre arama ajanının konumunu güncelle
    end if
  end for
  Her nesneyi değerlendir ve en iyi uygunluk değerine sahip olanı seç
  X^* değerini güncelle
  $t = t + 1$
end while
return X^*

Gri Kurt Optimizasyonu (GKO)

Mirjalili ve arkadaşları [51] doğadaki gri kurtların avlanma davranışlarından ve sosyal liderliğinden ilham alarak yeni bir metasezgisel optimizasyon algoritması önermişlerdir. GKO'da diğer metasezgisel algoritmalara benzer şekilde bir dizi rastgele aday çözüm üreterek optimizasyon sürecini başlatmaktadır. Her iterasyonda en iyi üç aday çözüm, arama uzayının gelecek vaat eden bölgelerine öncülük eden alfa, beta ve delta kurt olarak kabul edilmektedir. Geri kalan gri kurtlar omega olarak kabul edilmektedir. Temel olarak üç adımdan oluşmaktadır. Bunlar kuşatma, avlanma ve avına saldırmadır. Daha iyi çözümler bulma umuduyla omega kurtlar; alfa, beta ve deltayı çevrelemeleri gerekmektedir. Omega kurtların matematiksel formülasyonu aşağıdaki gibidir:

$$D = \left|C \times X_p(t) - X(t)\right| \quad (16)$$

$$X(t+1) = X_p(t) - A \times D \quad (17)$$

Burada X_p avın konumunu, X gri kurdun konum vektörünü, t ise mevcut iterasyon sayısını temsil etmektedir. C ve A katsayı vektörleridir. Denklem (18) ve Denklem (19)'daki gibi hesaplanmaktadır:

$$A = 2 \times A \times r_1 - a(t) \quad (18)$$

$$C = 2 \times r_2 \quad (19)$$

Burada r_1 ve r_2 0 ile 1 arasında rastgele vektörlerdir ve a vektörü iterasyonlar boyunca 2'den 0'a doğru lineer bir şekilde Denklem (20)'deki gibi azalmaktadır.

$$a(t) = 2 - (2 \times t)/MaxIter \quad (20)$$

Kurtların avlanma davranışlarını matematiksel olarak modellemek için alfa, beta ve deltanın konumu hakkında daha iyi bir bilgiye sahip olduğu varsayılmaktadır. Bu nedenle her bir omega kurtu alfa, beta ve deltanın konumunu dikkate alarak takip etmek zorundadır. Denklem (21)'de avlanma davranışı açıklanmıştır:

$$\begin{matrix} D_\alpha = |C_1 \times X_\alpha - X(t)| \\ D_\beta = |C_2 \times X_\beta - X(t)| \\ D_\delta = |C_3 \times X_\delta - X(t)| \end{matrix} \quad (21)$$

Burada C_1, C_2 ve C_3 Denklem (19) ile hesaplanır. alfa, beta ve deltanın konumu Denklem 22'deki gibi hesaplanmaktadır.

$$\begin{matrix} X_{i1}(t) = X_\alpha(t) - A_{i1} \times D_\alpha(t) \\ X_{i2}(t) = X_\beta(t) - A_{i2} \times D_\beta(t) \\ X_{i3}(t) = X_\delta(t) - A_{i3} \times D_\delta(t) \end{matrix} \quad (22)$$

$$X(t+1) = \frac{X_{i1}(t) + X_{i2}(t) + X_{i3}(t)}{3} \quad (23)$$

Mirjalili ve arkadaşları A ve C parametrelerinin GKO algoritmasını arama uzayını keşfetmeye ve kullanmaya mecbur bıraktıklarını savunmaktadırlar. İterasyonların yarısı keşfetme aşaması için geri kalanı ise sömürü aşaması için ayrılmıştır. C parametresi optimizasyon süresince yerel optimum durgunluğu çözmek için rastgele seçilmektedir. GKO'nun sözde kodları Algoritma 3'te gösterilmiştir.

Algoritma 3. GKO'nun sözde kodları

Gri kurtların başlangıç popülasyonunun ayarlanması
$X_i (i = 1, 2, \dots, n)$
a, A ve C parametrelerine ilk değerlerin verilmesi
Her bir arama ajanının uygunluk değerini hesapla
X_α=en iyi arama ajanı
X_β=en iyi ikinci arama ajanı
X_δ=en iyi üçüncü arama ajanı
while (t<maksimum iterasyon sayısı)
  for her bir arama ajanı
    Mevcut arama ajanının konumunu güncelle
  end for
  a, A ve C parametrelerini güncelle
  Tüm arama ajanlarının uygunluğunu hesapla
  X_α, X_β, X_δ değerlerini güncelle
  $t = t + 1$
end while
return X_α

Harris Şahin Optimizasyonu (HŞO)

Heidari ve arkadaşları [52] Harris şahinlerinin davranışlarından ve avcılık modelinden esinlenilen popülasyon tabanlı metasezgisel bir optimizasyon algoritması önermişlerdir. HŞO optimal çözümleri bulmak için karmaşık arama uzaylarını keşfedebilen stokastik bir algoritmadır. HŞO'nun temel adımları, çeşitli enerji durumlarına göre elde edilebilmektedir. Keşif aşaması, Harris şahininin avı doğru bir şekilde izleyemediği durumlardaki mekanizmayı simüle eder. Böyle bir durumda şahinler yeni avın izini sürmek ve bulmak için ara verirler. HŞO yönteminde aday çözümler şahinlerdir ve her adımda en iyi çözüm avdır (X_{rabbit}). Şahinler rastgele farklı konumlara yerleşirler ve Denklem (24)'te verilen q olasılığına göre seçilen iki operatör kullanarak avlarını beklerler. $q < 0.5$ olduğunda şahinlerin diğer popülasyon üyelerinin ve avın (örneğin tavşan) bulunduğu yere yerleştiğini göstermektedir. $q \geq 0.5$ olduğu durumlarda şahinler popülasyon aralığında rastgele konumlardadır. Keşif aşaması Denklem (24)'teki gibidir.

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 |X_{rand}(t) - 2r_2 X(t)|, & q \geq 0.5 \\ \left(X_{rabbit}(t) - X_m(t)\right) - r_3 (LB + r_4 (UB - LB)), & q < 0.5 \end{cases} \quad (24)$$

$X(t+1)$ t. iterasyondaki şahinlerin konum vektörüdür. X_{rabbit}, avın en iyi konumunu, $X(t)$ şahinlerin mevcut konum vektörünü, r_1, r_2, r_3, r_4 ve q değerleri 0 ile 1 arasında rastgele sayılardır ve her iterasyonda güncellenmektedir. LB ve UB değişkenlerin üst ve alt sınırlarını, $X_{rand}(t)$ mevcut popülasyondan rastgele seçilmiş bir şahin, X_m mevcut şahin popülasyonunun ortalama konumunu temsil etmektedir ve Denklem (25)'teki gibi hesaplanmaktadır.

$$X_m(t) = \frac{1}{N} \sum_{i=1}^{N} X_i(t) \quad (25)$$

Burada $X_i(t)$ t. iterasyondaki her şahinin konumunu, N toplam şahin sayısını temsil etmektedir. Keşiften sömürüye iyi bir geçiş gereklidir, burada avın kaçış davranışı sırasında önemli ölçüde azalan enerji faktörüne dayanan farklı simüle edilmiş sömürü davranışlar arasında bir geçiş olması beklenmektedir. Avın enerjisi Denklem (26)'daki gibi modellenmiştir.

$$E = 2E_0 \left(1 - \frac{t}{\max\_t}\right) \quad (26)$$

Burada E avın kaçan enerjisini, E_0 enerjinin ilk durumunu, t mevcut iterasyon sayısını, max $\_t$ maksimum iterasyon sayısını temsil etmektedir. HŞO'da E_0 her iterasyonda (-1,1) aralığı arasında rastgele değişmektedir. E_0 değeri 0'dan -1'e düştüğü zaman tavşan fiziksel olarak işaretlenirken, E_0 değeri 0'dan 1'e yükseldiği zaman tavşan (av) güçleniyor demektir. Dinamik kaçış enerjisi E iterasyonlar sırasında azalan bir eğilime sahiptir. Kaçan enerji $|E| \geq 1$olduğunda şahinler bir tavşanın yerini keşfetmek için farklı bölgeleri aramaktadır dolayısıyla HŞO keşif aşamasını gerçekleştirmektedir ve $|E| < 1$ olduğu zaman algoritma sömürü aşaması boyunca çözümlerin komşuluğundan yararlanmaya çalışmaktadır. Kısacası $|E| \geq 1$ olduğu zaman keşif aşaması $|E| < 1$ olduğu zaman ise sömürü aşaması gerçekleşmektedir. p eşit koşullar altında değerlendirildiğinde $p \geq 0.5$ olduğu durumlar başarılı $p <$

0.5 olduğu durum başarısız olduğu anlamına gelmektedir. Ayrıca avcının (tavşan) enerjisine bağlı olarak şahinler $|E| \geq 0.5$) olduğunda yumuşak $|E| < 0.5$) olduğunda sert bir kuşatma gerçekleştirecektir. Yumuşak kuşatma denklemleri aşağıdaki gibi formüle edilmiştir.

$$X(t+1) = \Delta X(t) - E|J.X_{rabbit}(t) - X(t)| \quad (27)$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \quad (28)$$

$$J = 2(1 - rand) \quad (29)$$

$\Delta X(t)$ şahin ve tavşanın pozisyonları arasındaki farktır. Tavşanın rastgele atlama gücü J, rastgele bir sayı kullanılarak çizilmektedir. Sert kuşatma denklemi Denklem (30)'daki gibidir.

$$X(t+1) = X_{rabbit}(t) - E|\Delta X(t)| \quad (30)$$

$p < 0.5$ ve $|E| \geq 0.5$) olduğu durumlarda tavşan başarıyla hücum edebildiğinden aşamalı hızlı dalışlarla yumuşak kuşatma yapar. Şahin mümkün olan en iyi dalışı seçmektedir. Levy flight avın birbirini takip etmesinde kullanılmaktadır. Dalışın iyi olup olmadığına karar vermek için de şahinin bir sonraki hareketi aşağıdaki denklemler kullanılarak tahmin edilmektedir.

$$Y = X_{rabbit}(t) - E|J.X_{rabbit}(t) - X(t)| \quad (31)$$

Önceki dalış faydalı değilse şahin Levy flight modelini kullanarak dalış yapmaktadır. Kullandığı model Denklem (32)'deki gibidir:

$$Z = Y + S \times LF(D) \quad (32)$$

Burada D problemin boyutudur ve S $1 \times D$ boyutunda rastgele bir vektördür. Levy flight fonksiyonu (LF) Denklem (33) kullanılarak hesaplanan yük uçuş fonksiyonudur.

$$LF(x) = 0.001 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \quad , \quad \sigma = \left(\frac{r(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{r\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \quad (33)$$

Burada u ve v o ile 1 arasında rastgele değerlerdir β 1.5 olan bir sabittir. Bu nedenle şahinlerin pozisyonlarını güncellemek için son strateji yumuşak kuşatma aşaması Denklem (34) ile gerçekleşmektedir.

$$X(t+1) = \begin{cases} Y, \ if \ F(Y) < F(X(t)) \\ Z, \ if \ F(Z) < F(X(t)) \end{cases} \quad (34)$$

$p < 0.5$ ve $|E| < 0.5$ olduğu durumlarda tavşanın kaçmak için yeterli enerji yoktur ve avı yakalamak ve öldürmek için sürpriz saldırıdan önce sert bir kuşatma yapmaktadır. Burada yumuşak kuşatma gibi Denklem (32) ve Denklem (34) aynıdır. Y değerinin hesaplanması farklıdır. Y değerinin denklemi Denklem (35)'teki gibi hesaplanmaktadır.

$$Y = X_{rabbit}(t) - E|J.X_{rabbit}(t) - X_m(t)| \quad (35)$$

HŞO'nun sözde kodları Algoritma 4'te gösterilmiştir.

Algoritma 4. HŞO'nun sözde kodu

Şahinlerin başlangıç popülasyonunun ayarlanması
$X_i(i = 1, 2, \dots, n)$
while $(t < maksimum\ iterasyon\ sayısı)$
  Her bir arama ajanının uygunluk değerini hesapla
  X_{rabbit}'i avın en iyi konum olarak ayarla
  for her bir arama ajanı
  E_o ve j parametrelerini güncelle
  Denklem (26)'da kullanılan E değerini güncelle
    if $(|E| \geq 1)$
      Denklem (24)'ü kullanarak konum vektörünü güncelle
    end if
    if $(|E| < 1)$
      if $(p \geq 0.5$ ve $|E| \geq 0.5)$
        Denklem (27)'yi kullanarak konum vektörünü güncelle
      else if $(p \geq 0.5$ ve $|E| < 0.5)$
        Denklem (30)'u kullanarak konum vektörünü güncelle
      else if $(p < 0.5$ ve $|E| \geq 0.5)$
        Denklem (34)'ü kullanarak konum vektörünü güncelle
      else if $(p < 0.5$ ve $|E| < 0.5)$
        Denklem (34)'ü kullanarak konum vektörünü güncelle
    end if
  end if
  $t = t + 1$
end while
return X_{rabbit}

Arşimet Optimizasyon Algoritması (AOA)

Hashim ve arkadaşları [53] fizik kanunu olan Arşimet Prensibinden ilham alarak yeni bir metasezgisel algoritma önermişlerdir. Kısmen veya suya batırılmış bir nesneye yukarı doğru uygulanan kaldırma kuvveti ilkesinden yola çıkarak oluşturulmuştur. Diğer popülasyon tabanlı metasezgisel algoritmalar gibi, AOA da rastgele hacimler, yoğunluklar ve ivmelerle nesnelerin ilk popülasyonu (aday çözümler) ile arama sürecini başlatmaktadır. Bu aşamada, her nesne aynı zamanda sıvı içindeki rastgele konumu ile başlatılır. Başlangıç popülasyonunun uygunluğunu değerlendirdikten sonra, AOA, sonlandırma koşulu sağlanana kadar çalışır. Her iterasyonda AOA, her nesnenin yoğunluğunu ve hacmini güncellemektedir. Nesnenin ivmesi, herhangi bir komşu nesneyle çarpışmasının durumuna göre güncellenmektedir. Güncellenen yoğunluk, hacim, ivme bir nesnenin yeni konumunu belirlemektedir. Nesnelerin yoğunluk ve hacimlerinin güncelleme denklemi Denklem (36)'da verilmiştir.

$$\begin{aligned} den_i^{t+1} &= den_i^t + rand \times (den_{best} - den_i^t) \\ vol_i^{t+1} &= vol_i^t + rand \times (vol_{best} - vol_i^t) \end{aligned} \quad (36)$$

Burada den_{best} ve vol_{best}, şimdiye kadar bulunan en iyi nesneye ilişkin yoğunluk ve hacim, $rand$ eşit olarak dağıtılmış rastgele sayıdır. Nesneler arasında çarpışma

meydana gelmektedir ve bir süre sonra nesneler denge durumuna ulaşmaya çalışmaktadır. Aramayı keşiften sömürüye dönüştüren transfer operatörü TF'nin yardımıyla AOA'ya entegre edilmektedir. TF'nin formülü Denklem (37)'de verilmiştir.

$$TF = exp\left(\frac{t-\max\_t}{\max\_t}\right) \quad (37)$$

Burada TF 1'e ulaşana kadar zamanla kademeli olarak artmaktadır. t iterasyon sayısını $\max\_t$ maksimum iterasyon sayını temsil etmektedir. Benzer şekilde yoğunluğu azaltıcı faktör olan d, küreselden yerel aramada AOA'ya yardımcı olmaktadır. Formülü Denklem (38)'deki gibidir.

$$d^{t+1} = exp\left(\frac{\max\_t-t}{\max\_t}\right) - \left(\frac{t}{\max\_t}\right) \quad (38)$$

d^{t+1} önceden belirlenmiş umut verici bölgede yakınsama yeteneği vererek zamanla azalmaktadır. Bu değişken AOA'da keşif ve sömürü arasındaki dengeyi sağlamaya çalışmaktadır. $TF \leq 0.5$ ise, nesneler arasında çarpışma meydana gelmektedir ve rastgele bir malzeme (m) seçmek gerekmektedir ve Denklem (39) kullanılarak nesnenin ivmesi güncellenmektedir.

$$acc_i^{t+1} = \frac{den_m + vol_m \times acc_m}{den_i^{t+1} \times vol_i^{t+1}} \quad (39)$$

Burada den_i, vol_i, acc_i i. nesnenin yoğunluğu, hacmi ve ivmesidir. den_m, vol_m, acc_m rastgele malzemenin yoğunluğu, hacmi ve ivmesidir. $TF > 0.5$ ise, nesneler arasında çarpışma yoktur ve Denklem (40) kullanılarak nesnenin ivmesi güncellenmektedir.

$$acc_i^{t+1} = \frac{den_{best} + vol_{best} \times acc_{best}}{den_i^{t+1} \times vol_i^{t+1}} \quad (40)$$

acc_{best} en iyi nesnenin ivmesidir. Denklem (41)'i kullanarak değişim yüzdesini hesaplamak için ivmeyi normalleştirmek gerekmektedir.

$$acc_{i-norm}^{t+1} = u \times \frac{acc_i^{t+1} - \min(acc)}{\max(acc) - \min(acc)} + 1 \quad (41)$$

Burada u ve l normalizasyon aralığıdır ve 0.9 ve 0.1'e ayarlanmıştır. acc_{i-norm}^{t+1} her nesnenin değiştireceği adım yüzdesini belirlemektedir. i nesnesi küresel optimumdan uzaksa hızlanma değeri yüksek olacaktır. Yani nesne keşif aşamasında olacaktır aksi takdirde sömürü aşamasında olacaktır. Böylece aramanın keşif aşamasından sömürü aşamasına nasıl dönüştüğü görülmektedir. Normal durumda ivme vektörü büyük bir değerle başlamaktadır ve zamanla azalmaktadır. Bu sayede nesnelerin küresel en iyi çözüme doğru ilerlemesine ve aynı zamanda yerel çözümlerden uzaklaşmasına yardımcı olmaktadır. $TF \leq 0.5$ (keşif aşaması) ise, i. nesnenin bir sonraki iterasyondaki konumunu güncellediği denklem Denklem (42)'deki gibidir.

$$X_i^{t+1} = X_i^t + C_1 \times rand \times acc_{i-norm}^{t+1} \times d \times (X_{rand} - X_i^t) \quad (42)$$

Burada C_1 sabiti 2 değerine eşittir. $TF > 0.5$ (sömürü aşaması) ise, i. nesnenin bir sonraki iterasyondaki konumunu güncellediği denklem Denklem (43)'teki gibidir.

$$X_i^{t+1} = X_{best}^t + F \times C_2 \times rand \times acc_{i-norm}^{t+1} \times d \times (T \times X_{best} - X_i^t) \quad (43)$$

Burada C_2 sabiti 6 değerine eşittir. T zamanla artmaktadır ve transfer operatörü ile doğru orantılıdır ve $T = C_3 \times TF$'dir. T, $[C_3 \times 0.3, 1]$ aralığında zamanla artmaktadır ve başlangıçta en iyi konumdan belirli bir yüzde almaktadır. Arama ilerledikçe, en iyi konum ile mevcut konum arasındaki farkı azaltmak için bu yüzde kademeli olarak artmaktadır. Bu, keşif ve sömürü arasında uygun bir dengenin oluşumunu sağlamaktadır. F, Denklem (44) kullanılarak hareket yönünü değiştiren işarettir.

$$F = \begin{cases} +1, & if\ p \leq 0.5 \\ -1, & if\ p > 0.5 \end{cases} \quad (44)$$

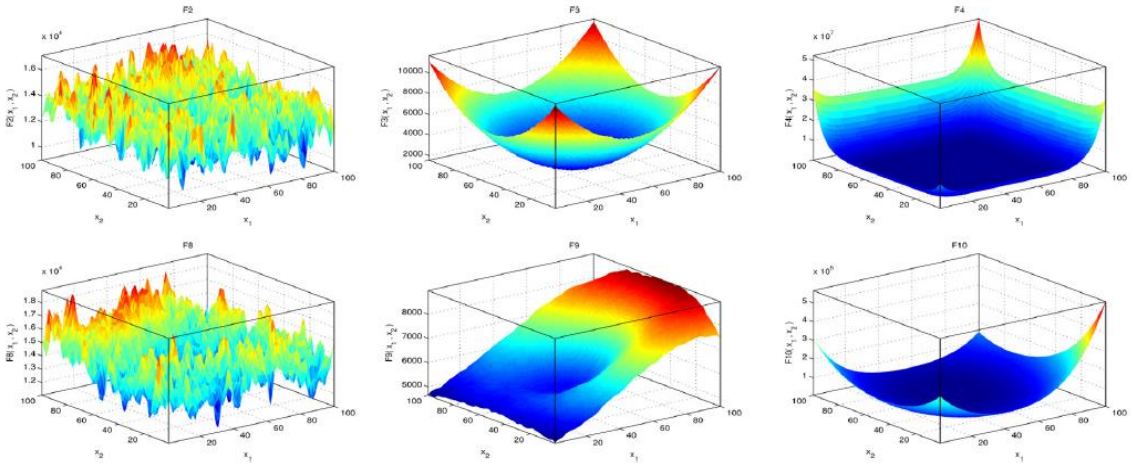
Burada $p = 2 \times rand - C_4$ ile hesaplanmaktadır. AOA'nın sözde kodu Algoritma 5'te verilmiştir.

Algoritma 5. AOA'nın sözde kodu

Başlangıç popülasyonunun ayarlanması $X_i (i = 1, 2, \dots, n)$
while $(t < maksimum\ iterasyon\ sayısı)$
  Her bir arama ajanının uygunluk değerini hesapla
  for her bir arama ajanı
    Denklem (36)'yı kullanarak her nesnenin yoğunluğunu ve hacmini güncelle
    Denklem (37) ve Denklem (38)'i kullanarak transfer ve yoğunluk azaltıcı faktörleri güncelle
    if $(TF \leq 0.5)$
      Denklem (39)'u kullanarak ivmeyi güncelle ve Denklem (41)'i kullanarak ivmeyi normalleştir
      Denklem (42)'yi kullanarak konumu güncelle
    else
      Denklem (40)'ı kullanarak ivmeyi güncelle ve Denklem (41)'i kullanarak ivmeyi normalleştir
      Denklem (44)'ü kullanarak yön bayrağını güncelle
      Denklem (43)'ü kullanarak konumu güncelle
    end if
  end for
  $t = t + 1$
end while
return en iyi uygunluk değeri

Deneysel Sonuçlar

CMO, BOA, GKO, HŞO ve AOA yöntemlerinin performansını analiz etmek için IEEE Congress on Evolutionary Computation (CEC) test fonksiyonları seçilmiştir [54]. CEC 2020 test fonksiyonları unimodal, multimodal, hibrit ve composition fonksiyonlarını içeren 10 test fonksiyonunu içermektedir. Unimodal fonksiyonlar, algoritmanın yakınsama performansının ölçülmesi için kullanılırken; multimodal fonksiyonlar algoritmanın erken yakınsama problemlerinin ve yerele takılma sorunun olup olmadığının değerlendirilmesinde kullanılmaktadır [55]. Hibrit ve composition fonksiyonlar ise çok sayıda yerel optimuma sahip olan yerel optimumdan kaçınma yeteneğini ve keşif ve sömürü arasındaki dengeyi değerlendirmek için kullanılmaktadır.



Şekil 2. Bazı CEC2020 test fonksiyonlarının iki boyutlu görünümü

Tablo 1. CEC2020 test fonksiyonları

No	Fonksiyon adı	Fi*
	Unimodal fonksiyon	
F1	Shifted and Rotated Bent Cigar fonksiyonu	100
	Multimodal fonksiyon	
F2	Shifted and Rotated Schwefel's fonksiyonu	1100
F3	Shifted and Rotated Lunacek bi-Rastrigin fonksiyonu	700
F4	Expanded Rosenbrock's plus Griewangk's fonksiyonu	1900
	Hibrit fonksiyonlar	
F5	Hibrit fonksiyon 1 (N = 3)	1700
F6	Hibrit fonksiyon 2 (N = 4)	1600
F7	Hibrit fonksiyon 3 (N = 5)	2100
	Composition fonksiyonlar	
F8	Composition fonksiyon 1 (N = 3)	2200
F9	Composition fonksiyon 2 (N = 4)	2400
F10	Composition fonksiyon 3 (N = 5)	2500

Tablo 1, CEC2020 test fonksiyonlarının özelliklerini göstermektedir. Fi* fonksiyonun optimal global değerini ifade etmektedir. Şekil 2, her bir problemin doğasının anlaşılmasını kolaylaştırmak için CEC2020 test fonksiyonlarının iki boyutlu bir görselleştirmesini sunmaktadır.

Yapılan tüm deneyler Manisa Celal Bayar Üniversitesi tarafından lisanslı Matlab 2021a platformunda Windows 10 işletim sistemli 32 GB RAM ve CPU of Intel (R) core i9-10900 k (3.7 GHz) işlemcili bilgisayarda yapılmıştır. Bu çalışmada analiz edilen algoritmaların parametreleri Tablo 2'de sunulmuştur. Algoritmaların kontrol parametrelerinin çoğu literatürde kullanılan varsayılan değerlerdir. Eşit koşullar altında adil bir değerlendirme yapabilmek adına değerlendirme sayısı 1000, popülasyon sayısı 30 olarak seçilmiştir. Algoritmalar tüm deneylerde 20 kez çalıştırılmıştır ve minimum, maksimum, ortalama ve standart sapma değerlerinin sonuçları karşılaştırmalı bir şekilde Tablo 3'te sunulmuştur. Algoritmaların CEC2020 test fonksiyonları üzerindeki yakınsama performansı da algoritmaların en iyi değerlerine göre Şekil 3'te verilmiştir.

Tablo 2. Karşılaştırma yapılan algoritmaların parametreleri

Algoritma	Parametreler
CMO	$z = 0.03$
BOA	$A1 = [2, 0]; a2 = [-2, -1]; b = 1$
GKO	$a = [2, 0]$
HŞO	$EO \in [-1, 1];\ \beta = 1.5$
AOA	$C1 = 2; C2 = 6;\ u = 0.9;\ l = 0.1$

Tablo 3 incelendiğinde tüm test fonksiyonlarını değerlendirmeye alırsak; ortalama değer bakımından 10 test fonksiyonunun 7'sinde AOA birinci sırada yer alırken 3 en iyi ortalama sonuçla AOA'yı CMO takip etmektedir. BOA ortalama değerde 10 test fonksiyonunun hiçbirinde en iyi sonuca ulaşamamıştır. Minimum değer bakımından incelediğimiz zaman AOA ve CMO 10 test fonksiyonunun 5'inde en iyi sonuca ulaşarak üstünlüklerini göstermişlerdir. Bu yöntemleri GKO ve HŞO 3 tanesinde en iyi sonucu vererek takip etmektedir. BOA ise sadece 1 tanesinde en iyi sonuca ulaşabilmiştir.

Unimodal fonksiyonlar algoritmaların sömürü aşamasındaki yeteneklerini araştırmak için kullanılmıştır. Unimodal fonksiyonlarda AOA tüm değerlendirme kriterlerinde en iyi sonuca ulaşmıştır. Buna dayanarak unimodal fonksiyonlarda AOA'nın diğer algoritmalara göre daha üstün olduğunu söyleyebiliriz.

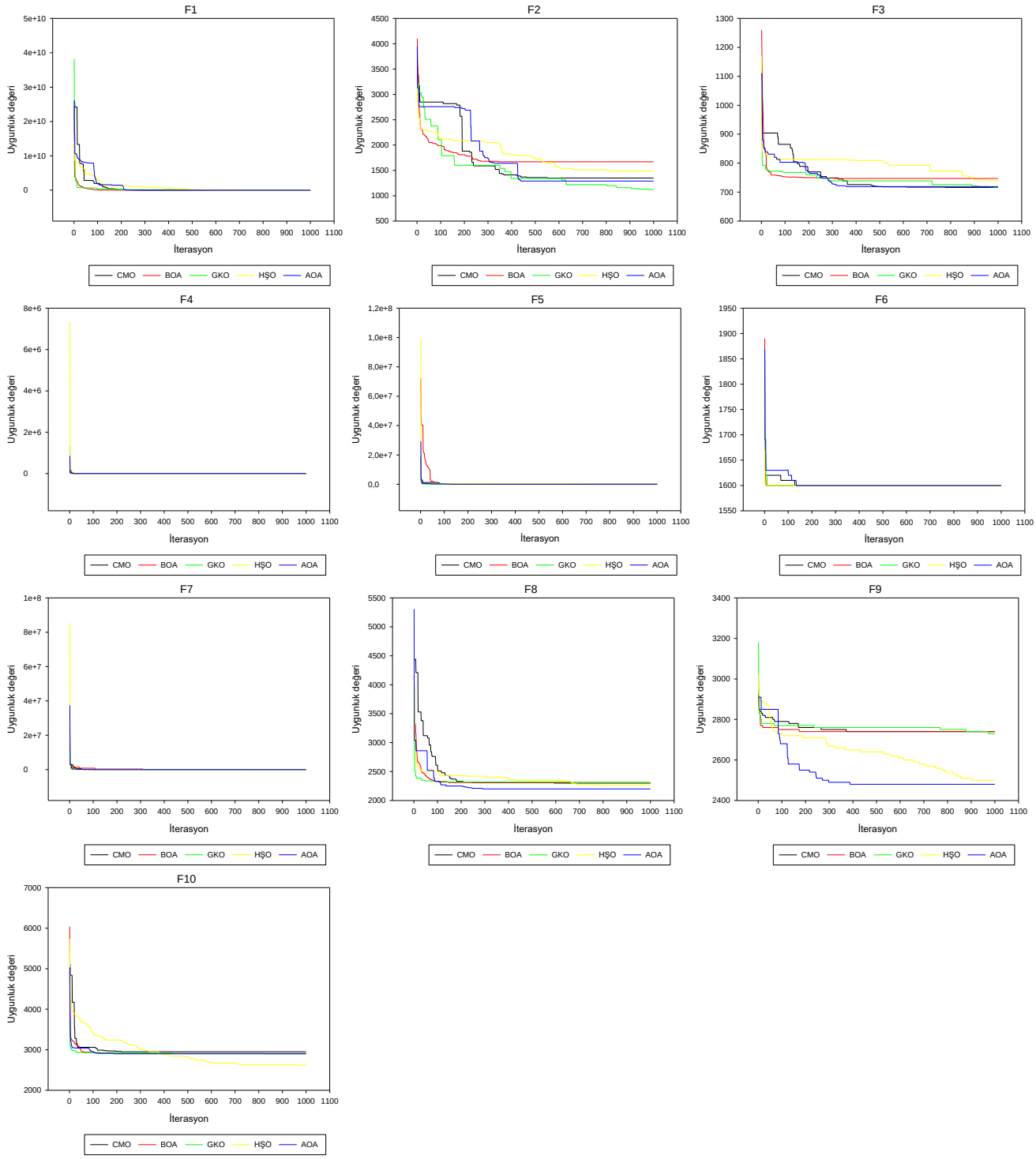
Multimodal fonksiyonlar çok sayıda yerel optimuma sahip olduğu için ve unimodal fonksiyonlara kıyasla problemin boyutu ile tasarım değişkenlerinin sayısı katlanarak arttığı için kullanılmıştır. Bu nedenle bu fonksiyonlar rekabetçi algoritmaların keşif kabiliyetini değerlendirmek için kullanışlıdır. Multimodal fonksiyonlarda minimum değer açısından karşılaştırma yaptığımızda CMO en iyi sonuca ulaşmıştır.

Tablo 3. CEC2020 test fonksiyonlarından elde edilen sonuçlar

Fonksiyon	Metrik	CMO	BOA	GKO	HŞO	AOA
F1	Minimum	1.46E+03	4.77E+05	7.34E+03	3.01E+05	**2.06E+02**
	Maksimum	1.27E+04	3.43E+07	4.81E+08	1.99E+06	9.72E+03
	Ortalama	8.17E+03	7.17E+06	4.53E+07	6.11E+05	**2.64E+03**
	Standart sapma	3.59E+03	8.38E+06	1.34E+08	3.88E+05	2.81E+03
F2	Minimum	1.35E+03	1.67E+03	**1.12E+03**	1.49E+03	1.29E+03
	Maksimum	2.01E+03	2.69E+03	2.47E+03	2.70E+03	2.08E+03
	Ortalama	1.65E+03	2.19E+03	**1.56E+03**	2.08E+03	1.66E+03
	Standart sapma	2.01E+02	2.78E+02	2.59E+02	2.83E+02	1.97E+02
F3	Minimum	**7.16E+02**	7.47E+02	7.17E+02	7.39E+02	7.19E+02
	Maksimum	7.43E+02	8.26E+02	7.48E+02	8.26E+02	8.09E+02
	Ortalama	**7.28E+02**	7.86E+02	7.30E+02	7.88E+02	7.46E+02
	Standart sapma	7.95E+00	2.26E+01	9.01E+00	2.29E+01	1.91E+01
F4	Minimum	**1.90E+03**	**1.90E+03**	**1.90E+03**	**1.90E+03**	**1.90E+03**
	Maksimum	1.90E+03	1.91E+03	1.98E+03	1.92E+03	1.90E+03
	Ortalama	**1.90E+03**	1.91E+03	1.91E+03	1.91E+03	**1.90E+03**
	Standart sapma	4.21E-01	3.20E+00	1.77E+01	4.21E+00	5.25E-01
F5	Minimum	**1.72E+03**	1.05E+04	2.36E+03	3.69E+03	2.12E+03
	Maksimum	1.92E+04	1.63E+06	5.63E+05	2.18E+05	8.55E+03
	Ortalama	5.96E+03	2.31E+05	5.23E+04	4.95E+04	**4.02E+03**
	Standart sapma	4.85E+03	3.80E+05	1.38E+05	6.10E+04	2.06E+03
F6	Minimum	**1.60E+03**	**1.60E+03**	**1.60E+03**	**1.60E+03**	**1.60E+03**
	Maksimum	1.60E+03	1.62E+03	1.66E+03	1.63E+03	1.60E+03
	Ortalama	**1.60E+03**	1.61E+03	1.62E+03	1.62E+03	**1.60E+03**
	Standart sapma	2.63E-01	7.61E+00	2.28E+01	1.01E+01	2.46E-01
F7	Minimum	**2.12E+03**	8.35E+03	3.01E+03	3.19E+03	2.20E+03
	Maksimum	2.47E+04	9.43E+05	1.91E+04	3.24E+04	4.05E+03
	Ortalama	9.17E+03	1.28E+05	9.71E+03	1.20E+04	**2.63E+03**
	Standart sapma	7.36E+03	1.94E+05	4.90E+03	9.15E+03	4.79E+02
F8	Minimum	2.23E+03	2.23E+03	2.30E+03	2.26E+03	**2.20E+03**
	Maksimum	3.56E+03	4.10E+03	2.33E+03	3.74E+03	3.53E+03
	Ortalama	2.38E+03	2.41E+03	**2.31E+03**	2.45E+03	**2.31E+03**
	Standart sapma	2.84E+02	3.89E+02	8.47E+00	4.12E+02	2.81E+02
F9	Minimum	2.74E+03	2.74E+03	2.73E+03	2.50E+03	**2.48E+03**
	Maksimum	2.78E+03	2.85E+03	2.77E+03	2.90E+03	2.81E+03
	Ortalama	2.76E+03	2.78E+03	2.75E+03	2.76E+03	**2.70E+03**
	Standart sapma	8.99E+00	2.65E+01	1.18E+01	1.15E+02	1.18E+02
F10	Minimum	2.90E+03	2.91E+03	2.90E+03	**2.61E+03**	2.90E+03
	Maksimum	2.97E+03	3.04E+03	2.95E+03	2.95E+03	2.95E+03
	Ortalama	2.93E+03	2.95E+03	2.93E+03	**2.91E+03**	2.93E+03
	Standart sapma	2.65E+01	2.92E+01	1.78E+01	7.12E+01	1.97E+01

Ancak ortalama değer açısından bir değerlendirme yaptığımız zaman hem CMO hem de AOA'nın 3 multimodal fonksiyonun 2'sinde en iyi sonuca ulaştığını söyleyebiliriz. Buna dayanarak multimodal fonksiyonlarda CMO'nun minimum değerde, CMO ve AOA'nın ortalama değerde diğer algoritmalara göre daha üstün olduğu sonucuna varabiliriz.

Hibrit fonksiyonlar çok sayıda yerel optimuma sahip olan yerel optimumdan kaçınma yeteneğini ve keşif ve sömürü arasındaki dengeyi değerlendirmek için kullanılmıştır. Deneysel sonuçlar incelendiğinde minimum değer açısından karşılaştırma yaptığımızda CMO en iyi sonuca ulaşmıştır. Ancak ortalama değer açısından bir değerlendirme yaptığımız zaman AOA 3 hibrit fonksiyonun tamamında en iyi sonuca ulaşmıştır. Buna dayanarak hibrit fonksiyonlarda CMO'nun minimum değerde, AOA'nın da ortalama değerde diğer algoritmalara göre daha üstün olduğu sonucuna varabiliriz.



Şekil 3. CEC2020 test fonksiyonlarda karşılaştırma yapılan yöntemlerin yakınsama performansı

Composition fonksiyonlarda hibrit fonksiyonlar gibi çok sayıda yerel optimuma sahip olan yerel optimumdan kaçınma yeteneğini ve keşif ve sömürü arasındaki dengeyi değerlendirmek için kullanılmıştır. Deneysel sonuçlar incelendiğinde ortalama değer açısından karşılaştırdığımızda en iyi sonucu AOA'nın verdiğini söyleyebiliriz.

Sonuçlar

Metasezgisel yöntemler arasında her problemde en iyi performansı gösteren bir algoritma bulunmadığı için araştırmacılar için güncelliğini korumaktadır. Bu yüzden yeni yöntemler önerilmekte ve önerilmeye devam edeceği gözükmektedir. Önerilen yöntemler ile paralel bir şekilde yöntemlerin uygulama alanlarının çoğalması da bu alanın popülerliğini olumlu yönde etkilemektedir. Yapılan bu çalışmada son yıllarda önerilen ve popüler

olan 5 farklı metasezgisel optimizasyon problemi seçilerek, optimizasyon problemlerinin performanslarının analiz edilmesinde literatürde yer alan CEC fonksiyonlarına uygulanmıştır. CEC fonksiyonları arasından CEC2020 test fonksiyonlarında yer alan metasezgisel optimizasyon algoritmalarının farklı yeteneklerini gösterecek farklı türdeki problemler seçilmiştir. Deneysel olarak elde edilen sonuçlar yöntemlerin performanslarının problemin türüne göre değişiklik olabileceğini göstermiştir. Yapılan çalışmanın sonucunda AOA, genel performans olarak en iyi sonuçları elde ederken onu CMO izlemiştir. Sonuç olarak araştırmacılar problemlerine göre optimizasyon yöntemi seçerken elde edilen sonuçları göz önünde bulundurmaları yani problemlerinin türüne göre en iyi optimizasyon yöntemini seçmeleri gerekmektedir. Bunlara ek olarak, diğer gerçek ölçekli optimizasyon problemlerini çözmek için farklı kaotik haritalar, ikili ve çok amaçlı yetenekler eklenerek yöntemlerin farklı modifikasyonları gerçekleştirilebilir.

Kaynaklar

[1] B. Bunday, Basic Optimization Methods, London: Edward Arnold Ltd, 1984.

[2] E. V. Altay, B. Alatas, "Bird swarm algorithms with chaotic mapping", Artificial Intelligence Review, vol. 53 no. 2, pp. 1373-1414, 2020.

[3] E. Varol, B. Alataş, "Sürü zekâsında yeni bir yaklaşım: Kuş sürüsü algoritması", Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, vol. 8, no. 1, pp. 133-146, 2017.

[4] S. İ. Birbil, S. C. Fang, "An electromagnetism-like mechanism for global optimization", Journal of global optimization, vol. 25, no. 3, pp. 263-282, 2003.

[5] B. Xing, W. J. Gao, "Central force optimization algorithm", In Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms, Springer International Publishing, pp. 333-337, 2014.

[6] L. Xie, Y. Tan, J. Zeng, Z. Cui, "Artificial physics optimisation: a brief survey", International Journal of Bio-Inspired Computation, vol. 2, no. 5, pp. 291-302, 2010.

[7] M. Kripka, R. M. L. Kripka, "Big crunch optimization method", In International conference on engineering optimization. Brazil, 2008, pp. 1-5.

[8] H. Shah-Hosseini, "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation", Int. J. Computational Science and Engineering, vol. 6, pp. 132-140, 2011.

[9] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, "Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems", Comput. Struct., vol. 110, pp. 151–166, 2012.

[10] B. Xing, W. J. Gao, "Charged system search algorithm", In Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms, Springer International Publishing, 2014, pp. 339-346.

[11] E. Atashpaz-Gargari, C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition", In 2007 IEEE Congress on Evolutionary Computation, 2007, pp. 4661-4667.

[12] R. V. Rao, V. Patel, "An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems", Scientia Iranica, vol. 20, no. 3, pp. 710–720, 2013.

[13] A. Borji, M. Hamidi, "A new approach to global optimization motivated by parliamentary political competitions", International Journal of Innovative Computing, Information and Control, vol. 5 no. 6, pp. 1643-1653, 2009.

[14] Z. Cui, Z. Shi, J. Zeng, "Using social emotional optimization algorithm to direct orbits of chaotic systems", International Conference on Swarm, Evolutionary, and Memetic Computing, Springer, 2010, pp. 389-395

[15] Y. Shi, "Brain storm optimization algorithm", In International Conference in Swarm Intelligence, Springer Berlin Heidelberg, 2011, pp. 303-309.

[16] A. Daskin, S. Kais, "Group leaders optimization algorithm", Molecular Physics, vol. 109, no. 5, pp. 761-772, 2011.

[17] S. Balochian, H. Baloochian, "Social mimic optimization algorithm and engineering applications", Expert Systems with Applications, vol. 134, pp. 178-191, 2019.

[18] F. Ramezani, S. Lotfi, "Social-based algorithm (SBA)", Applied Soft Computing, vol. 13, no. 5, pp. 2837-2856, 2013.

[19] E. V. Altay, B. Alatas, "Performance comparisons of socially inspired metaheuristic algorithms on unconstrained global optimization", In Advances in Computer Communication and Computational Sciences, Springer, Singapore, 2019, pp. 163-175.

[20] J. Kennedy, R. C. Eberhart, "Particle swarm optimization", IEEE International Conference on Neural Networks, Piscataway, NJ. Nov/Dec 1995, pp. 1942-1948.

[21] M. Dorigo, T. Stützle, "Ant colony optimization", MIT Press, Cambridge, 2004.

[22] D. Karaboga, B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm", J. Global Optim., vol. 39, pp. 459-171, 2007.

[23] A. Karcı, "A new metaheuristic algorithm based chemical process: Atom Algorithm", 1st

International Eurasian Conference on Mathematical Sciences and Applications, 2012, pp. 03-07.

[24] B. Alataş B., "ACROA: Artificial chemical reaction optimization algorithm for global optimization", Expert Systems with Applications, vol. 38, no.10, pp. 13170–13180, 2011.

[25] D. E. Goldberg, "Genetic algorithm in search: optimization and machine learning", Kluwer Academic Publishers, Boston, USA, 1989.

[26] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control", IEEE control systems magazine, vol. 22, no. 3, pp. 52–67, 2002.

[27] R. Storn, K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces", Journal of global optimization, vol. 11, no. 4, pp. 341–359, 1997.

[28] S. A. Salem, "BOA: A novel optimization algorithm", IEEE 2012 International Conference on Engineering and Technology (ICET), 2012, pp. 1-5.

[29] S. Mirjalili, "SCA: A Sine cosine algorithm for solving optimization problems", Knowledge-Based Systems, vol. 96, pp. 120-133, 2016.

[30] A. Karci, B. Alatas, "Thinking capability of saplings growing up algorithm", In: Intelligent data engineering and automated learning—IDEAL 2006, Lecture notes in computer Science, Springer, Berlin, vol. 4224, pp. 386–393, 2006.

[31] F. Merrikh-Bayat, "The runner-root algorithm: a metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature", Applied Soft Computing, vol. 33, pp. 292–303, 2015.

[32] Y. Zhou, Y. Wang, X. Chen, L. Zhang, K. Wu, "A Novel path planning algorithm based on plant growth mechanism", Soft Computing, vol. 21 no. 2, pp. 435-445, 2017.

[33] Y. Labbi, D. B. Attous, H. A. Gabbar, B. Mahdad, A. Zidan, "A new rooted tree optimization algorithm for economic dispatch with valve-point effect", International Journal of Electrical Power & Energy Systems, vol. 79, pp. 298–311, 2016.

[34] A. A. Kamarudin, Z. A. Othman, H. M. Sarim, "Water flow algorithm decision support tool for travelling salesman problem", In Proceedings of the International Conference on Applied Science and Technology 2016 (ICAST'16), AIP Publishing, 2016, vol. 1761, no. 1.

[35] A. Sadollah, H. Eskandar, A. Bahreininejad, J. H. Kim, "Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems", Applied Soft Computing, vol. 30, pp. 58–71, 2015.

[36] A. Kaveh, T. Bakhshpoori, "Water evaporation optimization: A novel physically inspired optimization algorithm", Computers & Structures, vol. 167, pp. 69-85, 2016.

[37] A. Ibrahim, S. Rahnamayan, M. V. Martin, "Simulated raindrop algorithm for global optimization", IEEE 27th Canadian Conference Electrical and Computer Engineering (CCECE), 2014, pp. 1-8.

[38] A. H. Kashan, "League Championship Algorithm: A new algorithm for numerical function optimization", In 2009 international conference of soft computing and pattern recognition, 2009, pp. 43-48.

[39] E. Khaji, "Soccer League Optimization: A heuristic Algorithm Inspired by the Football System in European Countries", 2014, arXiv preprint arXiv:1406.4462.

[40] H. D. Purnomo, H. M. Wee, "Soccer game optimization: an innovative integration of evolutionary algorithm and swarm intelligence algorithm", Meta-Heuristics optimization algorithms in engineering, business, economics, and finance. IGI Global, Pennsylvania, pp. 386-420, 2013.

[41] N. Moosavian and B. K. Roodsari, "Soccer league competition algorithm, a new method for solving systems of nonlinear equations", International Journal of Intelligence Science, vol. 4, no. 1, pp. 7, 2013.

[42] E. Osaba, F. Diaz, E. Onieva, "A novel meta-heuristic based on soccer concepts to solve routing problems", In Proceedings of the 15th annual conference companion on Genetic and evolutionary computation, ACM, pp. 1743-1744, 2013.

[43] Z. W. Geem, J. H. Kim, G. V. Loganathan, "A new heuristic optimization algorithm: harmony search", Simulation, vol. 76, no. 2, pp. 60-68, 2001.

[44] S. M. Ashrafi, A. B. Dariane, "A novel and effective algorithm for numerical optimization: melody search (MS)", In 2011 11th international conference on hybrid intelligent systems (HIS), IEEE, 2011, pp. 109-114.

[45] R. A. Mora-Gutiérrez, J. Ramírez-Rodríguez, E. A. Rincón-García, A. Ponsich, O. Herrera, & P. Lara-Velázquez, "Adaptation of the musical composition method for solving constrained optimization problems", Soft Computing, vol. 18, no. 10, pp. 1931-1948, 2014.

[46] E. V. Altay, B. Alatas, "Randomness as source for inspiring solution search methods: Music based approaches", Physica A: Statistical Mechanics and its Applications, vol. 537, no. 122650, 2020.

[47] E. V. Altay, B. Alatas, "Music based metaheuristic methods for constrained optimization", In 2018 6th

International Symposium on Digital Forensic and Security (ISDFS), IEEE, 2018, pp. 1-6.

[48] Y. C. Ho, D. L. Pepyne, "Simple explanation of the no-free-lunch theorem and its implications", Journal of optimization theory and applications, vol. 115, no. 3, pp. 549-570, 2002.

[49] S. Li, H. Chen, M. Wang, A. A. Heidari, S. Mirjalili, "Slime mould algorithm: A new method for stochastic optimization", Future Generation Computer Systems, vol. 111, pp. 300-323, 2020.

[50] S. Mirjalili, A. Lewis, "The whale optimization algorithm", Advances in engineering software, vol. 95, pp. 51-67, 2016.

[51] S. Mirjalili, S. M. Mirjalili, A. Lewis, "Grey wolf optimizer", Advances in engineering software, vol. 69, pp. 46-61, 2014.

[52] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, "Harris hawks optimization: Algorithm and applications", Future generation computer systems, vol. 97, pp. 849-872, 2019.

[53] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, "Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems", Applied Intelligence, vol. 51, no. 3, pp. 1531-1551, 2021.

[54] A.W. Mohamed, A.A. Hadi, A. K. Mohamed, N.H. Awad, "Evaluating the performance of adaptive gainingsharing knowledge based algorithm on CEC 2020 benchmark problems", In: 2020 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2020, pp. 1–8.

[55] O. Altay, "Chaotic slime mould optimization algorithm for global optimization", Artificial Intelligence Review, pp. 1-62, 2021.