Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Güncel Metasezgisel Yöntemlerin Standart Kalite Testi Fonksiyonlarında Karşılaştırılması

Osman Altay
Manisa Celal Bayar Üniversitesi, Hasan Ferdi Turgutlu Teknoloji Fakültesi, Yazılım Mühendisliği Bölümü, Manisa, Türkiye
osman.altay@cbu.edu.tr
Makale gönderme tarihi:08.02.2022, Makale kabul tarihi:24.05.2022

Öz

Metasezgisel yöntemler genellikle doğadan ilham alınarak oluşturulmuş algoritmalardır. Bu yöntemler özellikle karmaşık problemlerin çözümünde oldukça başarılı sonuçlar üretmektedir. Önerilen yöntemlerin performansları, uygulanan probleme göre değişiklik göstermektedir. Bu çalışmada son dönemlerde ortaya çıkmış ve popüler olan Harris Şahin Optimizasyon Algoritması, Serçe Arama Algoritması, Çoklu Evren Optimizasyonu, Deniz Avcıları Algoritması ve Coot Optimizasyon Algoritması detaylı bir şekilde incelenmiştir. Bu algoritmalar 23 standart kalite testi fonksiyonlarında analiz edilmiştir. Analiz edilen fonksiyonlar tek modlu kalite testi fonksiyonları, çok modlu kalite testi fonksiyonları, karmaşık boyutlu çok modlu kalite testi fonksiyonlarından oluşmaktadır.

Anahtar Kelimeler: Metasezgisel algoritmalar, kalite testi fonksiyonları, global optimizasyon

Comparison of Current Metaheuristic Methods in Standard Benchmark Functions

Abstract

Metaheuristic methods are generally algorithms inspired by nature. These methods produce very successful results especially in solving complex problems. The performances of the proposed methods vary according to the applied problem. In this study, the recently emerged and popular Harris Hawk Optimization Algorithm, Sparrow Search Algorithm, Multi-verse Optimization, Marine Predators Algorithm and Coot Optimization Algorithm are examined in detail. These algorithms were analyzed in 23 standard benchmark functions. The analyzed functions consist of unimodal benchmark functions, multimodal benchmark functions, fixed dimension multimodal benchmark functions.

Keywords: Metaheuristic algorithms, benchmark functions, global optimization

GİRİŞ

Optimizasyon, belirli bir probleme en uygun çözümü bulma sürecidir. Ayrık, sürekli, statik, dinamik, tek amaçlı, çok amaçlı problemler gibi çeşitli optimizasyon türleri vardır (Dhiman & Kumar 2018). Bu problemler genellikle doğrusal olmayan kısıtlamalar, geniş çözüm alanı ve yüksek hesaplama maliyeti, boyutunun artması ile çözülmesi zor bir hal alabilmektedir. Bu problemlerin geleneksel yöntemlerle çözülmesi çok fazla zaman almaktadır ve elde edilen çözümler genellikle gerçek problemler için fizibilite ve doğruluk gereksinimlerini tam olarak karşılayamayan yerel optimal çözümlerdir (Dhiman ve Kumar, 2019). Bu nedenle bilim adamları klasik optimizasyon yöntemlerine alternatif olarak geleneksel optimizasyon tekniklerinin sınırlamalarını aşmak için doğadan ilham alan metasezgisel algoritmaları geliştirmeyi ve kullanmayı başarmışlardır (Blum ve Roli, 2003; Weise, 2011; Yang, 2010). Metasezgisel algoritmalar hızlı olmaları, tutarlılıkları ve optimuma yakın çözüm doğruluğu ile tanınırlar ve son 20 yıldır oldukça popüler bir hale gelmişlerdir. Şaşırtıcı bir şekilde Genetik Algoritma (GA) (Bonabeau vd., 1999), Parçacık Sürü Optimizasyonu (PSO) (Kennedy ve Eberthan, 1995), Karınca Koloni Optimizasyonu (KKO) (Dorigo vd., 2006) gibi bazı yöntemler sadece bilgisayar bilimcileri arasında değil, birçok farklı alanda da oldukça iyi bilinmekte ve etkili bir şekilde

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

kullanılmaktadır. Metasezgisel yöntemlerin bu kadar yaygın ve popüler hale gelmesi dört ana nedenle özetlenebilmektedir. Bunlar basitlik, esneklik, türev gerektirmeyen mekanizma ve optimumdan kaçınmadır.

Metasezgisel yöntemler oldukça basittir ve çoğunlukla çok basit kavramlardan ilham alarak ortaya çıkmaktadırlar. İlham kaynakları genel olarak fizik kanunları, hayvanların davranışları veya evrimsel kavramlarla ilgilidir. Basitlik, bilgisayar bilimcilerinin farklı doğal kavramları simüle etmesine, yeni metasezgisel yöntemler önermesine, iki veya daha fazla metasezgisel yöntemi hibritleştirmesine veya mevcut metasezgisel yöntemleri geliştirmesine olanak tanımaktadır. Ayrıca, basitlik, diğer bilim adamlarının metasezgisel yöntemleri hızlı bir şekilde öğrenmelerine ve bunları problemlerine uygulamalarına yardımcı olmaktadır. İkincisi, esneklik, algoritmanın yapısında herhangi bir özel değişiklik olmaksızın metasezgisel yöntemlerin farklı problemlere uygulanabilirliğini ifade etmektedir. Metasezgisel yöntemler, problemleri çoğunlukla kara kutular olarak kabul ettikleri için farklı problemlere kolaylıkla uygulanabilmektedir. Başka bir deyişle, bir sistemin yalnızca girdileri ve çıktıları bir metasezgisel yöntem için önemlidir. Bu nedenle, bir tasarımcının tek ihtiyacı, problemini metasezgisel olarak nasıl temsil edeceğini bilmektir. Üçüncüsü, metasezgisel yöntemler türev gerektirmeyen bir mekanizmaya sahiptir. Gradyan tabanlı optimizasyon yaklaşımlarının aksine, metasezgisel yöntemler problemleri stokastik olarak optimize ederler. Optimizasyon süreci rastgele çözümler ile başlar ve optimumu bulmak için arama uzaylarının türevini hesaplama ihtiyacı duymaz. Son olarak, metasezgisel yöntemler, geleneksel optimizasyon tekniklerine kıyasla yerel optimumdan kaçınma konusunda üstün yeteneklere sahiptir. Bunun nedeni, yerel çözümlerde durgunluktan kaçınmalarına ve tüm arama uzayını kapsamlı bir şekilde aramalarına izin veren metasezgisellerin stokastik doğasıdır. Gerçek problemlerin arama uzayı genellikle bilinmez ve çok sayıda yerel optima ile çok karmaşıktır, bu nedenle metasezgisel yöntemler bu zorlu gerçek problemleri optimize etmek için iyi seçeneklerdir (Mirjalili vd., 2014).

Genel amaçlı metasezgisel yöntemler, bitki, biyoloji, fizik, kimya, matematik, müzik, sürü, spor, sosyal, su ve melez tabanlı olmak üzere 11 kategoride incelenebilir (Altay ve Alatas, 2019). Köklü ağaç optimizasyonu (Labbi vd., 2016) bitki tabanlı, GA ve DA biyoloji tabanlı, merkezi kuvvet algoritması (Xing ve Gao, 2014) ve büyük çöküş algoritması (Kripta vd., 2008) fizik tabanlı, yapay atom algoritması (Karcı, 2012) kimya tabanlı, sinüs kosinüs algoritması (Mirjalili, 2016) matematik tabanlı, harmoni arama algoritması (Geem vd., 2001) müzik tabanlı, lig şampiyonası (Kashan, 2009) algoritması spor tabanlı, beyin fırtınası algoritması (Shi, 2011) sosyal tabanlı, su buharlaştırma algoritması (Kaveh ve Bakhshpoori, 2016) su tabanlı yöntemlere örnek olarak verilebilir. Bu yöntemlerin performanslarının incelendiği çalışmalar literatürde bulunmaktadır (Altay ve Alatas, 2020b; Kızıloluk ve Can, 2021).

Metasezgisel algoritmaların kendilerine özgü türleri olmasına rağmen, tüm bu algoritmaların iki ortak aşaması vardır. Bunlar keşif ve sömürü aşamasıdır. Keşif aşaması, algoritmanın belirli bir uygulanabilir bölgede farklı umut vaat eden alanları araştırmasını sağlar ve algoritmanın yerel optimumdan kaçınma yeteneğini belirler. Daha güçlü keşif yeteneğine sahip bir algoritmanın yerel optimumdan kaçınması daha kolay olacaktır. Sömürü aşaması, algoritmanın keşif aşamasında elde edilen umut verici alanda küresel optimumu daha fazla aramasını sağlamaktadır (Altay ve Alatas, 2021). Algoritmanın daha yüksek bir kullanım kapasitesine sahip olması, daha doğru optimal çözümler getirecektir. Bu nedenle algoritma performansının iyileştirilmesinde bu iki aşama arasındaki dengenin sağlanabilmesi, büyük önem arz etmektedir (Altay ve Alatas, 2020a; Altay, 2021). Tüm optimizasyon problemlerini verimli bir şekilde çözmek için kullanılabilecek en iyi optimizasyon algoritması yoktur. Bu no free lunch (NFL) teoremi ile mantıksal olarak kanıtlanmıştır (Ho vd., 2002). Başka bir deyişle, belirli bir metasezgisel yöntem, bir dizi problemde çok umut verici sonuçlar gösterebilir, ancak aynı algoritma, farklı bir dizi problemde düşük performans gösterebilir. Bu teorem; metasezgisel yöntemlerin çalışma alanını oldukça aktif hale getirmektedir ve bu da mevcut yaklaşımların geliştirilmesi ve her yıl yeni metasezgisel yöntemler ortaya çıkmasıyla sonuçlanmaktadır. Literatür incelendiğinde de yüzlerce hatta binlerce optimizasyon yönteminin olduğu görülmektedir. Bu çalışma ile son zamanlarda ortaya çıkmış Harris şahin optimizasyonu, serçe arama algoritması, çoklu evren

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

optimizasyonu ve coot optimizasyonu 23 tane klasik kalite testi fonksiyonlarında performansları karşılaştırılmıştır.

Makalenin geri kalan kısmı şu şekilde planlanmıştır. İkinci bölümde güncel metasezgisel algoritmaların çalışma prensipleri ve sözde kodları ayrıntılı bir şekilde açıklanmıştır. Üçüncü bölümde 23 klasik kalite testi fonksiyonlarının özellikleri ve tanımları verilmiş olup deneysel sonuçlara yer verilmiştir. Son bölüm olan dördüncü bölümde ise sonuçlar kısmı yer almaktadır.

GÜNCEL METASEZGİSEL YÖNTEMLER

Bu bölümde bu çalışmada kullanılan algoritmalar tanıtılmış ve çalışma prensipleri hakkında bilgiler verilmiştir.

Harris Şahin Optimizasyonu (HŞO)

Heidari ve arkadaşları (2019) 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 (1)'de 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 (1)'deki gibidir.

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1|X_{rand}(t) - 2r_2X(t)|, & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)), & q < 0.5 \end{cases} \quad (1)$$

$X(t+1)$, 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 (2)'deki gibi hesaplanmaktadır.

$$X_m(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \quad (2)$$

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 (3)'teki gibi modellenmiştir.

$$E = 2E_0\left(1 - \frac{t}{\max\_t}\right) \quad (3)$$

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.

$$X(t+1) = \Delta X(t) - E|J.X_{rabbit}(t) - X(t)| \quad (4)$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \quad (5)$$

$$J = 2(1 - rand) \quad (6)$$

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

$\Delta X(t)$, şahin ve tavşanın pozisyonları arasındaki farktır. Tavşanın rastgele atlama gücü J, rastgele bir sayı kullanılarak çizilmektedir. Sert kuşatma denklemi Denklem (7)'deki gibidir.

$$X(t+1) = X_{rabbit}(t) - E|\Delta X(t)| \quad (7)$$

$p < 0.5$ 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.

$$Y = X_{rabbit}(t) - E|J.X_{rabbit}(t) - X(t)| \quad (8)$$

Önceki dalış faydalı değilse şahin Levy flight modelini kullanarak dalış yapmaktadır. Kullandığı model Denklem (9)'daki gibidir:

$$Z = Y + S \times LF(D) \quad (9)$$

Burada D problemin boyutudur ve S $1 \times D$ boyutunda rastgele bir vektördür. Levy flight fonksiyonu (LF) Denklem (10) kullanılarak hesaplanan yük uçuş fonksiyonudur.

$$LF(x) = 0.001 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \quad \sigma = \left(\frac{r(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{r\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \quad (10)$$

Burada u ve v, 0 ile 1 arasında rastgele değerlerdir β, 1.5 olan bir sabittir. Bu nedenle şahinlerin pozisyonlarını güncellemek için son strateji yumuşak kuşatma aşaması Denklem (11) ile gerçekleşmektedir.

$$X(t+1) = \begin{cases} Y, & if\ F(Y) < F(X(t)) \\ Z, & if\ F(Z) < F(X(t)) \end{cases} \quad (11)$$

$p < 0.5$ 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 (9) ve Denklem (11) aynıdır. Y değerinin hesaplanması farklıdır. Y değerinin denklemi Denklem (12)'deki gibi hesaplanmaktadır.

$$Y = X_{rabbit}(t) - E|J.X_{rabbit}(t) - X_m(t)| \quad (12)$$

HŞO'nun sözde kodları Şekil 1' de gösterilmiştir (Altay ve Altay, 2021).

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 (3)'te kullanılan E değerini güncelle
    if ($|E| \geq 1$)
      Denklem (1)'i kullanarak konum vektörünü güncelle
    end if
    if ($|E| < 1$)
      if ($p \geq 0.5$ ve $|E| \geq 0.5$)
        Denklem (4)'ü kullanarak konum vektörünü güncelle
      else if ($p \geq 0.5$ ve $|E| < 0.5$)
        Denklem (7)'yi kullanarak konum vektörünü güncelle
      else if ($p < 0.5$ ve $|E| \geq 0.5$)
        Denklem (11)'i kullanarak konum vektörünü güncelle
      else if ($p < 0.5$ ve $|E| < 0.5$)
        Denklem (11)'i kullanarak konum vektörünü güncelle
      end if
  end if
  $t = t + 1$
end while
return X_{rabbit}

Şekil 1. HŞO algoritmasının sözde kodu

Serçe Arama Algoritması (SAA)

Serçelerin grup bilgeliği, yiyecek arama ve yırtıcılığa karşı davranışlarından esinlenerek ortaya çıkmış yeni bir sürü optimizasyonu yaklaşımı olan serçe arama algoritması (SAA) Xue ve Shen tarafından 2020 yılında önerilmiştir. Serçeler, iyi bir hafızaya sahip akıllı bir sosyal yaratık olduğu için, serçe popülasyonunun yiyecek arama sürecinde bazı biyolojik özellikleri vardır:

(1) Serçe popülasyonu genellikle üretici ve tüketici olarak adlandırılan iki türe ayrılmaktadırlar. Üreticilerin yiyecek kaynakları aramak için daha

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

geniş bir arama alanı varken, tüketiciler üreticiler sayesinde yiyecek bulmaktadır.
(2) Serçe avcıyı tespit ettiğinde, bireyler endişe verici sinyaller olarak cıvıldamaya başlarlar. Alarm değeri güvenlik eşiğinden büyük olduğunda, üreticilerin tüm tüketicileri güvenli alana yönlendirmesi gerekmektedir.
(3) Serçeler daha güçlü bir anti-predasyon kabiliyetine sahiptir ve bazı serçeler, yiyecek arama sürecinde yırtıcılardan kaçınmak için izci olarak seçilmektedir.
(4) Üreticiler ve tüketiciler, daha iyi bir besin kaynağı elde etmek için dinamik olarak dönüştürülebilir.
(5) Tüketiciler her zaman üreticiler tarafından sağlanan daha iyi bir besin kaynağı bulabilirler ve hatta bazı tüketiciler daha fazla yiyecek almak için üreticileri izlerler.

Simülasyon deneyinde, yiyecek bulmak için sanal serçeler kullanılmaktadır. Serçelerin konumu da aşağıdaki matriste gösterilmektedir:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,d} \\ x_{2,1} & x_{2,2} & x_{2,d} \\ & \dots & \\ x_{n,1} & x_{n,2,} & x_{n,d} \end{bmatrix} \quad (13)$$

Burada n serçe sayısıdır d optimize edilecek değişkenlerin boyutu göstermektedir. Daha sonra tüm serçelerin uygunluk değerleri Denklem (14)'teki vektörle ifade edilmektedir.

$$F(x) = \begin{bmatrix} f(\left|x_{1,1} \right. & x_{1,2} & \left. x_{1,d}\right|) \\ f(\left|x_{2,1} \right. & x_{2,2} & \left. x_{2,d}\right|) \\ & \dots & \\ f(\left|x_{n,1} \right. & x_{n,2,} & \left. x_{n,d}\right|) \end{bmatrix} \quad (14)$$

SAA'da, daha iyi uygunluk değerlerine sahip üreticiler, arama sürecinde yiyecek elde etme önceliğine sahiptir. Üreticiler yiyecek aramak ve tüm nüfusun hareketine rehberlik etmekten sorumludur. Bu nedenle üreticiler, tüketicilerden daha geniş bir yer yelpazesinde yiyecek arayabilirler. Kural (1) ve (2)'ye göre, her iterasyon sırasında üreticinin konumu Denklem (15)'teki gibi güncellenmektedir.

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} \times \exp\left(\frac{-i}{\alpha.\max\_t}\right), & if\ R_2 < ST \\ X_{i,j}^{t} + Q \times L, & if\ R_2 \geq ST \end{cases} \quad (15)$$

Burada t iterasyon sayısını, $j = 1,2,3, \dots . d$. $X_{i,j}^{t}$ t iterasyonunda i. serçenin j. boyutunun değerini temsil etmektedir. $\alpha \in (0, 1]$ rastgele bir sayıdır. R_2 ($R_2 \in [0,1]$) ve ST ($ST \in [0.5, 1.0]$) sırasıyla alarm değerini ve güvenlik eşiğini temsil etmektedir. Q, normal dağılıma uyan rastgele bir sayıdır. L, içindeki her elemanın 1 olduğu $1 \times d$ matrisini göstermektedir.

$R_2 < ST$ olması, etrafta yırtıcı hayvanın olmadığı ve üreticilerin geniş arama moduna geçtiği anlamına gelmektedir. $R_2 \geq ST$ olması ise, bazı serçelerin avcıyı keşfettiği ve tüm serçelerin hızla diğer güvenli alanlara uçması gerektiği anlamına gelmektedir.

Tüketiciler üreticileri izlemektedirler. Üreticinin iyi yiyecek bulduğunu öğrendiklerinde, yiyecek için rekabet etmek için hemen mevcut konumlarını terk ederler. Tüketiciler için konum güncelleme formülü Denklem (16)'daki gibidir.

$$X_{i,j}^{t+1} = \begin{cases} Q \times \exp\left(\frac{X_{worst}^{t} - X_{i,j}^{t}}{i^2}\right), & if\ i < n/2 \\ X_P^{t+1} + \left|X_{i,j}^{t} - X_P^{t+1}\right| \times A^{+} \times L, & otherwise \end{cases} \quad (16)$$

Burada X_P üretici tarafından işgal edilen en uygun konumdur. X_{worst} mevcut küresel en kötü konumu belirtmektedir. A, içindeki her öğeye rastgele 1 veya-1 atandığı ve $A^{+} = A^{T}\left(AA^{T}\right)^{-1}$ olduğu $1 \times d$ matrisini temsil etmektedir. $i > n/2$ olduğunda, daha kötü uygunluk değerine sahip i. tüketicinin büyük olasılıkla açlıktan ölmek üzere olduğunu göstermektedir.

Simülasyon deneyinde tehlikenin farkında olan bu serçelerin toplam popülasyonun %10 ila %20'sini oluşturduğu varsayılmaktadır. Bu serçelerin başlangıç konumları popülasyonda rastgele oluşturulmaktadır. Matematiksel modeli ise Denklem (17)'deki gibi ifade edilmektedir:

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^{t} + \beta \times \left|X_{i,j}^{t} - X_{best}^{t}\right|, & if\ f_i > f_g \\ X_{i,j}^{t} + K \times \left(\frac{\left|X_{i,j}^{t} - X_{worst}^{t}\right|}{(f_i - f_w) + \varepsilon}\right), & if\ f_i = f_g \end{cases} \quad (17)$$

Burada X_{best} mevcut küresel en uygun konumdur. Adım boyutu kontrol parametresi olarak β, ortalama değeri 0 ve varyansı 1 olan rasgele sayıların normal dağılımıdır. $K \in [-1, 1]$ rastgele

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

bir sayıdır. f_i mevcut serçenin uygunluk değeridir. f_g ve f_w sırasıyla mevcut küresel en iyi ve en kötü uygunluk değeridir. ε sıfır bölme hatasından kaçınmak için en küçük sabittir. SAA'nın sözde kodları Şekil 2'de gösterilmiştir.

Serçelerin başlangıç popülasyonunun ayarlanması
while ($t < maksimum\ iterasyon\ sayısı$)
  Uygunluk değerlerini sırala ve mevcut en iyi ve en kötü bireyi bul
  $R_2 = rand(1)$
  for i= 1: üretici sayısı
    Denklem (15)'i kullanarak serçenin konumunu güncelle
  end for
  for i= 1: (üretici sayısı+1):Serçe sayısı
    Denklem (16)'yı kullanarak serçenin konumunu güncelle
  end for
  for i= 1: tehlikeyi algılayan serçe sayısı
    Denklem (17)'yi kullanarak serçenin konumunu güncelle
  end for
  Geçerli yeni konumu al
  Yeni konum öncekinden iyiyse güncelle
  $t = t + 1$
end while
return X_{best}

Şekil 2. SAA'nın sözde kodları

Çoklu Evren Optimizasyonu (ÇEO)

Çoklu evren optimizasyonu (ÇEO), kozmoloji kavramından esinlenerek Mirjalili ve arkadaşları (2016) tarafından önerilmiş metasezgisel yöntemlerden biridir. Arama uzaylarını keşfetmek için beyaz delik ve kara delik kavramlarını kullanmışlardır. Buna karşılık solucan delikleri ÇEO'nun arama alanlarından faydalanmasına yardımcı olmaktadır. Her çözümün bir evrene benzediğini ve çözümdeki her değişkenin o evrendeki bir nesne olduğunu varsaymışlardır. Bunlara ek olarak, her çözüme çözümün karşılık geldiği uygunluk fonksiyonu değeriyle orantılı olan enflasyon oranı atamışlardır. Optimizasyon sırasında ÇEO'nun evrenlerine aşağıdaki kurallar uygulanmıştır.

1. Enflasyon oranı ne kadar yüksekse, kara deliklere sahip olma olasılığı o kadar düşüktür.
2. Enflasyon oranı ne kadar yüksekse, beyaz deliklere sahip olma olasılığı o kadar yüksektir.
3. Daha yüksek enflasyon oranına sahip evrenler, nesneleri beyaz deliklerden gönderme eğilimindedir.
4. Daha düşük enflasyon oranına sahip evrenler, kara deliklerden daha fazla nesne alma eğilimindedir.
5. Tüm evrenlerdeki nesneler, enflasyon oranından bağımsız olarak solucan delikleri aracılığıyla en iyi evrene doğru rastgele hareketle karşı karşıya kalabilir.

ÇEO'da solucan deliği tünelleri her zaman o ana kadar sağlanan en iyi evren ile bir evren arasında kurulmaktadır. Bu mekanizma matematiksel olarak Denklem (18)'deki gibi tanımlanabilir.

$$x_i^j = f(x) = \begin{cases} \begin{cases} X_j + TDR \times \left((ub_j - lb_j) \times r4 + lb_j \right) & r3 < 0.5 \\ X_j - TDR \times \left((ub_j - lb_j) \times r4 + lb_j \right) & r3 > 0.5 \end{cases} & r2 < WEP \\ x_i^j\ , & r2 \geq WEP \end{cases} \quad (18)$$

X_j o ana kadar oluşturulmuş en iyi evrenin j. parametresini, Seyahat mesafe oranı (TDR) bir katsayı, solucan deliği varoluş olasılığı (WEP) başka bir katsayıyı, lb_j j. Değişkenin alt sınırını, ub_j j. Değişkenin üst sınırını, x_i^j i. evrenin j. parametresini, $r2$, $r3$ ve $r4$, 0 ile 1 arasında oluşturulmuş rastgele sayıları ifade etmektedir.

$$WEP = WEP_{min} + iter \times \left(\frac{WEP_{max} - WEP_{min}}{Maxiter} \right) \quad (19)$$

Burada WEP_{min} WEP'nin minimum değeridir, WEP_{max} WEP'nin maksimum değeridir, iter o anki iterasyonu, $Maxiter$ ise maksimum iterasyon sayısını ifade etmektedir.

$$TDR = 1 - \frac{iter^{1/p}}{maxiter^{1/p}} \quad (20)$$

Burada p iterasyonlar boyunca sömürünün doğruluğudur. p değeri ne kadar yüksekse, kullanım o kadar doğru olur. ÇEO algoritmasında optimizasyon süreci bir dizi rastgele evren oluşturmakla başlamaktadır. Her iterasyonda yüksek enflasyon oranlarına sahip evrendeki nesneler,

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

beyaz/kara delikler aracılığıyla düşük enflasyon oranlarına sahip evrenlere doğru hareket etme eğilimindedir. Bu arada, her bir evren nesnelerinde solucan delikleri aracılığıyla en iyi evrene doğru rastgele ışınlamalarla karşı karşıya kalır. Bu süreç durdurma kriteri sağlanıncaya kadar devam eder. ÇEO'nun sözde kodları Şekil 3'te gösterilmiştir.

Başlangıç popülasyonunun ayarlanması (U)
WEP, TDR ve X_j değerlerini başlat
SU: Sıralanmış evrenler
NI: Evrenlerin enflasyon oranını normalleştir
while ($t < maksimum\ iterasyon\ sayısı$)
  Her bir evrenin uygunluk değerini hesapla
  for her bir evren
  WEP ve TDR parametrelerini güncelle
  $Kara\_delik\_indeksi = i$
    for her bir evren
      $r1 = random[0,1]$
      if ($r1 < NI$
        $Beyaz\_delik\_indeksi = RouletteWheelSelection(-NI)$
        $U(Kara\_delik\_indeksi, j) = SU(beyaz\_delik\_indeksi, j)$
    end if
    $r2 = random[0,1]$
    if ($r2 < WEP$)
      $r3 = random[0,1]$
      $r4 = random[0,1]$
      if (r3< 0.5)
        $X_j + TDR \times \left(\left(ub_j - lb_j\right) \times r4 + lb_j\right)$
      else
        $X_j - TDR \times \left(\left(ub_j - lb_j\right) \times r4 + lb_j\right)$
      end if
    end if
  end for
end for
end while

Şekil 3. ÇEO'nun sözde kodları

Deniz Avcıları Algoritması (DEA)

Faramarzi ve arkadaşları (2020) tarafında önerilen deniz avcıları algoritması (DEA), okyanus avcıları arasındaki farklı yiyecek arama stratejilerine ve biyolojik etkileşimde optimal karşılaşma oranları politikasına dayanan metasezgisel bir yöntemdir. Optimal yiyecek arama için deniz avcıları tarafından seçilen iki tür stratejiye dayanmaktadır. Bunlar Lévy ve Brownian hareketidir.

- Deniz avcıları, avın bol olduğu alanlar için Brownian hareketini kullanırken, av yoğunluğunun düşük olduğu çevre için Lévy stratejisini kullanmaktadırlar.
- Farklı habitatlarda yaşamları boyunca aynı Lévy ve Brownian hareketi yüzdelerini göstermektedirler.
- Doğal (girdap oluşumu) veya insan kaynaklı (FAD'ler) gibi çevresel etkiler nedeniyle, farklı bir av dağılımına sahip alanlar bulma umuduyla davranışlarını değiştirmektedirler.
- Düşük hız oranında ($v = 0.1$), bir avcı için en iyi strateji Lévy'dir; her iki av da Brownian veya Lévy'de hareket etmektedir;
- Birim hız oranında ($v = 1$), av Lévy'de hareket ederse, bir avcı için en iyi strateji Brownian'dır. Diğer senaryolar sistem boyutuna bağlı olarak değişmektedir.
- Yüksek hız oranında ($v \geq 10$) bir avcı için en iyi strateji hiç hareket etmemektir. Bu durumda, avlardan biri Brownian veya Lévy'yi hareket ettirmektedir.
- Başarılı yiyecek aramanın yanı sıra birlikteliklerini hatırlatmak için de iyi bir hafızadan yararlanmaktadırlar.

DEA optimizasyon süreci, farklı hız oranları dikkate alınarak ve aynı zamanda bir avcı ve avın tüm yaşamını taklit eden üç ana optimizasyon aşamasına ayrılmaktadır.

1. Aşama: Yüksek hız oranında veya avcının avdan hızlı hareket etmesi durumunda, Denklem (21)'deki matematiksel model uygulanmaktadır.

$$While\ Iter < \frac{1}{3} Max\_iter$$
$$\overrightarrow{stepsize_i} = \overrightarrow{R_B} \otimes \left(\overrightarrow{Elite_i} - \overrightarrow{R_B} \otimes \overrightarrow{Prey_i}\right) \quad i = 1,2,\ldots.n$$
$$\overrightarrow{Prey_i} = \overrightarrow{Prey_i} + P \times \vec{R} \otimes \overrightarrow{stepsize_i} \qquad (21)$$

Burada R_B, Brownian hareketini temsil eden normal dağılıma dayalı rasgele sayılar içeren bir vektördür. R_B'nin av ile çarpımı, avın hareketini simüle eder. $P = 0.5$ sabit bir sayıdır ve $R, [0, 1]$'deki tek tip rastgele sayıların bir vektörüdür. Bu senaryo, yüksek keşif yeteneği için adım boyutu veya hareket hızı yüksek olduğunda yinelemelerin ilk üçte birinde gerçekleşmektedir.

2. Aşama: Birim hız oranında veya hem avcı hem de av aynı hızda hareket ettiğinde 2. Aşama uygulanmaktadır. Her ikisinin de avlarını aradığını taklit eder. Bu bölüm, keşfin geçici olarak sömürüye dönüştürülmeye çalışıldığı optimizasyonun ara

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

aşamasında gerçekleşmektedir. Bu aşamada hem keşif hem de sömürü önemlidir. Sonuç olarak, nüfusun yarısı keşif, diğer yarısı ise sömürü için ayrılmıştır. Bu aşamada av, sömürüden, avcı ise keşiften sorumludur. Kurala göre, birim hız oranında (v ≈ 1), av Lévy'de hareket ederse, yırtıcı için en iyi strateji Brownian'dır. Bu nedenle, bu çalışma Lévy'de av hareketlerini, Brownian'da yırtıcı hareketlerini dikkate almaktadır.

$$While\ \frac{1}{3}Max_{ıter} < \frac{2}{3}Max_{ıter}$$
$$\overrightarrow{stepsıze_ı} = \overrightarrow{R_L} \otimes \left(\overrightarrow{Elite_ı} - \overrightarrow{R_L} \otimes \overrightarrow{Prey_ı}\right) \quad i = 1,2,\dots.n/2$$
$$\overrightarrow{Prey_ı} = \overrightarrow{Prey_ı} + P \times \vec{R} \otimes \overrightarrow{stepsıze_ı} \qquad (22)$$

Burada R_L, Lévy hareketini temsil eden Lévy dağılımına dayalı rastgele sayıların bir vektörüdür. R_L ve Prey çarpımı, avın hareketini Lévy tarzında simüle ederken, adım boyutunu av pozisyonuna eklemek avın hareketini simüle etmektedir. Lévy dağıtım adım boyutunun çoğu küçük adımlarla ilişkilendirildiğinden, bu bölüm sömürüye yardımcı olmaktadır. Popülasyonların ikinci yarısı için bu çalışma Denklem (23)'teki gibidir:

$$\overrightarrow{stepsıze_ı} = \overrightarrow{R_B} \otimes \left(\overrightarrow{Elite_ı} - \overrightarrow{Prey_ı}\right) \quad i = n/2,\dots.n$$
$$\overrightarrow{Prey_ı} = \overrightarrow{Elite_ı} + P \times CF \otimes \overrightarrow{stepsıze_ı} \qquad (23)$$

CF, yırtıcı hareketi için adım boyutunu kontrol etmek için uyarlanabilir bir parametredir. R_B ve Elite'in çarpımı, avcının Brownian tarzı hareketini simüle ederken, av, Brownian hareketinde yırtıcı hayvanların hareketine dayalı olarak konumunu güncellemektedir.

3. Aşama: Düşük hız oranında veya yırtıcı, avdan daha hızlı hareket ettiğinde 3. Aşama kullanılmaktadır. Bu senaryo, çoğunlukla yüksek sömürü yeteneği ile ilişkilendirilen optimizasyon sürecinin son aşamasında gerçekleşmektedir. Düşük hız oranında ($v = 0.1$) avcı için en iyi strateji Lévy'dir. Bu aşama Denklem (24)'teki gibidir:

$$While\ Iter > \frac{2}{3}Max\_ıter$$
$$\overrightarrow{stepsıze_ı} = \overrightarrow{R_L} \otimes \left(\overrightarrow{R_L} \otimes \overrightarrow{Elite_ı} - \overrightarrow{Prey_ı}\right) \quad i = 1,2,\dots.n$$
$$\overrightarrow{Prey_ı} = \overrightarrow{Elite_ı} + P \times CF \otimes \overrightarrow{stepsıze_ı} \qquad (24)$$

Deniz yırtıcılarında davranış değişikliğine neden olan diğer bir nokta ise girdap oluşumu veya Balık Toplama Cihazları (FAD'ler) etkileri gibi çevresel sorunlardır. Köpekbalıkları zamanlarının %80'inden fazlasını FAD'lerin yakın çevresinde geçirmektedirler ve geri kalan %20'si için, muhtemelen başka bir av dağılımına sahip bir ortam bulmak için farklı boyutlarda daha uzun bir sıçrama yapmaktadırlar. FAD'ler yerel optimumlar ve etkileri arama uzayında bu noktalara hapsolma olarak kabul edilmektedir. Simülasyon sırasında bu daha uzun atlamaların dikkate alınması, yerel optimumda durgunluğu önlemektedir. Böylece, FAD'lerin etkisi matematiksel olarak Denklem (25)'teki gibidir:

$$\overrightarrow{Prey_ı} = \begin{cases} \overrightarrow{Prey_ı} + CF[\overrightarrow{x_{mın}} + \vec{R} \otimes (\overrightarrow{x_{max}} - \overrightarrow{x_{mın}}] \otimes \vec{U}, & r \le FADs \\ \overrightarrow{Prey_ı} + [FADs(1-r) + r](\overrightarrow{Prey_{r1}} - \overrightarrow{Prey_{r2}}), & r > FADs \end{cases} \qquad (25)$$

Burada FADs = 0.2, FADs optimizasyon süreci üzerindeki etki olasılığıdır. U, sıfır ve bir içeren dizilere sahip ikili vektördür. Bu, [0,1]'de rastgele bir vektör oluşturularak ve dizi 0.2'den küçükse, sıfıra ve 0.2'den büyükse bire değiştirilerek oluşturulmaktadır. r, [0, 1]'deki tek tip rastgele sayıdır. X_{min} ve X_{max} boyutların alt ve üst sınırlarını içeren vektörlerdir. r_1 ve r_2 alt simgeleri, av matrisinin rastgele dizinlerini belirtmektedir. DEA'nın sözde kodları Şekil 4'te gösterilmiştir.

Başlangıç popülasyonunun ayarlanması ($i = 1, 2, \dots, n$)
while ($t < maksimum\ iterasyon\ sayısı$)
  Uygunluk değerini hesapla, elite matrisini oluştur ve hafızadan tasarruf et
  if $iter < Maxiter/3$
    Denklem (21)'e göre avı güncelle
  else if $iter < Maxiter/3 < Iter < 2 * Maxiter/3$
    Popülasyonun ilk yarısı için ($i = 1, 2, \dots, n/2$)
      Denklem (22)'ye göre avı güncelle
    Popülasyonun diğer yarısı için ($i = n/2, .., n$)
      Denklem (23)'e göre avı güncelle
  else if $Iter > 2 * Maxiter/3$
    Denklem (24)'e göre avı güncelle
  end if
  Bellek tasarrufu ve elite güncellemesini gerçekleştir.
  FAD etkisini uygula ve Denklem (25)'e göre güncelle.
end while

Şekil 4. DEA'nın sözde kodları

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Coot Optimizasyon Algoritması (COOT)

Naruei ve Keynia tarafından 2021 yılında önerilen yöntem, kuşların su yüzeyindeki davranışından esinlenerek oluşturulmuş yeni bir metasezgisel yöntemdir. Coot sürüsünün su üzerindeki davranışı üç hareketten oluşmaktadır. Bunlar düzensiz bir aktivite hareketi, senkronize hareket etme ve su yüzeyinde zincir hareketidir. Coot'ların farklı kolektif davranışları bulunmaktadır. Su yüzeyinde dört farklı su küreği hareketi bulunmaktadır. Bunlar; rastgele hareket etmek, zincir hareketi, grup liderlerine göre pozisyonun ayarlanması, liderler tarafından grubu optimal alana yönlendirmektir. Tüm optimizasyon algoritmalarında olduğu gibi öncelikle başlangıç popülasyonu oluşturulmaktadır. Başlangıç popülasyonu oluşturulduktan sonra amaç fonksiyonu kullanılarak çözümün uygunluk değeri hesaplanmaktadır. Liderlerin seçimi rastgele olmaktadır.

1. Rastgele hareket: Bu hareketi uygulamak için arama uzayında Denklem (26)'ya göre rastgele bir konum göz önünde bulundurulmaktadır ve coot rastgele bir şekilde bu konuma doğru hareket ettirilmektedir.

$$Q = rand(1, d) \times (ub - lb) + lb \quad (26)$$

Bu coot hareketi, arama uzayının farklı kısımlarını incelemektedir. Algoritma yerel optimalde takılırsa, bu hareket algoritmanın yerel optimalden kaçmasına neden olmaktadır. Coot'un yeni konumu Denklem (27)'deki gibi güncellenmektedir:

$$CootPos(i) = CootPos(i) + A \times R2 \times (Q - CootPos(i)) \quad (27)$$

Burada $R2$ 0 ile 1 arasında rastgele bir sayıdır, A Denklem (28)'e göre hesaplanmaktadır.

$$A = 1 - L \times \left(\frac{1}{max - iter}\right) \quad (28)$$

Burada L mevcut iterasyon sayısıdır $max - iter$ ise maksimum iterasyon sayısını temsil etmektedir.

2. Zincir hareketi: Zincir hareketini uygulamak için iki başlığın ortalama konumu kullanılmaktadır. Bir zincir hareketi uygulamanın başka bir yolu da, önce iki kuyruk arasındaki mesafe vektörünü hesaplamak ve ardından kuyruğu mesafe vektörünün yaklaşık yarısı kadar diğer kümeye doğru hareket ettirmektir. İlk yöntem kullanılmıştır ve Coot'un yeni konumu Denklem (29)'daki gibi hesaplanmıştır:

$$CootPos(i) = 0.5 \times (CootPos(i - 1) + CootPos(i)) \quad (29)$$

3. Grup liderlerine göre pozisyonun ayarlanması: Genellikle grup, grubun önünde birkaç kişi tarafından yönetilmektedir ve geri kalanı grup liderlerine göre konumlarını ayarlamaktadır ve onlara doğru hareket etmek zorundadır. Her bir coot'un konumunu hangi lidere göre ayarlayacağı önemli bir sorudur. Liderlerin ortalama konumu dikkate alınmalıdır ve Cootlar bu ortalama konuma göre konumlarını güncellemelidir. Ortalama konumu dikkate almak erken yakınsamaya neden olmaktadır. Bu hareketi uygulamak ve lideri seçmek için Denklem (30)'a göre bir mekanizma kullanılmaktadır:

$$K = 1 + (iMODNL) \quad (30)$$

Burada i mevcut cootun indeks numarasıdır, NL lider sayısıdır ve K liderin indeks numarasıdır. $Coot(i)$ liderin K'sına göre pozisyonunu güncellemektedir. Denklem (31) seçilen lidere dayalı olarak coot'un bir sonraki konumunu hesaplamaktadır.

$$CootPos(i) = LeaderPos(k) + 2 \times R1 \times \cos(2R\pi) \times (LeaderPos(k) - CootPos(i)) \quad (31)$$

$CootPos(i)$ coot'un mevcut konumu olduğunda, $LeaderPos(k)$ lider konumu seçilmektedir. $R1$ 0 ile 1 arasında rastgele bir sayıdır, R ise -1 ile 1 aralığında rastgele bir sayıyı temsil etmektedir.

4. Grubu liderler tarafından optimal alana yönlendirmek: Grup bir hedefe (optimal alan) yönelik olmalıdır, bu nedenle liderlerin hedefe yönelik konumlarını güncellemeleri gerekmektedir. Liderlerin konumunu güncellemek için Denklem (32) kullanılmaktadır. Bu denklem, mevcut optimal nokta etrafında daha iyi konumlar aramaktadır. Bazen liderler daha iyi pozisyonlar bulmak için mevcut optimal pozisyondan uzaklaşmak zorunda kalırlar. Bu denklem, optimum konuma yaklaşmak ve ondan uzaklaşmak için iyi bir yol sağlamaktadır.

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

$$LeaderPos(i) = \begin{cases} B \times R3 \times \cos(2R\pi) \times \left(gBest - LeaderPos(i)\right) \\ \qquad +gBest, \ R4 < 0.5 \\ B \times R3 \times \cos(2R\pi) \times \left(gBest - LeaderPos(i)\right) \\ \qquad -gBest, \ R4 \geq 0.5 \end{cases} \quad (32)$$

Burada $gBest$ şimdiye kadar bulunan en iyi konumu temsil etmektedir. $R3$ ve $R4$, 0 ile 1 arasında rastgele bir sayıyı temsil etmektedir. B değeri de Denklem (33)'teki gibi hesaplanmaktadır:

$$B = 2 - L \times \left(\frac{1}{max-iter}\right) \quad (33)$$

Coot algoritmasının sözde kodları Şekil 5'te gösterilmiştir.

Başlangıç popülasyonunun ayarlanması (U)
$P = 0.5$, NL (liderlerin sayısı) ve $Ncoot$ (cootların sayısını) değerlerini başlat
Global optimum olarak en iyi coot veya lideri bul ($gBest$)
while ($t < maksimum\ iterasyon\ sayısı$)
  Her bir evrenin uygunluk değerini hesapla
  A ve B parametrelerini Denklem (28) ve Denklem (33)'e göre hesapla
  if rand<P
    R, $R1$ ve $R3$ problemin boyutları boyunca rastgele vektördür.
  else
    R, $R1$ ve $R3$ rastgele bir sayıdır
  for 1: coot sayısı
    Denklem (30)'a göre K parametresini güncelle
    if $rand > 0.5$
      Denklem (31)'e göre pozisyonu güncelle
    else if $rand < 0.5$ $i \cong 1$
      Denklem (29)'a göre pozisyonu güncelle
    else
      Denklem (27)'ye göre pozisyonu güncelle
    end
  end
  Coot'un uygunluk değerini hesapla
  if coot'un uygunluğu<liderin uygunluğu
    temp=lider(k); lider(k)=coot; coot=temp
  end
 end
 for her lider sayısı
    if $rand < 0.5$
      Liderin pozisyonunu Denklem (32.1)'deki gibi güncelle
    else
      Liderin pozisyonunu Denklem (32.2)'deki gibi güncelle
    end
    if liderin uygunluğu< $gBest$
    temp= $gBest$; $gBest$ = lider; lider=temp
  end
 end
 t=t+1
end while

Şekil 5. COOT'un sözde kodları

KALİTE TEST FONKSİYONLARI

Bu bölümde incelenen yöntemlerin performanslarını değerlendirmek için 23 standart kalite testi fonksiyonu seçilerek deneyler yapılmıştır. Bunlar tek modlu, çok modlu ve karmaşık boyutlu çok modlu fonksiyonlardan oluşmaktadır. Tek modlu fonksiyonlar, algoritmanın yakınsama performansının ölçülmesi için kullanılırken, çok modlu fonksiyonlar ise, algoritmanın erken yakınsama probleminin ya da yerele takılma sorununun olup olmadığının öğrenilmesi için kullanılmaktadır. Test için kullanılan standart kalite testi fonksiyonları ve özellikleri fonksiyonun özelliğine göre ayrı ayrı Tablo 1-3 arasında verilmiştir.

DENEY SONUÇLARI

Bu bölümde HŞO, SAA, ÇEO, DEA ve COOT yöntemlerinin performansları 23 tane standart kalite testi fonksiyonunda test edilmiştir. Adil bir değerlendirme yapabilmek için tüm yöntemler için popülasyon sayısı 30, maksimum iterasyon sayısı 1000 ve boyut 30 olarak ele alınmıştır ve tüm deneyler 30 kere çalıştırılarak kaydedilmiştir. Deneylerden elde edilen sonuçlar Tablo 4'te gösterilmiş olup ortalama değer, minimum değer, maksimum değer, medyan değeri, standart sapma değeri ve süre metrikleri cinsinden karşılaştırmalı olarak sunulmuştur. Tek modlu fonksiyonlar, algoritmanın yakınsama performansının ölçülmesi için kullanılmaktadır. Bu fonksiyonlar F1 ile F7 arasındaki fonksiyonları içermektedir. Tablo 4 incelendiğinde; ortalama değer bakımından 7 kalite testi fonksiyonun 6'sında en iyi sonucu HŞO algoritması üretmiştir. Bunlardan sadece F6 fonksiyonunda ortalama değer bakımından en iyi sonucu SAA algoritması üretmiştir.

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Tablo 1. *Tek modlu kalite testi fonksiyonları*

Tek modlu fonksiyonları	Boyut	Aralık	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	n	[-100, 100]	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	n	[-10, 10]	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	n	[-100, 100]	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	n	[-100, 100]	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	n	[-30, 30]	0
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	n	[-100, 100]	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1]$	n	[-128, 128]	0

Tablo 2. *Çok modlu kalite testi fonksiyonları*

Çok modlu kalite testi fonksiyonları	Boyut	Aralık	f_{min}
$f_8(x) = \sum_{i=1}^{n} -x_i sin\left(\sqrt{\lvert x_i \rvert}\right)$	n	[-500, 500]	-418.9829*n
$f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	n	[-5.12, 5.12]	0
$f_{10}(x) = -20\exp\left(-0.2\left(\frac{1}{n}\sum_{i=1}^{n} x_i^2\right)^{0.5}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	n	[-32, 32]	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	n	[-600, 600]	0
$f_{12}(x) = \frac{\pi}{n}\{10sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10,100,4), y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m x_i < a \end{cases}$	n	[-50, 50]	0
$f_{13}(x) = 0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	n	[-50, 50]	0

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Tablo 3. *Karmaşık boyutlu çok modlu kalite testi fonksiyonları*

Karmaşık boyutlu çok modlu kalite testi fonksiyonları	Boyut	Aralık	f_{min}
$f_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	[-65, 65]	1
$f_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[-5, 5]	0.00030
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5, 5]	-1.0316
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)cosx_1 + 10$	2	[-5, 5]	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[-2, 2]	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	[1, 3]	-3.86
$f_{20}(x) = -\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	[0, 1]	-3.32
$f_{21}(x) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.1532
$f_{22}(x) = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.4028
$f_{23}(x) = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.5363

Süre bakımından değerlendirildiğinde ise tek modlu fonksiyonlarda en hızlı yöntemin COOT optimizasyon algoritması olduğu sonucu elde edilmiştir. HŞO diğer rekabetçi algoritmalar ile karşılaştırıldığında, diğer yöntemlere göre daha iyi bir sömürü yeteneği sergilediği sonucuna ulaşılmıştır.

Çok modlu fonksiyonlar, algoritmanın erken yakınsama probleminin ya da yerele takılma sorununun olup olmadığını öğrenmek için kullanılmıştır. Deneysel sonuçlar incelendiğinde ortalama değer bakımından 6 çok modlu fonksiyonun 5 tanesinde SAA algoritması üstünlük gösterirken, onu hemen 4 tane fonksiyonda iyi sonuç üreten HŞO yöntemi takip etmektedir. Yine bu fonksiyonlar süre metriği bakımından değerlendirildiğinde 6 fonksiyonun 6'sında en hızlı yöntem COOT optimizasyon algoritması olmuştur.

Karmaşık boyutlu çok modlu fonksiyonlar çok sayıda yerel optimuma sahip olan yerel optimumdan kaçınma yeteneğini ve keşif ve sömürü arasındaki dengeyi değerlendirmek için kullanılmıştır. Deneysel sonuçlar incelendiğinde F16 ile F19 fonksiyonları arasındaki fonksiyonlarda ortalama değer bakımından tüm algoritmalar aynı sonucu üretmişlerdir. Bu yüzden bu fonksiyonlarda incelenen yöntemlerin birbirlerine üstünlükleri yoktur. Ancak geriye kalan beş fonksiyonun beşinde de DEA algoritması en iyi sonucu üreterek en iyi yöntem

Int. J. Pure Appl. Sci. 8(2);286-301 (2022)

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Tablo 4. *Deneysel sonuçlar*

BF	OA	Ortalama Değer	Minimum Değer	Maksimum Değer	Medyan Değer	Standart Sapma	Süre
F1	HŞO	**9.78E-97**	2.18E-113	**2.60E-95**	1.49E-105	4.75E-96	1.57
	SAA	3.32E-60	**6.94E-206**	9.93E-59	1.17E-80	1.81E-59	2.54
	ÇEO	1.28E+00	7.38E-01	1.89E+00	1.27E+00	3.15E-01	2.48
	DEA	1.28E-22	4.08E-25	1.40E-21	5.63E-23	2.54E-22	3.82
	COOT	4.51E-15	5.72E-53	1.35E-13	2.80E-38	2.47E-14	**1.10**
F2	HŞO	**6.01E-51**	3.77E-61	**1.68E-49**	4.93E-54	3.06E-50	1.59
	SAA	1.28E-25	**2.06E-166**	3.85E-24	8.97E-40	7.03E-25	2.76
	ÇEO	5.77E+00	4.73E-01	1.11E+02	8.70E-01	2.04E+01	2.21
	DEA	2.39E-13	5.65E-15	1.23E-12	1.81E-13	2.48E-13	3.91
	COOT	2.55E-10	1.28E-23	7.64E-09	1.30E-17	1.39E-09	**1.17**
F3	HŞO	**8.08E-70**	6.32E-96	**2.42E-68**	1.88E-83	4.42E-69	7.96
	SAA	1.88E-27	**0.00E+00**	5.38E-26	1.88E-37	9.81E-27	6.97
	ÇEO	2.36E+02	6.80E+01	6.19E+02	2.23E+02	1.15E+02	**4.97**
	DEA	1.22E-04	3.76E-08	1.40E-03	4.77E-05	2.55E-04	10.31
	COOT	6.73E-25	6.23E-53	2.02E-23	2.05E-37	3.68E-24	3.62
F4	HŞO	**1.62E-48**	4.37E-58	**4.39E-47**	1.55E-51	7.99E-48	1.92
	SAA	2.70E-29	**1.17E-140**	8.11E-28	1.66E-47	1.48E-28	2.61
	ÇEO	1.83E+00	8.68E-01	3.41E+00	1.74E+00	6.49E-01	2.55
	DEA	2.80E-09	6.69E-10	5.48E-09	2.51E-09	1.42E-09	3.71
	COOT	5.68E-10	8.96E-25	1.03E-08	1.34E-18	2.06E-09	**1.10**
F5	HŞO	8.37E-03	1.66E-05	4.40E-02	4.83E-03	1.07E-02	3.08
	SAA	**1.77E-05**	**6.33E-09**	**1.49E-04**	2.95E-06	3.32E-05	3.28
	ÇEO	5.64E+02	3.51E+01	2.86E+03	1.08E+02	8.28E+02	2.76
	DEA	2.51E+01	2.45E+01	2.62E+01	2.51E+01	4.26E-01	4.39
	COOT	4.98E+01	2.79E+01	2.36E+02	2.88E+01	4.30E+01	**1.38**
F6	HŞO	1.26E-04	2.23E-07	6.14E-04	5.06E-05	1.49E-04	2.33
	SAA	**5.36E-12**	**7.44E-15**	**6.05E-11**	7.43E-13	1.45E-11	2.60
	ÇEO	1.27E+00	7.18E-01	2.01E+00	1.20E+00	3.01E-01	2.45
	DEA	4.26E-08	1.58E-08	8.80E-08	4.06E-08	1.80E-08	3.74
	COOT	3.01E-01	8.32E-02	8.89E-01	2.29E-01	2.01E-01	**1.12**
F7	HŞO	**1.43E-04**	**2.82E-06**	**4.75E-04**	1.21E-04	1.13E-04	5.10
	SAA	1.75E-03	5.32E-05	4.70E-03	1.76E-03	1.26E-03	4.95
	ÇEO	3.17E-02	1.23E-02	4.80E-02	3.06E-02	1.04E-02	3.87
	DEA	1.11E-03	2.65E-04	2.33E-03	9.90E-04	5.06E-04	6.90
	COOT	5.90E-03	2.15E-04	2.48E-02	4.40E-03	5.57E-03	**2.47**
F8	HŞO	**-1.26E+04**	**-1.26E+04**	**-1.26E+04**	**-1.26E+04**	8.55E-01	3.24
	SAA	-8.78E+03	-9.90E+03	-7.30E+03	-8.68E+03	6.19E+02	3.31
	ÇEO	-7.59E+03	-9.32E+03	-6.52E+03	-7.60E+03	6.67E+02	2.08
	DEA	-9.06E+03	-1.01E+04	-7.86E+03	-8.98E+03	5.33E+02	4.56
	COOT	-7.37E+03	-9.96E+03	-5.71E+03	-7.32E+03	9.70E+02	**1.47**
F9	HŞO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	2.73
	SAA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	2.98
	ÇEO	1.22E+02	6.71E+01	2.16E+02	1.19E+02	3.19E+01	2.84
	DEA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	3.97
	COOT	3.41E-14	0.00E+00	7.39E-13	0.00E+00	1.40E-13	**1.31**
F10	HŞO	**8.88E-16**	**8.88E-16**	**8.88E-16**	**8.88E-16**	0.00E+00	2.73
	SAA	**8.88E-16**	**8.88E-16**	**8.88E-16**	**8.88E-16**	0.00E+00	2.92
	ÇEO	1.80E+00	3.87E-01	3.49E+00	1.83E+00	5.92E-01	2.92
	DEA	1.37E-12	2.64E-13	2.96E-12	1.36E-12	8.04E-13	3.99
	COOT	1.07E-12	8.88E-16	1.70E-11	7.99E-15	3.34E-12	**1.41**

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

F11	HŞO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	3.37
	SAA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	3.70
	ÇEO	8.70E-01	7.36E-01	9.93E-01	8.62E-01	6.57E-02	3.08
	DEA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	4.54
	COOT	2.34E-15	0.00E+00	5.04E-14	0.00E+00	9.67E-15	**1.60**
F12	HŞO	5.14E-06	2.41E-08	2.23E-05	2.30E-06	6.40E-06	11.48
	SAA	**1.45E-12**	**1.76E-15**	**1.75E-11**	**1.39E-13**	3.66E-12	9.37
	ÇEO	2.54E+00	2.06E-01	6.37E+00	2.46E+00	1.66E+00	6.24
	DEA	3.59E-06	1.82E-09	1.07E-04	6.33E-09	1.95E-05	12.87
	COOT	1.65E-01	1.39E-03	1.95E+00	4.49E-03	4.22E-01	**4.93**
F13	HŞO	1.03E-04	6.21E-09	7.62E-04	4.50E-05	1.57E-04	11.59
	SAA	**2.64E-11**	**1.12E-16**	**3.84E-10**	**2.83E-12**	7.50E-11	9.34
	ÇEO	1.81E-01	5.93E-02	3.28E-01	1.63E-01	6.62E-02	6.35
	DEA	1.59E-02	3.31E-08	9.08E-02	1.10E-02	2.20E-02	12.95
	COOT	4.82E-01	6.61E-02	2.08E+00	3.74E-01	4.11E-01	**5.00**
F14	HŞO	1.43E+00	9.98E-01	5.93E+00	9.98E-01	9.94E-01	19.89
	SAA	5.22E+00	9.98E-01	1.27E+01	9.98E-01	5.53E+00	14.56
	ÇEO	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	3.07E-11	8.42
	DEA	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	1.51E-16	17.14
	COOT	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	4.66E-16	**8.23**
F15	HŞO	3.34E-04	3.09E-04	3.94E-04	3.28E-04	2.06E-05	1.94
	SAA	3.39E-04	3.07E-04	6.37E-04	3.07E-04	8.42E-05	2.53
	ÇEO	7.93E-03	4.29E-04	5.66E-02	1.00E-03	1.27E-02	1.11
	DEA	**3.07E-04**	**3.07E-04**	**3.07E-04**	**3.07E-04**	2.67E-15	2.43
	COOT	1.36E-03	3.20E-04	2.04E-02	6.74E-04	3.60E-03	**0.95**
F16	HŞO	**-1.03E+00**	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	1.89
	SAA	**-1.03E+00**	-1.03E+00	-1.03E+00	-1.03E+00	5.38E-16	2.34
	ÇEO	**-1.03E+00**	-1.03E+00	-1.03E+00	-1.03E+00	4.50E-07	0.98
	DEA	**-1.03E+00**	-1.03E+00	-1.03E+00	-1.03E+00	4.59E-16	2.24
	COOT	**-1.03E+00**	-1.03E+00	-1.03E+00	-1.03E+00	7.07E-11	**0.93**
F17	HŞO	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	1.67E-06	1.63
	SAA	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	0.00E+00	2.18
	ÇEO	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	6.51E-07	0.89
	DEA	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	1.09E-14	2.05
	COOT	**3.98E-01**	3.98E-01	3.98E-01	3.98E-01	2.28E-07	**0.82**
F18	HŞO	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	3.12E-06	1.58
	SAA	**3.90E+00**	3.00E+00	3.00E+01	3.00E+00	4.93E+00	2.08
	ÇEO	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	2.30E-05	0.87
	DEA	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	2.10E-15	1.98
	COOT	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	2.40E-10	**0.81**
F19	HŞO	**-3.86E+00**	-3.86E+00	-3.84E+00	-3.86E+00	4.50E-03	2.26
	SAA	**-3.86E+00**	-3.86E+00	-3.86E+00	-3.86E+00	2.33E-15	2.72
	ÇEO	**-3.86E+00**	-3.86E+00	-3.86E+00	-3.86E+00	1.20E-06	1.11
	DEA	**-3.86E+00**	-3.86E+00	-3.86E+00	-3.86E+00	2.45E-15	2.61
	COOT	**-3.86E+00**	-3.86E+00	-3.86E+00	-3.86E+00	8.52E-12	**1.06**
F20	HŞO	-3.10E+00	-3.30E+00	-2.92E+00	-3.10E+00	8.95E-02	2.37
	SAA	-3.29E+00	-3.32E+00	-3.20E+00	-3.32E+00	5.11E-02	2.74
	ÇEO	-3.26E+00	-3.32E+00	-3.20E+00	-3.26E+00	6.14E-02	1.19
	DEA	**-3.32E+00**	**-3.32E+00**	**-3.32E+00**	**-3.32E+00**	**1.28E-11**	2.87
	COOT	-3.30E+00	-3.32E+00	-3.20E+00	-3.32E+00	4.84E-02	**1.09**
F21	HŞO	-5.21E+00	-9.80E+00	-5.00E+00	-5.05E+00	8.67E-01	2.62
	SAA	-8.28E+00	-1.02E+01	-5.06E+00	-1.02E+01	2.50E+00	3.25
	ÇEO	-7.29E+00	-1.02E+01	-2.63E+00	-7.63E+00	3.01E+00	1.30

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

	DEA	**-1.02E+01**	**-1.02E+01**	**-1.02E+01**	**-1.02E+01**	2.63E-11	2.96
	COOT	-8.82E+00	-1.02E+01	-2.63E+00	-1.02E+01	2.76E+00	**1.19**
F22	HŞO	-5.26E+00	-1.03E+01	-5.05E+00	-5.09E+00	9.57E-01	2.91
	SAA	-8.81E+00	-1.04E+01	-5.09E+00	-1.04E+01	2.48E+00	3.17
	ÇEO	-8.83E+00	-1.04E+01	-2.75E+00	-1.04E+01	2.70E+00	1.40
	DEA	**-1.04E+01**	**-1.04E+01**	**-1.04E+01**	**-1.04E+01**	6.11E-11	3.21
	COOT	-1.01E+01	-1.04E+01	-2.75E+00	-1.04E+01	1.40E+00	**1.38**
F23	HŞO	-5.46E+00	-1.01E+01	-5.11E+00	-5.13E+00	1.27E+00	3.42
	SAA	-8.73E+00	-1.05E+01	-5.13E+00	-1.05E+01	2.59E+00	3.46
	ÇEO	-8.40E+00	-1.05E+01	-2.42E+00	-1.05E+01	3.39E+00	1.62
	DEA	**-1.05E+01**	**-1.05E+01**	**-1.05E+01**	**-1.05E+01**	2.04E-11	3.57
	COOT	-9.83E+00	-1.05E+01	-2.43E+00	-1.05E+01	2.18E+00	**1.53**

olmuştur. Böylece keşif ve sömürü arasındaki dengeyi en iyi sağlayan yöntemin DEA olduğu sonucuna varılmıştır.

SONUÇLAR

Yapılan bu çalışmada son yıllarda önerilen güncel ve popüler metasezgisel optimizasyon yöntemleri (HŞO, SAA, ÇEO, DEA ve COOT) 23 farklı kalite testi fonksiyonunda karşılaştırılmıştır. Yöntemlerin performansı problemin türüne göre değiştiği için 23 farklı kalite testi fonksiyonları 3 farklı kategoriden oluşmaktadır. Bunlar tek modlu kalite fonksiyonları, çok modlu kalite fonksiyonları ve karmaşık boyutlu çok modlu kalite fonksiyonlarıdır. Yapılan çalışmanın sonucunda tek modlu kalite fonksiyonlarının çoğunluğunda HŞO algoritmasının daha iyi sonuç verdiği gözlemlenmiştir. Çok modlu fonksiyonların 5 tanesinde SAA algoritması iyi sonuç verirken, 4 tanesinde HŞO algoritmasının iyi sonuç verdiği gözlemlenmiştir. Karmaşık boyutlu çok modlu kalite fonksiyonlarında ise 4 fonksiyonda incelenen yöntemler benzer sonuçları üretirken, geri kalan 5 faklı fonksiyonda DEA yöntemi en iyi performansı gösterdiği gözlemlenmiştir. Elde edilen sonuçlar yöntemlerin performansının probleme göre tamamen değişeceğini göstermektedir. Farklı problemlerin çözümünde farklı yöntemlerin denenmesi gerekmektedir. Araştırmacılar yapılan çalışmadan elde edilen sonuçları kullanarak problemlerinin türüne göre karşılaştırılan yöntemlerden birini seçebilecektir.

ÇIKAR ÇATIŞMASI BEYANI

Yazar bu çalışmada herhangi bir şekilde çıkar çatışması olmadığını beyan eder.

ARAŞTIRMA VE YAYIN ETİĞİ BEYANI

Yazar yapılan çalışmada, araştırma ve yayın etiğine uygun olduğunu beyan eder.

KAYNAKLAR

Altay, E. V. ve Alatas, B. (2020a). Bird swarm algorithms with chaotic mapping. Artificial Intelligence Review, 53(2), 1373-1414.

Altay, E. V. ve Alatas, B. (2021). Differential evolution and sine cosine algorithm based novel hybrid multi-objective approaches for numerical association rule mining. Information Sciences, 554, 198-221.

Altay, E. V. ve Alatas, B. (2020b). Randomness as source for inspiring solution search methods: Music based approaches. Physica A: Statistical Mechanics and its Applications, 537, 122650.

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

Altay, E. V. ve Altay, O. (2021). Güncel metasezgisel optimizasyon algoritmalarının CEC2020 test fonksiyonları ile karşılaştırılması, Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi, 12 (5), pp. 729-741.

Altay, O. (2021). Chaotic slime mould optimization algorithm for global optimization. Artificial Intelligence Review, 1-62.

Bonabeau, E., Dorigo, M. ve Theraulaz, G. Swarm intelligence: from natural toartificial systems: OUP USA; 1999.

Blum, C. ve Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM computing surveys (CSUR), 35 (3), 268–308.

IJPAS
ijpas@munzur.edu.tr
ISSN: 2149-0910

Research article/Araştırma makalesi
DOI:10.29132/ijpas.1070287

Dhiman, G. ve Kumar, V. (2018). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. Knowledge-Based Systems, 159, 20-50.

Dhiman, G. ve Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. Knowledge-Based Systems, 165, 169-196.

Dorigo, M., Birattari, M. ve Stutzle, T. (2006). Ant colony optimization. Comput Intell Magaz, IEEE, 1,28–39.

Faramarzi, A., Heidarinejad, M., Mirjalili, S. ve Gandomi, A. H. (2020). Marine Predators Algorithm: A nature-inspired metaheuristic. Expert Systems with Applications, 152, 113377.

Geem, Z. W., Kim, J. H. ve Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony search. simulation, 76(2), 60-68.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. ve Chen, H. (2019). Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872.

Ho, Y. C. ve Pepyne, D. L. (2002). Simple explanation of the no-free-lunch theorem and its implications. Journal of optimization theory and applications, 115(3), 549-570.

Karcı, A. (2012). A new metaheuristic algorithm based chemical process: Atom Algorithm (p:85). Proc. 1st International Eurasian Conference on Mathematical Sciences and Applications, September 03-07, Pristina, Kosova.

Kashan, A. H. (2009). League Championship Algorithm: A new algorithm for numerical function optimization. In SoCPaR, 43-48.

Kaveh, A. ve Bakhshpoori T. (2016). Water evaporation optimization: A novel physically inspired optimization algorithm. Computers & Structures, 167, 69-85.

Kennedy, J. ve Eberhart, R. (1995). Particle swarm optimization, in Neural Networks, In: Proceedings, IEEE international conference on. 1942–1948.

Kızıloluk, S. ve Can, Ü. (2021). Kalite Test Fonksiyonları Kullanılarak Güncel Metasezgisel Optimizasyon Algoritmalarının Karşılaştırılması. International Journal of Pure and Applied Sciences, 7(1), 100-112.

Kripka, M. ve Kripka, R. M. L. (2008). Big crunch optimization method. In International conference on engineering optimization. Brazil, 1-5.

Labbi, Y., Attous, D. B., Gabbar, H. A., Mahdad, B. ve Zidan, A. (2016). A new rooted tree optimization algorithm for economic dispatch with valve-point effect. Int J Electr Power Energy Syst 79, 298–311.

Naruei, I. ve Keynia, F. (2021). A New Optimization Method Based on Coot Bird Natural Life Model. Expert Systems with Applications, 115352.

Mirjalili, S., Mirjalili, S. M. ve Hatamlou, A. (2016). Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Computing and Applications, 27(2), 495-513.

Mirjalili, S., Mirjalili, S. M. ve Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.

Mirjalili S. (2016). SCA: A Sine cosine algorithm for solving optimization problems. Knowledge-Based Systems, 96, 120-133.

Shi, Y. (2011). Brain storm optimization algorithm. In International Conference in Swarm Intelligence. 303-309, Springer Berlin Heidelberg.

Weise, T. (2011). Global optimization algorithms-theory and application (third edition) Online E-Book http://www.it-weise.de/projects/bookNew.pdf.

Xing, B. ve Gao, W. J. (2014). Central force optimization algorithm. In Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms. 333-337, Springer International Publishing.

Xue, J. ve Shen, B. (2020). A novel swarm intelligence optimization approach: sparrow search algorithm. Systems Science & Control Engineering, 8(1), 22-34.

Yang, X. S. (2010). Engineering optimization: An introduction with metaheuristic applications. Hoboken new jersey: John Wiley & Sons.