Fırat Üniversitesi Müh. Bil. Dergisi
33(2), 721-733, 2021

*Araştırma Makalesi*
https://doi.org/10.35234/fumbd.956011

# Global Optimizasyon için Yeni Bir Hibrit Yöntem: Kaya Kartalı Optimizasyonu-Tanjant Arama Algoritması

**Sinem AKYOL**[1*]
[1] Fırat Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği Bölümü, 23119, Elazığ
[*1] sakyol@firat.edu.tr

**(Geliş/Received: 22/06/2021; Kabul/Accepted: 12/08/2021)**

**Öz:** Optimizasyon, belirli koşullarda bir problem için tüm çözümler arasından en iyisini bulma, en iyileme anlamına gelmektedir. Birçok doğrusal optimizasyon modellerinde klasik optimizasyonun yetersiz kalmasından dolayı metasezgisel algoritmalar önerilmiştir. Metasezgisel algoritmalar, kesin çözümü bulma işleminin tanımlanamadığı durumlarda, kesin çözüme en yakın çözümleri bulmak için kullanılmaktadır. Tüm problemler için en iyi çözümü veren bir algoritma bulunmadığından, yeni metasezgisel algoritmalar önerilmeye veya var olan algoritmalar geliştirilmeye devam edilmektedir. Metasezgisel algoritmaların sahip olduğu keşif ve sömürü yetenekleri dengeli bir şekilde çalışmalıdır. Bazı metasezgisel algoritmalarda bu iki yetenekten biri iyi çalışmaktayken diğeri yetersiz kalabilmektedir. Algoritmaların hibritleştirilmesi ile iki algoritmanın güçlü yanları birleştirilerek daha etkin bir algoritma elde edilebilmektedir. Bu çalışmada Kaya Kartalı Optimizasyonu (KKO)'nun sömürü kabiliyetini arttırmak için daraltılmış keşif aşaması yerine, Tanjant Arama Algoritması (TAA)'nın yoğunlaştırma aşaması uygulanarak yeni bir hibrit yöntem olan Kaya Kartalı Optimizasyonu-Tanjant Arama Algoritması (KKO-TAA) önerilmiştir. KKO-TAA, KKO ve TAA'nın performanslarını karşılaştırmak için, altı adet kalite testi fonksiyonu kullanılmıştır. Deneysel sonuçlar hibrit KKO-TAA'nın, KKO ve TAA'ya göre daha iyi sonuçlar verdiğini ve global optimizasyon için etkili bir yöntem olduğunu göstermektedir.

**Anahtar kelimeler:** Hibrit Kaya Kartalı Optimizasyonu-Tanjant Arama Algoritması, Kaya Kartalı Optimizasyonu, Tanjant Arama Algoritması

## A Novel Hybrid Method for Global Optimization: Aquila Optimizer - Tangent Search Algorithm

**Abstract:** Optimization means finding the best among all solutions for a problem under certain conditions. Because of the inadequacy of classical optimization in many linear optimization models, metaheuristic algorithms have been proposed. Metaheuristic algorithms are used to find the closest solutions to the optimum solution when the optimum solution cannot be defined. Since there is no algorithm that gives the best solution for all problems, new metaheuristic algorithms continue to be proposed or existing algorithms are developed. The exploration and exploitation capabilities of metaheuristic algorithms should work in a balanced way. In some metaheuristic algorithms, one of these two capabilities may work well while the other may be insufficient. By hybridizing the algorithms, a more efficient algorithm can be obtained by combining the strengths of the two algorithms. In this study, a new hybrid method, the Aquila Optimizer-Tangent Search Algorithm has been proposed by applying the intensification search of the Tangent Search Algorithm instead of the narrowed exploration stage in order to increase the exploitation capability of the Aquila Optimizer. Six benchmark test functions are used to compare the performances of Aquila Optimizer-Tangent Search Algorithm, Aquila Optimizer, and Tangent Search Algorithm. Experimental results show that hybrid Aquila Optimizer-Tangent Search Algorithm gives better results than the other two algorithms and is an effective method for global optimization.

**Key words:** Aquila Optimizer-Tangent Search Algorithm, Aquila Optimizer, Tangent Search Algorithm.

## 1. Giriş

En iyileme anlamına gelen optimizasyon, belirli koşullarda bir problem için tüm çözümler arasından en iyisini bulma işidir. Optimizasyon problemi, belirli kısıtlamaları sağlamak şartıyla, bilinmeyen değişken değerlerinin bulunmasını hedefleyen herhangi bir problem olarak ifade edilebilmektedir [1]. Birçok optimizasyon algoritması, amaç fonksiyonu ve sistem modeli için matematiksel modellere ihtiyaç duymaktadır ve matematiksel modellerin kurulması genellikle zor olmaktadır ya da yüksek maliyetli olduğundan dolayı kullanılamamaktadır [2]. Doğrusal olmayan ve büyük ölçekli tümleşik problemlerde ve tamsayı ya da ayrık karar parametrelerinin kullanıldığı birçok doğrusal optimizasyon modellerinde klasik optimizasyon yetersiz kalmaktadır. Bu nedenle klasik optimizasyon algoritmaları, verilen bir probleme çözüm uyarlamada etkin olamamaktadır [3].

Çözüm yöntemi, klasik optimizasyon algoritmalarında daha çok problemi modellemede kullanılan değişkenlerin tipine (tamsayı, gerçel), amaç ve sınırlayıcıların tipine (doğrusal, doğrusal olmayan vb.) bağlı

[*] Sorumlu yazar: sakyol@firat.edu.tr.Yazarların ORCID Numarası: [1] 00000-0001-9308-3500

olurken, etkin olması da benzer şekilde problem modellemede çözüm uzayı (içbükey, dışbükey vb.), karar değişken sayısı ve sınırlayıcı sayısına bağlı olmaktadır. Diğer önemli bir eksiklik ise; farklı tipte karar değişkenleri, amaç ve sınırlayıcıların olması durumunda problem formülasyonlarına uygulanabilecek genel çözüm stratejileri sunmamalarıdır. Yani çoğu algoritma belirli tipteki amaç fonksiyonu ya da sınırlayıcıların olduğu modelleri çözmektedir. Ancak çoğu yönetim bilimi, bilgisayar, mühendislik gibi bir çok farklı alandaki optimizasyon problemleri eşzamanlı olarak formülasyonlarında farklı tipteki karar değişkenleri, amaç fonksiyonu ve sınırlayıcıları gerektirir. Bu yüzden klasik ve genel amaçlı metasezgisel optimizasyon algoritmaları önerilmiştir. Bu yöntemlerin hesaplama gücünün iyi ve dönüşümlerinin kolay olmasından dolayı, son yıllarda oldukça popüler yöntemler haline gelmiştir. Örneğin, tek amaç fonksiyonlu bir problem için yazılmış bir metasezgisel program, kolaylıkla çok amaçlı bir probleme ya da farklı bir probleme uyarlanabilmektedir [3-5].

Gerçek yaşam problemlerinin çoğunda problemin çözüm uzayı sonsuz veya tüm çözümlerin değerlendirilemeyeceği kadar büyük olur. Bunun için kabul edilebilir bir sürede çözümlerin değerlendirilerek iyi bir çözümün bulunması gerekmektedir. Böyle problemler için kabul edilebilir bir sürede çözümlerin değerlendirilmesiyle aslında tüm çözüm uzayında "bazı çözümlerin" değerlendirilmesi aynı anlama gelmektedir. Bazı çözümlerin neye göre ve nasıl seçileceği metasezgisel tekniğe göre değişmektedir. Değerlendirmeye dahil olan çözümlerin içerisinde optimal çözümün yer alması garanti edilememektedir. Bu sebeple de metasezgisel tekniklerin bir optimizasyon problemine önerdiği çözüm, optimal değil iyi çözüm olarak algılanmalıdır [6, 7].

Genel amaçlı metasezgisel yöntemler; biyoloji tabanlı, fizik tabanlı, sürü tabanlı, sosyal tabanlı, müzik tabanlı, kimya tabanlı, spor tabanlı ve matematik üzere sekiz farklı grupta değerlendirilebilmektedir. Ayrıca bunların birleşimi olan melez yöntemler de vardır [3-5]. Genetik Algoritma (GA) [8], Diferansiyel Gelişim Algoritması (DEA) [9] ve Karınca Koloni Algoritması (ACO) [10] biyolojik tabanlı; Emperyalist Yarışmacı Algoritma (ICA) [11] ve Parlamenter Optimizasyon Algoritması (POA) [12] sosyal tabanlı; Yapay Kimyasal Reaksiyon Algoritması (ACROA) [13] kimya tabanlı; Armoni Arama Algoritması (HS) [14] müzik tabanlı; Yerçekimsel Arama Algoritması (GSA) [15] ve Zeki Su Damlacıkları Algoritması (IWD) [16] fizik tabanlı, Parçacık Sürü Optimizasyonu (PSO) [17], Kedi Sürüsü Optimizasyonu (CSO) [18] sürü tabanlı, Lig Şampiyonası Algoritması (LCA) [19] spor tabanlı ve Baz Optimizasyon Algoritması [20] ile Matheuristics [21] Matematik Tabanlı Algoritma ve modellerdir. Kültürel Algoritma [22] da hem biyoloji hem de sosyal tabanlı algoritma olarak sınıflandırılabilir [3-5, 23]. Bazı çalışmalarda ise bitki zekasından esinlenerek geliştirilen algoritmaları, bitki tabanlı yöntemler olarak ayrı bir grupta incelenmiştir [24].

Metasezgisel algoritmaların keşif ve sömürü yeteneklerine sahip olması gerekmektedir. Bu iki yetenek dengeli bir şekilde çalışmalıdır. Bazı metasezgisel algoritmalarda sömürü yeteneği iyi çalışırken keşif yeteneği eksik kalabilmektedir ya da keşif yeteneğinde iyi çalışmaktayken sömürü yeteneğinde yeterli olamamaktadır. Algoritmaların hibritleştirilmesi ile iki algoritmanın güçlü yanları birleştirilerek daha etkin bir algoritma elde edilebilmektedir. Bu makalede yeni bir hibrit algoritma olan KKO-TAA önerilmiştir. KKO yönteminde keşif aşaması iterasyon sayısının 2/3 oranında uygulanırken, sömürü aşaması yetersiz kalmıştır. Sömürü kabiliyetine daha fazla yer vermek için TAA'nın etkili olan yoğunlaştırma aşaması KKO'nun daraltılmış keşif aşaması yerine uygulanmıştır. Önerilen hibrit algoritmanın, KKO ve TAA'ya göre daha erken sürede daha iyi sonuçlar elde ettiği görülmüştür.

Bu çalışmanın ikinci bölümünde, KKO ve TAA detaylı olarak açıklanmıştır ve akış diyagramları verilmiştir. Üçüncü bölümde, hibrit KKO-TAA algoritması anlatılarak, akış diyagramında iki algoritmanın nasıl birleştirildiği gösterilmiştir. Dördüncü bölümde KKO, TAA ve KKO-TAA algoritmalarının performansları, kalite testi fonksiyonları kullanılarak elde edilen deneysel sonuçlar karşılaştırılarak sunulmuştur.

## 2. Standart KKO ve TAA

Bu bölümde kaya kartallarının avlanma becerisinden esinlenerek geliştirilen KKO [25] ve matematiksel tanjant fonksiyonuna dayanan TAA [26] detaylıca açıklanarak akış diyagramları verilmiştir.

### 2.1. Kaya kartalı optimizasyonu

Kartalların en yaygın türü olan kaya kartalları, Kuzey Yarımküre'deki en popüler yırtıcı kuşlardandır. Kaya kartalları tarafından kullanılan dört avlanma yöntemi şu şekilde ifade edilmektedir:

İlk yöntem olan dikey eğimli yüksek uçmayı, kaya kartalı yerden yüksek bir seviyede yükseldiği uçuş sırasında kuşları avlamak için kullanmaktadır. Avını keşfettikten sonra ve avına daha da yaklaştıkça, kanatlar hızla yükselen uzun ve düşük açılı bir süzülüşe girmektedir. Kaya kartalı, bu yöntemin başarısı için avının üzerinde bir yükseklik özelliğine ihtiyaç duymaktadır. Avına yönelmeden hemen önce, bir gök gürültüsü gibi görünmesini

sağlamak için, kanatlar ile kuyruk açılmaktadır ve ayaklar, avı yakalamak için öne doğru itilmektedir [27]. İkinci yöntem olan kısa süzülme atağıyla kontur uçuşu, kaya kartalının yerden düşük bir seviyede yükseldiği uçuştur. Bu kartallar tarafından en sık kullanılan yöntem olarak kabul edilmektedir. Av, ister koşuyor ister uçuyor olsun, yakından takip edilmektedir. Bu yöntem, yer sincaplarını, orman tavuğunu veya deniz kuşlarını avlamak için kullanılmaktadır [25, 28].

Üçüncü yöntem, yavaş bir alçalma saldırısı ile uçuş hareketidir. Bu yöntemde kaya kartalı yere inmektedir ve aşamalı olarak avına saldırmaktadır. Kaya kartalı kurbanını seçmekte ve avının boynuna ve sırtına konmaya çalışmaktadır. Bu av yöntemi, çıngıraklı yılanlar, kirpi, tilki ve kaplumbağa gibi yavaş avlar veya kaçış tepkisi olmayan herhangi bir av için kullanılmaktadır [29]. Dördüncü yöntem, kaya kartalının karada yürüdüğü ve avını çekmeye çalıştığı avlanma türü olan, yürüme ve avı yakalamadır. Geyik veya koyun gibi hayvanların daha genç olanlarını kapsama alanından çıkarmak için kullanmaktadır [30]. KKO algoritması kaya kartallarının, bu dört avlanma yönteminden esinlenerek geliştirilmiştir.

Popülasyon tabanlı bir yöntem olan KKO'da, optimizasyon kuralı, üst sınır ($UB$) ve alt sınır ($LB$) arasında stokastik olarak üretilen aday çözümlerin ($X$) popülasyonu ile başlamaktadır. Şimdiye kadar elde edilen en iyi çözüm, yaklaşık olarak her yinelemede en uygun çözüm olarak belirlenmektedir. $X$, Denklem (1) kullanılarak rastgele üretilen bir dizi aday çözümden oluşmaktadır. $X_i$, $i$. çözümün karar değerlerini (konumlarını), $N$ ise aday çözümlerin (popülasyon) toplam sayısını ve $D$ problemin boyutunu ifade etmektedir. $rand$ rasgele bir sayı, $LB_j$, $j$. alt sınır ve $UB_j$ ise $j$. üst sınırdır [25].

$$X_{ij} = rand * \left(UB_j - LB_j\right) + LB_j, i = 1,2, \dots, N \; j = 1,2, \dots, D \tag{1}$$

KKO algoritması, kaya kartalının avlanma sırasındaki dört farklı davranışını simüle etmektedir. Bu nedenle, KKO algoritmasının optimizasyon prosedürleri dört yöntemde temsil edilmektedir: Dikey eğimle yüksekten süzülerek arama alanını seçme, kısa süzülme saldırısı ve sınır çizgisi ile farklı bir arama alanı içinde keşif yapma, yavaş inişli saldırıyla bir yakınsama arama alanı içinde alçak uçuşla sömürme ve yürüyerek avı yakalama. KKO algoritmasında, $T$ maksimum iterasyon sayısı ve $t$ mevcut iterasyon sayısı olmak üzere, $t \leq \left(\frac{2}{3}\right) * T$ koşulunun sağlandığı durumlarda keşif adımları uygulanmaktadır, aksi durumlarda ise sömürü adımlarına geçilmektedir.

Kaya kartallarının davranışlarını matematiksel bir optimizasyon paradigması olarak modellenmiştir ve bu, belirli kısıtlamalara göre en iyi çözümü belirlemektedir [25].

### 2.1.1. Adım 1: Genişletilmiş keşif ($X_1$)

İlk yöntemde ($X_1$), kaya kartalı av bölgesini tanır ve dikey eğimde yüksek süzülme ile en iyi avlanma alanını seçmektedir. Burada, KKO, avın bulunduğu arama alanını belirlemek için yüksekten uçarak geniş çapta keşifler yapmaktadır. Bu davranış matematiksel olarak Denklem (2)'de olduğu gibi sunulmuştur.

$$X_1(t+1) = X_{best}(t) * \left(1 - \frac{t}{T}\right) + (X_M(t) - X_{best}(t) * rand) \tag{2}$$

Burada, $X_1(t+1)$, ilk arama yöntemi ($X_1$) tarafından üretilen $t+1$. iterasyonun çözümüdür. $X_{best}(t)$, $t$. iterasyona kadar elde edilen en iyi çözümdür ve bu avın yaklaşık yerini yansıtmaktadır. $\left(1 - \frac{t}{T}\right)$ denklemi, iterasyonların sayısı aracılığıyla genişletilmiş aramayı (keşif) kontrol etmek için kullanılmaktadır. $X_M(t)$, Denklem (3) kullanılarak hesaplanan, $t$. iterasyondaki mevcut çözümlerin konumlarının ortalama değerini belirtmektedir. $rand$, 0 ile 1 arasında rastgele bir değerdir [25].

$$X_M(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t), \forall j = 1,2, \dots, D \tag{3}$$

### 2.1.2. Adım 2: Daraltılmış keşif ($X_2$)

İkinci yöntemde ($X_2$), av alanı yüksekten uçularak bulunmaktadır. Kaya kartalı hedef avın üzerinde daireler çizmekte, av bölgesini hazırlamakta ve ardından saldırmaktadır. Bu yönteme kısa bir süzülme saldırısıyla kontur uçuşu denmektedir. Burada KKO, saldırıya hazırlanırken hedef avın seçilen bölgesini dar bir şekilde araştırmaktadır. Bu davranış matematiksel olarak Denklem (4)'te olduğu gibi sunulmuştur.

$$X_2(t+1) = X_{best}(t) * Levy(D) + X_R(t) + (y - x) * rand \quad (4)$$

$X_2(t+1)$, ikinci arama yöntemi ($X_2$) tarafından üretilen $t+1$. iterasyonun çözümüdür. $Levy(D)$, Denklem (5) kullanılarak hesaplanan Levy uçuş dağılımı fonksiyonudur. $X_R(t)$ $t$. iterasyonda $[1\ N]$ aralığında alınan rastgele bir çözümdür [25].

$$Levy(D) = s * \frac{u*\sigma}{|v|^{\frac{1}{\beta}}} \quad (5)$$

Burada, $s$ değeri 0.01 olan bir sabittir. $u$ ve $v$, 0 ile 1 arasında rasgele sayılardır. $\sigma$ Denklem (6) kullanılarak hesaplanmaktadır.

$$\sigma = \left( \frac{\Gamma(1+\beta)*\sin\frac{\pi\beta}{2}}{\Gamma\left(\frac{1+\beta}{2}\right)*\beta*2^{\left(\frac{\beta-1}{2}\right)}} \right) \quad (6)$$

Burada, $\beta$ değeri 1.5 olan bir sabittir. Denklem (4)'te $x$ ve $y$ aramadaki spiral şekli sunmak için kullanılmaktadır ve Denklem (7), (8), (9), (10) ve (11) kullanılarak hesaplanmaktadır.

$$y = r * \cos(\theta) \quad (7)$$

$$x = r * \sin(\theta) \quad (8)$$

$$r = r_1 + U * D_1 \quad (9)$$

$$\theta = -\omega * D_1 + \theta_1 \quad (10)$$

$$\theta_1 = \frac{3*\pi}{2} \quad (11)$$

$r_1$, arama döngülerinin sayısını belirlemek için 1 ile 20 arasında bir değer alır ve $U$, değeri 0.00565 olan bir sabittir. $D_1$, 1'den arama alanının uzunluğuna ($D$) kadar olan tam sayılardır ve $\omega$ değeri 0.005 olan küçük bir sabittir [25].

### 2.1.3. Adım 3: Genişletilmiş sömürü ($X_3$)

Üçüncü yöntemde ($X_3$), av alanı doğru bir şekilde belirlendiğinde ve kaya kartalı iniş ile saldırı için hazır olduğunda, av reaksiyonunu keşfetmek için bir ön saldırı ile dikey olarak aşağı inmektedir. Bu yönteme yavaş iniş saldırısı ile alçak uçuş denmektedir. Burada KKO, hedefin seçilen bölgesini avına yaklaşmak ve saldırmak için kullanmaktadır. Bu davranış matematiksel olarak Denklem (12)'de olduğu gibi sunulmaktadır.

$$X_3(t+1) = (X_{best}(t) - X_M(t)) * \alpha - rand + ((UB - LB) * rand + LB) * \delta \quad (12)$$

$X_3(t+1)$, üçüncü arama yöntemi ($X_3$) tarafından üretilen $t+1$. iterasyonun çözümüdür. $\alpha$ ve $\delta$ 0.1 gibi küçük bir değere sabitlenmiş sömürü ayarlama parametreleridir [25].

### 2.1.4. Adım 4: Daraltılmış sömürü ($X_4$)

Dördüncü yöntemde ($X_4$), kaya kartalı avına yaklaştığında, stokastik hareketlerine göre karada saldırmaktadır. Bu adıma yürüyüş ve av yakalama adı verilmektedir. Burada ve nihayetinde, KKO ava son konumda saldırmaktadır. Bu davranış matematiksel olarak Denklem (13)'te olduğu gibi sunulmaktadır.

$$X_4(t+1) = QF * X_{best}(t) - (G_1 * X(t) * rand) - G_2 * Levy(D) + rand * G_1 \quad (13)$$

$X_4(t+1)$, dördüncü arama yöntemi ($X_4$) tarafından üretilen $t+1$. iterasyonun çözümüdür. $QF$, Denklem (14) kullanılarak hesaplanmaktadır ve arama stratejilerini dengelemek için kullanılan bir kalite fonksiyonunu ifade

etmektedir. Denklem (15) kullanılarak hesaplanan $G_1$, kaçırma sırasında avı izlemek için kullanılan, KKO'nun çeşitli hareketlerini belirtmektedir. Denklem (16) kullanılarak hesaplanan $G_2$, ilk konumdan (1) son konuma ($t$) kaçırma sırasında avı takip etmek için kullanılan KKO'nun uçuş eğimini ifade eden, 2'den 0'a azalan bir değerdir. $X(t)$, $t$. yinelemedeki mevcut çözümdür [25].

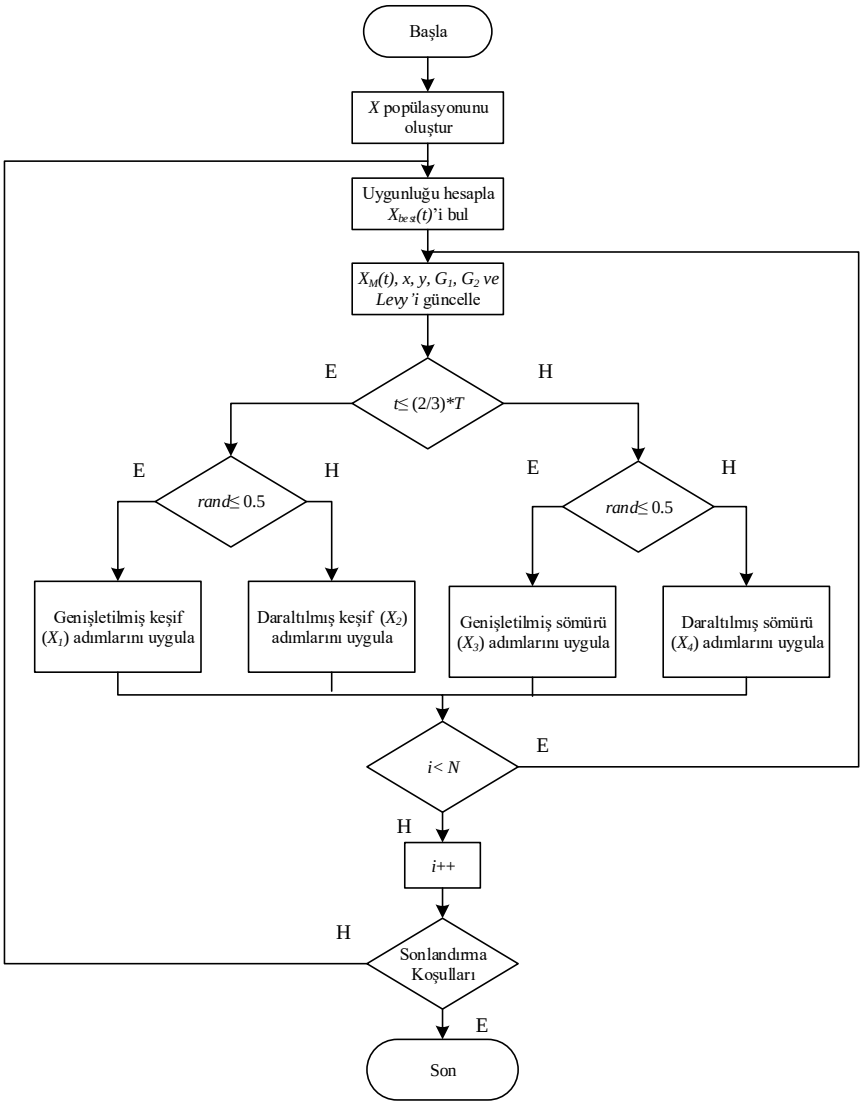
$$QF(t) = t^{\frac{2*rand()-1}{(1-T)^2}} \tag{14}$$

$$G_1 = 2 * \text{rand}() - 1 \tag{15}$$

$$G_2 = 2 * (1 - \frac{t}{T}) \tag{16}$$

$QF(t)$, $t$. iterasyonun kalite fonksiyonu değeridir.

Özetlemek gerekirse, KKO'nun arama stratejileri (yani keşif ve sömürü) arasındaki dengeyi vurgulamak için, keşif ve sömürü için dört farklı arama stratejisi (genişletilmiş keşif, daraltılmış keşif, genişletilmiş sömürü ve daraltılmış sömürü) sunulmaktadır [25]. KKO'nun akış diyagramı Şekil 1'de verilmiştir.




**Şekil 1.** KKO'nun akış diyagramı

### 2.2. Tanjant arama algoritması

Tanjant Arama Algoritması, tanjant fonksiyonu olan basit bir matematiksel fonksiyona dayanmaktadır. Bu fonksiyon, arama alanını iyi keşfetmek için büyük bir kapasite sunmaktadır. Bu fonksiyonun -∞ ve +∞ arasındaki varyasyonu ve bu fonksiyonun periyodikliği, keşif ve sömürü arasında iyi bir denge sağlamaya yardımcı olmaktadır. TAA algoritmasında, tüm hareket denklemi, "$(\theta) step * \tan(\theta)$" formundaki küresel bir adım tarafından yönetilmektedir. Burada tanjant fonksiyonu, Levy uçuş fonksiyonunda olduğu gibi uçuş fonksiyonunun bir rolünü sergilemektedir, bu nedenle buna tanjant uçuşu denmektedir.

Başarılı bir optimizasyon algoritması, sömürü ve keşif arasında daha iyi bir dengeye sahip olmalıdır; çok yoğunlaştırma, algoritmaların hızlı bir şekilde yerel minimuma yakınsamasına neden olmaktadır ve çok fazla keşif, algoritmayı çok yavaşlatmaktadır ve bazen farklılaştırmaktadır. Keşif ve yoğunlaştırma arasındaki dengeye ulaşmak için, TAA üç ana bileşenden oluşturulmuştur: Yoğunlaştırma, keşif ve yerel minimum bileşenlerden kaçış. Son olarak, yerel minimumda tuzağa düşmekten kaçınmak için rasgele bir arama bireyi (çözüm) üzerinde her yinelemede, yerel minimumdan kaçış prosedürü uygulanmaktadır. Diğer popülasyon tabanlı algoritmalarda olduğu gibi, TAA'da ilk popülasyonda bireyler çözüm uzayının sınırları içinde eşit olarak dağıtılmak üzere, Denklem (1) kullanılarak hesaplanmaktadır [26].

#### 2.2.1. Yoğunlaştırma aşaması

Yoğunlaştırma aşamasında TAA, önce aşağıdaki Denklem (17) tarafından yönlendirilen rasgele bir yerel yürüyüş yapmaktadır. Daha sonra elde edilen çözümün bazı değişkenleri, Denklem (18) kullanılarak, mevcut optimal çözümde karşılık gelen değişkenin değerleri ile değiştirilmektedir. Değiştirilen değişkenlerin oranı, boyutu 4'ten büyük bir problem için %20'ye ve 4'e eşit veya daha az değişkene sahip problemler için %50'ye eşit olmaktadır.

$$X_i(t+1) = X_i(t) + step * \tan(\theta) * (X_i(t) - X_{best}(t)) \quad (17)$$

$X_i(t+1) = X_{best}(t)$, eğer $i$ değeri seçilmişse (18)

Sonuç olarak, elde edilen çözüm $X_i(t+1)$, mevcut çözümün yerel olarak geliştirilmesine yardımcı olan mevcut en iyi çözüm ile %50'nin altında bir benzerlik oranına sahiptir. Elde edilen her çözüm $X$, eğer değerleri problemin $LB$ ve $UB$ sınırlarını aşarsa, Denklem (19) ile düzeltilmektedir [26].

$$X = rand * (UB - LB) + LB \quad (19)$$

#### 2.2.2. Keşif aşaması

Yerel arama yöntemlerinin aksine, global rasgele yürüyüş sayesinde büyük bir keşif kapasitesine sahiptir. Bu algoritma, global rasgele yürüyüş için tanjant uçuşunu ve değişken adım boyutunu kullanmaktadır. Tanjant fonksiyonu, arama alanını verimli bir şekilde keşfetmeye yardımcı olmaktadır. Aslında, $\theta$'nın $\pi/2$'ye yakın olması tanjant değerini büyütecek ve elde edilen çözüm mevcut çözümden uzak olacaktır. $\theta$'nın 0'a yakın olması tanjant fonksiyonuna küçük değerler vermektedir ve elde edilen çözüm mevcut çözüme yakın olacaktır. Bu nedenle, keşif aşamasına ait olan Denklem (20), genel ve yerel rasgele yürüyüş arasında birleşmektedir. Keşif arama denklemi her değişkene $1/D$ olasılıkla uygulanmaktadır [26].

$$X_i(t+1) = X_i(t) + step * \tan(\theta) \quad (20)$$

#### 2.2.3. Yerel minimumdan kaçış

TAA, yerel minimum durgunluk probleminden kaçmak için, belirli bir prosedürü kullanan bir mekanizma içermektedir. Prosedür, $Pesc$ olasılık değeri ile yürütülen iki bölümden oluşmaktadır. Her yinelemede rastgele bir ajan araması seçilir ve ardından Denklem (21) ve Denklem (22)'den biri kullanılmaktadır. Ayrıca, yeni rastgele çözüm, en kötü çözümü 0.01 olasılıkla değiştirebilmektedir [26].

$$X = X + R.* \left(best - rand * (best - X)\right) \quad (21)$$

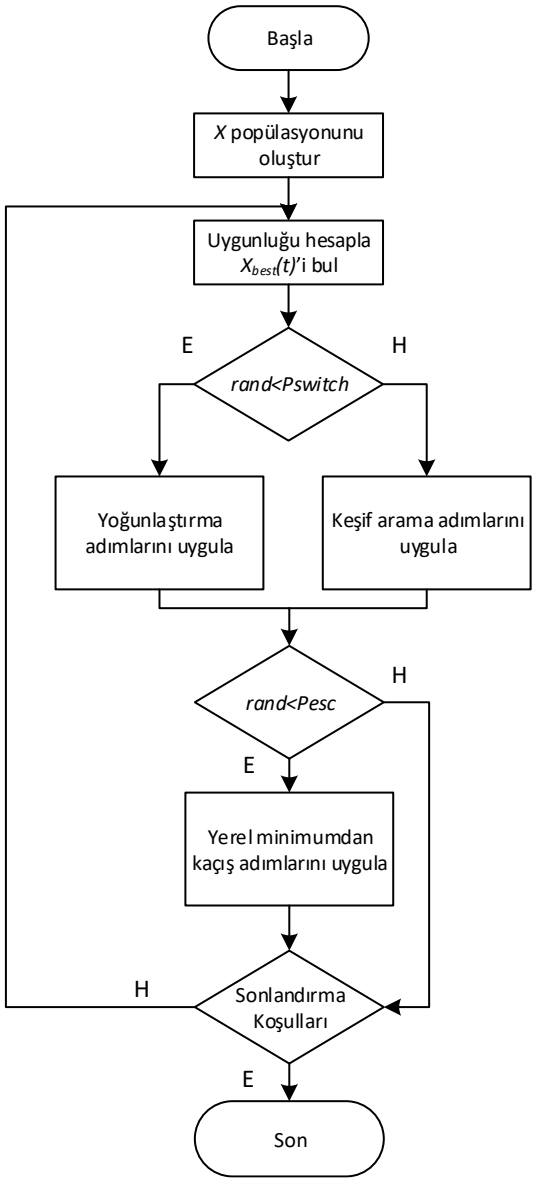
$$X = X + \tan(\theta) * (UB - LB) \quad (22)$$

TAA, birçok optimizasyon algoritmasına kıyasla daha az parametre kullanmaktadır. $Pswitch$, $Pesc$, $step$, $\theta$, yoğunlaştırma ve keşif aramasını vurgulamak için kullanılan ana parametrelerdir. Yerel ve global rasgele yürüyüşler arasındaki denge, $Pswitch\ \epsilon[0,1]$ anahtarlama parametresi tarafından kontrol edilmektedir, ikinci parametre, kaçış prosedürü olasılığı olan $Pesc\ \epsilon[0,1]$'dir. $step$ parametresi, arama sürecinin sömürülmesini ve keşfini yönlendirmek ve vurgulamak için kullanılmaktadır. TAA, en iyi çözüme iyi bir şekilde yaklaşmak ve hassasiyet eksikliğini önlemek için değişken bir adım boyutu kullanmaktadır. Erken aşamada, TAA, arama sürecinin ilerlemesi ile büyük bir adım boyutu benimsemektedir ve adım boyutu yinelemeden yinelemeye doğrusal olmayan bir şekilde azalmaktadır. Bu uyarlanabilir boyut davranışı, TAA'nın keşif ve yoğunlaştırma arasında iyi bir denge sağlamasına yardımcı olmaktadır. Tanjant uçuşun adım boyutu üzerinde büyük etkisi olmasının yanı sıra, ona salınımlı ve periyodik davranış kazandırmaktadır. Keşif ve yoğunlaştırma arama sürecini uyarlamak için TAA, logaritma işlevine dayalı olarak uyarlanabilir adım boyutu için doğrusal olmayan bir azaltma şeması kullanır. Logaritma işlevi, ince bir yakınsamayı sürdürmeye yardımcı olan yavaş bir işlevdir. Öte yandan, aynı algoritmada farklı adım boyutu fonksiyonlarının kullanılmasının özellikle sert yakınsama olan fonksiyonlar için daha iyi sonuçlar verdiği görülmektedir. Bu nedenle, daha verimli olması açısından, TAA adım boyutunun iki çeşidini kullanmaktadır. İlk adım boyutu varyantı yoğunlaştırma aramasında kullanılır ve Denklem (23)'te verildiği gibi hesaplanmaktadır. İkinci adım boyutu ise keşif aramasında Denklem (24) kullanılarak hesaplanmaktadır.

$$step1 = 10 * sign(rand - 0.5) * norm(best) * \log(1 + 10 * d/t) \quad (23)$$

$$step2 = 1 * sign(rand - 0.5) * norm(best - X)/\log(20 + t) \quad (24)$$

Burada: $norm()$ belirli bir matematiksel normdur. $sign(-,+)$ bileşeni, keşif ve yoğunlaştırma aşamasının yönünü kontrol etmektedir [26]. TAA'nın akış diyagramı Şekil 2'de gösterilmektedir.



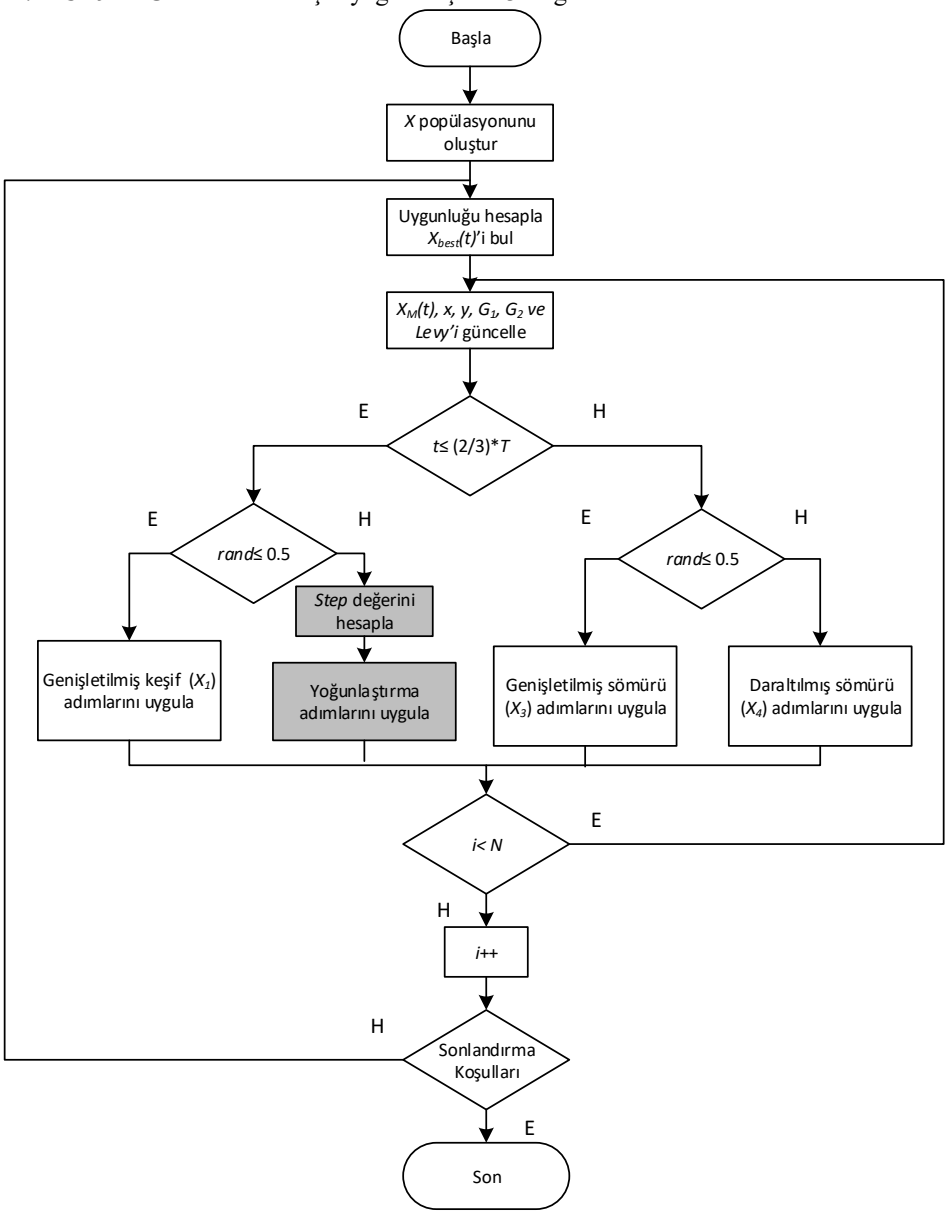

**Şekil 2.** TAA'nın akış diyagramı

## 3. Hibrit Kaya Kartalı-Tanjant Arama Algoritması

Metasezgisel algoritmaların yüksek keşif ve sömürü yeteneklerine sahip olması gerekmektedir. Keşif aşamasının amacı, arama alanını iyi keşfetmek ve en umut verici adayları bulmaktır. Sömürü aşaması ise arama sürecini popülasyondaki en iyi mevcut çözüme yönlendirmek için kullanılmaktadır. Bir metasezgisel yöntemin doğruluğu ve yakınsama hızı, kullanım ve keşif performansını uygun şekilde dengeleyerek geliştirilebilir. Sömürü yeteneği güçlü ancak keşif yeteneği zayıf olan bir algoritmayla, keşif yeteneği güçlü ancak sömürü yeteneği zayıf olan bir algoritmayı hibritleştirerek, bu iki algoritmadan daha güçlü yeni bir hibrit algoritma elde edilebilir. İlk algoritmanın güçlü sömürü yeteneği ile ikinci algoritmanın güçlü olan keşif yeteneği birleştirilerek daha güçlü yeni bir algoritma hedeflenmektedir. Böylece, iki yöntemin hibridizasyonu, her bir yöntemin gücünü tek bir yaklaşımda birleştirmek ve her bir yöntemin dezavantajlarını ortadan kaldırarak avantajlarından yararlanmak için kullanılabilir.

KKO yönteminde keşif aşaması iterasyonun 2/3 oranında uygulanırken, sömürü aşaması yetersiz kalmıştır. Sömürü kabiliyetine daha fazla yer vermek için TAA'nın etkili olan yoğunlaştırma aşaması KKO'nun daraltılmış

keşif aşaması yerine uygulanmıştır. Dolayısıyla hibrit KKO-TAA'nın sömürü aşaması güçlendirilerek optimum çözüme daha erken ulaşması hedeflenmektedir. Bu nedenle bu makalede, KKO'nun daraltılmış keşif aşamasına TAA'nın yoğunlaştırma adımları entegre edilerek hibrit KKO-TAA önerilmiştir. Önerilen bu hibrit algoritma, sömürü ile keşif aşamasını uygun şekilde dengeleyerek, yerel optimuma takılıp kalmadan global çözümü daha hızlı bulmaktadır. Hibrit KKO-TAA'nın akış diyagramı Şekil 3'te gösterilmektedir.




**Şekil 3.** Hibrit KKO-TAA'nın akış diyagramı

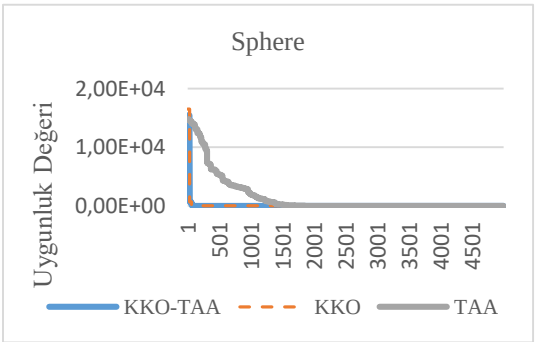
## 4. Deneysel Sonuçlar

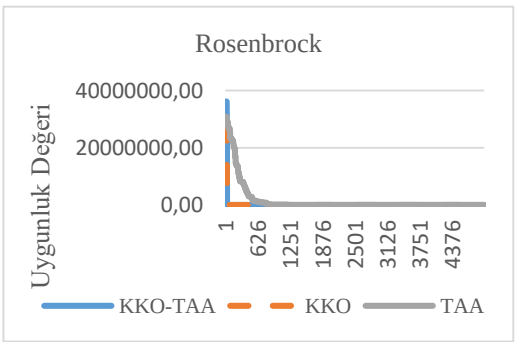
KKO, TAA ve KKO-TAA algoritmalarının performanslarını karşılaştırmak için, kalite testi fonksiyonlarından iki tane unimodal (Sphere ve Rosenbrock), iki tane multimodal (Schwefel ve Levy Function) ve iki tane de fixed-dimension multimodal (Foxholes ve Kowalik) kullanılmıştır. Kullanılan kalite testi fonksiyonlarında problem boyutu Sphere, Rosenbrock Schwefel ve Levy Function'da 10, Foxholes'da 2 ve Kowalik'te ise 4 olarak seçilmiştir. Bu altı adet test fonksiyonunun denklemleri ile parametre ve minimum değerleri Tablo 1'de gösterilmektedir.

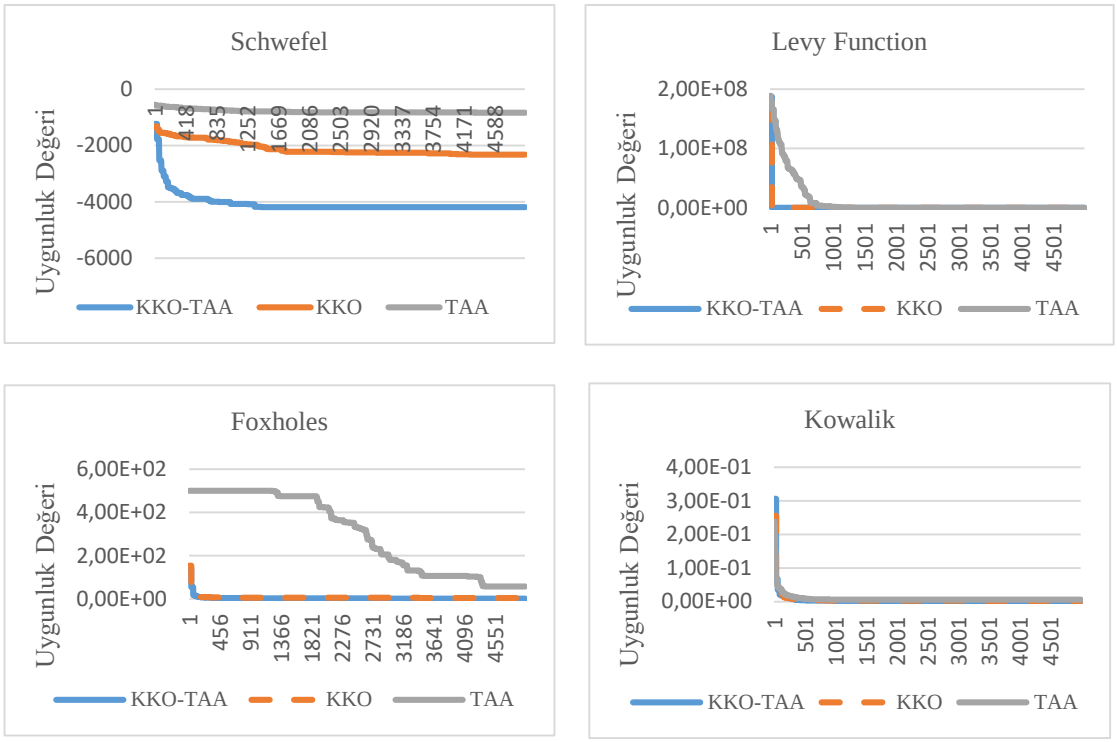
**Tablo 1.** Test fonksiyonları

| Adı | Fonksiyon Denklemi | Parametreler | MİN |
|---|---|---|---|
| **Sphere** | $f(x) = \sum_{i=1}^{N} x_i^2$ | $-100 < x_i < 100$ | 0 |
| **Rosenbrock** | $f(x) = \sum_{i=1}^{N-1} [100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2]$ | $-30 < x_i < 30$ | 0 |
| **Schwefel** | $f(x) = \sum_{i=1}^{N} (-x_i \sin(\sqrt{\lvert x_i \rvert}))$ | $-500 < x_i < 500$ | $-418.9829 * \mathrm{N}$ |
| **Levy Function** | $f(x) = 0.1\Big(sin^2(3\pi x_1) + \sum_{i=1}^{N}(x_i - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_i - 1)^2[1 + sin^2(2\pi x_i)]\Big) + \sum_{i=1}^{N} u(x_i, 5,100,4)$ | $-50 < x_i < 50$ | 0 |
| **Foxholes** | $f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{i,j})^6}\right)^{-1}$ | $-65.536 < x_i < 65.536$ | 1 |
| **Kowalik** | $f(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$ | $-5 < x_i < 5$ | 0.00030 |

Bu çalışmada tüm deneyler için, kullanılan algoritmaların başlangıç popülasyon boyutu 30 olarak alınmıştır. Algoritmaların sonlandırma şartı olarak fonksiyon değerlendirme sayısı kullanılmıştır. Buna göre fonksiyon değerlendirme sayısı 5000'e ulaşınca algoritma sonlandırılmaktadır. Her algoritma her bir test fonksiyonu için 20 defa çalıştırılmıştır ve elde edilen ortalama sonuçlar sunulmuştur. Şekil 4'te Hibrit KKO-TAA, KKO ve TAA'nın Sphere, Rosenbrock, Schwefel, Levy Function, Foxholes ve Kowalik test fonksiyonlarındaki deney sonuçlarının ortalama uygunluk değeri/fonksiyon değerlendirme sayısı değişimi görülmektedir.






**Şekil 4.** Fonksiyon değerlendirme sayısına göre uygunluk değerinin değişimi

**Tablo 2.** Tüm deneyler için istatistiksel sonuçlar

| | | | KKO-TAA | KKO | TAA |
|---|---|---|---|---|---|
| **Unimodal** | Sphere | En İyi | **7.90361E-82** | 1.68053E-76 | 5.36E-16 |
| | | En Kötü | 8.14708E-72 | 1.05325E-16 | **3.85E-07** |
| | | Ortalama | **4.15992E-73** | 5.26623E-18 | 3.80E-08 |
| | | Std. Sapma | 1.77377E-72 | 2.2955E-17 | 9.43E-08 |
| | Rosenbrock | En İyi | **0.000089** | 0.000187 | 0.093677 |
| | | En Kötü | 0.034303 | 0.086325 | **134.753997** |
| | | Ortalama | **0.007296** | 0.008634 | 16.691922 |
| | | Std. Sapma | 0.010006039 | 0.018418326 | 31.22474037 |
| **Multimodal** | Schwefel | En İyi | **-4189.828869** | -3005.438440 | -837.965775 |
| | | En Kötü | -4186.338211 | -1888.326106 | **-837.953101** |
| | | Ortalama | **-4189.536068** | -2329.271288 | -837.964062 |
| | | Std. Sapma | 0.75282628 | 445.8335264 | 0.00357791 |
| | Levy Function | En İyi | **1.7788E-07** | 5.75208E-07 | 4.61E-07 |
| | | En Kötü | 7.80114E-05 | 0.000129198 | **4.40E-02** |
| | | Ortalama | **1.05E-05** | 2.98795E-05 | 1.58E-02 |
| | | Std. Sapma | 1.96522E-05 | 3.41083E-05 | 1.35E-02 |
| **Fixed-Dimension Multimodal** | Foxholes | En İyi | **0.998003879** | 0.998008573 | 9.98E-01 |
| | | En Kötü | 2.982165788 | 12.67050581 | **5.00E+02** |
| | | Ortalama | **1.32E+00** | 4.198826196 | 5.77E+01 |
| | | Std. Sapma | 0.514987142 | 4.163971672 | 1.48E+02 |
| | Kowalik | En İyi | **3.39E-04** | 4.38E-04 | 6.32E-04 |
| | | En Kötü | 8.28E-04 | 1.22E-02 | **2.04E-02** |
| | | Ortalama | **5.27E-04** | 2.38E-03 | 5.75E-03 |
| | | Std. Sapma | 0.000145119 | 2.83E-03 | 8.44E-03 |

Şekil 4 incelendiğinde, Sphere ve Rosenbrock unimodal test fonksiyonları için, hibrit KKO-TAA ile KKO'nun yakınsama hızlarının birbirine yakın oldukları ve en iyi çözüme TAA'ya göre çok daha erken ulaştıkları görülmektedir. Schwefel için grafiğe bakıldığında, yakınsama hızları üç algoritma için de benzerdir, fakat KKO-TAA'nın, KKO ve TAA'ya göre oldukça iyi sonuçlar verdiği görülmektedir. Levy Function ve Foxholes test fonksiyonları için bakıldığında, KKO-TAA ile KKO optimum çözüme daha erken ulaşırken TAA ise daha geç ulaşmaktadır. Kowalik test fonksiyonunda ise üç algoritmanın da yakınsama hızları birbirine yakındır.

KKO-TAA, KKO ve TAA'nın her bir test fonksiyonu için 20'şer defa çalıştırılıp ortalaması alındıktan sonra elde edilen en iyi, en kötü, ortalama ve standart sapma değerleri Tablo 2'de gösterilmektedir. Tabloya bakıldığında, bütün test fonksiyonları için en iyi sonuçları KKO-TAA algoritması vermektedir. En kötü sonuçlar ise TAA'dan elde edilmiştir. Bütün test fonksiyonları için ortalama değerine bakılacak olunursa, en iyi ortalama değerleri hibrit KKO-TAA'dan elde edilmiştir. Standart sapma değerleri incelendiğinde, kullanılan altı test fonksiyonu için de minimum değeri veren KKO-TAA olmuştur.

## 5. Sonuçlar

Metasezgisel algoritmalar, hesaplama gücünün iyi ve dönüşümlerinin kolay olmasından dolayı, son yıllarda oldukça popüler yöntemler haline gelmiştir. Ancak tüm problemler için en iyi çözümü veren bir algoritma bulunmamaktadır. Bu nedenle, yeni metasezgisel algoritmalar önerilmeye veya var olan algoritmalar geliştirilmeye devam edilmektedir. Metasezgisel algoritmalar keşif ve sömürü yeteneklerine sahip olmaktadır. Bazı metasezgisel algoritmalarda bu iki yetenekten biri çok iyi çalışabilmekteyken diğeri yetersiz kalabilmektedir. İki metasezgisel algoritmanın güçlü yanları birleştirilerek daha etkin yeni bir hibrit algoritma elde edilebilmektedir. Bu çalışmada KKO'nun sömürü kabiliyetini arttırmak için daraltılmış keşif aşaması yerine, TAA'nın yoğunlaştırma aşaması entegre edilerek yeni bir hibrit yöntem olan KKO-TAA önerilmiştir.

KKO-TAA, KKO ve TAA'nın performanslarını karşılaştırmak için, iki tane unimodal, iki tane multimodal ve iki tane de fixed-dimension multimodal kalite testi fonksiyonları kullanılmıştır. 20'şer defa çalıştırılarak elde edilen ortalama sonuçlar incelendiğinde; hibrit KKO-TAA'nın, KKO ve TAA'ya göre daha erken iyi çözümlere ulaştığı görülmüştür. Tüm test fonksiyonlarında En İyi, Ortalama ve Standart Sapma kriterleri için en iyi değerler KKO-TAA tarafından bulunmuştur. Deneysel sonuçlar hibrit KKO-TAA'nın global optimizasyon için etkili bir yöntem olduğunu göstermektedir. İleriki çalışmalarda, bu algoritmanın farklı versiyonlarının önerilip çok amaçlı hale getirilerek gerçek dünya problemlerinde kullanılması planlanmaktadır. Hibrit KKO-TAA'nın doğruluğunu ve arama gücünü arttırmak için farklı arama yöntemleri önerilebilir. Ayrıca gelecekteki çalışmalar için bu algoritmanın dağıtık ve paralel versiyonları da geliştirilebilinir.

## Kaynaklar


[1] Murty KG. Optimization Models for Decision Making. Internet Edition, Chapter 1: Models for Decision Making, 2003; 1-18.

[2] Güden H, Vakvak B, Özkan BE, Altıparmak F ve Dengiz B. Genel Amaçlı Arama Algoritmaları ile Benzetim Eniyilemesi: En İyi Kanban Sayısının Bulunması. Endüstri Mühendisliği Dergisi, 2005; 16: 2-15.

[3] Akyol S. Güncel Akıllı Optimizasyon Algoritmalarıyla Duygu Sınıflandırılması. Doktora Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ, 2018.

[4] Akyol S. Güncel Sürü Zekâsı Algoritmalarıyla Sınıflandırma Kurallarının Keşfi. Yüksek Lisans Tezi, Tunceli Üniversitesi, Fen Bilimleri Enstitüsü, Tunceli, 2013.

[5] Alataş B. Kaotik Haritalı Parçacık Sürü Optimizasyon Algoritmaları Geliştirme. Doktora Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ, 2007.

[6] Cura T. Modern Sezgisel Teknikler ve Uygulamaları. Papatya Yayıncılık Eğitim, 2008.

[7] Karaboğa D. Yapay Zeka Optimizasyon Algoritmaları. Nobel Yayın Dağıtım, 2011.

[8] Holland J H. Adaption in Natural and Artificial Systems, University of Michigan Pres, Ann Arbor, MI, 1975.

[9] Storn R. and Price K. Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley, 1995.

[10] Dorigo M, Maniezzo V, and Colorni A. The Ant System: An Autocatalytic Optimizing Process. Tech. Rep. No. 91- 016, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1991.

[11] Atashpaz-Gargari E and Lucas C. Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition. IEEE Congress on Evolutionary Computation, CEC 2007; 2007; 4661-4667.

[12] Borji A. A New Global Optimization Algorithm Inspired by Parliamentart Political Competitions. Lecture Notes in Computer Science, 2007; 4827/2007: 61-71.

[13] Alataş B. ACROA: Artificial Chemical Reaction Optimization Algorithm for Global Optimization. Expert Systems with Applications, 2011; 38(10): 13170-13180.

[14] Geem ZW, Kim JH and Loganathan GV. A New Heuristic Optimization Algorithm: Harmony Search. Simulation, 2001; 76, 60-68.
[15] Rashedi E, Nezamabadi-pour H and Saryazdi S. GSA: A Gravitational Search Algorithm. IEEE Congress on Information Sciences; 2009; 179: 2232-2248.
[16] Shah-Hosseini H. The Intelligent Water Drops Algorithm: A Nature-Inspired Swarm-Based Optimization Algorithm. International Journal of Bio-Inspired Computation, 2009; 1: 71-79.
[17] Kennedy J and Eberhart RC. Particle Swarm Optimization. IEEE International Conference on Neural Networks; 1995; Piscataway, NJ, 1942-1948.
[18] Chu SC, Tsai PW and Pan JS. Cat Swarm Optimization. 9th Pacific Rim International Conference on Artificial Intelligence, LNAI; 2006. 4099: 854-858.
[19] Kashan AH. League Championship Algorithm: A New Algorithm For Numerical Function Optimization. IEEE International Conference of Soft Computing and Pattern Recognition (SoCPAR); 2009. 43–48.
[20] Salem SA. BOA: A Novel Optimization Algorithm. IEEE 2012 International Conference on Engineering and Technology (ICET); 2012. 1(5).
[21] Maniezzo V, Stützle T and VOß S. Matheuristics: Hybridizing Metaheuristics and Mathematical Programming. 10, Springer, New York, 2009.
[22] Jin X, Reynolds RG. Using knowledge-based evolutionary computation to solve nonlinear constraint optimization problems: a cultural algorithm approach. 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406); 1999, July. 1672-1678. IEEE.
[23] Akyol S, Alatas B. Sentiment classification within online social media using whale optimization algorithm and social impact theory based optimization. Physica A: Statistical Mechanics and its Applications, 2020; 540: 123094.
[24] Akyol S, Alatas B. Plant intelligence based metaheuristic optimization algorithms. Artificial Intelligence Review, 2017; 47(4): 417-462.
[25] Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, and Gandomi AH. Aquila Optimizer: A Novel Meta-Heuristic Optimization Algorithm, Computers & Industrial Engineering, 2021; 157: 107250.
[26] Layeb A. The Tangent Search Algorithm for Solving Optimization Problems. arXiv preprint arXiv:2104.02559, 2021.
[27] Carnie SK. Food Habits of Nesting Golden Eagles in The Coast Ranges of California, The Condor, 1954; 56 (1): 3-12.
[28] Meinertzhagen R. How Do Larger Raptorial Birds Hunt Their Prey. Ibis, 1940; 4 (3): 530-535.
[29] Dekker D. Hunting Behavior of Golden Eagles, Aquila-Chrysaetos, Migrating in Southwestern Alberta, 1985.
[30] Watson J. The Golden Eagle. Bloomsbury Publishing, 2010.