Fırat Üniv. Müh. Bil. Dergisi
29(1), 225-236, 2017

Science and Eng. J of Fırat Univ.
29(1), 225-236, 2017

Optimizasyon Problemlerinin Çözümünde Sinüs Kosinüs Algoritması (SKA)'nın Kullanılması

Gökhan DEMİR, Erkan TANYILDIZI
Fırat Üniversitesi Teknoloji Fakültesi Yazılım Mühendisliği Bölümü, Elazığ
gokhan.demir.dg@gmail.com

(Geliş/Received: 17.02.2017; Kabul/Accepted: 25.02.2017)

Özet

Bu çalışmada; optimizasyon problemlerinin çözümü için yeni bir popülasyon tabanlı optimizasyon algoritması olan Sinüs Kosinüs Algoritması (SKA) tanıtılmaktadır. SKA, sinüs ve kosinüs fonksiyonlarına dayalı bir matematiksel model ile içe veya dışa doğru dalgalanma yaparak en iyi çözümü bulmaya çalışır. Birkaç rastgele ve adaptif değişken de arama uzayının keşfini ve sömürülmesini garanti etmek için algoritmaya entegre edilmiştir. Ayrıca bu çalışma kapsamında SKA'nın performansını değerlendirmek için literatürde yaygın olarak kullanılan kalite testi fonksiyonları üzerinde testler yapılmıştır. Algoritma, mühendislik tasarım problemlerinden biri olan germe / sıkıştırma yay tasarım probleminin çözümünde kullanılarak algoritmanın kısıtlı problemler üzerindeki etkinliği de test edilmiştir. Test sonuçlarından SKA'nın karşılaştırılan metasezgisel algoritmalara göre daha hızlı sonuç verdiği görülmektedir. SKA'nın henüz yeni bir algoritma olması ve az sayıda parametre içermesi gelecek çalışmalarda algoritmada iyileştirmeler yapılmasına olanak tanımaktadır.

Anahtar Kelimeler: Sinüs Kosinüs Algoritması, Optimizasyon, Metasezgisel Algoritmalar, Yapay Zeka.

The Use of Sine Cosine Algorithm (SCA) in Solution of Optimization Problems

Abstract

In this study, Sine Cosine Algorithm (SCA), a new population-based optimization algorithm, is introduced to solve optimization problems. SCA tries to find the best solution by fluctuating inwards or outwards with a mathematical model based on sine and cosine functions. Several random and adaptive variables have also been integrated into the algorithm to guarantee exploration and exploitation of search space. In addition, benchmark functions commonly used in the literature have been tested to evaluate the performance of the SCA. The algorithm is also used in the solution of the tension / compression spring design problem, which is one of the engineering design problems, and the effectiveness of the algorithm on the constrained optimization problems has been tested. From the test results, it is seen that SCA is faster than the metaheuristic algorithms which are compared. Since SCA is a new algorithm and it contains few parameters, it allows the algorithm to be improved in future studies.

Keywords: Sine Cosine Algorithm, Optimization, Metaheuristic Algorithms, Artificial Intelligence.

1. Giriş

Optimizasyon, bir problemin global optimumum çözümünü belirli koşullar altında arama sürecidir. Optimizasyon algoritmaları temel olarak deterministik ve stokastik tabanlı algoritmalar olmak üzere iki sınıfa ayrılır. Deterministik yöntemler de kendi içinde hesaplamalı yöntemler ve gradyan tabanlı yöntemler olarak sınıflandırılır. Hesaplamalı yöntemler gradyan fonksiyonu hesaplamalarına ihtiyaç duymazlar ancak oldukça yavaş ve etkisiz yöntemlerdir. Gradyan tabanlı yöntemler amaç fonksiyonunun eğimini veya türevini kullanırlar. Ancak bu yöntemler amaç fonksiyonunun düz ve pürüzsüz olmadığı durumlarda global optimum noktaya yakınsamayı garanti etmezler. Deterministik yöntemler ile yüksek boyutlu, çok modlu veya türevlenemeyen problemleri çözmek oldukça zor hatta imkansızdır [1].

Mühendislik, ekonomi ve işletme gibi pek çok bilim dalında, klasik yöntemlerle makul bir

sürede veya kesin olarak çözülemeyen karmaşık optimizasyon problemleri ortaya çıkmıştır. Doğa, bu gibi karmaşık optimizasyon problemlerini çözmede kullanılabilecek yapay hesaplama yöntemleri tasarlamak için mekanizmalar, ilkeler ve kavramlar gibi kaynaklar sağlar [2]. Son yıllarda araştırmacılar, problemlerin çözümü için doğanın belirli olgu veya davranışlarını taklit eden doğadan esinlenmiş birçok metasezgisel algoritma geliştirdiler.

Metasezgisel yöntemler gradyan gilgisine ihtiyaç duymazlar ve amaç fonksiyonu defalarca çağırarak global optimumu ararlar. Bu algoritmalar arama uzayını daraltırlar ve etkili bir şekilde çözüm bulmaya çalışırlar. Geliştirilen algoritmalar arasındaki temel fark, keşif (global arama yeteneği) ile sömürü arasındaki denge (optimal çözüm çevresinde yerel arama yeteneği) yaklaşımında yatmaktadır [1].

Doğadan ilham alan algoritmaların büyük kısmı biyolojiden ilham almıştır. Biyoloji tabanlı algoritmalar sürü davranışlarını doğrudan kullanmazlar. Genetik Algoritma [3] doğal seçilim sürecinden ilham alır. Diferansiyel Gelişim Algoritması [4] işleyiş ve operatörleri itibariyle genetik algoritmaya dayanır. Eko-İlhamlı Evrimsel Algoritma [5], habitatlar, ekolojik ilişkiler ve ekolojik süksesyon gibi ekolojik kavramlara dayanır. İstilacı Yabani Ot Optimizasyon Algoritması [6], istilacı yabani otlardan ilham alır. Atmosfer Bulutları Modeli Optimizasyon Algoritması [7] bulutların hareketini taklit eder. Termit Koloni Optimizasyon Algoritması [8] termitlerin akıllı davranışlarından esinlenen popülasyon tabanlı bir optimizasyon tekniğidir.

Biyolojiden ilham alan algoritmalar arasında, sürü zekasından ilham alınarak özel bir algoritma sınıfı geliştirilmiştir. Sürü zekası, bazı kuralları takip eden birden fazla ajanın ortaklaşa hareket etmesi sonucu ortaya çıkan davranışlar ile ilgilenir. Parçacık Sürü Optimizasyonu [9], kuş ve balık sürülerinin hareketlerinden ilham alır. Karınca Koloni Algoritması [10] karıncaların yiyecek arama için en kısa yolu bulma davranışını taklit eder. Yapay Arı Koloni Algoritması [11], bal arısı sürüsünün zeki davranışlarını taklit eder. Kurt Arama Algoritması [12], kurtların yiyecek arama ve düşmanlarından kaçarak hayatta kalma davranışlarını modeller. Guguk Kuşu Arama Algoritması [13], bazı guguk kuşu türlerinin kuluçka davranışlarıyla birlikte bazı meyve sineklerinin ve kuşların Levy uçuş davranışlarını taklit eder. Gri Kurt Optimizasyon Algoritması [14], doğadaki gri kurtların liderlik hiyerarşisini ve avlanma mekanizmasını taklit eder. Balina Optimizasyon Algoritması [15] kambur balinaların kabarcık ağı avlanma stratejisine dayanır.

Fizik ve kimyadan ilham alan algoritmalar elektrik yükleri, yerçekimi, nehir sistemleri vb. içeren belli fiziksel ve kimyasal kanunları taklit eder. Büyük Patlama - Büyük Çöküş Algoritması [16], Büyük Patlama ve Büyük Çöküş teorilerinden ilham alır. Merkezi Kuvvet Optimizasyonu [17], yerçekimi kinematiği metaforuna dayanır. Yerçekimi Arama Algoritması [18], yerçekimi ve kütle etkileşimi yasalarına dayanır. Su Döngüsü Algoritması [19] su döngüsü sürecinden ve nehirlerin, akarsuların denize akışından ilham alır. Spiral Optimizasyon Algoritması [20] doğadaki spiral olguyu taklit eder.

Ayrıca spor kavramlarından ilham alınarak geliştirilen spor tabanlı metasezgisel algoritmalar da günümüzde oldukça popülerdir. Bu algoritmalarda insanların davranışları veya spor ligleri simüle edilmektedir. Lig Şampiyonası Algoritması [21], yapay bir ligde yapay takımların oynadığı bir şampiyona ortamını taklit etmeye çalışır. Futbol Lig Müsabakası Algoritması [22], futbol liglerindeki takımlar arasındaki müsabakalardan ve oyuncular arasındaki mücadelelerden ilham alır. Altın Top Algoritması [23] futbol kavramlarına dayanır.

Tüm metasezgisel algoritmalar matematiksel model kullanmasına rağmen matematiksel kurallardan ilham alan algoritma sayısı oldukça azdır. Matematikten ilham alan az sayıdaki algoritmalardan biri olan Baz Optimizasyon Algoritması [24] çözümleri optimum noktaya yönlendiren bir yer değiştirme parametresi ile birlikte temel aritmetik operatörlerin kombinasyonunu kullanır.

Bu çalışmada tanıtılan Sinüs Kosinüs Algoritması (SKA) Seyedali Mirjalili tarafından sinüs ve kosinüs fonksiyonlarına dayalı bir matematiksel model kullanılarak optimizasyon problemlerinin çözümü için geliştirilen matematik tabanlı metasezgisel bir algoritmadır [25].

2. Sinüs Kosinüs Algoritması (SKA)

Popülasyon tabanlı optimizasyon teknikleri genel olarak rastgele çözüm kümesi ile optimizasyon sürecini başlatır. Bu rastgele küme uygunluk fonksiyonu ile defalarca değerlendirilir ve optimizasyon tekniğinin temeli olan kural seti ile iyileştirilir. Stokastik tabanlı optimizasyon teknikleri, optimizasyon probleminin optimumunu stokastik olarak aradığından tek bir adımda çözümü bulmayı garanti etmez. Ancak yeterli sayıda rastgele çözüm ve optimizasyon adımları ile global optimumun bulunma olasılığı artar.

Popülasyon tabanlı stokastik algoritmalar arasındaki farklar her ne olursa olsun, optimizasyon süreci keşif ve sömürü olmak üzere ortak iki aşamadan oluşur. Optimizasyon algoritması arama uzayının umut verici alanlarını bulmak için yüksek rastgelelik oranı ile çözüm kümesindeki rastgele çözümleri hızlıca bir araya getirir. Bununla birlikte, sömürü aşamasında rasgele çözümlerde kademeli olarak değişiklikler yapılır ve rastgele varyasyonlar keşif aşamasındakinden çok daha azdır.

Bu çalışmada, her iki aşama için önerilen pozisyon güncelleme denklemleri Denklem 1 ve Denklem 2'deki gibidir:

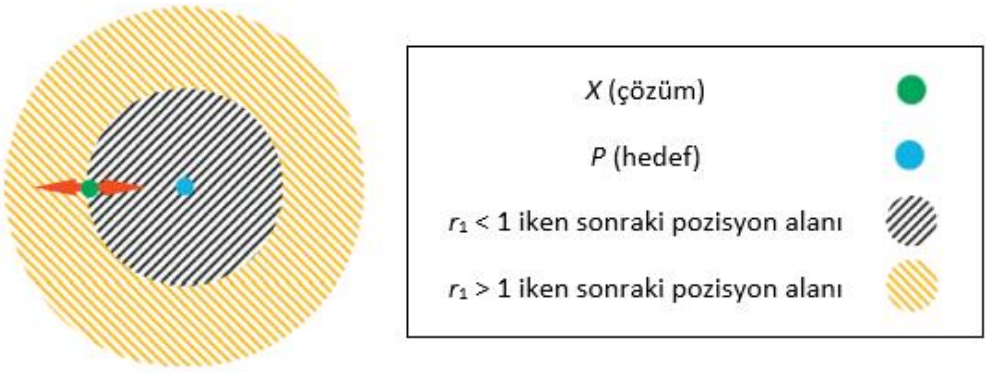
$$X_i^{t+1} = x_i^t + r_1 \times \sin(r_2) \times \left| r_3 P_i^t - X_i^t \right| \quad (1)$$

$$X_i^{t+1} = x_i^t + r_1 \times \cos(r_2) \times \left| r_3 P_i^t - X_{ii}^t \right| \quad (2)$$

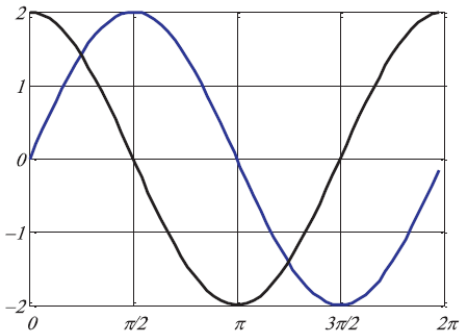
x_i^t: i. boyutun t. iterasyonundaki güncel çözümü.
r_1, r_2, r_3 : Rastgele sayılar.
r_4 : [0, 1] aralığında rastgele bir sayıdır.
P_i : i. boyuttaki hedef noktanın pozisyonu.
|| : Mutlak değer.

Bu iki denklem Denklem 3'teki gibi birlikte kullanılır.

$$X_i^{t+1} = \begin{cases} x_i^t + r_1 \times \sin(r_2) \times \left| r_3 P_i^t - X_i^t \right|, r_4 < 0.5 \\ x_i^t + r_1 \times \cos(r_2) \times \left| r_3 P_i^t - X_{ii}^t \right|, r_4 \geq 0.5 \end{cases} \quad (3)$$



Şekil 1. Denklem 1 ve Denklem 2'deki sinüs ve kosinüsün bir sonraki pozisyon üzerindeki etkileri [25]



Şekil 2. [-2, 2] aralığında sinüs ve kosinüs [25]

Yukarıdaki denklemlerde görüleceği üzere SKA'da; r_1, r_2, r_3, r_4 olmak üzere 4 ana parametre kullanılır.

r_1: Bir sonraki pozisyon bölgesini (veya hareket yönü) belirler.

r_2: Hedefe ulaşmak için, içe doğru ya da dışa doğru ne kadar hareket edileceğini belirler.

r_3: Stokastik ağırlığı rastgele belirler. $r_3 > 1$ olması stokastikliğin önemli olduğunu, $r_3 < 1$ olması ise stokastikliğin daha az etkili olduğunu belirtir.

r_4: Denklemdeki sinüs ve kosinüs bileşenleri arasındaki geçişi sağlar.

Bu yöntemde sinüs ve kosinüs kullanıldığından algoritma Sinüs Kosinüs Algoritması (SKA) olarak adlandırılmaktadır. Denklemlerdeki sinüs ve kosinüs fonksiyonlarının etkileri Şekil 1'de gösterilmektedir. Bu şekil, önerilen denklemlerin arama uzayındaki iki çözüm arasındaki bir alanın nasıl belirlendiğini gösterir. Şekil 1'de iki boyutlu bir model gösterilmesine rağmen bu denklemin daha yüksek boyutlar için genişletilebileceği unutulmamalıdır. Sinüs ve kosinüs fonksiyonlarının döngüsel deseni diğer bir çözüm etrafında yeniden konumlandırılabilir bir çözüm sağlar. Bu durum iki çözüm arasındaki tanımlı alanı sömürmeyi garanti eder. Arama alanını keşfetmek için, ilgili hedeflere karşılık gelen alanlar arasındaki boşluk dışında da çözümler aramak gerekir. Bu, Şekil 2'de gösterilen sinüs ve kosinüs fonksiyonlarının değişim aralığı ile elde edilebilir.

Sinüs ve kosinüs fonksiyonlarının [-2, 2] aralığındaki etkilerinin kavramsal modeli Şekil 3'te gösterilmiştir. Bu şekil, sinüs ve kosinüs fonksiyon aralığındaki değişimi ve başka bir çözüm ile kendisi arasındaki alanda pozisyonu içe doğru veya dışa doğru güncellemek için nasıl bir çözüm bulunması gerektiğini göstermektedir. Rastgele konum, içeride veya dışarıda olmak üzere Denklem 3'teki gibi [0, 2π] aralığında rastgele bir r_2 sayısı belirlenmesiyle elde edilir. Bu nedenle bu mekanizma arama uzayında keşif ve sömürüyü garanti eder.

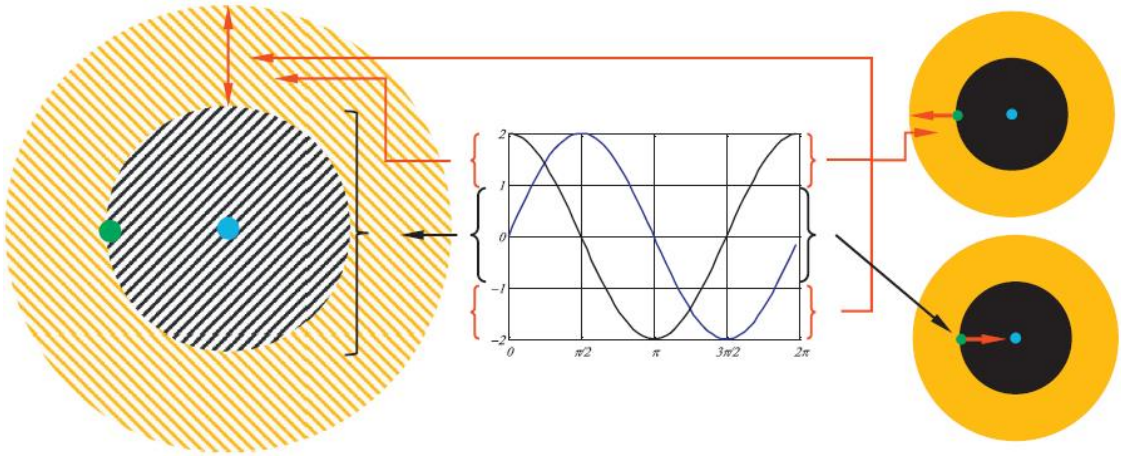
Algoritma, arama uzayının umut verici alanlarını bulmak ve global optimuma yakınsamak için keşif ve sömürüyü dengelemelidir. Keşif ve sömürüyü dengelemek amacıyla, Denklem 4 kullanılarak Denklem 1, 2 ve 3'teki sinüs ve kosinüs aralığı adaptif olarak değiştirilir:

$$r_1 = a - t\frac{a}{T} \tag{4}$$

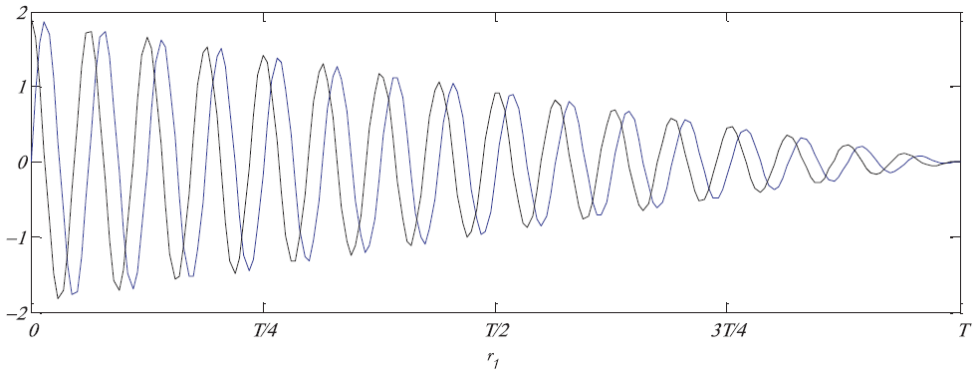
t, mevcut iterasyondur. *T* maksimum iterasyon sayısıdır ve *a* bir sabittir.

Şekil 4, bu denklemin iterasyonlar boyunca sinüs ve kosinüs fonksiyon aralığının azalışını gösterir. Şekil 3 ve Şekil 4'ten tahmin edilebileceği gibi, SKA'nın sinüs ve kosinüs fonksiyonları, arama uzayını (1, 2] ve [-2, 1) aralığında araştırır. Ancak, algoritma [-1, 1] arama alanını sömürür.

Son olarak SKA'nın sözde kodu Şekil 5'te sunulmuştur. SKA optimizasyon sürecinde, rastgele çözüm kümesi ile başlamaktadır. Algoritma, elde edilen en iyi çözümü kaydeder, onu hedef noktası olarak atar ve bu çözüme göre diğer çözümleri günceller. Bu arada, sinüs ve kosinüs fonksiyon aralığı sömürüyü garanti etmek için iterasyon sayısı arttıkça güncellenir. Algoritma varsayılan olarak iterasyon sayacının maksimum iterasyon sayısını aşması ile optimizasyon sürecini sonlandırır. Bununla birlikte sonlandırma koşulu olarak maksimum fonksiyon değerlendirme sayısı veya elde edilen global optimumun doğruluğu dikkate alınabilir.



Şekil 3. [-2, 2] aralığında sinüs ve kosinüs; bir çözümün hedefin etrafında veya ötesinde dolaşmasına verir [25]



Şekil 4. Sinüs ve kosinüs aralığı için azalan model (a = 3) [25]

Başlat : Arama ajanları kümesini (çözümler) (X) başlat.

Do

Değerlendir: Her arama ajanını uygunluk fonksiyonu ile değerlendir.

Güncelle : r_1, r_2, r_3 ve r_4 güncelle.

Güncelle : Denklem 3'ü kullanarak arama ajanlarının pozisyonlarını güncelle.

While (t < iterasyon sayısı)

Return : Şimdiye kadar global optimum olarak elde edilen en iyi çözümü döndür.

Şekil 5. SKA'nın sözde kodu

SKA'nın belirtilen operatörler ile optimizasyon problemlerinin global optimumunu belirlemesi teorik olarak aşağıdaki nedenlerden dolayı mümkündür:

- SKA belirli bir problem için rastgele çözümler dizisi oluşturur ve geliştirir, bu nedenle bireysel tabanlı algoritmalara kıyasla yüksek keşif ve yerel optimumdan kaçınmayı sağlar.
- Sinüs ve kosinüs fonksiyonları 1'den büyük veya -1'den düşük bir değer döndürdüğünde arama uzayının farklı bölgeleri keşfedilir.
- Arama uzayının umut verici alanları sinüs ve kosinüs -1 ile 1 aralığında değer döndürdüğünde sömürülür.
- SKA, sinüs ve kosinüs fonksiyon aralığının adaptif kullanımı ile keşiften sömürüye kolayca geçer.
- Global optimumun en iyi yakınsaması hedef nokta olarak bir değişkende saklanır ve optimizasyon sırasında asla kaybolmaz.
- Çözümler, pozisyonlarını elde edilen en iyi çözümler etrafında güncellediğinden optimizasyon boyunca arama uzayının en iyi bölgelerine doğru bir eğilim vardır.
- Önerilen algoritma, optimizasyon problemini kara kutu olarak gördüğünden farklı alanlardaki problemlere kolayca uygulanabilir.

SKA'nın sözde kodu Şekil 5'te gösterilmektedir.

3. Deney ve Sonuçlar

SKA'nın performansını değerlendirmek için literatürde yaygın olarak kullanılan 23 kalite testi fonksiyonu üzerinde testler yapılmıştır. Bu fonksiyonlar tek modlu fonksiyonlar, çok modlu fonksiyonlar ve sabit boyutlu çok modlu fonksiyonlar olmak üzere üç farklı kategoride incelenebilir. Tablo 1'de gösterilen F1 - F7 arasındaki fonksiyonlar tek modlu fonksiyonlardır ve bir tek global optimuma

sahiptirler. Bu fonksiyonlar arama algoritmalarının yakınsama oranını test etmek için tasarlanmıştır. Birden fazla lokal minimuma sahip olan ve bundan dolayı optimize edilmesi oldukça zor olan F8 – F13 arasındaki çok modlu fonksiyonlar Tablo 2'de gösterilmiştir. Çok modlu fonksiyonlarda problem boyutu sayısı arttıkça yerel optimum sayısı da artmaktadır. Bu nedenle bu tür test problemleri optimizasyon algoritmalarının arama kapasitelerini değerlendirmede oldukça önemlidir. Tablo 3'te gösterilen F14 – F23 arasındaki sabit boyutlu çok modlu fonksiyonların çok modlu fonksiyonlardan tek farkı boyutlarının düşük sayıda olmasından dolayı az sayıda yerel minimum içermeleridir. SKA, ilk kez mühendislik tasarım problemlerinden biri olan germe / sıkıştırma yay tasarım probleminin çözümünde kullanılarak algoritmanın kısıtlı problemler üzerindeki etkinliği de test edilmiştir.

SKA iyi bilinen sürü tabanlı algoritmalardan Parçacık Sürü Optimizasyonu (PSO), Karınca Koloni Optimizasyon (KKO) algoritması ve güncel algoritmalar Balina Optimizasyon Algoritması (BOA), Yerçekimi Arama Algoritması (YAA) olmak üzere 4 metasezgisel algoritma ile karşılaştırılmıştır. SKA matematikten ilham alması yönüyle karşılaştırılan algoritmalardan farklıdır. Algoritmaların popülasyon büyüklüğü 30 ve iterasyon sayıları 1000 olarak kabul edilmiştir. Boyutları fonksiyon tanımlarında *Vno* olarak belirtilen her kalite testi fonksiyonu için algoritmalar 30 kez çalıştırılmıştır. Elde edilen ortalama, standart sapma, en iyi sonuç, en kötü sonuç ve ortalama çalışma zamanı istatistiksel sonuçları Tablo 4'te gösterilmektedir.

Kalite testi fonksiyonları sonuçları incelendiğinde SKA'nın optimum sonuca karşılaştırılan diğer tüm algoritmalardan (PSO, BOA, KKO, YAA) çok daha kısa sürede yakınsadığı açıkça görülmektedir. SKA F14, F16, F17 ve F18 kalite testi fonksiyonlarında optimum sonucu elde etmiştir.

Karşılaştırılan algoritmalarda kullanılan parametreler şu şekildedir:

1. PSO: *Atalet ağırlığı* = 1, *Atalet ağırlığı sönüm oranı* = 0.99, *Kişisel öğrenme katsayısı* = 1.5, *Küresel öğrenme katsayısı* = 2.0
2. KKO: *Örnek boyutu* = 40, *Yoğunlaşma faktörü* = 0.5, *Sapma - uzaklık oranı*=1
3. BOA: *b* = 1
4. YAA: *R_norm* = 2, *R_gücü* = 1, *Elitist kontrolü* = 1

Tablo 1. Tek modlu kalite testi fonksiyonlarının tanımı

Fonksiyon	V_{no}	*Aralık*	F_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]	0
$F_2(x) = \sum_{i=1}^{n} x_i^2 \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	[-10, 10]	0
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30	[-100, 100]	0
$F_4(x) = max\{\lvert x_i\rvert, 1 \le i \le n\}$	30	[-100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30, 30]	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[-100, 100]	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1]$	30	[-1.28, 1.28]	0

Tablo 2. Çok modlu kalite testi fonksiyonlarının tanımı

Fonksiyon	V_{no}	*Aralık*	F_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i\rvert})$	30	[-500, 500]	-418.9829 x 5
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12, 5.12]	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[-32, 32]	0

$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{i}\right) + 1$	30	[-600, 600]	0
$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})] + (y_n - 1)^2 + \sum_{i=1}^{n} u(x_i, 10, 100, 4)\}$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m \; x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m \; x_i < -a \end{cases}$	30	[-50, 50]	0
$F_{13}(x) = 0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[-50, 50]	0

Tablo 3. Sabit boyutlu çok modlu kalite testi fonksiyonlarının tanımları

Fonksiyon	V_{no}	*Aralık*	F_{min}
$F_{14}(x) = (\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6})^{-1}$	2	[-65, 65]	1
$F_{15}(x) = \sum_{i=1}^{11}[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	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
$F_{17}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)cosx_1 + 10$	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
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[1, 3]	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	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
$F_{23}(x) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.5363

Tablo 4. Kalite testi fonksiyonları sonuçları (ort: ortalama çözüm, sd: standart sapma, en iyi: en iyi çözüm, en kötü: en kötü çözüm, süre: saniye cinsinden ortalama çalışma süresi, İ: istatistikler)

F	İ	SKA	PSO	BOA	KKO	YAA
F1	*ort*	0.0310	6.5085e-16	1.4109e-154	1.2766	1.0976e-16
	sd	0.0866	1.1800e-15	7.0872e-154	0.8429	3.5117e-17
	en iyi	2.4792e-06	1.0232e-17	3.3634e-166	0.4405	6.6897e-17
	en kötü	0.3945	5.4858e-15	3.8875e-153	4.4251	2.1405e-16
	süre	1.2570	8.3309	4.2849	43.7911	12.0372
F2	*ort*	2.1026e-05	2.1071e-05	5.0254e-102	3.6260e+03	5.2733e-08
	sd	4.3726e-05	1.0554e-04	1.9523e-101	1.9618e+04	1.6126e-08
	en iyi	2.9768e-09	6.9921e-10	1.4491e-114	1.1190	2.7875e-08
	en kötü	2.1087e-04	5.7914e-04	9.2622e-101	1.0750e+05	1.0028e-07
	süre	1.3360	8.5741	4.5383	46.1645	10.7295
F3	*ort*	4.0282e+03	7.6568	1.9940e+04	1.0596e+05	432.9601
	sd	3.2698e+03	5.5415	1.1221e+04	2.9680e+04	155.3030
	en iyi	109.8140	1.0370	3.6202e+03	4.7676e+04	217.7493
	en kötü	1.4489e+04	22.0031	4.3344e+04	1.6262e+05	780.2057
	süre	6.2230	17.1235	9.6729	51.2220	14.0386

	ort	19.9222	0.6941	40.8308	78.1506	1.5515
	sd	11.2356	0.3446	32.4037	9.6495	1.5699
F4	*en iyi*	1.5395	0.2272	0.0519	42.8106	1.0044e-08
	en kötü	42.7265	1.3777	92.0102	92.1774	5.3571
	süre	1.6804	12.2517	5.2079	39.2201	7.3923
	ort	1.0127e+03	41.3337	27.2594	5.0124e+04	36.3915
	sd	2.8445e+03	32.4499	0.6228	4.5712e+04	53.9666
F5	*en iyi*	28.3801	2.3651	26.5891	6.4021e+03	24.5371
	en kötü	1.2227e+04	110.3879	28.7495	1.9159e+05	322.0843
	süre	1.7429	11.0633	4.7569	38.6504	9.5483
	ort	4.5788	2.7841e-15	0.1062	1.0898	0.2000
	sd	0.5852	8.7185e-15	0.1165	0.5781	0.7611
F6	*en iyi*	3.8463	1.7238e-17	0.0094	0.4107	0
	en kötü	6.7212	4.7394e-14	0.4399	3.1241	4.0000
	süre	1.8441	9.5745	4.7322	36.7716	10.1557
	ort	0.0421	0.0148	0.0024	0.1854	0.0647
	sd	0.0528	0.0059	0.0023	0.0757	0.0254
F7	*en iyi*	0.0047	0.0063	5.6430e-05	0.0546	0.0094
	en kötü	0.2775	0.0282	0.0090	0.4039	0.1131
	süre	2.3571	8.5231	6.2950	42.1256	10.7440
	ort	-3.9367e+03	-6.3686e+03	-1.1224e+04	-4.4590e+149	-2.4485e+03
	sd	258.2417	867.5216	1.6018e+03	2.4407e+150	425.4308
F8	*en iyi*	-4.5524e+03	-8.0276e+03	-1.2569e+04	-1.3368e+151	-3.5570e+03
	en kötü	-3.5344e+03	-4.3764e+03	-7.8490e+03	-3.0059e+98	-1.7879e+03
	süre	2.0996	8.6910	5.0582	40.6682	9.4981
	ort	23.4859	44.7399	0	252.9477	27.0629
	sd	32.5145	13.7505	0	18.7779	6.2785
F9	*en iyi*	6.9014e-06	23.8790	0	193.7615	17.9093
	en kötü	168.7336	81.5864	0	276.6462	41.7882
	süre	1.9521	9.8125	4.8869	39.8226	13.6292
	ort	12.6852	1.0915	4.0856e-15	0.6876	7.8281e-09
	sd	9.5190	0.7594	2.3511e-15	0.3804	1.6719e-09
F10	*en iyi*	2.5556e-04	2.7719e-09	8.8818e-16	0.1548	5.7326e-09
	en kötü	20.3227	2.3162	7.9936e-15	1.7909	1.3408e-08
	süre	2.1039	10.4712	5.0623	42.3656	15.1583
	ort	0.3685	0.0240	0.0050	0.9188	8.2013
	sd	0.3339	0.0246	0.0193	0.0798	3.2014
F11	*en iyi*	7.5275e-04	0	0	0.6492	2.6444
	en kötü	0.9431	0.0860	0.0875	1.0283	14.4065
	süre	2.2472	11.9485	5.6325	41.3976	14.9480
	ort	3.1533	0.2319	0.0110	3.2754e+04	0.1608
	sd	6.2240	0.5471	0.0177	7.9615e+04	0.2849
F12	*en iyi*	0.3258	1.0560e-18	0.0012	18.9615	3.5333e-19
	en kötü	33.4485	2.7038	0.0956	3.2040e+05	1.4847
	süre	3.9721	16.8464	4.5318	42.5764	15.5377
	ort	18.5548	0.1020	0.1962	8.1417e+04	0.0033
	sd	65.8292	0.4602	0.1581	1.1285e+05	0.0104
F13	*en iyi*	2.2029	5.5146e-18	0.0398	961.1513	4.1030e-18
	en kötü	365.0753	2.5085	0.7707	3.9818e+05	0.0548
	süre	3.9894	11.5014	7.6812	43.5993	13.4597
	ort	1.5275	4.4098	2.7961	1.3235	3.4221
	sd	0.8922	2.9700	3.2852	1.7829	2.6992
F14	*en iyi*	0.9980	0.9980	0.9980	0.9980	0.9980
	en kötü	2.9821	11.7187	10.7632	10.7632	13.8192
	süre	10.3350	20.2909	10.0869	19.4695	17.3979

	ort	9.7180e-04	3.4190e-04	5.6780e-04	0.0011	0.0023
	sd	3.8543e-04	1.6780e-04	2.7009e-04	3.1570e-04	8.7184e-04
F15	*en iyi*	3.4077e-04	3.0749e-04	3.0836e-04	8.8731e-04	6.2116e-04
	en kötü	0.0015	0.0012	0.0015	0.0019	0.0052
	süre	1.9942	9.4237	2.0983	8.7483	9.6892
	ort	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
	sd	2.5344e-05	6.7752e-16	1.6070e-10	6.7752e-16	4.8787e-16
F16	*en iyi*	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
	en kötü	-1.0315	-1.0316	-1.0316	-1.0316	-1.0316
	süre	1.3675	9.8322	1.4481	5.5078	7.5780
	ort	0.3985	0.3979	0.3979	0.3979	0.3979
	sd	6.0681e-04	0	3.9088e-06	0	0
F17	*en iyi*	0.3979	0.3979	0.3979	0.3979	0.3979
	en kötü	0.4003	0.3979	0.3979	0.3979	0.3979
	süre	1.2613	7.8293	1.3837	5.69707	7.2001
	ort	3.0000	3.0000	3.0000	3.0000	3.0000
	sd	3.2611e-05	1.7954e-15	9.2119e-05	1.3194e-15	3.1250e-15
F18	*en iyi*	3.0000	3.0000	3.0000	3.0000	3.0000
	en kötü	3.0001	3.0000	3.0005	3.0000	3.0000
	süre	1.3814	9.1013	1.4617	5.8880	7.2429
	ort	-3.8551	-3.8370	-3.8609	-3.8628	-3.8628
	sd	0.0024	0.1411	0.0022	2.7101e-15	2.2913e-15
F19	*en iyi*	-3.8618	-3.8628	-3.8628	-3.8628	-3.8628
	en kötü	-3.8519	-3.0898	-3.8549	-3.8628	-3.8628
	Süre	2.3266	10.1333	2.4045	7.7821	9.2076
	ort	-2.8866	-3.2863	-3.2429	-3.2665	-3.3220
	sd	0.4145	0.0554	0.1113	0.0603	1.4889e-15
F20	*en iyi*	-3.2333	-3.3220	-3.3219	-3.3220	-3.3220
	en kötü	-1.2291	-3.2031	-2.9868	-3.2031	-3.3220
	süre	1.8505	9.9590	3.1532	11.7540	8.5954
	ort	-2.2850	-5.4786	-9.0471	-5.6870	-6.4134
	sd	1.8892	3.4558	2.2807	3.7094	3.6846
F21	*en iyi*	-5.9900	-10.1532	-10.1530	-10.1532	-10.1532
	en kötü	-0.4965	-2.6305	-2.6303	-2.6829	-2.6305
	süre	2.6262	11.3522	2.7609	11.0864	8.8242
	ort	-3.5487	-6.9147	-7.7342	-6.8594	-10.4029
	sd	1.6587	3.6103	2.9214	2.5485	1.4378e-15
F22	*en iyi*	-5.2185	-10.4029	-10.4027	-10.4029	-10.4029
	en kötü	-0.9071	-2.7519	-3.7235	-5.0877	-10.4029
	süre	3.1160	11.5451	3.1399	14.0219	9.9985
	ort	-4.0940	-5.9754	-8.5166	-7.2484	-10.2897
	sd	1.7448	3.6315	2.9532	3.0040	1.3511
F23	*en iyi*	-7.9749	-10.5363	-10.5363	-10.5363	-10.5363
	en kötü	-0.9460	-2.4217	-2.4217	-2.4217	-3.1359
	süre	3.5207	12.0961	3.8171	13.2174	11.2024

3.1. Parametrik Olmayan İstatistiksel Analiz Sonuçları

Metasezgisel algoritmaların stokastikliği nedeniyle algoritmalar karşılaştırılırken daha güvenilir sonuçlar elde etmek için istatistiksel testlerden faydalanılmaktadır. Wilcoxon sıra-toplamı testi iki çözüm kümesinin istatistiksel açıdan anlamlı olup olmadığının belirlenmesi / doğrulaması için kullanılabilen parametrik olmayan bir testtir. İstatiksel anlamlılık değeri α = 0.05 olmak üzere SKA ile diğer metasezgisel algoritmalar arasında Wilcoxon sıra toplamı testi yapılarak istatiksel sonuçlar Tablo 5'te

gösterilmiştir. Burada $p < 0.05$ olması durumu karşılaştırılan algoritmalardan elde edilen sonuçlar arasında istatiksel açıdan anlamlı bir farkın olduğunu göstermektedir.

Tablo 5. Wilcoxon sıra toplamı testi karşılaştırma sonuçları

F	SKA / PSO	SKA / BOA	SKA / KKO	SKA / YAA
	p - değeri	*p* - değeri	*p* - değeri	*p* - değeri
F1	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11
F2	9.7917e-05	3.0199e-11	3.0199e-11	8.4848e-09
F3	3.0199e-11	5.5329e-08	3.0199e-11	1.4110e-09
F4	3.0199e-11	0.0933	3.0199e-11	1.4643e-10
F5	1.3250e-04	4.0772e-11	4.9752e-11	5.0723e-10
F6	3.0199e-11	3.0199e-11	3.0199e-11	4.3909e-12
F7	0.0058	1.7769e-10	5.5727e-10	2.6806e-04
F8	3.3384e-11	3.0199e-11	3.0199e-11	3.3384e-11
F9	7.7364e-06	1.2118e-12	3.0199e-11	0.0392
F10	6.8971e-04	1.6711e-11	0.0184	3.0199e-11
F11	2.2780e-05	2.0973e-11	1.2870e-09	3.0199e-11
F12	3.3520e-08	3.0199e-11	4.9752e-11	2.3701e-10
F13	4.9752e-11	3.0199e-11	3.0199e-11	3.0199e-11
F14	4.0001e-05	0.0484	4.5618e-11	3.0036e-04
F15	2.1532e-10	1.8682e-05	0.0594	6.5183e-09
F16	1.2118e-12	3.0199e-11	1.2118e-12	4.0806e-12
F17	1.2118e-12	1.6132e-10	1.2118e-12	1.2118e-12
F18	1.0975e-11	0.1120	2.3657e-12	2.8936e-11
F19	4.5618e-11	2.4386e-09	1.2118e-12	2.3638e-12
F20	2.9744e-11	1.4294e-08	6.0455e-11	8.8675e-12
F21	2.2821e-05	1.3289e-10	5.4561e-05	8.1951e-06
F22	0.0031	9.0632e-08	1.0398e-09	1.2455e-11
F23	0.1892	1.7294e-07	1.1947e-06	2.0585e-10

3.2. Kısıtlı Mühendislik Tasarım Problemlerinin Çözümünde SKA Kullanılması

Gerçek sistemler genellikle kısıtlı problemlerden oluşur. Tasarım sürecinde çözümlerin kullanılabilirliğini sağlayan kısıtlar; Denklem 5'te görüldüğü üzere eşitlik ve eşitsizlik kısıtları olmak üzere iki çeşittir.

Genel olarak kısıtlı optimizasyon problemleri Denklem 5'teki gibi tanımlanır:

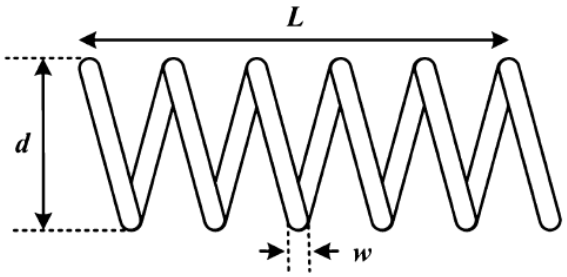
$$\begin{aligned} &min\ f(x) \\ &\text{kısıtlar: } g_k(x) \leq 0, k = 1,2,3,\dots,q \\ &\qquad\quad h_j(x) = 0, j = 1,2,3,\dots,m \\ &\qquad\quad alt_i \leq x_i \leq ust_i, i = 1,2,3,\dots,D \end{aligned}$$

Burada $f(x)$ amaç fonksiyondur. $x = (x_1, x_2, \dots, x_D)$ olmak üzere D boyutlu bir vektördür. $g_k(x)$ eşitsizlik kısıtıdır ve $h_j(x)$ eşitlik kısıtıdır. alt_i ; x_i'nin alt sınırıdır ve ust_i ; x_i'nin üst sınırıdır.

Şekil 6'da gösterilen germe / sıkıştırma yay tasarım probleminin çözümündeki temel amaç değişkenlerin optimum değerlerini elde ederek minimum ağırlıklı bir yay tasarlamaktır. Problem dört lineer olmayan eşitsizlik kısıtı ve tel çapı w (x_1), ortalama bobin çapı d (x_2), uzunluk (veya bobin sayısı) L (x_3) olmak üzere üç sürekli değişkenden oluşur. Bu kısıtlar ve problem Denklem 6'da gösterilmektedir.

SKA ve karşılaştırılan diğer algoritmalar (PSO, BOA, KKA, YAA) germe / sıkıştırma yay tasarım problemi üzerinde test edilirken popülasyon sayısı 200, iterasyon sayısı 10000 olarak belirlenmiştir ve algoritmalar 30 kez çalıştırılmıştır. Elde edilen sonuçlar Tablo 6'da gösterilmektedir. SKA problemi 71.69195 saniyede çözerek karşılaştırılan algoritmalar arasında sonucu en düşük sürede bulan algoritma

olmuştur. Ayrıca optimum maliyeti 0.012676 bularak PSO'dan sonra optimum maliyeti en düşük bulan ikinci algoritma konumundadır.



Şekil 6. Germe / sıkıştırma yay tasarım problemi

$$\min f(x_1, x_2, x_3) = (x_3 + 2)x_1^2 x_2$$

$$g_1(X) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \leq 0$$

$$g_2(X) = \frac{x_2(4x_2 - x_1)}{12566x_1^3(x_2 - x_1)} + \frac{1}{5108x_1^2} - 1 \leq 0$$

$$g_3(X) = 1 - \frac{140.45x_1}{x_2^2 x_3} \leq 0$$

$$g_4(X) = \frac{2(x_1 + x_2)}{3} - 1 \leq 0$$

$$Değişken\ aralığı:\ 0.05 \leq x_1 \leq 2,\quad 0.25 \leq x_2 \leq 1.3,\quad 2.0 \leq x_3 \leq 15.0$$

Tablo 6. Germe / sıkıştırma yay tasarım problemi karşılaştırma sonuçları

Algoritma	Optimum değişkenler			*Optimum Maliyet*	*Süre*
	w	*d*	*L*		
SKA	0.051108490847	0.342889153904	12.15302541048	0.012676201923	71.69195
PSO	0.051580471304	0.354110970914	11.44344489883	0.012665448278	517.93111
BOA	0.052614959700	0.379403901376	10.07314988814	0.012680631117	89.987781
KKO	0.057606226982	0.516451973173	5.739265672904	0.013263818150	511.46341
YAA	0.053462849967	0.395370059884	9.564393481895	0.013068653733	2463.76132

4. Sonuç

Farklı kaynaklardan ilham alınarak pek çok metasezgisel algoritma geliştirilmesine rağmen matematikten ilham alan metasezgisel algoritma sayısı oldukça azdır. Bu çalışmada yakın zamanda önerilen matematik tabanlı Sinüs Kosinüs Algoritması (SKA)'nın optimizasyon problemlerinin çözümünde kullanılması incelenmiştir. SKA 23 kalite testi fonksiyonu üzerinde ve mühendislik tasarım problemlerinden biri olan germe / sıkıştırma yay tasarım probleminin çözümünde diğer metasezgisel algoritmalar (PSO, BOA, KKA, YAA) ile karşılaştırılarak test edilmiştir. Ayrıca stokastik yöntemlerin performansının daha güvenilir bir şekilde belirlenmesini sağlayan istatiksel testlerden Wilcoxon sıra toplamı testi kullanılarak algoritmalardan elde edilen sonuçlar istatiksel olarak karşılaştırılmıştır. SKA'nın henüz yeni bir algoritması ve az sayıda parametre içermesinden dolayı gelecek çalışmalarda algoritmada iyileştirmeler yapılarak daha iyi sonuçlar elde edilmesi amaçlanmaktadır.

5. Kaynaklar

1. Ebrahimi, A., Khamehchi, E. (2016). Sperm whale algorithm: an effective metaheuristic algorithm for production optimization problems. *Journal of Natural Gas Science and Engineering*, **29**, 211-222.

2. Li, M.D., Zhao, H., Weng, X.W., Han, T. (2016). A novel nature-inspired algorithm for optimization: virüs colony search. *Advances in Engineering Software*, **92**, 65-88.

3. Holland, J.H. (1975). Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, *U Michigan Press*.

4. Storn, R., Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, **11**(4), 341-359.

5. Parpinelli, R.S., Lopes H.S. (2011). An eco-inspired evolutionary algorithm applied to numerical optimization. *In Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress,* 466-471.

6. Mehrabian, A.R., Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. *Ecological Informatics*, **1**(4):355-366.

7. Yan, G.W. (2013) A novel optimization algorithm based on atmosphere clouds model. *International Journal of Computational Intelligence and Applications*, **12**(1).

8. Hedayatzadeh, R., Salmassi, F.A., Keshtgari, M., Akbari, R., Ziarati, K. (2010). Termite colony optimization: A novel approach for optimizing continuous problems. *Electrical Engineering (ICEE), 2010 18th Iranian Conference*.

9. Kennedy, J., Eberhart, R. (1995). Particle swarm optimization in neural networks, *IEEE International Conference*, **4**, 1942–1948.

10. Dorigo, M. (1992). Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano, Italy.

11. Karaboga, D., Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of global optimization*, **39**(3), 459-471.

12. **T**ang, R., Fong, S., Yang, X.S., Deb, S. (2012). Wolf search algorithm with ephemeral memory. *In Digital Information Management (ICDIM)*. 2012 Seventh International Conference, 165-72.

13. Yang, X.S., Deb, S. (2009). Cuckoo search via l´evy flights. *In Nature & Biologically Inspired Computing*, 210-214.

14. Mirjalili, S., Mirjalili, S.M., Lewis, A. (2014). Grey wolf optimizer, *Adv Eng Softw*, **69**, 46-61.

15. Mirjalili, S., Mirjalili, S.M. (2016). The whale optimization algorithm, *Adv Eng Softw*, **95**, 51-67.

16. Zandi, Z., Afjei, E., Sedighizadeh, M. (2012). Reactive power dispatch using big bang-big crunch optimization algorithm for voltage stability enhancement. *In Power and Energy (PECon)*, 2012 IEEE International Conference, 239-244.

17. Formato, R.A. (2007). Central force optimization: a new metaheuristic with applications in applied electromagnetics. *Progress In Electromagnetics Research*, **77**, 425-491.

18. Rashedi, E., Pour H.N., Saryazdi, S. (2009). GSA: a gravitational search algorithm, *Information sciences*, **179**(13), 2232-2248.

19. Eskandar, H., Sadollah, A., Bahreininejad, A., Hamdi, M. (2012). Water cycle algorithm-a novel metaheuristic optimization method for solving constrained engineering optimization problems. *Computers & Structures*.

20. Benasla, L., Belmadani, A., Rahli, M. (2014). Spiral optimization algorithm for solving combined economic and emission dispatch. *International Journal of Electrical Power & Energy Systems*, **62**, 163–174.

21. Kashan, A.H. (2009). League championship algorithm: a new algorithm for numerical function optimization, *In Soft Computing and Pattern Recognition, SOCPAR'09*, 43-48.

22. Moosavian, N., Roodsari, B.K. (2014). Soccer league competition algorithm, a new method for solving systems of nonlinear equations. *Int. J. Intell. Sci.*, **4**(1), 7-16.

23. Osaba, E., Diaz, F., Onieva, E. (2014). Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts, *Applied Intelligence*, 2014.

24. Salem, S.A. (2012). BOA: A novel optimization algorithm. *In International Conference on Engineering and Technology (ICET)*, 1-5, Egypt.

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