International Journal of Innovative Engineering Applications

Journal homepage: https://dergipark.org.tr/ijiea

INVESTIGATION OF THE PERFORMANCE OF METAHEURISTIC OPTIMIZATION ALGORITHMS USED IN SOLVING REAL-WORLD ENGINEERING DESIGN PROBLEMS

Elif Varol Altay*[1]

[1]*Department of Software Engineering, Manisa Celal Bayar University, Manisa, Turkey, 45400*

Abstract

Original scientific paper

Real-world engineering design problems are relatively difficult problems to solve because they have complex objective functions with many constraints. It is widely used both in industry and in different interdisciplinary studies. Many optimization algorithms have been used to deal with such problems. But the performance of the algorithm decreases significantly with the increasing scale and difficulty of the problem. Different optimization methods and their different versions have been proposed to effectively deal with the engineering design problems in the literature. In this study, new artificial intelligence-based optimization algorithms that have emerged recently and are popular for solving engineering design problems are examined. The analyzed algorithms; constraints, objective functions, and decision variables were adapted to five different real-world engineering design problems, and performance analyses were performed.

Keywords: *Engineering design problems, metaheuristic algorithms, optimization.*

GERÇEK DÜNYA MÜHENDİSLİK TASARIM PROBLEMLERİNİN ÇÖZÜMÜNDE KULLANILAN METASEZGİSEL OPTİMİZASYON ALGORİTMALARININ PERFORMANSLARININ İNCELENMESİ

Özet

Orijinal bilimsel makale

Gerçek dünya mühendislik tasarım problemleri çok sayıda kısıtlamaları olan karmaşık amaç fonksiyonlarına sahip olmasından dolayı çözülmesi nispeten zor problemlerdir. Hem endüstride hem de disiplinler arası çalışmalarda yaygın olarak kullanılmaktadır. Bu tür problemlerle başa çıkmak için birçok optimizasyon algoritması kullanılmıştır. Fakat algoritmanın performansı, ölçeğin artması ve problemin zorluğu ile önemli ölçüde azalmaktadır. Literatürde yer alan mühendislik tasarım problemlerini etkin bir şekilde ele almak için farklı optimizasyon yöntemleri ve onların farklı versiyonları önerilmiştir. Bu çalışmada, mühendislik tasarım problemlerini çözmek için son dönemlerde ortaya çıkmış ve popüler olan metasezgisel optimizasyon algoritmaları incelenmiştir. İncelenen algoritmalar; kısıtları, amaç fonksiyonları ve karar değişkenleri farklı beş gerçek dünya mühendislik tasarım problemine uyarlanmıştır ve performans analizleri gerçekleştirilmiştir.

Anahtar Kelimeler: *Metasezgisel algoritmalar, mühendislik tasarım problemleri, optimizasyon.*

1 Giriş

Son yıllarda, çeşitli kısıtlı mühendislik problemlerini çözmek için çeşitli algoritmalar geliştirilmiştir. Bu tür algoritmaların çoğu önemli gradyan bilgisi gerektirebilecek ve genellikle çözümü bir başlangıç noktasının yakınında iyileştirmeye çalışan sayısal, doğrusal ve doğrusal olmayan programlama yöntemlerine dayanmaktadır. Bu sayısal optimizasyon algoritmaları, basit ve ideal modeller için global optimum çözümü elde etmek için faydalı bir strateji sağlamaktadır. Fakat birçok gerçek dünya mühendislik problemi doğası gereği çok karmaşıktır ve çözülmesi oldukça zor problemlerdir. Problemde birden fazla yerel optimum varsa sonuçlar elde edilen optimal çözümün mutlaka global optimum olmayabileceği başlangıç noktasının seçimine bağlı olabilir. Ayrıca amaç fonksiyonu ve kısıtlamalar çoklu veya keskin tepelere sahip olduğundan gradyan araması kararsız hale gelebilir. Klasik yöntemlerin bu dezavantajları araştırmacıları mühendislik tasarım problemlerini çözmek için simülasyonlara dayalı doğadan ilham alan metasezgisel yöntemlere güvenmeye teşvik etmiştir. Metasezgisel optimizasyon algoritmaları; birçok farklı optimizasyon probleminin çözümünü bulmak için

* Corresponding author.

E-mail address: elif.altay@cbu.edu.tr (E. Varol Altay)

Received 15 March 2022; Received in revised form 26 April 2022; Accepted 27 April 2022

Doi: https://doi.org/10.46460/ijiea.1088408

yaygın olarak kullanılan, büyük ölçekli arama ve optimizasyon problemleri için iyi bilinen bir global optimizasyon yaklaşımıdır. Metasezgisel yöntemler genellikle doğadaki olayları, hayvanların davranışlarını taklit etmek için kuralları ve rastgeleliği birleştirerek çalışmaktadır [1].

Metasezgisel algoritmalar keşif ve sömürü aşamasından oluşmaktadır. Keşif aşaması, algoritmanın belirli bir arama uzayında farklı umut vaat eden bölgeleri araştırmasını sağlarken, sömürü aşaması umut verici bölgeler etrafında optimal çözümlerin aranmasını sağlamaktadır. Ancak metasezgisel algoritmaların stokastik doğası nedeniyle bu aşamalar arasında denge kurmak zordur. Bu nedenle optimuma yakın çözümlere ulaşmak için bu iki aşamanın iyi bir şekilde ayarlanması ve dengenin sağlanması gerekmektedir [2, 3].

Son yıllarda çok fazla sayıda metasezgisel optimizasyon algoritması önerilmiştir. Ancak her tür optimizasyon yöntemini çözebilecek tek bir algoritma mevcut değildir [4]. Bazı algoritmalar diğerlerine kıyasla daha iyi optimal sonuçlar sağlarlar. Bu nedenle yeni bir metasezgisel algoritma geliştirmek açık bir problemdir ve araştırmacıları yeni metasezgisel yöntemler geliştirmek için motive etmektedir [5, 6].

Metasezgisel optimizasyon algoritmaları birçok mühendislik problemine başarıyla uygulanmıştır. Bu çalışmada gri kurt optimizasyon algoritması (GKO), güve-alevi optimizasyon algoritması (GAO), şempanze optimizasyon algoritması (ŞOA), denge optimize edici algoritması (DOE), balina optimizasyon algoritması (BAO) olmak üzere beş metasezgisel optimizasyon algoritması, farklı zorluk seviyesindeki kısıtlama ve arama alanı ile basınçlı kap, kaynaklı kiriş, hız düşürücü, çok diskli kavrama fren ve konsol kiriş tasarım problemi olmak üzere beş gerçek dünya mühendislik tasarım problemindeki etkinlikleri analiz edilmiştir ve birbirleriyle karşılaştırılmıştır.

Çalışmanın geri kalanı şu şekilde yapılandırılmıştır: 2. Bölümde çalışmada kullanılan metasezgisel algoritmalar yer almaktadır. 3. Bölümde gerçek dünya mühendislik problemlerinin tanımı, 4. Bölümde elde edilen deney sonuçlarının analizi bulunmaktadır. Son bölüm olan 5. Bölümde çalışmanın sonuçları ve ileride yapılabilecek çalışmalardan bahsedilmiştir.

2 Metasezgisel Optimizasyon Algoritmaları

Bu bölümde, bu çalışmada kullanılan algoritmalardan kısaca bahsedilmiştir. Daha fazla ayrıntı ve literatür için okuyucular alıntı yapılan makaleleri inceleyebilir.

2.1 Şempanze Optimizasyon Algoritması

ŞOA, Khishe ve Mosavi tarafından şempanzelerin grup avlarında bireysel zeka ve cinsel motivasyonlarından esinlenerek geliştirilmiş biyoloji tabanlı optimizasyon algoritmasıdır [7]. Toplumdaki diğer etoburlardan farklıdır. Bu metodoloji sırasında farklı zekayı modellemek için dört farklı aşama kullanılmıştır. Burada ilk çözümün hedefi kovalayan, sürücü, saldırgan ve bariyer tarafından daha iyi bilindiği varsayılmaktadır. Bir sonraki aşamada, elde edilen dört optimum çözüm saklanır ve diğer şempanzeler, şempanzelerin en iyi yerlerine kendi konumlarını güncellemeye zorlanır. Önerilen algoritmanın matematiksel modeli aşağıdaki gibi tanımlanmıştır:

$$D_r = \left| c \times a_{prey}(n_i) - m a_{chmp}(n_i) \right| \quad (1)$$

$$a_{chimp}(n_i + 1) = a_{prey} - a \times d \quad (2)$$

Toplam iterasyon sayısı n_i ile, katsayı vektörleri c, m ve a ile temsil edilmektedir. c, m ve a katsayıları aşağıdaki denklemlerle çözülmektedir:

$$a = 2 \times l \times r_1 - l \quad (3)$$

$$c = 2 \times r_2 \quad (4)$$

$$m = kaotik_{değer} \quad (5)$$

r_1 ve r_2 [0,1] arasında rastgele bir değerdir, m kaotik vektör ve iterasyon boyunca l 2.5'ten 0'a doğru doğrusal bir şekilde düşürülmektedir.

$$d_{attacker} = |c_1 \times a_{attacker} - m_1 \times x| \quad (6)$$

$$d_{barrier} = |c_2 \times a_{barrier} - m_2 \times x| \quad (7)$$

$$d_{chaser} = |c_3 \times a_{chaser} - m_3 \times x| \quad (8)$$

$$d_{driver} = |c_4 \times a_{driver} - m_4 \times x| \quad (9)$$

Rastgele vektörler [-1, 1] aralığında olduğunda, şempanzenin bir sonraki konumu, şimdi olduğu yer ile hedef veya avın olduğu yer arasında bir yerde olabilir.

$$x_1 = a_{attacker} - a_1 \times d_{attacker} \quad (10)$$

$$x_2 = a_{barrier} - a_2 \times d_{barrier} \quad (11)$$

$$x_3 = a_{chaser} - a_3 \times d_{chaser} \quad (12)$$

$$x_4 = a_{driver} - a_4 \times d_{driver} \quad (13)$$

Arama sırasında şempanzelerin yerini değiştirmek için aşağıdaki matematiksel denklem kullanılmaktadır:

$$x_{ni+1} = \frac{x_1 + x_2 + x_3 + x_4}{4} \quad (14)$$

Arama alanındaki arama işlemi sırasında şempanzelerin konumunu güncellemek için Denklem (15) uygulanmaktadır.

$$a_{chimp}(n_i + 1) = \begin{cases} a_{prey}(n_i) - x \times d, & \emptyset < 0.5 \\ kaotik_{değer}, & \emptyset > 0.5 \end{cases} \quad (15)$$

2.2 Denge Optimize Edici Algoritması

DOE, Faramarzi ve arkadaşları tarafından 2020 yılında geliştirilmiş bir kontrol hacmindeki basit iyi karıştırılmış dinamik kütle dengesini taklit eden metasezgisel bir algoritmadır [8]. Bu algoritmada, çeşitli kaynak ve yutak mekanizmalarının bir fonksiyonu olarak

bir kontrol hacmindeki reaktif olmayan bir bileşenin konsantrasyonunu karakterize etmek için bir kütle dengesi denklemi kullanılmaktadır. Başlatma sürecinden sonra, DOE'deki aday çözümler aşağıdaki arama denklemi kullanılarak güncellenmektedir:

$$C_i^{t+1} = C_{eq}^t + \left(C_i^t - C_{eq}^t\right) \times F_i^t + (1 - F_i^t) \times \frac{G_i^l}{\lambda_i^t v_i^t} \quad (16)$$

C_i^t ve C_i^{t+1} sırasıyla t ve $t+1$. iterasyondaki i'ninci aday çözümler için konsantrasyon vektörleridir. C_{eq}^t Denklem (17)'deki gibi oluşturulduğunda denge aracından rastgele seçilen bir vektördür.

$$C_{eq}^t = \left[C_{eq(1)}^t, C_{eq(2)}^t, C_{eq(3)}^t, C_{eq(4)}^t, C_{eq(mean)}^t\right] \quad (17)$$

$C_{eq(1)}^t, C_{eq(2)}^t, C_{eq(3)}^t, C_{eq(4)}^t \; ve \; C_{eq(mean)}^t$ vektörleri sırasıyla t iterasyonuna kadar en iyi dört aday çözüm ve bunların ortalama konumudur. Üstel terim olarak bilinen sömürü ile keşif arasında bir denge sağlamaya çalışan F_i^t vektörü Denklem (18)'deki gibi hesaplanmaktadır:

$$F_i^{t+1} = e^{\lambda_i^t (t - t_0)} \quad (18)$$

$$t = \left(1 - \frac{t}{t_{max}}\right)^{a_2 \frac{t}{t_{max}}} \quad (19)$$

$$t_0 = t + \frac{1}{\lambda} \ln(-a_1 sign(rand - 0.5) \times (1 - e^{\lambda t})) \quad (20)$$

a_1 keşif kabiliyetini kontrol eden bir değişken olduğunda, $sign(rand - 0.5)$ keşif ve sömürünün yönünü göstermektedir, $rand$ $(0, 1)$ aralığında rastgele seçilmiş bir değişkendir ve a_2 sömürü yeteneğini dengelemek için kullanılan bir sabittir. Tüm değerleri yerleştirdikten sonra F_i^t için ifade Denklem (21)'deki gibi verilebilir:

$$F_i^t = a_1 sign(rand - 0.5) \times (e^{-\lambda \times t} - 1)) \quad (21)$$

İterasyon oranı G_i^t Denklem (22)'deki gibi hesaplanmaktadır:

$$G_i^t = G_0^t e^{\lambda_i^t (t - t_0)} \quad (22)$$

$$G_0^t = \overrightarrow{GCP^t} \times (C_{eq}^t - \lambda^t C_i^t) \quad (23)$$

$$\overrightarrow{GCP^t} = \begin{cases} 0.5 \; rand_1, & if \; rand_2 \geq 0.5 \\ 0, & aksi \; takdirde \end{cases} \quad (24)$$

$\overrightarrow{GCP^t}$ iterasyon hızı kontrol parametresidir.

2.3 Gri Kurt Optimizasyon Algoritması

GKO, Mirjalili ve arkadaşlarının doğadaki gri kurtların sosyal liderlik ve avlanma davranışlarından ilham alarak geliştirdikleri metasezgisel bir optimizasyon algoritmasıdır [9]. Gri kurtlar sürüler halinde yaşamaktadır ve çok katı bir sosyal baskın hiyerarşiye sahiptirler. Alfa kurtları en üstte ve omega kurtları yukarıdan aşağıya doğru azalan baskınlık piramidinin en altında bulunmaktadır. Üstten ikinci katman beta kurtları ve sondan bir önceki katman delta kurtları gri kurt sürüsünün sosyal hiyerarşi piramidini tamamlamaktadır. Bu yöntem kuşatma, avlanma ve avına saldırma olmak üzere üç aşamadan oluşmaktadır. Herhangi bir optimizasyon problemini çözmek için sosyal hiyerarşinin matematiksel modellemesi, sırasıyla en uygun ve en iyi çözüm alfa, ikinci ve üçüncü en iyi çözümler beta ve delta olarak sınıflandırılmaktadır. Diğer tüm çözümler omega olarak sınıflandırılmaktadır. Herhangi bir kurt ile av arasındaki mesafenin matematiksel denklemi aşağıdaki gibidir:

$$D = \left|C \times X_p(t) - X(t)\right| \quad (25)$$

$$X(t+1) = X_p(t) - A \times D \quad (26)$$

Burada X gri kurdun konum vektörünü, X_p avın konumunu, t ise mevcut iterasyon sayısını temsil etmektedir. C ve A katsayı vektörleridir ve aşağıdaki denklemlerle hesaplanmaktadır:

$$A = 2 \times a \times r_1 - a(t) \quad (27)$$

$$C = 2 \times r_2 \quad (28)$$

Burada r_1 ve r_2 [0, 1] arasında rastgele sayılardır ve a 2'den 0'a doğru lineer bir şekilde Denklem (29)'daki gibi azalmaktadır.

$$a(t) = 2 - (2 \times t)/MaxIter \quad (29)$$

Avın konumu veya çözüm ortamında aranan optimum çözüm bilinmediği için kurtların sosyal hiyerarşi metaforu kullanılmaktadır. Kurtların avlanma davranışları modellenirken alfa, beta ve deltanın konumu hakkında daha iyi bir bilgiye sahip olduğu varsayılmaktadır. Bu nedenle her bir omega kurtu konumlarını alfa, beta ve deltayı dikkate alarak güncellemektedir. Avlanma davranışı Denklem (30)'daki gibidir:

$$\begin{aligned} D_\alpha &= |C_1 \times X_\alpha - X(t)| \\ D_\beta &= |C_2 \times X_\beta - X(t)| \\ D_\delta &= |C_3 \times X_\delta - X(t)| \end{aligned} \quad (30)$$

Burada C_1, C_2 ve C_3 Denklem (28) ile hesaplanmaktadır. Denklem (31) alfa, beta ve deltanın konumunu hesaplanmaktadır. Yeni konumu hesaplamak için Denklem (32) kullanılmaktadır.

$$\begin{aligned} X_{i1}(t) &= X_\alpha(t) - A_{i1} \times D_\alpha(t) \\ X_{i2}(t) &= X_\beta(t) - A_{i2} \times D_\beta(t) \\ X_{i3}(t) &= X_\delta(t) - A_{i3} \times D_\delta(t) \end{aligned} \quad (31)$$

$$X(t+1) = \frac{X_{i1}(t) + X_{i2}(t) + X_{i3}(t)}{3} \quad (32)$$

Kuşatma ve avlanma operatörleri tekrar tekrar uygulanarak av veya en iyi çözüm bulunmaktadır.

2.4 Güve-Alevi Optimizasyon Algoritması

GAO, Mirjalili tarafından güvenin doğal davranışlarından ve enine yönlendirme adı verilen gezinme yapısından esinlenerek geliştirilmiş metasezgisel

optimizasyon algoritmasıdır [10]. GAO'da güveler arama prosedürü için aday çözümler veya arama ajanları olarak kabul edilirken, alevler arama alanı içinde o zamana kadar elde edilen en iyi konumları temsil etmektedir. Bu nedenle arama prosedürü sırasında güveler tarafından düşen alevler bayrak olarak kabul edilmektedir. Her güve yeni gelecek vaat eden bölgeleri keşfetmeye çalışmaktadır ve daha iyi çözüm sağlamak için konumunu güncellemektedir. En iyi alevin konumu, algoritmanın bir sonraki iterasyonuna iletilmektedir ve bu nedenle, bu bilgi arama prosedürü sırasında asla ölmemektedir. GAO'da önce güve popülasyonu başlatılır. Herhangi bir iterasyonda, bu popülasyon aşağıdaki arama mekanizmasını kullanarak güncellenmektedir:

$$M_i^{t+1} = F_j + D_i \times e^{bl} \times \cos(2\pi i) \quad (33)$$

M_i^{t+1} $t+1$ iterasyonda $i.$ güvenin konumudur. F_j $j.$ alevdir ve D_i $i.$ güve ile $j.$ alev arasındaki mesafedir. b logaritmik spiralin şeklini tanımlayan bir sabittir ve l $[r, 1]$ aralığında rastgele bir sayıdır. r adaptif yakınsama sabiti olarak bilinmektedir ve iterasyon boyunca alevlerin etrafındaki yakınsama hızını arttırmak için -1 değerinden -2 değerine doğrusal olarak azalmaktadır. Arama prosedürü sırasında alev sayısı N_f Denklem (34) kullanılarak değiştirilmektedir:

$$N_f = round\left[N - \frac{t}{t_{max}} \times (N-1)\right] \quad (34)$$

N maksimum alev sayısını temsil ederken t ve t_{max} sırasıyla mevcut ve maksimum iterasyon sayısını temsil etmektedir.

2.5 Balina Optimizasyon Algoritması

BOA, Mirjalili ve Lewis tarafından karmaşık optimizasyon problemlerini çözmek için kambur balinaların sosyal davranışlarını taklit eden ve doğadan ilham alan metasezgisel bir optimizasyon algoritmasıdır [11]. Kambur balinalar avın yerini tanıyabilmektedir ve onları tamamen kaplayabilmektedir. BAO'da, mevcut en iyi arama aracısının hedef av olduğu varsayılmaktadır ve kambur balinalar, iterasyonlar boyunca konumlarını en iyi arama aracına doğru güncellemektedirler. Bu davranışı matematiksel olarak formüle etmek için aşağıdaki denklemler kullanılmaktadır.

$$\vec{D} = \left|\vec{C}.\overrightarrow{X^*}(t) - \vec{\mathrm{X}}(t)\right| \quad (35)$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A}.\vec{D} \quad (36)$$

$\overrightarrow{X^*}$ o zamana kadar elde edilen en iyi çözümün konum vektörünü temsil etmektedir. $\vec{A}$ ve $\vec{C}$ katsayı vektörleridir ve aşağıdaki denklemlerle hesaplanmaktadır.

$$\vec{A} = 2.\vec{a}.\vec{r} - \vec{a} \quad (37)$$

$$\vec{a} = 2 - 2\frac{t}{t_{max}} \quad (38)$$

$$\vec{C} = 2.\vec{r} \quad (39)$$

Burada $\vec{a}$ iterasyonlar boyunca ve hem keşif hem de sömürü aşamalarında 2'den 0'a doğru lineer bir şekilde azalan bir sayıdır. $\vec{r}$ [0,1] arasında üretilen rastgele bir vektörü, t_{max} maksimum iterasyon sayısı, t mevcut iterasyon sayısıdır. Denklem (37) ve (39)'un amacı keşif ve sömürü arasındaki dengeyi sağlamaktır. Her iki denklemde de r rastgele bir sayı oluşturmaktadır. Bu da popülasyonun konum güncellemesi için stokastik bir davranış sağlamaktadır. $A \geq 1$ olduğunda keşif, $A < 1$ olduğunda sömürü aşaması gerçekleşmektedir. Bu olay, optimizasyonun herhangi bir aşamasında sömürünün araştırılmasına yol açmaktadır.

Küçülen çevreleyen ve spiral güncelleme konum mekanizmaları, kambur balinaların kabarcık ağ saldırı yöntemini modellemek için aynı anda kullanılmaktadır. Daralan çevreleme mekanizması, iterasyonlar boyunca a'nın değerini doğrusal olarak azaltırken, katsayı vektörü A'yı [-1, 1]'de ayarlayarak elde etmektedir. Bunu yaparken yeni konum, mevcut konumu ile en iyi arama ajanının konumu arasında yer almaktadır. Kambur balinaların sarmal şeklindeki hareketlerini taklit etmek için avın konumu ile balina arasındaki spiral denklem aşağıdaki gibi oluşturulmaktadır.

$$\vec{X}(t+1) = \overrightarrow{D'}.e^{bl}.\cos(2\pi l) + \overrightarrow{X^*}(t) \quad (40)$$

$$\overrightarrow{D'} = \left|\overrightarrow{X^*}(t) - \vec{\mathrm{X}}(t)\right| \quad (41)$$

$\overrightarrow{D'}$ $i.$ balinanın ava olan mesafesini gösterirken, l [-1, 1] aralığında rastgele bir sayıdır, b logaritmik spiralin şeklini tanımlamak için bir sabittir. Kambur balinalar, avın etrafında daralan bir daire içinde yüzdükleri ve aynı anda spiral şeklinde yol boyunca hareket ettikleri için, küçülen çevreleme yöntemi ve spiral yaklaşım aynı anda kullanılmaktadır. Bu davranışı modellemek için her bir mekanizmanın %50 olasılıkla gerçekleştiği varsayılmaktadır.

A vektörünün varyasyonuna dayanan yaklaşım, av arama (keşif) içinde kullanılabilmektedir. Kambur balinalar birbirlerinin konumuna göre rastgele arama yapmaktadırlar. Bu nedenle arama ajanını referans balinadan uzaklaşmaya zorlamak için 1'den küçük veya 1'den büyük rastgele değerlerle A kullanılmaktadır. Sömürü aşamasının aksine keşif aşamasında bir arama ajanının konumunu o ana kadar bulunan en iyi arama ajanı yerine rastgele seçilen bir arama ajanına göre güncellemektedirler. Bu mekanizma $A > 1$ keşif aşamasını vurgulamaktadır ve BOA'nın global bir arama yapmasına izin vermektedir. Av aramanın matematiksel modeli aşağıdaki gibidir.

$$\vec{D} = \left|\vec{C}.\overrightarrow{X_{rand}} - \vec{\mathrm{X}}\right| \quad (42)$$

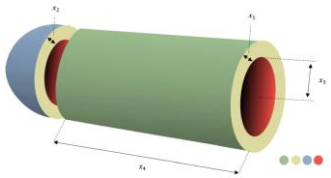
$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A}.\vec{D} \quad (43)$$

3 Mühendislik Tasarım Problemleri

Bu bölümde en yaygın gerçek dünya mühendislik tasarım problemlerinin tanımı ve matematiksel modelleri sunulmaktadır.

3.1 Basınçlı Kap Tasarım Problemi

Bu problemin temel amacı, bir geminin kaynak maliyetini, malzemesini ve oluşumunu optimize etmektir [12]. Bu problem, yerine getirilmesi gereken dört kısıtlamayı içerir ve amaç fonksiyonunu hesaplamak için dört değişken kullanılır: kabuk kalınlığı (x_1), kafa kalınlığı (x_2), iç yarıçap (x_3) ve gemi yüksekliğini dahil etmeden uzunluğudur (x_4). Bu parametrelerin alabileceği değer aralıkları $1 \times 0.0625 \leq x_1, x_2 \leq 99 \times 0.0625$ ve $10 \leq x_3,\ x_4 \leq 200$ olarak tanımlanmıştır. Basınçlı kap tasarım probleminin şematik yapısı Şekil 1'de gösterilmiştir.



Şekil 1. Basınçlı kap tasarım problemi.

Bu problemin amaç fonksiyonu Denklem (44)'te ve kısıtlar Denklem (45-48)'de belirtilmiştir.

Minimize:

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (44)$$

Kısıtlar:

$$g_1(x) = -x_1 + 0.0193x_3 \leq 0 \quad (45)$$

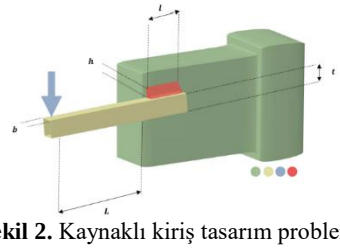
$$g_2(x) = -x_2 + 0.00954x_3 \leq 0 \quad (46)$$

$$g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0 \quad (47)$$

$$g_4(x) = x_4 - 240 \leq 0 \quad (48)$$

3.2 Kaynaklı Kiriş Tasarım Problemi

Kaynaklı kiriş tasarım probleminin temel amacı, belirli kısıtlamalar altında minimum maliyetli bir kiriş üretmektir [12]. Şekil 2, kaynaklı kiriş tasarım probleminin şematik biçimini göstermektedir. Problem, dört tasarım parametresinden ve beş eşitsizlik kısıtlamasından oluşmaktadır. Bu tasarım parametreleri $h\ (x_1)$, $l\ (x_2)$, $t\ (x_3)$ ve $b\ (x_4)$ olup sırasıyla kaynak kalınlığını, kaynak bağlantı uzunluğunu, eleman genişliğini ve eleman kalınlığını temsil etmektedir. Bu parametrelerin alabileceği değer aralıkları sırasıyla $0.125 \leq x_1 \leq 5$ ve $0.1 \leq x_2, x_3, x_4 \leq 10$ olarak tanımlanmıştır.



Şekil 2. Kaynaklı kiriş tasarım problemi.

Bu problemin amaç fonksiyonu Denklem (49)'da ve kısıtlar Denklem (50-56)'da belirtilmiştir.

Minimize:

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \quad (49)$$

Kısıtlar:

$$g_1(x) = \tau(x) - \tau_{max} \leq 0 \quad (50)$$

$$g_2(x) = \sigma(x) - \sigma_{max} \leq 0 \quad (51)$$

$$g_3(x) = x_1 - x_4 \leq 0 \quad (52)$$

$$g_4(x) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0 \quad (53)$$

$$g_5(x) = 0.125 - x_1 \leq 0 \quad (54)$$

$$g_6(x) = \delta(x) - \delta_{max} \leq 0 \quad (55)$$

$$g_7(x) = P - P_c(x) \leq 0 \quad (56)$$

Burada;

$$\tau(x) = \sqrt{(\tau')^2 + (2\tau'\tau'')\frac{x_2}{2R} + (\tau'')^2} \quad (57)$$

$$\tau' = \frac{6000}{\sqrt{2}x_1x_2} \quad (58)$$

$$\tau'' = \frac{MR}{J} \quad (59)$$

$$M = 6000(14.0 + \frac{x_2}{2}) \quad (60)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2} \quad (61)$$

$$J = \left\{x_1x_2\sqrt{2}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \quad (62)$$

$$\sigma(x) = \frac{504000}{x_4x_3^2} \quad (63)$$

$$\delta(x) = \frac{2.1952}{x_4x_3^3} \quad (64)$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{\sqrt[x_3]{\frac{E}{4G}}}{28}\right) \quad (65)$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{\sqrt[x_3]{\frac{E}{4G}}}{28}\right) \quad (66)$$

$\tau_{max} = 13600psi$

$\sigma_{max} = 30000psi$

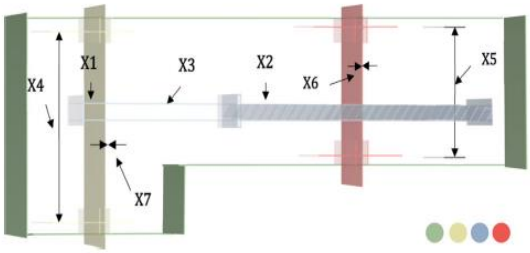
$\delta_{max} = 0.25in$

$E = 30 \times 10^6 psi$

$G = 12 \times 10^6 psi$

3.3 Hız Düşürücü Tasarım Problemi

Bu problem, hava taşıtı motorunun en verimli hızda dönmesini sağlayan basit bir vites kutusu problemidir [13]. Bu problemde yedi karar değişkeninin minimum değerleri yüz genişliği $b\,(x_1)$, diş modülü $m\,(x_2)$, pinyondaki diş sayısı $z\,(x_3)$, yataklar arasındaki ilk şaftın uzunluğu $l_1\,(x_4)$, yataklar arasındaki ikinci şaftın uzunluğu $l_2\,(x_5)$, birinci şaftın çapı $d_1\,(x_6)$ ve ikinci şaftın çapı $d_2\,(x_7)$ bulunarak optimize edilmesi gerekmektedir. Verilen tüm değişkenler, sabit aralıklara giren pozitif tam sayılardır. Bu parametrelerin alabileceği değer aralıkları $0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 2.6 \le x_1 \le 3.6, 5 \le x_7 \le 5.5, 7.3 \le x_5, x_4 \le 8.3, 2.9 \le x_6 \le 3.9$'dur. Hız düşürücü tasarımının şematik gösterimi Şekil 3'te verilmiştir. Bu tasarım problemi, hız düşürücünün minimum maliyet ağırlığını bulmayı amaçlamaktadır. Bu problemin matematiksel gösterimi aşağıdaki gibidir.



Şekil 3. Hız düşürücü tasarım problemi.

Minimize:

$$F(x) = 0.7854x_2^2x_1(14.9334x_3 - 43.0934 + 3.3333x_3^2) + 0.7854(x_5x_7^2 + x_4x_6^2) - 1.508x_1(x_7^2 + x_6^2) + 7.477(x_7^3 + x_6^3) \quad (67)$$

Kısıtlar:

$$g_1(x) = -x_1x_2^2x_3 + 27 \le 0 \quad (68)$$

$$g_2(x) = -x_1x_2^2x_3^2 + 397.5 \le 0 \quad (69)$$

$$g_3(x) = -x_2x_6^4x_3x_4^{-3} + 1.93 \le 0 \quad (70)$$

$$g_4(x) = -x_2x_7^4x_3x_5^{-3} + 1.93 \le 0 \quad (71)$$

$$g_5(x) = 10x_6^{-3}\sqrt{16.91 \times 10^6 + (745x_4x_2^{-1}x_3^{-1})^2} - 1100 \le 0 \quad (72)$$

$$g_6(x) = 10x_7^{-3}\sqrt{157.5 \times 10^6 + (745x_5x_2^{-1}x_3^{-1})^2} - 850 \le 0 \quad (73)$$

$$g_7(x) = x_2x_3 - 40 \le 0 \quad (74)$$

$$g_8(x) = -x_1x_2^{-1} + 5 \le 0 \quad (75)$$

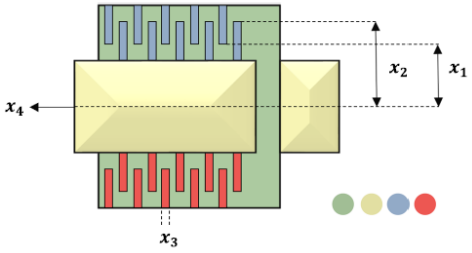
$$g_9(x) = x_1x_2^{-1} - 12 \le 0 \quad (76)$$

$$g_{10}(x) = 1.5x_6 - x_4 + 1.9 \le 0 \quad (77)$$

$$g_{11}(x) = 1.1x_7 - x_5 + 1.9 \le 0 \quad (78)$$

3.4 Çok Diskli Kavrama Fren Tasarım Problemi

Bu problemin temel amacı, çok diskli kavrama frenin kütlesini en aza indirmektir. Bu problemde iç yarıçap (x_1), dış yarıçap (x_2), disk kalınlığı (x_3), aktüatör kuvveti (x_4) ve sürtünme kuvveti (x_5) olmak üzere beş adet karar değişkeni bulunmaktadır [14]. Bu problem dokuz doğrusal olmayan kısıtlama içermektedir. Problemin şematik gösterimi Şekil 4'te çizilmiştir. Bu problemin amaç fonksiyonu Denklem (79)'da ve kısıtlar Denklem (80-87)'de belirtilmiştir.



Şekil 4. Çok diskli kavrama fren tasarım problemi.

Minimize:

$$f(\bar{x}) = \pi * (x_2^2 - x_1^2) * x_3(x_5 + 1)\rho \quad (79)$$

Kısıtlar:

$$g1(\bar{x}) = -P_{max} - P_{rz} \le 0 \quad (80)$$

$$g2(\bar{x}) = P_{rz}V_{sr} - V_{sr,max}P_{max} \le 0 \quad (81)$$

$$g3(\bar{x}) = \Delta R + x_1 - x_2 \le 0 \quad (82)$$

$$g4(\bar{x}) = -L_{max} + (x_5 + 1)(x_3 + \delta \le 0 \quad (83)$$

$$g5(\bar{x}) = sM_s - M_h \le 0 \quad (84)$$

$$g6(\bar{x}) = T \ge 0 \quad (85)$$

$$g7(\bar{x}) = -V_{sr,max} + V_{sr} \le 0 \quad (86)$$

$$g8(\bar{x}) = T - T_{max} \le 0 \quad (87)$$

Burada, $M_h = \frac{2}{3}\mu x_4 x_5 \frac{x_2^3 - x_1^3}{x_2^2 - x_1^2}\ N.mm, \omega = \frac{\frac{\pi n}{30} rad}{s}$,

$A = \pi(x_2^2 - x_1^2)mm^2$,

$P_{rz} = \frac{\frac{x_4}{A}N}{mm^2}, V_{sr} = \frac{\frac{\pi R_{sr} n}{30} mm}{s}, R_{sr} = \frac{2}{3}\frac{x_2^3 - x_1^3}{x_2^2 x_1^2} mm,$

$T = \frac{I_z \omega}{M_h + M_f}$ olarak hesaplanmaktadır.

$\Delta R = 20\ mm, L_{max} = 30\ mm, \mu = 0.6,$

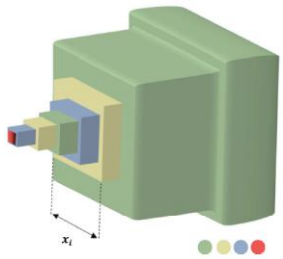
$V_{sr,max} = 10\ m/s, \delta = 0.5\ mm, s = 1.5,$

$T_{max} = 15\ s, n = 250\ rpm, I_z = 55\ kg.m^2,$

$M_s = 40\ Nm, M_f = 3\ Nm, and\ P_{max} = 1$'dir.

3.5 Konsol Kiriş Tasarım Problemi

Konsol kiriş tasarım problemi yaygın bir gerçek dünya mühendislik optimizasyon problemidir. Bu problemde x_1, x_2, x_3, x_4, x_5 olmak üzere beş karar değişkenine bağlı olarak $f(x)$ fonksiyonun optimize edilmesi gerekmektedir [15]. Verilen tüm değişkenler, sabit aralıklara giren pozitif tam sayılarıdır. Konsol kiriş tasarım probleminin şematik gösterimi Şekil 5'te gösterilmiştir. Bu tasarım problemi, konsol kirişin minimum maliyet ağırlığını bulmayı amaçlamaktadır. Parametrelerin alabileceği değer aralıkları $0.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100$ olarak tanımlanmaktadır.



Şekil 5. Konsol kiriş tasarım problemi.

Bu problemin matematiksel gösterimi aşağıdaki gibidir:

Minimize:

$$f(\bar{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5) \quad (88)$$

Kısıtlar:

$$g(x) = \frac{60}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \quad (89)$$

4 Deney Sonuçları

Bu bölümde, uygulanan her bir algoritmanın arama performansını doğrulamak için beş tane gerçek dünya mühendislik tasarım problemi kullanılmıştır. Bu problemler basınçlı kap, kaynaklı kiriş, hız düşürücü tasarım problemi, çok diskli kavrama fren tasarım problemi ve konsol kiriş tasarım problemidir. Tüm bu problemler doğası gereği sınırlıdır ve bu nedenle tasarım kısıtlamalarını çözmek için bir dış ceza yaklaşım mekanizması kullanılmıştır.

Deneyler, 16 GB RAM ve Intel(R) Core (TM) i7-10750H CPU (2.60GHz) içeren bir Windows 11 işletim sistemi kullanılarak gerçekleştirilmiştir. Karşılaştırma algoritmalarını kodlamak için MATLAB R2021a kullanılmıştır. Tüm problemler için maksimum iterasyon sayısı 1000, popülasyon sayısı 30, ve değerlendirme sayısı 30000 olarak belirlenmiştir. Algoritma parametreleri literatürde orijinal makalede yer alan varsayılan değerler olarak seçilmiştir. İncelenen algoritmaların yakınsama davranışını görselleştirmek ve karşılaştırma yapmak için genellikle yakınsama eğrisi olarak adlandırılan her bir problem için elde edilen en iyi uygunluk değerleri çizilmiştir. Her bir algoritmaya karşılık gelen tüm deneyler bağımsız olarak 30 kez çalıştırılmıştır. En iyi, ortalama, en kötü ve standart sapma değerleri karşılaştırmalı olarak incelenmiştir ve okunabilirliğin kolay olması için algoritmalar arasından elde edilen en iyi çözüm koyu renkle vurgulanmıştır.

ŞOA, DOE, GKO, GAO, BOA yöntemlerinin basınçlı kap tasarım problemi üzerindeki performanslarının karşılaştırmalı değerleri Tablo 1'de sunulmuştur. Tablo 1'de kullanılan yöntemlerin en iyi, ortalama, en kötü ve standart sapma değerleri bulunmaktadır. Ayrıca kullanılan yöntemlerin bu problem üzerindeki 30 çalıştırma sonucundaki en iyi değere bağlı karar değişkenleri Tablo 2'de verilmiştir. Tablo 1 incelendiğinde en iyi ve ortalama değerleri üzerinden kıyaslama yapıldığında DOE'nin en iyi değer bazında diğer yöntemlerden daha üstün olduğu ortalama değer bazında inceleme yapıldığında GKO'nun daha başarılı olduğu görülmektedir.

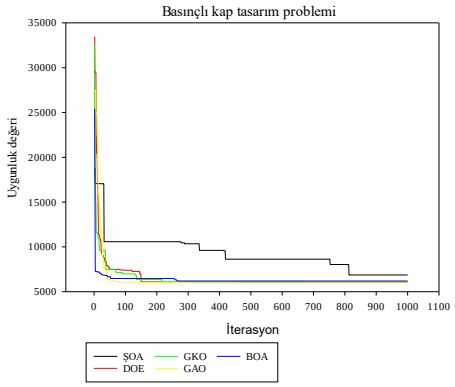
Tablo 1. Basınçlı kap tasarım probleminin farklı metasezgisel yöntemler üzerindeki performanslarının istatistiksel analizi.

Algoritma	En iyi değer	Ortalama	En kötü değer	Standart sapma
ŞOA	6870.394	7861.371	8779.217	437.9636
DOE	**6058.721**	6755.818	7544.493	566.1359
GKO	6058.779	**6280.352**	**7375.882**	442.5123
GAO	6058.73	6688.407	7544.493	494.9048
BOA	6186.282	8188.003	12737.62	1563.279

Tablo 2. Basınçlı kap tasarım probleminin aldığı en iyi değere göre karar değişkenleri.

Algoritma	Karar değişkenleri				f_{min}
	x_1	x_2	x_3	x_4	
ŞOA	1.125	0.5625	57.89439	46.34211	6870.394
DOE	0.8125	0.4375	42.11523	176.4288	6058.721
GKO	0.8125	0.4375	42.11118	176.4793	6058.779
GAO	0.8125	0.4375	42.11317	176.4542	6058.730
BOA	0.8125	0.4375	41.09101	189.5334	6186.282

Kullanılan yöntemlerin basınçlı kap tasarım problemi üzerindeki yakınsama grafiği Şekil 6'da gösterilmiştir.



Şekil 6. Kullanılan yöntemlerin basınçlı kap tasarım problemi üzerindeki yakınsama grafiği.

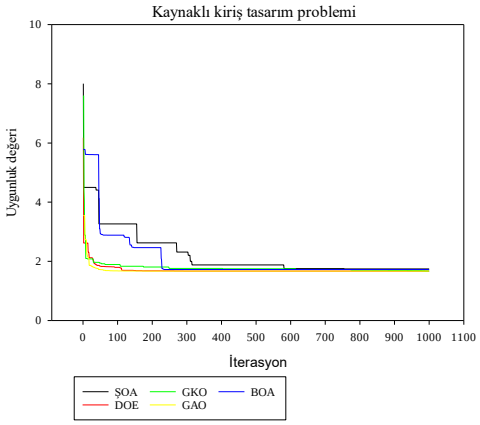
ŞOA, DOE, GKO, GAO, BOA yöntemlerinin kaynaklı kiriş tasarım problemi üzerindeki performanslarının karşılaştırmalı değerleri Tablo 3'te verilmiştir. Tablo 3'te kullanılan yöntemlerin en iyi, ortalama, en kötü ve standart sapma değerleri bulunmaktadır. Ayrıca kullanılan yöntemlerin bu problem üzerindeki 30 çalıştırma sonucundaki en iyi değere bağlı karar değişkenleri Tablo 4'te verilmiştir. Tablo 3 incelendiğinde en iyi, ortalama, en kötü değerleri üzerinden kıyaslama yapıldığında DOE ve GAO'nun en iyi değer bazında diğer yöntemlerden daha üstün olduğu ortalama değer bazında inceleme yapıldığında DOE'nin daha başarılı olduğu görülmektedir. Kullanılan yöntemlerin kaynaklı kiriş tasarım problemi üzerindeki yakınsama grafiği Şekil 7'de gösterilmiştir.

Tablo 3. Kaynaklı kiriş tasarım probleminin farklı metasezgisel yöntemler üzerindeki performanslarının istatistiksel analizi.

Algoritma	En iyi değer	Ortalama	En kötü değer	Standart sapma
ŞOA	1.741815	1.800423	1.869049	0.030705
DOE	**1.670218**	**1.670394**	1.671995	0.000424
GKO	1.671147	1.672948	1.674953	0.001099
GAO	**1.670218**	1.766117	2.282913	0.12409
BOA	1.723824	2.279853	5.15366	0.720567

Tablo 4. Kaynaklı kiriş tasarım probleminin aldığı en iyi değere göre karar değişkenleri.

Algoritma	Karar değişkenleri				f_{min}
	x_1	x_2	x_3	x_4	
ŞOA	0.202303	3.540479	9.26422	0.202325	1.741815
DOE	0.198832	3.337364	9.192024	0.198832	1.670218
GKO	0.198813	3.341102	9.189659	0.198944	1.671147
GAO	0.198832	3.337365	9.192024	0.198832	1.670218
BOA	0.18606	3.51438	9.481026	0.198955	1.723824



Şekil 7. Kullanılan yöntemlerin kaynaklı kiriş tasarım problemi üzerindeki yakınsama grafiği.

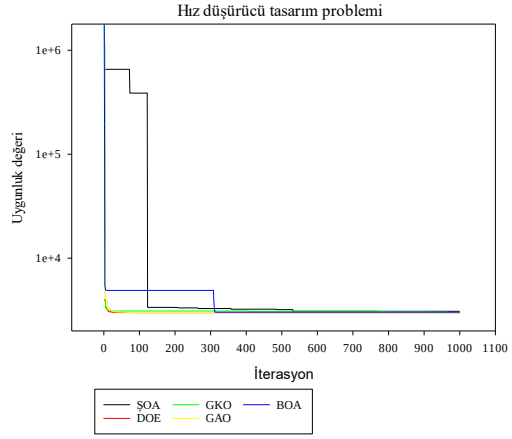
ŞOA, DOE, GKO, GAO, BOA yöntemlerinin hız düşürücü tasarım problemi üzerindeki performanslarının karşılaştırmalı değerleri Tablo 5'te verilmiştir. Tablo 5'te kullanılan yöntemlerin en iyi, ortalama, en kötü ve standart sapma değerleri bulunmaktadır. Ayrıca kullanılan yöntemlerin bu problem üzerindeki 30 çalıştırma sonucundaki en iyi değere bağlı karar değişkenleri Tablo 6'da verilmiştir. Tablo 5 incelendiğinde en iyi, ortalama, en kötü değerleri üzerinden kıyaslama yapıldığında DOE ve GAO'nun en iyi değer bazında diğer yöntemlerden daha üstün olduğu ortalama değer bazında inceleme yapıldığında DOE'nin daha başarılı olduğu görülmektedir. Kullanılan yöntemlerin hız düşürücü tasarım problemi üzerindeki yakınsama grafiği Şekil 8'de gösterilmiştir.

Tablo 5. Hız düşürücü tasarım probleminin farklı metasezgisel yöntemler üzerindeki performanslarının istatistiksel analizi.

Algoritma	En iyi değer	Ortalama	En kötü değer	Standart sapma
ŞOA	3087.550	3163.708	3204.067	32.87455
DOE	**2994.423**	**2994.423**	2994.423	1.16E-12
GKO	3000.972	3007.094	3015.825	4.013066
GAO	**2994.423**	3000.282	3033.702	13.31119
BOA	3022.720	3194.600	4847.088	364.7137

Tablo 6. Hız düşürücü probleminin aldığı en iyi değere göre karar değişkenleri.

Algoritma	Karar değişkenleri							f_{min}
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
ŞOA	3.600000	0.700000	17	8.300000	7.969982	3.364339	5.342464	3087.550
DOE	3.499990	0.700000	17	7.300000	7.715319	3.350541	5.286654	2994.423
GKO	3.502985	0.700048	17	7.327072	7.792379	3.362377	5.286969	3000.972
GAO	3.499990	0.700000	17	8.300000	7.715319	3.352532	5.286654	2994.423
BOA	3.500988	0.700000	17	7.300000	8.073702	3.427115	5.286778	3022.720



Şekil 8. Kullanılan yöntemlerin hız düşürücü tasarım problemi üzerindeki yakınsama grafiği.

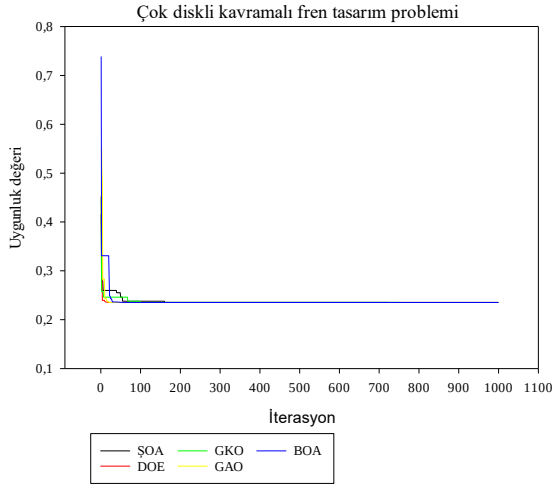
ŞOA, DOE, GKO, GAO, BOA yöntemlerinin çok diskli kavrama tasarım problemi üzerindeki performanslarının karşılaştırmalı değerleri Tablo 7'de verilmiştir. Tablo 7'de kullanılan yöntemlerin en iyi, ortalama, en kötü ve standart sapma değerleri bulunmaktadır. Ayrıca kullanılan yöntemlerin bu problem üzerindeki 30 çalıştırma sonucundaki en iyi değere bağlı karar değişkenleri Tablo 8'de verilmiştir. Tablo 7 incelendiğinde en iyi, ortalama, en kötü değerleri üzerinden kıyaslama yapıldığında DOE, GAO ve BOA'nın en iyi değer bazında diğer yöntemlerden daha üstün olduğu, ortalama değer bazında inceleme yapıldığında DOE'nin daha başarılı olduğu görülmektedir. Kullanılan yöntemlerin çok diskli kavrama tasarım problemi üzerindeki yakınsama grafiği Şekil 9'de gösterilmiştir.

Tablo 7. Çok diskli kavrama fren tasarım probleminin farklı metasezgisel yöntemler üzerindeki performanslarının istatistiksel analizi.

Algoritma	En iyi değer	Ortalama	En kötü değer	Standart sapma
ŞOA	0.235243	0.235616	0.236596	0.00033
DOE	**0.235242**	**0.235242**	0.235242	1.69E-16
GKO	0.235243	0.235276	0.235429	3.84E-05
GAO	**0.235242**	0.235392	0.239724	0.000804
BOA	**0.235242**	0.235243	0.235244	3.17E-07

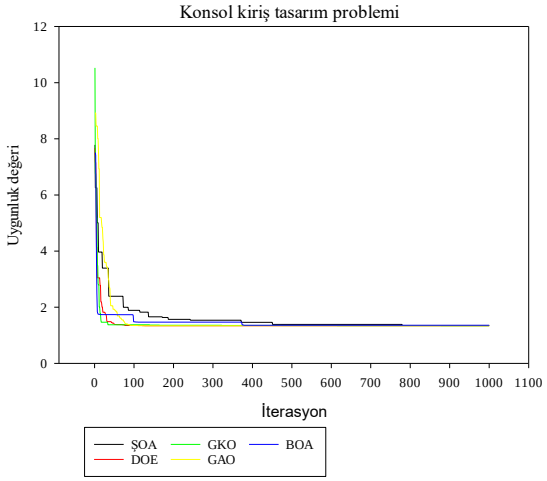
Tablo 8. Çok diskli kavrama fren tasarım probleminin aldığı en iyi değere göre karar değişkenleri.

Algoritma	Parametre değerleri					f_{min}
	x_1	x_2	x_3	x_4	x_5	
ŞOA	69.9999	90.0000	1.0000	24.16887	2.0000	0.235243
DOE	70.0000	90.0000	1.0000	999.87226	2.0000	0.235242
GKO	69.9999	90.0000	1.0000	782.2061	2.0000	0.235243
GAO	70.0000	90.0000	1.0000	664.3994	2.0000	0.235242
BOA	70.0000	90.0000	1.0000	326.8523	2.0000	0.235242



Şekil 9. Kullanılan yöntemlerin çoklu disk kavrama fren tasarım problemi üzerindeki yakınsama grafiği.

ŞOA, DOE, GKO, GAO, BOA yöntemlerinin konsol kiriş tasarım problemi üzerindeki performanslarının karşılaştırmalı değerleri Tablo 9'da verilmiştir. Tablo 9'da kullanılan yöntemlerin en iyi, ortalama, en kötü ve standart sapma değerleri bulunmaktadır. Ayrıca kullanılan yöntemlerin bu problem üzerindeki 30 çalıştırma sonucundaki en iyi değere bağlı karar değişkenleri Tablo 10'da verilmiştir. Tablo 9 incelendiğinde en iyi, ortalama, en kötü değerleri üzerinden kıyaslama yapıldığında DOE'nin en iyi değer bazında ve ortalama değer bazında diğer yöntemlerden daha başarılı olduğu görülmektedir. Kullanılan yöntemlerin çoklu disk kavramalı tasarım problemi üzerindeki yakınsama grafiği Şekil 10'da gösterilmiştir.



Şekil 10. Kullanılan yöntemlerin konsol kiriş tasarım problemi üzerindeki yakınsama grafiği.

Tablo 9. Konsol kiriş tasarım probleminin farklı metasezgisel yöntemler üzerindeki performanslarının istatistiksel analizi.

Algoritma	En iyi değer	Ortalama	En kötü değer	Standart sapma
ŞOA	1.34819	1.376082	1.413464	0.014557
DOE	**1.339957**	**1.339964**	1.339988	6.9E-06
GKO	1.339974	1.340038	1.340212	5.4E-05
GAO	1.340111	1.340851	1.342382	0.000514
BOA	1.356559	1.516685	2.144997	0.175521

Tablo 10. Konsol kiriş tasarım probleminin aldığı en iyi değere göre karar değişkenleri.

Algoritma	Karar değişkenleri					f_{min}
	x_1	x_2	x_3	x_4	x_5	
ŞOA	6.518072	5.247723	4.225419	3.522175	2.092226	1.34819
DOE	6.012134	5.310015	4.496472	3.500466	2.154584	1.339957
GKO	6.000132	5.299499	4.501253	3.516721	2.156344	1.339974
GAO	6.007219	5.299754	4.473095	3.493925	2.202152	1.340111
BOA	5.491058	5.819903	4.428201	3.886131	2.114441	1.356559

5 Sonuç

Gerçek dünya mühendislik tasarım problemlerini çözmek, yeni geliştirilen her metasezgisel algoritmanın verimliliğini değerlendirirken zor bir problem olarak kabul edilmektedir. Bu problemler kinematik koşullar, üretim gereksinimleri ve performans üzerindeki çeşitli doğrusal olmayan kısıtlamalara ek olarak, birden fazla amaç ve karma değişkenler içerebilmektedir. Problemin boyutu ve zorluğu arttıkça algoritmaların başarısı da

düşebilmektedir. Literatürde mühendislik tasarım problemlerini verimli bir şekilde ele almak için metasezgisel optimizasyon yöntemleri ve onların çeşitli versiyonları önerilmektedir. Bu çalışmada; ŞOA, DOE, GKO, GAO ve BOA olmak üzere beş popüler ve güncel metasezgisel algoritma kullanılarak beş gerçek dünya mühendislik tasarım problemi çözülmüştür. Bu yöntemlerin performansları, çözüm kalitesi ve yakınsama hızı açısından karşılaştırılmıştır. Genel bir değerlendirme yapıldığında; beş mühendislik tasarım problemlerinin dördünde ortalama değer bakımından en iyi sonuçları DOE'nin ürettiğini en iyi değer bakımından değerlendirdiğimizde problemlerin tamamında DOE'nin en iyi sonucu ürettiği sonucunu çıkarabiliriz. GAO'da en iyi değer bakımından DOE'den sonra mühendislik tasarım problemlerinin üçünde en iyi sonucu vererek en iyi ikinci yöntem olmuştur. Son olarak, bu çalışmada kullanılan algoritmaların otomotiv ve diğer endüstriler gibi farklı gerçek dünya optimizasyon problemlerini çözmek için önemli alternatifler olduğu sonucuna varılabilir.

Açıklamalar

Bu çalışmada Etik kurul Onay belgesine gerek yoktur.

Kaynaklar

[1] 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*, 110, 151-166.

[2] Altay, E. V., & 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.

[3] Altay, E. V., & Altay, O. Güncel metasezgisel optimizasyon algoritmalarının CEC2020 test fonksiyonları ile karşılaştırılması. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 12(5), 729-741.

[4] Varol Altay, E., & Alatas, B. (2020). Bird swarm algorithms with chaotic mapping. *Artificial Intelligence Review*, 53(2), 1373-1414.

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

[6] Bingol, H., & Alatas, B. (2016). Chaotic league championship algorithms. *Arabian journal for science and engineering*, 41(12), 5123-5147.

[7] Khishe, M., & Mosavi, M. R. (2020). Chimp optimization algorithm. *Expert systems with applications*, 149, 113338.

[8] Faramarzi, H. (2020). Faramarzi A., Heidarinejad M., Stephens B., Mirjalili S. *Equilibrium optimizer: A novel optimization algorithm*, Knowledge-Based Systems, 191.

[9] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, 46-61.

[10] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249.

[11] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, 95, 51-67.

[12] He, X., & Zhou, Y. (2018). Enhancing the performance of differential evolution with covariance matrix self-adaptation. *Applied Soft Computing*, 64, 227-243.

[13] Dhiman, G. (2021). ESA: a hybrid bio-inspired metaheuristic optimization approach for engineering problems. *Engineering with Computers*, 37(1), 323-353.

[14] Dhiman, G. (2021). SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications. *Knowledge-Based Systems*, 222, 106926.

[15] Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with computers*, 29(1), 17-35.