Mühendislik Bilimleri ve Tasarım Dergisi
9(4), 1200 – 1211, 2021
e-ISSN: 1308-6693

Journal of Engineering Sciences and Design
DOI: 10.21923/jesd.930887

Araştırma Makalesi Research Article

ÇOK AMAÇLI MÜHENDİSLİK TASARIMI VE KISITLI PROBLEMLER İÇİN HİBRİT BİRÇOK AMAÇLI OPTİMİZASYON ALGORİTMASI

Murat KARAKOYUN[1*], Halife KODAZ[2]

[1] Necmettin Erbakan Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Konya, Türkiye
[2] Konya Teknik Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar Mühendisliği, Konya, Türkiye

Keywords

Çok Amaçlı Optimizasyon, Gri Kurt Optimizasyonu, Mühendislik Tasarımı, Pareto Teoremi.

Abstract

Gerçek dünya problemlerine bakıldığında çoğunun birden fazla hedefi gerçekleştirmeye yönelik olduğu görülmektedir. Bu problemlerin çözümü için kullanılan birçok klasik yöntem mevcuttur. Klasik yöntemlerin çözüm geliştirme noktasında farklı sebeplerden dolayı eksik kalması araştırmacıları farklı yaklaşımlar geliştirmeye yöneltmiştir. Genellikle doğada sürü halinde yaşayan hayvanların veya farklı yaşam alanlarına sahip bitkilerin davranışlarından esinlenilerek geliştirilen doğa esinli algoritmalar bu yaklaşımlardan bir tanesi olmuştur. Bu çalışmada, tek amaçlı problemlerin çözümü için geliştirilmiş olan kurbağa sıçrama (SFLA) ve gri kurt optimizasyonu (GWO) algoritmaları hibrit bir şekilde kullanılarak çok amaçlı optimizasyon problemlerine uygulanmıştır. Önerilen algoritma bazı çok amaçlı mühendislik tasarımı ve çok amaçlı kısıtlı problemlerin üzerinde uygulanmıştır. Önerilen algoritmanın performansı NSGA-II, IBEA, MOCell ve PAES algoritmalarının performansı ile kıyaslanmıştır. Performans karşılaştırma metriği olarak HV, IGD, Spread ve Epsilon metrikleri kullanılmıştır. Performans analizi; elde edilen ortalama sonuçlar, Friedman sıralama testi ve Wilcoxon anlamlılık testi ile yapılmıştır. Deneysel sonuçlar, önerilen algoritmanın diğer algoritmalardan daha başarılı sonuçlar ürettiğini göstermiştir.

A HYBRID MULTI OBJECTIVE OPTIMIZATION ALGORITHM FOR MULTI OBJECTIVE ENGINEERING DESIGN AND CONSTRAINED PROBLEMS

Anahtar Kelimeler

Multi-Objective Optimization, Grey Wolf Optimizer, Engineering Design, Pareto Theorem.

Öz

When looking to the real-world problems, it is seen that many of them are aimed at achieving more than one goal. The deficiency of the classical methods at the point of developing solutions due to different reasons has led researchers to develop different approaches. Nature-inspired algorithms developed by taking inspiration from the behavior of animals that generally live with a swarm in nature or plants with different habitats have been one of these approaches. In this study, shuffled frog leaping (SFLA) and gray wolf optimizer (GWO) algorithms, which developed for solving single-objective problems, are applied to multi-objective optimization problems in a hybrid manner. The proposed algorithm has been applied on some multi-objective engineering design and multi-objective constrained problems. The performance of the proposed algorithm has been compared with the performance of NSGA-II, IBEA, MOCell and PAES algorithms. HV, IGD, Spread and Epsilon metrics are used as performance comparison metrics. Performance analysis was performed using the average results obtained, Friedman ranking test and Wilcoxon significance test. Experimental results have shown that the proposed algorithm generates more successful results than other algorithms.

Alıntı / Cite

Karakoyun, M., Kodaz, H., (2021). Çok Amaçlı Mühendislik Tasarımı ve Kısıtlı Problemler için Hibrit Birçok Amaçlı Optimizasyon Algoritması, Journal of Engineering Sciences and Design, 9(4), 1200-1211.

[*] İlgili yazar/Corresponding author: mkarakoyun@erbakan.edu.tr

Yazar Kimliği / Author ID (ORCID Number)	Makale Süreci / Article Process	
M. Karakoyun, 0000-0002-0677-9313	**Başvuru Tarihi / Submission Date**	01.05.2021
H. Kodaz, 0000-0001-8602-4262	**Revizyon Tarihi / Revision Date**	20.08.2021
	Kabul Tarihi / Accepted Date	31.08.2021
	Yayım Tarihi / Published Date	20.12.2021

1. Giriş (Introduction)

Optimizasyon, ele alınan problem için belirtilen kısıtlamaları sağlayacak şekilde en iyi çözümü üretme sürecidir. Bir problemin çözüme kavuşabilmesi öncelikli olarak matematiksel modelinin çıkarılmasına bağlıdır. Matematiksel model, bir problemin değişken, kısıt, amaç fonksiyonu gibi unsurlarının matematiksel olarak ifade edilmesidir. Bu matematiksel modelde hedef, amaç fonksiyonunu minimize veya maksimize eden değişken değerlerini, belirtilen kısıtlar çerçevesinde elde etmektir. Matematiksel modeli basit, değişken sayısı ve kısıtları az olan problemler için uygun çözümlerin elde edilmesi zor olmamaktadır. Ancak matematiksel model karmaşıklaştıkça ve karar değişken sayısı arttıkça problemin çözüme kavuşması farklı açılardan daha zor bir hale gelmektedir. Bu tür karmaşık problemlerde çözüm süresinin çok uzun olması ve uygun çözüme ulaşılamaması en önemli sorunların başında gelmektedir. Karşılaşılan bu sorunlar araştırmacıları farklı çözümler üretmeye teşvik etmiştir. Tarih boyunca insanoğlu için esin kaynağı olan doğa ve doğadaki canlılar bu noktada da ilham vermiştir. Birçok araştırmacı, doğa ve canlıların yaşamlarını model alarak farklı özellikteki algoritmalar geliştirmiştir. Literatüre bakıldığında Yapay Alg Algoritması (*Artificial Algae Algorithm, AAA*) (Uymaz ve ark., 2015), Gri Kurt Optimizasyonu (*Grey Wolf Optimizer, GWO*) (Mirjalili ve ark., 2014), Kurbağa Sıçrama Algoritması (*Shuffled Frog Leaping Algorithm, SFLA*) (Eusuff ve ark., 2006), Yapay Arı Kolonisi (*Artificial Bee Colony, ABC*) (Karaboga, 2005), Parçacık Sürü Optimizasyonu (*Particle Swarm Optimization, PSO*) (Kennedy ve Eberhart, 1995), Genetik Algoritma (*Genetic Algorithm, GA*) (Holland, 1992) gibi birçok doğa esinli algoritmanın geliştirildiği görülmektedir. Bununla beraber araştırmacılar, performansı arttırmak amacıyla bu algoritmaların çoğunu modifiye ederek farklı türevlerini elde etmişlerdir. Çalışmalar sonucunda doğa esinli algoritmaların optimizasyon problemleri üzerinde kısa sürelerde kabul edilebilir çözümler ürettiği görülmüştür.

Optimizasyon problemleri amaç fonksiyonları sayısı bakımından iki ana başlıkta toplanabilir. Şayet optimize edilecek problem bir tane amaç fonksiyonuna sahip ise tek amaçlı, birden fazla amaç fonksiyonuna sahip ise çok amaçlı optimizasyon problemi olarak adlandırılmaktadır. Tek amaçlı optimizasyon problemlerinde, problemin sahip olduğu tek amaç fonksiyonunun en uygun değerini veren karar değişkenlerine sahip tek çözüm elde edilmeye çalışılmaktadır. Çok amaçlı optimizasyon problemlerinde ise birden fazla amaç için uygun olan bir çözüm kümesi sunulmaktadır. Birden fazla amaç fonksiyonu olan çok amaçlı optimizasyon problemlerinde amaç fonksiyonları birbiri ile uyum içinde olabilir veya çatışma durumunda olabilmektedir. Amaç fonksiyonları arasındaki bu ilişkiye göre problemin ele alınışı ve işlenişi farklılık gösterebilmektedir. Amaç fonksiyonlarının uyum içinde olduğu problemlerde, bir amaç için uygun bir çözüm sunan karar değişkenleri diğer amaç için de uygunluk göstermektedir. Bu tür problemler uygun yöntemler kullanılarak tek amaçlı bir problem haline getirilebilir ve tek amaçlı optimizasyon için geliştirilen algoritmalar ile çözüme kavuşturulabilir. Problem tek amaçlı olarak ele alındığından sonuç, tüm amaç fonksiyonları için uygun değere karşılık gelen karar değişkenlerine sahip bir çözüm olacaktır. Ancak gerçek dünya problemlerinin çoğu amaçları birbiri ile çakışan çok amaçlı problemlerdir ve bu problemlerin her bir amaç fonksiyonu için uygun değer veren tek bir çözüm söz konusu değildir. Bu tür problemlerde tek çözüm yerine bir çözüm kümesi sunulmaktadır. Bu kümedeki çözümler amaç fonksiyonları arasında bir denge sağlamayı hedeflemektedir. Uygun çözümlerden oluşması hedeflenen bu kümeyi elde edebilmek adına birçok yaklaşım ve algoritma önerilmiştir. Bunlardan bazıları direk olarak çok amaçlı problemlerin çözülmesi için geliştirilmişken bir kısmı da tek amaçlı optimizasyon yaklaşımlarının modifiye edilmesiyle elde edilmiştir. Her problemin çözümü için başarılı sonuç veren bir algoritmanın olamaması sebebiyle geçmişten günümüze birçok yeni yaklaşımın önerildiği görülmektedir. Literatüre bakıldığında, optimizasyon yaklaşımlarının çok amaçlı olarak ele alınması ve uygulanması ile ilgili sayısız örneğin olduğu görülmektedir. Son yıllarda literatüre kazandırılmış bu çalışmalardan bazıları: Yapay Alg algoritmasının çok amaçlı optimizasyon alanındaki çalışmalarına MOAAA (Babalik ve ark., 2018) ve MO-AAA (Tawhid ve Savsani, 2018); Karınca Aslanı optimizasyonu algoritmasının çok amaçlı versiyonu (Mirjalili ve ark., 2017), Kumawat ve arkadaşları tarafından önerilen Balina Optimizasyonu Algoritmasının çok amaçlı versiyonu (Kumawat ve ark., 2017) ve son olarak Şahin optimizasyonu algoritmasının çok amaçlı versiyonu (Du ve ark., 2020) örnek olarak gösterilebilir.

Bu çalışmada tek amaçlı optimizasyon problemlerinin çözümü için geliştirilmiş olan SFLA ve GWO algoritmaları hibrit bir şekilde ele alınarak bazı çok amaçlı mühendislik tasarımı ve çok amaçlı kısıtlı problemlerin çözümü için kullanılmıştır. MOSG olarak isimlendirilen önerilen algoritmanın performansı literatürde sıklıkla kullanılan 4 adet çok amaçlı optimizasyon algoritmasının performansı ile kıyaslanmıştır. Algoritmaların performansı 4 farklı metrik ile değerlendirilmiştir. Ayrıca Friedman istatistiksel testi ve Wilcoxon anlamlılık testleri ile sonuçlar analiz

edilmiştir. Elde edilen deneysel sonuçlar önerilen algoritmanın, karşılaştırma yapılan algoritmalardan daha başarılı sonuçlar ürettiğini göstermiştir.

2. Çok Amaçlı Optimizasyon (Multi-objective Optimization)

Çok amaçlı optimizasyon problemlerinde temel amaç birden fazla amacı birlikte optimize eden karar değişkenlerine ulaşmaktır (Osyczka, 1985). Çok amaçlı bir optimizasyon probleminin matematiksel olarak ifade edilmesi Eşitlik (1) ile gösterilmiştir.

$$(min/max)\ y = f(x) = \{f_1(x), f_2(x),..., f_M(x)\} \quad (1)$$

Kısıtlar:

$$g(x) \leq \{g_1(x), g_2(x),..., g_i(x)\} \leq 0 \quad (2)$$

$$h(x) = \{h_1(x), h_2(x),..., h_j(x)\} = 0 \quad (3)$$

$$X = [x_1, x_2, ..., x_B]^T \quad (4)$$

$$A_b \leq x_b \leq U_b,\ b = 1,2, ..., B \quad (5)$$

Eşitlik (1)'de verilen y=$f(x)$ dizisi M-amaçlı bir vektör, m=1,2, ..., M olmak üzere $f_m(x)$ minimize/maksimize edilmek istenen $m.$ amaç fonksiyonu, $g(x)$ ve $h(x)$ sırasıyla sağlanması gereken eşitsizlik ve eşitlik kısıtları dizisi, X karar değişkeni vektörü ve A_b ile U_b sırasıyla ($b.$karar değişkeni için) alt ve üst sınırları temsil etmektedir.

Tek amaçlı optimizasyon problemlerinde bilindiği üzere algoritmalar bir amacın en iyi sağlandığı durumu elde etmeye çalışmaktadır. Elde edilen iki aday çözümün kalite kıyaslanması karar değişkenlerine bağlı olarak elde edilen amaç fonksiyonlarının değerleri karşılaştırıldığında rahatlıkla yapılabilmektedir. Bu şekilde minimizasyon problemlerinde amaç fonksiyonu değeri daha düşük olan aday çözümün daha iyi olduğu veya maksimizasyon problemleri için tam tersi söylenebilmektedir. Ancak durum çok amaçlı optimizasyon problemleri için daha karışık bir hal almaktadır. Birden fazla amacı olan ve genellikle amaçları birbiri ile çatışan çok amaçlı optimizasyon problemlerinde elde edilen aday çözümlerin kalite açısından kıyaslanması tek amaçlı problemlere göre daha zor olmaktadır. Çok amaçlı optimizasyondaki bu karmaşıklığı gidermek adına Goldberg, Pareto teoreminin bu problemlere uygulanabilirliğini dile getirmiştir (Golberg, 1989).

Kökeni ve çıkış noktası ekonomik refah üzerine olan Pareto teoremine göre: Toplumdaki bazı bireylerin yaşam kalitesini daha iyi hale getirip bazı bireylerin durumunu daha kötü hale getiriyorsa yapılan değişim/değişimlerin refahı ne yönde (olumlu veya olumsuz) etkilediği net bir şekilde söylenemez. Refahın artması, hiçbir bireyin durumunu kötü yapmadan bazı bireylerin durumunu iyi hale getiren değişimlerin yapılması ile söz konusu olabilir. Ayrıca hiçbir bireyin durumu kötüleşmeden bir bireyin dahi durumunu iyileştirme olanağı yok ise bu toplumda maksimum refah seviyesine ulaşılmıştır (Sağ, 2008; Özkış ve Babalık, 2017).

Bu aşamadan sonra bu teoremin temel alındığı yeni yaklaşımlar geliştirilmiştir. Ayrıca çoğu tek amaçlı optimizasyon algoritmasının çok amaçlı problemlere uyarlanması da bu teoreme dayanmaktadır. Çok amaçlı algoritmalar elde ettikleri aday çözümlerin kalite kıyasını yapmak için Pareto teoremine başvurmaktadır. Pareto teoreminin çok amaçlı optimizasyon algoritmalarına uygulanışı dört temel kurala dayandırılmıştır (Coello ve ark., 2007; Deb, 2011; Özkış ve Babalık, 2017). Ω arama uzayı, $X\ (x_1, x_2, ..., x_B)$ ve $Y\ (y_1, y_2, ..., y_B)$ arama uzayında birer uygun aday çözüm ($X, Y \in \Omega$) olmak üzere:

i) ***Baskınlık (dominance)*****:** X aday çözümü amaç fonksiyonlarının hiçbirinde Y aday çözümünden daha kötü olmadan eğer en az bir amaç fonksiyonu için Y aday çözümünden daha iyi bir değere sahip ise X aday çözümü Y aday çözümünü baskılar (bastırır veya domine eder) denilmektedir. Bu durum $X \prec Y$ şeklinde gösterilmektedir. M amaç fonksiyonu sayısı olmak üzere Eşitlik (6) baskınlık durumunun matematiksel ifadesini vermektedir.

$$Eğer \begin{matrix} \forall i & \in & \{1,2,\dots,M\} f_i(X) \leq f_i(Y) \\ \wedge\ \exists j & \in & \{1,2,\dots,M\} f_j(X) < f_j(Y)' \end{matrix} \qquad X \prec Y \quad (6)$$

Aday çözümlerin her zaman birbirini bastırması beklenilemez. X ve Y aday çözümleri amaç fonksiyonlarının en az birer tanesi için birbirlerine üstünlük sağlamışlarsa birbirini bastıramama (*non-dominated*) durumu söz konusudur. Bu durumda bu iki aday çözüm birbirini bastıramamış denmektedir.

ii) ***Pareto optimal*:** X aday çözümü arama uzayındaki hiçbir çözüm tarafından bastırılamıyorsa bu çözüme Pareto optimal çözüm denmektedir. Bu durum matematiksel olarak: $Eğer\ \neg\exists Z \in \Omega: Z \prec X$ gösterilmekte ve bu durum sağlandığında X aday çözümü Pareto optimal bir çözüm olarak değerlendirilmektedir.

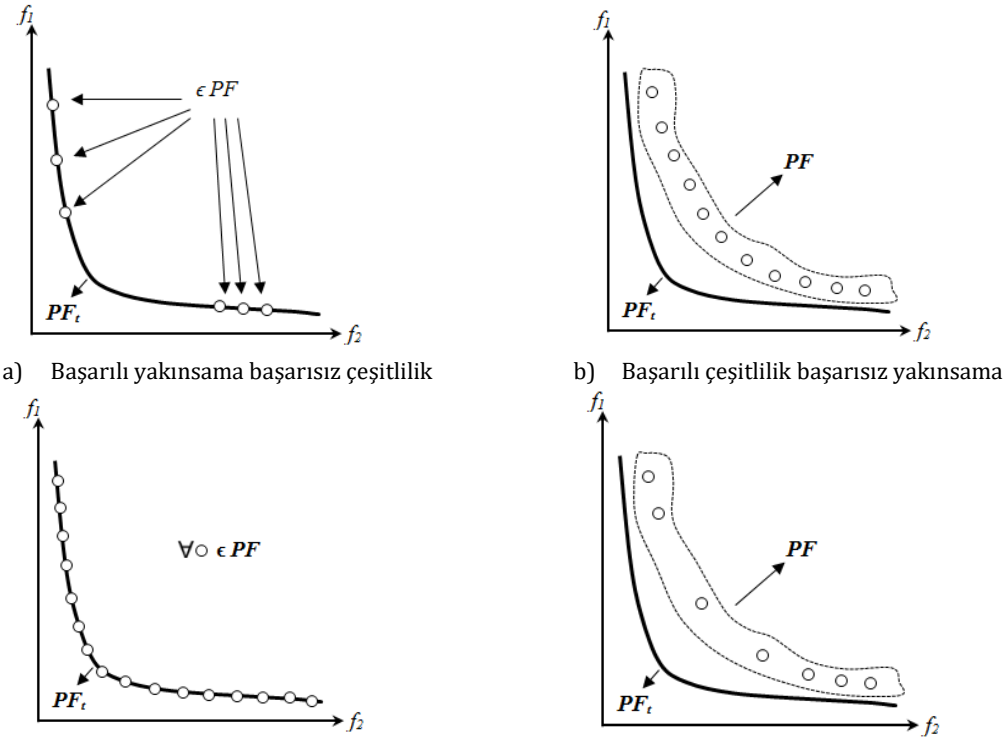
iii) ***Pareto optimal set (PS)*:** Arama uzayındaki bastırılamayan (Pareto optimal) çözümlerden oluşan kümeye denmektedir. PS matematiksel olarak Eşitlik (7) ile gösterilmiştir.

$$PS = \{X \in \Omega | \neg\exists Z \in \Omega: Z \prec X\} \quad (7)$$

iv) ***Pareto optimal yüzey (Pareto front, PF)*:** Pareto optimal çözümlerin karar değişkenlerine karşılık gelen amaç uzayındaki konumların oluşturduğu yüzeye denmektedir. PF matematiksel olarak Eşitlik (8) ile gösterilmiştir.

$$PF = \{F(X) | X \in PS\} \quad (8)$$

Çok amaçlı optimizasyon algoritmaları problemin çıktısı olarak en iyi tek çözüm yerine birbirini bastıramayan bir çözümler kümesi sunmaktadır. Söz konusu algoritma için Pareto optimal yüzeyi (*Pareto front, PF*) oluşturan bu çözümlerin gerçek Pareto optimal (*Pareto front true, PF_t*) yüzeyi en başarılı şekilde tahmin etmesi istenmektedir. Çok amaçlı bir algoritmanın elde ettiği çözümler kümesinin başarısını ölçmek için kullanılan temel iki ölçüt vardır: Çeşitlilik (*diversity*) ve yakınsama (*convergence*). *PF*'yi oluşturan çözümlerin PF_t çözümleri boyunca geniş bir dağılıma sahip olması çeşitlilik açısından başarı anlamına gelmektedir. Yakınsama başarısında ise *PF* çözümlerinin PF_t çözümlerine yakın olması beklenmektedir. Şekil 1 çok amaçlı bir algoritmanın elde ettiği *PF* çözümlerinin olası durumlarını göstermektedir.



a) Başarılı yakınsama başarısız çeşitlilik
b) Başarılı çeşitlilik başarısız yakınsama
c) İdeal yakınsama ve çeşitlilik
d) Başarısız yakınsama ve çeşitlilik

Şekil 1. *PF* için yakınsama ve çeşitlilik durumları (Possible convergence and diversity situations for PF)

3. Materyal ve Metot (Material and Method)

3.1. Problem Seti (Problem Set)

Bu çalışmada kullanılan problem seti, farklı karar değişkeni sayılarına sahip on adet mühendislik tasarım ve kısıtlı problemlerden oluşmaktadır. Bu problem setinde bulunan problemlerin özellikleri Tablo 1'de verilmiştir. *Water* problemi hariç geriye kalan problemlerin amaç sayısı ikidir. *Four-bar truss* ve *gear train* problemleri kısıtsız iken diğer tüm problemlerin kısıtları mevcuttur (Durillo ve Nebro, 2011; Özkış, 2017).

Tablo 1. Çok amaçlı mühendislik tasarımı ve kısıtlı problemler (Multi-objective engineering design and constrained problems)

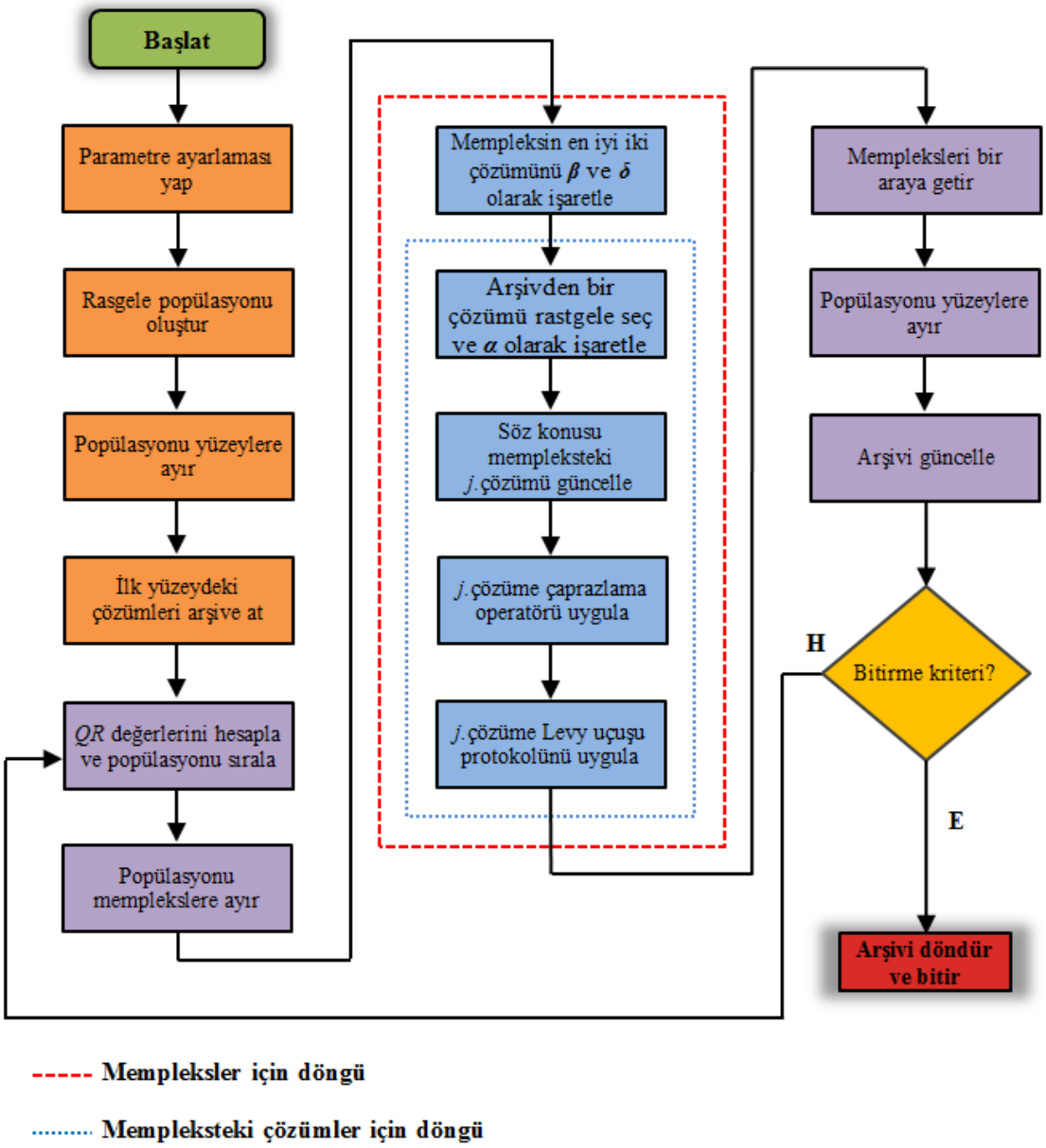
Problem	Boyut	#Amaç	Kısıt
Binh2	2	2	Var
ConstrEx	2	2	Var
Srinivas	2	2	Var
KITA	2	2	Var
Water	3	5	Var
Four-bar truss tasarım	4	2	Yok
Disk brake tasarım	4	2	Var
Centilever beam tasarım	2	2	Var
Spring	3	2	Var
Gear train	4	2	Yok

3.2. Önerilen MOSG Algoritması (The Proposed MOSG Algorithm)

MOSG (Karakoyun ve ark., 2020) algoritması SFLA'nın mempleks yapısının GWO algoritması üzerine uygulanması sonucu elde edilen hibrit bir algoritmadır. Algoritma çok amaçlı optimizasyon problemlerinin çözümü için önerilmiştir. Temel amaç mempleks stratejisi ile arama uzayını detaylı ve etkin bir şekilde tarayarak en uygun çözüm kümesini tespit etmektir. Bununla beraber algoritmanın başarısını arttırmak amacıyla bazı modifikasyon ve yaklaşımlar da kullanılmıştır. GWO algoritmasındaki alfa çözümünün konum güncellemesine etkisini arttırmak amacıyla dinamik katsayı yaklaşımı önerilmiştir. Ayrıca yakınsama ve yerel en iyi çözüme takılma sorunlarını aşmak için çaprazlama operatörü ve Levy uçuşu stratejisi kullanılmıştır.

Önerilen algoritma ilk olarak kullanılan parametrelerin ayarlanması ile başlar. Bu aşamada algoritmada kullanılan popülasyon büyüklüğü, iterasyon sayısı, mempleks sayısı vb. gibi değişkenlerin değerleri atanır. Daha sonra arama uzayı sınırları içerisinde belirtilen popülasyon büyüklüğü kadar rastgele çözüm üretilerek ilk popülasyon oluşturulur. Arama uzayında elde edilen çözümlerin amaç uzayındaki karşılıkları, yani uygunluk değerleri hesaplanır. Popülasyonun yüzeylere ayrılıp bastırılamayan çözümler ile ilk arşivin oluşturulması için hızlı bastırılamayan yaklaşım uygulanır. Burada arşiv, algoritmanın mevcut aşamaya kadar elde ettiği çözümlerin içerisinde bastırılamayan çözümlerin listesini temsil etmektedir. Çözümler yüzeylere ayrılıp ilk arşiv elde edildikten sonra popülasyonun mempleklere ayrılması için sırasıyla kalabalık mesafeler ve ardından kalite sıralaması yaklaşımı uygulanır. Popülasyondaki tüm çözümlerin *QR* değeri hesaplandıktan sonra sıralama işlemi gerçekleştirilir. Sıralama işleminden sonra popülasyon mempleklere ayrılır. Her bir memplekste çözümlerin güncellenmesi için gerekli döngüler başlatılır. İlk döngüde her mempleksin ele alınması sağlanırken, içerideki ikinci döngüde ise söz konusu memplekste çözümlerin işlenmesi yapılır. GWO algoritmasında konum güncelleme aşamasında alfa, beta ve delta bireylerden oluşan lider takım referans olarak alınmaktadır. MOSG algoritmasında her mempleks kendi lider takımına sahiptir. Bu lider takımın oluşturulması *alfa* çözümün arşivden seçilmesi ve memplekste en iyi *QR* değerlerine sahip ilk iki çözümün sırasıyla *beta* ve *delta* olarak işaretlenmesi şeklinde olmaktadır. Bundan sonra çaprazlama operatörü ve ardından Levy uçuşu protokolü uygulanır. Bu şekilde her mempleks ve içerisindeki her çözüm için konum güncellemeleri yapıldıktan sonra mempleksler bir araya getirilerek yeni popülasyon elde edilir. Bu aşamadan sonra arşiv güncellenir. Arşiv güncellemesi, elde edilen yeni çözümler ile arşivin hızlı bastırılamayan sıralama prosedürüne sokularak bastırılamayan çözümlerin elde edilmesi ve arşive aktarılması şeklinde olmaktadır. Bu çalışmada arşiv büyüklüğü popülasyon büyüklüğüne eşit olarak belirlenmiştir. Dolayısıyla arşiv güncellemesi yapıldığında arşivdeki çözüm sayısı maksimum sınırı aştığında *QR* değerlerine göre en iyi çözümlerin arşivde kalması sağlanmaktadır. Arşiv güncellemesi yapıldıktan sonra bitirme kriterinin sağlanıp sağlanmadığı kontrol edilir. Eğer bitirme kriteri henüz sağlanmadıysa popülasyon tekrar sıralanır ve mempleklere ayrılır. Bu adımlar bitirme kriteri sağlanana kadar devam ettirilir. Bitirme kriteri sağlandığında ise elde edilen arşiv (*PF*) algoritmanın çıktısı olarak verilir. MOSG algoritması için daha detaylı bilgi

edinmek istenirse algoritmanın ilk önerildiği (Karakoyun ve ark., 2020) çalışması incelenebilir. MOSG algoritmasının akış diyagramı Şekil 2'de verilmiştir.



Şekil 2. Önerilen algoritmanın (MOSG) akış diyagramı (Flowchart of the proposed (MOSG) algorithm)

4. Deneysel Sonuçlar (Experimental Results)

Mühendislik tasarımı ve kısıtlı fonksiyonlardan oluşan problem setinde MOSG algoritmasının performansı dört farklı çok amaçlı algoritma (NSGA-II (Srinivas ve Deb, 1994), IBEA (Zitzler ve Künzli, 2004), MOCell (Nebro ve ark., 2007), PAES (Knowles ve Corne, 1999)) ile karşılaştırılmıştır. Bu problem seti üzerinde yapılan deneylerde MAXFes (maksimum uygunluk fonksiyonu çağrılma sayısı) değeri 5000 ile sınırlı tutulmuştur. Bunun nedeni kullanılan problemlerin karar değişken sayılarının az olmasıdır. Algoritmalar her problem için 100 tekrar ile çalıştırılmıştır. Performans karşılaştırması için dört metrik (HV (Zitzler ve Thiele, 1999), IGD (Coello ve Cortés, 2005), Spread (Li, 2003), Epsilon (Durillo ve Nebro, 2011)) kullanılmıştır. Metriklerden HV ve IGD algoritmaların hem yakınsama hem de çeşitlilik başarısını ölçerken, Spread metriği çeşitlilik başarısını ve Epsilon metriği yakınsama başarısını ölçmektedir. Bu metriklerden HV metriğinin değeri maksimum olması istenirken diğer üç metriğin değerinin minimum olması istenmektedir. HV metriğinin hesaplanmasında kullanılan denklem Eşitlik (9)'da verilmiştir.

$$HV = volume\left(\bigcup_{i=1}^{|PF|} v_i\right) \quad (9)$$

HV değerinin hesaplanmasında PF_t çözümlerine ihtiyaç duyulmamaktadır. Ancak bazı çalışmalarda bu metriğin normalize edilmiş değeri kullanılmaktadır. Normalize *HV* değerinin hesaplanacağı durumlarda ise PF_t çözümlerine ve bu çözümlerden elde edilecek olan HV_t değerine ihtiyaç duyulmaktadır. HV_t Eşitlik (9) ile hesaplanır ve bu aşamada *PF* çözümleri yerine PF_t çözümleri kullanılır. Normalize edilmiş *HV (HVN)* değeri Eşitlik (10) ile hesaplanmaktadır.

$$HV_N = \frac{HV}{HV_t} \quad (10)$$

IGD metriğinin hesaplanmasında kullanılan denklem Eşitlik (11)'de verilmiştir. Eşitlik (11)'deki *n* değeri PF_t çözümlerinin sayısını ve d_i ise PF_t 'deki i. çözüm ile kendisine en yakın olan *PF* çözümü arasındaki Öklit mesafesini göstermektedir.

$$IGD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \quad (11)$$

x: *PF* çözümler kümesinde bir çözümü
d(x, PF): *x* çözümü ile en yakın çözümler arasındaki en düşük mesafe
$\overline{d}$: *d(x, PF)* mesafelerinin ortalaması
e_i: i. amaç fonksiyonu için PF_t çözümler kümesine ait uç çözüm
d(ei, PF): *PF* ve PF_t kümelerinin uç çözümleri arasındaki mesafe olmak üzere Spread metriğinin hesaplanmasında Eşitlik (12) kullanılmaktadır.

$$Spread = \frac{\sum_{i=1}^{m} d(e_i, PF) + \sum_{x \in PF} |d(x, PF) - \overline{d}|}{\sum_{i=1}^{m} d(e_i, PF) + |PF| * \overline{d}} \quad (12)$$

$$d(x, PF) = \min_{y \in PF, y \neq x} \|f(x) - f(y)\|^2 \quad (13)$$

$$\overline{d} = \frac{1}{|PF_t|} \sum_{x \in PF_t} d(x, PF) \quad (14)$$

PF çözümlerinin her birinin tek tek ele alınarak PF_t çözümlerinden en az birini bastırması için gereken minimum değişiminin hesaplanması ile elde edilen Epsilon ($\boldsymbol{\varepsilon}$) metriğinin değeri Eşitlik (15) kullanılarak hesaplanır.

$$\varepsilon = inf_{\varepsilon \in R^+} \{\forall \vec{p} \in PF_t, \exists \vec{\alpha} \in PF : \vec{\alpha} < \varepsilon \vec{p}\} \quad (15)$$

Algoritmaların problem seti üzerinde 100 tekrarlı çalıştırma sonucu ürettikleri ortalama ve standart sapma değerleri Tablo 2-5'te verilmiştir. Kullanılan problem setindeki her bir problem için en iyi ortalama değeri bulan algoritmanın bulduğu değer koyu gri ile işaretlenirken ikinci en iyi ortalama değer ise açık gri ile işaretlenmiştir.

Tablo 2'deki sonuçlar algoritmaların problem seti üzerindeki çalışmalarının HV metriğindeki karşılıklarını göstermektedir. HV metriği üzerinde elde edilen sonuçlara bakıldığında MOSG algoritmasının 10 problemden 9 tanesi için en iyi ortalama değerleri üretirken 1 tane problem için de en iyi ikinci ortalama değeri ürettiği görülmektedir. Diğer algoritmaların bu problemler üzerinde en iyi ve ikinci en iyi ortalama değerleri bulma sayıları tablodaki sıralamalarına göre sırasıyla [1/5], [1/1], [1/5] ve [0/0] şeklindedir. HV metriği üzerinde elde edilen ortalama sonuçlara bakıldığında MOSG algoritmasının diğer algoritmalara bariz bir üstünlük sağladığı görülmektedir.

Tablo 2. HV metriği için algoritmaların ortalama ve standart sapma değerleri (The mean and standard deviation values of the algorithms for the HV metric)

Algoritma	**NSGA-II**		**IBEA**		**MOCell**		**PAES**		**MOSG**	
Problem	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.
Binh2	8.10E-01	3.2E-04	8.08E-01	1.8E-03	8.11E-01	7.1E-05	8.07E-01	3.1E-03	8.12E-01	4.3E-05
ConstrEx	7.73E-01	4.5E-04	7.59E-01	5.1E-03	7.71E-01	3.0E-03	7.36E-01	3.9E-02	7.73E-01	2.2E-03
Srinivas	5.32E-01	3.2E-04	5.34E-01	1.6E-04	5.34E-01	1.4E-04	5.29E-01	8.7E-04	5.35E-01	6.9E-05
KITA	6.42E-01	1.4E-03	6.36E-01	2.0E-03	6.41E-01	4.0E-03	6.22E-01	1.7E-02	6.45E-01	1.8E-03
Water	3.90E-01	9.7E-03	2.60E-01	5.3E-02	4.03E-01	8.5E-03	2.91E-01	8.3E-02	4.32E-01	1.7E-03
FourBarTruss	1.82E-01	9.4E-04	1.86E-01	3.0E-04	1.85E-01	5.8E-04	1.57E-01	4.6E-02	1.85E-01	2.8E-04
DiskBrake	8.73E-01	1.4E-03	8.69E-01	3.7E-03	8.71E-01	4.1E-03	8.70E-01	5.0E-03	8.77E-01	3.8E-04
CentileverBeam	8.71E-01	2.9E-04	7.91E-01	3.0E-02	8.72E-01	3.4E-05	8.70E-01	3.7E-04	8.72E-01	2.5E-05
Spring	7.46E-01	6.5E-03	6.70E-01	3.8E-02	7.29E-01	1.6E-02	6.99E-01	3.2E-02	7.63E-01	9.0E-04
GearTrain	9.02E-01	1.3E-02	9.00E-01	6.5E-03	8.93E-01	1.9E-02	7.85E-01	1.3E-01	9.05E-01	1.1E-06

Tablo 3'teki sonuçlar algoritmaların problem seti üzerindeki çalışmalarının IGD metriğindeki karşılıklarını göstermektedir. Tablo 3'teki sonuçlara bakıldığında MOSG algoritmasının 10 problemden 9 tanesi için en iyi ortalama değerleri üretirken 1 tane problem için de en iyi ikinci ortalama değeri ürettiği görülmektedir. Diğer algoritmaların bu problemler üzerinde ortalama en iyi ve ikinci en iyi değerleri bulma sayıları tablodaki sıralamalarına göre sırasıyla [1/5], [1/0], [0/5] ve [0/0] şeklindedir. IGD metriği üzerinde elde edilen ortalama sonuçlara bakıldığında MOSG algoritmasının diğer algoritmalardan daha iyi ortalama sonuçlar ürettiği açıkça görülmektedir.

Tablo 3. IGD metriği için algoritmaların ortalama ve standart sapma değerleri (The mean and standard deviation values of the algorithms for the IGD metric)

Algoritma	**NSGA-II**		**IBEA**		**MOCell**		**PAES**		**MOSG**	
Problem	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.
Binh2	1.50E-04	8.9E-06	1.42E-03	4.2E-04	1.11E-04	3.2E-06	6.85E-04	5.9E-04	1.07E-04	2.0E-06
ConstrEx	2.01E-04	3.5E-05	4.40E-03	1.4E-03	3.21E-04	2.3E-04	2.40E-03	2.2E-03	2.47E-04	1.1E-04
Srinivas	1.44E-04	7.7E-06	1.25E-04	8.9E-06	1.04E-04	4.9E-06	2.12E-04	3.5E-05	9.85E-05	1.9E-06
KITA	5.44E-04	6.2E-05	9.65E-04	1.9E-04	6.43E-04	2.6E-04	1.63E-03	1.1E-03	5.19E-04	1.4E-04
Water	2.51E-03	1.4E-04	1.52E-02	1.9E-03	2.35E-03	1.9E-04	9.62E-03	3.0E-03	1.99E-03	1.1E-04
FourBarTruss	2.02E-02	2.1E-05	2.01E-02	1.3E-05	2.02E-02	3.4E-05	2.22E-02	4.0E-03	2.01E-02	5.8E-06
DiskBrake	8.60E-04	5.8E-04	4.92E-03	3.9E-04	2.08E-03	1.1E-03	2.60E-03	1.1E-03	2.89E-04	3.0E-05
CentileverBeam	3.09E-04	2.3E-05	1.14E-02	1.5E-03	2.24E-04	4.2E-06	4.15E-04	5.4E-05	2.23E-04	4.8E-06
Spring	2.85E-03	1.1E-03	1.40E-02	4.2E-03	5.83E-03	3.1E-03	9.36E-03	4.9E-03	1.43E-03	8.6E-04
GearTrain	1.31E-02	7.0E-03	3.28E-02	6.2E-03	2.10E-02	9.1E-03	3.71E-02	2.0E-02	6.82E-03	3.6E-03

Tablo 4'teki sonuçlar MOSG ve diğer algoritmaların problem seti üzerindeki çalışmalarının Spread metriğindeki karşılıklarını göstermektedir. Spread metriği üzerinde elde edilen sonuçlara bakıldığında MOSG algoritmasının 10 problemden 8 tanesi için en iyi ortalama değerleri üretirken 1 tane problem için de en iyi ikinci ortalama değeri ürettiği görülmektedir. Diğer algoritmaların bu problemler üzerinde en iyi ve ikinci en iyi ortalama değerleri bulma sayıları tablodaki sıralamalarına göre sırasıyla [0/1], [0/0], [2/8] ve [0/0] şeklindedir. Spread metriği üzerinde elde edilen ortalama sonuçlara bakıldığında MOSG algoritmasının diğer algoritmalara bariz bir üstünlük sağladığı görülmektedir. Bununla beraber Spread metriği üzerinde genellikle iyi sonuçlar elde eden MOCell algoritması, MOSG algoritmasından sonra diğer üç algoritmaya üstünlük sağlamıştır.

Tablo 4. Spread metriği için algoritmaların ortalama ve standart sapma değerleri (The mean and standard deviation values of the algorithms for the Spread metric)

Algoritma	**NSGA-II**		**IBEA**		**MOCell**		**PAES**		**MOSG**	
Problem	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.
Binh2	3.98E-01	3.5E-02	5.40E-01	1.4E-02	1.59E-01	1.7E-02	6.79E-01	4.7E-02	1.29E-01	1.6E-02
ConstrEx	4.53E-01	3.2E-02	1.11E+00	4.1E-02	3.80E-01	7.9E-02	8.84E-01	8.3E-02	2.91E-01	5.2E-02
Srinivas	4.00E-01	3.4E-02	3.61E-01	2.9E-02	1.17E-01	1.5E-02	6.28E-01	5.0E-02	7.64E-02	1.1E-02
KITA	6.74E-01	1.3E-01	1.19E+00	4.2E-02	6.09E-01	1.6E-01	1.17E+00	9.4E-02	5.81E-01	1.6E-01
Water	5.60E-01	5.2E-02	7.51E-01	1.3E-01	5.57E-01	4.2E-02	6.60E-01	6.3E-02	5.79E-01	3.7E-02
FourBarTruss	6.57E-01	1.9E-02	6.58E-01	1.6E-02	5.47E-01	1.8E-02	8.18E-01	4.5E-02	5.20E-01	8.7E-03
DiskBrake	5.35E-01	5.4E-02	7.54E-01	4.2E-02	5.10E-01	4.9E-02	7.84E-01	5.9E-02	3.23E-01	3.8E-02
CentileverBeam	4.03E-01	3.6E-02	6.71E-01	4.1E-02	1.28E-01	1.6E-02	6.40E-14	4.4E-02	1.24E-01	1.7E-02
Spring	1.26E+00	6.3E-02	1.17E+00	1.1E-01	6.42E-01	8.3E-02	1.45E+00	9.3E-02	3.29E-01	3.5E-02
GearTrain	1.44E+00	6.4E-02	1.43E+00	8.3E-02	7.72E-01	3.8E-02	1.31E+00	1.2E-01	7.76E-01	3.0E-02

Tablo 5'teki sonuçlar algoritmaların problem seti üzerindeki çalışmalarının Epsilon metriğindeki karşılıklarını göstermektedir. Tablo 5'teki sonuçlara bakıldığında MOSG algoritmasının 10 problemden 7 tanesi için en iyi ortalama değerleri üretirken 3 tane problem için de en iyi ikinci ortalama değerleri ürettiği görülmektedir. Diğer algoritmaların bu problemler üzerinde ortalama en iyi ve ikinci en iyi değerleri bulma sayıları tablodaki sıralamalarına göre sırasıyla [2/3], [1/1], [0/3] ve [0/0] şeklindedir. Epsilon metriği üzerinde elde edilen ortalama sonuçlara bakıldığında MOSG algoritmasının diğer algoritmalardan daha iyi ortalama sonuçlar ürettiği görülmektedir.

Tablo 5. Epsilon metriği için algoritmaların ortalama ve standart sapma değerleri (The mean and standard deviation values of the algorithms for the Epsilon metric)

Algoritma	**NSGA-II**		**IBEA**		**MOCell**		**PAES**		**MOSG**	
Problem	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.	Ortalama	Std. Sap.
Binh2	1.01E+00	2.5E-01	1.08E+02	2.8E-01	4.69E-01	5.5E-02	1.60E+00	1.4E+00	4.06E-01	1.9E-02
ConstrEx	1.65E-02	2.5E-03	3.95E-01	1.2E-01	2.25E-02	8.2E-03	1.08E-01	1.6E-01	1.84E-02	7.6E-03
Srinivas	3.21E+00	5.7E-01	2.21E+00	4.4E-01	1.79E+00	3.0E-01	5.49E+00	1.7E+00	1.42E+00	1.6E-01
KITA	4.72E-02	8.2E-03	7.38E-02	1.1E-02	5.32E-02	1.6E-02	1.50E-01	8.5E-02	4.79E-02	1.3E-02
Water	8.23E+04	1.8E+04	1.09E+06	3.6E+05	5.63E+04	1.5E+04	8.52E+05	6.9E+05	2.39E+04	2.5E+03
FourBarTruss	6.90E+01	5.7E+00	7.03E+01	1.3E+01	9.14E+01	2.5E+01	2.06E+02	1.4E+02	6.36E+01	0.0E+00
DiskBrake	7.34E-02	1.9E-02	2.81E-01	8.4E-02	1.40E-01	7.8E-02	1.21E-01	1.3E-01	3.12E-02	4.6E-03
CentileverBeam	7.00E-04	6.8E-04	2.94E-04	8.0E-05	5.59E-04	5.7E-04	4.52E-03	3.5E-03	3.09E-04	2.9E-04
Spring	2.58E+03	3.0E+03	2.71E+04	1.0E+04	8.59E+03	7.9E+03	1.61E+04	1.2E+04	2.41E+02	2.1E+02
GearTrain	7.22E-01	1.0E+00	6.65E-01	8.4E-01	1.56E+00	1.5E+00	6.29E+00	5.6E+00	4.30E-06	2.9E-05

Tablo 2-5'teki sonuçlar MOSG ve diğer algoritmaların, dört metrik üzerinde, ortalama değerlerini göstermektedir. Algoritmaların her problem üzerindeki 100 bağımsız çalıştırılması değerlendirilerek elde edilen Friedman testi sonuçları ise Tablo 6'da verilmiştir. Her bir metrik için en iyi Friedman (Friedman, 1937) testi sonucuna sahip algoritmanın değeri koyu gri, ikinci en iyi değere sahip algoritmanın değeri ise açık gri ile işaretlenmiştir. Tablo 6'daki sonuçlara bakıldığında MOSG algoritmasının dört metrik için de en iyi Friedman sıralamasına sahip olduğu görülmektedir. Öte yandan NSGA-II ve MOCell algoritmaları HV ve IGD metrikleri için ikinci en iyi dereceyi paylaşırken Spread metriğinde MOCell, Epsilon metriğinde NSGA-II ikinci en iyi Friedman skoruna sahiptir.

Tablo 6. Algoritmaların dört metrik üzerindeki Friedman test sonuçları (Friedman test results of algorithms on four metrics)

	HV	**IGD**	**Spread**	**Epsilon**
NSGA-II	3.30	2.60	3.30	2.50
IBEA	2.10	4.20	4.20	3.70
MOCell	3.30	2.60	1.80	3.00
PAES	1.40	4.40	4.40	4.50
MOSG	4.90	1.20	1.30	1.30

MOSG algoritması ve diğer algoritmaların dört metrik için elde ettikleri ortalama değerler ve bu değerlere bağlı Friedman sonuçları önceki tablolarda gösterilmiştir. Elde edilen bu ortalama sonuçların birbirinden farklı ve anlamlı sonuçlar olup olmadığını kontrol etmek amacıyla Wilcoxon (Johnson ve Bhattacharyya, 2019) sıra toplamı testi uygulanmıştır. %95 güven aralığı ile yapılan Wilcoxon testi sonuçları HV, IGD, Spread ve Epsilon metrikleri için sırasıyla Tablo 7-10'da verilmiştir. MOSG algoritması ve kıyas yapılan algoritma için anlamlı bir sonuç elde edilmesi durumunda "+" simgesi ile aksi takdirde "-" simgesi ile işaretlenmiştir.

Tablo 7'deki sonuçlar algoritmaların HV metriği üzerinde elde ettikleri değerlere Wilcoxon testi uygulanması sonucu elde edilen bulgulardır. Tablodaki sonuçlara bakıldığında MOSG algoritmasının problem setindeki 10 problem için elde ettiği değerlerin tamamının diğer algoritmaların ürettiği değerlerden farklı ve anlamlı olduğu görülmektedir.

Tablo 7. HV metriği için Wilcoxon testi sonuçları (Wilcoxon test results for the HV metric)

MOSG vs.	**NSGA-II**		**IBEA**		**MOCell**		**PAES**	
	p-değeri	S	*p*-değeri	S	*p*-değeri	S	*p*-değeri	S
Binh2	7.07E-18	+	1.01E-17	+	7.07E-18	+	7.07E-18	+
ConstrEx	7.07E-18	+	7.80E-11	+	7.07E-18	+	7.07E-18	+
Srinivas	9.26E-03	+	7.15E-09	+	2.07E-17	+	2.20E-17	+
KITA	7.07E-18	+	7.97E-18	+	7.07E-18	+	7.07E-18	+
Water	7.07E-18	+	2.80E-03	+	3.13E-17	+	5.64E-17	+
FourBarTruss	1.49E-16	+	1.86E-14	+	6.12E-18	+	6.14E-18	+
DiskBrake	4.20E-13	+	5.40E-11	+	7.97E-18	+	7.07E-18	+
CentileverBeam	7.07E-18	+	7.07E-18	+	7.07E-18	+	7.07E-18	+
Spring	7.07E-18	+	7.07E-18	+	7.07E-18	+	7.07E-18	+
GearTrain	7.07E-18	+	7.07E-18	+	7.07E-18	+	7.07E-18	+

Tablo 8'deki, IGD metriği için yapılan Wilcoxon testi sonuçlarına bakıldığında MOSG algoritmasının problem setindeki 10 problemden; NSGA-II algoritması için 9, IBEA algoritması için 8, MOCell ve PAES algoritmaları için 10 tanesinden farklı ve anlamlı sonuçlar ürettiği görülmektedir.

Tablo 8. IGD metriği için Wilcoxon testi sonuçları (Wilcoxon test results for the IGD metric)

MOSG vs.	**NSGA-II**		**IBEA**		**MOCell**		**PAES**	
	p-değeri	S	*p*-değeri	S	*p*-değeri	S	*p*-değeri	S
Binh2	7.07E-18	+	4.12E-10	+	7.07E-18	+	7.07E-18	+
ConstrEx	7.07E-18	+	4.06E-01	-	7.07E-18	+	7.07E-18	+
Srinivas	1.12E-01	-	7.82E-02	-	7.07E-18	+	1.55E-14	+
KITA	7.07E-18	+	7.07E-18	+	7.07E-18	+	7.07E-18	+
Water	7.07E-18	+	1.01E-17	+	1.31E-15	+	7.97E-18	+
FourBarTruss	3.24E-08	+	4.55E-15	+	6.82E-18	+	4.07E-17	+
DiskBrake	8.71E-03	+	2.35E-04	+	2.54E-16	+	4.46E-17	+
CentileverBeam	1.15E-11	+	1.13E-16	+	8.46E-18	+	2.47E-17	+
Spring	7.07E-18	+	6.31E-13	+	7.07E-18	+	7.07E-18	+
GearTrain	1.29E-17	+	3.02E-15	+	7.07E-18	+	7.07E-18	+

Tablo 9'daki, Spread metriği için yapılan Wilcoxon testi sonuçlarına bakıldığında MOSG algoritmasının problem setindeki 10 problemden; IBEA algoritması için 7, NSGA-II, MOCell ve PAES algoritmaları için tamamından farklı ve anlamlı sonuçlar ürettiği görülmektedir.

Tablo 9. Spread metriği için Wilcoxon testi sonuçları (Wilcoxon test results for the Spread metric)

MOSG vs.	NSGA-II		IBEA		MOCell		PAES	
	p-değeri	S	*p*-değeri	S	*p*-değeri	S	*p*-değeri	S
Binh2	7.07E-18	+	8.59E-12	+	7.07E-18	+	7.07E-18	+
ConstrEx	7.07E-18	+	1.99E-01	-	7.07E-18	+	7.07E-18	+
Srinivas	1.07E-16	+	1.12E-09	+	7.07E-18	+	7.07E-18	+
KITA	7.07E-18	+	8.99E-18	+	7.07E-18	+	7.07E-18	+
Water	7.07E-18	+	3.10E-12	+	7.07E-18	+	7.07E-18	+
FourBarTruss	6.85E-18	+	2.68E-01	-	6.85E-18	+	6.85E-18	+
DiskBrake	7.39E-04	+	4.14E-01	-	7.07E-18	+	7.97E-18	+
CentileverBeam	7.07E-18	+	7.07E-18	+	7.07E-18	+	7.07E-18	+
Spring	7.07E-18	+	1.54E-17	+	7.07E-18	+	7.07E-18	+
GearTrain	5.89E-03	+	5.29E-03	+	1.76E-13	+	4.28E-11	+

Tablo 10'daki, Epsilon metriği için yapılan Wilcoxon testi sonuçlarına bakıldığında MOSG algoritmasının problem setindeki 10 problemden; algoritmaların tablodaki dizilişlerine göre sırasıyla 8, 10, 9 ve 10 tanesinden farklı ve anlamlı sonuçlar ürettiği görülmektedir.

Tablo 10. Epsilon metriği için Wilcoxon testi sonuçları (Wilcoxon test results for the Epsilon metric)

MOSG vs.	NSGA-II		IBEA		MOCell		PAES	
	p-değeri	S	*p*-değeri	S	*p*-değeri	S	*p*-değeri	S
Binh2	7.07E-18	+	6.53E-14	+	7.07E-18	+	7.07E-18	+
ConstrEx	6.20E-04	+	1.06E-02	+	3.40E-01	-	8.38E-16	+
Srinivas	4.80E-01	-	2.87E-03	+	7.07E-18	+	3.46E-14	+
KITA	7.07E-18	+	2.33E-17	+	7.07E-18	+	7.07E-18	+
Water	3.31E-20	+	3.31E-20	+	3.31E-20	+	3.31E-20	+
FourBarTruss	4.24E-16	+	4.16E-16	+	5.14E-18	+	5.87E-18	+
DiskBrake	5.79E-01	-	3.58E-02	+	3.99E-13	+	4.46E-17	+
CentileverBeam	8.56E-15	+	1.10E-13	+	7.07E-18	+	1.29E-17	+
Spring	7.07E-18	+	2.22E-11	+	2.07E-17	+	7.07E-18	+
GearTrain	7.07E-18	+	7.97E-18	+	7.07E-18	+	7.07E-18	+

Problem seti için yapılan Wilcoxon sıra toplamı testi sonuçlarına genel olarak bakıldığında MOSG algoritmasının ürettiği sonuçların kıyaslama yapılan algoritmaların ürettiği sonuçlardan farklı ve anlamlı olduğu açık bir şekilde söylenebilir.

5. Sonuç ve Tartışmalar (Result and Discussion)

Bu çalışmada önerilen MOSG algoritması çok amaçlı mühendislik tasarımı ve kısıtlı problemlerden oluşan problem setine uygulanmıştır. MOSG algoritmasının performansı çok amaçlı optimizasyon için önerilen NSGA-II, IBEA, MOCell ve PAES algoritmalarının performansı ile kıyaslanmıştır. Algoritmaların performansı farklı özellikteki, algoritmaların çeşitlilik ve yakınsama başarısını ölçen HV, IGD, Spread ve Epsilon metrikleri ile ölçülmüştür. Algoritmaların performans analizi elde edilen ortalama sonuçlar, Friedman sıralama testi ve Wilcoxon anlamlılık testi kullanılarak yapılmıştır. Elde edilen deneysel sonuçlar önerilen algoritmanın diğer algoritmalardan tüm metriklerde daha başarılı sonuçlar ürettiğini göstermiştir. Bu da önerilen algoritmanın, çok amaçlı optimizasyon problemlerindeki başarı kıstaslarından olan gerek yakınsama gerekse çeşitlilik yönünden umut verici bir algoritma olduğunu göstermektedir. Elde edilen sonuçlar, önerilen algoritmanın çok amaçlı optimizasyon alanında başarılı olduğunu göstermiştir. Budan sonraki çalışmalarda önerilen algoritma farklı özellikteki çok amaçlı optimizasyon problemlerine uygulanabilir. Ayrıca algoritmanın yapısı paralel programlama teknolojisine uygun olduğundan bu teknoloji kullanılarak algoritma daha hızlı bir hale getirilebilir.

Çıkar Çatışması (Conflict of Interest)

Yazarlar tarafından herhangi bir çıkar çatışması beyan edilmemiştir. No conflict of interest was declared by the authors.

Kaynaklar (References)

Babalik, A., Ozkis, A., Uymaz, S. A. ve Kiran, M. S., 2018, A multi-objective artificial algae algorithm, *Applied Soft Computing*, 68, 377-395.

Coello, C. A. C. ve Cortés, N. C., 2005, Solving multiobjective optimization problems using an artificial immune system, *Genetic programming and evolvable machines*, 6 (2), 163-190.

Coello, C. A. C., Lamont, G. B. ve Van Veldhuizen, D. A., 2007, Evolutionary algorithms for solving multi-objective problems, Springer, p.

Deb, K., 2011, Multi-objective optimisation using evolutionary algorithms: an introduction, In: Multi-objective evolutionary optimisation for product design and manufacturing, Eds: Springer, p. 3-34.

Du, P., Wang, J., Hao, Y., Niu, T. ve Yang, W., 2020, A novel hybrid model based on multi-objective Harris hawks optimization algorithm for daily PM2. 5 and PM10 forecasting, *Applied Soft Computing*, 96, 106620.

Durillo, J. J. ve Nebro, A. J., 2011, jMetal: A Java framework for multi-objective optimization, *Advances in engineering software*, 42 (10), 760-771.

Eusuff, M., Lansey, K. ve Pasha, F., 2006, Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization, *Engineering optimization*, 38 (2), 129-154.

Friedman, M., 1937, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, *Journal of the american statistical association*, 32 (200), 675-701.

Golberg, D. E., 1989, Genetic algorithms in search, optimization, and machine learning, *Addion wesley*, 1989 (102), 36.

Holland, J. H., 1992, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, MIT press, p.

Johnson, R. A. ve Bhattacharyya, G. K., 2019, Statistics: principles and methods, John Wiley & Sons, p.

Karaboga, D., 2005, An idea based on honey bee swarm for numerical optimization, *Citeseer*.

Karakoyun, M., Ozkis, A. ve Kodaz, H., 2020, A new algorithm based on gray wolf optimizer and shuffled frog leaping algorithm to solve the multi-objective optimization problems, *Applied Soft Computing*, 96, 106560.

Kennedy, J. ve Eberhart, R., 1995, Particle swarm optimization, *Proceedings of ICNN'95-international conference on neural networks*, 1942-1948.

Knowles, J. ve Corne, D., 1999, The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation, *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, 98-105.

Kumawat, I. R., Nanda, S. J. ve Maddila, R. K., 2017, Multi-objective whale optimization, *Tencon 2017-2017 ieee region 10 conference*, 2747-2752.

Li, X., 2003, A non-dominated sorting particle swarm optimizer for multiobjective optimization, *Genetic and evolutionary computation conference*, 37-48.

Mirjalili, S., Jangir, P. ve Saremi, S., 2017, Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems, *Applied Intelligence*, 46 (1), 79-95.

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

Nebro, A. J., Durillo, J. J., Luna, F., Dorronsoro, B. ve Alba, E., 2007, Design issues in a multiobjective cellular genetic algorithm, *International Conference on Evolutionary Multi-Criterion Optimization*, 126-140.

Osyczka, A., 1985, Multicriteria optimization for engineering design, In: Design optimization, Eds: Elsevier, p. 193-227.

Özkış, A. ve Babalık, A., 2017, A novel metaheuristic for multi-objective optimization problems: The multi-objective vortex search algorithm, *Information Sciences*, 402, 124-148.

Özkış, A., 2017, Girdap arama ve yapay alg algoritmalarının çok amaçlı optimizasyon problemlerine uyarlanması, Doktora Tezi, *Selçuk Üniversitesi Fen Bilimleri Enstitüsü*.

Sağ, T., 2008, Çok kriterli optimizasyon için genetik algoritma yaklaşımları, *Selçuk Üniversitesi Fen Bilimleri Enstitüsü*.

Srinivas, N. ve Deb, K., 1994, Muiltiobjective optimization using nondominated sorting in genetic algorithms, *Evolutionary computation*, 2 (3), 221-248.

Tawhid, M. A. ve Savsani, V., 2018, A novel multi-objective optimization algorithm based on artificial algae for multi-objective engineering design problems, *Applied Intelligence*, 48 (10), 3762-3781.

Uymaz, S. A., Tezel, G. ve Yel, E., 2015, Artificial algae algorithm (AAA) for nonlinear global optimization, *Applied Soft Computing*, 31, 153-171.

Zitzler, E. ve Künzli, S., 2004, Indicator-based selection in multiobjective search, *International conference on parallel problem solving from nature*, 832-842.

Zitzler, E. ve Thiele, L., 1999, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE transactions on Evolutionary Computation*, 3 (4), 257-271.