Mühendislik Bilimleri ve Tasarım Dergisi
8(5), 156-172, 2020
e-ISSN: 1308-6693

Araştırma Makalesi

Journal of Engineering Sciences and Design
DOI: 10.21923/jesd.829508

Research Article

Özel Sayı: Uluslararası Mühendislikte Yapay Zekâ ve Uygulamalı Matematik Konferansı (UMYMK 2020)

Special Issue: International Conference on Artificial Intelligence and Applied Mathematics in Engineering (ICAIAME 2020)

ARZ-TALEP TABANLI OPTİMİZASYON ALGORİTMASININ FDB YÖNTEMİ İLE İYİLEŞTİRİLMESİ: MÜHENDİSLİK TASARIM PROBLEMLERİ ÜZERİNE KAPSAMLI BİR ARAŞTIRMA

Mehmet KATI[1], Hamdi Tolga KAHRAMAN[2]

[1]mkati@havelsan.com.tr, *Havelsan, Ankara-Türkiye*
[2]htolgakahraman@ktu.edu.tr, *Karadeniz Teknik Üniversitesi, Yazılım Mühendisliği Bölümü, Trabzon-Türkiye*

Anahtar Kelimeler

Arz-talep tabanlı arama algoritması
Uygunluk-mesafe dengesi seçim yöntemi
Komşuluk araması ve çeşitlilik
Meta-sezgisel optimizasyon
Mühendislik tasarım problemleri

Öz

Bu makale çalışmasında son zamanlarda geliştirilmiş güncel bir meta-sezgisel arama (MSA) yöntemi olan arz-talep tabanlı (Supply-Demand-Based Optimization, SDO) algoritmasının iyileştirilmiş bir versiyonu geliştirilmektedir. SDO'da arz-talep süreçlerini daha etkili bir şekilde modelleyebilmek amacıyla arama sürecine rehberlik eden çözüm adayları uzaklık-uygunluk dengesi (fitness-distance balance, FDB) yöntemi kullanılarak belirlenmiştir. Geliştirilen FDB-tabanlı SDO algoritmasının performansını test etmek ve doğrulamak amacıyla güncel bir karşılaştırma problemleri havuzu olan CEC 2017 kullanılmıştır. Bu havuzda dört farklı tipte ve otuz adet kısıtsız test problemi bulunmaktadır. Önerilen algoritmanın farklı tiplerdeki ve farklı boyutlardaki arama uzaylarındaki performansını test etmek ve doğrulamak için test problemleri 3/50/100 boyutta tasarlanmıştır. Ayrıca, önerilen FDB-SDO varyasyonlarının kısıtlı mühendislik problemlerindeki performanslarını test etmek ve doğrulamak için ise 20 adet mühendislik tasarım problemi kullanılmıştır. Her iki deneysel çalışmadan elde edilen veriler parametrik olmayan istatistiksel test yöntemleri kullanılarak analiz edilmiştir. Analiz sonuçlarına göre kısıtlı/kısıtsız, tekmodlu/çokmodlu/melez/komposizyon problem türlerinde ve farklı boyutlarda olmak üzere tüm deneysel çalışmalarda FDB-SDO varyasyonlarının baz algoritmaya kıyasla üstün bir performans sergilemişlerdir. FDB yönteminin tatbik edilmesiyle birlikte SDO algoritmasının prematüre yakınsama problemi çözülmüştür. Önerilen FDBSDO algoritması hassas arama yapabilme, çeşitliliği etkili bir şekilde sağlamaya bilme ve dengeli arama yapabilme yeteneklerine sahiptir. *Önerilen FDBSDO'nun kaynak kodu:* https://www.mathworks.com/matlabcentral/fileexchange/84560-fdbsdo-an-improved-version-of-supply-demand-optimizer

IMPROVING SUPPLY-DEMAND-BASED OPTIMIZATION ALGORITHM WITH FDB METHOD: A COMPREHENSIVE RESEARCH ON ENGINEERING DESIGN PROBLEMS

Keywords

Supply-Demand-Based Optimization
Fitness-distance balance
Exploitation and exploration
Meta-heuristic optimization
Engineering design problems

Abstract

In this study, an improved version of the supply-demand-based optimization (SDO) algorithm, a recently developed meta-heuristic search method, was developed. In order to model the supply-demand processes more effectively in SDO, solution candidates guiding the search process were determined using the fitness-distance balance (FDB) method. In order to test and verify the performance of the developed FDB-based SDO algorithm, CEC 2017, a modern benchmark suite, was used. This suite has four different types and thirty unconstrained test problems. These problems are designed in 30/50/100 dimensions to test and verify the performance of the proposed algorithm in search spaces of different types and dimensions. In addition, twenty engineering design problems were used to test and verify the performance of the proposed FDBSDO variations in constrained engineering design problems.

Data from both experimental studies were analyzed using non-parametric statistical test methods. According to the results of the analysis, FDBSDO variations showed superior performance compared to the base algorithm in all experimental studies, with constrained/unconstrained, unimodal/multi-modal/hybrid/composition problem types and different dimensions. The implementation of the FDB selection method eliminated the premature convergence problem of the SDO algorithm. The proposed FDBSDO algorithm has the ability to sensitively search, effectively provide diversity, and build a strong balance between exploitation-exploration. *Source code of the proposed FDBSDO:* https://www.mathworks.com/matlabcentral/fileexchange/84560-fdbsdo-an-improved-version-of-supply-demand-optimizer

Alıntı / Cite

Kahraman H., Katı M. (2020). Arz-Talep Tabanlı Optimizasyon Algoritmasının FDB Yöntemi ile İyileştirilmesi: Mühendislik Tasarım Problemleri Üzerine Kapsamlı Bir Araştırma, Mühendislik Bilimleri ve Tasarım Dergisi, 8(5), 156-172.

Yazar Kimliği / Author ID (ORCID Number)	Makale Süreci / Article Process	
Hamdi Tolga Kahraman, 0000-0001-9985-6324	**Başvuru Tarihi / Submission Date**	21.11.2020
Mehmet Katı, 0000-0003-0723-2739	**Revizyon Tarihi / Revision Date**	21.12.2020
	Kabul Tarihi / Accepted Date	21.12.2020
	Yayım Tarihi / Published Date	29.12.2020

1. Giriş (Introduction)

Meta-sezgisel arama algoritmaları (MSA), optimizasyon problemlerinin çözümlenmesinde kullanılan ve birçoğu topluluk tabanlı ve doğa-esinli olarak geliştirilmiş yöntemlerdir. MSA yöntemlerini geliştirme çabaları 1950'li yıllara dayanmaktadır. Michigan üniversitesinden Prof. John Holland ve öğrencilerinin geliştirdikleri Genetik Algoritma bu çalışmaların miladı niteliğindedir (Holland, 1975).

MSA algoritmalarını geliştirme çalışmaları 90'lı yılların başından itibaren hız kazanmıştır. Tavlama Benzetimi (SA,1987) (Laarhoven, Aarts, 1987), Parçacık Sürü Optimizasyonu (PSO, 1995) (Eberhart, Kennedy, 1995), Diferansiyel Evrim (DE, 1997) (Storn, Price, 1997), Karınca Kolonisi Algoritması (ACO, 1999) (Dorigo, Di Caro, 1999) bu yılların en popüler MSA yöntemleridir. 2000'li yıllardan itibaren ise MSA geliştirme çalışmaları ikiye ayrılarak daha da hızlanmıştır. Bu çalışmalar, doğadaki süreçlerin ve problem çözme davranışlarının taklit edilmesi sayesinde yeni MSA algoritmalarının geliştirilmesi ya da halihazırdaki MSA yöntemlerinin performanslarını iyileştirmek için yeniden tasarlanması şeklinde yürütülmüştür. Harmoni Arama (HS, 2001) (Loganathan vd., 2001), Yapay Arı Kolonisi (ABC, 2007) (Karaboga, Basturk, 2007), Yerçekimsel Arama Algoritması (GSA, 2009) (Rashedi vd., 2009), Guguk Kuşu Arama Algoritması (CS, 2009) (Yang, Deb, 2009), 2000 li yıllarda geliştirilmiş en popüler MSA yöntemlerindendirler. Bunun yanında aynı yıllarda GA, PSO, SA ve ACO algoritmalarının çok sayıda varyasyonları da geliştirilmiştir. Geliştirilen MSA algoritmaları birçok farklı alanda optimizasyon probleminin çözümlenmesi için kullanılmışlardır.

2010 yılı sonrasında MSA'lar üzerine yapılan çalışmalar, her yıl onlarca algoritmanın geliştirilmesi şeklinde gerçekleşmiştir. MSA algoritmalarının geliştirilmesinde, test edilmesinde ve doğrulanmasında karşılaşılan standartsızlıklar geliştirilen yüzlerce algoritmanın birbirlerine kıyasla performanslarının anlaşılamamasına yol açmıştır (Piotrowski vd., 2014; Yang vd., 2019). Bu durum MSA algoritmalarının çalışmalarının yürütülmesi için standartların tanımlanmasını zorunlu kılmıştır. Günümüzde bile tüm kesimler tarafından kabullenilen ve uygulanan standartlar olmasa da özellikle 2005'li yıllardan itibaren MSA çalışmaları için takip edilmesi gereken ilkeler belirlenmeye başlanmıştır (Suganthan vd., 2005; Suganthan vd., 2013; Suganthan vd., 2016). Uluslararası düzenlenen CEC konferanslarında bu konuda epeyce ilerlemeler kaydedilmiştir (Suganthan vd., 2005). CEC konferanslarında, farklı tiplerdeki optimizasyon problemlerinin çözümlenmesi için rakip MHS algoritmalarının katıldığı yarışmalar düzenlenmekte ve bu yarışmalarda deneysel çalışmaların yürütülmelerinde dikkate alınacak şartlar tanımlanmaktadır. 2014 yılındaki CEC konferansı bu konuda bir milat olarak tanımlanabilir. MSA'ların algoritma karmaşıklıklarının hesaplanması, algoritmaların arama süreci yaşam döngülerinde uymaları gereken kuralların tanımlanması, deneysel verilerin istatistiksel analizi için gerekli şartların tanımlanması CEC 2014'te gerçekleşmiştir (Suganthan vd., 2013).

MSA'ler üzerine yapılan tüm bu çalışmalar neticesinde yüzlerce algoritma geliştirilmiş ve geliştirilen bu algoritmalar kullanılarak binlerce optimizasyon problemi çözümlenmiştir. Günümüzde yeni MSA algoritmaları geliştirme çalışmalarından çok daha fazla mevcut MSA'lerin tasarımsal hatalarının ve eksikliklerinin düzeltilerek performanslarının iyileştirilmesi üzerine araştırmalar yürütülmektedir. Bunun nedeni geliştirilen MHS algoritmalarının performanslarının yetersiz ve birbirine benzer olması ve karmaşık problemlerin daha etkili bir şekilde çözümlenebilmesi için daha fazla sayıda yeni yönteme değil mevcutlardan daha güçlü arama yeteneklerine sahip algoritmalara ihtiyaç duyulmasıdır. MSA'ların arama yetenekleri iki görevi dengeli bir şekilde yerine getirmelerine bağlıdır. Bunlar komşuluk araması (hassas arama) ve çeşitliliktir. Arama sürecinde, problemin karakteristiğine bağlı olarak komşuluk araması ve çeşitlilik işlemleri de dinamik ve uyarlanır bir şekilde MSA'lar tarafından yerine getirilmelidir. MSA'larının arama performanslarının iyileştirilmesi için izlenen yol doğadaki işleyişin olabildiğince etkili bir şekilde taklit edilmesidir. Kaos yöntemi (Demir vd., 2020; Anand ve Arora 2020; Sayed vd., 2019; Mozaffari vd., 2019), Levy uçuşu (Barshandeh ve Haghzadeh, 2020), zıtlık-tabanlı öğrenme (Glover ve Hao, 2019; Ibrahim vd., 2019) gibi yöntemler MSA'ların tasarımlarını doğayla uyumlu hale getirmek ve bu yolla performanslarını iyileştirmek amacıyla yaygın bir şekilde kullanılmaktadırlar. Son dönemlerde MSA'larının arama süreci yaşam döngüsünde çözüm adaylarına rehberlik edecek referans konumların etkili bir şekilde belirlenebilmesi için Uygunluk-Uzaklık Dengesi (Fitness-Distance Balance, FDB) (Kahraman vd., 2020) ismi verilen bir seçim yöntemi geliştirilmiştir. FDB seçim yöntemi, topluluk tabanlı arama yöntemleri olan meta-sezgisel arama algoritmalarına (MSA) rehberlik edecek çözüm adaylarını en etkili şekilde belirlemek amacıyla geliştirilmiştir. Bu amaçla tatbik edildiği Symbiotic Organism Search (FDBSOS) ve Stochastic Fractal Search (FDBSFS) algoritmalarının arama performanslarında büyük iyileşmeler sağlamıştır (Kahraman vd. 2020; Aras vd. 2021). Bu özelliği, FDB seçim yönteminin iktisadi gereksinimleri taklit etmek için geliştirilmiş olan SDO algoritmasının (Zhao vd., 2019) tasarımında kullanılması için onu güçlü bir seçenek haline getirmektedir. Bu makale çalışmasında SDO algoritmasının arz-talep dengesini güçlü bir şekilde taklit etmek ve bu yolla algoritmanın performansında iyileşme sağlamak amacıyla FDB seçim yöntemi ile yeniden tasarlanmış SDO varyasyonları geliştirilmiştir. Geliştirilen varyasyonların performanslarını test etmek ve doğrulamak amacıyla güncel ve standartlara göre tasarlanmış CEC 2017 karşılaştırma problemleri havuzunda yer alan 30 test fonksiyonu ve 20 adet kısıtlı mühendislik tasarım problemi kullanılmıştır (Suganthan vd., 2016). Deneysel çalışma sonuçları parametrik olmayan Wilcoxon ve Friedman test yöntemleri kullanılarak analiz edilmiştir (Carrasco vd., 2020, Eftimov vd., 2017). Analiz sonuçları FDB yönteminin SDO algoritmasının arama performansını büyük oranda iyileştirdiğini göstermektedir. Makalenin literatüre katkıları aşağıda sırasıyla verilmektedir:

i) Güncel bir MSA algoritması olan SDO'nun gerçek yaşamdakine yani doğaya daha uygun bir işleyişle yeniden tasarlanması sağlanmıştır.
ii) SDO algoritmasında yaşanan prematüre yakınsama problemi ortadan kaldırılmıştır. Algoritmanın yerel arama ve çeşitlilik yetenekleri iyileştirilmiştir.
iii) Geliştirilen FDB-SDO algoritması küçük/orta/büyük boyutlu arama uzaylarında ve farklı tipteki problemlerde baz modele göre daha güçlü bir arama performansına sahiptir. Üstelik geliştirilen algoritmanın kısıtlı mühendislik tasarım problemlerindeki performansı da iyileştirilmiştir. Tüm bu iyileşmeler deneysel çalışma sonuçlarının analiz edilmesi yoluyla ispatlanmıştır. Böylelikle, farklı özelliklerdeki optimizasyon problemlerinin çözümlenmesinde kullanılabilecek güçlü bir MSA algoritması literatüre sunulmaktadır.

2. Materyal ve Yöntem (Material and Method)

MSA algoritmaları optimizasyon sürecinde esas olarak iki gereksinimi yerine getirmektedirler. Bunlar; arama uzayındaki referans bir konumun yakın komşuluğunda hassas bir şekilde araştırma yapabilmek ve ihtiyaç duyulduğunda arama uzayındaki umut vadeden konumları etkili bir şekilde keşfedebilmektir. MSA algoritmaları bu gereksinimleri yerine getirirken benzer adımlardan oluşan bir arama süreci yaşam döngüsünü takip ederler. Buna göre MSA algoritmalarında arama sürecinin genel adımları Algoritma 1'de verilmektedir.

Algoritma 1. MSA algoritmalarında arama sürecinin genel adımları (Kahraman vd., 2020)
1) Optimizasyon probleminin tanımlanması (amaç fonksiyonu, tasarım parametreleri vd. öğeler)

2) Algoritma parametrelerinin tanımlanması ve çözüm adayları topluluğunun oluşturulması
3) Aday çözümlerin uygunluk değerlerinin hesaplanması
4) Arama Süreci Yaşam Döngüsü
    a. Seçim süreci
    b. Komşuluk araması ve çeşitliliğin sağlanması
    c. Çözüm adayı topluluğunun güncellenmesi
5) Sonlandırma kriteri sağlandı mı?
    a. Hayır (Adım 4'e dön)
    b. Evet (Arama sürecini sonlandır ve en iyi çözüm adayını kaydet)

Algoritma 1'de verilen adımlara göre MSA algoritmalarında ilk üç adım ortaktır. Algoritmaların performansları arasındaki farklılıklar ise dördüncü adımdaki işlemlere bağlıdır. Dördüncü adımda her algoritmanın kendine özgü arama süreci yaşam döngüsü işletilmektedir. Arama süreci yaşam döngüsünün ilk adımında arama sürecine rehberlik edecek olan adaylar seçilir. Seçim süreci popülasyon üyeleri arasından rastgele, aç gözlü, olasılıksal yöntemlerden biri kullanılarak gerçekleştirilebilir. Rastgele seçim yöntemi arama sürecinde çeşitliliğe katkı sunmak amacıyla kullanılır. Ancak özellikle büyük boyutlu ve karmaşıklık düzeyi yüksek olan arama uzaylarında etkisi düşüktür. Aç gözlü yöntem ile popülasyon üyeleri arasından en başarılı olanın seçilir. Başarı ölçütü ise çözüm adaylarının uygunluk değerleridir. Örneğin ABC'de elit arılar (Karaboga, Basturk, 2007), ALO'da elit karınca (Mirjalili, 2015), ASO'da en iyi parçacık (Zhao vd., 2019), EFO'da pozitif yüklü parçacıklar (Jawawi vd., 2016), COA'da alpha birey (Pierezan ve Coelho, 2018), GWO'da alfa, beta, gama bireyleri (Mirjalili, 2014) açgözlü yöntem kullanılarak seçilirler. Olasılıksal seçim yöntemi, rastgele ve açgözlü yöntemlerin karmasıdır. Çözüm adaylarının uygunluk değerleri ile doğru orantılı olacak şekilde seçilme olasılıkları belirlenir. Olasılıksal seçim yönteminin en yaygın bilinen iki tekniği rulet ve ikili turnuvadır. Rulet tekerleğinin her bir parçası temsil ettiği bireyin uygunluk değeri ile doğru orantılı olacak şekilde tasarlanır. Tekerlek üzerinden rastgele seçilen parça hangi bireye aitse o seçilmiş olur. İkili turnuva tekniğinde ise popülasyondan rastgele seçilen iki birey arasından uygunluk değeri büyük olan seçilir. Olasılıksal seçim yöntemi Genetik Algoritma 'da (Holland, 1975) ve bu evrimsel esaslı algoritmaların varyasyonlarında ebeveyn seçiminde kullanılır. MSA algoritmalarında seçim sürecinden sonra komşuluk araması ve çeşitlilik işlemleri yerine getirilir. Seçim sürecinde belirlenen bireyler referans (rehber) alınarak hassas ve etkili bir arama gerçekleştirilmeye çalışılır. Bu aşamanın başarısı arama operatörlerinin yeteneklerine bağlı olduğu kadar referans konumların doğru seçilmesine de bağlıdır. Dolayısıyla seçim sürecinin başarısı, MSA algoritmalarının nihai başarıları üzerinde oldukça etkilidir. MSA algoritmalarında arama süreci yaşam döngüsünün son adımı çözüm adayları topluluğunun güncellenmesidir. MSA'larında popülasyon büyüklüğü sabittir. Bundan dolayı arama sürecinde popülasyona eklenen her yeni bireyin yerine popülasyondan bir birey çıkarılmak durumundadır. Bireylerden hangisinin popülasyonda kalıp hangisinin ayrılacağına karar vermek için ise istisnalar olmakla birlikte aç gözlü yöntem kullanılarak karar verilir. Popülasyonda kalacak olan bireye karar verilirken evrim yasası işletilerek "güçlü olan hayatta kalır" ilkesine paralel olarak uygunluk değeri büyük olan birey popülasyonda kalır. MSA algoritmalarında son adım arama süreci yaşam döngüsünün sonlandırılmasıdır. MSA'larının arama süreci yaşam döngülerinin her bir adımında amaç fonksiyonlarını çağırma sayıları değişebilmektedir. Örneğin ABC algoritmasında bu sayı kolonideki takipçi/izci/kâşif arılar kadarken (Karaboga, Basturk, 2007), SOS algoritmasında dört (Cheng vd., 2014), GA'da ebeveyn sayısı kadar (genellikle iki) (Holland, 1975), HHO'da bir (Mirjalili vd., 2019) ve MRFO'da (Zhao vd., 2020) popülasyondaki birey sayısı kadardır. Dolayısıyla MSA'larında arama sürecini sonlandırmak amacıyla iterasyon sayısını kullanmak algoritmaların eşit şartlarda arama yapmaları konusunda adaletsizliğe neden olur. MSA'larından fırsat eşitliğini sağlamanın yolu her algoritma için amaç fonksiyonunu azami çağırma sayısını eşit tutmaktır. MSA'ları konusunda 90'lı yıllardan bu yana düzenli bir şekilde yürütülen CEC konferanslarında arama süreci sonlandırma kriteri olarak genellikle amaç fonksiyonunu çağırma sayısı 10.000*d (problem boyutunun on bin katı) olarak tanımlanmaktadır (Suganthan vd., 2005; Suganthan vd., 2013; Suganthan vd., 2016).

2.1. SDO Algoritması (SDO Algorithm)

SDO, iktisattaki arz-talep mekanizmasından esinlenilerek oluşturulmuş popülasyon tabanlı optimizasyon algoritmasıdır. SDO algoritması hem tüketicilerin talep hem de üreticilerin arz karakteristiklerini ve ilişkilerini taklit etmektedir. Arz-talep mekanizmasında emtia fiyatı ve miktarı dengeli olduğunda istikrar durumunu işaret ederken, fiyat ve miktar arasındaki dengenin bozulması istikrarsızlık/kararsızlık durumunu işaret etmektedir.

Emtia fiyat ve miktarının dengesine göre işaret edilen istikrar durumu SDO algoritması için komşuluk araması veya çeşitlilik görevlerini yerine getirmektedir. Bu noktada kararsızlık durumu SDO algoritmasında çeşitlilik görevini sağlarken, istikrar durumu hassas arama görevini yerine getirmektedir (Zhao vd., 2019).

SDO algoritmasında her biri d çeşit farklı emtiaya sahip n pazar ve her bir emtia türü belirli bir miktar ve fiyata sahiptir. Bir piyasanın *d* emtia fiyatları, optimizasyon probleminin değişkenleri olarak bir aday çözümünü temsil eder ve bir piyasanın d emtia miktarları olası bir aday çözüm olarak gözden geçirilir. Bu olası aday çözümü; mevcut aday çözümünden daha iyi ise, aday çözümün yerine olası çözüm getirilir. SDO algoritması popülasyon tabanlı bir algoritma olduğundan emtia fiyatı ve emtia miktarı sırasıyla iki farklı matriste tutulur.

SDO algoritması iki farklı denklem kullanmaktadır. Eşitlik (1), yani emtia miktarını güncelleyen arz denklemi; üreticilerin arz ilişkisini temsil etmektedir. Eşitlik (2) ise emtia fiyatını güncelleyen talep denklemi olarak ifade edilmiştir ve tüketicilerin talep ilişkisini temsil etmektedir.

$$Y_i(t) = Y_\Delta + \propto * (X_i(t-1) - X_\Delta) \quad (1)$$

$$X_i(t) = X_\Delta + \beta * (Y_i(t) - Y_\Delta) \quad (2)$$

Verilen formüllerde $X_i(t)$, t. Zamandaki i. emtia fiyat vektörü, $Y_i(t)$, t. Zamandaki i. emtia miktar vektörüdür. α ve β sırasıyla arz ve talep ağırlığıdır. Denklem 1'de her bir piyasadaki emtia miktar vektörü, hem denge miktarı vektörüne (Y_Δ), hem de denge fiyat vektörüne (X_Δ) göre güncellenebilir. Arz ve talep denklemleri düzenlenirse Eşitlik (3)'deki denklem elde edilir.

$$X_i(t) = X_\Delta - \beta * \propto * (X_i(t-1) - X_\Delta) \quad (3)$$

α (arz ağırlığı) ve β (talep ağırlığı) ağırlıklarının değerleri ayarlanarak farklı emtia fiyat vektörleri elde edilebilir. SDO'da keşif ve sömürü gerçekleştirmek için hem arz ağırlığı hem de talep ağırlığı uygun şekilde sunulmalıdır. Bu iki ağırlık Eşitlik (4) ve Eşitlik (5)'te verilmiştir.

$$\alpha = \frac{2 * (T-t+1)}{T} * \sin(2 * \pi * r) \quad (4)$$

$$\beta = 2 * \cos(2 * \pi * r) \quad (5)$$

Burada T, maksimum yineleme sayısıdır ve r, [0, 1] 'de rastgele bir sayı veya vektördür. L değişkeni (Eşitlik 6) α ve β ağırlıklarının çarpımına eşit olsun,

$$L = \alpha * \beta = \frac{4 * (T-t+1)}{T} * \sin(2 * \pi * r) * \cos(2 * \pi * r) \quad (6)$$

| L | <1 stabilite moduna karşılık gelir ve istikrar modunu taklit eder. α ve β ağırlıkları ayarlanarak mevcut fiyat vektörüne göre X_Δ denge fiyatı etrafındaki farklı emtia fiyat vektörleri elde edilebilir. Bu mekanizma sömürüyü vurgular ve SDO algoritmasını yerel olarak arama yapmaya teşvik eder.

| L | > 1, herhangi bir piyasadaki emtia fiyat vektörünün denge fiyat vektöründen çok uzaklaşmasına izin veren kararsızlık moduna karşılık gelir. Bu durum her pazarı arama alanında yeni umut verici bölgeler aramaya zorlar. Bu mekanizma keşif üzerine yoğunlaşır ve SDO algoritmasını global olarak arama yapmaya zorlar. Dolayısıyla *L* değeri, SDO'nun keşif ve sömürü arasında sorunsuz geçişine yardımcı olurken rasgele dalgalanma ile doğrusal olarak azalmaktadır.

2.2. FDB Yöntemi (FDB Method)

FDB (Kahraman vd., 2020), MSA algoritmalarının tasarımı için geliştirilmiş ve arama sürecinde popülasyonundaki en iyi çözüm adayına en fazla katkıyı sağlayabilecek olanın etkili bir şekilde belirlenmesini sağlayan bir seçim yöntemidir. MHS algoritmalarında topluluk içindeki bireyler için FDB seçim yöntemi kullanılarak çözüm adaylarının arama sürecine katkılarına işaret eden skor değerleri hesaplanmaktadır. Skor hesabında bireylerin iki özelliği dikkate alınmaktadır. Bunlar, uygunluk değerleri ve popülasyonda anlık olarak

en iyi çözüm adayına olan uzaklık değerleridir. Buna göre, m-adet amaç fonksiyonundan, J+K adet kısıttan oluşan bir kısıtlı optimizasyon problemi Eşitlik (7)'de verildiği gibi temsil edilir.

$$\text{Amaç} \quad \underset{x \in R^n}{minimize\ /\ maximize}\ G = f(x_1, x_2, \ldots, x_m)$$
$$\emptyset_j(x) = 0, (j = 1, 2, \ldots, J),$$
$$\varphi_k(x) \leq 0, (k = 1, 2, \ldots, K) \qquad (7)$$

Eşitlik 7'de $\emptyset_j$ve φ_k sırasıyla eşitlik ve eşitsizlik kısıtlarını temsil ederler. Eşitlik 7'de verilen bir optimizasyon probleminin çözümlenmesi için geliştirilmiş bir MSA algoritmasının n-adet çözüm adayından oluştuğu kabul edilsin. Buna göre çözüm adayları vektörü (popülasyon, *P*) ve bu adayların uygunluk değerleri vektörü (*F*) Eşitlik (8)'de verildiği gibi temsil edilebilir.

$$P \equiv \begin{bmatrix} p_1 \\ \vdots \\ p_n \end{bmatrix} \equiv \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}_{nxm}, \; F \equiv \begin{bmatrix} f_1 \\ . \\ . \\ . \\ f_n \end{bmatrix}_{nx1} \qquad (8)$$

Yukarıda verilen tanımlamalara göre çözüm adaylarının FDB skorlarının hesaplanma adımları aşağıda verilmektedir.

i) Popülasyondaki *i*-inci çözüm adayı P_i'nin, popülasyonun *t*-inci anındaki en iyi çözüm adayı olan P_{best}'e olan oklit uzaklığı Eşitlik 9'da verildiği gibi hesaplanır:

$$_{i=1}^{n}\forall\ P_i, D_{P_i} = \sqrt{\left(p_{i[1]} - p_{\text{best}[1]}\right)^2 + \left(p_{i[2]} - p_{\text{best}[2]}\right)^2 + \cdots + \left(p_{i[m]} - p_{\text{best}[m]}\right)^2} \qquad (9)$$

ii) Popülasyondaki bireylerin P_{best}'ten uzaklığını temsil eden D_p vektörü Eşitlik (10) 'da verilmiştir.

$$D_P \equiv \begin{bmatrix} d_1 \\ . \\ . \\ . \\ d_n \end{bmatrix}_{nx1} \qquad (10)$$

iii) Çözüm adaylarının FDB skorları hesaplanırken Eşitlik 8'de verilen uygunluk değerleri vektörü (*F*) ve Eşitlik 10'da verilen uzaklık değerleri vektörü D_p kullanılır. Bu iki parametrenin skor hesaplamasında birbirlerine hakim olmaması için normalleştirilmesi gerekir. Buna göre, her iki parametrenin [0, 1] normalize edilmiş uygunluk ve uzaklık değerleri, sırasıyla normF ve normDp ile temsil edilir. Çözüm adaylarının FDB skorları (S_p) Eşitlik 11'de verildiği gibi hesaplanır.

$$_{i=1}^{n}\forall\ P_i,\ S_{P_{[i]}} = w * \ normF_{[i]} + (1 - w) * normD_{P_{[i]}} \qquad (11)$$

Eşitlik 11'de *w*, uygunluk ve uzaklık parametrelerinin FDB skoru üzerindeki etkilerini temsil eden ağırlık katsayısıdır. FDB yönteminin tanıtıldığı çalışmada *w*=0.5 olarak alınmıştır.

iv) Buna göre, *P*-popülasyonundaki bireylerin FDB skorlarını temsil eden *n*-boyutlu S_p vektörü Eşitlik 12'de verilmektedir.

$$S_P \equiv \begin{bmatrix} S_1 \\ . \\ . \\ . \\ S_n \end{bmatrix}_{nx1} \quad (12)$$

Çözüm adaylarının FDB skorlarını temsil eden S_p oluşturulduktan sonra, MHS algoritmalarında arama sürecine rehberlik edecek çözüm adaylarının seçiminde açgözlü yöntem kullanılarak FDB skoru en yüksek olan birey seçilir. Buna karşın olasılıksal seçim yöntemleri kullanılarak da popülasyon içerisinden bireylerin seçilmesi mümkündür. FDB yöntemi hakkında daha ayrıntılı bilgi referans çalışmasından (Kahraman vd., 2020) elde edilebilir.

2.3. SDO Algoritmasının FDB Yöntemi ile Tasarlanması (Designing the SDO Algorithm with FDB Method)

Bu bölümde SDO algoritmasının FDB yöntemi ile tasarlanması süreci tanıtılmaktadır. Bu amaçla SDO'u finansal süreçteki doğal işleyişe daha uygun hale getirebilecek emtia fiyatlandırma modelleri üzerine çalışmalar yürütülmüştür. Çünkü SDO algoritmasının temel işleyişi emtia pazarındaki denge fiyatı ve denge miktarı hesabına bağlıdır. Bu açıdan Eşitlikler 1 ve 2, FDB yöntemi ile seçilen fiyat ve denge miktarı vektörleri kullanılarak yeniden tasarlanmıştır. Amaç pazardaki fiyat ve denge miktarlarını olabildiğince gerçek değerlerine yaklaştırabilmektir.

Algoritma 2'de FDB-SDO algoritmasının sözde kodu verilmektedir. Sözde kod üzerinde önerilen algoritmanın işleyişi ve FDB seçim yönteminin SDO algoritmasında fiyat ve denge miktarı belirleme süreçlerine tatbik edilişi adım-adım verilmektedir.

	Algoritma 2. FDB-SDO algoritmasının sözde kodu
1.	**Başla**
2.	**Pazar popülasyonun oluşturulması**
3.	X: Emtia fiyat vektörü rastgele oluşturulur.
4.	Y: Emtia miktar vektörü rastgele oluşturulur.
5.	**for** each pazar (i =1, ….., n)
6.	Emtia fiyat vektörünün uygunluğunun hesaplanması, $F_{xi} = f(X_i)$
7.	Emtia miktar vektörünün uygunluğunun hesaplanması, $F_{yi} = f(Y_i)$
8.	**if** ($F_{yi} < F_{xi}$), $X_i = Y_i$, **end if**
9.	**end for**
10.	**Arama sürecinin başlaması**
11.	**While** iter = 1 **to** amaç fonksiyon değerlendirme sayısı
12.	Q ve P, sırasıyla denge miktarı ve denge fiyatı için güncel değerleri hesapla: 23 numaralı referansta 8 ve 11 numaralı denklemler kullanılır.
13.	**for** each pazar (i =1, ….., n)
14.	**Yerel arama ve çeşitlilik süreci**
15.	α: Eşitlik 4
16.	β: Eşitlik 5
17.	Y_Δ (Denge miktarı) = Y_k, k = RuletSeçimYöntemi(Q)
18.	**if** (rand > 0.5), X_Δ (Denge fiyatı) = Orijinal algoritmadaki 12 numaralı denklem.
19.	**else** X_Δ (Denge fiyatı) = X_k, k = RuletSeçimYöntemi(P), **end if**
20.	rfdb = RuletFitnessDistanceBalance(Y, F_y)
21.	$Y_{new} = X_{rfdb} + \alpha * (X_i - X_\Delta)$ (Formülde Y_Δ yerine X_{rfdb} kullanılmıştır.)
22.	$X_{new} = X_\Delta - \beta * (Y_{new} - Y_{rfdb})$ (Formülde Y_Δ yerine Y_{rfdb} kullanılmıştır.)
23.	**Pazar popülasyonunun güncellenme süreci**
24.	Yeni emtia fiyat vektörünün uygunluğunun hesaplanması, $F_{xnew} = f(X_{new})$
25.	Yeni emtia miktar vektörünün uygunluğunun hesaplanması, $F_{ynew} = f(Y_{new})$
26.	**if** ($F_{Xnew} < F_{Xi}$), $X_i = X_{new}$, **end if**
27.	**if** ($F_{ynew} < F_{yi}$), $Y_i = Y_{new}$, **end if**

28.	**end for**
29.	**Emtia fiyat vektörünün güncellenme süreci**
30.	**for** each pazar (i =1, ….., n)
31.	**if** ($F_{yi} < F_{xi}$), $X_{i=} Y_i$, **end if**
32.	**end for**
33.	**end while**
34.	**end**

SDO algoritmasının gerçek yaşamdaki işleyişle daha uyumlu bir hale getirilmesi için varyasyonları oluşturulmuştur. Oluşturulan varyasyonlarda algoritmanın baz modelindeki arz ve talep süreçleri yeniden tasarlanmıştır. Tasarımsal değişiklikler arz veya talep süreçlerinin modellenmesinde kullanılan denge vektörü üzerine olmuştur. Buna göre, bu makale çalışmasında tasarlanan dört farklı SDO varyasyonunun ve SDO'nun baz modelinin arz ve talep modellerine ilişkin denklemler Tablo 1'de özetlenmiştir.

Tablo 1. SDO ve FDB-SDO varyasyonlarında arz ve talep denklemlerinin tanımlanması
(Defining supply and demand equations in SDO and FDB-SDO variations)

SDO ve FDB'li Varyasyonları	Arz ve Talep Denklemleri
SDO	Arz : $Y_i = Y_\Delta + \propto * (X_i - X_\Delta)$
	Talep : $X_i = X_\Delta + \beta * (Y_i - Y_\Delta)$
Case-1: Rulet FDB	Arz : $Y_i = X_{rfdb} + \propto * (X_i - X_\Delta)$
	Talep : $X_i = X_\Delta + \beta * (Y_i - Y_{rfdb})$
Case-2: Rulet FDB	Arz : $Y_i = Y_{rfdb} + \propto * (X_i - X_\Delta)$
	Talep : $X_i = X_\Delta + \beta * (Y_i - Y_\Delta)$
Case-3: Rulet FDB	Arz : $Y_i = Y_\Delta + \propto * (X_i - Y_{rfdb})$
	Talep : $X_i = X_\Delta + \beta * (Y_i - Y_\Delta)$
Case-4: FDB	Arz : $Y_i = Y_\Delta + \propto * (X_i - Y_{fdb})$, %90 oranında uygulandı Arz : $Y_i = Y_\Delta + \propto * (X_i - X_\Delta)$, %10 oranında uygulandı
	Talep : $X_i = X_\Delta + \beta * (Y_i - Y_\Delta)$

Tablo 1'de verilen her bir varyasyon SDO algoritmasında arz-talep modellerinin yeni tasarımlarına karşılık gelmektedir. Tasarımsal bu değişikliklerin etkili olup olmadıkları ise kapsamlı bir deneysel çalışmayla araştırılmıştır.

3. Deneysel Çalışma Ayarları (Experimental Run Settings)

3.1. Ayarlar (Settings)

Önerilen FDBSDO algoritmasının arama performansını test etmek ve doğrulamak için kapsamlı bir deneysel çalışma gerçekleştirilmiştir. Amaç, geliştirilen varyasyonların etkisini ve önerilen yöntemin performansını açıkça ve kesin bir şekilde göstermektir. Bu amaçla, deneysel çalışmalarda kısıtsız karşılaştırma problemlerinin dört farklı tipi de kullanılmıştır. SDO algoritmasının baz modelinin ve FDB-SDO varyasyonlarının farklı boyutlardaki arama uzaylarındaki performansları test edilmiştir. Deneysel çalışmaları objektif ve adil bir şekilde yürütebilmek için aşağıdaki prosedürler izlenmiştir:

- Deneysel çalışma ayarları için CEC 2017 konferansında tanımlanan şartlar (Suganthan vd., 2016) referans alınmıştır.
- SDO algoritmasının parametrelerinin ayarlanmasında, kendi çalışmasında verilen ayarlar, yani popülasyon boyutu ve diğer ayarları referans olarak alınmıştır.
- Algoritmalar arasında fırsat eşitliğini sağlamak için, amaç fonksiyonunu azami değerlendirme sayısı üzerinden sonlandırma kriteri tanımlanmıştır. Bu değer 10.000*d (d:problem boyutu) dir.
- Önerilen yöntemin düşük, orta ve yüksek boyutlu arama alanlarındaki performansını ortaya çıkarmak için dinamik olarak boyutlandırılabilen CEC 2017 karşılaştırma fonksiyonları kullanılmıştır.

- Deneysel çalışmalar MATLAB®R2018a'da, Intel (R) Core ™ i7-4770KCPU @ 3.50GHz ve 16 GB RAM ve x64 tabanlı işlemci üzerinde gerçekleştirildi.

3.2. Karşılaştırma Problemleri (Comparison Problems)

Deneysel çalışmalarda 30'u kısıtsız 20'si kısıtlı olmak üzere 50 farklı optimizasyon problemi kullanılmıştır. Problemlerin tamamına yakını CEC2017 (Suganthan vd., 2016) ve CEC 2020 (Suganthan vd., 2020) karşılaştırma havuzlarından alınmıştır.

3.3. Kısıtsız Test Problemleri (Unconstrained Testing Problems)

Deneysel çalışmada CEC 2017 havuzunda yer alan 4 farklı tipte ve 30 adet kısıtsız test problemi kullanılmıştır. Problemler 30/50/100 boyutlu olarak tasarlanmıştır. Tasarım değişkenleri için arama uzayı sınırları [-100, 100] dür. Problemlerin özellikleri Tablo 2'de verilmektedir (Kahraman vd., 2020).

Tablo2. CEC 2017 karşılaştırma problemleri ve özellikleri
(CEC 2017 comparision problems and features)

Problem tipi	Definition
Tek modlu	Algoritmaların yerel arama yeteneklerini test etmek için kullanılır.
Çok modlu	Algoritmaların küresel arama (çeşitlilik) yeteneklerini test etmek için kullanılır.
Melez	Algoritmaların dengeli arama yeteneklerini araştırmak için kullanılır.
Komposizyon	Algoritmaların karmaşıklık düzeyi yüksek olan arama uzaylarındaki performanslarını test etmek için kullanılır.

3.4. Kısıtlı Mühendislik Tasarım Problemleri (Constrained Engineering Design Problems)

Önerilen yöntemin kısıtlı mühendislik tasarım problemlerindeki performansını test etmek ve doğrulamak amacıyla 20 adet gerçek mühendislik problemi kullanılmıştır. Bu problemlerden 9'u literatürdeki saygın dergilerde yayınlanan çalışmalardan (Long vd., 2019; Dong vd., 2014; Amir, 2013; Lin vd., 2017; Mirjalili vd., 2017; Chen vd., 2019; Zhao vd., 2020; Ragsdell vd., 2006) ve 11'i de CEC 2020 mühendislik problemleri havuzundan (Suganthan vd., 2020) elde edilmiştir. Problemlere ilişkin bilgi Tablo 3'te verilmektedir.

Tablo 3. Kısıtlı mühendislik tasarım problemleri ve özellikleri
(Constreined engineering design problems and features)

Problem	Problem ismi	Kısıt sayısı	Parametre sayısı
P_1 (Long vd., 2019)	Tension / Compression Spring Design	4	3
P_2 (Dong vd., 2014)	Pressure Vessel Design	4	4
P_3 (Amir, 2013)	Speed Reducer Design	11	7
P_4 (Lin vd., 2017)	Welded Beam Design	7	4
P_5 (Suganthan vd., 2020)	Four Stage Gear Box Problem	86	22
P_6 (Suganthan vd., 2020)	Optimal Design Of Industrial Refrigeration System	15	14
P_7 (Suganthan vd., 2020)	Blending-Pooling-Separation Problem	32	38
P_8 (Suganthan vd., 2020)	Optimal Operation of Alkylation Unit	14	7
P_9 (Mirjalili vd., 2017)	Cantilever Beam Design	1	5
P_{10} (Suganthan vd., 2020)	Multiple Disk Clutch Brake Design Problem	8	5
P_{11} (Suganthan vd., 2020)	Planetary Gear Train Design Optimization Problem	11	9
P_{12} (Suganthan vd., 2020)	Ten Bar Truss Design	3	10
P_{13} (Chen vd., 2019)	L-Beam Design	1	4
P_{14} (Suganthan vd., 2020)	Hydro-Static Thrust Bearing Design	7	4
P_{15} (Zhao vd., 2020)	Belleville Spring Design	7	4
P_{16} (Ragsdell vd., 2006)	Corrugated Bulkheads	6	4

P_{17} (Suganthan vd., 2020)	Heat Exchanger Network Design (*Case 1*)	8	9
P_{18} (Suganthan vd., 2020)	Heat Exchanger Network Design (*Case 2*)	9	11
P_{19} (Suganthan vd., 2020)	Propane, Isobutane, n-Butane Nonsharp Separation	38	48
P_{20} (Suganthan vd., 2020)	Step-Cone Pulley Problem	11	5

4. Analiz Sonuçları (Analysis Results)

Bu bölümde deneysel çalışmalardan elde edilen veriler kullanılarak SDO'nun ve FDB-SDO varyasyonlarının karşılaştırmalı olarak performansları analiz edilmektedir. Algoritmaların birbirlerine göre performanslarını karşılaştırmak ve onları sıralamak için Friedman Testi, SDO algoritmasını baz modeli ile FDB-SDO varyasyonlarının her birini ayrı-ayrı ikili olarak karşılaştırmak için Wilcoxon testi uygulanmıştır. Buna göre analiz sonuçları takip eden alt bölümlerde sırasıyla sunulmaktadır.

4.1. İstatistiksel Analiz Sonuçları (Statistical Analysis Results)

Tablo 4'te bu makalede geliştirilen dört farklı FDB-SDO varyasyonunun ve SDO algoritmasının baz modelinin Friedman test yöntemine göre sıralamaları verilmektedir. Sıralamalar CEC 2017 karşılaştırma havuzundaki problemlerin 30/50/100 boyutları için gerçekleştirilen 51 bağımsız çalışmadan elde edilen veriler kullanılarak elde edilmiştir.

Tablo 4. Friedman analiz yöntemine göre algoritmaların sıraları
(Order of algorithms according to Friedman analysis method)

Algoritmalar	Boyut = 30	Boyut = 50	Boyut = 100	Ortalama Sıra
Case-2	2.821	2.732	2.861	2.804
Case-1	2.888	2.804	2.938	2.876
Case-4	2.776	3.057	2.922	2.918
Case-3	3.032	3.050	3.002	3.028
SDO	3.482	3.356	3.274	3.370

Tablo 4'te verilen Friedman sıralamalarına göre, deneylerin tümünde FDB-SDO varyasyonlarının dördü de SDO algoritmasını geride bırakmışlardır. Bu sonuç, FDB tabanlı tasarımsal değişikliğin SDO'nun doğal sürecine daha uygun bir yapıya kavuşmasını sağladığının güçlü bir işaretidir. SDO'da arz ve talep süreçlerinin modellenmesinde FDB'nin olumlu bir etkisi olduğu anlaşılmaktadır.

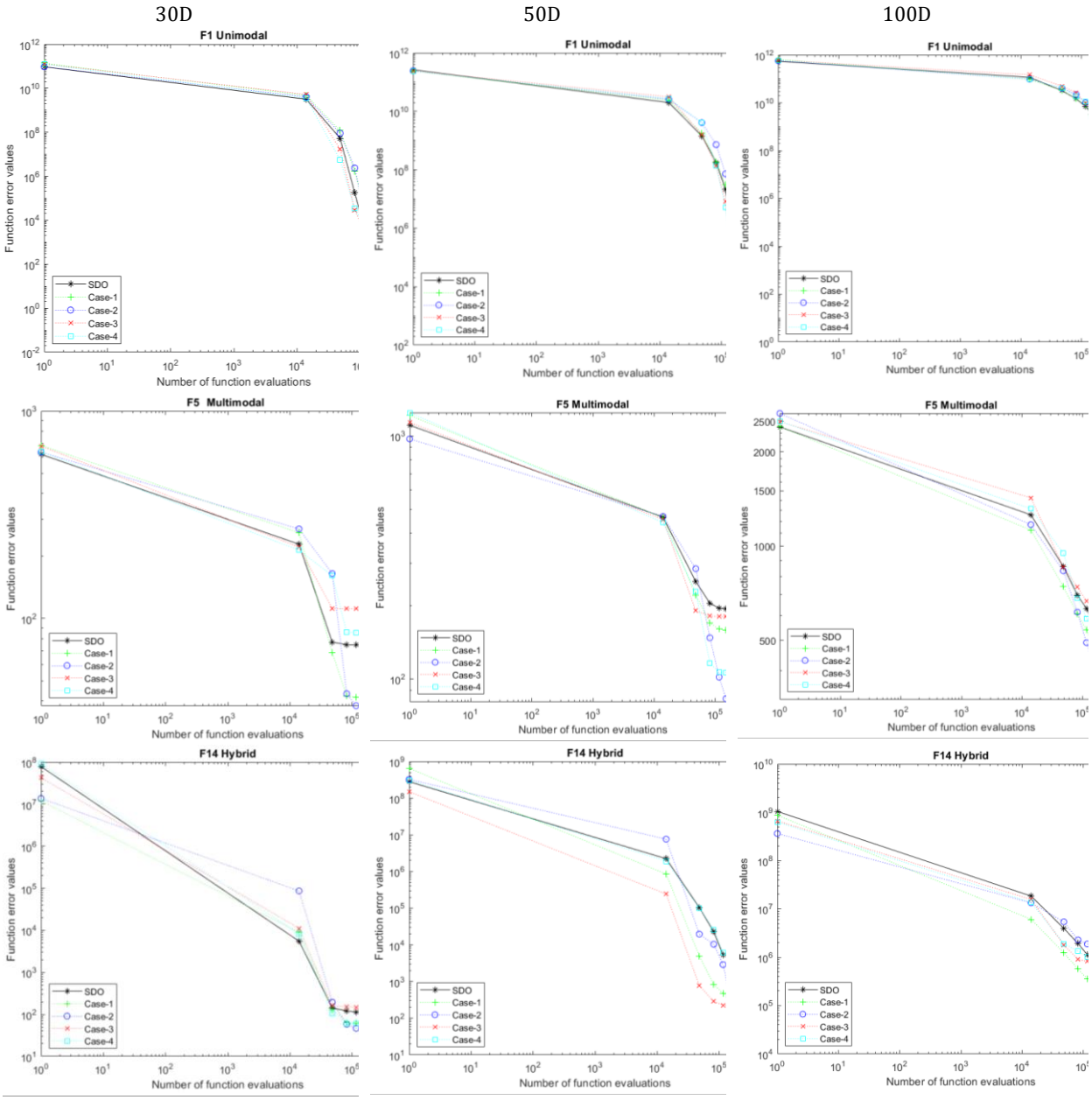
Tablo 5. Wilcoxon analiz yöntemine göre SDO baz modeli ile FDB-tabanlı SDO'lar arasında ikili karşılaştırma
(Paired comparison between SDO base model and FDB-SDO according to Wilcoxon analysis method)

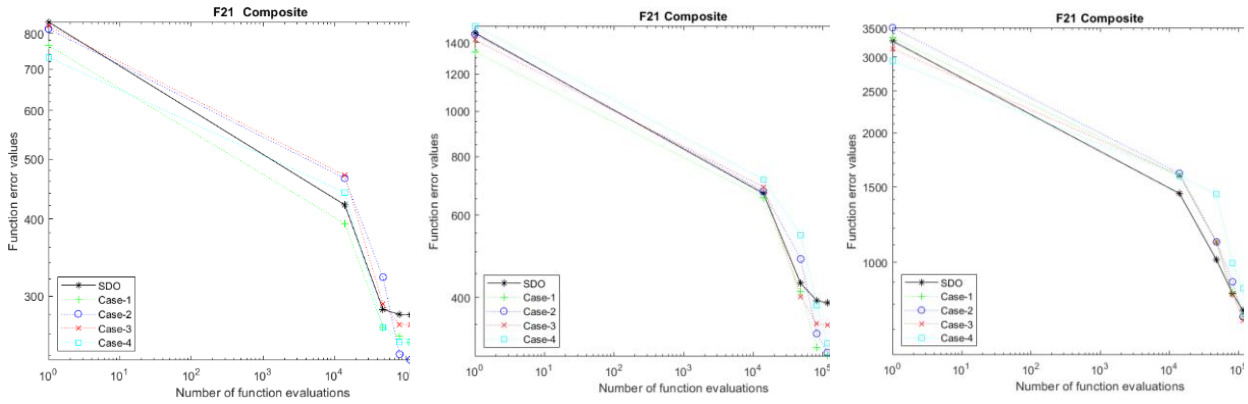
vs. SDO +/=/-	Boyut = 30	Boyut = 50	Boyut = 100
Case-2	16/11/3	13/17/0	12/18/0
Case-1	10/19/1	9/21/0	4/25/1
Case-4	10/20/0	6/23/1	7/20/3
Case-3	8/21/1	8/19/3	9/18/3

SDO algoritmasının baz modeli ile FDB-SDO varyasyonları arasındaki ikili karşılaştırmalara ait analiz sonuçları Tablo 5'te verilmektedir. Buna göre FDB-SDO varyasyonları ikili karşılaştırmalarda SDO algoritmasına büyük bir üstünlük kurmuşlardır. Özellikle Case-2 ile verilen FDB-SDO varyasyonu 50 ve 100 boyutlu arama uzaylarında 30 problemin hiçbirinde rakibine yenilmemiştir. Case-2 yöntemi 50 boyuttaki 30 problemin 13'ünde SDO algoritmasına üstünlük kurarken 17 problemde de iki algoritma arasında eşitlik olmuştur. Bu durum 10.000*d için olağandır. Çünkü algoritmalar bu kadar fazla sayıda deneme yaptıklarında nispeten kolay olan arama uzaylarında küresel en iyi çözüme yakınsayabilmektedirler. Ancak Case 2 varyasyonunun üstünlüğü karmaşıklık düzeyi yüksek problemlerde ortaya çıkmıştır.

4.2. Yakınsama Analiz Sonuçları (Convergence Analysis Results)

Bu bölümde SDO ve FDB-SDO varyasyonlarının yakınsama eğrileri sunulmaktadır. Algoritmaların yakınsama yeteneklerini incelemek için CEC2017 problem setinde dört farklı tipteki problemlerden birer adet seçilmiştir. Seçilen problemlerden 30/50/100 boyutlarında arama uzayları tasarlanmış ve algoritmaların yakınsama davranışları gözlemlenmiştir. Seçilen problemler F1 (Tek Modlu), F5 (Multimodal), F14(Hibrit) ve F21(Kompozisyon)'dir.





Şekil 1. Algoritmaların dört farklı tipteki ve üç farklı boyuttaki arama uzaylarındaki yakınsama eğrileri (Convergence curves of algorithms in four different types and three different sizes of search spaces)

Şekil 1'de dört satır ve üç sütun olmak üzere, algoritmaların on iki farklı durum için yakınsama eğrileri verilmektedir. Buna göre satırlar sırasıyla tekmodlu/çokmodlu/melez/komposizyon tiplerdeki problemleri ve sütunlar da bu problemlerin 30/50/100 boyutlu tasarımlarının olduğu durumları göstermektedir. Bu şartlar altında verilen yakınsama eğrileri incelendiğinde, FDB-SDO varyasyonlarının SDO algoritmasının baz modeline kıyasla çok daha hızlı yakınsadığı söylenemez. Hatta yakınsama hızları arasında önemli bir farklılık olmadığı söylenebilir. Ancak grafiklerde siyah renk ile çizdirilen yakınsama eğrisi incelendiğinde SDO algoritmasının tüm problem türlerinde yerel çözüm tuzaklarına yakalandığı ve prematüre bir yakınsama davranışı gösterdiği anlaşılmaktadır. Yani SDO algoritması hızlı yakınsamaya başlasa da kısa sürede yerel çözüm tuzaklarına yakalanmakta ve küresel çözüme yakınsayamamaktadır. Bu durum, Şekil 1'deki 4 farklı problem türü ve 3 farklı problem boyutu için tekrarlanmıştır. FDB-SDO varyasyonlarının arasında Case-2'nin küresel çözüme yakınsama performansı dikkat çekicidir. Case-2, dört problem tipinde ve üç farklı boyutta küresel çözümün keşfedilmesinde rakiplerine kıyasla daha dengeli bir arama performansı sergilemiştir.

4.3. Mühendislik Tasarım Problemleri için Sonuçlar (Consequences for Engineering Design Problems)

Bir önceki bölümde FDB-SDO varyasyonları arasında en gürbüz yöntemin Case-2 olduğu belirlenmişti. Bu bölümde FDB-SDO (Case-2 varyasyonu) ile SDO algoritması arasında mühendislik tasarım problemleri üzerinden karşılaştırma yapılmaktadır.

Tablo 6. FDB-SDO ile SDO algoritmalarının mühendislik tasarım problemlerindeki performansları (Performance of FDB-SDO and SDO algorithms in engineering design problems)

No.	Algoritma	Tasarım parametreleri ($x_1, x_2, ..., x_n$)				En iyi
P1	SDO (-)	X_1 = 0.050000	X_2 = 0.403646	X_3 = 6.821993		0.008902
	FDB-SDO (+)	X_1 = 0.050000	X_2 = 0.403646	X_3 = 6.821995		0.008902
P2	SDO (-)	X_1 = 0.778169	X_2 = 0.384649	X_3 = 40.319657	X_4 = 199.999464	5885.336754
	FDB-SDO (+)	X_1 = 0.778168	X_2 = 0.384649	X_3 = 40.319619	X_4 = 199.999988	5885.332810
P3	SDO (-)	X_1 = 3.500000 X_5 = 7.800000	X_2 = 0.700000 X_6 = 3.350214	X_3 = 17.000000 X_7 = 5.286683	X_4 = 7.300000	2996.348223
	FDB-SDO (+)	X_1 = 3.500000 X_5 = 7.800000	X_2 = 0.700000 X_6 = 3.350214	X_3 = 17.000000 X_7 = 5.286683	X_4 = 7.300000	2996.348165
P4	SDO (-)	X_1 = 0.244368	X_2 = 6.218607	X_3 = 8.291473	X_4 = 0.244368	2.381134
	FDB-SDO (+)	X_1 = 0.244368	X_2 = 6.218606	X_3 = 8.291471	X_4 = 0.244368	2.381134
P5	SDO (-)	X_1 = 16.780102 X_5 = 16.932178 X_9 = 1.114831 X_{13} = 3.372937 X_{17} = 6.457888 X_{21} = 5.031203	X_2 = 36.306321 X_6 = 35.567793 X_{10} = 1.323825 X_{14} = 4.313777 X_{18} = 1.938174 X_{22} = 3.277657	X_3 = 15.586479 X_7 = 16.993992 X_{11} = 1.275650 X_{15} = 3.814983 X_{19} = 4.787740	X_4 = 33.743963 X_8 = 34.852153 X_{12} = 0.623018	36.249827

					X_{16} = 4.416287 X_{20} = 5.261783	
	FDB-SDO (+)	X_1 = 13.192124 X_5 = 18.151393 X_9 = 0.818227 X_{13} = 7.270444 X_{17} = 5.850686 X_{21} = 2.580448	X_2 = 28.292477 X_6 = 38.006440 X_{10} = 1.025305 X_{14} = 3.927826 X_{18} = 1.713945 X_{22} = 4.923850	X_3 = 19.822663 X_7 = 19.010083 X_{11} = 0.870235 X_{15} = 6.881352 X_{19} = 3.097177	X_4 = 39.799388 X_8 = 41.108390 X_{12} = 1.133115 X_{16} = 4.116422 X_{20} = 5.085194	35.360089
P6	SDO (=)	X_1 = 0.001000 X_5 = 0.001002 X_9 = 4.999982 X_{13} = 0.009879	X_2 = 0.001000 X_6 = 0.001000 X_{10} = 2.036008 X_{14} = 0.118598	X_3 = 0.001000 X_7 = 1.524400 X_{11} = 0.001872	X_4 = 0.001000 X_8 = 1.524013 X_{12} = 0.001872	0.033002
	FDB-SDO (=)	X_1 = 0.001000 X_5 = 0.001000 X_9 = 4.999999 X_{13} = 0.010084	X_2 = 0.001000 X_6 = 0.001000 X_{10} = 2.000000 X_{14} = 0.121067	X_3 = 0.001000 X_7 = 1.524000 X_{11} = 0.001992	X_4 = 0.001000 X_8 = 1.524004 X_{12} = 0.001992	0.032908

No.	Algoritma	Tasarım parametreleri ($x_1, x_2, \ldots, x_n$)				En iyi
P7	SDO (=)	X_1 = 50.389711 X_5 = 44.092184 X_9 = 39.850677 X_{13} = 63.500150 X_{17} = 33.683259 X_{21} = 0.390658 X_{25} = 2.957035E-09 X_{29} = 0.308984 X_{33} = 0.366611 X_{37} = 0.379631	X_2 = 45.590499 X_6 = 39.724032 X_{10} = 17.947459 X_{14} = 43.297046 X_{18} = 29.810937 X_{22} = 0.262476 X_{26} = 0.422023 X_{30} = 0.287535 X_{34} = 1.097946E-10 X_{38} = 0.144051	X_3 = 54.858671 X_7 = 6.888159 X_{11} = 11.920655 X_{15} = 2.200065 X_{19} = 29.810196 X_{23} = 0.170595 X_{27} = 0.399884 X_{31} = 0.289444 X_{35} = 0.780917	X_4 = 149.161116 X_8 = 32.835871 X_{12} = 9.982560 X_{16} = 7.413720 X_{20} = 0.000740 X_{24} = 0.381426 X_{28} = 0.133886 X_{32} = 0.192093 X_{36} = 0.326475	7152278.625299
	FDB-SDO (=)	X_1 = 39.475454 X_5 = 45.638187 X_9 = 61.378571 X_{13} = 68.541208 X_{17} = 2.373693 X_{21} = 0.360753 X_{25} = 0.923567 X_{29} = 0.168827 X_{33} = 0.329771 X_{37} = 0.143331	X_2 = 67.067122 X_6 = 23.144502 X_{10} = 10.431182 X_{14} = 43.150375 X_{18} = 33.639044 X_{22} = 0.252196 X_{26} = 0.023721 X_{30} = 0.352514 X_{34} = 0.437300 X_{38} = 0.533077	X_3 = 60.911379 X_7 = 23.136730 X_{11} = 4.458523 X_{15} = 22.119846 X_{19} = 0.571869 X_{23} = 0.173140 X_{27} = 0.357790 X_{31} = 0.149442 X_{35} = 0.110979	X_4 = 132.546042 X_8 = 0.007771 X_{12} = 46.488864 X_{16} = 18.656834 X_{20} = 33.067174 X_{24} = 0.500272 X_{28} = 0.257794 X_{32} = 0.526568	3897337.084377

					X_{36} = 0.351861	
P8	SDO (-)	X_1 = 1364.496877 X_5 = 91.019936	X_2 = 99.935019 X_6 = 3.286289	X_3 = 2000.168323 X_7 = 141.470885	X_4 = 90.735811	-140.612602
	FDB-SDO (+)	X_1 = 1364.956438 X_5 = 91.015619	X_2 = 99.999857 X_6 = 3.279213	X_3 = 2000.001639 X_7 = 141.460437	X_4 = 90.740827	-142.587460
P9	SDO (-)	X_1 = 5.978575 X_5 = 2.139180	X_2 = 4.876200	X_3 = 4.465813	X_4 = 3.479359	1.306601
	FDB-SDO (+)	X_1 = 5.978248 X_5 = 2.139134	X_2 = 4.876169	X_3 = 4.466100	X_4 = 3.479477	1.306601
P10	SDO (-)	X_1 = 70.000000 X_5 = 2.000000	X_2 = 90.000000	X_3 = 1.000000	X_4 = 589.405570	0.235242
	FDB-SDO (+)	X_1 = 70.000000 X_5 = 2	X_2 = 90.000000	X_3 = 1	X_4 = 405.663570	0.235242
P11	SDO (=)	X_1 = 30.288659 X_5 = 17.934252 X_9 = 0.751569	X_2 = 18.139872 X_6 = 68.809826	X_3 = 15.763622 X_7 = 1.285352	X_4 = 19.104663 X_8 = 2.317589	0.523250
	FDB-SDO (=)	X_1 = 30.367270 X_5 = 18.819102 X_9 = 0.907223	X_2 = 17.565489 X_6 = 68.679376	X_3 = 16.253535 X_7 = 1.631192	X_4 = 18.543061 X_8 = 2.148823	0.523250
P12	SDO (-)	X_1 = 30.358092 X_5 = 0.100020 X_9 = 0.100046	X_2 = 0.100000 X_6 = 0.559127 X_{10} = 21.467767	X_3 = 23.225613 X_7 = 21.110724	X_4 = 15.335825 X_8 = 7.457769	5060.956520
	FDB-SDO (+)	X_1 = 30.532730 X_5 = 0.100000 X_9 = 0.100000	X_2 = 0.100000 X_6 = 0.552175 X_{10} = 21.536840	X_3 = 23.210559 X_7 = 21.035646	X_4 = 15.190353 X_8 = 7.456812	5060.856754
P13	SDO (-)	X_1 = 49.999999	X_2 = 79.999999	X_3 = 1.764705	X_4 = 4.999999	0.006625
	FDB-SDO (+)	X_1 = 49.999999	X_2 = 79.999999	X_3 = 1.764705	X_4 = 4.999999	0.006625
P14	SDO (=)	X_1 = 5.957399	X_2 = 5.388710	X_3 = 5.527577E-06	X_4 = 2.392145	1637.779103
	FDB-SDO (=)	X_1 = 5.960484	X_2 = 5.394210	X_3 = 5.358827E-06	X_4 = 2.259852	1617.869093
P15	SDO (-)	X_1 = 0.037269	X_2 = 0.050000	X_3 = 5.000000	X_4 = 5.166669	0.014036
	FDB-SDO (+)	X_1 = 0.037268	X_2 = 0.050000	X_3 = 5.000000	X_4 = 5.166667	0.014035
P16	SDO (=)	X_1 = 57.692307	X_2 = 34.147620	X_3 = 57.692307	X_4 = 1.050000	6.842958
	FDB-SDO (=)	X_1 = 57.692307	X_2 = 34.147620	X_3 = 57.692307	X_4 = 1.050000	6.842958
P17	SDO (-)	X_1 = 0.341117. X_5=1999950.595952 X_9 = 700.004940	X_2 = 51.569406 X_6 = 193.908631	X_3 = 49.404202 X_7 = 100.004940	X_4 = 0.724151 X_8 = 599.995059	10520.919358
	FDB-SDO (+)	X_1 = 0.031376 X_5=1999998.675888 X_9 = 700.000132	X_2 = 64.768677 X_6 = 154.395515	X_3 = 1.324053 X_7 = 100.000132	X_4 = 0.210988 X_8 = 599.999867	786.513758

No.	Algoritma	Tasarım parametreleri ($x_1, x_2, ..., x_n$)				En iyi
P18	SDO (-)	X_1 = 818999.999984. X_5 = 0.025503 X_9 = 218.100000	X_2=1130999.996091 X_6 = 0.049057 X_{10} = 286.900000	X_3 = 2049999.998731 X_7 = 181.900000 X_{11} = 395.000000	X_4 = 0.050733 X_8 = 295.000000	13363.215122
	FDB-SDO (+)	X_1 = 818999.999998 X_5 = 0.050739 X_9 = 218.100000	X_2=1130999.999999 X_6 = 0.050739 X_{10} = 286.900000	X_3 = 2049999.999999 X_7 = 181.899999 X_{11} = 395.000000	X_4 = 0.050739 X_8 = 294.999999	7049.037004
P19	SDO (-)	X_1 = 23.239409. X_5 = 32.024933 X_9 = 15.835581 X_{13} = 36.783033 X_{17} = 13.287958 X_{21} = 0.566028 X_{25} = 8.782083 X_{29} =10.029434. X_{33}= 0.274365 X_{37} = 11.661574 X_{41} = 0.000244 X_{45} = 0.185034	X_2 = 48.342592 X_6 = 15.508427 X_{10} = 1.464186 X_{14} = 31.605567 X_{18} = 30.036755 X_{22} = 0.457980 X_{26} = 0.854854 X_{30} = 0.468706 X_{34} = 0.313175 X_{38} = 0.316898 X_{42} = 0.078262 X_{46} = 0.193544	X_3 = 88.888937 X_7 = 0.069206 X_{11} = 14.320061 X_{15} = 4.994188 X_{19} = 0.000646 X_{23} = 0.556847 X_{27} = 16.932430 X_{31} = 0.875136 X_{35} = 7.653472 X_{39} = 0.460332 X_{43} = 0.493199 X_{47} = 0.081824	X_4 = 139.529062 X_8 = 15.439218 X_{12} = 0.051346 X_{16} = 13.323420 X_{20} =30.036108 X_{24} = 0.999558 X_{28} = 0.879212 X_{32}=16.087071 X_{36} = 0.238984 X_{40} = 0.437350 X_{44} = 0.000298 X_{48} = 0.063553	16167628.052351
	FDB-SDO (+)	X_1 = 27.654384 X_5 = 23.515798 X_9 = 14.760699 X_{13} = 44.635996 X_{17} = 2.964437 X_{21} = 0.551671 X_{25} = 9.568400. X_{29} =11.469084 X_{33}= 0.406892 X_{37} = 14.809980 X_{41} = 0.000067 X_{45} = 0.115279	X_2 = 47.298134 X_6 = 17.342580 X_{10} = 0.338425 X_{14} = 18.812538 X_{18} = 23.551238 X_{22} = 0.794426 X_{26} = 0.932380 X_{30} = 0.597965 X_{34} = 0.487718 X_{38} = 0.331794 X_{42} = 0.002127 X_{46} = 0.536756	X_3 = 85.483066 X_7 = 0.006457 X_{11} = 14.410784 X_{15} = 2.826833 X_{19} = 0.000039 X_{23} = 0.774501 X_{27} = 16.029073 X_{31} = 0.892795 X_{35} = 8.875738 X_{39} = 0.359106 X_{43} = 0.492181 X_{47} = 0.046022	X_4 = 139.564415 X_8 = 17.336122 X_{12} = 0.011491 X_{16} = 13.021268 X_{20} =23.551200 X_{24} = 0.999895 X_{28} = 0.996783 X_{32}=15.7738531 X_{36} = 0.377437 X_{40} = 0.353388 X_{44} = 0.023036 X_{48} = 0.089888	3357968.219098
P20	SDO (-)	X_1 = 38.413874 X_5 = 89.999896	X_2 = 52.858271	X_3 = 70.472312	X_4 = 84.495813.	16.090152
	FDB-SDO (+)	X_1 = 38.413759 X_5 = 89.999999	X_2 = 52.858081.	X_3 = 70.471953.	X_4 = 84.495716	16.090069

FDB-SDO ve SDO arasındaki karşılaştırma sonuçlarına göre yirmi adet mühendislik tasarım probleminin 15'inde FDB-SDO algoritmasının rakibine üstünlüğü bulunmaktadır. Bu sonuç, önerilen algoritmanın kısıtsız problemlerde olduğu gibi kısıtlı mühendislik problemlerinde de SDO algoritmasının arama yeteneklerini iyileştirdiğinin açık bir kanıtıdır. Yirmi problemin beşinde ise algoritmalar aynı sonucu bulmuşlardır.

5. Sonuç ve Tartışma (Results and Discussion)

Bu makale çalışmasında güncel bir meta-sezgisel arama algoritması olan SDO'nun genel arama performansını iyileştirmek amacıyla arama süreci yaşam döngüsünde seçim işlemleri aşamasında tasarımsal değişiklikler yapılmıştır. Bu amaçla, Fitness-Distance Balance (FDB) isimli güncel bir şeçim yönteminden faydalanılmıştır. FDB seçim yöntemi, MSA algoritmalarında arama sürecine rehberlik eden çözüm adaylarının etkili bir şekilde belirlenmesini sağlamaktadır. FDB sayesinde SDO algoritması taklit ettiği doğal süreçle daha uyumlu olacak şekilde tasarlanabilmiştir. FDB seçim yöntemi kullanılarak emtia fiyatlandırma modelleri üzerine yürütülen çalışmalar neticesinde emtia pazarındaki fiyat ve denge miktarları gerçek değerlerine yaklaştırılabilmiştir. Makalede geliştirilen FDB-SDO algoritmasının performansını test etmek ve doğrulamak amacıyla gerçekleştirilen deneysel çalışmalar CEC konferanslarında tanımlanan kurallar ve standartlar çerçevesinde yürütülmüştür. Deneysel çalışmalarda CEC 2017 karşılaştırma havuzu ve literatürde sıklıkla kullanılan mühendislik tasarım problemleri kullanılmıştır. Üstelik deneysel çalışmalar farklı problem türleri ve arama

uzayı boyutları için gerçekleştirilmiştir. Çalışmalardan elde edilen verilerin istatistiksel analiz yöntemleri ile değerlendirilmesi neticesinde makalede önerilen FDBSDO algoritmasının baz modele karşı üstün bir performans sergilediği istatistiksel analiz sonuçlarıyla açıkça ortaya koyulmuştur. Üstelik kısıtsız problemlerde olduğu gibi gerçek mühendislik tasarım problemlerinde de önerilen algoritmanın 20 problemin 15'inde rakibinden daha iyi performans sergilediği ve 5 problemde ise rakibiyle aynı performansı sergilediği görülmüştür. Bu sonuçlar, FDB yönteminin SDO algoritmasında hem yakınsama hem de çeşitlilik süreçlerinde önemli bir iyileşme etkisi yarattığına ve komşuluk araması-çeşitlilik arasında hassas bir denge kurduğuna işaret etmektedir.

6. Teşekkür (**Acknowledgement**)

Bu çalışmada yürütülen faaliyetler, 2020 yılında TÜBİTAK 2209-A Üniversite Öğrencileri Yurt İçi Araştırma Projeleri Destek Programı kapsamında 1919B011903812 numaralı proje olarak TUBİTAK tarafından desteklenmiştir.

Çıkar Çatışması (Conflict of Interest)

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

Kaynaklar (References)

Abedinpourshotorban, H., Shamsuddin, S. M., Beheshti, Z., & Jawawi, D. N. (2016). Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. Swarm and Evolutionary Computation, 26, 8-22.

Amir M.: Towards An Approach For Effectively Using Intuition In Large-Scale Decision-Making Problems, PhD Thesis, University of Debrecen (2013).

Aras, S., Gedikli, E., & Kahraman, H. T. (2020). A novel Stochastic Fractal Search Algorithm with Fitness-Distance Balance for Global Numerical Optimization. Swarm and Evolutionary Computation, 100821.

Awad, N. H., Ali, M. Z., Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2016). Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization. Tech. Rep.

Barshandeh, S., & Haghzadeh, M. (2020). A new hybrid chaotic atom search optimization based on tree-seed algorithm and Levy flight for solving optimization problems. Engineering with Computers, 1-44.

Carrasco, J., García, S., Rueda, M. M., Das, S., & Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. Swarm and Evolutionary Computation, 54, 100665.

Chen, H., Xu, Y., Wang, M., & Zhao, X. (2019). A balanced whale optimization algorithm for constrained engineering design problems. Applied Mathematical Modelling, , 71, 45-59.

Cheng, Min-Yuan, and Doddy Prayogo. "Symbiotic organisms search: a new metaheuristic optimization algorithm." Computers & Structures 139 (2014): 98-112.

Del Ser, J., Osaba, E., Molina, D., Yang, X. S., Salcedo-Sanz, S., Camacho, D., ... & Herrera, F. (2019). Bio-inspired computation: Where we stand and what's next. Swarm and Evolutionary Computation, 48, 220-250.

Demir, F.B., Tuncer, T. & Kocamaz, A.F. A chaotic optimization method based on logistic-sine map for numerical function optimization. Neural Comput & Applic (2020). https://doi.org/10.1007/s00521-020-04815-9

Dong, M., Wang, N., Cheng, X., Jiang, C.: Composite differential evolution with modified oracle penalty method for constrained optimization problems. Mathematical problems in engineering, 1-15 (2014), http://dx.doi.org/10.1155/2014/617905.

Dorigo, M., & Di Caro, G. (1999, July). Ant colony optimization: a new meta-heuristic. In Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406) (Vol. 2, pp. 1470-1477). IEEE.

Eberhart, R., & Kennedy, J. (1995, October). A new optimizer using particle swarm theory. In Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on (pp. 39-43). IEEE.

Eftimov, T., Korošec, P., & Seljak, B. K. (2017). A novel approach to statistical comparison of meta-heuristic stochastic optimization algorithms using deep statistics. Information Sciences, 417, 186-215.

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

Glover, F., & Hao, J. K. (2019). Diversification-based learning in computing and optimization. Journal of Heuristics, 25(4-5), 521-537.

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

Holland, J.H., 1975. Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. Q. Rev. Biol. 1, 211. http://dx.doi.org/10.1086/418447.

Ibrahim, R. A., Elaziz, M. A., Oliva, D., Cuevas, E., & Lu, S. (2019). An opposition-based social spider optimization for feature selection. Soft Computing, 23(24), 13547-13567.

Kahraman, H. T., Aras, S., & Gedikli, E. (2020). Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms. Knowledge-Based Systems, 190, 105169.

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.

Kumar, A., Wu, G., Ali, M. Z., Mallipeddi, R., Suganthan, P. N., & Das, S. (2020). A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evolutionary Computation, 100693.

Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.

Lin, X., Zhang, F., Xu, L.: Design of Gear Reducer Based on FOA Optimization Algorithm. In International Conference on Smart Vehicular Technology, Transportation, Communication and Applications, pp. 240-247. Springer, Cham (2017).

Long, W., Wu, T., Liang, X., Xu, S.: Solving high-dimensional global optimization problems using an improved sine cosine algorithm. Expert systems with applications 123, 108-126 (2019).

Mirjalili, S. (2015). The ant lion optimizer. Advances in engineering software, 83, 80-98.

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

Mozaffari, A., Emami, M. & Fathi, A. A comprehensive investigation into the performance, robustness, scalability and convergence of chaos-enhanced evolutionary algorithms with boundary constraints. Artif Intell Rev 52, 2319–2380 (2019). https://doi.org/10.1007/s10462-018-9616-4

P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger and S. Tiwari, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization", Technical Report, Nanyang Technological University, Singapore, May 2005 AND KanGAL Report #2005005, IIT Kanpur, India.

Pierezan, J., & Coelho, L. D. S. (2018, July). Coyote optimization algorithm: a new metaheuristic for global optimization problems. In 2018 IEEE Congress on Evolutionary Computation (CEC) (pp. 1-8). IEEE.

Piotrowski, A. P., Napiorkowski, J. J., & Rowinski, P. M. (2014). How novel is the “novel” black hole optimization approach?. Information Sciences, 267, 191-200.

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

Ravindran, A., Reklaitis, G. V., & Ragsdell, K. M. (2006). Engineering optimization: methods and applications. John Wiley & Sons.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: theory and application. Advances in Engineering Software, 105, 30-47.

Sayed, G.I., Tharwat, A. & Hassanien, A.E. Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection. Appl Intell 49, 188–205 (2019). https://doi.org/10.1007/s10489-018-1261-8

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.

Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In Simulated annealing: Theory and applications (pp. 7-15). Springer, Dordrecht.

Yang, X. S., & Deb, S. (2009, December). Cuckoo search via Lévy flights. In 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC) (pp. 210-214). IEEE.

Zhao, W., Wang, L., & Zhang, Z. (2019). Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowledge-Based Systems, 163, 283-304.

Zhao, W., Wang, L., & Zhang, Z. (2019). Supply-demand-based optimization: a novel economics-inspired algorithm for global optimization. IEEE Access, 7, 73182-73206.

Zhao, W., Zhang, Z., & Wang, L. (2020). Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. Engineering Applications of Artificial Intelligence, 87, 103300.