Avrupa Bilim ve Teknoloji Dergisi
Sayı 21, S. 493-500, Ocak 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 21, pp. 493-500, January 2021

Research Article

100 Basamak Probleminin Jade Algoritması ile Çözülmesi

Gürcan Yavuz[1*]

[1*] Kütahya Dumlupınar Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Kütahya, Türkiye, (ORCID: 0000-0002-2540-1930), gurcan.yavuz@dpu.edu.tr

(İlk Geliş Tarihi 11 Aralık 2020 ve Kabul Tarihi 26 Ocak 2021)

(**DOI:** 10.31590/ejosat.839083)

ATIF/REFERENCE: Yavuz, G., (2021). 100 Basamak Probleminin Jade Algoritması ile Çözülmesi. *Avrupa Bilim ve Teknoloji Dergisi*, (21), 493-500.

Öz

Gerçek parametre optimizasyon problemlerinin çözümü için metasezgisel algoritmalara sıklıkla başvurulmaktadır. Bu algoritmalar, problemlerin çözümüne uygulanmadan önce tasarımcıları tarafından yeterli performans elde edene kadar test edilirler. Tasarımcılar önerdikleri algoritmaları test etmek için literatürde sunulmuş çok sayıda sentetik fonksiyon setleri yer almaktadır. Bunlardan bir tanesi de CEC 2019 yarışmasında yer alan ve "100 basamak problemi" olarak adlandırılmış settir. Gerçek parametre optimizasyonunun çözümü için başvurulan önemli algoritmalardan biri de Diferansiyel Gelişim (DE) algoritmasıdır. Basit yapısı, kolay gerçeklenebilmesi ve elde ettiği başarılı sonuçlar DE'nin yaygın kullanılmasına ve performansının iyileştirilerek yeni varyantların ortaya çıkmasına yol açmıştır. Literatürdeki DE varyantlarının en bilinenlerinin başında JADE algoritması gelmektedir. Bu çalışmada, CEC 2019 yarışmasına ait olan 100 basamak probleminin çözümü JADE algoritması kullanılarak gerçekleştirilmiştir. Elde edilen sonuçlar, iki adet metasezgisel ile karşılaştırılmıştır. Bunlar; Diferansiyel Gelişim ve Yapay Arı Kolonisi (ABC) algoritmalarıdır. Üç algoritmanın katıldığı deneylerin adil bir şekilde yapılması için otomatik parametre aracı ile algoritmaların parametreleri yapılandırılmıştır. Ayrıca, deneylere katılan bütün algoritmalar farklı fonksiyon çağrım sayıları (FES) ile çalıştırılarak algoritmaların çalışma davranışları incelenmiştir. Sonuçlar göstermiştir ki, JADE çalıştırıldığı her FES değerinde karşılaştırıldığı algoritmalardan daha iyi sonuçlar elde etmiştir. Ayrıca FES değeri artırıldıkça algoritmanın başarımının iyileştiği görülmüştür.

Anahtar Kelimeler: JADE, CEC 2019, 100 basamak problemi, Optimizasyon.

Solving 100-Digit Challenge with Jade Algorithm

Abstract

Metaheuristic algorithms are frequently used to solve real parameter optimization problems. Before these algorithms are applied to the solution of problems, they are tested by their designers until they have enough performance. There are many benchmark sets presented in the literature to test the algorithms proposed by the designers. One of them is the one included in the CEC 2019 competition and named as the "100-digit problem". One of the most important algorithms used for the solution of real parameter optimization is the Differential Evolution (DE) algorithm. Its simple structure and easy implementation have led to the widespread use of DE and the emergence of new variants by improving its performance. In this study, the solution of the 100-digit problem belonging to the CEC 2019 competition was carried out using the JADE algorithm. The results obtained are compared with two metaheuristics. These algorithms are Differential Evolution and Artificial Bee Colony (ABC). For the experiments involving three algorithms to be fair, the parameters were configured with the automatic parameter tool. Besides, all algorithms participating in the experiments were run with different function evaluation numbers (FES) and the working behavior of the algorithms was examined. The results showed that; JADE achieved better results than other algorithms at all FES values. Also, as the FES value increased, the performance of the JADE algorithm improved.

Keywords: JADE, CEC 2019, 100-Digit Challenge, Optimization.

[*] Sorumlu Yazar: gurcan.yavuz@dpu.edu.tr

1. Giriş

Diferansiyel Gelişim (Differential Evolution, DE) algoritması, 1995 yılında Storn ve Price tarafından geliştirilmiş popülasyon tabanlı bir algoritmadır (R Storn & Price, 1995). Algoritmanın yapısının basit oluşuna karşın gösterdiği üstün başarı algoritmaya ilginin artmasına sebep olmuştur. Bu ilgi, algoritmanın üzerine yapılan çalışmaların çok fazla olmasına yol açmıştır (Çelik, Yıldız, & Karadeniz, 2019; Özyön, 2020). Bu çalışmaların başında DE'ye iyileştirmeler getirilerek çeşitli varyantlarının ortaya çıkması yer alır.

DE'nin ortaya çıkışından bu yana algoritma üzerine yapılan iyileştirilmeler ile algoritmanın performansı artırılmaya çalışılmıştır. Bu iyileştirilmelerin yapıldığı ilk yer algoritmanın ilklendirme adımında gerçekleştirilmiştir. Rahnamayan vd. DE algoritmasına karşıt-tabanlı öğrenme yöntemini entegre etmişlerdir (Rahnamayan, Tizhoosh, & Salama, 2008). Yüzgeç ve Eser, DE başlangıç popülasyonunu belirlemek için kaotik sistemden faydalanmışlardır (Yüzgeç & Eser, 2018). Bunun dışında yapılan bir diğer değişiklik, farklı mutasyon stratejilerinin önerilmesidir. Piotrowski global ve yerel komşuluk tabanlı mutasyon operatörü önermiştir (Piotrowski, 2013). Cai vd., arama yeteneklerini geliştirmek için yönlü bir mutasyon operatörü önermişlerdir (Cai vd., 2017). Bir başka iyileştirme de çaprazlama adımında yapılan geliştirmelerdir. Guo ve Yang, eigen vektör tabanlı bir çaprazlama operatörü önermişlerdir (Guo & Yang, 2014). Fan ve Zhang, uyarlanabilir çaprazlama stratejileri önermişlerdir (Fan & Zhang, 2016). DE her ne kadar performanslı bir algoritma olsa da başka optimizasyon teknikleri ile melezleştirme işlemi yapılmıştır. Zhang vd., DE algoritmasını Guguklu Arama ve Krill Sürü Optimizasyonu ile melezleştirmişlerdir (Z. Zhang, Dong, & Gao, 2016). Liu, Cai ve Wang ise Parçacık Sürü Optimizasyon algoritması ile DE'yi melezleştirerek kısıtlı optimizasyon problemlerinin çözümünde kullanmışlardır (Liu, Cai, & Wang, 2010). Bir diğer geliştirme de DE'nin sahip olduğu parametrelerini belirleme yöntemleridir. Kumar, Mandal ve Chakraborty DE'ye ait parametreleri kaotik haritalamalar kullanılarak belirlemişlerdir (Kumar, Mandal, & Chakraborty, 2019). Bir diğer parametre belirleme yöntemi ise uyarlanabilir olarak parametrelerin belirlenmesidir. Bunlara örnek olarak, (Ye vd., 2014) (J. Zhang & Sanderson, 2009) (Piotrowski, 2018) verilebilir.

Price vd. CEC 2019 gerçek parametre optimizasyonu için 100 basamak problemini sunmuşlardır (K. V. Price, N. H. Awad, M. Z. Ali, 2018). Bu problem, birbirinden zor ve karmaşık 10 adet problem içermektedir. Buradaki amaç, 10 problemin global optimum değerini 10 basamak doğruluk seviyesine kadar hesaplayarak 100 tam puanı elde etmektir. Literatürde bu ölçüt setinin çözümü için çeşitli metasezgisel algoritmalar sunulmuştur. Xu vd. iki tane güçlü optimizasyon algoritması PSO ve CMA-ES algoritmalarını hibritleştirme yoluna gitmişler ve CEC 2019 problemlerinin çözümlerinde kullanmışlardır (Xu, Luo, Lin, Qiao, & Zhu, 2019). Salgotra vd. önerdikleri adaptif parametre yöntemi ile SALSHADE-cnEpSin adını verdikleri yeni bir DE varyantı önermişler ve CEC 2019 ölçüt setindeki problemleri çözmüşlerdir (Salgotra, Singh, Saha, & Nagar, 2019). Pelusi vd. de ortaya koydukları Kütleçekim Arama algoritması varyantı olan Hiperbolik Kütleçekim Arama Algoritmasının performansını bu ölçüt seti ile test etmişlerdir (Pelusi vd., 2020). Lu vd. CEC 2019 problemleri için İş Bölümü Teorisi' ne dayanan DLABC adını verdikleri bir ABC algoritması önermişlerdir (Lu vd., 2019). Bugar ve Beyhan ise, optimizasyon problemlerinin çözümü için önerdikleri yeni bir metasezgisel olan Ergen Kimlik Arama Algoritmasının (AISA) başarısını CEC 2019 ölçüt seti kullanarak test etmişlerdir (Bogar & Beyhan, 2020). Bu çalışmada ise, uyarlanabilir ve önemli birkaç DE varyantından biri (Piotrowski & Napiorkowski, 2018) olan JADE algoritması ile bu 100 basamak problemi çözülmüş ve çeşitli deneyler gerçekleştirilmiştir. Ayrıca, algoritmanın farklı fonksiyon çağrım sayılarındaki davranışı incelenmiştir. Elde edilen sonuçlar, Yapay Arı Kolonisi (Karaboga & Basturk, 2007) ve Diferansiyel Gelişim algoritmaları ile karşılaştırılmıştır.

Bu çalışma şu şekilde organize edilmiştir. Birinci bölümde, çalışmayı kısaca tanıtan Giriş kısmı yer almaktadır. İkinci bölümde, çalışmada kullanılan JADE algoritmasının detaylarına yer verilmiştir. Üçüncü bölümde, 100 basamak problemi deneyleri ve elde edilen sonuçlar paylaşılmıştır. Son bölüm olan dördüncü bölümde ise, yapılan çalışma değerlendirilerek elde edilen sonuçlar yorumlanmıştır.

2. Jade Algoritması

JADE, Zhang ve Sanderson tarafından 2009 yılında önerilmiş olan bir uyarlanabilir DE varyantıdır (J. Zhang & Sanderson, 2009). Orijinal DE algoritmasından ayrılan iki iyileştirmeye sahiptir. Birincisi, algoritmanın seçme aşamasında başarısız olan bireylerin bilgileri arşivlenir ve mutasyon adımında arşivden rastgele seçimine dayanan "current-to-pbest/1" mutasyon stratejisine sahip olmasıdır. İkincisi ise, algoritmanın mutasyon ve çaprazlama oranlarını belirleyen F ve CR değerlerinin belirlenmesi için uyarlanabilir yöntemler bulunmasıdır. JADE'nin sözde kodu Algoritma 1'de verilmiştir.

JADE algoritması, problem boyutu sınırlarına uyarak rastgele bireyler üreterek çalışmaya başlamaktadır. Buna ek olarak, seçme adımında seçilemeyen bireyleri tutan A arşivi, başarılı olduğu kabul edilen F değerlerini tutan S_F ve başarılı olarak tanımlanan CR değerlerini tutan S_{CR} oluşturulmaktadır.

İlklendirme adımından sonra algoritma, bitme kriteri sağlanıncaya kadar çalışan birbirlerini takip eden iterasyonlara girmektedir. Bu iterasyonların her birinde Orijinal DE'den farklı olarak JADE de her birey için ayrı F ve CR değerleri atanmaktadır. Değer atanması her iterasyonda uyarlanabilir olarak yapılmaktadır. Değerler belirlenirken önceki adımdaki güncellemelerde başarılı olmuş F ve CR değerleri S_{CR} ve S_F kümelerinde saklanmaktadır. Bu kümelerde yer alan bilgiler ile yeni F ve CR değerleri üretilmektedir.

CR parametresi Denklem (1)'de Cauchy dağılım vasıtasıyla, F parametresi ise Denklem (3) ile normal dağılım kullanılarak, üretilmektedir.

$$CR_i = randn(\mu_{CR}, 0.1) \tag{1}$$

Denklemde yer alan μ_{CR}, Cauchy dağılımının yer parametresini göstermektedir. Algoritma başlangıcında ilk değeri 0.5 olarak atanmaktadır. Ölçek parametresi de 0.1 olarak belirlenmiştir. İterasyonlar ilerledikçe aşağıdaki denklem ile parametre güncellenmektedir.

$$\mu_{CR} = (1-c)\mu_{CR} + c\, mean_A(S_{CR}) \tag{2}$$

Burada, c değeri 0 ile 1 arasında belirlenmiş olan pozitif bir sayıdır. $mean_A$, normal aritmetik ortalamayı göstermektedir. S_{CR} ise iterasyonlarda başarılı olan çaprazlama olasılıklarının kümesini göstermektedir.

$$F_i = randc(\mu_F, 0.1) \quad (3)$$

μ_F; normal dağılımın ortalaması, 0.1 de varyansıdır. μ_F değeri algoritma başlangıcında 0.5 olarak atanır, iterasyonlar ilerledikçe değeri Denklem (4) ile güncellenir.

$$\mu_F = (1-c)\mu_F + c\, mean_L(S_F) \quad (4)$$

SF, önceki iterasyonlarda başarılı olan F değerlerinin tutulduğu kümedir. $mean_L$, ise Lehmer ortalamasını göstermektedir ve aşağıda yer alan denklem kullanılarak hesaplanmaktadır:

$$mean_L(S_F) = \frac{\sum F \in S_F F^2}{\sum F \in S_F F} \quad (5)$$

F ve CR değerlerinin belirlenmesinden sonra algoritma, mutasyon adımına geçmektedir. JADE, mutasyon stratejisi olarak "current-to-pbest/1" denklemini kullanmaktadır. Bu yaklaşımda bireylerin bilgileri bir arşive kaydedilmekte ve daha sonra bu arşivden rastgele seçilmektedir. JADE algoritması, mutasyon vektörünü şu şekilde üretmektedir:

$$v_i = x_i + F_i \times (x_{pbest} - x_i) + F_i \times (x_{r1} - x_{r2}) \quad (6)$$

Denklemde bulunan x_{pbest}, popülasyondaki en iyi 100 p% birey içerisinden rastgele seçilmiş olan bireyi göstermektedir. Burada $p \in [0,1]$ bir sayıdır. F_i, değeri x_i bireyine ait olan mutasyon faktörünü temsil etmektedir. Bu değer her iterasyonda yeniden belirlenmektedir. x_{r1}, popülasyondan rastgele seçilmiş olan bir bireydir. x_{r2}, algoritmanın ilklendirme adımında tanımlanmış olan A arşivi ile mevcut popülasyonun birleşiminin oluşturduğu kümeden rastgele seçilen bireyi göstermektedir. Burada bahsedilen A arşivi, seçme adımında seçilmeyen bireyler ile doldurulmaktadır.

Mutasyon adımından sonra algoritma, çaprazlama ve seçme adımları ile çalışmasına devam etmektedir. Çaprazlama adımında her bireyin (x_i) sahip olduğu çaprazlama oranı (CR_i) kullanılarak deneme (u_i) vektörleri üretilmektedir. Daha sonra seçme adımında, çaprazlama adımında üretilmiş olan deneme vektörü mevcut olan birey vektörünün fitness değeri ile karşılaştırılır ve hangisinin daha iyi olduğuna karar verilir. Deneme vektörü daha iyi ise, mevcut vektör A arşivine dahil edilir. Böylelikle, bu bireye ait olan bilgilerin kaybolması engellenmiş olur. Bu işlem, algoritmanın arama yönünü güçlendirmekte ve popülasyon çeşitliliğini artırmaktadır. Deneme vektörü mevcut vektör olarak atanır. Seçme adımında deneme vektörü mevcut vektörden daha iyi olduğu için bu iterasyonda kullanılan CR_i ve F_i değerleri bir sonraki iterasyondaki F ve CR değerlerini belirlemek için S_{CR} ve S_F kümelerine koyulur.

İterasyonlar tamamlanırken, μ_F ve μ_{CR} değerleri sırasıyla Denklem (4) ve (2) ile bir sonraki iterasyonda kullanılması için belirlenmektedir. Ayrıca A arşivinin boyutunun kontrolü yapılmaktadır. Algoritmanın ilklendirme adımında, A arşivinin boyutu popülasyonun boyutu olan NP kadar tanımlanmaktadır. Eğer ki, algoritma süresince A'nın boyutu NP değerini geçer ise A'dan rastgele eleman silinerek A'nın boyutunun NP'ye eşitlenmesi sağlanır.

Algoritma 1. JADE algoritmasının sözde kodu.

```
1: G = 1, N_G = NP_init
2: A, S_F, S_CR kümelerini oluştur
3: A = {}, NP, p ilk değerlerini ata
4: P_g popülasyonunu (P_g = x_1,G, ...., x_N,G) rastgele üret.
5:
6: μ_F , μ_CR değerlerini 0.5 olarak ata.
7: while bitme kriteri sağlanmadığı sürece do
8:    for i = 1: Np
9:       CR_i ve F_i değerlerini Denklem (1) ve Denklem (3) ile
üret
10:      Denklem (6) ile mutasyon işlemini vektörünü hesapla
12:      çaprazlama işlemi uygula
13:      For j=1:D
14:            if j = randint(1, D) veya rand(0,1) < CR_i
15:                  u_i,j = v_i,j
16:            Else
17:                  u_i,j = x_i,j
18:
19:      if f(u_i) ≤ f(x_i)
20:            A ← x_i , S_CR ← CR_i , S_F ← F_i
21:
22:      μ_F , μ_CR değerlerini Denklem (4) ve (2) ile hesapla
23:      A arşivinin boyutunu ayarla
24:
25: G = G + 1
```

3. Deneysel Sonuçlar

Bu çalışmada, önemli bir DE varyantı olan JADE algoritması ile 100 basamak problemi çözülmüştür. Algoritmanın elde ettiği sonuçlar, iki farklı metasezgisel olan ABC (Karaboga, 2005) ve DE (Rainer Storn & Price, 1995) algoritmaları ile karşılaştırılmıştır.

100 basamak problemi, CEC 2019 yarışmasında ölçüt seti olarak kullanılmıştır (Price, Awad, Ali, & Suganthan, 2018). Bu ölçüt seti 10 adet fonksiyon içermektedir ve özellikleri Tablo 1'de verilmiştir. Bu ölçüt setindeki amaç, ölçüt setinde yer alan her bir fonksiyonun değerini 10 basamağa kadar doğru olarak belirlemektir. Sette yer alan bütün fonksiyonların en küçük değeri 1.000000000'dir. Yarışmada, 10 fonksiyonun değeri 10 basamağa kadar doğru olarak belirlenip 100 puanlık skorun toplanmasına çalışılmaktadır. Fonksiyonların optimum değerleri belirlenirken (Price vd., 2018)'de tanımlamış kriterlere göre değerlendirme yapılmaktadır. Ölçüt setinde fonksiyonlar çözülürken bir bitirme kriteri olmadan çalıştırılmaktadır. Yani herhangi bir fonksiyon çağırım sayısı sınırı bulunmamaktadır.

Çalışmada kullanılan üç algoritma da, performanslarını etkileyen parametrelere sahiptirler. Bundan dolayı bu parametrelerin değerlerinin titizlikle seçilmesi gerekmektedir. Bunun için deneylerde, algoritmaların kontrol parametrelerini

belirlemek amacıyla otomatik parametre yapılandırma aracı irace'den faydalanılmıştır (López-Ibáñez, Dubois-Lacoste, Pérez Cáceres, Birattari, & Stützle, 2016).

irace, F-Race yöntemini tekrarlı şekilde uygulayan "iterated F-Race" metodunu kullanıcılarına bir R paketi olarak sunmaktadır. F-Race, aslında bir yarışma metodudur. F-Race, kendisine verilen parametreleri yapılandırılması istenen algoritmalar için aday parametre yapılandırmaları üretir ve yine kendisine verilen test setinde bunları yarıştırarak en iyi parametre değerlerini belirlemeye çalışır (Liao, Molina, & Stützle, 2015). Iterated F-Race ise bu işlemi tekrarlı olarak gerçekleştirir. Çalışması ise kabaca şu şekildedir: Başlangıçta F-Race, verilen algoritma için yine kendisine verilmiş olan test setine özgü aday yapılandırmalar üretir ve bu test seti üzerinde aday yapılandırmalarını algoritmaya uygulayarak algoritmayı çalıştırır. Deneme sonucunda, bütün aday yapılandırmaların içinde en iyi olan yapılandırmayı istatistiksel testler yardımıyla belirlemeye çalışır. Kötü olan aday yapılandırmaları denemelerden çıkarır. Böylelikle, birinci tur gerçekleşmiş olur. İkinci tura geçildiğinde ise bir önceki turun galibi olan aday yapılandırmanın bilgileri kullanılarak yeni aday yapılandırmalar üretilir. Bu tur için de yine en iyi yapılandırma istatistiksel testler ile belirlenir. Bu işlem F-Race çalıştırma bütçesi kadar devam eder. irace işlemini tamamladığında algoritmaya ait en iyi yapılandırma çıktı verilir.

Tablo 1. 100 basamak probleminde yer alan fonksiyonlar ve özellikleri

Id	Fonksiyonlar	Boyut	Aralık
1	Storn's Chebyshev Polynomial Fitting	9	[-8192, 8192]
2	Inverse Hilbert Matrix Problem	16	[-16384,
3	Lennard-Jones Minimum Energy	18	[-4, 4]
4	Rastrigin's Function	10	[-100, 100]
5	Griewangk's Function	10	[-100, 100]
6	Weierstrass Function	10	[-100, 100]
7	Modified Schwefel's Function	10	[-100, 100]
8	Expanded Schaffer's F6 Function	10	[-100, 100]
9	Happy Cat Function	10	[-100, 100]
10	Ackley Function	10	[-100, 100]

Bu çalışmada irace, ön tanımlı ayarları ile çalıştırılmıştır. Deneylerde kullanılan ölçüt seti eğitim ve deneme olmak üzere ikiye ayrılmıştır. Eğitim seti için, algoritmalar 100 basamak probleminde düşük FES değeri kullanılarak çalıştırılmıştır. Bu değer 1E+05 fonksiyon çağırım sayısı kadardır. JADE, DE ve ABC algoritmalarına ait olan kontrol parametrelerinin özellikleri ve irace aracının belirlemiş olduğu yapılandırılmış parametre değerleri Tablo 2'de verilmiştir.

JADE, DE ve ABC algoritmaları Cpp ile kodlanmıştır. Ölçüt setinde yer alan fonksiyonlar, Tablo 2'deki parametre değerleri algoritmalara uygulanarak çözülmüştür. Bütün algoritmalar her bir fonksiyon için 51 defa bağımsız olarak çalıştırılmıştır. Algoritmaların farklı fonksiyon çağırım sayılarındaki davranışlarının incelenmesi için 1E+05, 5E+05, 1E+06 ve 5E+06 FES değerlerinde çalıştırılmıştır. Çalıştırma işlemleri, AMD Ryzen 5 1600 3.2 GHz işlemcili ve 16 GB RAM belleğe sahip bilgisayarda gerçekleşmiştir.

Tablo 2. Deneylerde yer alan algoritmaların parametreleri, özellikleri, aralık değerleri ve irace ile belirlenmiş olan parametre değerleri. (r: reel sayı, i: tam sayı, c: kategorik)

Algo.	Param.	Tür	Aralıklar	Değerler	Açıklama
ABC	NP	r	(7, 300)	37	Popülasyon boyutu
	limitf	r	(0, 4)	3.21	Limit faktörü
DE	NP	r	(7, 300)	34	Popülasyon boyutu
	F	c	(0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0)	0.5	Mutasyon faktörü
	CR	c	(0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0)	0.9	Çaprazlama oranı
JADE	NP	r	(7, 300)	252	Popülasyon boyutu
	p	c	(0.05, 0.06, 0.07, 0.08, 0.09, 0.10, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.20, 0.21, 0.22, 0.23, 0.24, 0.25, 0.26, 0.27, 0.28, 0.29, 0.30)	0.06	Mutasyon açgözlülük oranı

Deneyler sonucunda JADE'nin doğru olarak belirlediği basamak sayılarına ait sonuçlar Tablo 3'te verilmiştir. Tablo 3a'da 5E+05 FES kadarlık bütçe kullanılarak elde edilen sonuçlar verilmiştir. F1 ve F2 fonksiyonlarında 10 puan alarak bütün basamakların doğru belirlenmiş olduğu gözükmektedir. F3, F4, F7 ve F10 fonksiyonlarında algoritma sadece bir basamağı doğru olarak belirlemiştir. F8 fonksiyonunda ise JADE 51 çalıştırmanın hiçbirinde doğru bir basamak belirleyememiştir. Tablo 3b de ise çalıştırma bütçesi artırılarak algoritma 1E+06 FES'lik bütçe ile çalıştırılmıştır. Algoritma; F1, F2 ve F6 fonksiyonlarında 51 çalıştırmanın hepsinde 10 basamağın tamamını doğru olarak hesaplamıştır. F3, F7, F8 fonksiyonlarında ise JADE sadece iki basamağı belirleyebilmiştir. F6 fonksiyonunda ise 51 çalıştırmanın büyük bir kısmında 4 ve 5 basamak belirlerken sadece üç tane çalıştırmada 10 basamak doğru olarak belirlemiştir. Son olarak Tablo 3c'de çalıştırma bütçesi daha da artılarak 5E+06 olarak seçilmiştir. Bu bütçe ile JADE'nin daha iyi sonuçlar aldığı görülmektedir. Tablodaki değerlere göre, F1, F2, F4, F5, F6 ve

F10 fonksiyonlarında 10 puan alarak tam puan almıştır. Geri kalan fonksiyonlarda algoritma sadece F7'de 51 çalıştırmanın bir tanesinde 10 basamağın tamamını belirleyebilmiştir. F3'de ise algoritma sadece 1 basamağın değerini doğru olarak hesaplayabilmiştir. Bütçe artırılması, bu fonksiyon için algoritmanın arama yeteneğini geliştirememiştir.

Tablo 3. JADE algoritmasının 51 çalıştırmanın sonuçları. (a)5E+05, (b)1E+06, (c)5E+06 fonksiyon çağırım sayısı için sonuçları göstermektedir.

Fonksiyonlar 5E+05 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	0	0	0	0	0	0	0	0	0	0	50	10
f2	5	0	0	0	0	0	0	0	0	0	45	10
f3	48	2	0	0	0	0	0	0	0	0	0	0.08
f4	49	1	0	0	0	0	0	0	0	0	0	0.04
f5	0	0	50	0	0	0	0	0	0	0	0	4
f6	0	0	1	1	29	13	1	1	1	0	3	9.48
f7	49	1	0	0	0	0	0	0	0	0	0	0.04
f8	50	0	0	0	0	0	0	0	0	0	0	0
f9	0	49	1	0	0	0	0	0	0	0	0	2.04
f10	18	32	0	0	0	0	0	0	0	0	0	1.28
											Toplam:	36.96

(a) 5E+05 FES

Fonksiyonlar 1E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	0	0	0	0	0	0	0	0	0	0	50	10
f2	7	1	0	0	0	0	0	0	0	0	42	10
f3	48	2	0	0	0	0	0	0	0	0	0	0.08
f4	30	15	5	0	0	0	0	0	0	0	0	1
f5	0	0	45	2	1	0	1	0	0	0	1	4.64
f6	0	0	0	0	0	0	0	0	0	0	50	10
f7	42	8	0	0	0	0	0	0	0	0	0	0.32
f8	48	2	0	0	0	0	0	0	0	0	0	0.08
f9	0	41	9	0	0	0	0	0	0	0	0	2.36
f10	29	21	0	0	0	0	0	0	0	0	0	0.84
											Toplam:	39.32

(b) 1E+06 FES

Fonksiyonlar 5E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	0	0	0	0	0	0	0	0	0	0	50	10
f2	1	0	0	0	0	0	0	0	0	0	49	10
f3	41	9	0	0	0	0	0	0	0	0	0	0.36
f4	0	0	0	0	0	0	0	0	0	0	50	10
f5	0	0	0	0	0	0	0	0	0	0	50	10
f6	0	0	0	0	0	0	0	0	0	0	50	10
f7	3	26	17	1	1	0	0	0	1	0	1	3.4
f8	10	39	1	0	0	0	0	0	0	0	0	1.64
f9	0	0	50	0	0	0	0	0	0	0	0	4
f10	9	3	0	0	0	0	0	0	0	0	38	10
											Toplam:	69.4

(c) 5E+06 FES

Tablo 4. ABC algoritmasının 51 çalıştırmanın sonuçları. (a)5E+05, (b)1E+06, (c)5E+06 fonksiyon çağırım sayısı için sonuçları göstermektedir

Fonksiyonlar 5E+05 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	46	3	1	0	0	0	0	0	0	0	0	0.2
f2	46	4	0	0	0	0	0	0	0	0	0	0.16
f3	0	39	2	6	3	0	0	0	0	0	0	2.92
f4	46	4	0	0	0	0	0	0	0	0	0	0.16
f5	0	0	34	10	1	0	3	1	0	1	0	5.44
f6	26	23	1	0	0	0	0	0	0	0	0	1
f7	44	6	0	0	0	0	0	0	0	0	0	0.24
f8	49	1	0	0	0	0	0	0	0	0	0	0.04
f9	0	11	39	0	0	0	0	0	0	0	0	3.56
f10	24	26	0	0	0	0	0	0	0	0	0	1.04
											Toplam:	14.76

(a) 5E+05 FES

Fonksiyonlar 1E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	47	3	0	0	0	0	0	0	0	0	0	0.12
f2	46	4	0	0	0	0	0	0	0	0	0	0.16
f3	0	36	0	9	5	0	0	0	0	0	0	3.32
f4	48	2	0	0	0	0	0	0	0	0	0	0.08
f5	0	0	13	17	2	9	3	0	1	2	3	8.16
f6	15	33	2	0	0	0	0	0	0	0	0	1.48
f7	46	4	0	0	0	0	0	0	0	0	0	0.16
f8	49	1	0	0	0	0	0	0	0	0	0	0.04
f9	0	3	47	0	0	0	0	0	0	0	0	3.88
f10	39	11	0	0	0	0	0	0	0	0	0	0.44
											Toplam:	17.84

(b) 1E+06 FES

Fonksiyonlar 5E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	40	9	1	0	0	0	0	0	0	0	0	0.44
f2	47	2	1	0	0	0	0	0	0	0	0	0.16
f3	0	12	3	11	23	1	0	0	0	0	0	5.92
f4	50	0	0	0	0	0	0	0	0	0	0	0
f5	0	0	0	13	2	2	8	3	3	3	16	10
f6	0	43	7	0	0	0	0	0	0	0	0	2.28
f7	42	8	0	0	0	0	0	0	0	0	0	0.32
f8	49	1	0	0	0	0	0	0	0	0	0	0.04
f9	0	0	50	0	0	0	0	0	0	0	0	4
f10	21	27	2	0	0	0	0	0	0	0	0	1.24
											Toplam:	24.4

(c) 5E+06 FES

Tablo 5. DE algoritmasının 51 çalıştırmanın sonuçları. (a)5E+05, (b)1E+06, (c)5E+06 fonksiyon çağırım sayısı için sonuçları göstermektedir.

Fonksiyonlar 5E+05 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	38	4	6	0	0	0	0	0	0	0	2	1.44
f2	47	3	0	0	0	0	0	0	0	0	0	0.12
f3	14	36	0	0	0	0	0	0	0	0	0	1.44
f4	47	3	0	0	0	0	0	0	0	0	0	0.12
f5	0	1	41	5	0	0	0	0	0	0	3	5.12
f6	11	13	0	0	0	0	0	0	0	0	26	10
f7	48	2	0	0	0	0	0	0	0	0	0	0.08
f8	48	2	0	0	0	0	0	0	0	0	0	0.08
f9	0	32	18	0	0	0	0	0	0	0	0	2.72
f10	0	49	0	0	0	0	0	0	0	0	1	2.36
											Toplam:	23.48

(a) 5E+05 FES

Fonksiyonlar 1E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	35	8	4	2	0	0	0	0	0	0	1	1.28
f2	49	1	0	0	0	0	0	0	0	0	0	0.04
f3	8	42	0	0	0	0	0	0	0	0	0	1.68
f4	45	5	0	0	0	0	0	0	0	0	0	0.2
f5	0	0	39	8	0	0	0	0	0	0	3	5.28
f6	10	18	0	0	0	0	0	0	0	0	22	9.52
f7	45	5	0	0	0	0	0	0	0	0	0	0.2
f8	49	1	0	0	0	0	0	0	0	0	0	0.04
f9	0	29	21	0	0	0	0	0	0	0	0	2.84
f10	0	48	0	0	0	0	0	0	0	0	2	2.72
											Toplam:	23.8

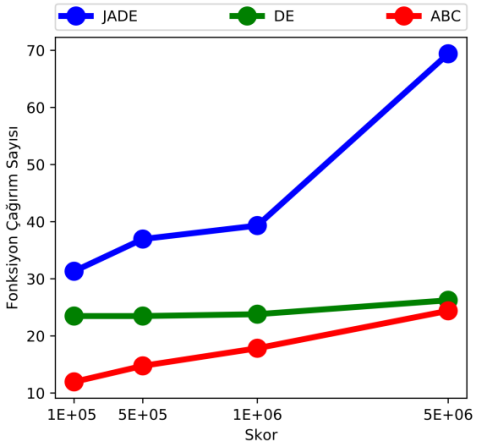
(b) 1E+06 FES

Fonksiyonlar 5E+06 FES	Doğru basamak sayısı											Skor
	0	1	2	3	4	5	6	7	8	9	10	
f1	23	3	3	3	4	2	4	2	2	0	4	5.52
f2	49	1	0	0	0	0	0	0	0	0	0	0.04
f3	12	38	0	0	0	0	0	0	0	0	0	1.52
f4	46	4	0	0	0	0	0	0	0	0	0	0.16
f5	0	2	39	8	0	0	0	0	0	0	1	4.56
f6	9	19	0	0	0	0	0	0	0	0	22	9.56
f7	44	6	0	0	0	0	0	0	0	0	0	0.24
f8	44	6	0	0	0	0	0	0	0	0	0	0.24
f9	0	40	10	0	0	0	0	0	0	0	0	2.4
f10	0	50	0	0	0	0	0	0	0	0	0	2
											Toplam:	26.24

(c) 5E+06 FES

JADE'nin elde ettiği sonuçların test edilmesi için ABC ve DE algoritmaları da çalıştırılmış ve elde ettikleri sonuçlar kaydedilmiştir. Bu algoritmalar da JADE'ye uygulanan şartlar altında test edilmişlerdir. ABC'nin 1E+05, 5E+05, 1E+06 ve 5E+06 fonksiyon çağırım sayılarındaki sonuçları Tablo 4'de listelenmiştir. DE'nin sonuçları ise Tablo 5'de verilmiştir.

Şekil 1. JADE, DE ve ABC algoritmalarının fonksiyon çağırım sayısı ilerledikçe elde ettiği skor değerlerinin değişimi



JADE, DE ve ABC'nin farklı çağırım sayılarındaki skor değerleri özet olarak Şekil 1'de verilmiştir. Şekil incelenirse, bütün FES değerlerinde JADE algoritması ABC ve DE'ye göre daha iyi skorlar elde etmiştir. 5E+05 FES'de JADE 36.96'lık skor değeri ile birinci olmuştur. 1E+06 FES'de ise 39.32, 5E+06 FES'de ise 69.4 skor değeri alarak diğer algoritmaların önünde yer almıştır. Diğer iki algoritma birbirleri ile karşılaştırıldığında ise DE algoritması düşük FES değerlerinde elde ettiği daha iyi skorlar ile ABC'nin önünde yer alır. Ancak FES değeri artırıldıkça ABC algoritması iyileşmekte ve skor değeri DE'ye yaklaşmakta olduğu görülmektedir.

4. Sonuç

Bu çalışmada, uyarlanabilir bir DE algoritması varyantı olan JADE algoritması ile 100 basamak probleminin çözümü gerçekleştirilmiştir. Ayrıca bu problemin çözümü DE ve ABC ile

de gerçekleştirilerek bu algoritmaların bu problem setindeki davranışları incelenmiştir. Yapılan deneyler; düşük, orta ve yüksek fonksiyon çağırım sayısı olmak üzere üç farklı FES ile gerçekleştirilmiştir. Kullanılan FES değerleri; 5E+05, 1E+06 ve 5E+06'dır. Düşük FES değeri ile yapılan deneylerde ABC, DE ve JADE birbirlerine yakın değerler almışlardır. FES değeri 5E+06'ya çıktığında JADE 69.4'luk skor değeri alarak diğer algoritmalar ile arayı açmıştır. Yani FES değeri yükseldikçe JADE'nin gösterdiği performans da artmıştır. Deneylerden elde edilen sonuçlar göstermiştir ki; bütün FES değerlerinde JADE algoritmasının elde ettiği skorlar, karşılaştırıldığı diğer algoritmalardan daha yüksek olmuştur. Bunun anlamı, JADE problem setindeki fonksiyonların global optimum değerlerini daha fazla sayıda doğru basamak sayısı ile hesaplayabilmiştir. Deneylerden çıkarılan bir diğer sonuç ise, ABC algoritması düşük FES değerlerinde en geride kalan algoritma olmuştur. Ancak FES değeri yükseldikçe ABC ikinci sıradaki algoritma olan DE'yi yakalamıştır. Yani, FES değeri yükseldikçe ABC'nin performansı iyileştiği söylenebilir. Buna karşın düşük, orta ve yüksek FES değerlerinin tamamında DE algoritmasının elde ettiği sonuçların birbirine yakın olduğu görülmüştür. FES değerinin yükselmesinin DE algoritma performansına bir katkısı olmamıştır.

İlerleyen çalışmalarda, JADE algoritmasına yeni stratejiler eklenerek algoritmanın performansı artırılabilir ve deneyler tekrar gerçekleştirilebilir. Ayrıca FES değeri daha da yükseltilerek problem çözülebilir. Aynı zamanda farklı uyarlanabilir DE varyantları da deneylere dahil edilebilir.

Kaynakça

Bogar, E., & Beyhan, S. (2020). Adolescent Identity Search Algorithm (AISA): A novel metaheuristic approach for solving optimization problems. Applied Soft Computing Journal, 95, 106503. https://doi.org/10.1016/j.asoc.2020.106503

Cai, Y., Sun, G., Wang, T., Tian, H., Chen, Y., & Wang, J. (2017). Neighborhood-adaptive differential evolution for global numerical optimization. Applied Soft Computing Journal, 59, 659–706. https://doi.org/10.1016/j.asoc.2017.06.002

Çelik, Y., Yıldız, İ., & Karadeniz, A. T. (2019). Son Üç Yılda Geliştirilen Metasezgisel Algoritmalar Hakkında Kısa Bir İnceleme. Avrupa Bilim ve Teknoloji Dergisi. Osman SAĞDIÇ. https://doi.org/10.31590/ejosat.638431

Fan, Q., & Zhang, Y. (2016). Self-adaptive differential evolution algorithm with crossover strategies adaptation and its application in parameter estimation. Chemometrics and Intelligent Laboratory Systems, 151, 164–171.

Guo, S.-M., & Yang, C.-C. (2014). Enhancing differential evolution utilizing eigenvector-based crossover operator. IEEE Transactions on Evolutionary Computation, 19(1), 31–49.

K. V. Price, N. H. Awad, M. Z. Ali, P. N. S. (2018). Problem Definitions and Evaluation Criteria for the 100-Digit Challenge Special Session and Competition on Single Objective Numerical Optimization, (November), 22. Tarihinde adresinden erişildi http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2019

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization.

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, S., Mandal, K. K., & Chakraborty, N. (2019). Optimal DG placement by multi-objective opposition based chaotic differential evolution for techno-economic analysis. Applied Soft Computing, 78, 70–83.

Liao, T., Molina, D., & Stützle, T. (2015). Performance evaluation of automatically tuned continuous optimizers on different benchmark sets. Applied Soft Computing Journal, 27, 490–503. https://doi.org/10.1016/j.asoc.2014.11.006

Liu, H., Cai, Z., & Wang, Y. (2010). Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. Applied Soft Computing, 10(2), 629–640.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. Operations Research Perspectives, 3, 43–58. https://doi.org/10.1016/j.orp.2016.09.002

Lu, J., Zhou, X., Ma, Y., Wang, M., Wan, J., & Wang, W. (2019). A Novel Artificial Bee Colony Algorithm with Division of Labor for Solving CEC 2019 100-Digit Challenge Benchmark Problems. 2019 IEEE Congress on Evolutionary Computation, CEC 2019- Proceedings, 1, 387–394. https://doi.org/10.1109/CEC.2019.8790252

Özyön, S. (2020). Yenilenebilir Enerji Üretim Birimleri İçeren Çevresel-Ekonomik Güç Dağıtımı Probleminin Yüklü Sistem Arama Algoritması ile Çözümü. Avrupa Bilim ve Teknoloji Dergisi, 81–90. https://doi.org/10.31590/ejosat.669543

Pelusi, D., Mascella, R., Tallini, L., Nayak, J., Naik, B., & Deng, Y. (2020). Improving exploration and exploitation via a Hyperbolic Gravitational Search Algorithm. Knowledge-Based Systems, 193(xxxx), 105404. https://doi.org/10.1016/j.knosys.2019.105404

Piotrowski, A. P. (2013). Adaptive memetic differential evolution with global and local neighborhood-based mutation operators. Information Sciences, 241, 164–194.

Piotrowski, A. P. (2018). L-SHADE optimization algorithms with population-wide inertia. Information Sciences, 468, 117–141. https://doi.org/10.1016/j.ins.2018.08.030

Piotrowski, A. P., & Napiorkowski, J. J. (2018). Step-by-step improvement of JADE and SHADE-based algorithms: Success or failure? Swarm and Evolutionary Computation, 43(August 2017), 88–108. https://doi.org/10.1016/j.swevo.2018.03.007

Price, K. V, Awad, N. H., Ali, M. Z., & Suganthan, P. N. (2018). Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. Içinde Technical Report. Nanyang Technological University.

Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2008). Opposition-Based Differential Evolution. IEEE Transactions on Evolutionary Computation, 12(1), 64–79. https://doi.org/10.1109/TEVC.2007.894200.

Salgotra, R., Singh, U., Saha, S., & Nagar, A. (2019). New Improved SALSHADE-cnEpSin Algorithm with Adaptive Parameters. 2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings, 3150–3156. https://doi.org/10.1109/CEC.2019.8789983

Storn, R, & Price, K. (1995). Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute, 11(TR-95-012), 1–15. https://doi.org/10.1023/A:1008202821328

Storn, Rainer, & Price, K. (1995). Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces (C. 3).

Xu, P., Luo, W., Lin, X., Qiao, Y., & Zhu, T. (2019). Hybrid of PSO and CMA-ES for Global Optimization. 2019 IEEE Congress on Evolutionary Computation, CEC 2019 - Proceedings, 27–33. https://doi.org/10.1109/CEC.2019.8789912

Ye, S., Dai, G., Peng, L., Wang, M., Sishi, Y., Guangming, D., Maocai, W. (2014). A hybrid adaptive coevolutionary differential evolution algorithm for large-scale optimization. Evolutionary Computation (CEC), 2014 IEEE Congress on, 1277–1284. https://doi.org/10.1109/CEC.2014.6900259

Yüzgeç, U., & Eser, M. (2018). Chaotic based differential evolution algorithm for optimization of baker's yeast drying process. Egyptian Informatics Journal, 19(3), 151–163. https://doi.org/10.1016/j.eij.2018.02.001

Zhang, J., & Sanderson, A. C. (2009). JADE: adaptive differential evolution with optional external archive. Evolutionary Computation, IEEE Transactions on, 13(5), 945–958.

Zhang, Z., Dong, Y., & Gao, T. (2016). A Hybrid Method Based on Cuckoo Search and Krill Herd Optimization with Differential Evolution. Içinde 2016 13th Web Information Systems and Applications Conference (WISA) (ss. 138–143).