Optimizasyon Problemlerinin Çözümü için Parçaçık Sürü Optimizasyonu Algoritması

M. Yasin ÖZSAĞLAM, Mehmet ÇUNKAŞ

ÖZET

Optimizasyon problemlerinin çözümü için kullanılan bir çok optimizasyon tekniği doğadaki olaylardan esinlenilerek geliştirilmiştir. Parçacık Sürü Optimizasyonu (PSO), kuş ve balık sürülerinin sosyal davranışları gözlemlenerek geliştirilen popülasyon temelli bir optimizasyon algoritmasıdır. Bu makalede PSO ile Genetik Algoritma (GA) ve Diferansiyel Evrim Algoritmasının(DEA) performansları, test fonksiyonları kullanılarak karşılaştırılmaktadır. Elde edilen sonuçlardan, PSO nun her iki algoritmaya göre yakınsama hızı ve performans bakımından daha iyi çözümler ürettiği görülmektedir.

Anahtar Kelimeler: Parçacık Sürü Optimizasyonu, Genetik Algoritmalar, Diferansiyel Evrim Algoritması

Particle Swarm Optimization Algorithm for Solving Optımızation Problems

ABSTRACT

Many optimization techniques used in solving optimization problems has been developed by inspiring from the events in nature. Particle Swarm Optimization (PSO) algorithm is a population based optimization technique inspired by social behavior of bird flocking and fish schooling. In this paper, PSO is compared with Genetic Algorithms and Differential Evolution Algorithm by using test functions. The results show that the PSO, in most problems, is able to find much better solutions and better convergence compared to other two algorithms.

Keywords: Particle Swarm Optimization, Genetic Algorithms, Differential Evolution Algorithm.

1. GİRİŞ

Optimizasyon, verilen amaç veya amaçlar için belirli kısıtlamaların sağlanarak en uygun çözümün elde edilme sürecidir. Bilim adamları yeni bir fikir ortaya koyar ve optimizasyon aracılığıyla bu fikir geliştirilir. Bir fikri etkileyen parametre veya bilgi elektronik formata dönüştürülebildiği sürece, bilgisayar mükemmel bir optimizasyon aracıdır. Optimizasyon terminolojisinde her zaman en iyiye ulaşma arzusu söz konusudur. En iyi tanımlaması probleme, çözüm metoduna ve izin verilen toleransa bağlıdır. Geçmişten günümüze, karşılaşılan problemlerin çözülmesi amacıyla bir çok optimizasyon teknikleri geliştirilerek değişik alanlara uygulanmıştır (1).

Optimizasyon problemlerinin çözümünde klasik yöntemler olarak adlandırılan matematiksel yöntemler önceleri çok yaygın olarak kullanılmaktaydı. Bu tür yöntemlerin esnek olmaması ve matematiksel fonksiyonlarla tanımlama gereksinimi gibi dez avantajları, son zamanlarda, bilim adamlarında genel amaçlı ve performansı yüksek yöntemler geliştirme çabalarını artırmış ve doğadaki olaylardan esinlenmeye başlamışlardır. Tabiattaki olaylar temel alınarak geliştirilen optimizasyon algoritmaları sezgisel yöntemler olarak adlandırılmaktadır (2). Bunlardan Genetik Algoritmalar(GA), Diferansiyel evrim algoritması ve Parçacık Sürü optimizasyonu algoritmaları, optimizasyon problemlerinde yaygın olarak kullanılmaktadır. (3-6)

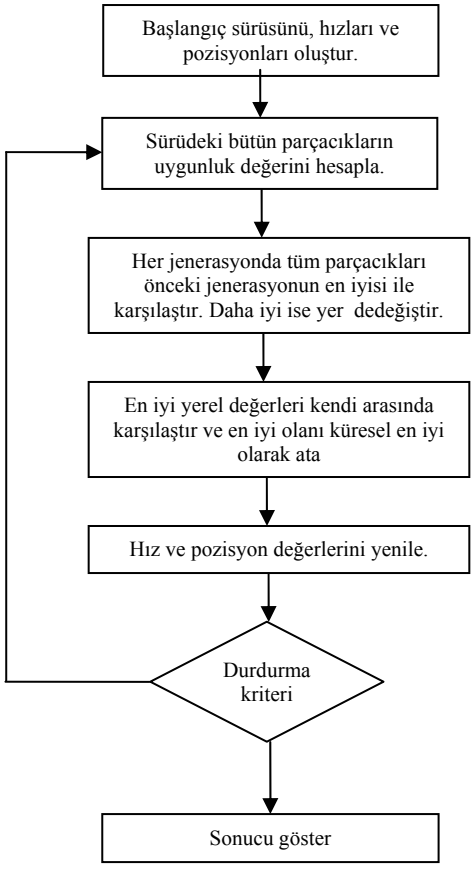
Genetik Algoritmalar (GA), canlılardaki genetik kalıtımı örnek alarak geliştirilen popülasyon tabanlı bir evrim algoritmasıdır. Her bir jenerasyonda en iyiye ulaşmayı amaçlar. Diferansiyel Evrim Algoritması(DEA), genetik çaprazlama işlemi, bireysel farklardan yola çıkılarak geliştirilmiştir. Bu bilinçli çaprazlama en iyiye ulaşmada oldukça etkilidir. Parçacık Sürü Optimizasyonu algoritması temelde sürüdeki bireylerin birbirini geliştirmesine dayanan yeni bir algoritmadır.

Yakınsama hızı, sezgisel optimizasyon tekniklerinin performansını en iyi ölçen kriterlerden biridir (7). Bu çalışmada, matematiksel test fonksiyonları kullanılarak Parçacık Sürü Optimizasyonu algoritması ile Genetik ve Diferansiyel Evrim algoritmalarının performans ve yakınsama hızları karşılaştırılmıştır. Makalenin organizasyonu şu şekildedir. Bölüm 2,3 ve 4'de algoritmalar hakkında genel bilgiler verilmektedir. Bölüm 5 de test fonksiyonları ve Bölüm 6 da benzetim sonuçları ele alınmaktadır. Bölüm 7 de genel sonuçlar verilmektedir.

Makale 25.03.2008 tarihinde gelmiş,06.10.2008 tarihinde yayınlanmak üzere kabul edilmiştir.
M. Y. ÖZSAĞLAM, Selçuk Üniversitesi Bozkır MYO Bilgisayar Bölümü Bozkır/KONYA
e-posta : myasinozsaglam@gmail.com
M. CUNKAŞ, Selçuk Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi 42003 KONYA
e-posta : mcunkas@selcuk.edu.tr
Digital Object Identifier 10.2339/2008.11.4. 299-305

2. PARÇACIK SÜRÜ OPTİMİZASYONU

Parçacık Sürü Optimizasyonu (PSO), sürü halinde hareket eden balıklar ve böceklerden esinlenerek Kenedy ve Eberhart (1995) tarafından geliştirilmiş bir optimizasyon yöntemidir (8). Temel olarak sürü zekâsına dayanan bir algoritmadır. Sürü halinde hareket eden hayvanların yiyecek ve güvenlik gibi durumlarda, çoğu zaman rasgele sergiledikleri hareketlerin, amaçlarına daha kolay ulaşmalarını sağladığı görülmüştür. PSO bireyler arasındaki sosyal bilgi paylaşımını esas alır. Arama işlemi genetik algoritmalarda olduğu jenerasyon sayısınca yapılır. Her bireye parçacık denir ve parçacıklardan oluşan popülasyona da sürü (swarm) denir. Her bir parçacık kendi pozisyonunu, bir önceki tecrübesinden yararlanarak sürüdeki en iyi pozisyona doğru ayarlar. PSO, temel olarak sürüde bulunan bireylerin pozisyonunun, sürünün en iyi pozisyona sahip olan bireyine yaklaştırılmasına dayanır. Bu yaklaşma hızı rasgele gelişen durumdur ve çoğu zaman sürü içinde bulunan bireyler yeni hareketlerinde bir önceki konumdan daha iyi konuma gelirler ve bu süreç hedefe ulaşıncaya kadar devam eder(8). PSO, Sipariş miktarı belirleme, çizelgeleme problemleri, güç ve voltaj kontrolü, motor parametrelerini belirleme, tedarik seçimi ve sıralama problemleri gibi bir çok optimizasyon problemlerinde başarı ile kullanılmıştır (9). Şekil-1 de PSO'nun akış diyagramı görülmektedir.



Şekil 1 Parçacık Sürü Optimizasyonu akış diyagramı

Algoritma temel olarak aşağıdaki basamaklardan oluşur;

i. Rasgele üretilen başlangıç pozisyonları ve hızları ile başlangıç sürüsü oluşturulur.

ii. Sürü içerisindeki tüm parçacıkların uygunluk değerleri hesaplanır.

iii. Her bir parçacık için mevcut jenerasyondan yerel en iyi (pbest) bulunur. Sürü içerisinde en iyilerin sayısı parçacık sayısı kadardır.

iv. Mevcut jenerasyondaki yerel eniyiler içerisinden küresel en iyi (gbest) seçilir.

v. Pozisyon ve hızlar aşağıdaki gibi yenilenir.

$$V_{id} = W * V_{id} + c_1 * rand_1 * (P_{id} - X_{id}) + c_2 * rand_2 * (P_{gd} - X_{id})$$
$$X_{id} = X_{id} + V_{id}$$

Burada X_{id} pozisyon ve V_{id} hız değerlerini verirken, $rand_1$ ve $rand_2$ değerleri rasgele üretilmiş sayılardır. W atalet ağırlık değeri ve C_1, C_2 ölçeklendirme faktörleridir.

vi. Durdurma kriteri sağlanıncaya kadar 2,3,4,5 adımları tekrar edilir.

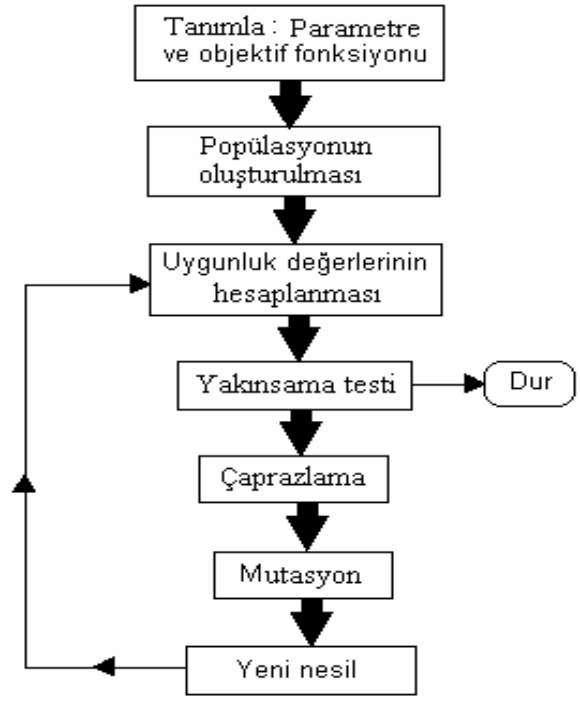
3 GENETİK ALGORİTMALAR

Doğada gözlemlenen evrimsel sürece benzer bir şekilde çalışan bir optimizasyon yöntemidir. Karmaşık, çok boyutlu arama uzayında en iyinin hayatta kalması ilkesine göre küresel çözümü arar. GA'lar , uzun çalışmaların neticesinde ilk defa John Holland tarafından optimizasyon problemlerine uygulandı ve öğrencisi David Goldberg tezinde; gaz boru hattının optimizasyonunu içeren bir problemin çözümünü GA ile gerçekleştirdi (10). Son zamanlarda GA'lar ile ilgili çalışmalar belirgin bir şekilde artmış ve bilgisayar teknolojisindeki ilerlemelere bağlı olarak çoğu uygulama alanlarında etkili bir şekilde kullanılmaya başlanmıştır.

GA parametreleri, biyolojideki genleri temsil ederken, parametrelerin toplu kümesi de kromozomu oluşturmaktadır. GA'ların her bir ferdi, yani her bir olası çözüm, kromozom şeklinde temsil edilir. Bu aday çözümler kümesi de popülâsyon olarak adlandırılır. Popülâsyonun uygunluğu, belirli kurallar dâhilinde maksimize veya minimize edilir. Her yeni nesil, rasgele bilgi değişimi ile oluşturulan diziler içinde hayatta kalanların birleştirilmesi ile elde edilmektedir.

Genetik algoritmalarda Çaprazlama ve Mutasyon olarak bilinen iki adet temel genetik işlemci vardır. Çaprazlama için popülâsyondan iki adet birey seçilir. Bu bireylerde çaprazlanacak nokta belirlenir ve bu noktadan itibaren bireylerin elemanları karşılıklı olarak yer değiştirilir. Böylece iki adet yeni birey elde edilir. Mutasyon işlemcisi ile bireylerin genleri değiştirilir. Bu değişim popülasyonun genel olarak %1-%5 ini kapsamaktadır. Mutasyon popülasyonda çeşitliliğe neden olur ve problem sonucunun yerel çözümlere takılmasını önler.

Şekil 2 de Genetik algoritmaların akış diyagramı görülmektedir.



Şekil 2. Genetik Algoritma akış diyagramı

Genetik Algoritmaların çalışma adımları aşağıda verilmektedir.

i. Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur. (başlangıç populasyonu). Populasyonda bulunacak birey sayısı için bir standart yoktur, problemin türüne göre bu sayı değişebilir. Başlangıç populasyonu rasgele oluşturulur ve problemin türüne göre çeşitli yöntemlerle kodlanır (ikilik, gerçek veya permütasyon gibi).

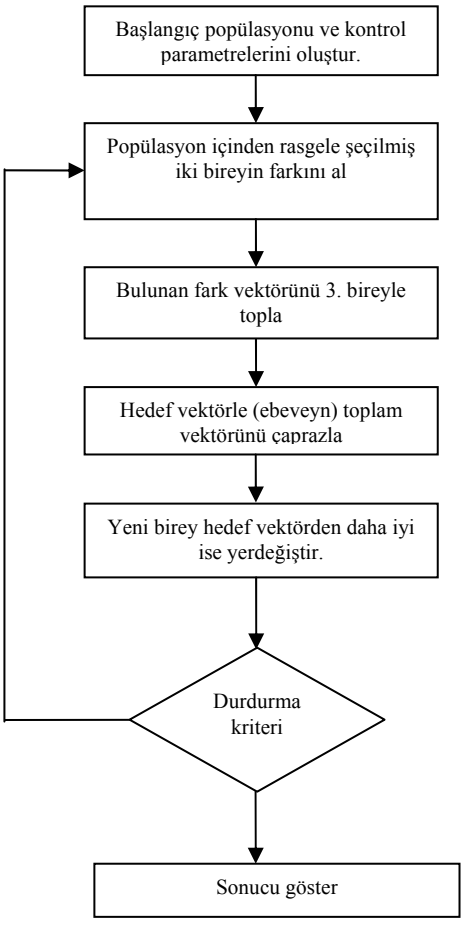
ii. Her kromozomun (bireyin) uygunluk değeri amaç fonksiyonu kullanılarak hesaplanır. Genetik algoritmanın başarısı çoğu zaman bu fonksiyonun iyi tespit edilmesine bağlıdır. Bu adımda en iyi birey seçilerek yerel çözüm olarak elde tutulur.

iii. Durdurma kriteri olarak daha önceden belirlenmiş jenerasyon sayısı veya herhangi bir ideal küresel çözüm verilebilir. Durdurma kriteri test edilir. Eğer kriter sağlanıyorsa algoritma durdurulur ve son olarak elde tutulan yerel çözüm, küresel çözüm olarak alınır. Kriter sağlanmıyorsa sonraki adıma geçilerek yeni çözümler aramaya devam edilir.

iv. Popülâsyonda çaprazlamaya tabii tutulacak bireyler oluşturulurken uygunluk değerleri temel alınır ve çeşitli seçim yöntemleri kullanılır. Seçilen bireyler çaprazlama ve mutasyona tabii tutularak yeni bireyler oluşturulur ve Adım 2 ye gidilir. Ancak popülâsyonun boyutu sabit büyüklükte kalmalıdır.

4. DİFERANSİYEL EVRİM ALGORİTMASI

Optimizasyon problemlerinde yaygın olarak kullanılan Diferansiyel Evrim algoritması, popülasyon temelli güçlü bir gelişim algoritmasıdır [11,12]. Bu tür algoritmalar genel amaçlı sayısal optimizasyon algoritmaları olarak da nitelendirilmektedir. Diferansiyel evrim algoritmalarında, genetik algoritmalardan farklı olarak, gelişmiş ve etkili bir mutasyon işlemi kullanılmaktadır. Amaç vektör çiftlerinin farkına dayalı olan mutasyon işlemi, amaç vektörlerinin kendi dağılımları tarafından belirlenir. Ayrıca bir ebeveyn vektörden, bir deneme vektörü üretmek için mutasyon ve çaprazlama birlikte kullanılır (2). Şekil 3 de diferansiyel evrim algoritmasının akış diyagramı verilmiştir.



Şekil 3. Diferansiyel Evrim Algoritması akış diyagramı

Diferansiyel Evrim Algoritmasının çalışma adımları aşağıdaki gibidir.

i. Başlangıç popülasyonu ve kontrol parametreleri aşağıdaki şartları sağlayacak şekilde tanımlanır. (NP:populasyon sayısı, CR: Kombinasyon oranı, F:ölçeklendirme faktörü)

NP>=4 , F ϵ (0,1+) , CR ϵ [0,1].

ii. Başlangıç popülasyonunun aşağıdaki gibi oluşturulur. Burada D popülasyon üyelerinin boyutunu ifade ederken, G ise "0" olarak atandığı için bireyin ilk parametresini ifade eder.

$$\forall i \le NP \wedge \forall j \le D : x_{i,j,G=0} = X_j^{lo} + rand_j[0,1].\left(x_j^{hi} - x_j^{lo}\right)$$
$$i = (1,2.....,\ NP),\ j = (1,2.....,\ D),\ G = 0,\ rand_j[0,1] \varepsilon [0,1]$$

iii. Mutasyon ve Çaprazlama aşağıda belirtildiği gibi yapılır.

$$r_1, r_2, r_3 \varepsilon \{1,2,....,\ NP\},\ r_1 \neq r_2 \neq r_3 \neq i (rasgele\ -seçim)$$
$$j_{rand} \varepsilon \{1,2,.....,\ D\},\ (rasgele\ -seçim)$$
$$\forall j \le D, u_{i,j,g+1} = \begin{cases} x_{j,r3,G} + F.(x_{j,r1,G} - x_{j,r2,G}) \\ eger\ (rand_j[0,1] \langle CR \vee j = j_{rand}) \\ X_{j,i,G}\ (diger\ -durumlarda) \end{cases}$$

iv. Yeni bireyin uygunluk değeri hedef vektör olarak seçilen ebeveynin uygunluk değerinden daha iyi ise yeni birey ebeveyn ile yer değiştirilir.

$$\bar{x}_{i,G+1} = \begin{cases} u_{i,G+1} - eger\ -f(u_{i,G+1}) \le f\left(x_{i,G}\right) \\ x_{i,G} - diger \end{cases}$$

v. Durdurma kriteri sağlanıyorsa sonuç gösterilir değilse 3 ve 4. adımlar tekrar edilir

5. TEST FONKSİYONLARI

Söz konusu Algoritmalar performanslarının ölçülmesi amacıyla literatürde kullanılan sekiz farklı test fonksiyonu seçilmiştir. De Jong tarafından önerilen F_1, F_2, F_3 ve F_4 test fonksiyonları bir çok araştırmada, optimizasyon algoritmalarının performanlarının değerlendirilmesinde kullanılmıştır (13). F_5, F_6, F_7 ve F_8 test fonksiyonları ise Toksarı'nın (14) çalışmasından alınmıştır. Tablo-1'de verilen test fonksiyonları hakkında aşağıda kısa bilgiler verilmektedir.

F_1 (Küresel fonksiyon): Bu fonksiyonun küresel minimumu '0' ve yapılacak çözüm sonunda x_1, x_2 ve x_3 değişkenlerinin alacağı değerler de '0' dır. Fonksiyon küresel olmasından dolayı söz konusu algoritmaların yerel minimuma takılma olasılıkları oldukça düşüktür.

F_2 (Muz fonksiyonu): Bu fonksiyonun küresel minumumu '0' ve yapılacak çözüm sonucunda x_1 ve x_2 değişkenlerinin alacağı değerler '1' olacaktır.

Tablo-1 Test Fonksiyonları

Fonksiyon No	Fonksiyon	Sınırlar
F_1	$\sum_{i=1}^{3} x_i^{\,2}$	$-5.12 \le x_i \le 5.1$ $i=1..2$
F_2	$100\left(x_1^{\,2} - x_2\right)^2 + \left(1 - x_1\right)^2$	$-2.048 \le x_i \le 2.048$ $i=1,2$
F_3	$\sum_{i=1}^{5} \text{int}(x_i)$	$-5.12 \le x_i \le 5.1$ $i=1..5$
F_4	$\left[0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_1 - a_{ij})^6}\right]$	$-65.536 \le x_i \le 65.536$ $i=1,2$
		Optimum Değerler
F_5	$x_1^2 + 2x_2^2 + 0,3\cos\left(3\pi x_1\right) - 0,4\left(4\pi x_2\right) + 0,7$	Küresel Min.=0 $x_1 = 0,\ x_2 = 0$
F_6	$x^2 \quad if\ x \le 1$; $(x-3)^2 - 3 \quad if\ x > 1$	Küresel Min.=-3 $x = 3$
F_7	$\frac{(x_1-3)^8}{1+(x_1-3)^8} + \frac{(x_2-3)^4}{1+(x_2-3)^4}$	Küresel Min.=0 $x_1 = 3,\ x_2 = 3$
F_8	$\frac{x_1}{1 + \lvert x_2 \rvert}$	Küresel Min.=-10 $x_1 = -10,\ x_2 = 0$

Fonksiyonun optimum noktası dar, parabolik bir alanda olduğu için algoritmaların optimuma ulaşması daha zor olmaktadır.

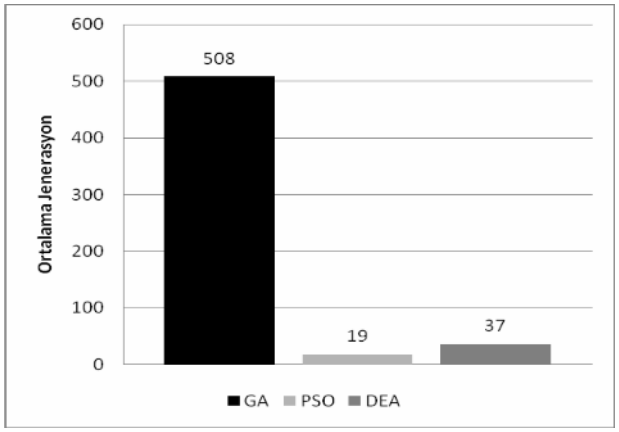
F_3 (Basamak fonksiyonu) : Tüm değişkenlerin alacağı değerler '-6' olarak kabul edilirse, bu fonksiyonun küresel minumumu '-30' olacaktır. Fonksiyonun yapısı gereği bir çok optimizasyon algoritması yerel minumuma takılmaktadır.

F_4 (Tilki delikleri fonksiyonu): Bu fonksiyonun küresel minimumu '1' ve yapılacak çözüm sonucunda x_1 ve x_2 değişkenlerinin alacağı değerler '-32' olmaktadır. Fonksiyonun bir çok yerel minimumu olduğu için çoğu standart optimizasyon algoritmaları yerel minimuma takılmaktadır.

Diğerleri nispeten biraz zor problemler olup F_5 test fonksiyonu Kwon ve ark (15) tarafından, F_6, F_7 ve F_8 test fonksiyonları ise Hamzaçebi ve Kutay (16) tarafından yaptıkları çalışmalarda geliştirdikleri algoritmaların performansını ölçmek amacıyla kullanılmışlardır. Tablo 1'de küresel minimum ve parametrelerinin alacağı değerler verilmiştir.

6. BENZETİM SONUÇLARI

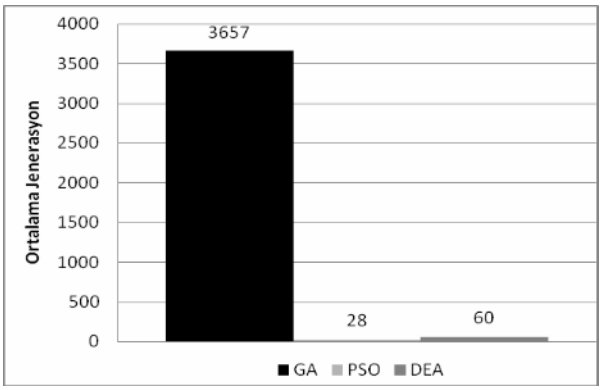
Gerçekleştirilen yazılımda, söz konusu algoritmaların bir jenerasyonu tamamlama süreleri birbirine yakın olduğu için, kaçıncı jenerasyonda optimuma ulaştıkları esas alınarak karşılaştırmalar yapılmıştır. En iyi sonuca kaçıncı jenerasyonda ulaştığını belirmek için, program her bir test fonksiyonu için 50 kez koşturulmuş ve ortalama değerler hesaplanmıştır. Popülasyon sayısı arttıkça optimum sonucu elde etme olasılığı artıyor fakat jenerasyon süresi uzamaktadır. Testlerde tüm algoritmalar için popülasyon sayısı sabit seçilmiştir. Parametreler için kullanılan değerler Tablo 2 de verilmiştir. Tabloda gösterilen parametre değerleri literatürdeki çalışmalar göz önüne alınarak seçilmiştir. Fakat optimize edilen problemin karakteristiğine göre parametre değerleri farklılık gösterebileceğinden, başka problemlerde benzer parametre değerleri kullanılacak diye bir gereklilik söz konusu değildir. İlk dört test fonksiyonunun yakınsama grafikleri gösterilmiş ayrıca bütün test fonksiyonlarının ortalama yakınsama jenerasyon sayıları Tablo 3 de verilmiştir.



Şekil 4. F_1 Fonksiyonu için ortalama jenerasyon sayısı

Şekil 4'de F_1 fonksiyonu için yakınsama hızları görülmektedir. GA hariç PSO ve DEA yakınsama hızları birebirine yakındır. F_1 fonksiyonu için ortalama jenerasyon değerleri, PSO 19, DEA 37 ve GA 508 olarak elde edildi.

Şekil 5'de F_2 fonksiyonu için yakınsama hızları görülmektedir. Küresel minimumun bulunmasında GA'nın, PSO ve DEA 'ya göre kötü bir performans sergilediği görülmektedir. F_2 fonksiyonu için ortalama jenerasyon değerleri, PSO 28, DEA 60 ve GA 3657 olarak elde edildi.

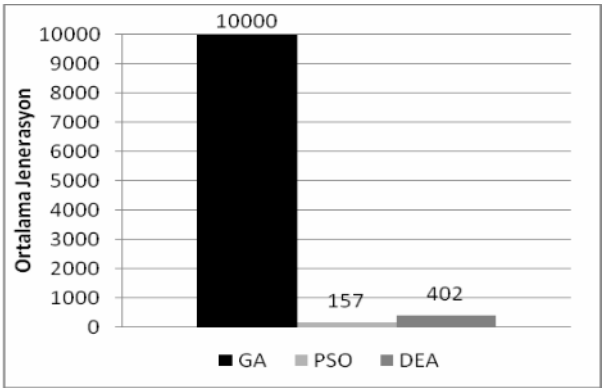


Şekil 5. F_2 Fonksiyonu için ortalama jenerasyon sayısı

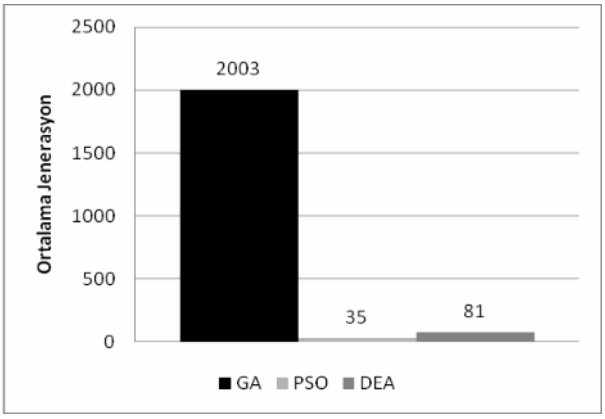
Tablo 2 Parametreler için kullanılan değerler

PSO	GA	DEA
Populasyon Sayısı:100	Populasyon Sayısı:100	Populasyon Sayısı:100
Atalet ağırlık değeri(W)=0,99	Çaprazlama oranı=0,8	Kombinasyon oranı(CR)=0,8
Ölçeklendirme faktörleri (C_1 ve C_2)=1,99	Mutasyon oranı=0,02	Ölçeklendirme faktörü (F)=0,8

F_3 fonksiyonu için yakınsama hızları Şekil 6 da verilmiştir. Her üç algoritmada, F_3 fonksiyonunun optimum çözümünü diğer test fonksiyonlarına göre daha çok jenerasyon gerçekleştirerek bulabildiler. Özellikle GA 10000 jenerasyon sayısını geçmesine rağmen sonuca ulaşamadı. Çözüm için ortalama jenerasyon sayıları, PSO 157, DEA 402 ve GA 10000+ dir. Burada GA 10000 jenerasyon sayısını geçtiği için 10000+ ile gösterildi.



Şekil 6. F_3 Fonksiyonu için ortalama jenerasyon sayısı



Şekil 7. F_4 Fonksiyonu için ortalama jenerasyon sayısı

7. SONUÇLAR

Kuş ve balık sürülerinin iki boyutlu hareketlerinden esinlenerek geliştirilen Parçaçık Sürü Optimizasyonu, daha az parametre içermesi ve program yapısının daha kolay olması gibi avantajlara sahiptir. Bu çalışmada, Parçacık Sürü optimizasyonu (PSO), Differansiyel Evrim (DEA) ve Genetik Algoritmaların (GA) performansları literatürden seçilen test fonksiyonları kullanılarak karşılaştırıldı.

Tablo 3. Herbir algoritma için ortalama yakınsama jenerasyon sayıları

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
PSO	19	28	157	35	854	134	1042	1854
DEA	37	60	402	81	912	178	1159	2432
GA	508	3657	10000+	2003	4652	816	5634	9122

F_4 fonksiyonu için yakınsama hızları Şekil 7 de görülmektedir. GA hariç PSO ve DEA kısa bir zamanda optimum çözüme ulaştılar. GA'nın bu sonuca oldukça uzun jenerasyon sayısında ulaştığı görüldü. F_4 fonksiyonu için ortalama jenerasyon değerleri, PSO 35, DEA 81 ve GA 2003 olarak elde edildi.

Tüm Algoritmaların ortalama yakınsama jenerasyon sayıları Tablo 3 de verilmiştir. Tablodaki değerlerden anlaşıldığı gibi, genel olarak PSO ve DEA, yakınsama hızları açısından GA'dan oldukça iyidir. Nispeten zor diyebileceğimiz F_5, F_7 ve F_8 test problemlerinde PSO nun yakınsama jenerasyon sayısının arttığı gözlemlenmektedir. Özellikle F_2, F_3, F_4, F_5 F_7 ve F_8 fonksiyonlarında GA daha başarısızdır. PSO ile DEA nın performansları birebirine yakın olmakla beraber en iyi yakınsama hızı PSO da ölçülmüştür.

Elde edilen sonuçlardan PSO nun yakınsama hızı, bu çalışmada sunulan GA ve DEA göre daha iyi olduğu görüldü. Sonuç olarak, optimizasyon problemlerinin çözümünde alternatif bir yöntem olarak PSO algoritmasının kullanılması avantaj sağlayacağı söylenebilir.

8. KAYNAKLAR

1. Çunkaş M., Elektrik Makinalarının Genetik Algoritmayla Optimizasyonu, Doktora Tezi, Selçuk Ünv. Fen Bilimleri Enst. 2004, Konya
2. Karaboğa D., Yapay zeka optimizasyon algoritmaları, Atlas Yayınları, 2004.
3. Rahimpour E., Rashtchi V., Pesaran M., "Parameter identification of deep-bar induction motors using genetic algorithm", Electrical Engineering, 89(7): 547-552, 2007.

4. Çunkaş M., Akkaya R., Bilgin O.,"Cost optimization of submersible motors using a genetic algorithm and a finite element method", Int. J of Adv. Manuf. Technol., 33: 223-232, 2007.

5. Tušar T., Korošec P., Papa G.,Filipič B., Šilc J. "A comparative study of stochastic optimization methods in electric motor design", Applied Intelligence, 27(2):101-111, 2007.

6. Wrobel R., Mellor P. H., "Particle Swarm Optimization for the design of Brushless Permanent Magnet Machines", IEEE Industry Applications Conference, 41st IAS Annual Meeting, 2006.

7. Karaboğa D., ÖKDEM S., "A Simple and Global Optimization Algorithm for Engineering Problems: Differential EvolutionAlgorithm", Turk J Elec Engin, 12(1), 53-60, 2004.

8. Kennedy, J.; Eberhart, R. C., "Particle Swarm Optimization", Proc. of the IEEE Int. Conference on Neural Networks, 4, 1942-1948, 1995.

9. Wilke D. N., Analysis of the particle swarm optimization algorithm, Master Thesis, 2005, Mechanical and Aeronautical Engineering, University of Pretoria.

10. Goldberg D.E., Genetic Algorithms in Search, Optimization and Machine Learning, 1999, Addison Wesley.

11. Storn R., "Diferential Evolution, A Simple and Efficient Heuristic Strategy for Global Optimization over Continuous Spaces", Journal of Global Optimization, 11: 341-359, 1997.

12. Kenneth V.P., An introduction to Differential Evolution, in New Ideas in Optimization, McGraw-Hill publishing Company, pp.79-108., 1999.

13. De Jong K.A., An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Phd Thesis, University of Michigan, Dissertation Abstracts International 36(10), 5140B. (University Micro_lms No. 76-9381), 1975.

14. Toksarı M.D., "A heuristic approach to find the global optimum of function", Journal of Computational and Applied Mathematics 209:160–166, 2007.

15. Kwon Y.D., Kwon S.B., Kim J., "Convergence enhanced genetic algorithm with successive zooming method for solving continuous optimization problems", Comput. Struct. 81:1715–1725, 2003.

16. Hamzacebi C., Kutay F., "A heuristic approach for finding the global minimum: adaptive random search technique", Appl. Math. Comput. 173(2): 1323–1333, 2006