

**TÜBİTAK 2209-A ÜNİVERSİTE ÖĞRENCİLERİ**

**YURT İÇİ ARAŞTIRMA PROJELERİ**

**DESTEK PROGRAMI**

**META-SEZGİSEL OPTİMİZASYON ALGORİTMALARININ**

**DSK YÖNTEMİYLE GELİŞTİRİLMESİ**

**VE**

**KISITLI MÜHENDİSLİK TASARIM PROBLEMLERİNİN OPTİMİSAZYONU**

**KARADENİZ TEKNİK ÜNİVERSİTESİ**

**YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ**

**TEMATİK ALANI**

**YAZILIM MÜHENDİSLİĞİ**

**BİLİŞİM TEKNOLOJİLERİ MÜHENDİSLİĞİ**

**PROJE YÜRÜTÜCÜSÜ**

**MEHMET KATI**

**PROJE DANIŞMANI**

**DOÇ. DR. HAMDİ TOLGA KAHRAMAN**

**İÇİNDEKİLER**

|  |  |
| --- | --- |
| 1. Özet | 3 |
| 2. Giriş | **3** |
| 3. Problem | **5** |
| 3.1. Problemin Tanımı | **5** |
| 3.2. Çalışmanın Amacı | **7** |
| 3.3. Araştırma Sorusu ve/veya Hipotez | **7** |
| 4. Projede Kullanılan Yöntemler ve Metotlar | **7** |
| 4.1. DSK Yöntemi | **7** |
| 4.1.1. Dağılım Prosedürü | **8** |
| 4.1.2. Kontrol Prosedürü | **8** |
| 4.1.3. Seçim Prosedürü | **8** |
| 4.2. Tatbik Edilecek Teknikler | **8** |
| 4.2.1. Levy Uçuşu | **8** |
| 4.2.2. Kaos Haritaları | **9** |
| 4.2.2.1. Chebyshey Haritası | **9** |
| 4.2.2.2. Singer Haritası | **10** |
| 4.2.2.3. Tent Haritası | **10** |
| 4.2.3 Rulet Tekerleği | **11** |
| 4.2.4 Gauss Dağılımı | **12** |
| 4.3. Tekniklerin Tatbik Edilme Yöntemi | **12** |
| 4.4. Deneysel Çalışmalar | **14** |
| 5. Proje İş-Zaman Planı | **20** |
| 6. Sonuç | **20** |
| 7. Kaynaklar | **20** |

# **Özet**

Klasik optimizasyon teknikleri, problem boyutunun yani tasarım değişkenlerinin sayısının çok olması, problemin lineer olmaması, arama uzayının büyük olması durumlarında kabul edilebilir bir çözüm bulmakta yetersiz ve etkisiz kalmaktadırlar. Karmaşıklık düzeyi yüksek optimizasyon problemleri için en uygun çözümü bulmak zor bir görevdir. Günümüzde karmaşıklık düzeyi yüksek arama uzaylarına sahip optimizasyon problemlerinin çözümlenmesinde sıklıkla meta-sezgisel arama algoritmaları kullanılmaktadır. Meta sezgisel arama (MSA) algoritmaları doğadan esinlenilerek geliştirilmiş yöntemlerdir. [1].

MSA algoritmalarının tatbik edildiği alanlar optimizasyon problemleri ile de sınırlı değildir. MSA algoritmaları başta tahmin, sınıflandırma ve kümeleme problemlerinin modellenmesinin yanı sıra melez yapay zekâ algoritmalarının tasarımında ve geliştirilmesinde de yaygın ve başarılı bir şekilde uygulanmaktadırlar. Özellikle yapay sinir ağlarının optimizasyonu [2], k-en yakın komşu sınıflandırıcının [3], karar ağaçlarının [4-5] ve bulanık mantık-temelli algoritmaların [6] melezleştirilmesi ve son dönemlerin popüler araştırma konularından derin öğrenme [7], büyük veri uygulamalarında [8], Endüstri 4.0 [9-10] gibi modern otomasyon sistemlerinin ve uygulamalarının geliştirilmesinde meta-sezgisel optimizasyon tekniklerinden faydalanılmaktadır.

MSA algoritmalarının performansları, komşuluk araması ve çeşitlilik görevlerini yerine getirmelerindeki başarılarına bağlıdır. Özellikle karmaşıklık düzeyi yüksek problemlerin çözümlenmesinde MSA algoritmaları iki zorlukla karşılaşmaktadırlar. Bunlar, çoklu modal problemlerin (multi-modal) arama uzaylarındaki çok sayıda yer alan yerel minimum tuzakları ve küresel çözüme yeterince yakınsayamama problemleridir. Yerel çözüm tuzaklarına yakalanmanın temel sebebi, algoritmaların çeşitlilik görevlerini etkili bir şekilde yerine getirememeleridir. Yakınsama konusundaki problemler ise algoritmaların komşuluk aramasını hassas bir şekilde gerçekleştirememelerinden kaynaklanmaktadır. Mevcut yöntemlerden daha güçlü arama performansı sergileyen MSA algoritmaları geliştirmek için 1980’li yıllardan bu yana yüzlerce çalışma yapılmıştır. Bu çalışmaların bir kısmı doğadan esinlenilerek geliştirilen yeni MSA algoritmaları iken, büyük bir bölümü ise mevcut MSA algoritmalarının çeşitli yöntemlerle yeniden tasarlanarak (modifiye edilerek) performanslarının iyileştirilmesi esasına dayanmaktadır [11-25]. MSA algoritmalarının yeniden tasarlanmalarında ve melezleştirilmelerinde ise çoğunlukla doğadan esinlenilerek geliştirilmiş çeşitli yöntemler kullanılmaktadır. Bu yöntemleri üç başlık altında toplamak mümkündür. Bunlar, “dağılım”, “seçim” ve “kontrol” (DSK) olarak adlandırılan yöntemlerdir.

Günümüzde üzerinde yoğun bir şekilde çalışılan optimizasyon uygulamalarının başında kısıtlı mühendislik tasarım problemleri gelmektedir. Mevcut MSA algoritmalarının baz modelleri ile bu problemlerin birçoğu çözümlenememekte ya da kabul edilebilir bir çözüme ulaşılamamaktadır. Araştırmacılar bu durumda MSA algoritmalarını kendi problemlerine yönelik olarak özelleştirmek suretiyle modifiye etmekte ve daha iyi çözümleri keşfetmeye çalışmaktadırlar. Algoritmaların probleme özgü olarak tasarlanması ise uzmanlık isteyen ve zorluklarla dolu bir süreçtir. Bu süreçte MSA algoritmaları çok çeşitli tekniklerin tatbik edilebildiği DSK yöntemleri ile esnek ve etkili bir şekilde tasarlanabilmeli, test edilebilmeli ve doğrulanabilmelidirler. Böylesi bir çalışma ise ancak, yazılım tasarım prensiplerine bağlı olarak modüler yapıda geliştirilmiş, DSK yöntemlerini, çeşitli karşılaştırma ve mühendislik test problemlerini, güçlü ve çok sayıda alternatif MSA algoritmalarını içeren ve algoritmaların performanslarını karşılaştırmak için istatistiksel analiz yöntemlerinin tatbik edilebildiği bir platformda gerçekleştirilebilir.

Bu proje çalışmasının amacı, günümüzde üzerinde yoğun olarak çalışılan mühendislik tasarım problemlerini mevcut MSA algoritmalarından daha başarılı bir şekilde çözümleyen melez MSA algoritmaları geliştirmektir. Bu amaca yönelik olarak literatürde yer alan ve MSA algoritmalarının performanslarını iyileştirmek amacıyla araştırmalar yürütülecektir. Araştırma sürecinde literatürdeki 15 MSA algoritması, çeşitli DSK yöntemleri, 30 adet klasik karşılaştırma problemi (CEC 2017 problem havuzu [71]), 5 adet mühendislik tasarım problemi (MSA makalelerinde 2-4 arası probleme yer verilmektedir) [26-29] ve Wilcoxon ve Friedman test ve analiz yöntemleri kullanılacaktır. Bu süreçte kullanılacak olan MSA algoritmalarının kaynak kodları MATLAB File Exhange platformundan elde edilmiştir. Çalışma sürecinin ilk adımında, literatürdeki en güncel ve en yaygın kullanılan MSA algoritmaları arasından 15’i (on beşi) seçilecektir. Hâlihazırda bu algoritmaların belirlenmesi için ön çalışma yapılarak 25 MSA algoritmasının makalelerine ve MATLAB kodlarına erişilmiştir[28-52]). Bu 25 algoritma arasından 15’i seçilecektir. Bu algoritmalar arasından ise mühendislik tasarım problemlerinde en iyi performansa sahip olan ilk 3’ü belirlenecektir. İkinci adımda, bu üç (3) algoritmaya çeşitli DSK yöntemleri tatbik edilerek algoritmaların arama performansları iyileştirilmeye çalışılacaktır. DSK yöntemleri ile güçlendirilen algoritmaların mühendislik tasarım problemlerindeki performansları araştırılacaktır. Bu süreçte algoritmaların komşuluk araması ve çeşitlilik görevlerini dengeli ve daha etkili bir şekilde yerine getirebilmeleri için DSK yöntemlerinden faydalanılacaktır. Üçüncü adımda, modifiye edilmiş MSA algoritmalarının CEC 2017 problem havuzu [71] ve mühendislik tasarım problemleri üzerindeki performansları araştırılacaktır. Bu süreçte algoritmaların baz modelleri ile yeniden tasarlanmış modelleri arasında performans karşılaştırmaları yapılacaktır. Dolayısıyla toplamda altı (6) rakip yöntem arasından en başarılı olanı belirlenmiş olacaktır. Son olarak algoritmaların performansları istatistiksel test ve analiz yöntemleri (wilcoxon ve friedman testleri) ile analiz edilerek kısıtlı mühendislik problemleri için en güçlü MSA algoritması literatüre kazandırılacaktır.

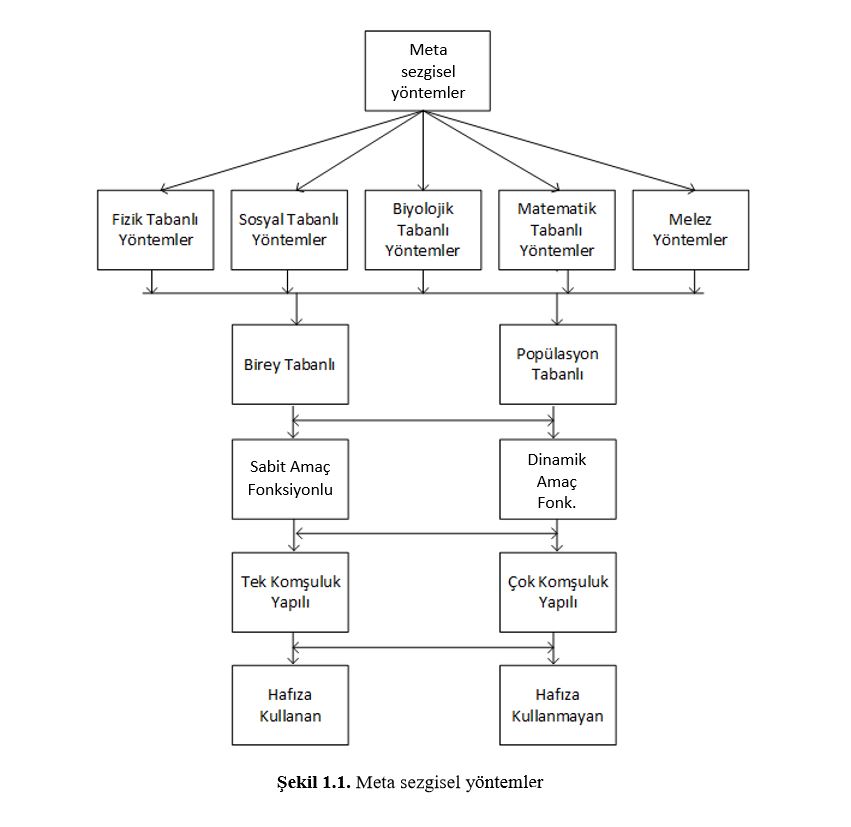
1. **Giriş**

Optimizasyon çeşitli problemleri çözmek amacıyla karar değişkenleri ve kısıtlar altında,en uygun çözümü arama yöntemlerine denir [53]. Herhangi bir probleme optimizasyon işlemi uygulanarak performans ve minimum maliyeti sağlayan çözüm elde edilmeye çalışılır [1].

Değişen dünya ve gelişen teknoloji ortamında ortaya çıkan problemler gün geçtikçe zorlaşmaktadır [1]. Çoğunda problemin çözüm uzayı tüm çözümleri değerlendirilmeyeceği kadar büyük olur [53]. Klasik optimizasyon metotları ile gerçek dünya problemlerini çözmek çok zaman almakta ve etkili bir şekilde çözülememektedir [1].

Optimizasyon üzerine çalışan bazı bilim insanları, yüksek performanslı, dinamik ve esnek yöntemler geliştirmek için dikkatlerini doğaya vermişlerdir. Bunun sebebi, doğada var olan ve kendiliğinden muazzam bir işleyişe sahip olan sistemlerin, optimizasyon problemlerini çok daha efektif bir şekilde çözebilecekleri düşüncesidir. Çünkü doğa, zaten var olan kompleks ve zor olan optimizasyon problemini, yine doğada var olan çoğu zaman gizemini sürdüren yöntemlerle çözmektedir. Doğada var olan sistemleri ve olayları temel alarak oluşturulan optimizasyon teknikleri sezgisel yöntemler olarak adlandırılır. Sezgisel yaklaşım en iyiyi bulma garantisine sahip değildir ve bu nedenle genel olarak optimumdan daha kötü çözümler getirir. Bununla birlikte, sezgisel algoritmalar genellikle 'makul' bir sürede iyi çözümler bulurlar. Problemlerin bu denli efektif çözülmesindeki ana sebeplerden birisi rasgele davranma hareketidir. Rasgele hareketlilik, problemin çözümünde tek bir yoldan gidilmemesini sağlar ve arama uzayında taranmayan alan bırakmama başarısını getirir [54].

Meta sezgisel yöntemlerin çalışmaları 1950’li yıllara dayanmaktadır. Michigan üniversitesi Prof. John Holland ve öğrencilerinin geliştirdikleri Genetik Algoritma bu çalışmalara hız kazandırmıştır [60]. Çözüm uzayında etkili bir şekilde arama yapmak için, farklı yapılardaki alt kademe sezgisel algoritmaların zekice birleştirilmesi ile oluşturulmuş iteratif problem çözme yöntemlerdir. Bu yöntemler her iterasyonda bir çözümden veya çözüm koleksiyonundan yola çıkarak yeni çözümler üretirler. Çoğu meta sezgisel yaklaşım, çözüm uzayında stokastik fakat bilinçli bir şekilde arama yapar [55]. Şekil 1.1’de sezgisel yöntemlerin sınıflandırılması gösterilmiştir [57].

****

Sezgisel algoritmalar işleyişlerinde rasgele hareketliliklerinde oluşabilecek daralma nedeniyle, çözüm uzayını aramak için oluşturulan çözüm adayları prematüre yakınsama problemine takılabilmektedir. Bu problem tüm çözüm adaylarının en iyi çözüm adayına benzemesine ve bir süre sonra arama uzayında birbirine çok yakın hatta aynı konumu temsil eden çözüm adaylarının oluşmasına neden olur. Böylece bulunan doğa-esinli sezgisel algoritmalar kesin çözüm yerine bulunabilecek en iyi çözüme yöneldiklerinden dolayı yerel minimum veya maksimum noktalarına takılır. Gerçek dünya problemlerindeki çok farklı özelliklere sahip olması ve doğa-esinli bu algoritmaların işleyişlerindeki oluşabilecek zayıf yönler yüzünden istenilen çözümlere ulaşılamamaktadır.

Sezgisel algoritmaların işleyişlerinde oluşan zayıf yönleri iyileştirmek için çok sayıda çalışma yapılmıştır. Bu bağlamda yapılan çalışmalara ilk örneklerden biri olarak Song ve arkadaşlarının 1999 yılındaki çalışması örnek verilebilir. Bu çalışmada, arı kolonisi algoritması, çok hedefli bir problem yaklaşımındaki kısıtlamaları çözmek için dağıtık hesaplama ve sezgisel bir aç gözlü yaklaşımla iyileştirilmiştir. İlerleyen yıllarda gelindiğinde, 2005 yılında Yan ve arkadaşları, sistematik iş planlama problemi için karınca kolonisini, geçilen yollardaki feromon dengesiyle oynayarak geliştirmişlerdir. 2005 yılında ise Liu ve arkadaşları, parçacık sürü optimizasyonunu, kaos optimizasyon algoritması ile hibritleyerek iyileştirmişlerdir [54]. Mirjalili ve Gandomi [56] yerçekimsel arama algoritmasında yer çekim sabiti olarak kullanılan G parametresinin doğrusal azalış yerine rasgele haraketlilik kazandırmak isen kaos haritalarını kullanarak iyileştirme yapmışlardır. Cigal [57] kaos haritaları ile balina optimizasyon algoritmasını birleştirerek algoritmada iyileşme hedeflemiştir. Haklı ve Uğuz’un 2014’te önerdikleri metotta Parçacık Sürü Optimisazyonu’nu Levy Uçuş Mekanizması ile gerçeklemişlerdir. Bu çalışmada, optimizasyon sırasında ajanların lokal minimaya takılması ve erken yakınsama problemi sebebiyle Levy Uçuş Mekanizması ile bir modifikasyon gerçeklenmiş ve başarılı sonuçlar alınmıştır [54].

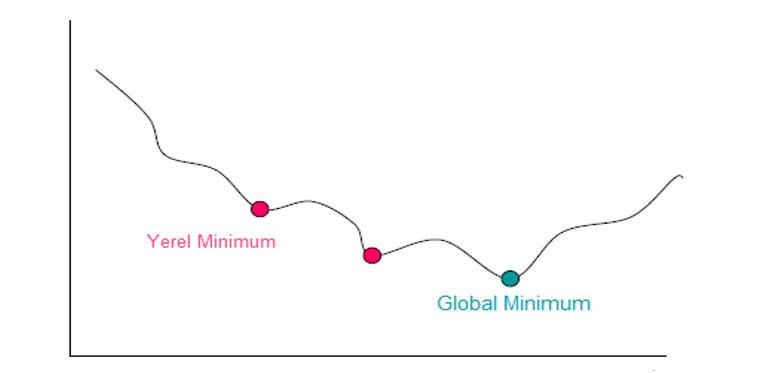
Yapılan çalışmalarında kullanılan yöntemleri DSK olarak kısaltmak ve üç başlık altında toplamak mümkündür. Bunlar sırasıyla popülasyon yaratma (D: dağılım), çözüm adayı seçimi (S: seçim) ve algoritma parametrelerinin kontrolüdür (K: kontrol). Bu çalışmalardan elde edilen tecrübeler göstermektedir ki DSK yöntemlerinden biri ya da birkaçı kullanılarak algoritmaların baz modelleri üzerinde yapılan çalışmalar algoritmaların performansları üzerinde oldukça başarılı sonuçlara yol açmaktadırlar. Bu proje çalışmasının amacı, MSA algoritmasını Kısıtlı Mühendislik Tasarım Problemleri için optimize ederek en etkili MSA yöntemlerinden biri haline getirmektir. Bu amaçla ilk olarak PSO, GSA, EFO ve ASO üzerinde başarılı olmuş yöntemler başta olmak üzere literatürdeki DSK yöntemleri araştırılacaktır. Elde edilecek bilgiler dikkate alınarak kısıtlı mühendislik tasarım problemlerindeki arama performansı DSK yöntemleri ile optimize edilmiş bir MSA algoritması geliştirilecektir.

# **Problem**

## **Problemin tanımı**

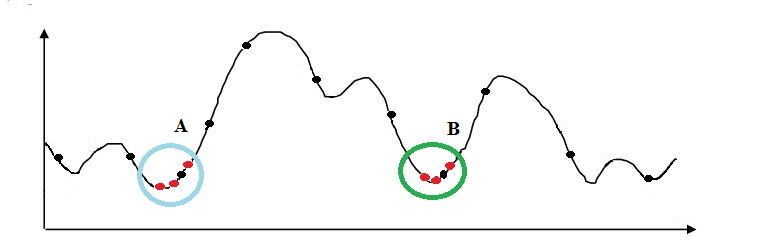
Karşılaştığımız problemler için sürekli olarak en iyi çözümleri aramaktayız. Şirketler, kurumlar ve tüm organizasyonlar karını en üst düzeye çıkarmak, maliyetleri düşürmek ve performansı en üst düzeye çıkartmak isterler. Bir şeyi maksimize etmeyi veya en aza indirmeyi planlıyor ve bunun için optimizasyon yapıyoruz. Optimizasyon belirlenen amaç fonksiyonunu, verilen kısıtlar dahilinde en uygun değeri bulmaktır. Aranan değer amaç fonksiyonunu en yüksek yapan ise maksimize problemi, en minimum yapan ise minimize problemidir. Mühendislik tasarımından finansal piyasalara, bilgisayar bilimlerinden endüstriyel uygulamalara kadar çok geniş alanda uygulanmaktadır.

En uygun çözümü bulmayı hazine avcılığına benzetebiliriz. Bulmak istediğimiz Şekil 1.2’de verilen minimize problemi için global minimum noktasıdır. Gün geçtikçe optimizasyon yapacağımız problemler zorlaşmakta ve çözüm uzayları büyümekte. Çok geniş bir alanda klasik optimizasyon yöntemleri ile hazine avcılığı yapmak zorlaşmakta ve uzun zaman almaktadır. Bu durum maliyeti artırmaktadır. Optimum çözüme veya optimum çözüme yakın sonuçları kısa zamanda bulmak için doğa-esinli sezgisel algoritmalar geliştirilmiştir.



**Şekil 1.2**

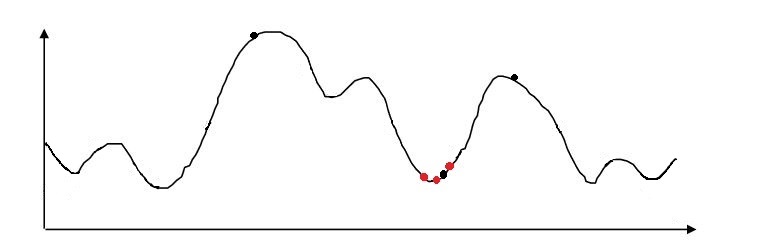
Sezgisel algoritmalarda temel iki adım bulunmaktadır. Komşuluk araması ve çeşitlilik. Çeşitlilik ile çözüm uzayında büyük atlamalar yapılarak farklı noktalar bulunur. Komşuluk araması adımında ise, çeşitlilik adımında bulunan noktaların etrafında küçük adımlar ile hassas arama yapılır.

Doğada var olan mükemmel işleyişlerin matematiksel dönüşümleri olsalar da algoritmaların işleyişlerinde bazı problemlerle karşılaşılmaktadır. Geliştirilen sezgisel algoritmalar çok zor ve karmaşık problemlerde beklenilen çözümlerde global çözümler yerine yerel çözüm tuzaklarına takılabilmektedir. Bu problemin temel nedenlerinden birisi çeşitlilik adımının algoritmanın işleyişinde zamanla azalmasıdır. Arama sürecinde uygunluk değeri en büyük olan çözüm adaylarının kullanılması halinde hızlı bir yakınsama oluşmakta ancak bu yakınsama genellikle yerel çözüm tuzağına yakalanmaya neden olmaktadır. Bu durum literatürde prematüre (engelli) yakınsama problemi olarak ifade edilmektedir. Bu durumu minimize problemi olan Şekil 1.3 ve 1.4 üzerinde inceleyelim.

**Şekil 1.3**

Popülasyon tabanlı sezgisel algoritmalarda, oluşturulan popülasyon ile çözüm uzayı gezinir. Keşif ( çeşitlilik ) ve sömürü ( komşuluk araması ) yapılarak en uygun değer bulunmaya çalışılır. Şekil 1.3’de grafik üzerinde verilen siyah daireler çeşitliliği ve kırmızı daireler ise komşuluk aramasında bulunan adımı ifade etmektedir. Siyah noktaların neredeyse tüm çözüm uzayına dağıldığı ve mavi daire - A ile işaretlenmiş alanda görüldüğü gibi komşuluk araması yapılarak global minimum noktasına yaklaşılmaktadır. Çeşitlilik sayesinde yeşil daire - B ile işaretlenmiş alana takılması yani yerel minimum tuzaklarından kaçınılması sağlanır.

Sezgisel algoritmalarda temel iki adım bulunmaktadır. Komşuluk araması ve çeşitlilik. Çeşitlilik ile çözüm uzayında büyük atlamalar yapılarak farklı noktalar bulunur. Komşuluk araması adımında ise, çeşitlilik adımında bulunan noktaların etrafında küçük adımlar ile hassas arama yapılır. Şekil 1.4’de sezgisel algoritmada çeşitlilik azalmıştır. Bu durumda çözüm uzayı tamamen taranamamaktadır. Algoritma hızlı yakınmasa problemine yakalanarak yerel çözüm tuzaklara takılır. Sezgisel algoritmaların performansları keşif ve sömürü arasındaki dengeye bağlıdır ve bu dengelerin ince ayarlar ile iyileştirilmesi gerekmektedir.



**Şekil 1.4**

## **Çalışmanın amacı**

Optimizasyon mühendislik tasarımından finansal piyasalara, bilgisayar bilimlerinden endüstriyel uygulamalara kadar çok geniş alanda uygulanmaktadır. Geliştirilen MSA algoritmaları karmaşıklık düzeyi yüksek olan optimizasyon problemlerini yerel çözüm tuzaklarına takılarak kabul edilebilir çözümlere ulaşamamaktadır. Literatürde MSA algoritmalarının performanslarını artırarak yerel çözüm tuzaklarına takılma problemleri, çoğunluğunu doğadan esinlenerek geliştirilen tekniklerin tatbik edilmesiyle çözülmeye çalışılmıştır.

MSA algoritmalarını geliştirmek için kullanılan teknikleri ve MSA algoritmalarının yaşam döngülerinin adımlarını üç başlık altında toplanacaktır. Bu çalışmayla birlikte daha sistematik bir çalışma ortamı hazırlanarak optimizasyon ve mühendislik tasarım problemlerini başarılı bir şekilde çözümleyebilen melez MSA algoritmaları geliştirilecektir.

## **Araştırma sorusu ve/veya hipotez**

Literatürdeki MSA algoritmaları üzerindeki iyileştirmeler incelendiğinde, çalışmaların birçoğu mevcut MSA tekniklerinin çeşitli yöntemlerle iyileştirilmesini ve varyasyonlarının geliştirilmesini konu almaktadır. Algoritmalarda arama performansları üzerinde etkili olan iki temel öğe seçim yöntemleri ve arama operatörleridir ve iyileştirme çalışmaları bu noktalar üzerinde yoğunlaşmaktadır. MSA algoritmalarının performanslarını etkileyebilecek noktalar olan popülasyon yaratma (D: dağılım), çözüm adayı seçimi (S: seçim) ve algoritma parametrelerinin kontrol (K: kontrol) bölümlerini DSK başlığı altında toplayarak iyileştirme yaklaşımlarını daha sistematik hale getirmek mümkündür. MSA algoritmalarına DSK yöntemlerinden biri ya da birkaçı kullanılarak algoritmaların baz modelleri üzerinde iyileşme elde etmek mümkündür.

## **Projede Kullanılan Yöntem ve Metotlar**

* 1. **DSK Yöntemi**

MSA algoritmaları doğadaki işleyişlerin taklit edilmesiyle oluşturulmuştur. Bu algoritmaların yetenekleri ve özellikleri farklı olsa da temel olarak algoritma yaşam döngüleri aynı adımlardan oluşmaktadır. Bir problemin MSA algoritma ile çözümlenmesi algoritma 1’de verilmiştir.

|  |
| --- |
| Algoritma 1. MSA algoritmalarının arama süreci |
| 1. Problemin yaratılması (uygunluk fonksiyonunun, ceza fonksiyonunun tanımlanması) 2. Çözüm adayının tasarımı ve çözüm adayları topluluğunun yaratılması 3. Adayların uygunluk değerlerinin hesaplanması 4. İteratif süreç    1. Komşuluk Araması    2. Çeşitliliğin Sağlanması    3. Çözüm adayı topluluğunun güncellenmesi 5. Sonlandırma kriteri sağlandı mı?    1. Hayır (Adım 4’e dön)    2. Evet (Arama sürecini sonlandır ve en iyi çözüm adayını kaydet) |

Arama sürecinde verilen 1, 2, 3 ve 5 numaralı adımlar MSA algoritmaları için ortak adımlardır. 4 numaralı adım ise MSA algoritmalarına özgü operatörlerin ve işlemlerin uygulandığı adımdır. Arama sürecinin başarısı bu adıma bağlıdır.

MSA algoritmalarının yerine getirmesi gereken iki gereksinimi olan komşuluk araması ve çeşitliliği başarılı bir şekilde yerine getirmesi için birçok faktör bulunmaktadır. Bu faktörler dört ana başlık altında toplanabilir. Bunlar sırayla dağılım yöntemleri [61], seçim yöntemleri [62], arama operatörleri [33-63] ve arama stratejisidir [64-65]. Arama operatörleri ve arama stratejisi kontrol başlığı altında gruplandırılarak algoritmalar için etkili olan faktörleri üç ana başlık altında toplayıp DSK olarak adlandırılmıştır. Algoritmalar kendi içlerinde DSK yöntemine göre bölünecektir. Bu bölümler MSA algoritmaları için temel adımlar olan dağılım, seçim ve kontrol bölümlerinden oluşmaktadır.

* + 1. **Dağılım Prosedürü**

MSA algoritmaların yaşam döngüsü, algoritmalarının arama sürecinde 2 numaralı adım olan çözüm adayının tasarımı ve çözüm adayları topluluğunun yaratılması ile başlar. Çözüm adayları alternatif dağılım yöntemleri kullanılarak arama uzayında konumlandırılırlar. Bu adım **dağılım** prosedürlerine dahil olmaktadır. Algoritmalarda iyileştirme elde edebilmek için çözüm adaylarının oluşturulmasında tatbik edilecek teknikler Levy Uçuşları, Gauss Dağılımı, Düzgün Dağılım ve Kaos Haritalarıdır.

* + 1. **Kontrol Prosedürü**

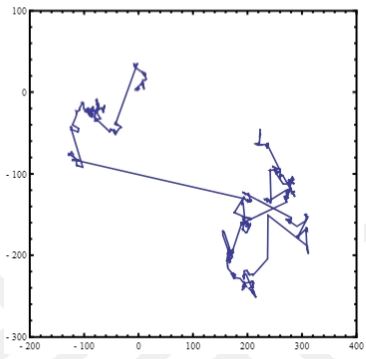
MSA algoritmalarında sezgisel arama sürecine 4 numaralı adıma eşlik eden algoritmalara ait tasarım parametreleri bulunmaktadır. Bu tasarım parametrelerine yerçekimsel arama algoritmasında kullanılan yerçekimi sabiti örnek verilebilir. Bu adım **kontrol** prosedürlerine dahil olmaktadır. Kontrol işleminde amaç MSA algoritmalarına ait tasarım parametrelerinin optimize edilmesidir. Kontrol prosedürlerinde iyileşme elde edebilmek için tatbik edilecek teknikler Levy Uçuşları ve Kaos Haritalarıdır.

* + 1. **Seçim Prosedürü**

Sezgisel arama sürecinde 4 numaralı adımda bulunur. Arama sürecini yönlendirecek – rehberlik edecek konumlar belirlenir. Bu adım **seçim** prosedürlerine dahil olmaktadır. Komşuluk araması ve çeşitlilik formüllerinde kullanılan çözüm adaylarını kapsamaktadır. Seçim prosedürlerinde iyileşme elde edebilmek için tatbik edilecek teknik Rulet Tekerleğidir.

* 1. **Tatbik Edilecek Teknikler**
     1. **Levy Uçuşu**

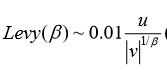
Randomizasyon, sürü zekasına dayalı optimizasyon tekniklerinde en önemli bileşendir. Sezgisel algoritmalarda randomizasyonu sağlayabilmek için düzgün dağılmış sayılar üreten rand fonksiyonu kullanılır. Bu fonksiyonun yerine ve MSA algoritmalarının performanslarını artıran Levy Uçuş Mekanizması kullanılmaktadır. Bu mekanizma α-kararlı dağılımına dayanan ve farklı ölçülerde adım büyüklüğü kullanarak büyük boyuttaki mesafelerde hareket etme kabiliyeti olan rastgele işlemlerdir. Levy α-kararlı dağılımı; ağır kuyruklu olasılık yoğunluk fonksiyonu, fraktal istatistik ve anormal dağınım ile son derece bağlantılıdır [58]. Şekil 1.5’te Levy Uçuşu’nun ilk 1000 adımdaki örneği [54]



**Şekil 1.5**

Randomizasyon işleminin Levy Uçuşu ile belirlenmesi şu şekildedir:

|  |  |
| --- | --- |
| Levy formülde 0.01 sayısı adım boyutunun kontrolü için kullanılmaktadır. |  |
| U ve N sayıları iki normal olasılıksal dağılımı ifade etmektedir |  |
| u formülünde kullanılan Г ifadesi standart gamma fonksiyonudur |  |



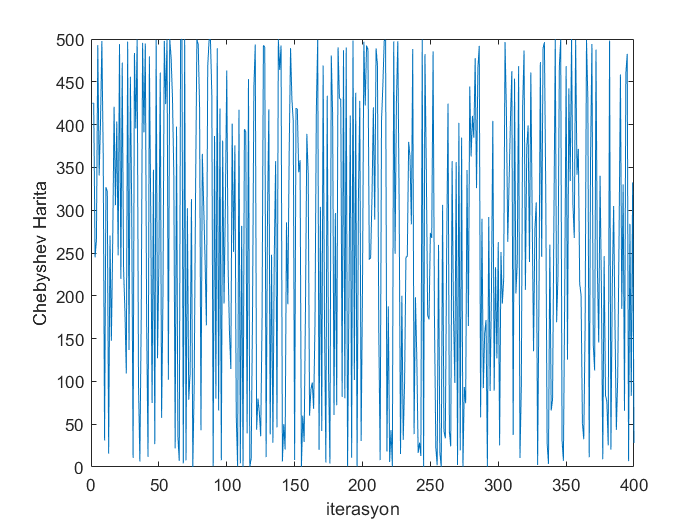
* + 1. **Kaotik Haritalar**

MSA algoritmalarında rastgele hareketi arttırmak için kullanılan Kaos Haritaları periyodik olmayan, yakınsamayan ve sınırlı olan, doğrusal olmayan dinamik sistemler bulunan deterministtik, rasgele benzeri bir süreçtir [59]. Temelinde kaotik haritalar olarak adlandırılan fonksiyonlar bulunmaktadır[999]. Ayrıca başlangıç şartları ve parametrelerine oldukça bağlıdır. Kaosun doğası görünürde rasgele ve tahmin edilemezdir. Ayrıca kendi içerisinde bir düzene sahiptir. Kaotik temelli araştırmalarda ayrı bir inceleme alanı olarak dikkatleri üzerine çekmektedir.

MSA algoritmaları, opmizasyon problemlerinde arama uzayında taranmayan alan bırakmamak ve rastgele hareketlilikte oluşabilecek daralmaları engellemek adına birçok Kaotik Harita çalışması kullanılmaktadır. Genellikle arama operatörleri üzerinde tatbik edilen Kaos Haritalarının literatürde on (10) farklı versiyonu bulunmaktadır. Bazı Kaos Haritalarının grafikleri ve sözde kodları verilmiştir.

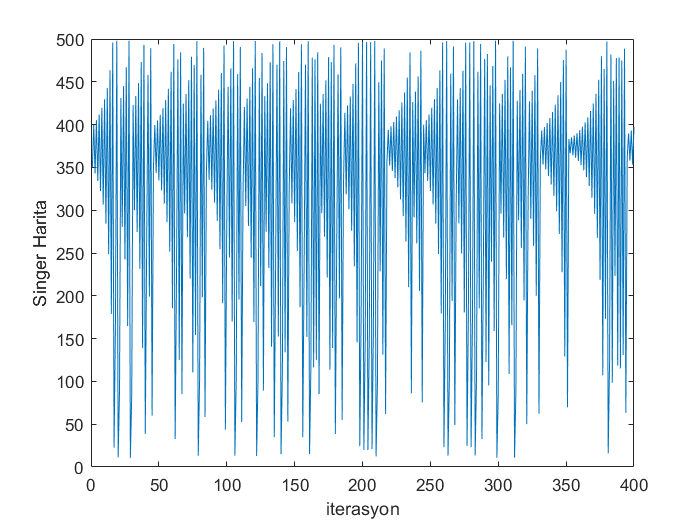
* + - 1. **Chebyshev Haritası**

|  |
| --- |
| Chebyshev Haritası sözde kodu |
| Value = 500; // Üretilecek en büyük sayı  X(1) = 0.7;  MaxGen = Maksimum iterasyon sayısı  for i:1= MaxGen  x(i+1)=cos(i\*acos(x(i)));  KaosDeger(i)=((x(i)+1)\*Value)/2;  end |

****

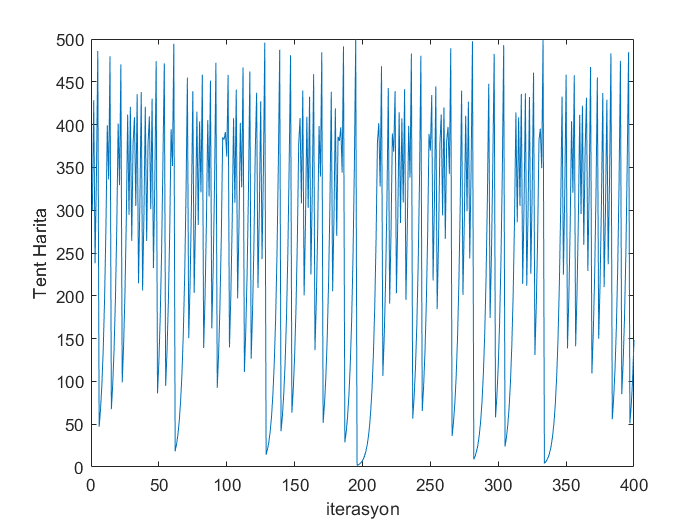
|  |
| --- |
| Singer Haritası sözde kodu |
| Value = 500; // Üretilecek en büyük sayı  X(1) = 0.7;  MaxGen = Maksimum iterasyon sayısı  u=1.07;  for i:1= MaxGen  x(i+1) = u\*(7.86\*x(i)-23.31\*(x(i)^2)+28.75\*(x(i)^3)-13.302875\*(x(i)^4));  KaosDeger(i) = (x(i))\*Value;  end |
|  |

* + - 1. **Singer Haritası**



* + - 1. **Tent Haritası**

|  |
| --- |
| Tent Haritası sözde kodu |
| Value = 500; // Üretilecek en büyük sayı  X(1) = 0.6;  MaxGen = Maksimum iterasyon sayısı  for i:1= MaxGen  if x(i)<0.7  x(i+1)=x(i)/0.7;  end  if x(i)>=0.7  x(i+1)=(10/3)\*(1-x(i));  end  KaosDeger(i) = (x(i))\*Value;  end |
|  |



* + 1. **Rulet Tekerleği**

Rulet Tekerleği olasılıksal seçim yöntemidir. Topluluk içerisindeki çözüm adaylarının uygunluk değerlerine bağlı olarak seçilme olasılıkları hesaplanır. Seçim işlemi bu olasılıklara bağlı olarak tek adımda gerçekleşir. Rulet Tekerleğinin sözde kodu algoritma 2’de verilmiştir.

|  |
| --- |
| Algoritma 2. Rulet seçim yöntemi sözde kodu |
| n = MSA algoritmasında çözüm adayı sayısı  m = Optimizasyon probleminin boyutu  X[n,m] = Çözüm adayları topluluğu,  U[n] = Çözüm adaylarının uygunluk değerleri  R[n] = Çözüm adaylarının rulet tekerleği yüzdeleri  K[n] = Çözüm adaylarının rulet tekerleği konumları,  t = 0, K[0] = 0;  for i=1:n  t=t+ U[n]  end  for i=1:n  R[i]=U[i]/ U[n]  K[i]=R[i] + K[i-1]  end  konum=rand (0,1) // rulet tekerleğini döndür ve tekerleğin durduğu konumu belirle  for i=1:n  if (K[i-1]<konum<= K[i])  Seçilen çözüm adayı=X[i]  end |

* + 1. **Gauss Dağılımı**

Gauss, iki parametreye sahip ve sürekli dağılım sağlayan bir yöntemdir. Bu parametreler konum (değer) bilgisini temsil eden μ ( aritmetik ortalama ) ve varyans bilgisini temsil eden ϭ2 yayılımdır [66-67]. Aritmetik ortalama parametresi çan eğrisinin tepe noktasını belirler. Normal dağılımın matematiksel ifadesi Eşitlik-1’de verilmektedir [61].

(1)

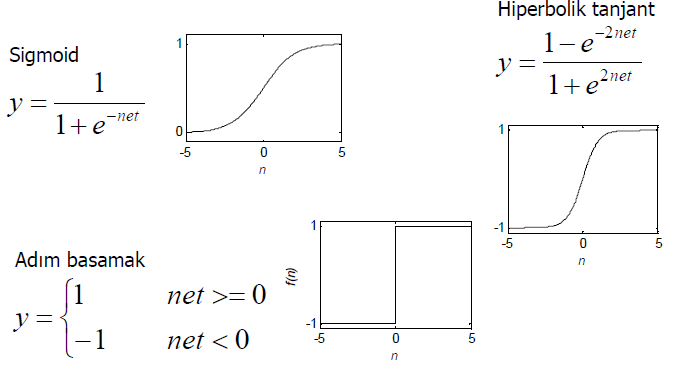
Gauss dağılımında değerler [-1, 1] aralığında üretilmektedir. Değerleri bu aralıkta üretmek için mean=0, varyans=0,4 olarak belirlenmiştir. Elde edilen değerlerin yoğunluk grafiği Şekil 6’da gösterilmektedir [61].

**Şekil 1.6**

* 1. **Tekniklerin Tatbik Edilme Yöntemi**

Levy Uçuşu, Kaos Haritaları ve Rulet Tekerleğinin MSA algoritmalarında tatbik ederken hangi oranda ve nasıl uygulanacağının bulunması gerekmektedir. Bunun için deney çalışması yapılacaktır. Bu deney çalışmaları:

1. Tatbik edilecek yöntemi belli oranlarda uygulayarak algoritmanın değişime verdiği tepkiyi test etmektir. Bu amaçla tatbik edilen yöntemleri % 0.1 - 100 arasında problemin boyutuna göre dinamik değişen veya sabit oran ile algoritma içerisinde uygulanması.
2. Tatbik edilecek yöntemi, algoritmanın yaşam döngüsü içerisinde azalan oranda veya artan oranda uygulamaktır. Bu oranı belirlemede kullanılacak yöntemler doğrusal veya doğrusal olmayan fonksiyonlardır. Bu amaçla kullanılacak ilk fonksiyon Şekil 1.7 de verilen Sigmoid ‘ir.

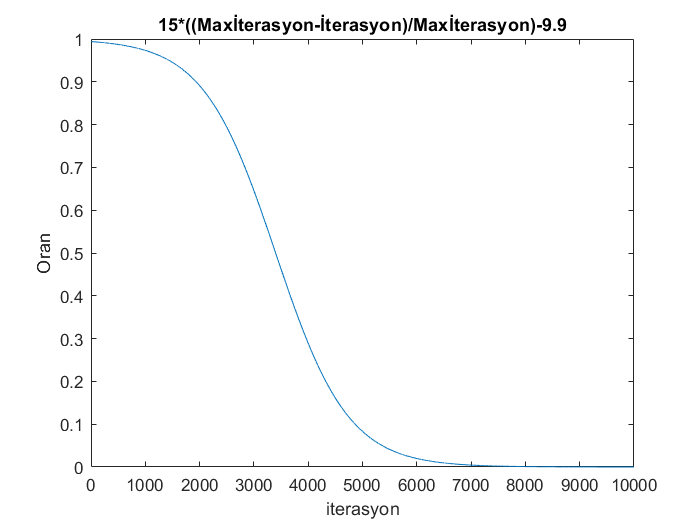


**Şekil 1.7**

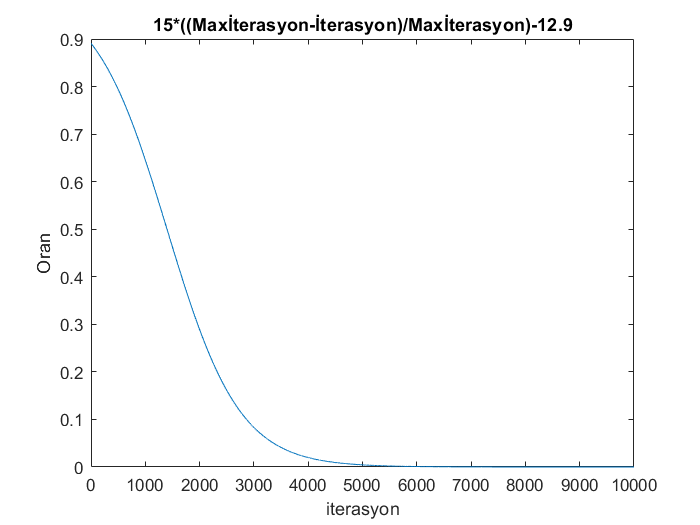
Sigmoid fonksiyonu yapay sinir ağlarında aktivasyon fonksiyonu olarak kullanılmaktadır. Verilen değişkeni farklı bir boyuta taşıyan doğrusal olmayan bir fonksiyondur. Sigmoid fonksiyonunda net değerin hesaplanması için önerilen formüller:

* Artan oranda Levy Uçuşu, Kaos Haritaları ve Rulet Tekerleğinin tatbik edilmesi için 15\*(İterasyon/Maxİterasyon)-A
* Azalan oranda Levy Uçuşu, Kaos Haritaları ve Rulet Tekerleğinin tatbik edilmesi için 15\*((Maxİterasyon-İterasyon)/Maxİterasyon)-A

Önerilen formüllerde normalizasyon işlemi kullanılarak iterasyon değeri 0-1 arasında ölçeklenmiştir. A katsayısı ile oranlardaki değişim kontrol edilmektedir. Bu değişim miktarları azalan oran için Şekil 1.8 ve Şekil 1.9 de görülmektedir.



**Şekil 1.8**

****

**Şekil 1.9**

* 1. **Deneysel Çalışmalar**

Literatürde sürekli değer optimizasyon problemleri tiplerine göre genellikle dört kategoriye ayrılmaktadır. Bunlar tek modlu, çok modlu, melez ve komposizyon (compostion) problemlerdir. Bu problem tipleri, 1994 yılından buyana her yıl düzenli bir şekilde gerçekleştirilen IEEE evrimsel hesaplama (IEEE CEC) konferanslarındaki çalışmaların da katkılarıyla ortaya çıkmıştır [72]. MSA algoritmalarını test etmek ve rakip algoritmalarla karşılaştırarak arama performanslarını doğrulamak amacıyla bu dört problem tipinden test fonksiyonlarını içeren karşılaştırma problemleri havuzu oluşturulmaktadır. Buna göre tek modlu problemlerde yerel çözüm tuzakları bulunmamaktadır. Bu problemler algoritmaların yakınsama hızlarının test edilmesi amacıyla kullanılmaktadırlar [73-34]. Komşuluk arama yeteneği yüksek algoritmalar bu problem türünde başarılı olmaktadırlar. Çok modlu problemler ise yerel çözüm tuzakları barındıran problem türüdür. Örneğin Michalewicz test fonksiyonunda yerel minimum sayısı, problem boyutuna (n) bağlı olarak faktöriyeli (n!) ifadesiyle değişmektedir. Yerel çözüm tuzakları çok modlu problemlerin optimizasyonu zorlaştırmaktadır [74]. Çok modlu problemlerin arama uzaylarındaki tuzaklardan kurtulmak algoritmaların çeşitlilik sağlama yeteneğine bağlıdır. MSA algoritmalarının çeşitlilik işlevlerini test etmek amacıyla çok modlu test fonksiyonları kullanılmaktadır. Melez ve derleme problem türleri ise algoritmaların hem komşuluk araması hem de çeşitlilik yeteneklerini dengeli bir şekilde yönetmelerini gerektirmektedir. Dolayısıyla bu iki problem türündeki test fonksiyonları da MSA algoritmalarının yakınsama hızı ve çeşitlilik dengesini ölçmek amacıyla kullanılmaktadırlar.

MSA algoritmalarında geliştirme çalışmaları Matlab R2018a programında yapılacaktır. Geliştirme çalışmalarında kullanılacak test ve doğrulama problemleri 30’u Klasik Benchmark, 30’u CEC 2014 [69, 70] ve 30’u CEC 2017 [71] olmak üzere toplam 90 test problemi üzerinde çalışılacaktır. CEC2017 ve CEC2014 ‘de sürekli değerli ve dinamik yapılı test problemleri bulunmaktadır. Dolayısıyla problemlere ait tasarım parametrelerinin arama uzayındaki optimum konumlarının da kaydırma ve döndürme işlevleri yoluyla dinamik olarak değiştirilebildiği ve bu yolla MSA algoritmalarının çeşitli yollarla optimum noktaları yakalayacak avantajlar yaratmasının önüne geçilmeye çalışılacaktır. CEC konferanslarındaki test problemlerinin tamamı dinamik olarak boyutlandırılabilen problemlerle oluşturulmuştur. Böylelikle küçük boyutlu arama uzaylarında hızlı yakınsama özellikleri sayesinde başarılı görülen algoritmaların aynı problemlerin orta ve büyük boyutlu arama uzaylarındaki performanslarını da ortaya çıkarmak amaçlanmıştır. Tablo 1’de MSA algoritmalarının geliştirildiği çalışmalarda en sık kullanılan test ve karşılaştırma problemleri[68], Tablo 2’de CEC2017 ‘den alınmış çok modlu, kaydırma ve döndürme işlevli test ve karşılaştırma problemlerinin birkaçı ve bu problemlere ait bilgiler verilmektedir.

|  |  |  |  |
| --- | --- | --- | --- |
| **Problem Adı** | **Grafiksel Gösterim ve Matematiksel İfade** | **Kountur Grafiği** | **Özellikleri** |
| Rastrigin | Rastrigin Function | Rastrigin Function | * Sürekli * Konveks * n-boyutlu uzayda tanımlanabilir * Çok modlu (birden fazla yerel min.) * Türevlenebilir * Ayrılabilir |
| Sphere | Sphere Function |  | * Sürekli * Konveks * n-boyutlu uzayda tanımlanabilir * Tek modlu * Türevlenebilir * Ayrılabilir * xi∈[−5.12,5.12],i=1..n |
| Ackley |  |  | * Sürekli * Konveks değil * n-boyutlu uzayda tanımlanabilir * Çok modlu * Ayrılabilir * xi∈[−32,32], i=1..n |

**Tablo 1. En Fazla Kullanılan Benchmark Problemlerinin Birkaçı ve Bu Problemlere Ait Bilgiler**

**Tablo 2. CEC2017 Çok Modlu Problemlerinin Birkaçı ve Bu Problemlere Ait Bilgiler**

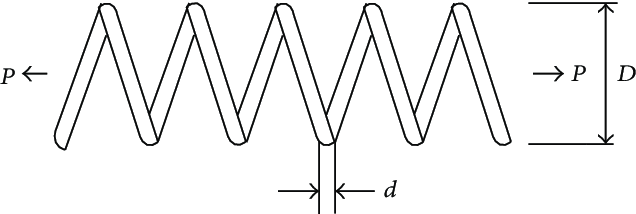
|  |  |  |  |
| --- | --- | --- | --- |
| **Problem Adı** | **Grafiksel Gösterim ve Matematiksel İfade** | **Kountur Grafiği** | **Özellikleri** |
| Rastrigin |  |  | * Çok modlu (birden fazla yerel min.) * Ayrılmaz * İkinci daha iyi yerel optimum, küresel optimumdan uzaktır. |
| Schaffer |  |  | * Çok modlu (birden fazla yerel min.) * Ayrılmaz * Asimetrik |
| Rosenbrock |  |  | * Çok modlu (birden fazla yerel min.) * Ayrılmaz |

Çalışmalarda amaç fonksiyonu azami çağırma sayısı baz alınacaktır. Amaç fonksiyonu azami çağırma sayısı problemin boyutunun 10 bin katı olarak belirlenmiştir. Deneysel çalışma sonuçları istatistiksel olarak wilcoxon test ile analiz edilecektir. Wilcoxon testi iki örneklem ortalamaları arasında anlamlı olan farklıları tespit etmeyi amaçlar. Eğer iki algoritmanın çıkışlarının kıyaslanması için kullanılacaksa, test pratik olarak iki algoritmanın karşılıklı davranışkarını değerlendirir [77].

Çalışmalarda iyileştirme yapılan algoritmalar, kısıtlı mühendislik tasarım problemlerinde tatbik edilecektir. Tatbik edilerek optimizasyonu yapılacak ve kısıtlı mühendislik tasarım problemlerinin tanımları aşağıda verilmiştir.

1. **Gerginlik / Sıkıştırma Yayı Tasarımı**

Bu sorunun asıl amacı yayın ağırlığını en aza indirmektir. Sorunun tanımında üç parametre vardır. Bunlar tel çapı (d), ortalama bobin çapı (D) ve aktif bobinlerin sayısı (P). Optimizasyon sürecinde, problem sınırlamaları dalgalanma frekansı, minimum sapma ve kayma gerilmesi dikkate alınmaktadır. Şekil 2.0 ‘da Gerginlik / Sıkıştırma Yayı Tasarım Sorunu [75] verilmiştir.

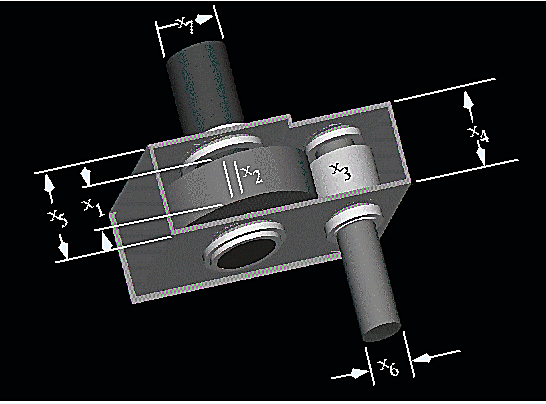


**Şekil 2.0**

Şekil 2.0, sorunun şematik gösterimini göstermektedir. Değişkenlerin aralıkları, amaç fonksiyonun matematiksel formülasyonu ve problemin kısıtlamaları aşağıdaki şekilde tanımlanır:

1. Parametreler
2. Formül
3. Kısıtlar
4. Parametre Çözüm Aralığı
5. **Hız Düşürücü Tasarımı**

Hız düşürücü tasarımı, minimize etme sorunudur. Bu sorunun amacı, asgari hız düşürücü ağırlığını bulmaktır. Yeni tasarım parametresine sahiptir. Bunlar yüz genişliği (x1), diş modülü (x2), piyondaki diş sayısı (x3), yataklar arasındaki ilk milin uzunluğu (x4), yataklar arasındaki ikinci milin uzunluğu (x5) ve iki şaftın çapları (x6, x7). Şekil 2.1’ de Hız Düşürücü Tasarım Problemi [76] verilmiştir.



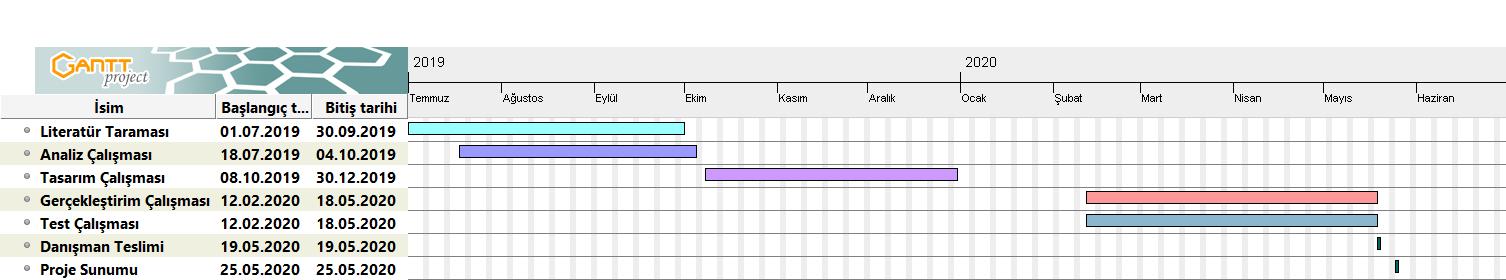
**Şekil 2.1**

Şekil 2.1, sorunun şematik gösterimini göstermektedir. Değişkenlerin aralıkları, amaç fonksiyonun matematiksel formülasyonu ve problemin kısıtlamaları aşağıdaki şekilde tanımlanır:

1. Formül
2. Kısıtlar
3. Parametre Çözüm Aralığı

# **Proje İş-Zaman Planı**

Proje iş zaman çizelgesi, yöntem bölümünün takvime bağlanmasıyla hazırlanmıştır. Bu proje çalışmasının hazırlıkları danışmanımızın yönlendirmesiyle 2020 yılı mayıs ayına kadar uzanmaktadır. Dolayısıyla ilişkili çalışmalara yönelik literatür taraması temmuz ayından itibaren başlanmıştır. Temmuz – Eylül döneminde literatür taraması neticesinde elde edilen dokümanların analizi yapılarak, MSA algoritmaları üzerinde yapılmış iyileştirme çalışmaları ve bu çalışmalarda kullanılan yöntemlerin uygulanış teknikleri incelenmiştir. Ekim ayı itibariyle, belirlenen MSA algoritmaları DSK yöntemlerine uyun biçimde kendi içlerinde bölümlendirilerek uygulanabilecek yöntemler belirlenecektir. Hazırlanan MSA algoritmalarına belirlenen yöntemleri, metotlar kısmında belirlenmiş uygulanış teknikleri ile gerçekleştirim çalışmalarına başlanırken aynı zamanda test çalışmaları sürdürülecektir. Proje iş zaman planı Şekil 2.3 ‘ de verilmiştir.



**Şekil 2.3**

# **Sonuç**

Klasik optimizasyon tekniklerinin yetersiz kalmasından dolayı geliştirilen doğa esinli MSA algoritmaları birçok alanda kullanılmaktadır. Yerel çözüm tuzaklarına takılabilen MSA algoritmalarında iyileştirme yapılarak performanslarını artırmak için literatürde yüzlerce çalışma yapılmıştır. Yapılan çalışmaların büyük çoğunluğu MSA algoritmalarının arama stratejisi ve arama operatörleri üzerinde yapılmaktadır.

Bu proje önerisinde MSA algoritmalarında yapılacak iyileştirme çalışmaları DSK yöntemine göre algoritmalar kendi içlerinde üç bölüme ayrılmıştır. Bu bölümler ve bu bölümlerde kullanılabilecek teknikler tanımlanmış ve tanıtılmıştır. DSK yöntemine göre bölünmüş güncel MSA algoritmaları belirlenen teknikler ile performansları iyileştirilerek optimizasyon problemlerinde maliyetler azaltılacaktır.

# **Kaynaklar**

1. Haklı, H. (2013). Sürekli fonksiyonların optimizasyonu için doğa esinli algoritmaların geliştirilmesi (Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü).
2. Bayindir, R., Colak, I., Sagiroglu, S., & Kahraman, H. T. (2012, December). Application of adaptive artificial neural network method to model the excitation currents of synchronous motors. In 2012 11th International Conference on Machine Learning and Applications (Vol. 2, pp. 498-502). IEEE.
3. Kahraman, H. T., Sagiroglu, S., & Colak, I. (2013). The development of intuitive knowledge classifier and the modeling of domain dependent data. Knowledge-Based Systems, 37, 283-295.
4. Lai, W., Zhou, M., Hu, F., Bian, K., & Song, Q. (2019). A New DBSCAN Parameters Determination Method Based on Improved MVO. IEEE Access, 7, 104085-104095.
5. Pham, H. N. A., & Triantaphyllou, E. (2009). An application of a new meta-heuristic for optimizing the classification accuracy when analyzing some medical datasets. Expert Systems with Applications, 36(5), 9240-9249.
6. Kahraman, H. T. (2016). A novel and powerful hybrid classifier method: Development and testing of heuristic k-nn algorithm with fuzzy distance metric. Data & Knowledge Engineering, 103, 44-59.
7. Fong, S., Deb, S., & Yang, X. S. (2018). How meta-heuristic algorithms contribute to deep learning in the hype of big data analytics. In Progress in Intelligent Computing Techniques: Theory, Practice, and Applications (pp. 3-25). Springer, Singapore.
8. Tayal, A., & Singh, S. P. (2018). Integrating big data analytic and hybrid firefly-chaotic simulated annealing approach for facility layout problem. Annals of Operations Research, 270(1-2), 489-514.
9. Dosoglu, M. K., Guvenc, U., Duman, S., Sonmez, Y., & Kahraman, H. T. (2018). Symbiotic organisms search optimization algorithm for economic/emission dispatch problem in power systems. Neural Computing and Applications, 29(3), 721-737.
10. Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under Industry 4.0. Journal of Intelligent Manufacturing, 30(4), 1809-1830.
11. Tian, D., Zhao, X., & Shi, Z. (2019). Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization. Swarm and Evolutionary Computation, 100573.
12. Gupta, S., & Deep, K. (2019). A hybrid self-adaptive sine cosine algorithm with opposition based learning. Expert Systems with Applications, 119, 210-230.
13. Jana, B., Mitra, S., & Acharyya, S. (2019). Repository and Mutation based Particle Swarm Optimization (RMPSO): A new PSO variant applied to reconstruction of Gene Regulatory Network. Applied Soft Computing, 74, 330-355.
14. Wu, L., Liu, Q., Tian, X., Zhang, J., & Xiao, W. (2018). A new improved fruit fly optimization algorithm IAFOA and its application to solve engineering optimization problems. Knowledge-Based Systems, 144, 153-173.
15. Sun, G., Ma, P., Ren, J., Zhang, A., & Jia, X. (2018). A stability constrained adaptive alpha for gravitational search algorithm. Knowledge-Based Systems, 139, 200-213.
16. Long, W., Jiao, J., Liang, X., & Tang, M. (2018). An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. Engineering Applications of Artificial Intelligence, 68, 63-80.
17. Awad, N. H., Ali, M. Z., Mallipeddi, R., & Suganthan, P. N. (2018). An improved differential evolution algorithm using efficient adapted surrogate model for numerical optimization. Information Sciences, 451, 326-347.
18. Al-Bahrani, L. T., & Patra, J. C. (2018). A novel orthogonal PSO algorithm based on orthogonal diagonalization. Swarm and Evolutionary Computation, 40, 1-23.
19. Torabi, S., & Safi-Esfahani, F. (2018). Improved raven roosting optimization algorithm (IRRO). Swarm and Evolutionary Computation, 40, 144-154.
20. Tian, D., & Shi, Z. (2018). MPSO: Modified particle swarm optimization and its applications. Swarm and Evolutionary Computation.
21. Chegini, S. N., Bagheri, A., & Najafi, F. (2018). PSOSCALF: A new hybrid PSO based on Sine Cosine Algorithm and Levy flight for solving optimization problems. Applied Soft Computing, 73, 697-726.
22. Zhong, F., Li, H., Zhong, S. 2017. “An improved artificial bee colony algorithm with modified-neighborhood-based update operator and independent-inheriting-search strategy for global optimization”, Engineering Applications of Artificial Intelligence, 58, 134-156.
23. Ouyang, H. B., Gao, L. Q., Li, S., Kong, X. Y., Wang, Q., Zou, D. X. 2017. “Improved harmony search algorithm: LHS”, Applied Soft Computing, 53, 133-167.
24. Harfouchi, F., Habbi, H., Ozturk, C., & Karaboga, D. (2017). Modified multiple search cooperative foraging strategy for improved artificial bee colony optimization with robustness analysis. Soft Computing, 1-24.
25. Awad, N. H., Ali, M. Z., Suganthan, P. N., & Reynolds, R. G. (2017). CADE: a hybridization of cultural algorithm and differential evolution for numerical optimization. *Information Sciences*, *378*, 215-241.
26. Mortazavi, A., Toğan, V., & Nuhoğlu, A. (2018). Interactive search algorithm: a new hybrid metaheuristic optimization algorithm. *Engineering Applications of Artificial Intelligence*, *71*, 275-292.
27. Ewees, A. A., Elaziz, M. A., & Houssein, E. H. (2018). Improved grasshopper optimization algorithm using opposition-based learning. *Expert Systems with Applications*, *112*, 156-172.
28. Arora, S., & Singh, S. (2019). Butterfly optimization algorithm: a novel approach for global optimization. *Soft Computing*, *23*(3), 715-734.
29. Civicioglu, P., Besdok, E., Gunen, M. A., & Atasever, U. H. (2018). Weighted differential evolution algorithm for numerical function optimization: a comparative study with cuckoo search, artificial bee colony, adaptive differential evolution, and backtracking search optimization algorithms. *Neural Computing and Applications*, 1-15.
30. 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.
31. Mohamed, A. W., & Mohamed, A. K. (2019). Adaptive guided differential evolution algorithm with novel mutation for numerical optimization. International Journal of Machine Learning and Cybernetics, 10(2), 253-277.
32. Yadav, A. (2019). AEFA: Artificial electric field algorithm for global optimization. Swarm and Evolutionary Computation.
33. W. Zhao, L. Wang and Z. Zhang, Atom search optimization and its application to solve a hydrogeologic parameter estimation problem, Knowledge-Based Systems (2019), 163, 283-304.
34. Tang, D., Liu, Z., Yang, J., & Zhao, J. (2018). Memetic frog leaping algorithm for global optimization. Soft Computing, 1-29.
35. Wang, G. G. (2018). Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memetic Computing, 10, 151-164.
36. Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software, 114, 163-191.
37. 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.
38. Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Computers & Structures, 169, 1-12.
39. Punnathanam, V., & Kotecha, P. (2016). Yin-Yang-pair Optimization: A novel lightweight optimization algorithm. Engineering Applications of Artificial Intelligence, 54, 62-79.
40. Mittal, H., Pal, R., Kulhari, A., & Saraswat, M. (2016, August). Chaotic kbest gravitational search algorithm (ckgsa). In Contemporary Computing (IC3), 2016 Ninth International Conference on (pp. 1-6). IEEE.
41. Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Software, 95, 51-67.
42. Shareef, H., Ibrahim, A. A., & Mutlag, A. H. (2015). Lightning search algorithm. Applied Soft Computing, 36, 315-333.
43. Cheng, Min-Yuan, and Doddy Prayogo. "Symbiotic organisms search: a new metaheuristic optimization algorithm." Computers & Structures 139 (2014): 98-112.
44. Gandomi, A. H. (2014). Interior search algorithm (ISA): a novel approach for global optimization. ISA transactions, 53(4), 1168-1183.
45. Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.
46. P. Civicioglu, "Backtracking Search Optimization Algorithm for numerical optimization problems", Applied Mathematics and Computation, 219, 8121–8144, 2013.
47. P. Civicioglu, "Transforming Geocentric Cartesian Coordinates to Geodetic Coordinates by Using Differential Search Algorithm", Computers and Geosciences, 46, 229-247, 2012.
48. 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.
49. Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. Applied mathematics and computation, 214(1), 108-132.
50. Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: a gravitational search algorithm. Information sciences, 179(13), 2232-2248.
51. Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. Swarm intelligence, 1(1), 33-57.
52. 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.
53. Katırcıoğlu, F. (2016). Yerçekimi arama algoritması için yeni operatörlerin geliştirilmesi (Doktora Tezi, Düzce Üniversitesi Fen Bilimleri Enstitüsü).
54. Gülcan, H. (2018). Yusufçuk algoritmasının brownian hareketi ile iyileştirilmesi (Yüksek Lisans Tezi, Mersin Üniversitesi Fen Bilimleri Enstitüsü).
55. Kazak, N. (2011). Geliştirilmiş yerçekimsel arama algoritması(Yüksek Lisans Tezi, Bilecik Üniversitesi, Fen Bilimleri Enstitüsü).
56. Mirjalili, S., & Gandomi, A. H. (2017). Chaotic gravitational constants for the gravitational search algorithm. Applied soft computing, 53, 407-419.
57. Cigal, T. (2018). Sürekli zamanlı kaotik sistem tabanlı balina optimizasyon algoritmasının geliştirilmesi (Yüksek Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü).
58. Hınıslıoğlu, Y. (2018). Kaotik güve sürü algoritması kullanarak rüzgâr gücü entegreli optimal güç akışı (Yüksek Lisans Tezi, Düzce Üniversitesi, Fen Bilimleri Enstitüsü).
59. Alataş, B., Akın, E., & Özer, A. B. (2007). Kaotik Haritalı Parçacık Sürü Optimizasyon Algoritmaları. Eskişehir Osmangazi Üniversitesi Elektrik Elektronik Bilgisayar Biyomedikal Mühendisliği Ulusal Kongresi.
60. 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>.
61. Kahraman, H. T., Aras, S., Guvenc, U., & Sonmez, Y. (2017, October). Exploring the effect of distribution methods on meta-heuristic searching process. In 2017 International Conference on Computer Science and Engineering (UBMK) (pp. 371-376). IEEE.
62. Sun, W., Lin, A., Yu, H., Liang, Q., & Wu, G. (2017). All-dimension neighborhood based particle swarm optimization with randomly selected neighbors. Information Sciences, 405, 141-156.
63. Tu, Q., Chen, X., & Liu, X. (2019). Multi-strategy ensemble grey wolf optimizer and its application to feature selection. Applied Soft Computing, 76, 16-30.
64. Tian, M., & Gao, X. (2019). Differential evolution with neighborhood-based adaptive evolution mechanism for numerical optimization. Information Sciences, 478, 422-448.
65. Draa, A., Chettah, K., & Talbi, H. (2018). A Compound Sinusoidal Differential Evolution algorithm for continuous optimization. Swarm and Evolutionary Computation.
66. N. Higashi, H. Iba, Particle swarm optimization with gaussian mutation, in: Swarm Intelligence Symposium, 2003. SIS’03. Proceedings of the 2003 IEEE, 72–79.
67. Mahi, M., Baykan, Ö. K., Kodaz, H., “A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem”, Applied Soft Computing, 30, 484–490, (2015).
68. Kahraman H.T., Aras S., Gedikli E., “META-SEZGİSEL OPTİMİZASYON ÇALIŞMALARINDA BENCHMARK PROBLEMLERİNDE KARŞILAŞILAN STANDARTSIZLIKLAR VE ÇÖZÜM ÖNERİLERİ”, INTERNATIONAL ACADEMIC RESEARCH CONGRESS, ANTALYA, TÜRKİYE, 30 Ekim – 3Aralık 2018, pp. 1494-1501.
69. Xiang, W. L., Meng, X. L., Li, Y. Z., He, R. C., An, M. Q. 2018. “An improved artificial bee colony algorithm based on the gravity model”, Information Sciences, 429, 49-71.
70. J. J. Liang, B-Y. Qu, P. N. Suganthan, Alfredo G. Hernández-Díaz, "Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session and Competition on Real-Parameter Optimization", Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, January 2013.
71. N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu and P. N. Suganthan, "[Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization](http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2Fepnsugan%2FPublicSite%2FShared%20Documents%2FCEC%2D2017&View=%7bDAF31868%2d97D8%2d4779%2dAE49%2d9CEC4DC3F310%7d),"  Technical Report, Nanyang Technological University, Singapore, November 2016.
72. 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 Nanyang Technological University, Singapore.
73. Han, X., Liu, Q., Wang, H., & Wang, L. (2018). Novel fruit fly optimization algorithm with trend search and co-evolution. Knowledge-Based Systems, 141, 1-17.
74. W. Gao, S. Liu, L. Huang, A novel artificial bee colony algorithm based modified search equation and orthogonal learning, IEEE Trans. Cybern. 43 (3) (2013) 1011–1024.
75. 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).
76. 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).
77. Acılar, A. M. (2013). Yapay bağışıklık algoritmaları kullanılarak bulanık sistem tasarımı (Doktora Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü).

[999]http://www.emo.org.tr/ekler/a18f3fd78c47058\_ek.pdf