Düzce Üniversitesi Bilim ve Teknoloji Dergisi

Araştırma Makalesi

Çok Merkezli Girdap Arama Algoritması

Tahir SAĞ [a,*]

[a] *Bilgisayar Mühendisliği Bölümü, Teknoloji Fakültesi, Selçuk Üniversitesi, Konya, TÜRKİYE*
** Sorumlu yazarın e-posta adresi: tahirsag@selcuk.edu.tr*
DOI: 10.29130/dubited.644881

ÖZET

Girdap Arama Algoritması (GAA) karıştırılan sıvılarda oluşan girdap deseninden esinlenerek yakın zamanda geliştirilmiş tek-çözüm temelli meta-sezgisel bir optimizasyon algoritmasıdır. GAA algoritmasında, bir merkez etrafında iteratif olarak adaptif adım-boyutu ayarlaması ile daraltılan bir yarıçap içinde üretilen komşu çözümler aracılığıyla arama işlemi gerçekleştirilir. Bu strateji, algoritmaya bir kolaylık ve hız kazandırmasına rağmen ekstremum noktası fazla olan problemlerde yerel optimumlara takılma riski oluşturmaktadır. Bu çalışmada, bu dezavantajı gidermek ve GAA algoritmasının arama hassasiyetini iyileştirmek amacıyla bir modifikasyon önerilmektedir. Öncelikle arama uzayı birbiriyle örtüşmeyen 4 farklı alt-bölgeye ayrılır. Daha sonra, standart merkez noktası ile birlikte her bir alt-bölgede birer tane olmak üzere toplam 5 merkez noktası tanımlanır. Her merkezin yarıçap uzunluğu bulunduğu bölgenin aralığına göre ayrı ayrı hesaplanır. Böylece birbirinden bağımsız 5 girdap oluşturularak aday çözüm çeşitliliği arttırılmış olur. Düşük yerellikten faydalanılan ilk iterasyonlar boyunca bu 5 girdap paralel şekilde çalıştırılır. Toplam iterasyon sayısının yarısından sonra, merkez sayısı 2'ye indirilerek yüksek yerellikten daha etkin faydalanılması sağlanır. Önerilen Çok-Merkezli Girdap Arama Algoritması (ÇM-GAA) 50 test fonksiyonu üzerinde 50'şer defa bağımsız şekilde çalıştırılmış ve istatistiksel değerler hesaplanmıştır. Elde edilen sonuçlar standart GAA ile karşılaştırıldığında; önerilen ÇM-GAA algoritması hemen hemen tüm fonksiyonlarda kayda değer bir iyileştirme sağlayarak ciddi bir başarı göstermiştir.

Anahtar Kelimeler: *Çok-Merkezli Girdap Arama Algoritması, Girdap Arama Algoritması, Meta-sezgisel Optimizasyon*

Multi-Centered Vortex Search Algorithm

ABSTRACT

Vortex Search (VS) is a recently developed single-solution meta-heuristic optimization algorithm inspired by the vortex pattern in the mixed fluids. In the VS algorithm, the search is performed by neighbor-solutions generated within a radius decreased by iteratively adaptive step-length adjustment around a center. Although this strategy provides simplicity and speed to the algorithm, it poses a risk of falling into local optimums for problems having many extrema. In this study, a modification is proposed to overcome this disadvantage and improve the search capability of the GAA algorithm. First, the search space is divided into 4 different sub-regions that do not overlap each other. Next, five center points are identified, one per each sub-region, along with the standard center point. The radius length of each center is calculated separately according to the range of the region where it is located. Thus, 5 independent vortices are created to increase the variety of candidate solutions. These 5 vortices are run in parallel during the initial iterations, which benefit from low locality. After half of the total number of iterations, the number of centers is reduced to 2, enabling higher localization to be utilized more effectively. The proposed Multi-Centered Vortex Search Algorithm (MC-VS) was run 50 times independently on 50 test functions and

Geliş: 10/11/2019, Düzeltme: 02/02/2020, Kabul: 05/02/2020

statistical values were calculated. When the results were compared with the standard GAA; the proposed MC-VS algorithm has shown a significant success by providing a significant improvement in almost all functions.

Keywords: *Multi-Centered Vortex Search Algorithm, Vortex Search Algorithm, Metaheuristic Optimization*

I. GİRİŞ

Farklı alanlardaki birçok mühendislik problemi doğası gereği bir optimizasyon problemi şeklinde tanımlanabilmektedir. Örneğin; uzay mühendisleri toplam ağırlığın minimize edildiği uçak tasarımlarının oluşturulmasıyla; kimya mühendisleri optimal üretim oranını sağlayan proses tesisini tasarlamak ve işletmekle; makine mühendisleri, en düşük üretim maliyetini veya en yüksek bileşen ömrünü sağlayan mekanik bileşenlerin tasarımıyla; endüstri mühendisleri toplam iş bitirme zamanını kısaltmak amacıyla üretim sürecinde kullanılan makinelerin boşta kalma sürelerini en aza indirecek zamanlamanın tasarlanmasıyla; inşaat mühendisleri en düşük maliyet ve en yüksek dayanıklılığı sağlayan binaların tasarlanmasıyla; elektrik mühendisleri bir düğümden diğerine iletişim için gereken minimum zamanı elde eden iletişim ağlarını tasarlamakla ilgilenir. Bu nedenle, optimizasyon alanı onlarca yıldır popülerliğini yitirmeyen bir araştırma konusu olmaya devam etmektedir. Üstelik bu alanda geliştirilen algoritmaların pek çok mühendislik problemi için etkin birer çözüm aracı olarak kullanılmaya başlanması; daha karmaşık problemleri ele alma isteğini dolayısıyla yeni algoritmaların geliştirilmesi ihtiyacını doğurmuştur [1, 2].

Optimizasyon alanında odaklanılması gereken esas durum tüm problemler için genel geçer en iyi olan bir algoritma geliştirmek değil; belirli bir problem türü için adapte edilmiş algoritmanın geliştirilmesi veya aranmasıdır. Nitekim NFL teoremi, bazı varsayımlar altında, hiçbir optimizasyon algoritmasının tüm olası optimizasyon problemlerinde diğerlerinden üstün olmadığını söyler. Bu teoreme göre; bir meta-sezgisel algoritma verilen bir problem için diğer bir algoritmadan daha iyi performans sergiliyorsa, diğer algoritmanın her zaman daha başarılı olduğu başka bir problem vardır [3]. Literatürde zaman içinde bu teoreme dayanarak geliştirilmiş çok sayıda meta-sezgisel optimizasyon algoritması önerilmiştir. Bu algoritmalar genellikle (i) evrimsel algoritmalar, (ii) sürü-zekâsı algoritmaları ve (iii) fizik-temelli algoritmalar olmak üzere üç ana kategori altında toplanır. Evrimsel algoritmalar biyolojik evrimden ilham alan mekanizmaları kullanan popülasyona dayalı akıllı optimizasyon yöntemleridir. Genetik Algoritma (GA) [4] ve Diferansiyel Evrim Algoritması (DE) [5] bu kategorinin en bilinen algoritmalarıdır. Sürü Zekâsı algoritmaları doğada tekil olarak basit ajanlar olmalarına rağmen kolektif biçimde karmaşık görevleri yerine getirebilen kuş, balık, böcek kolonileri gibi merkeziyetçi ve kendi kendine organize olan canlıların ortak davranışlarından ilham alarak geliştirilen tekniklerdir. Parçacık Sürü Optimizasyonu (SPO) [6], Yapay Arı Kolonisi (ABC) [7], Karınca Koloni Optimizasyonu (ACO) [8] ve Guguk Kuşu Arama (CS) [9] algoritmaları bu kategoride en çok çalışma yapılmış olan algoritmalardır. Fizik-Temelli algoritmalar ise evrendeki fiziksel kuralları taklit ederek global optimuma yakınsamaya çalışan meta-sezgisel tekniklerdir. Bu kategorideki algoritmalardan bazıları Benzetimsel Tavlama (SA) [10], Yerçekimsel Arama Algoritması (GSA) [11], Su Buharlaşma Optimizasyonu [12] ve Termal Değişim Optimizasyonu [13] şeklinde sıralanabilir.

Meta-sezgisel optimizasyon algoritmalarında odaklanılan iki önemli süreç bulunur. Bunlar keşif (exploration) ve sömürme (explotation) süreçleridir. Keşif süreci, doğrudan global arama ile ilgilidir. Bu süreçte, global optimum değerin konumunu belirleyebilmek amacıyla problemin arama uzayı olabildiğince keşfedilmeye çalışılır. Birçok optimizasyon probleminde arama uzayının sonsuz olduğu düşünüldüğünde, bu sürecin ne denli önemli olduğu ve algoritmaların bu amaçla geliştirdikleri stratejilerin global optimuma ulaşmadaki etkisi anlaşılabilir. Sömürme süreci ise doğrudan yerel arama ile ilişkilendirilir. Bunun anlamı tüm arama uzayında global optimumun bulunduğu düşünülen daha küçük bir bölgeyi belirledikten sonra bu sınırlı bölge içinde daha hassas bir şekilde çözüme yakınsamayı sağlayacak arama teknikleri ile bölgenin sömürülmesidir. Diğer bir ifadeyle; bu süreçle amaçlanan şey çözümü hassas bir şekilde iyileştirmek ve büyük sıçramalardan kaçınmaya çalışmaktır. Sonuç olarak;

bir algoritmanın başarısı keşif ve sömürü süreçleri için içerdiği stratejilerin dengeli bir şekilde çalışabilmesine bağlıdır.

Bu çalışmada, yakın zamanda geliştirilmiş olan ve fizik-temelli algoritmalar kategorisinde değerlendirilebilecek başarılı bir meta-sezgisel olan Girdap Arama Algoritması (GAA) [14] üzerinde durulmaktadır. GAA algoritması karıştırılan sıvılarda meydana gelen dönüş hareketinden esinlenerek geliştirilmiştir. Bir merkez etrafında geniş bir eksende başlayıp, merkez noktasına doğru daralan alt eksenlerden oluşan girdap deseni anolojisi sayısal fonksiyon optimizasyonuna adapte edilerek özgün bir meta-sezgisel teknik elde edilmiştir. GAA tek-çözüm temelli meta-sezgisel bir algoritmadır. GAA'nın önerildiği çalışmada, algoritma kendisi gibi tek-çözüm temelli olan SA ve Desen Arama [15] algoritmalarının yanı sıra sezgisel optimizasyon alanının en trend algoritmaları sayılabilecek PSO, ABC ve PSO'nun başka bir versiyonu olan PSO2011 algoritmaları ile çok sayıda test fonksiyonu üzerinde çalıştırılarak karşılaştırılmış ve başarısı ispatlanmıştır. Diğer taraftan; matematiksel alt yapısı itibariyle iyi organize edilmiş hızlı çalışan, yeni ve yerli bir algoritma olmasına karşın bilinirliğinin az olmasından dolayı literatürde GAA ile yapılan çok az sayıda çalışma bulunmaktadır. Tüm meta-sezgisel algoritmalarda olduğu gibi NFL teorimi gereğince GAA algoritmasının da üzerinde çeşitli iyileştirme modifikasyonlarının yapılabileceği bazı yönleri bulunmaktadır.

Bu bağlamda; bu çalışmada tek merkez etrafında oluşturulan tek bir girdap ile arama uzayını araştıran GAA algoritmasının keşif sürecini iyileştirmek ve hızlandırmak amacıyla çok-merkezli bir strateji algoritmaya adapte edilmiştir. Önerilen yaklaşımda, algoritma arama uzayında 4 farklı bölgede belirlenen 5 farklı merkez noktası etrafında paralel ve bağımsız bir şekilde çalışan 5 girdap aracılığıyla keşif sürecini yürütür. Böylece algoritmanın arama uzayı üzerindeki kapsayıcılığı arttırılarak global optimumum bulunduğu bölgenin daha hızlı bir şekilde tespit edilmesi sağlanır. Bununla birlikte; her bir merkez için bir iyileştirilememe sayacı eklenerek sömürü sürecinin daha verimli geçmesi amacıyla algoritmaya eklenti yapılmıştır. Ayrıca, algoritmanın yürütüleceği toplam iterasyonun ilk yarısından sonra yüksek yerellikten daha etkin faydalanmak için girdap sayısı beşten ikiye düşürülür. Önerilen çok-merkezli girdap arama algoritması, algoritmanın orijinali kıyaslandığında performansının kayda değer ölçüde iyileştiği ortaya konulmuştur.

Bu bölümün devamında GAA algoritması ile ilgili literatür taramasına yer verilmekte ve çalışmalar bir alt bölüm şeklinde ele alınmaktadır. Çalışmanın kalan kısmı 3 bölümden oluşmaktadır. 2. bölümde, GAA algoritması ana hatlarıyla açıklanmaktadır. Daha sonra, önerilen çok-merkezli girdap arama algoritması 3.bölümde detaylı bir şekilde anlatılmaktadır. 4.bölümde, önerilen algoritmanın test fonksiyonları üzerinde çalıştırılması sonucunda bulunan istatiksel sonuçlar listelenmektedir. Son olarak; ulaşılan bulgular 5.bölümde açık bir şekilde ifade edilmektedir.

A. GAA İLE İLGİLİ ÇALIŞMALAR

Bu bölümde GAA algoritmasının önerildiği 2015 yılından bu zamana kadar algoritmanın kullanıldığı çalışmalarla ilgili literatür taramasına yer verilmiştir. Giriş bölümünde de vurgulandığı gibi tutarlı ve hızlı bir çalışma karakteristiği olmasına ve optimizasyon alanındaki diğer popüler algoritmalarla kıyaslandığında göze çarpan bir performansa sahip olmasına rağmen doğrudan kullanıldığı çok az sayıda çalışma vardır. Taramalar sonucunda, ikisi yazarın kendisine ait olmak üzere toplam 10 yayın çalışmasında GAA algoritmasının kullanıldığı görülmüştür.

(Doğan & Ölmez, 2015) [16] analog aktif filtre bileşen değerlerinin optimal seçimi için GAA algoritmasını kullanmışlardır. Elde edilen sonuçları PSO2011, ABC, DE ve Armoni Arama algoritmalarıyla karşılaştırmışlardır. GAA algoritmasının, literatürde önerilen iki farklı analog aktif filtre topolojisi üzerinde tanımlanan problem için literatürde en iyi olarak bilinen çözümleri bulduğu gösterilmiştir.

(Doğan, 2016) [17] birden fazla ekstremum noktası olan optimizasyon problemlerinde yerele takılma sorununu iyileştirmek için GAA algoritması için bir modifikasyon önermiştir. Bunun nedeni, GAA algoritmasının tek merkez etrafında çözüm üretmesi bir algoritmik basitlik ve hız kazandırmasına rağmen bu soruna neden olabilmektedir. Önerilen çalışmada sunulan fikir bu çalışmada geliştirilen yaklaşıma, merkez sayılarının artırılmasından dolayı benzerlik göstermesine rağmen; uygulanan stratejiler bakımından birbirlerinden ayrışmaktadırlar. Doğan bu çalışmasında m adet merkez noktasını rasgele oluşturmuş ve tüm merkezler için aynı yarıçap değeri ile komşu çözümlerin üretilmesini sağlamıştır. Ayrıca PSO algoritmasındakine benzer bir strateji ile her merkez noktası için bir pbest değeri tutarak sonraki iterasyon için bu değer ve sbest değerini kullanarak konum güncellemesi yapmıştır. Önerilen çalışma nispeten daha iyi sonuçlar vermesine rağmen rasgele üretilen merkezlerin aynı bölgelere denk düşme ihtimalinin giderilmemesinden dolayı yerele takılma sorununu tam olarak gidereememiştir.

(Özkış & Babalık, 2017) [18] tek amaçlı sayısal fonksiyon optimizasyonu için kullanılan GAA algoritmasını ilk kez çok-amaçlı optimizasyona uyarlayarak Çok-Amaçlı Girdap Arama (MOVS) Algoritmasını önerdiler. Bu amaçla; önerdikleri çalışmada, GAA algoritmasına NSGAII'de kullanılan hızlı-bastırılmamışlık-sıralama metodunu ve kalabalıklık-mesafesi hesaplamasını eklediler. Ayrıca Pareto-çözüm kümesindeki çözümler arasından ikili-turnuvayı kullanarak merkez noktası tayin etmişlerdir. Bunların yanı sıra, yerel optimumlara takılmayı engellemek amacıyla literatürden adapte edilen bir çaprazlama operatörü ilave edilirken; çeşitliliği güçlendirmek amacıyla da a parametresini rasgele (0,1) aralığında üreterek kullanmışlardır. Sonuç olarak; MOVS algoritmasının performansı çeşitli test problemleri üzerinde denemişler ve oldukça başarılı ve rekabetçi bir girdap arama temelli çok-amaçlı optimizasyon algoritması geliştirildiğini ortaya koymuşlardır.

(Koch et al., 2017) [19] global navigasyon uydu sistemleri önceden belirlenmiş belirli sayıda gözlem noktasını kullanır. Bu noktalar arasındaki mesafelerden tahminde bulunularak sistem oluşturulur. Bu bilgilerin önemi göz önüne alındığında, kesin bir ayarlama son derece gereklidir. Normal şartlarda aykırı değerler ve/veya sistematik hatalar olmadığı varsayılarak, en-küçük kareler metodu bu işlem için sıklıkla kullanılır. Ancak gerçek uygulamada aykırı değerler ortaya çıkabilir ve bu tahminlerin başarısız olmasına ve ağdaki birçok nokta üzerinde benzeri görülmemiş hatalara yol açmasına neden olabilir. Yapılmış olan çalışmada yazarlar; küçük veya büyük aykırı gözlem noktalarını tespit eden bir yaklaşım sunarak, Artıklık Kısıtlı En Küçük Kesilmiş Kareler (LTS-RC) olarak adlandırılan GAA algoritması temelli meta-sezgisel bir yöntem önerdiler.

(Aydin, Tezcan, Eke, & Taplamacioglu, 2017) [20] GAA algoritmasını kullanarak, sınırlı optimum güç akışı problemini (OPF) çözmeyi önerdiler. Kısmi yüklü ikinci kademe yakıt maliyeti ve valf noktası yükleri ile ikinci dereceli maliyet eğrisi test senaryoları, üretim limitleri, düğümlerdeki gerilimler, kademe ayarları gibi sistem kısıtlamaları göz önünde bulundurularak IEEE-30 bus test sisteminde çözmeyi denemişlerdir.

(Li, Niu, & Liu, 2018) [21] Lojik Self-Map Kaos ve Levy Uçuş stratejisine dayanan bir GAA algoritması önerdiler. I-VS olarak isimlendirdikleri algoritmayı yanma kazanı optimizasyonunda kullandılar. Yanma verimliliği modeli hızlı bir öğrenme ağı (FLN) ile oluşturuldu. I-VS algoritmasını FLN'yi optimize etmek için kullandılar.

(Ali, Qyyum, Qadeer, & Lee, 2018) [22] tek bir karışık soğutucu akışkan (SMR) doğal gaz sıvılaştırma işleminin optimizasyonu için GAA algoritmasını kullandılar. Gereken minimum enerji için karışık soğutucu akışkan debileri ve işlem çalıştırma basınçlarının optimal değerlerini GAA ile belirlemişlerdir. Ayrıca aynı optimizasyon işlemini GA ve PSO ile karşılaştırmışlar ve GAA'nın diğerlerinden daha üstün bir performans sergilediğini belirtmişlerdir.

(García, Amaya, & Correa, 2018) [23] bir mikrodalga ısıtma işlemi sırasında termal difüzyon ve iletkenliği tahmin etmek amacıyla Tepe Sinyal-Gürültü Oranını amaç fonksiyonu olarak kullandılar ve Spiral Optimizasyon Algoritması (SOA), GAA algoritması ve Ağırlık Cazibe Yöntemi (WAM)

algoritmalarını bu problemin çözümünde kullandılar. Üç algoritma da beklenen çözüme yakınsama sağlamış olmak birlikte GAA algoritmasının çok daha hızlı bir şekilde sonuca ulaştığını belirttiler.

(Chaniago et al., 2019) [24] Sera gazı emisyonlarının karbon dioksitin metanole dönüşmesi yoluyla azaltılması, enerji depolamak için ve bir yakıt kaynağı olarak kullanılabilecek değerli bir yan ürünün üretimi de dahil olmak üzere birkaç ikincil fayda sağlar. Bu bağlamda, kendi kendine iyileşme yardımlı ortak elektroliz ve MeOH sentezi süreci tanıtmışlardır. Aspen HYSYS V10 kullanılarak simule etmişler ve MATLAB ile HYSYS modelini birbirine bağlayarak GAA ile optimize etmişlerdir.

(Fathy, Elaziz, & Alharbi, 2020) [25] GAA ve DE algoritmalarının birlikte kullanıldığı yeni bir hibrid yaklaşım önerdiler. Bu yaklaşımda GAA'nın performansını arttırmak ve yerel optimumlara takılmasını önlemek amacıyla DE algoritmasını yerel arama olarak adapte ettiler. Bununla birlikte; bu hibrid algoritmayı proton değişim zarı yakıt hücresinin (PEMFC) optimal belirtilmemiş parametrelerinin tahmin edilmesinde kullandılar. Farklı PEMFC'ler (250Wstack, NedStack PS6, BCS 500-W ve SR-12 PEM 500W) üzerinde analizleri gerçekleştirerek sunulan yaklaşımın üstünlüğünü ve güvenilirliğini test ettiler.

II. GİRDAP ARAMA ALGORİTMASI

Girdap Arama Algoritması (GAA) Doğan ve Ölmez tarafından 2015 yılında sürekli optimizasyon problemlerinin çözümü için önerilen tek-çözüm temelli meta-sezgisel bir optimizasyon algoritmasıdır. Algoritma karıştırılan sıvılarda meydana gelen girdap deseninden esinlenilerek geliştirilmiştir [14]. Popülasyon-temelli evrimsel algoritmalar, arama uzayını araştırmaya rehberlik etmesi amacıyla tüm aday çözümleri kullanarak birden fazla aday çözümü geliştirir. Bu durumun aksine, tek-çözüm temelli algoritmalarda tek bir aday çözümün değiştirilmesi ve iyileştirilmesi söz konusudur [26]. Bu bağlamda; GAA'da arama uzayı girdap desenindeki gibi bir merkez noktası etrafında iteratif olarak daraltılır ve global optimuma ulaşmak hedeflenir. GAA algoritmasının sözde-kodu Şekil 1'de verilmiştir.

GİRDAP ARAMA ALGORİTMASI (GAA)

Başlangıç merkezini μ Denklem 1'i kullanarak hesapla
Başlangıç yarıçapını r Denklem 2'yi kullanarak hesapla
Bulunan en iyi çözümün uygunluk değeri $f(s_{best}) = inf$
$t = 0$
Repeat
    Gauss dağılımı ile μ merkezinde r yarıçaplı arama uzayı içinde komşu çözüm kümesi $C_t(s)$'yi random üret.
    $C_t(s)$ içinde sınırı aşan çözümleri Denklem 4'ü kullanarak sınır içine kaydır.
    $C_t(s)$ içindeki en iyi çözümü belirle $s' = C_t(s)$
    if $f(s') < f(s_{best})$
        $s_{best} = s'$
        $f(s_{best}) = f(s')$
    end
    Merkez noktasını bulunan en iyi çözüme kaydır $\mu_{t+1} = s_{best}$
    Yarıçap değeri Denklem 2 kullanılarak azaltılır.
    $t = t + 1$
Until $(t == MaxItr)$
Çıktı: Şu ana kadar bulunan en iyi değer olan s_{best}

Şekil 1. *GAA algoritmasının sözde-kodu*

GAA hesaplanan ilk merkez noktası etrafında rasgele komşu çözümlerin üretilmesiyle çalışmaya başlar. İki boyutlu bir optimizasyon problemi için girdap modeli yarıçapları gitgide küçülen iç içe geçmiş daireler şeklinde düşünülebilir. Bu durumda başlangıç adımında oluşturulacak en geniş çemberin merkezi arama uzayının orta noktası olacak şekilde hesaplanır. Başlangıç merkezi (μ_0) tüm karar değişkenleri için Denklem 1 kullanılarak belirlenir.

$$\mu_0^i = \frac{ustlimit_i + altlimit_i}{2} \quad (1)$$

Burada $i = 1,2,\dots,dim$ olmak üzere dim problemin boyutunu yani karar değişkenlerinin sayısını ifade eder. Çember yarıçapının hesaplanmasında ve iterasyonlar boyunca yarıçapın daraltılmasını sağlamak amacıyla ters tamamlanmamış gama fonksiyonundan faydalanılır. Yarıçapın değerini hesaplamak için kullanılan genel matematiksel ifade Denklem 2'de verilmiştir.

$$r_t = \sigma_0 . \frac{1}{x} . \Gamma(x, a_t) \quad (2)$$

Denklem 2'de *t* iterasyon indeksini gösterir. *x* ise $[0,1]$ aralığında rasgele bir sayıdır. GAA yazarlarının performans açısından önerdiği değer $x = 0.1$'dir. Γ ters gama fonksiyonunu ifade eder. Algoritmanın orijinalinde MATLAB'da bulunan gammaincinv fonksiyonu kullanılarak hesaplanmaktadır. Son olarak $a \in [0,1]$ olmak üzere şekil parametresi olarak bilinir. Arama işleminin kararlılığı her iterasyonda bu parametrenin güncellenmesi ile ayarlanır. Genel formülasyonu Denklem 3'te görülmektedir.

$$a_t = a_0 - \frac{t}{MaxItr} \quad (3)$$

İlk iterasyonda arama uzayını bütünüyle kapsayabilmek amacıyla $a_0 = 1$ olarak alınır. $MaxItr$ ise maksimum iterasyon sayısını ifade eder. Böylece μ_0 merskez noktasında r_0 yarıçapında bir çember içinde Gauss dağılımı kullanılarak n adet komşu çözüm üretilir. Daha sonra sınırları aşan çözümler Denklem 4'te olduğu gibi sınırlar içine kaydırılır.

$$s_k^i = \begin{cases} rand.\left(ustlimit^i - altlimit^i\right) + altlimit^i, & s_k^i < altlimit^i \\ s_k^i & altlimit^i \le s_k^i \le ustlimit^i \\ rand.\left(ustlimit^i - altlimit^i\right) + altlimit^i & altlimit^i,\ s_k^i > ustlimit^i \end{cases} \quad (4)$$

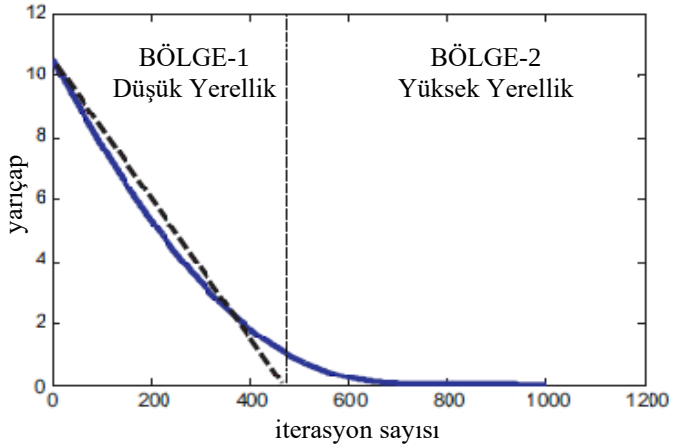
Denklem 4'de s_k^i üretilen k. komşu çözümün i. boyutunu göstermektedir. Burada $k = 1,2,\dots,n$ ve $i = 1,2,\dots,dim.$ $rand$ ise uniform olarak dağıtılmış rasgele bir sayıdır. İlk komşu çözüm vektörünün üretilmesinin ardından bulunan en iyi komşu çözüm s' hafızaya alınır ve ikinci çemberin merkezi olarak atanır. Daha sonra yarıçap boyutu azaltılır. Sonraki iterasyonda, yeni elde edilen merkez çevresinde yeni çözüm kümesi üretilir. Böylece yarıçap azaldıkça kısmen daha küçük bir bölgede aynı sayıda yeni çözümlerin üretilmesi sağlanmaktadır. İkinci iterasyonda üretilen en iyi çözüm önceki iterasyonda bulunan en iyi çözümden daha iyi ise yeni çözüm en iyi çözüm olarak hafızaya alınır. Daha sonra, üçüncü dairenin merkezi şimdiye kadar bulunan en iyi hafızaya alınmış çözüm olarak belirlenir. Bu işlem, sonlandırma koşulu yerine getirilinceye kadar tekrar eder.

III. GİRDAP ARAMA ALGORİTMASI İÇİN ÖNERİLEN MODİFİKASYON

Girdap Arama Algoritması arama uzayı içinde tek bir merkez çevresinde bir girdap oluşturarak global optimuma ulaşmayı hedefler. Bu amaçla, GAA algoritmasında ilk iterasyonlarda girdabın yarıçapı küçük adımlarla azaltılarak keşif sürecine odaklanılırken; toplam iterasyon sayısının yarısından sonraki bölümde yarıçap daha hızlı adımlarla küçültülerek yakınsamanın hızlanması ve sömürü sürecinin daha efektif işlemesi sağlanır. Bu durum Şekil 2'de grafik olarak ifade edilmektedir. Şekilde karar değişkenleri [-10,10] aralığında tanımlanmış bir problemin 1000 iterasyon çalıştırılması sonucunda iterasyona bağlı olarak yarıçaptaki değişim gösterilmiştir. Burada bölge-1 olarak isimlendirilen iterasyonlarda yarıçap yaklaşık olarak doğrusal biçimde değişir. Böylece algoritma bu bölgede zayıf bir

yerelliğe sahiptir. Diğer taraftan, bölge-2'de yarıçap önemli ölçüde azaltıldığı için algoritma yüksek yerellik ile çalışır.

Bu çalışmada, keşif sürecinin etkinliğini artırmaya odaklanıldı ve önerilen GAA algoritması "Çok-Merkezli Girdap Arama Algoritması (ÇM-GAA)" olarak isimlendirildi. Önerilen yaklaşımda, tek bir merkez etrafında tek bir girdap ile keşif yapmak yerine birden fazla merkez belirleyerek çoklu-girdapların eş-zamanlı çalışması arama uzayının daha kapsamlı bir şekilde taranmasını sağlamaktadır. Algoritmanın orijinali ile eşit sayıda amaç fonksiyon değerlendirmesini garantilemek amacıyla toplam üretilecek komşu çözümlerinin sayısı oluşturulacak girdap sayısına bölünerek her bir girdap için üretilecek komşu çözüm sayısı tanımlanır. Bununla birlikte, rasgele merkez noktalarının oluşturulması sakıncalıdır. Çünkü rasgelelik, aynı ya da birbirine yakın bölgeler tekrar tekrar taranırken global optimumum bulunduğu bölgenin göz ardı edilmesine yol açabilir. Dolayısıyla keşif sürecinin beklenenin aksine olumsuz şekilde ilerlemesine neden olabilir. Bu sebeple, önerilen yöntemin stratejisi arama uzayını birbiriyle örtüşmeyen alt uzaylara ayırmaya ve her bir alt uzayda farklı bir girdap oluşturmaya dayanır.



Şekil 2. *[10,10] aralığında tanımlanan bir problem için yarıçap değişimi [14]*

Bilindiği üzere; bir optimizasyon probleminde bir aday çözüm vektörü problemin karar değişkenleri ile tanımlanır ve her bir değişken kendi aralık değerlerine sahiptir. Bir aday çözüm vektörü ($\overrightarrow{cs}$)nin matematiksel gösterimi Denklem 5 ile ifade edilebilir.

$$i = \{1,2,\dots,dim\} \quad ve \quad x_i \epsilon [altlimit_i, ustlimit_i] \quad olmak\ üzere \quad \overrightarrow{cs} = \{x_1, x_2, \dots, x_{dim}\} \qquad (5)$$

Örtüşmeyen alt-uzaylar oluşturmak için seçilen karar değişkenlerinin aralıkları parçalara ayrılır. Böylece her bir parça için bir alt-uzay elde edilir. Seçilen boyut adedi oluşacak merkez sayısını belirler. Boyutların 2 parçaya ayrılacağı varsayıldığında merkez sayısı Denklem 6'daki gibi hesaplanır.

$$oluşacak\ merkez\ sayısı = 2^{seçilen\ boyut\ sayısı} \qquad (6)$$

Önerilen yöntemde, *dim* boyutlu bir optimizasyon problemi için iki boyut rasgele seçilir ve boyutların aralıkları iki parçaya ayrılır. Sonuç olarak 4 farklı alt arama uzayı için 4 merkez noktası elde edilir. Buna ilave olarak; algoritmanın orijinal merkez noktasının da kullanılmaya devam edilmesiyle toplamda 5 merkez noktası oluşturulur. Bu durumu örneklendirmek amacıyla; bir optimizasyon probleminde [-10,10] aralığında tanımlanan iki karar değişkenin seçilmesiyle oluşan merkez değerleri Tablo 1'de verilmiştir. Ayrıca ilk iterasyonda bu merkez noktaları ve aralık değerleri ile oluşturulan 5 çember ve algoritmanın orijinalinde oluşturulan başlangıç çemberi Şekil 3'te görülmektedir.

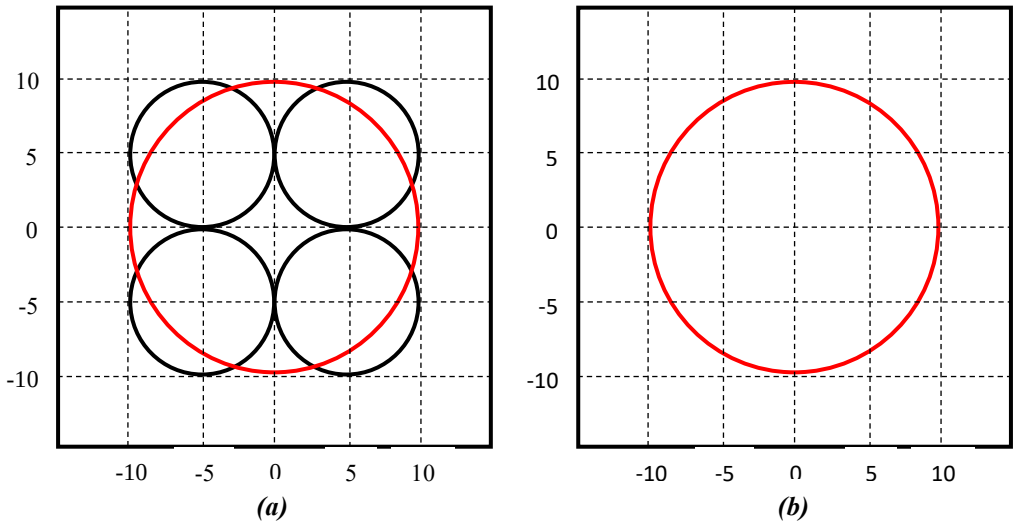
Tablo 1. *Temsili olarak [-10,10] Aralığında Seçilen 2 Boyut için Oluşturulan Çemberler*

	Boyut-1		**Boyut-2**		**Merkez** (μ)
	Alt Limit	**Üst Limit**	**Alt Limit**	**Üst Limit**	
Çember-1	-10	0	0	10	$\mu^1 = \{-5, 5\}$
Çember-2	0	10	0	10	$\mu^2 = \{5, 5\}$
Çember-3	-10	0	0	-10	$\mu^3 = \{-5, -5\}$
Çember-4	0	10	0	-10	$\mu^4 = \{5, -5\}$
Çember-5	-10	10	-10	10	$\mu^5 = \{0,0\}$

İki boyut seçilmesinin nedeni merkez sayılarının geometrik olarak artış göstermesi sonucunda alt arama uzayları için üretilecek komşu çözümlerin sayısını makul düzeyde tutabilmektir. Aksi halde seçilen boyut sayısı 3 olursa oluşacak merkez sayısı 5'ten 9'a; 4 olursa 17'ye yükselecektir. Bu durum umulanın aksine keşif işleminin yavaşlamasına neden olacağı için en uygun miktar 2 boyutun seçilmesidir. Seçilmeyen boyutların aralık değerleri değiştirilmeden olduğu gibi kullanılır.

ÇM-GAA algoritması rasgele belirlenen iki boyut için toplam 5 girdap oluşturarak çalışmaya başlar. Oluşturulan alt arama uzayları için Denklem 1 kullanılarak 5 merkez noktasının koordinatları ve Denklem 2 kullanılarak her bir girdabın yarıçapları algoritmanın orijinaline sadık kalınarak hesaplanır. Başlangıç merkezleri her bir girdaba ilk komşu çözüm olarak eklendikten sonra geriye kalan komşu çözümler Gauss dağılımıyla rasgele üretilir. Bu aşamadan sonra 5 girdap da birbirinden bağımsız ve paralel şekilde maksimum iterasyon sayısının yarısına kadar çalışır. Bu esnada her bir girdap için bir iyileştirilememe sayacı tutulur. Bu sayaç her iterasyon sonunda girdabın önceki iterasyondan daha iyi bir çözüme ulaşıp ulaşamadığını kontrol eder. Eğer daha iyi bir çözüm bulunmuşsa sıfırlanır aksi takdirde değeri bir arttırılır. Düşük yerellikle arama yapılan iterasyonlar boyunca 5 farklı merkezde keşif yapılması global optimuma yakınsama şansının artmasını sağlar.

Algoritma maksimum iterasyon sayısının yarısına ulaştığında en iyi uygunluk değerine ulaşan ile en düşük iyileştirilememe sayacına sahip olan 2 merkez noktası belirlenir. N bir iterasyonda üretilen toplam komşu çözüm sayısı olmak üzere; belirlenen bu iki merkez için üretilecek komşu çözüm sayısı N/5'ten N/2'ye çıkarılır. Diğer merkezlerin çalışması durdurulur. Geriye kalan iterasyonlar boyunca sadece bu iki merkez çevresindeki girdaplarda arama işlemine devam edilir. Bu sayede, sömürü sürecinin daha yüksek bir komşulukta ve global optimumun bulunma olasılığı en yüksek olan iki bölgede gerçekleşmesi sağlanır.



Şekil 3. (a) *Önerilen GAA ve* ***(b)*** *Orijinal GAA için ilk iterasyonda oluşturulan çemberlerin temsili gösterimi*

Maksimum iterasyon tamamlandığında algoritma çalışmasını durdurur ve sürdürülen 2 girdap arasından en iyi uygunluğa sahip olan merkez noktası algoritmanın sonucu olarak verilir. Önerilen ÇM-GAA algoritmasının detaylı sözde-kodu Şekil 4'te verilmiştir.

ÇOK-MERKEZLİ GİRDAP ARAMA ALGORİTMASI (ÇM-GAA)

dim ≥ 2 olmak üzere $[0, dim]$ aralığında rasgele 2 adet boyut indeksi belirle

Seçilen boyutların aralıklarını 2 parçaya ayır

$[altlimit, ustlimit] = \{\, [altlimit, (altlimit + ustlimit)/2] \cup [(altlimit + ustlimit)/2, ustlimit] \,\}$

Tüm arama uzayı ile birlikte 4 adet alt arama uzayı için toplam 5 başlangıç merkezini $\vec{\mu} = \{\mu^1, \mu^2, \mu^3, \mu^4, \mu^5\}$

Denklem 1'i kullanarak hesapla

Başlangıç yarıçaplarını $\{r^1, r^2, r^3, r^4, r^5\}$ Denklem 2'yi kullanarak hesapla

Üretilecek toplam komşu çözüm sayısı = N

for $i \leftarrow 1 \;\; to \; 5$ **do**
  i.girdap için üretilecek komşu çözüm sayısı = $N/5$
  i.girdap için bulunan en iyi çözüm $s_{best}^i = \mu^i$
  i.girdap için iyileştirilememe sayacı $trial^i = 0$
end
for $t \leftarrow 0 \;\; to \; MaxItr/2$ **do**
  for $i \leftarrow 1 \;\; to \; 5$ **do**
    $\mu^i = BirIterasyonGAA(\, i \,)$
  end
end

Uygunluk değeri en iyi olan merkez indeksini bul → $ind_{best\_fitness} = \min(\vec{\mu})$

$ind_{best\_fitness}$.girdap için üretilecek komşu çözüm sayısı = $N/2$

İyileştirilememe sayacı en küçük olan merkez indeksini bul → $ind_{best\_trial} = \min(\overrightarrow{trial})$

ind_{best_trial}.girdap için üretilecek komşu çözüm sayısı = $N/2$

for $t \leftarrow (MaxItr/2) + 1 \;\; to \; MaxItr$ **do**
    $\mu^{best\_fitness} = BirIterasyonGAA\,(ind_{best\_fitness})$
    $\mu^{best\_trial} = BirIterasyonGAA\,(ind_{best\_trial})$
end
ÇIKTI: $s_{best} = min(\mu^{best\_fitness}, \mu^{best\_trial})$

function $BirIterasyonGAA(\, i \,)$
  Gauss dağılımı ile μ^i merkezinde r^i yarıçaplı arama uzayı içinde komşu çözüm kümesi $C_t(s)$'yi random üret.
  $C_t(s)$ içinde sınırı aşan çözümleri Denklem 4'ü kullanarak sınır içine kaydır.
  $C_t(s)$ içindeki en iyi çözümü belirle $s' = C_t(s)$
  if $f(s') < f(s_{best}^i)$
    $s_{best}^i = s'$
    $f(s_{best}^i) = f(s')$
    $trial^i = 0$
  else
    $trial^i = trial^i + 1$
  end
  Merkez noktasını bulunan en iyi çözüme kaydır $\mu_{t+1}^i = s_{best}^i$
  r^i değeri Denklem 2 kullanılarak azaltılır.
end function

Şekil 4. *ÇM-GAA algoritmasının sözde-kodu*

IV. DENEYSEL SONUÇLAR VE TARTIŞMA

Bu çalışmada önerilen ÇM-GAA algoritması, 50 kıyaslama fonksiyonu üzerinde GAA algoritmasının orijinali ile kıyaslanarak test edildi. Çalışmada kullanılan test fonksiyonlarına ait detaylar Ek-1'de verilmiştir. Bu listedeki kıyaslama fonksiyonları unimodal, multimodal, normal, düzensiz, ayrılabilir, ayrılmaz ve çok boyutlu gibi birçok farklı problem türünü içermesi nedeniyle literatürde değerlendirme amacıyla sıklıkla kullanılmaktadır.

GAA ve önerilen ÇM-GAA algoritmaları için bir iterasyonda üretilecek komşu çözüm sayısı 50 olarak seçildi. Maksimum iterasyon sayısı 100 000 olarak ayarlandı. Böylece her iki algoritma için bir çalıştırmadaki toplam fonksiyon değerlendirme sayısı $50x100\,000 = 500\,000$ oldu. Bu eşitliği sağlayabilmek amacıyla ÇM-GAA algoritmasında ilk 50 000 iterasyonda kullanılan 5 merkezin her biri için 10'ar komşu çözüm üretilirken; sonraki 50 000 iterasyonda kullanılan 2 merkez için 25'er komşu çözüm üretildi. Bu şartlar altında, her iki algoritma için 50'şer bağımsız çalıştırma gerçekleştirildi. Bu çalıştırmalardan elde edilen sonuçlar için ortalama, standart sapma ve en iyi değerler bulunarak Tablo 2'de listelendi. Bu tabloda verilen istatistiksel değerler için 10^{-16}'nın altındaki değerler sıfır olarak kabul edildi.

Tablo 2. *GAA ve ÇM-GAA algoritmalarından 50 çalıştırma sonucunda elde edilen istatistiksel sonuçlar*

No	Fonksiyon	Min.		ÇM-GAA	GAA	
F1	Stepint	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F2	Step	0	ortalama	0	1,92	+
			standart sapma	0	1,70042012	
			en iyi	0	0	
F3	Sphere	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F4	SumSquares	0	ortalama	0	1,22594565e-15	+
			standart sapma	0	3,05629248e-15	
			en iyi	0	0	
F5	Quartic	0	ortalama	0,00621853389	0,00612155717	-
			standart sapma	0,00290741773	0,00358159586	
			en iyi	0,000451760545	0,00131398014	
F6	Beale	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F7	Easom	-1	ortalama	-1	-1	≡
			standart sapma	0	0	
			en iyi	-1	-1	
F8	Matyas	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F9	Colville	0	ortalama	2,14622915e-12	6,56872297e-16	-
			standart sapma	9,97054942e-12	2,48345043e-15	
			en iyi	0	0	
F10	Trid6	-50	ortalama	-50	-50	≡
			standart sapma	5,27050352e-14	5,99659719e-14	
			en iyi	-50	-50	
F11	Trid10	-210	ortalama	-210	-210	≡
			standart sapma	2,53339066e-12	1,25214929e-12	
			en iyi	-210	-210	

Tablo 2 (devam). *GAA ve ÇM-GAA algoritmalarından 50 çalıştırma sonucunda elde edilen istatistiksel sonuçlar*

No	Fonksiyon	Min.		ÇM-GAA	GAA	
F12	Zakharov	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F13	Powell	0	ortalama	0,00980399936	0,00712231078	-
			standart sapma	0,00606852221	0,00158418604	
			en iyi	0,00320410254	0,00338267596	
F14	Schwefel 2.22	0	ortalama	0	1,40485009e-10	+
			standart sapma	0	8,55542341e-10	
			en iyi	0	4,07648629e-15	
F15	Schwefel 1.2	0	ortalama	0	2,06959265e-11	+
			standart sapma	0	2,86470287e-11	
			en iyi	0	5,4648176e-13	
F16	Rosenbrock	0	ortalama	25,0024618	53,4005716	+
			standart sapma	2,57054922	70,1801164	
			en iyi	15,027714	21,1885789	
F17	Dixon-Price	0	ortalama	0,666673039	0,666982662	+
			standart sapma	7,62967052e-06	0,00174648506	
			en iyi	0,666666672	0,666666669	
F18	Foxholes	0,998	ortalama	0,998003838	0,998003838	≡
			standart sapma	8,97195691e-17	1,2285358e-16	
			en iyi	0,998003838	0,998003838	
F19	Branin	0,398	ortalama	0,397887358	0,397887358	≡
			standart sapma	3,36448384e-16	3,36448384e-16	
			en iyi	0,397887358	0,397887358	
F20	Bohachevsky1	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F21	Booth	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F22	Rastrigin	0	ortalama	0	73,6068441	+
			standart sapma	0	19,084938	
			en iyi	0	38,803373	
F23	Schwefel	-12569,5	ortalama	-11203,419	-10563,0111	+
			standart sapma	357,42362	546,784048	
			en iyi	-12056,1484	-11602,2239	
F24	Michalewicz2	-1,8013	ortalama	-1,80130341	-1,80130341	≡
			standart sapma	1,12149461e-15	1,12641838e-15	
			en iyi	-1,80130341	-1,80130341	

Tablo 2 (devam). *GAA ve ÇM-GAA algoritmalarından 50 çalıştırma sonucunda elde edilen istatistiksel sonuçlar*

No	Fonksiyon	Min.		ÇM-GAA	GAA	
F25	Michalewicz5	-4,6877	ortalama	-4,65694139	-4,56500451	+
			standart sapma	0,0399495466	0,111238746	
			en iyi	-4,68765818	-4,68765818	
F26	Michalewicz10	-9,6602	ortalama	-8,82742705	-8,5865565	+
			standart sapma	0,323877399	0,600823216	
			en iyi	-9,55150296	-9,51014955	
F27	Schaffer	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F28	SixHumpCamelBack	-1,03163	ortalama	-1,03162845	-1,03162845	≡
			standart sapma	2,83718189e-16	2,99252088e-16	
			en iyi	-1,03162845	-1,03162845	
F29	Bohachevsky2	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F30	Bohachevsky3	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F31	Shubert	-186,73	ortalama	-186,730909	-186,730909	≡
			standart sapma	3,06541715e-14	2,72369461e-14	
			en iyi	-186,730909	-186,730909	
F32	GoldStein-Price	3	ortalama	3	3	≡
			standart sapma	1,28302121e-15	1,3841261e-15	
			en iyi	3	3	
F33	Kowalik	0,00031	ortalama	0,000307485988	0,000327773036	+
			standart sapma	1,12662949e-18	0,000143451091	
			en iyi	0,000307485988	0,000307485988	
F34	Shekel5	-10,15	ortalama	-10,1531997	-10,1531997	≡
			standart sapma	8,08870555e-15	7,9075471e-15	
			en iyi	-10,1531997	-10,1531997	
F35	Shekel7	-10,4	ortalama	-10,4029406	-10,4029406	≡
			standart sapma	7,60872729e-15	7,84212677e-15	
			en iyi	-10,4029406	-10,4029406	
F36	Shekel10	-10,53	ortalama	-10,5364098	-10,5364098	≡
			standart sapma	9,36527914e-15	9,51535143e-15	
			en iyi	-10,5364098	-10,5364098	
F37	Perm	0	ortalama	0,0023136376	0,00307878852	+
			standart sapma	0,00252232467	0,00271293737	
			en iyi	4,37465803e-06	6,76318652e-11	

Tablo 2 (devam). *GAA ve ÇM-GAA algoritmalarından 50 çalıştırma sonucunda elde edilen istatistiksel sonuçlar*

No	Fonksiyon	Min.		ÇM-GAA	GAA	
F38	PowerSum	0	ortalama	0	0,000209571452	+
			standart sapma	0	0,000170687795	
			en iyi	0	7,58153506e-11	
F39	Hartman3	-3,86	ortalama	-3,86278215	-3,86278215	≡
			standart sapma	2,98039172e-15	3,03989178e-15	
			en iyi	-3,86278215	-3,86278215	
F40	Hartman6	-3,32	ortalama	-3,32236801	-3,26753321	-
			standart sapma	1,06157626e-15	0,0600151937	
			en iyi	-3,32236801	-3,32236801	
F41	Griewank	0	ortalama	0	0,0174274534	+
			standart sapma	0	0,0108450608	
			en iyi	0	0	
F42	Ackley	0	ortalama	8,8817842e-16	0,0268084258	+
			standart sapma	0	0,189564196	
			en iyi	8,8817842e-16	1,50990331e-14	
F43	Penalized	0	ortalama	0,0124402824	11,2291246	+
			standart sapma	0,0340304675	19,619814	
			en iyi	0	0	
F44	Penalized2	0	ortalama	0,0104547361	0	-
			standart sapma	0,0382442742	0	
			en iyi	0	0	
F45	Langerman2	-1,08	ortalama	-1,08093844	-1,08093844	≡
			standart sapma	7,28194665e-16	7,23342291e-16	
			en iyi	-1,08093844	-1,08093844	
F46	Langerman5	-1,5	ortalama	-1,49999922	-1,4678992	+
			standart sapma	1,34579354e-16	0,128345496	
			en iyi	-1,49999922	-1,49999922	
F47	Langerman10	NA	ortalama	-0,809353621	-0,800737021	+
			standart sapma	0,247146464	0,371353549	
			en iyi	-1,5	-1,5	
F48	Fletcher Powell2	0	ortalama	0	0	≡
			standart sapma	0	0	
			en iyi	0	0	
F49	Fletcher Powell5	0	ortalama	0,000511939849	0,00217775485	+
			standart sapma	0,00360921076	0,00834789194	
			en iyi	0	0	
F50	Fletcher Powell10	0	ortalama	2,12508851	11,071093	+
			standart sapma	4,18329857	31,4040572	
			en iyi	5,5350981e-13	1,91636414e-13	

Tablo 2'de verilen son sütun, önerilen yöntemin GAA algoritmasının orijinali karşısındaki başarı durumunu göstermektedir. MÇ-GAA daha iyi ise "+"; daha kötü ise "-"; yaklaşık olarak denk ise "≡" işareti ile gösterilmiştir. Bu kıyaslama ortalama değerleri baz alınarak yapılmıştır. Bu sonuçlara göre 50 test fonksiyonundan 25'inde algoritmalar denk yani yaklaşık olarak aynı ortalama değerlerini elde ederlerken; 20 test fonksiyonunda önerilen ÇM-GAA algoritması GAA algoritmasından daha başarılı sonuçlar elde etmiştir. Geriye kalan 5 test fonksiyonunda (F5, F9, F13, F40 ve F44) ortalama değerler açısından algoritmanın orijinali daha iyi sonuç almıştır. Önerilen yöntemin geride kaldığı fonksiyonların listesi Tablo-3'te daha açık bir şekilde görülmektedir. Tablo-3 incelendiğinde görülmektedir ki; F5 ve F13 fonksiyonlarında elde edilen en iyi değerlere göre önerilen yöntem daha iyi sonuçlara ulaşırken; geriye kalan F9, F40 ve F44 fonksiyonları için ÇM-GAA algoritması global optimum değerlere tam olarak ulaşabilmiştir. Sonuç olarak; sunulan istatiksel kıyaslamalar, test fonksiyonlarında yüzde yüze yakın bir kapsama ile önerilen çok-merkezli yaklaşımın girdap arama algoritmasının keşif sürecini iyileştirdiğini ve performansının artmasını sağladığını göstermiştir.

Tablo 3. *Önerilen algoritmanın ortalama değer bazında geride kaldığı fonksiyonlar*

	Min	**Algoritma**	**Ortalama**	**En iyi**
F5	0	ÇM-GAA	0,00621853389	0,000451760545
		GAA	0,00612155717	0,00131398014
F9	0	ÇM-GAA	2,14622915e-12	0
		GAA	6,56872297e-16	0
F13	0	ÇM-GAA	0,00980399936	0,00320410254
		GAA	0,00712231078	0,00338267596
F40	-3,32	ÇM-GAA	-3,32236801	-3,32236801
		GAA	-3,26753321	-3,32236801
F44	0	ÇM-GAA	0,0104547361	0
		GAA	0	0

V. SONUÇLAR

Bu çalışmada, yakın zamanda sayısal fonksiyon optimizasyonu amacıyla geliştirilmiş olan tek-çözüm temelli Girdap Arama Algoritmasına odaklanıldı ve bu algoritma için yeni bir modifikasyon önerildi. GAA algoritması adaptif adım-boyutu ayarlaması yaparak iteratif olarak arama uzayını daraltır. Böylece, keşif ve sömürü süreçleri arasındaki dengeyi kurmayı hedefleyerek arama işlemini gerçekleştirir. GAA algoritması, matematiksel olarak iyi organize edilmiş, hızlı ve kararlı bir çalışma karakteristiğine sahip olan bir meta-sezgisel optimizasyon algoritmasıdır. Buna karşın yeni yeni keşfedilmeye başlanmasından dolayı GAA ile ilgili henüz çok az sayıda çalışma bulunmaktadır.

Yerel optimumları fazla olan optimizasyon problemlerinde tek bir merkez etrafında komşu çözümlerin üretilmesi, global optimumun atlanarak algoritmanın yerel ektremumlara takılmasına neden olabilmektedir. Bu sorunun üstesinden gelmek ve GAA algoritmasının arama uzayı içindeki komşu çözüm çeşitliliğini arttırmak amacıyla iki aşamalı olarak algoritmanın merkez sayısı arttırıldı. İlk aşamada; arama uzayı birbiriyle kesişmeyen 4 alt bölgeye ayrıldı ve her bir alt bölge için bir merkez noktası belirlendi. Algoritmanın orijinal merkez noktasıyla birlikte toplamda 5 merkez noktası etrafında komşu çözümlerin üretilmesi sağlandı. Tüm merkezler için ayrı ayrı oluşturulan girdaplar, birbirinden bağımsız bir şekilde maksimum iterasyonun ilk yarısına kadar; diğer bir ifadeyle düşük yerellikle çalıştırılan iterasyonlar boyunca, paralel şekilde çalıştırıldı. Ayrıca bu aşamada, her bir merkez için bir iyileştirilememe sayacı tutuldu. İkinci aşamada; en iyi uygunluk değerine sahip olan merkez ile en düşük iyileştirilememe sayacı değerine sahip olan merkez belirlendi. Maksimum iterasyona kadar yani yüksek yerellikle çalıştırılan iterasyonlar boyunca sadece bu iki merkez etrafında komşu çözümler üretilerek algoritmanın çalışmasına devam etmesi sağlandı. Sonuç olarak, bu 2 girdaptan daha iyi uygunluk

değerine ulaşan merkez noktası algoritmanın çözümü olarak verildi. Önerilen Çok-Merkezli GAA algoritması (ÇM-GAA), 50 test fonksiyonu üzerinde 50'şer defa bağımsız şekilde çalıştırılarak test edildi ve algoritmanın orijinal versiyonuyla karşılaştırıldı. Elde edilen istatistiksel sonuçlar göstermiştir ki; önerilen çok merkezli yaklaşım, hemen hemen tüm fonksiyonlarda iyileştirme sağlayarak GAA algoritmasının performansını arttırmayı başarmıştır.

VI. KAYNAKLAR

[1] K. Deb, *Optimization for Engineering Design: Algorithms and Examples*, 2nd ed., New Delhi, India: PHI Learning Private Limited., 2012, ch. 1, pp. 1-42.

[2] P. Liu and J. Liu, "Multi-leader PSO (MLPSO): a new PSO variant for solving global optimization problems," *Applied Soft Computing*, vol. 61, no. 1, pp. 256-263, 2017, doi: 10.1016/j.asoc.2017.08.022.

[3] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1997, doi: 10.1109/4235.585893.

[4] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2, pp. 95-99, 1988, doi: 10.1023/a:1022602019183.

[5] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997, doi: 10.1023/a:1008202821328.

[6] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *International Conference on Neural Networks*, Perth, WA, Australia, 1995, pp. 1942-1948, doi: 10.1109/ICNN.1995.488968.

[7] D. Karaboga. (2019, 25 Ekim). *An idea based on honey bee swarm for numerical optimization* [Online]. Erişim: http://abc.erciyes.edu.tr/publ.htm.

[8] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997, doi: 10.1109/4235.585892.

[9] X. Yang and D. Suash, "Cuckoo search via lévy flights," in *World Congress on Nature & Biologically Inspired Computing*, 2009, pp. 210-214, doi: 10.1109/NABIC.2009.5393690.

[10] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no: 4598, pp. 671-680, 1983, doi: 10.1126/science.220.4598.671.

[11] E. Rashedi, H. Nezamabadi-pour and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232-2248, 2009, doi: 10.1016/j.ins.2009.03.004.

[12] A. Kaveh and T. Bakhshpoori, "Water evaporation optimization: a novel physically inspired optimization algorithm," *Computers & Structures*, vol. 167, no. 1, pp. 69-85, 2016, doi: 10.1016/j.compstruc.2016.01.008.

[13] A. Kaveh and A. Dadras, "A novel meta-heuristic optimization algorithm: thermal exchange optimization," *Advances in Engineering Software*, vol. 110, pp. 69-84, 2017, doi: 10.1016/j.advengsoft.2017.03.014.

[14] B. Dogan and T. Olmez, "A new metaheuristic for numerical function optimization: Vortex search algorithm," *Information Sciences*, vol. 293, pp. 125-145, 2015, doi: 10.1016/j.ins.2014.08.053.

[15] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *Journal of the ACM*, vol. 8, no. 2, pp. 212-229, 1961, doi: 10.1145/321062.321069.

[16] B. Doğan and T. Ölmez, "Vortex search algorithm for the analog active filter component selection problem," *AEU - International Journal of Electronics and Communications*, vol. 69, no. 9, pp. 1243-1253, 2015, doi: 10.1016/j.aeue.2015.05.005.

[17] B. Doğan, "A modified vortex search algorithm for numerical function optimization," *International Journal of Artificial Intelligence and Applications*, vol. 7, no. 3, pp. 37-54, 2016, doi: 10.5121/ijaia.2016.7304.

[18] A. Özkış and A. Babalık, "A novel metaheuristic for multi-objective optimization problems: The multi-objective vortex search algorithm," *Information Sciences*, vol. 402, pp. 124-148, 2017, doi: 10.1016/j.ins.2017.03.026.

[19] I. É. Koch *et al.*, "Least trimmed squares estimator with redundancy constraint for outlier detection in gnss networks," *Expert Systems with Applications*, vol. 88, pp. 230-237, 2017, doi: 10.1016/j.eswa.2017.07.009.

[20] O. Aydin, S. S. Tezcan, I. Eke and M. C. Taplamacioglu, "Solving the optimal power flow quadratic cost functions using vortex search algorithm," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 239-244, 2017, doi: 10.1016/j.ifacol.2017.08.040.

[21] X. Li, P. Niu and J. Liu, "Combustion optimization of a boiler based on the chaos and lévy flight vortex search algorithm," *Applied Mathematical Modelling*, vol. 58, pp. 3-18, 2018, doi: 10.1016/j.apm.2018.01.043.

[22] W. Ali, M. A. Qyyum, K. Qadeer and M. Lee, "Energy Optimization for single mixed refrigerant natural gas liquefaction process using the metaheuristic vortex search algorithm," *Applied Thermal Engineering*, vol. 129, pp. 782-791, 2018, doi: 10.1016/j.applthermaleng.2017.10.078.

[23] E. García, I. Amaya and R. Correa, "Estimation of thermal properties of a solid sample during a microwave heating process," *Applied Thermal Engineering*, vol. 129, pp. 587-595, 2018, doi: 10.1016/j.applthermaleng.2017.10.037.

[24] Y. D. Chaniago, M. A. Qyyum, R. Andika, W. Ali, K. Qadeer and M. Lee, "Self-Recuperative high temperature co-electrolysis-based methanol production with vortex search-based exergy efficiency enhancement," *Journal of Cleaner Production*, vol. 239, pp. 118029, 2019, doi: 10.1016/j.jclepro.2019.118029.

[25] A. Fathy, M. A. Elaziz and A. G. Alharbi, "A novel approach based on hybrid vortex search algorithm and differential evolution for identifying the optimal parameters of pem fuel cell," *Renewable Energy*, vol. 146, pp. 1833-1845, 2020, doi: 10.1016/j.renene.2019.08.046.

[26] E.-G. Talbi, *Metaheuristics: From Design to Implementation*, 1st. ed., New Jersey, USA: John Wiley & Sons, 2009, ch. 2, pp. 87-190.