**Exploring the Effect of Distribution Methods on Meta-Heuristic Searching Process**

Hamdi Tolga KAHRAMAN1, Sefa ARAS2, Ugur GUVENC3, Yusuf SÖNMEZ4

1,2Yazılım Mühendisliği Bölümü, Karadeniz Teknik Üniversitesi, Trabzon, Türkiye

[htolgakahraman@ktu.edu.tr](mailto:htolgakahraman@ktu.edu.tr), [sefaaras@ktu.edu.tr](mailto:sefaaras@ktu.edu.tr)

3 Elektrik-Elektronik Mühendisliği Bölümü, Düzce Üniversitesi, Düzce, Türkiye

ugurguvenc@duzce.edu.tr

4 Gazi Meslek Yüksekokulu, Gazi Üniversitesi, Ankara, Türkiye

ysonmez@gazi.edu.tr

**Özet**

Bu çalışmada, çözüm adaylarının problem uzayındaki dağılımlarının meta-sezgisel arama sürecine ve algoritmaların performansına etkisi araştırılmaktadır. Bu amaçla çözüm adayları, random ve gauss (normal) dağılımları ile yaratılmaktadır. Algoritmaların her iki dağılım tipi için arama performansları ayrı-ayrı ölçülmektedir. Algoritmaların performansları, en popüler olan ve yaygın olarak kullanılan benchmark problemleri üzerinden test edilmiştir. Deneysel çalışmalar son dönemlerin en etkili meta-sezgisel arama algoritmaları üzerinde yürütülmüştür. Algoritmaların arama performanslarının dağılım yöntemine bağlı olarak önemli ölçüde değiştiği görülmüştür. Hatta algoritmaların orijinal versiyonlarında kullanılan dağılım yöntemlerinden daha başarılı sonuçlar elde edilmiştir. Algoritmaların komşuluk araması, yerel minimum tuzaklarından kurtulma ve arama hızı açısından yetenekleri ortaya çıkarılmıştır.

**Key words:** Gaussian distribution, Meta-heuristic searching, Benchmark problem, Intensification, Diversification

# Giriş

Meta-sezgisel arama sürecinde doğadaki başarı öyküleri algoritmalarla taklit edilmektedir. 1950 li yıllardan bu yana çok sayıda meta-sezgisel algoritma geliştirilmiştir. Son 10 yıl içinde ise geçmiş 50 yıldakine yakın sayıda algoritma literatüre kazandırılmıştır. Bu durum meta-sezgisel algoritmalara ilginin giderek arttığının en önemli göstergesidir. Bu ilginin başlıca sebebi ise optimizasyon, veri madenciliği ve bilgi güvenliği gibi alanlardaki çalışmaların yaygınlaşması ve önem kazanmasıdır. Özellikle optimizasyon her alanda yaygınlaşmakta, problemlerin karmaşıklık düzeyleri artmakta ve maliyetleri azaltmanın yolları aranmaktadır. Meta-sezgisel algoritmalar optimizasyon problemlerinin çözümünde etkili sonuçlar üretmektedirler. Tüm bu faktörler meta-sezgisel arama algoritmalarının son yıllarda neden bu kadar önem kazandığını açıklamaktadır.

Algoritmaların arama performanslarının birçok parametreye bağlı olması ve arama sürecindeki belirsizlikler meta-sezgisel yöntemlerin başlıca handikaplarıdır [1-3]. Bunun yanında algoritmaların arama sürecindeki handikaplarını ve problemlerini ortaya çıkarmak ciddi bir zorluktur (challenge). Algoritmaların geliştirilmesinde ve doğrulanmasında yaygın olarak benchmark problemleri kullanılmaktadır. Algoritmaların testleri ise gerek benchmark problemleri gerekse de farklı disiplinlere ait optimizasyon problemleri üzerinde gerçekleştirilmektedir [4-9]. Tüm bu çalışmalara rağmen meta-sezgisel algoritmaların performansları ya da arama sürecindeki başarıları hakkında kesin bir yargıya varmak mümkün değildir. Yani makalelerde verilen deneysel çalışma sonuçlarını inceleyerek meta-sezgisel algoritmaların arama sürecindeki yeteneklerini, handikaplarını ve özelliklerini yeterince anlamak/gözlemlemek mümkün değildir. Ayrıca makalelerde verilen sonuçlar araştırmacılar için yanıltıcı da olabilmektedir. Bu problemin temelinde meta-sezgisel algoritmaların test ve doğrulama standartlarının geliştirilmemiş/tanımlanmamış olması yatmaktadır. Dolayısıyla, araştırmacılar için literatüre yeni kazandırılan bir algoritma hakkında ciddi belirsizlikler mevcuttur. Araştırmacılar yeni geliştirilen bir algoritmanın kendileri açısından önemini keşfetmek için onu kendi problemlerine tatbik etmek zorundadırlar. Tüm bu belirsizlikler ve problemler hem araştırmacıların hem de algoritma geliştiricilerin işini zorlaştırmakta ve zaman kaybına yol açmaktadır.

Bu çalışmada meta-sezgisel arama sürecinin önemli bir adımı olan çözüm uzayının yaratılma şeklinin algoritmanın arama performansına etkileri incelenmektedir.

# Meta-Heuristic Algorithms (MHAs)

Büyük ve karmaşık bir problem uzayında arama yapmanın etkili bir yolu MHAs’leri kullanmaktır. MHAs’ler arama sürecinde iki temel problemle karşılaşmaktadırlar. Bu problemler, yerel minimum tuzaklarına yakalanmak (çeşitliliği sağlayamamak) ve komşuluk aramasını hassas bir şekilde gerçekleştirememektir. MHAs bu problemleri aşmayı sağlayan composed of stochastic components. Bu bileşenler/operatörler algoritmaya özgü karakteristiklere sahiptirler. Dolayısıyla MHAs lerin arama sürecindeki davranışları ve performansları birbirlerinden farklı olmaktadır [10-13].

Algoritmaların stochastic bileşenlerinin yetenekleri ve özellikleri farklı olsa da bir meta-sezgisel arama süreci temel olarak aynı adımlardan oluşur. Bir optimizasyon probleminin MHAs kullanılarak çözümü Algortima 1’de verilmektedir.

**Algoritma 1.** Meta-sezgisel arama sürecinin temel adımları

|  |
| --- |
| 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 (Stochastic arama)    * Komşuluk Araması    * Çeşitliliğin Sağlanması    * Çözüm adayı setinin güncellenmesi 5. Sonlandırma kriteri sağlandı mı?    * Hayır (Adım iv’e dön)    * Evet (arama sürecini sonlandır ve en iyi çözüm adayını kaydet) |

Algoritma 1’de verilen (i, ii, iii ve v) numaralı adımlar bütün MHAs ler için aynıdır. (iv) numaralı adım ise tüm MHAs için farklıdır. (iv) numaralı adımda arama algoritmasına özgü operatörler/işlemler uygulanmaktadır. Arama sürecinin başarısı bu operatörlerin yeteneklerine bağlıdır. (iv) numaralı adımda verilen çözüm adayları topluluğunun güncellenmesinde temel olarak iki farklı yöntem kullanılmaktadır. Bunlar çözüm adaylarının arama uzayına normal/gauss dağılımı ile yerleştirilmesi ve rastgele (random) yerleştirilmesidir. MHAs lerle ilgili çalışmalar incelendiğinde bu dağılım tiplerinden biriyle algoritmaların geliştirildiği ve test edildiği görülmektedir [4-5, 10-15]. Oysaki bu iki dağılım şekli MHAs lerin arama performansını değiştirebilmektedir. Dolayısıyla MHAs’lerde çözüm adayı toplulukları her iki dağılım tipi kullanılarak yaratılmalı ve bunun etkileri incelenmelidir.

## Creating and Updating of Solution Candidates

MHAs m-sayıda çözüm adayından oluşan bir topluluktur. Çözüm adaylarının tasarımı probleme bağlı olarak gerçekleştirilir. Dolayısıyla n-boyutlu bir optimizasyon probleminin maliyetine etki eden parametreler <x1, x2, x3,…, xn> seti ile gösterilebilir. Bu parametrelerin her biri için arama uzayı sınır değerleri ise şeklinde tanımlanır. Problemin maliyet fonksiyonu ise şeklinde temsil edilebilir. Bu tanımlamalara bağlı olarak çözüm adayları topluluğu *S*-matrisi ile eşitlik-1’de verildiği gibi temsil edilir.

(1)

vektöründeki her bir elemanı verilen sınır değerleri arasında rastgele veya gauss yöntemleriyle yaratılabilir. Bu iki dağılım tipi ayrıca Algoritma 1’de verilen (iv) numaralı adımda çözüm adaylarını güncelleme sürecinde de yoğun bir şekilde kullanılmaktadır. Dolayısıyla dağılım tipi çözüm adayları topluluğunun (*S*) yaratılmasında ve güncellenmesinde önemli bir faktördür/yere sahiptir.

### Distribution Methods

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 [10-12]. Normal dağılımın matematiksel ifadesi Eşitlik-2’de verilmektedir.

(2)

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 1’de gösterilmektedir.

**Fig. 1.** Gauss Dağılım

Rastgele dağılım yöntemi kullanılarak probleme ait parametrelerin her biri Eşitlik 3’de verilmektedir.

(3)

Rastgele dağılımda değerler [-1, 1] aralığında üretilmektedir. Elde edilen değerlerin yoğunluk grafiği Şekil 2’de gösterilmektedir.

**Fig. 2.** Rastgele Dağılım

Rastgele ve Gauss dağılım yöntemiyle ayrı ayrı 1000000 (bir milyon) değer üretilmiştir. Grafiklerde gösterim yapılırken ölçek olarak 0,1 aralık belirlenmiştir. Grafiklerde -0,9 ile **-1 < değer < -0,9** aralığı ifade edilmektedir. Aynı şekilde 1 ile **0,9 < değer < 1** aralığı ifade edilmektedir. Bu şekilde 0,1 birimlik aralıklara etiketler verilerek grafikler oluşturulmuştur. Dikey eksende belirlenen aralıkta üretilen değerlerin adedi verilmektedir.

Rastgele ve Gauss dağılımının ortak olarak gösterimi Şekil 3 ile verilmektedir.

**Fig. 2.** Rastgele ve Gauss Dağılım

## Benchmark Problems

Benchmarklar, algoritmaların arama ve optimizasyon performanslarını test etmek için kullanılan fonksiyonlardır. Optimization problems are basically classified into two-categories of test functions as continuous and discrete. Practical discrete optimization problems are also known as np-hard “non-deterministic polynomial time” [17]. They rarely are used to test MHAs. The continuous test functions are commonly used in test process of MHAs. Molga [18] classified the continuous test functions mainly unimodal and multimodal. In this study, we used the unimodal and multimodal types of benchmarks in detailed given in reference study [4].

# Experimental Study

Deneysel çalışmalar aşağıda verilen şartlar altında gerçekleştirilmiştir:

1. Test Problemleri: 26 adet benchmark problemi
2. Algoritmalar: Güncel meta-sezgisel yöntemler SOS [4], LSA [5] ve CSA [6]
3. Çözüm adaylarının arama uzayına dağılımı: gauss dağılımı ve rastgele dağılım
4. Deneysel çalışmaların geçerliliği: her bir deney 100 defa tekrarlanmıştır. Deneyler arası standart sapmalar, deneylerin ortalaması, en kötü ve en iyi sonuçlar, ortalama hesaplama süresi
5. Toplam deney sayısı: Benchmark Sayısı\* Algoritma Sayısı\* Arama Uzayına Dağılım Tipi Sayısı\*Deneylerin tekrar sayısı (26\*3\*2\*100=15600)

Tablo 1’de 156 deney sonucu verilmektedir. Her bir deney 100’kez tekrarlanmış ve elde edilen sonuçların ortalaması “mean” sütununda, deneylerden elde edilen en iyi sonuç “best” sütununda, en kötü sonuç “worst” sütununda verilmektedir. Algoritmalarda rastgele dağılım kullanıldığında tabloda SOS, LSA, CSA ile gösterilmiştir. Gauss dağılımı ise SOS-G, LSA-G ve CSA-G ile gösterilmiştir. Tablonun ilk sütununda benchmark probleminin adı, ikinci sütunda global minimum noktası verilmektedir. Yeşil renkli hücreler problemin çözüldüğünü göstermektedir. Mavi renkli hücreler gauss dağılımının random dağılımdan daha başarılı olduğu durumları göstermektedir. Kırmızı renkli hücreler ise random dağılımın gauss dağılımdan daha başarılı sonuç ürettiği durumları göstermektedir.

## The Comparisons of Performances of Algorithms on Benchmark Problems

While the SOS used random distribution to create the solution candidates the LSA and CSA used the Gauss distribution to create them in the original version of algorithms. Deneysel çalışmalarda toplam altı farklı algoritma kombinasyonun performansı ölçülmüştür. Bunlar SOS (use random distribution), SOS-G (use gauss distribution), LSA, LSA-G (LSA with gauss), CSA ve CSA-G (CSA with gauss) dir.

Table 1. The searching performances of meta-heuristic algorithms depending on gauss and random distribution



Red: Random distribution finds better solution than Gauss distribution

Blue: Gauss distribution finds better solution than Random distribution

Green: Benchmark solved by algorithm (with random or gauss distribution)

Gauss ve Random dağılımın arama sürecine olan etkilerini ölçmek için beş farklı parametrenin değeri kaydedilmiştir. Bunlar:

1. Deneysel çalışma sonucunda elde edilen: en iyi ve en kötü çözüm adayları (best, worst),
2. Tekrar edilen deneylere ait: en iyi çözümlerin ortalaması (mean), deneyler arası standart sapma (Std.D.) ve ortalama hesaplama süresidir (time).

Tablo 1’de verilen standart sapma değerleri incelenerek deneysel çalışmanın geçerliliği doğrulanabilir. Standart sapmanın sıfır olması yada sıfıra çok yakın değerler alması deneyler arasındaki tutarlılığı gösterir. Tablo 1’de verilen standart sapma değerleri çok düşüktür ve deneysel çalışmalar arası kararlılık ve tutarlılık yüksektir.

Tablo 2’de, dağılım yöntemine bağlı olarak algoritmaların performansları karşılaştırılmaktadır. Bu karşılaştırmalar incelendiğinde SOS algoritması gauss dağılımı kullandığında daha başarılı bir arama gerçekleştirmektedir. SOS-G, 7 adet benchmark probleminde SOS’dan daha iyi sonuç elde etmeyi başarmıştır. Buna karşın SOS algoritması sadece 1 adet benhmark probleminde SOS-G den daha iyi arama yapmıştır. LSA ve CSA algoritmaları ise random dağılımı kullandıklarında daha başarılı arama gerçekleştirmişlerdir. Tablo 2’de verilen sonuçlar, dağılım yöntemine bağlı olarak algoritmaların arama başarısının değiştiğini açıkça göstermektedir. Bunun yanında arama sürecindeki başarı, dağılım yöntemine bağlı olduğu kadar algoritmanın arama stratejisine ve karakteristik özelliklerine de bağlıdır. Öyleki SOS algoritmasında Gauss yöntemi daha başarılı bir arama sağlarken LSA ve CSA’da random dağılım daha başarılı aramayı sağlamıştır.

Table 2. the numbers of bechmarks solved by meta-heuristic methods

|  |  |  |  |
| --- | --- | --- | --- |
|  | Red (Random) | Blue (Gauss) | Green |
| SOS, SOS-G | 1 | 7 | 22 |
| LSA, LSA-G | 10 | 5 | 22 |
| CSA, CSA-G | 16 | 3 | 11 |

Tablo 2’nin ortaya çıkardığı ilginç bir bilgi algoritmaların orijinal hallerindeki dağılım şekillerinden daha başarılı sonuçların elde edilmiş olmasıdır. SOS algoritmasının orijinal halinde random dağılım kullanılırken LSA ve CSA algoritmaları ise Gauss dağılımını kullanmaktadırlar. Bu durum algoritma geliştirme aşamasında her iki dağılım şeklinin de denenmesi gerektiğini göstermektedir.

## The Comparisons of Search Time of Algorithms on Benchmark Problems

Şekil 1’de SOS, LSA ve CSA algoritmalarının, dağılım tiplerine bağlı olarak arama süreleri (sn) verilmektedir. Grafiğin x-ekseninde benchmark problemi numarası ve y-ekseninde ortalama işlem süresi verilmektedir. Şekillerden görüldüğü gibi, gauss dağılımı hesaplama süresini artırmaktadır. Özellikle SOS algoritmasında bu durum daha açık bir şekilde görülmektedir.

Şekil 1, 2 ve 3’de SOS, LSA ve CSA algoritmalarının, dağılım tiplerine bağlı olarak arama süreleri (sn) verilmektedir. Grafiğin x-ekseninde benchmark problemi numarası ve y-ekseninde ortalama işlem süresi verilmektedir. Şekillerden görüldüğü gibi, gauss dağılımı hesaplama süresini artırmaktadır. Özellikle SOS algoritmasında bu durum daha açık bir şekilde görülmektedir.



**Fig. 1.** The search time of SOS depending on the distribution type



**Fig. 2.** The search time of LSA depending on the distribution type



**Fig. 3.** The search time of CSA depending on the distribution type

Şekiller (1, 2, 3) incelendiğinde algoritmaların arama yetenekleri ile ilgili önemli bilgilere ulaşılmaktadır. İşlem süresinin benchmark problemine ve algoritmaya bağlı olarak değiştiği açıktır. Örneğin SOS algoritmasının kolay benchmark problemlerindeki (1-12 numaralı problemler) ortalama işlem süresi ile zor problemlerdeki işlem süreleri arasında çok büyük fark oluşmamıştır. LSA algoritmasında ise problemin zorluk düzeyine bağlı olarak işlem süreleri arasında anlamlı farklılıklar oluşmuştur. LSA’nın kolay problemlerde global minimum çözümü (1-2, 4-11 numaralı benchmarklar) bulma süresi SOS algoritmasınınkinden çok daha kısadır. Bunun yanında LSA’nın zor problemlerdeki işlem süresi ciddi oranda artmıştır. Bu sonuçlar problemin zorluk düzeyi ile algoritmaların arama performansı arasındaki ilişki hakkında fikir vermektedir. Örneğin LSA algoritmasının arama uzayının küçük olduğu basit problemlerde çözüme çok hızlı ulaşması onun hızlı yakınsadığına bir işarettir. Yani LSA algoritmasında komşuluk araması etkili/güçlü bir şekilde yapılmaktadır. Bunun yanında problemin zorluk düzeyi arttıkça (yerel minimum tuzaklarının sayısı ve arama uzayının büyüklüğü arttıkça) LSA nın arama süresi üstel bir biçimde artmaktadır. Bu durum LSA’nın komşuluk araması yaparken yerel minimum tuzaklarına takıldığını göstermektedir. Yerel minimum tuzaklarından kurtulmanın yolu arama sürecinde çeşitliliğin sağlanmasıdır. Dolayısıyla LSA’da çeşitliliğin etkili bir şekilde sağlanamadığı ortaya çıkmaktadır. SOS algoritmasında ise komşuluk araması ve çeşitliliğin sağlanması konusunda bir denge sağlandığı görülmektedir. Ancak bu durum iyi analiz edilmelidir. SOS algoritmasının kolay problemlerde global çözüme ulaşma süresi ile LSA’nın ki karşılaştırıldığında arada çok büyük farklar olduğu görülmektedir. Bu durum SOS algoritmasında, Xbest (en iyi çözüm adayı) etrafında hassas bir komşuluk araması yapılmadığını göstermektedir. SOS algoritmasının bu özelliği bir handikap olarak değerlendirilebilir. Bununla birlikte SOS algoritması bu özelliği sayesinde LSA algoritmasına kıyasla yerel minimum tuzaklarından daha hızlı kurtulabilmektedir. Dolayısıyla SOS, farklı zorluk düzeylerindeki problemler için tutarlı bir arama performansı gösterebilmiştir. Sonuç olarak SOS algoritmasının problemlere karşı göstermiş olduğu kararlılık LSA dan daha yüksektir. CSA algoritması ise zor problemleri (13 numaralı problemden itibaren) çözememiştir. CSA algoritması bu nedenle max. iterasyona ulaşıp aramayı sonlandırmıştır.

# Sonuçlar

Bu çalışmadan elde edilen en önemli sonuç, meta-sezgisel arama algoritmalarının yeterince test edilmeden literatüre kazandırıldığının ispatlanmış olmasıdır. Makalede incelenen üç algoritmada da yanlış dağılım yöntemi kullanılmıştır. Gauss dağılımının SOS algoritması için ve random dağılımın ise LSA ve CSA algoritmaları için daha uygun olduğu görülmüştür.

Makalede elde edilen diğer önemli bir sonuç, algoritmaların yakınsama, komşuluk araması ve çeşitlilik gibi konulardaki performansları hakkında önemli ipuçlarının elde edilmiş olmasıdır. Örneğin hızlı yakınsayan ve yerel min. tuzaklarına yakalanan algoritmalarda gauss dağılımı arama süresini ciddi miktarda artırmıştır. Bu problem random dağılımla önemli oranda çözülmektedir. Random dağılım özellikle çeşitliliği artıran bir yöntemdir. Komşuluk araması problemi yaşayan SOS’da ise gauss dağılımın pozitif etki yarattığı görülmüştür. Sonuç olarak algoritma geliştirme çalışmalarında her iki dağılım tipi de mutlaka uygulanmalıdır. Ayrıca algoritmanın karakterisitiğine bağlı olarak hızlı yakınsayan meta-sezgisellerde random dağılım uygulanarak çeşitliliğin artması ve bu sayede yerel min. tuzağından kurtulma sağlanabilir.

**References**

1. Sahin, O., Akay, B., “Comparisons of metaheuristic algorithms and fitness functions on software test data generation”, Applied Soft Computing, 49, 1202–1214, (2016).
2. Costa, L. R., Aloise, D., Mladenovic, N., “Less is more: basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering”, Information Sciences, 415–416, 247–253, (2017).
3. Akbari, M., Rashidi, H., Alizadeh, S. H., “An enhanced genetic algorithm with new operators for task scheduling in heterogeneous computing systems”, Engineering Applications of Artificial Intelligence, 61, 35–46, (2017).
4. Cheng, M-Y; Prayogo, D; Symbiotic Organisms Search: A new metaheuristic optimization algorithm, Computers and Structures, 139, 98-112, (2014).
5. Shareef, H., Ibrahim, A. A., Mutlag, A. H., Lightning search algorithm, Applied Soft Computing, 36, 315-333, (2015).
6. Rajabioun, R., “Cuckoo Optimization Algorithm”, Applied Soft Computing, 11, 5508–5518, (2011).
7. Özkış, A., Babalık, A., “A novel metaheuristic for multi-objective optimization problems: The multi-objective vortex search algorithm”, Information Sciences, 402, 124–148, (2017).
8. Michalak, K., “ED-LS – A heuristic local search for the multiobjective Fire fighter Problem”, Applied Soft Computing, 59, 389–404, (2017).
9. Miranda, P, B, C., Prudencio, R. B.C., “Generation of Particle Swarm Optimization algorithms: An experimental study using Grammar-Guided Genetic Programming”, <http://dx.doi.org/doi:10.1016/j.asoc.2017.06.040>, (2017).
10. N. Higashi, H. Iba, Particle swarm optimization with gaussian mutation, in: Swarm Intelligence Symposium, 2003. SIS’03. Proceedings of the 2003 IEEE, 72–79.
11. B. Zhang, M. Zhang, Y.-J. Zheng, Improving enhanced fireworks algorithm with new gaussian explosion and population selection strategies, in: International Conference in Swarm Intelligence, Springer, 2014, pp. 53–63.
12. 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).
13. Naik, K. N., Panda, R., “A novel adaptive cuckoo search algorithm for intrinsic discriminant analysis based face recognition”, Applied Soft Computing, 38, 661–675, (2016).
14. Zhao, W., Wang, L., “An effective bacterial foraging optimizer for global optimization”, Information Sciences, 329, 719–735, (2016).
15. Noel, M. M., “A new gradient based particle swarm optimization algorithm for accurate computation of global minimum”, Applied Soft Computing, 12, 353–359, (2012).
16. Bindiya, T. S., Elias, E., “Meta-heuristic evolutionary algorithms for the design of optimal multiplier-less recombination filter banks”, InformationSciences, 339, 31–52, (2016).
17. Leeuwen, Jan van, ed. Handbook of Theoretical Computer Science. Vol. A, Algorithms and complexity. Amsterdam: Elsevier. ISBN 0262720140. OCLC 247934368, (1998).
18. Molga, M., Test functions for optimization needs, http://www.robertmarks.org/Classes/ENGR5358/Papers/functions.pdf, 1-43, 2005.