©2019 Published in 7th International Symposium on Innovative Technologies in Engineering and Science 22-24 November 2019 (ISITES2019 SanliUrfa - Turkey)
https://doi.org/10.33793/acperpro.02.03.123

Gezgin Satıcı Problemi için Feromonal Yapay Arı Koloni (fYAK) Algoritması

*[1]Salim Basköy, [2]Dursun Ekmekci, [3]Hüseyin Altinkaya, [3]Mustafa Yilmaz
*[1]Karabük Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, Karabük, Türkiye
[2]Karabük Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Karabük, Türkiye
[3]Karabük Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği Bölümü, Karabük, Türkiye

Özet

Yapay arı koloni (YAK) ve karınca koloni optimizasyonu (KKO), farklı türden birçok optimizasyon problemine başarıyla uygulanmış ve geçerli çözümler üretebilmiş algoritmalardır. YAK, kâşif arıların rassal arama faaliyetiyle başarılı bir keşif yeteneğine sahiptir. Ancak sömürü faaliyetine işçi arıların yanı sıra gözcü arılar da katılsa da yerel aramada yetersizdir. KKO ise başarılı bir sömürü yeteneğine rağmen, arama bölgelerini yeterince tarayamamaktadır. Bu çalışmada KKO'nun feromon bileşeni YAK'a entegre edilerek geliştirilen feromonal YAK (fYAK) algoritması tanıtılmış ve gezgin satıcı problemi (GSP) çözümündeki performansı tartışılmıştır.

Anahtar kelimeler: Gezgin Satıcı Problemi, Yapay Arı Koloni Algoritması, feromonal Yapay Arı Koloni Algoritması, Karınca koloni optimizasyonu

A Pheromonal Artificial Bee Colony (pABC) Algorithm for Traveling Salesman Problem

Abstract

Artificial bee colony (ABC) and ant colony optimization (ACO) have been successfully applied to many different types of optimization problems and can produce valid solutions. ABC is capable of successful exploration with the random search activity of scout bees. However, although the employed bees, as well as the onlooker bees, participate in the exploitation activity, it is insufficient in local exploration. On the other hand, the ACO is unable to adequately search the exploration areas despite its successful exploitation capability. In this study, the pheromone component of ACO is introduced to the pheromone ABC (pABC) algorithm, which is developed by integrating it into ABC, and its performance in the traveling salesman problem (TSP) solution is discussed.

Key words: Traveling salesman problem, Artificial bee colony algorithm, pheromonal Artificial bee colony algorithm, Ant colony optimization

1. Giriş

Optimizasyon (eniyileme), genel bir tanımla eldeki kısıtlı kaynakları optimal (en uygun) biçimde değerlendirmek olarak açıklanabilir. Yapay zekâ perspektifinden optimizasyon, bir problemin istenen kısıtları sağlayacak şekilde minimize ya da maksimize edilmesi olarak tanımlanabilir. Bu

*Sorumlu yazar: Adres: Mühendislik Fakültesi, Mekatronik Mühendisliği Bölümü, Karabük Üniversitesi, 78050, Karabük TÜRKİYE. E-posta adres: 2015010225048@ogrenci.karabuk.edu.tr, Tel: +905393608306

bağlamda amaç; bir gerçel fonksiyonu, tanım aralığında minimize veya maksimize etmek amacı ile gerçek ya da tam sayı değerlerini bulma işlemidir. Problem parametreleri, sonlu sayıdaki nesnelerden oluşuyorsa, bu problemler "kombinatoryal problemler" olarak adlandırılır. Kombinatoryal optimizasyon problemlerinde, problem parametrelerine eklenen her bir değişken, problemin çözüm uzayını (buna bağlı olarak çözüm zamanını) bir önceki duruma göre daha çok artırdığından NP problemler sınıfında değerlendirilir. Kombinatoryal optimizasyon problemlerinin çözümü için; yakınsama algoritmaları, olasılığa dayalı algoritmalar ve problemin özel yapısına duyarlı tarzda değişik algoritmik yöntemler geliştirilmiştir.

Gezgin satıcı problemi (GSP), yöneylem açısından, ayrık yapılı kombinatoryal optimizasyon problemlerinin en popüler örneklerinden biridir. İlk matematiksel problemler 1800'lü yıllarda İrlandalı matematikçi William Hamilton ve İngiliz matematikçi Thomas Penyntgon Kirkman tarafından tanımlanmıştır. GSP, n adet düğümden oluşan bir ağ modelinde, düğümler arası mesafelerin bilindiği ve düğümlerin her birini *"mutlaka ve yalnızca"* bir kez ziyaret etmek koşuluyla başlangıç düğümüne dönme problemidir [1]. $G\ (N, E)$ kapalı grafında N, i'den n'e kadar düğümleri ve E, bu düğümler arası kenarları temsil eder. Verilen C maliyet (mesafe) matrisinde, c_{ij}, i-j düğümleri arası maliyeti temsil eder. Her iki düğüm arası gidiş-dönüş mesafesi eşit uzaklıktaysa ($c_{ij}=c_{ji} \quad \forall\ i, j \in N$) problem simetriktir, aksi halde asimetrik problem olarak adlandırılır. İki düğüm arası mesafeyi hesaplamak için çoğu zaman Öklid Formülü kullanılır. Problem kapsamında amaç, en kısa (optimal) turun bulunmasıdır. GSP'de problemin düğüm sayısı arttıkça çözüm zamanı, deterministik polinomal zamanı aştığından, çözüm yöntemi olarak daha çok sezgisel teknikler tercih edilmiştir. Sezgisel teknikler, problemin optimal çözümünü bulmayı garanti etmeseler de makul sürelerde geçerli çözümler bulabilirler. Evrimsel hesaplama yeteneğine sahip bu yöntemlerin önemli bir bölümü koloni halinde yaşayan organizmaların aralarındaki iş bölümünü taklit eder. Bu yöntemlere, karınca koloni optimizasyonu (KKO) [2], yapay arı koloni (YAK) algoritması [3], parçacık sürü optimizasyonu (PSO) [4], ateş böceği algoritması (ABA) [5] örnek verilebilir. Literatürde GSP çözümü için, sürü zekâsı temelli bu yöntemler ve bunlardan farklı olarak evrimsel algoritmalar kullanılarak geliştirilen birçok çalışma mevcuttur [6] [7][8][9][10][11][12] [13][14][15][16].

Bu çalışmada, YAK algoritmasının farklı bir türevi olarak, algoritmanın yerel arama yeteneğini, KKO'daki feromon yaklaşımıyla güçlendiren feromonal YAK (fYAK) algoritması ele alınmış ve GSP için çözüm önerilmiştir. Önerilen algoritma, GSP için oluşturulan popüler test problemleri üzerinde denenmiş ve elde edilen sonuçlar YAK ve KKO algoritmalarının standart versiyonlarıyla elde edilmiş literatür sonuçlarla karşılaştırılmıştır. Çalışmanın kalan bölümleri şu şekilde tasarlanmıştır: Bölüm 2'de YAK, KKO ve bu algoritmaların karma versiyonu olan fYAK algoritmaları açıklanmış, Bölüm 3'te deneysel çalışmalarda kullanılan GSP modeli için tasarlanmış test problemlerine değinilmiştir. Bölümde ayrıca, çözüm için belirlenen parametre seti sunulmuştur. Bölüm 4'te önerilen yöntemin ve ayrıca standart YAK ve standart KKO yöntemlerinin GSP çözüm yaklaşımları analiz edilmiş ve test problemlerinde elde ettiği sonuçlar karşılaştırılmıştır. Bölüm 5'te çalışma genel hatlarıyla değerlendirilmiş ve yorumlanmıştır.

2. Materyal ve Metot

GSP çözümü için fYAK algoritması tercih edilmiştir. fYAK, YAK algoritmasına ve KKO'nun

feromon yaklaşımı entegre edilerek geliştirilmiş bir karma algoritma olduğundan, öncelikle YAK algoritması detaylı olarak açıklanmış ardından KKO'daki feromon güncelleme adımları anlatılmıştır.

YAK, Derviş Karaboğa tarafından arıların koloni halinde yiyecek bulma davranışlarını temel alarak geliştirilmiş bir optimizasyon algoritmasıdır [3]. YAK algoritması kabulünde, besin kaynağı sayısı, işçi arı ve gözcü arı sayılarına eşittir. Diğer bir ifadeyle koloni boyutu, besin kaynağının iki katı kadardır. Algoritmanın temel adımları şunlardır:

Başlangıç yiyecek kaynağı pozisyonlarının belirlenmesi
TEKRARLA
    İşçi arıların yiyecek kaynağına gönderilmesi ve yeni besinin kalitesinin hesaplanması
    Gözü arılar için kaynak seçiminde kullanılacak olasılık değerlerini hesaplanması
    Gözcü arılar hesaplanan olasılık değerlerine göre yiyecek kaynaklarına yönlendirilmesi
    Kaynaktan ayrılma kriteri: limit ve kâşif arı üretilmesi
TA Kİ şartlar sağlanıncaya kadar

Algoritma kabulünde üç tür arı grubu vardır: gideceği besin kaynağı belirli olan işçi arılar, kendi tercihlerine göre besin kaynaklarına yönelen gözcü arılar ve bağımsız olarak kaynak arayışındaki kâşif arılar. Arıların yiyecek toplama sürecinde aralarındaki koordinasyon kolektif bilginin oluşumunu sağlamaktadır. Arılar, aralarındaki iletişimi dans ile sağlarlar. Paylaşılan bilgi ile kaliteli yeni yiyecek kaynakları keşfedilir [17]. Yiyecek kaynağının konumunu öğrenen işçi arı kaynağa ulaşmak için güneş ışınlarından faydalanır. Arılar yörüngeler ile güneş ışınları arasındaki acıyı hesaplama yeteneğine sahiptir. Enerji tüketimine göre uzaklık belirleyen arılar yüklerine göre farklı yükseklikte uçarak enerjilerini ayarlarlar [18]. Algoritmadaki arı sayısını besin kaynağı sayısı belirler. Algoritmada besin sayısının iki katı kadar arı vardır bu arıların yarısı işçi arı diğer yarısı gözcü arıdır. Kaynakta görevli arı kaynaktaki nektar bitince kâşif arı olmaktadır. Algoritmada, yiyecek kaynakların konumları çözülmek istenilen problemin olası çözümlerine, nektar miktarı ise çözümün kalitesine karşılık gelmektedir. Arılar en fazla nektara sahip kaynağın yerini bulma eğilimindedir. Diğer bir ifadeyle arama uzaydaki optimum noktayı bulmaya çalışmaktadırlar [18].

KKO ise gerçek karınca kolonisinin yiyecek arama davranışını matematiksel modelleme üzerine dayalı bir algoritmadır. İlk çalışma Dorigo ve arkadaşları tarafından yapılmıştır [2]. Bu metodun farklı algoritmik modelleri oluşturulsa da karıncaların iletişimde kullandıkları feromon anlayışı, modelin tüm versiyonlarında sabit kalmıştır. KKO adımları şu şekildedir:

Başlangıç feromon değerleri belirlenir
Karıncalar her düğüme rastsal olarak yerleştirilir
TEKRARLA
    Her karınca, düğümler arası feromon miktarların dikkat ederek turunu tamamlar
    Her karıncanın turuna göre lokal feromon güncellemesi yapılır
    En iyi karıncanın turuna göre global feromon yenilemesi yapılır
TA Kİ şartlar sağlanıncaya kadar

Karıncalar çevresel etkenlere göre besin kaynağı ile yuvası arasındaki olası yolları belirlemektedir.

Yiyecek arayışındaki karıncalar yola feromon salgısı bırakır. Feromon, zamanla buharlaştığından, salgı yoğunluğu yol uzunluğuyla orantılıdır denebilir. Bu bağlamda daha yoğun salgı birikimi, daha kısa yolu temsil eder ve salgıyı referans alan sonraki karıncalar bu yolu kullanır. Yol ayrımına gelen bir karınca, hangi yolu seçeceğine iki şekilde karar verir (Geçiş Kuralı): Birinci alternatifte salgı yoğunluğunun en fazla olduğu güzergahı tercih eder. Genellikle bu seçim olasılığı %90 olarak belirlenir. İkinci alternatifte ise yollardaki feromon miktarına göre belirlenen olasılık dağılımına bağlı olarak yol seçer.

Tüm karıncalar turlarını tamamladıktan sonra feromon güncellemesi yapılır. Bu güncelleme iki şekilde olur: Lokal feromon güncellemesi ve global feromon güncellemesi. Lokal feromon miktarının güncellenmesi için yollardaki feromon miktarlarının bir kısmı belirlenen oranda buharlaştırır. Daha sonra karıncaların geçtiği yollardaki feromon miktarı, ilgili hattı kullanan karıncaların kat ettikleri toplam yol mesafeyle ters orantılı olarak artırılır. Global feromon güncellemesinde ise geçerli adımda en kısa turu bulan karıncanın, güzergahımdaki feromon düzeyi artırılır. Amaç, bulunan en iyi sonuçların ileriki iterasyonlara belli oranda aktarılmasıdır. Çalışmalar, GSP çözümünde kullanılacak optimum karınca sayısının şehir sayısına eşit olması gerektiğini göstermektedir[2].

Araştırmacılar bu iki algoritmanın ayrık optimizasyon problemlerinde elde ettiği sonuçları değerlendirdiğinde, YAK algoritmasının yerel aramada yeterince etkili olamadığı, KKO'nunsa arama uzayına yeterince dağılamadığı yorumunda birleşmektedir. Bu anlayıştan yola çıkarak, YAK algoritmasının yerel arama yeteneğini feromon yaklaşımıyla takviye eden ve kâşif arı fazıyla, algoritmanın keşif kabiliyetini koruyan fYAK modeli geliştirilmiştir. fYAK versiyonunun standart YAK algoritmasından ayrıldığı temel nokta, işçi ve gözcü arıların iletişim yöntemidir. Orijinal YAK'ta arılar dans ederek iletişim kurarken, fYAK'ta bu iletişim feromonla sağlanır. Buna bağlı olarak oluşan diğer bir farklılık ise, çözüm uzayındaki muhtemel çözümün temsilidir. YAK'ta olası çözümler besin kaynaklarının pozisyonuyla temsil edilirken, fYAK'ta polenlerden oluşan besinle temsil edilir. fYAK modelinde bal arıları, yüksek kalitede bal üretebilmek için daha kaliteli besin tüketmeye çalışır. Besin, farklı çiçek türlerinden toplanan polenlerden oluşur. Besin elde etmek için çiçeklerden polen toplayan işçi arılar, çiçekler arasında feromon yayarlar. İşçi arıları takip eden gözcü arılar da polen toplarken bu feromon salgılarına bağlı olarak çiçek seçimi yaparlar. Kâşif arılar, tıpkı standart YAK modelinde olduğu gibi bağımsız arama yaparlar. Bu sayede KKO'nun yerel optimuma takılma problemi aşılmış olur. fYAK, YAK algoritmasına KKO'nun Karınca Koloni Sistemi (KKS) versiyonundaki feromon yaklaşımı entegre edilerek geliştirilmiştir. fYAK algoritmasının sözde kodu şu şekilde tasarlanmıştır:

Besin kaynağı sayısı, limit, iterasyon sayısı, α, β, ρ, q0 parametreleri için değerler ata
Başlangıç besinlerini oluştur
Başlangıç besinlerinin kalitesini değerlendir
Döngü sayacını 1 olarak ayarla
TEKRARLA
    Her bir işçi arı için {
        Yeni çözüm üret *(vi)*
        Amaç fonksiyonuna göre *f(i)* değerini hesapla
        Açgözlü seçim uygula}

Tüm çiçekler arası feromon değerini ata
Yerel feromon güncellemesi
Global feromon güncellemesi
Her bir gözcü arı için {
Seçilecek her bir çiçek için {
Ziyaret edilmemiş her bir çiçeğin seçilme olasılığını hesapla
[0,1] arası rastgele *q* üret
Eğer(*q*<=*q0*)
Sonraki çiçek olarak seçilme olasılığı en yüksek çiçeği seç
Else
Sonraki çiçeği olasılık dağılımına göre seç}
Amaç fonksiyonuna göre türetilmiş *f(i)* değerini hesapla
Oluşturulan besine, en çok benzeyen mevcut besini bul
Açgözlü seçim uygula}
Keşif için terkedilmiş bir çözüm varsa daha sonra üretilen bir çözümle değiştir
En iyi çözümü hafızaya al
Döngü sayacı +1
TA Kİ (döngü sayacı = iterasyon sayısı)

3. Deneysel Çalışmalar

Çalışma kapsamında, fYAK algoritmasının GSP çözümündeki performansını analiz edebilmek için, C# programlama dilinde geliştirilen uygulama, GSP'nin iki popüler test problemi: KroA200 ve KroB200 örnekleri üzerinde birbirinden bağımsız olarak 30 kez çalıştırılmıştır. KroA200 ve KroB200 test problemleri, her birinde 200 düğüm bulunan, geniş ve dalgalı yapıda çözüm uzayına sahip problemlerdir. Algoritmanın elde ettiği sonuçlar, standart YAK ve standart KKO algoritmalarıyla elde edilen sonuçlarla karşılaştırılmıştır. Deneylerde, algoritmalar için belirlenen parametre değerleri Tablo 1'de gösterilmektedir.

Tablo 1. Parametre seti

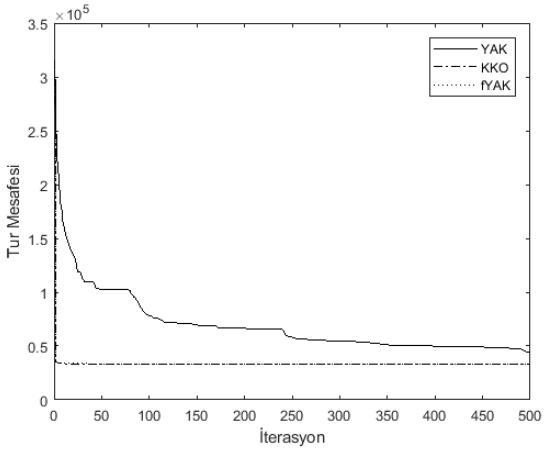
	Koloni Boyutu	Limit	α	β	ρ	*q0*
YAK	200	1000	-	-	-	-
KKO	200	-	1	5	0.1	0.9
fYAK	200	1000	1	5	0.1	0.9

KKO parametreleri için değerler belirlenirken, algoritmanın benzer GSP örnekleri ile diğer ayrık yapılı problem çözümlerinde elde ettiği en başarılı sonuçlar dikkate alınmıştır. Bu bağlamda, karınca sayısı, her bir düğüme bir karınca yerleştirilecek biçimde 200 olarak atanmış, α ve β sezgisel parametreleri ile, ρ ve *q0* parametreleri için, literatürde elde edilen en başarılı sonuçlardaki parametre değerleri seçilmiştir. YAK algoritması parametreleri belirlenirken, adil bir karşılaştırma için, algoritmanın koloni boyutu (*KB*), KKO koloni boyutuna eşitlenmiş, dolayısıyla kaynak sayısına 100 değeri atanmıştır. Algoritmanın *limit* parametresi için, genel kullanım formülü *"(KB*N)/2"* ((200*200)/2=20000) yerine, iterasyon sayısı düşük tutulduğundan 1000 değeri

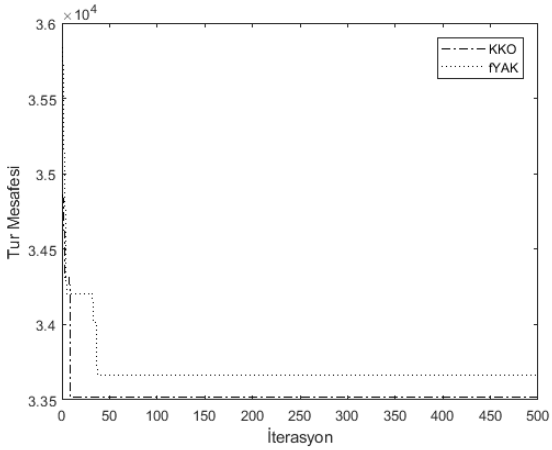
atanmıştır. Algoritmaların her bir iterasyondaki işlem sayıları ve toplam çözüm süreleri incelendiğinde, eşit sürelerde YAK algoritmasının, KKO ve fYAK algoritmalarına göre yaklaşık 20 kat daha fazla iterasyon yaptığı gözlemlenmiştir. Dolayısıyla her bir denemede YAK algoritmasının iterasyon sayısı için 10000, KKO ve fYAK algoritmalarının iterasyon değeri için 500 değerleri atanmıştır.

4. Bulgular ve Karşılaştırma

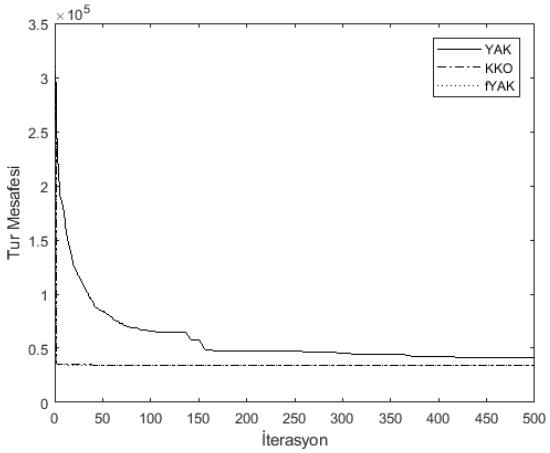
Algoritmaların KroA200 ve KroB200 test problemlerindeki yakınsama performanslarını değerlendirebilmek için KKO ve fYAK algoritmalarının 500 iterasyonda, YAK algoritmasının da 20'şer iterasyon aralıklarla (i=1:20:10000) elde ettiği sonuçlar Şekil 1a ve Şekil 2a'da gösterilmektedir. KKO ve fYAK ikinci iterasyondan itibaren birbirine çok yakın sonuçlar elde ettiklerinden, bu algoritmaların ikinci iterasyondan sonra elde ettikleri sonuçlar Şekil 1b ve Şekil 2b'de detaylı olarak gösterilmektedir.



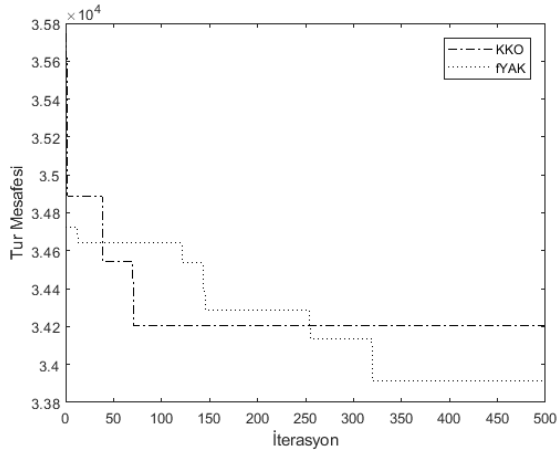
Şekil 1a. KroA200 için YAK, KKO ve fYAK sonuçları



Şekil 1b. KroA200 için KKO ve fYAK sonuçları



Şekil 2a. KroB200 için YAK, KKO ve fYAK sonuçları



Şekil 2b. KroB200 için KKO ve fYAK sonuçları

Şekil 1a ve Şekil 2a'dan da görülebileceği gibi fYAK algoritması, YAK algoritmasından daha güçlü bir yerel arama yeteneğine sahiptir. Her iki problem çözümünde de fYAK, KKO algoritmasına benzer çözümler üretmiştir. Şekil 2b'den net olarak görülebileceği gibi KKO, belli bir iterasyondan sonra keşif faaliyetinde yetersiz kalıp, arama bölgelerine yeterince dağılamazken; fYAK, YAK algoritmasının keşif yeteneğini korumuş ve sonraki iterasyonlarda daha başarılı sonuçlara ulaşabilmiştir.

Algoritmaların bu test problemleri için 30 bağımsız denemede elde ettikleri sonuçlar Tablo 2'de gösterilmektedir.

Tablo 2. Algoritmaların test problemleri çözümlerinde elde ettikleri sonuçlar

	KroA200			KroB200		
	YAK	KKO	fYAK	YAK	KKO	fYAK
En İyi	32608,90	**32410,31**	32608,90	40240,15	**33478,27**	32829,05
En Kötü	54585,16	**33911,41**	33930,08	54562,00	34352,40	**34115,08**
Ortalama	46691,55	33292,12	**33246,09**	46303,53	33946,02	**33651,32**
Standart Sapma	5117,79	**336,78**	384,33	3746,06	**272,26**	290,38

Tablo 2'deki sonuçlara bakarak KKO ve fYAK algoritmalarının çok daha az iterasyon sayısında bile YAK algoritmasına nispeten daha başarılı sonuçlar üretebildikleri görülmektedir. Her ne kadar her iki test probleminde de en başarılı sonuçlar KKO tarafından üretilmiş olsa da fYAK, KKO'nun çok az gerisinde kalmış ve her iki test problemi için 30'ar denemede en başarılı ortalamaya ulaşmıştır. KKO ve fYAK sonuçlarının standart sapma verileri incelendiğinde, fYAK'ın daha başarılı ortalama sonuçlara, daha yüksek standart sapma değerleriyle ile ulaşması, algoritmanın keşif yeteneğinin KKO'dan çok daha başarılı olduğu şeklinde yorumlanabilir.

4. Sonuç

Çalışmada, optimizasyon problem çözümlerinde yaygın olarak tercih edilen YAK algoritmasının yerel arama kabiliyetini geliştirmek için tasarlanan fYAK modeli tanıtılmış ve GSP çözümlerindeki performansı analiz edilmiştir. Test problemlerinden elde ettiği sonuçlar, algoritmanın kombinatoryal optimizasyon problemlerine uygulanabilirliğini göstermektedir. Algoritmanın, problem çözümlerindeki yakınsama performansı, elde edilen sonuçlarla birlikte değerlendirildiğinde; fYAK'ın, feromon yaklaşımıyla standart YAK algoritmasına kıyasla daha güçlü bir sömürü yeteneğin sahip olduğu ve standart KKO algoritmasına oranla arama bölgelerini daha başarılı keşfedebildiği söylenebilir.

Kaynaklar

[1] U. Cevre, "Çoklu Gezgin SaticiProblemi," Ege Üniversitesi, 2008.
[2] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive Feedback as a Search Strategy," Milano, 1991.
[3] D. Karaboğa, "AN IDEA BASED ON HONEY BEE SWARM FOR NUMERICAL OPTIMIZATION," Kayseri, 2005.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization-based feature selection for cognitive state detection," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2011, pp. 6556–6559.
[5] X. S. Yang, "Firefly algorithm, stochastic test functions and design optimization," *Int. J. Bio-Inspired Comput.*, vol. 2, no. 2, pp. 78–84, 2010.
[6] M. Bakhouya and J. Gaber, "An immune inspired-based optimization algorithm: Application to the traveling salesman problem," *Adv. Model. Optim.*, vol. 9, no. 1, pp. 105–116, 2007.
[7] N. Shine, "Ant colonies for the Traveling Salesman Problem," vol. 43, pp. 73–81, 2015.
[8] H. Jiang, "Artificial Bee Colony algorithm for Traveling Salesman Problem," no. March, 2015.
[9] G. Kavitha, "Bacterial Foraging Optimization (BFO) based Traveling Salesman Problem (TSP) for Data Collection in Mobile Sinks," vol. 2, no. 1, pp. 76–94, 2015.
[10] D. Bookstaber, "Simulated Annealing for Traveling Salesman Problem," no. 3, pp. 1–14, 2011.
[11] M. Lin, Y. Zhong, J. Lin, and X. Lin, "Discrete bird swarm algorithm based on information entropy matrix for traveling salesman problem," *Math. Probl. Eng.*, vol. 2018, 2018.
[12] M. Sornam, "Enhanced Bat Inspired Algorithm for Solving Travelling Salesman Problem," vol. 02, no. 11, pp. 26–31, 2017.
[13] T. Multiple, T. Salesman, and H. S. Methods, "Çoklu Gezgin Satıcı Problemi ve Sezgisel Çözüm Yöntemlerinin İncelenmesi."
[14] M. Saraei and P. Mansouri, "Solving of Travelling Salesman Problem using Firefly Algorithm with Greedy Approach," *Sci. J.*, vol. 36, no. 2015, 2015.
[15] A. Shaheen, A. Sleit, and S. Al-Sharaeh, "A solution for traveling salesman problem using grey wolf optimizer algorithm," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 18, pp. 6256–6266, 2018.
[16] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, 2010, vol. 4, no. 3, pp. 1942–1948.
[17] C. Grüter and W. M. Farina, "The honeybee waggle dance: can we follow the steps?," 2009.
[18] B. Akay, "Nümerik Optimizasyon Problemlerinde Yapay Ari Kolonisi (Artificial Bee Colony) Algoritmasinin Performans Analizi," Erciyes Üniversitesi, 2009.