Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı, S. 442-450, Ağustos 2020

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue, pp. 442-450, August 2020

Research Article

Optimizasyon Problemleri İçin Geliştirilmiş Feromonal Yapay Arı Koloni (gfYAK) Algoritması

Dursun Ekmekci[1*]

[1] Karabük Üniversitesi, TOBB Teknik Bilimler MYO, Bilgisayar Teknolojileri Bölümü, Karabük, Türkiye (ORCID: 0000-0002-9830-7793)

(Bu yayın 26-27 Haziran 2020 tarihinde HORA-2020 kongresinde sözlü olarak sunulmuştur.)

(**DOI:** 10.31590/ejosat.780695)

ATIF/REFERENCE: Ekmekci, D. (2020). Optimizasyon Problemleri İçin Geliştirilmiş Feromonal Yapay Arı Koloni (gfYAK) Algoritması. *Avrupa Bilim ve Teknoloji Dergisi*, (Special Issue), 442-450.

Öz

Çevreden bilgi toplayan ve davranışını buna göre belirleyen bal arısı kolonisi, sürü yaşamının en popüler örneklerinden biridir. Bu dinamik yapı, farklı yaklaşımlarla değerlendirilerek, birçok mühendislik problemine çözüm önerisi getirilmiştir. Bilgisayar sistemleri alanında, bilgisayar ağları, mobil ağ optimizasyonu, sayısal ve kombinasyonel optimizasyon için geliştirilen birçok çözüm önerisi, bal arısı kolonisinin davranışlarını model almaktadır. Yöneylem araştırmacıları ise daha çok, koloni üyelerinin, haberleşme, etkileşim, evlilik ve yem arama davranışlarına yoğunlaşmışlardır. Bu bağlamda, yiyecek arama ve toplama görevini üstlenen bal arılarını taklit eden Yapay Arı Koloni (YAK) Algoritması, optimizasyon literatüründe önemli bir başarıya sahiptir. Klasik algoritma yapısında, limit periyodunda oluşturulan rastgele çözümler, algoritmayı yerel optimumdan kurtarırken, daha başarılı çözümler üretebilmek için rulet tekerleği kullanılır. Ancak arılar arasındaki etkileşimi daha verimli kılan algoritma türevleri de geliştirilmiştir. Bu çalışmada bal arılarının daha verimli yerel arama yapabilmesi için geliştirilen feromonal YAK (fYAK) algoritması ele alınmıştır. fYAK'ta gözcü arıların, işçi arıların tecrübesinden daha fazla yararlanabilmesi için feromon salgısı kullanılır. Böylece gözcü arılar, yeni çözümler üreten değil, yeni çözümler oluşturan prosedürler kullanır. Çözüm önerisi olarak sunulan Geliştirilmiş fYAK (gfYAK) modelinde, çözüm bileşenleri arasındaki korelasyonu, çözüm başarısıyla daha çok ilişkilendiren hafıza ve algoritmanın daha etkili çözüm bölgelerine yönelmesini sağlayan transfer fonksiyonları kullanılmaktadır. Herbir çevrimde hafıza ve buna bağlı olarak feromon matrisi güncellenmektedir. İlgili çevrimde, o ana kadarki en iyi çözüm bulunmuşsa feromon matrisi için genel güncelleme yapılır. Algoritma yakınsama performansını araştırabilmek ve transfer fonksiyonlarının etkisini analiz edebilmek için, çalışma kapsamında üç farklı transfer fonksiyonu kullanılmıştır. Farklı boyutlardaki Gezgin Satıcı Problemi (GSP) üzerinde yapılan denemeler, algoritmanın klasik YAK ve fYAK'a oranla daha iyi çözümler üretebildiğini göstermiştir.

Anahtar Kelimeler: YAK, fYAK, gfYAK.

An Improved Pheromoal Artificial Bee Colony (ipABC) Algorithm for Optimization Problems

Abstract

Honey bee colony, which collects information from the environment and determines its behavior accordingly, is one of the most popular examples of colonial life. This dynamic structure has been evaluated with different approaches and solutions have been proposed for many engineering problems. In the field of computer systems, many solutions proposed for computer networks, mobile network optimization, numerical and combinatorial optimization model the behavior of the honey bee colony. Operations researchers are mostly concentrated on the communication, interaction, marriage, and foraging behaviors of colony members. In this context, the Artificial Bee Colony (ABC) Algorithm, which imitates honey bees, which take on the task of searching and collecting food, has significant success in the optimization literature. In the classical ABC structure to derive more successful solutions the roulette wheel is used, and to escape the algorithm from the trap of local optima, random solutions are evaluated during the "limit" period. However,

[*] Sorumlu Yazar: Karabü Üniversitesi, TOBB Teknik Bilimler MYO, Bilgisayar Teknolojileri Bölümü, Karabük, Türkiye, ORCID: 0000-0002-9830-7793, dekmekci@karabuk.edu.tr

algorithm derivatives have been developed that make the interaction between bees more efficient. In this study, the pheromonal ABC (pABC) algorithm developed for honey bees to search more efficiently is discussed. In pABC, the pheromone trail is used for the onlooker bees to benefit more from the experience of employed bees. Thus, onlookers use procedures that construct new solutions, not derive. In the improved pABC (ipABC) model, which is presented as a solution proposal, memory is used, which relations the correlation between the solution components with the success of the solution, and transfer functions are used to enable the algorithm to move to more effective solution regions. Memory and pheromone matrix are updated in each cycle. In the current cycle, if the best solution ever found, a general update is made for the pheromone matrix. Three different transfer functions were used in the study to investigate the convergence performance of the algorithm and analyze the effect of transfer functions. Experiments on different sizes of Traveling Salesman Problem (TSP) have shown that the algorithm can produce better solutions compared to classical ABC and pABC.

Keywords: ABC, pABC, ipABC.

1. Giriş

Toplu halde yaşayan ve hayatta kalabilmeleri için gerekli bazı aktiviteleri iş bölümüyle yerine getirebilen canlıların oluşturduğu grup için "sürü" tabiri kullanılmaktadır. Kolonideki bireyler, bazı işlevleri, diğer üyelerle etkileşime girmeden tamamlarken, birtakım görevlerin tamamlanabilmesi, bireylerin aynı anda farklı görevleri yerine getirilmesiyle mümkündür. Teknik açıdan, bireysel faaliyetler, gelen bilginin yorumlanarak tüm sistemi etkileyecek şekilde kullanılması, kolektif faaliyetler ise, sistem fonksiyonlarının etkileşimli olarak eşzamanlı yürütülmesi anlamına gelir. Araştırmacılar, bu şekilde eşzamanlı yürütülen grup çalışmasının, uzmanlaşmamış bireylerce sırayla devam ettirilmesinden daha verimli olduğuna dikkat çekmektedirler (Barnebau, Dorigo, & Theraulaz, 1999). Mühendislik alanında sürü zekasına dayalı yaklaşım ilk olarak, hücresel robotik sistemlerde kullanılmış, desen üretiminde kullanılan robot ajanlar, bir veya iki boyutlu ortamda en yakın komşu etkileşimiyle organize olmuşlardır (Beni & Wang, 1993). Bilgisayar bilimcileri ise, kolektif zekaya dayalı bu etkileşimi, daha çok yapay zekâ alanında kullanmışlar, çok sayıda optimizasyon algoritması modellemişlerdir. Bu modeller incelendiğinde, genel olarak, bireysel prosedürlerin algoritmanın keşif yeteneğini belirlediği, etkileşimli prosedürlerin ise algoritmanın sömürü performansını oluşturduğu söylenebilir.

Yapay Arı Koloni (YAK) Algoritması da bal arısı kolonisinin besin toplama faaliyetleri modellenerek, Karaboğa tarafından geliştirilmiş bir optimizasyon algoritmasıdır (Karaboga, 2005). Algoritma, mühendisliğin, literatür bazlı ve endüstriye dayalı farklı birçok alanında karşılaşılan, değişik karakteristiklere sahip optimizasyon problemlerine başarıyla uygulanabilmiş, geçerli çözümler oluşturabilmiştir (Bansal, Sharma, & Jadon, 2013). YAK modelinde, yapay kâşif arıların rastgele çözüm oluşturmasıyla başlayan işlemler, sırasıyla işçi ve gözcü arı prosedürlerinin ardışık ve iteratif çözüm türetme adımlarıyla devam eder. Bu bağlamda, başlangıç çözümlerinin oluşumunda ve çözümlerin daha iyiye yakınsanamadığı arama sürecinde kullanılan yapay kâşif arılarla, çözüm uzayının keşfi hedeflenir. Diğer yandan, yapay işçi ve gözcü arılar, mevcut çözümlerin komşuluğunda daha başarılı çözümler türetmekle görevlidirler (Akay & Karaboga, 2015). Literatür çalışmalarında, YAK performansını artırmaya yönelik çok sayıda öneri sunulmaktadır (Monteiro, Fontes, & Fontes, 2012).

Gözcü arıların, işçi arı tecrübesinden daha fazla yararlanabilmesi için geliştirilen bir modelde (feromonal YAK - fYAK), etkileşim için feromon salgısı kullanılmıştır (Ekmekci, 2019a). Klasik YAK modelindeki her bir çevrimde, işçi arı safhası tamamlandıktan sonra, güncel çözümler rulet tekerleğine yerleştirilmektedir. Gözcü arılar, çözüm türetmek için gerekli çözümleri rulet tekerleği ile seçer. Böylece daha iyi çözümlerin seçilme olasılığı artırılmaktadır. fYAK modelinde ise bu yaklaşım değiştirilmiş, algoritmaya, karınca koloni optimizasyonunda (KKO) kullanılan feromon modeli entegre edilmiştir. Yapay gözcü arılar çözüm türetmek yerine, çözüm bileşenleri arasındaki korelasyonu dikkate alarak, yeni çözümler oluştururlar. Bu çalışmada ise, çözüm başarısını, çözümün bileşenleriyle doğrudan ilişkilendiren ve algoritmanın arama yönünü, transfer fonksiyonlarıyla tayin eden Geliştirilmiş fYAK (gfYAK) yöntemi tanıtılmaktadır. Yaklaşım, standart KKO algoritmasına uygulandığında, KKO'nun başarısını artırdığı daha önceki çalışmada (Ekmekci, 2019b) tespit edilmiştir.

Makalenin kalan bölümleri şu şekilde tasarlanmıştır: Bölüm 2'de YAK modeli anlatılmış, fYAK yöntemindeki feromon entegresi detaylı olarak açıklanmış ve gfYAK'taki güncellenen adımlar vurgulanmıştır. Bölüm 3'te, her üç algoritma, gezgin satıcı problemi (GSP) için geliştirilen örnek problemler üzerinde test edilmiş, algoritma performansları analiz edilmiştir. Bölümde ayrıca, algoritmalar, diğer popüler optimizasyon algoritmalarıyla karşılaştırılmışlardır. Bölüm 4'te ise çalışma genel hatlarıyla değerlendirilmektedir.

2. Önerilen Yöntemin Altyapısı

Önerilen yöntem, klasik YAK algoritmasının sömürü yeteneğini artırmak için geliştirilen fYAK modelinde, feromon salgısını daha etkin kullanabilmek için geliştirilmiştir. Bu kapsamda, öncelikle YAK algoritması anlatılmakta ve algoritmanın fYAK versiyonu açıklanmakta, ardından gfYAK modeli sunulmaktadır.

2.1. Klasik YAK Algoritması

YAK algoritması, bir bal arısı kolonisindeki, besin kaynağı arama ve bulunan kaynaklardan besin toplamakla görevli üç farklı arı türünün (*kâşif, işçi* ve *gözcü*) davranışını modeller. Sosyal süreçte, öncelikle *kâşif* arılar, arama bölgesinde, rastgele yiyecek kaynakları ararlar. Yiyecek kaynaklarının bulunmasından sonra, besin toplama işlemi, *gözcü* ve *işçi* arıların, yuva-kaynak arasındaki tekrarlı uçuşuyla devam eder. Bu süreçte *işçi* arılar, uğradıkları kaynağın civarında daha elverişli bir besin kaynağı bulurlarsa, eski kaynağı bırakıp bu yeni kaynaktan besin toplamaya başlarlar. Kovana döndüklerinde ise, besin kaynağının yerini *gözcü* arılara tarif ederler.

Gözcü arılar, *işçi* arıların herbirinden bilgi alır ve en iyi kaynağa gitmeye çalışır. Onlar da tıpkı işçiler gibi, daha uygun bir kaynak bulduklarında eski kaynağı terk ederler. Kaynaklardaki besin tükendiğinde ise, görevli bu arılar, *kâşif* arı rolüyle yeni besin kaynakları araştırırlar.

Algoritmik modelde besin kaynakları, muhtemel çözümleri temsil eder. Dolayısıyla kâşif arı safhasında oluşturulan her bir rastsal çözüm, çözüm matrisine kaydedilir. Ardından iteratif adımlardaki her bir çevrimde, oluşturulan çözüm sayısı (ÇS) kadar işçi arı ve bir o kadar da gözcü arı prosedürü ile yeni çözümler türetilir. İşçi ya da gözcü arı tarafından ele alınan bir çözüm ile daha iyi bir çözüm türetildiğinde, mevcut çözüm, çözüm matrisinden silinirek yerine yeni çözüm kaydedilir. Bu yeni çözümün başarısızlık sayacı sıfırlanır. Aksi taktirde, yeni çözüm türetemeyen bir çözümün başarısızlık sayacı bir artırılır. Gözcü arı safhası da tamamlandıktan sonra, güncel çözümlerin başarısızlık sayaçları limit değerle karşılaştırılır. Başarısızlık sayacı limit seviyesine ulaşmış çözümler, çözüm matrisinden silinir ve bu çözümler yerine, başarısızlık sayacı sıfırlanmış yeni rastsal çözümler kaydedilir.

Kâşif arı safhasında, istenen çözüm sayısınca rastgele çözüm için (1) eşitliği kullanılır.

$$Ç_{m,i} = a_i + rastgele(0,1) * (a_i - ü_i) \quad (1)$$

$Ç_{m,i}$ çözüm *m*'nin *i.* elemanı olmak üzere, a_i *i.* elemanın alt sınırını, u_i ise üst sınırını ifade eder. Başlangıç çözümlerinin oluşturulmasının ardından, işçi arılar, (2) ile bu çözümlerden yeni çözümler türetirler.

$$Y_{m,i} = Ç_{m,i} + \phi_{m,i}(Ç_{m,i} - Ç_{r,i}) \quad (2)$$

Denklem (2)'de, Y_m türetilen yeni çözüm, $Ç_m$ sıradaki işçi arının görevli olduğu çözüm, $Ç_r$ rastgele seçilmiş bir çözüm, $\phi_{m,i}$ ise [-1, 1] kapalı aralığında rastgele türetilen bir sayıdır. Y_m çözümünün uygunluk değeri ($uyg(Y_m)$) (3) denklemiyle hesaplanır. (3) denklemindeki $f(Y_m)$, problemin amaç fonksiyonundan elde edilir.

$$uyg(Y_m) = \begin{cases} 1/(1+f(Y_m)) & eğer\ (f(Y_m)) \geq 0 \\ 1 + mutlak(f(Y_m)) & eğer\ (f(Y_m)) < 0 \end{cases} \quad (3)$$

Eğer $uyg(Y_m) > uyg(Ç_m)$ ise, $Ç_m$ çözüm matrisinden silinir ve bu çözüm alanına Y_m kaydedilir. Böylece çözümün başarısızlık sayacı (bs_m) da 0 olarak güncellenir. Aksi taktirde ($uyg(Y_m) \leq uyg(Ç_m)$) ise $bs_m = bs_m + 1$ işlemiyle başarısızlık sayacı artırılır.

Gözcü arı safhasında daha iyi çözümlerin değerlendirilebilmesi için, işçi arı safhasında güncellenen çözümler, rulet tekerleğine yerleştirilir. Bu bağlamda çözümlerin seçilme olasılıkları (*S*), (4) denklemiyle hesaplanır.

$$s_m = \frac{uyg(Ç_m)}{\sum_{n=1}^{ÇS} uyg(Ç_n)} \quad (4)$$

Gözcü arılar da yeni çözüm türetirken (2) denklemini kullanır. İşçi arı prosedüründen tek farkı, denklemdeki $Ç_m$'in (4) teki olasılığa göre seçilmesidir. Gözcü arılar da mevcut ve yeni çözümlerin uygunluk değerlerini karşılaştırarak aç gözlü seçim yaparlar.

Arama süreci, sırasıyla işçi ve gözcü arıların yeni çözümler türetmeleriyle devam eder. Bu süreçte başarısızlık sayacı *limit* değerine erişen çözümler, çözüm matrisinden silinir ve yerine (1) denklemiyle oluşturulacak rastgele çözümler kaydedilir. Çözüm matrisine kaydedilen her yeni çözümden sonra ilgili çözümün başarısızlık sayacı da sıfırlanmaktadır.

2.2. fYAK Algoritması

KKO algoritmaları, mevcut çözümlerden yeni çözümler türetmek yerine, çözüm bileşenleri arasındaki bağlantıyı değerlendirir ve en uygun dizilimi oluşturmaya çalışır (Marco Dorigo, Maniezzo, & Colorni, 1991). Bu yaklaşım, KKO algoritmalarına etkin bir sömürü yeteneği kazandırmaktadır. Çözüm arama sürecinde, daha iyi sonuçlar elde eden dizi elemanlarının arasındaki yapay feromon seviyesi (τ) yoğunlaştırılmakta, birbirinden uzak elemanlar arası salgı miktarı ise azaltılmaktadır. Feromon, tüm KKO algoritmaları için temel bileşendir.

fYAK'ta işçi arılar, besin toplama sürecinde, polen elde ettikleri çiçeklerin arasında feromon bırakırlar. Sonradan gelen gözcü arılar, hangi çiçeği tercih edeceklerine karar verirken, feromon yoğunluğunu da dikkate alırlar. Algoritma işleyişinde işçi arı safhası tamamlandıktan sonra, düğümler arası feromon yoğunlukları güncellenmektedir. Bu işlem iki aşamada gerçekleştirilir. İlk aşamada (5) kullanılarak, mevcut feromon seviyeleri, buharlaşma oranıyla (ρ $(0 \leq q0 \leq 1)$) azaltılır ve işçi arıların kullandığı düğümler arası feromon seviyesi, çözümün başarısıyla ilişkili olarak artırılır.

$$\tau_{i,j} \leftarrow (1-\rho)\,.\,\tau_{i,j} + \sum_{k=1}^{ÇS} \Delta\tau_{i,j}^{k} \quad (5)$$

Bu bağlamda, *i-j* düğümlerini kullanarak çözüm oluşturan *k* işçi arısının, bu düğümler arasında bıraktığı feromon miktarı ($\Delta\tau_{i,j}^{k}$), $1 / f(Ç_k)$ seviyesindedir. Feromon güncellemenin ikinci aşamasında ise, (6) ifadesiyle, ilgili çevrime kadar en iyi çözümü üretmiş karınca güzerhagı için güncelleme yapılır. ($\Delta\tau_{i,j}^{en\ iyi} = \frac{1}{} f(Ç_{en\ iyi})$)

$$\tau_{i,j} \leftarrow (1-\rho)\,.\,\tau_{i,j} + \rho.\Delta\tau_{i,j}^{en\ iyi} \quad (6)$$

fYAK algoritması için benimsenen geçiş kuralı, Karınca Koloni Sistemindeki (KKS) (M Dorigo & Gambardella, 1997) gibidir. Buna göre, *i* düğümündeki bir gözcü arı için sonraki *j* düğümünü seçerken iki alternatif söz konusudur. Alternatiflerin seçilme olasılıkları, algoritmanın $q0$ $(0 \leq q0 \leq 1)$ parametresiyle belirlenir. [0-1] aralığında rastgele seçilen *q* değeri, $q \leq q0$ seviyesindeyse ilk

alternatif kullanılır. (7)'da ifade edildiği gibi, *i* düğümünden sonra seçilebilecek diğer düğümlerin ($u \in V$) seçilme olasılıkları değerlendirilir ve en büyük olasılığa sahip düğüme gidilir.

$$j = \underset{u \in V}{max}\{{\tau_{i,u}}^{\alpha}.{\eta_{i,u}}^{\beta}\} \quad (7)$$

(7) ifadesinde, $\eta_{i,u}$ düğümler arası uzaklığın ($\delta_{i,u}$) tersidir ($\eta_{i,u} = \delta_{i,u}$). α feromon seviyesinin etkisini, β ise düğümler arası mesafenin etkisini belirleyen sezgisel parametrelerdir.

$q > q0$ durumunda ise (8) ile muhtemel düğümlerin seçilme olasılıkları dikkate alınarak tercih yapılır.

$$p_{i,j} = \frac{{\tau_{i,j}}^{\alpha}.{\eta_{i,j}}^{\beta}}{\sum_{u \in V} {\tau_{i,u}}^{\alpha}.{\eta_{i,u}}^{\beta}} \quad (8)$$

Çözüm tamamlandıktan sonra, bu yeni çözüm dizisi, çözüm matrisindeki dizilerle karşılaştırılır. Dizi elemanları ve sıralanışı itibariyle, oluşturulan yeni diziye en çok benzeyen çözüm belirlenir ve iki çözüm arasında uygunluk değerlerine göre aç gözlü yaklaşım uygulanır.

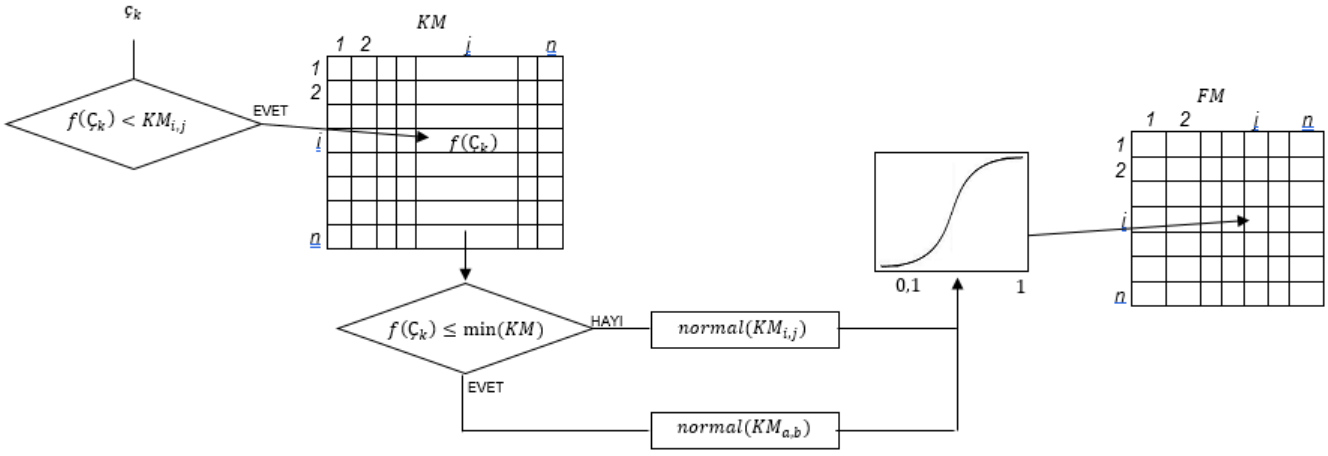
2.3. Önerilen Metot: Geliştirilmiş Feromonal Yapay Arı Koloni (gfYAK) Algoritması

Feromon salgısını kullanarak çözüm oluşturan KKO algoritmaları için iki önemli durum söz konusudur: düğümler arasına atanacak feromon miktarları, çözümle ilişkili olabilmeli ve algoritma, başarılı çözüm bölgelerine yönlendirilebilmelidir. Geliştirilen KKO algoritmaları, bu durumlar dikkate alınarak çeşitlendirilmiştir (Kwang Mong Sim & Weng Hong Sun, 2003). gfYAK yönteminde, düğümler arasına yerleştirilecek feromon seviyesi için feromon matrisinden (*FM*) ayrı olarak, çözümlere ait *f(Ç)* değerlerinin kaydedildiği kenarlar matrisi (*KM*) kullanılmakta ve algoritmanın daha etkin arama bölgelerine yönlendirilebilmesi için transfer fonksiyonlarından yararlanılmaktadır.

İşçi arı safhası tamamlandıktan sonra, tüm çözüm dizileri, KM'de, bileşenlerine karşılık gelen alanlarla karşılaştırılır. Örneğin minimizasyon problemi çözümünde *i-j* kenarını kullanan *k* işçi arısı için $f(Ç_k)$ değeri, KM'deki *i-j* hücresindeki değerden daha küçükse, *i-j* hücresindeki değer *f(Ç)* olarak güncellenir. Ardından (9) ile, $f(Ç_k)$, KM'deki en düşük ve en büyük değerlere göre normalize edilerek [0.1, 1] aralığına ölçeklenir.

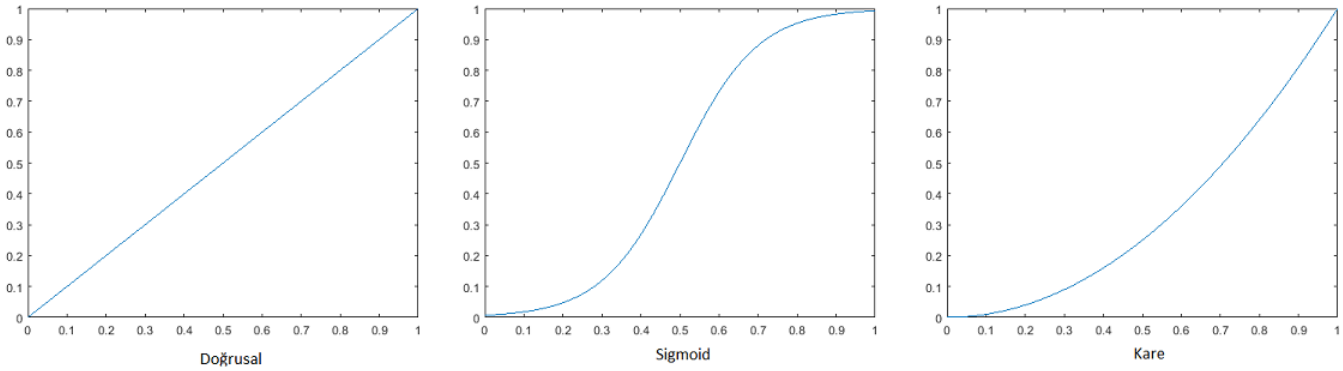
$$normalize_{i,j} = 1 - \frac{[f(Ç_k) - min(KM)] * 0.9}{[maks(KM) - min(KM)]} \quad (9)$$

KM'de *i-j* hücresi için normalize edilen değerin FM'deki feromon seviyesi karşılığını hesaplamak için, seçilen transfer fonksiyonu kullanılır. Eğer k işçi arısı, ilgili çevrime kadarki en iyi sonucu bulmuşsa, FM'deki tüm alanlar KM'ye göre güncellenir, aksi halde feromon güncelleme, yalnızca k işçi arısının kullandığı hat için hesaplanacaktır. *k* işçi arısının, *i-j* kenarında bırakacağı feromon miktarı hesabı, Şekil 1'de resmedilmektedir.



Şekil 1. gfYAK Yönteminde i-j Kenarını Kullanan k İşçi Arısının Bırakacağı Yapay Feromon Miktarı

KM kullanılarak, çözüm başarısı, çözümü oluşturan kenarlarla doğrudan ilişkilendirilmektedir. Ancak gözcü arıların düğüm seçimlerinde, transfer fonksiyonu daha çok etkilidir. Bu bağlamda, problem karmaşıklığına, parametre karakteristiğine ve tahmini çözüm alanına bağlı olarak farklı transfer fonksiyonları tercih edilebilir. Çalışma kapsamında, Şekil 2'deki transfer fonksiyonları kullanılmıştır.

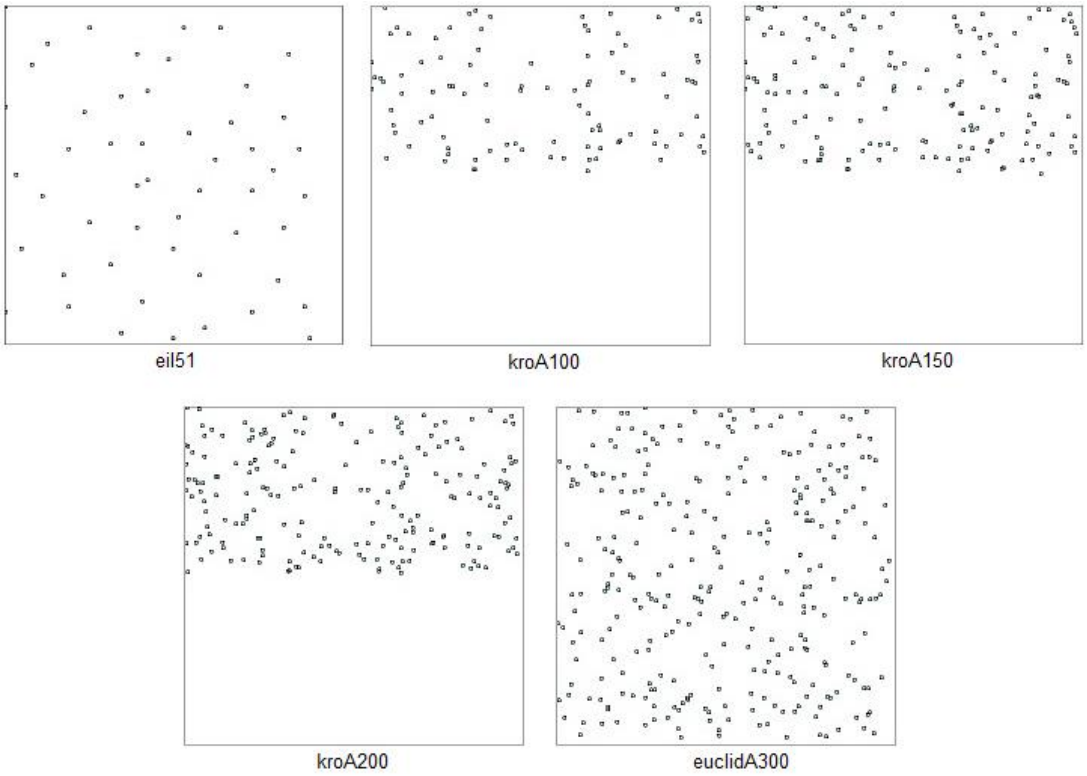


Şekil 2. Çalışma Kapsamında gfYAK İçin Kullanılan Transfer Fonksiyonları

gfYAK yönteminde, gözcü arıların çözüm oluşturma yaklaşımı ve oluşturulan çözümün mevcut çözümlerle kıyaslanarak aç gözlü seçim stratejisi fYAK yöntemindeki gibidir.

3. Araştırma Sonuçları ve Tartışma

Çalışmada, klasik YAK, fYAK ve gfYAK algoritmaları, .net platformunun C# dili ile kodlanmıştır. Geliştirilen yazılım, i7-5600U CPU 2.60 GHz x64 tabanlı işlemci ve 8 GB RAM donanıma sahip makinede, Windows 8.1 64 bit işletim sistemi ve Framework 4.5 zemininde çalıştırılmıştır. Algoritmaların çözüm üretme başarılarını ve yakınsama performanslarını test etmek için, GSP örnekleri olarak TSPLIB kütüphanesinden (http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/) indirilebilen, 51 şehirli eil51, 100 şehirli kroA100, 150 şehirli kroA150, 200 şehirli kroA200 ve 300 şehirli euclidA300 test problemleri seçilmiştir. Seçilen test problemlerinde, düğümlerin yerleşimi Şekil3'te gösterilmektedir.



Şekil 3. eil51, kroA100, kroA150, kroA200 ve euclidA300 problemlerindeki düğümlerin yerleşimi

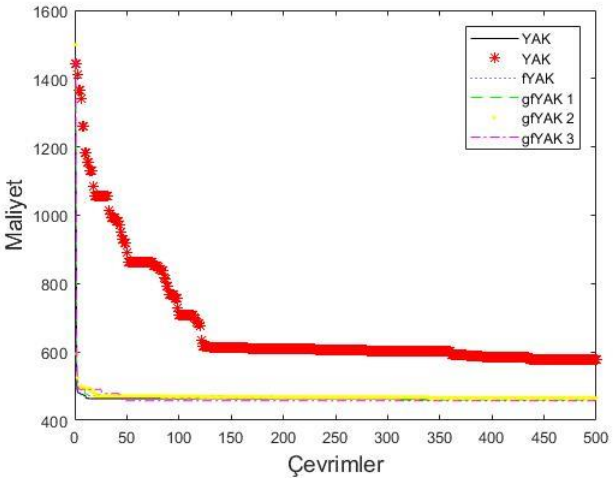
Adil bir karşılaştırma için algoritmalar Tablo 1'de verilen eşit koloni boyutlarıyla ve eşit sürelerde çalıştırılmıştır. n problemdeki şehir sayısı olmak üzere, algoritmalar, her bir test problemi için birbirinden bağımsız olarak n saniye çalışma süresiyle 30'ar kez çalıştırılmıştır. Analizler, YAK algoritması için en uygun limit değerinin (10) eşitliğindeki seviyede olduğunu göstermektedir (Akay & Karaboga, 2009). Buna göre YAK algoritmaları için limit değer $n^2/2$ olarak hesaplanmaktadır.

$$limit = (koloni\ boyutu * çözümdeki\ bileşen\ sayısı)/2 \quad (10)$$

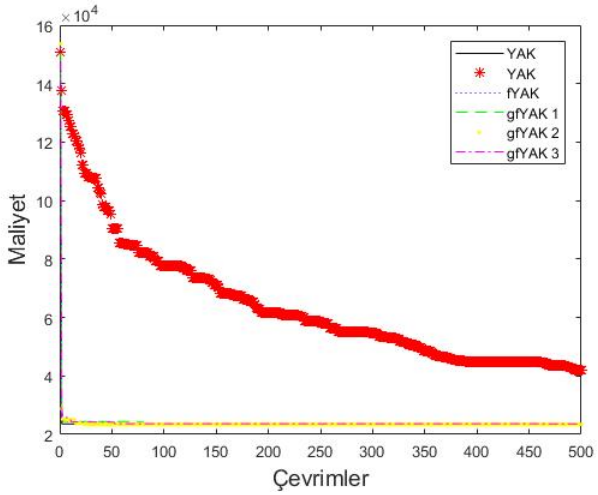
Tablo 1. Algoritmalar İçin Seçilen Parametre Değerleri

	Koloni Boyutu	Limit	α	β	q	q0	Transfer Fonksiyonu
KKS	n	-	1	5	0.1	0.9	-
YAK	n	$n^2 / 2$	-	-	-	-	-
fYAK	n	$n^2 / 2$	1	5	0.1	0.9	-
gfYAK_1	n	$n^2 / 2$	1	5	0.1	0.9	Doğrusal
gfYAK_2	n	$n^2 / 2$	1	5	0.1	0.9	Sigmoid
gfYAK_3	n	$n^2 / 2$	1	5	0.1	0.9	Kare

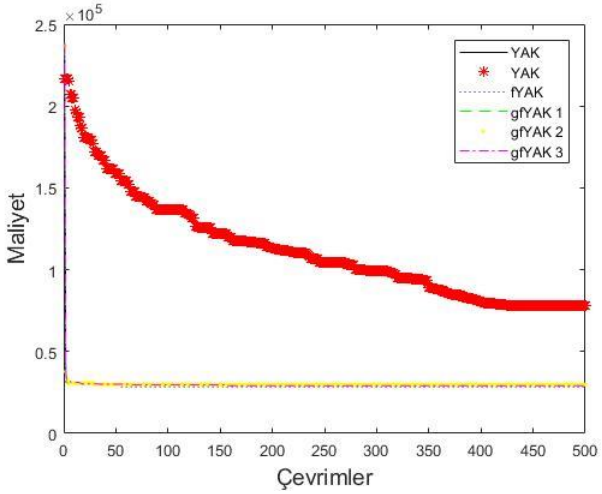
Algoritmaların problem çözümlerinde, ilk 500 iterasyondaki yakınsama performansları Şekil 4-8'de gösterilmektedir.



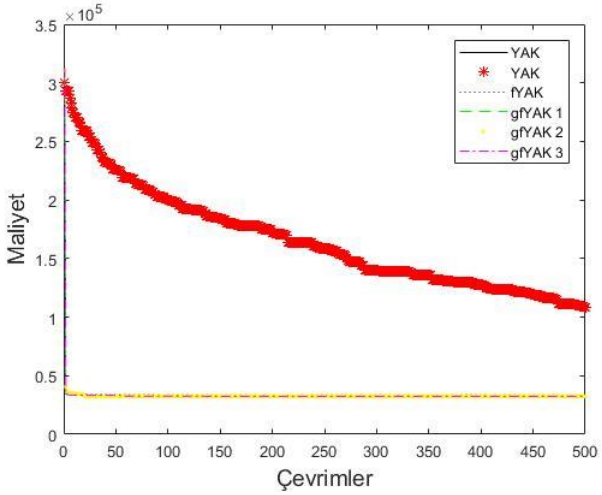
Şekil 4. Algoritmaların eil51 Test Problemi İçin Elde Ettiği Sonuçlar



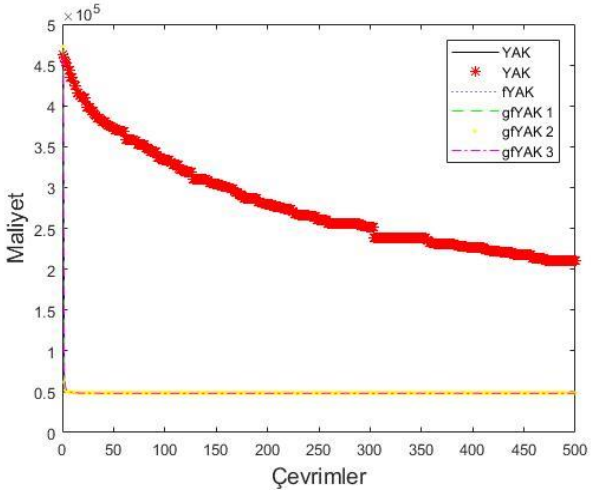
Şekil 5. Algoritmaların kroA100 Test Problemi İçin Elde Ettiği Sonuçlar



Şekil 6. Algoritmaların kroA150 Test Problemi İçin Elde Ettiği Sonuçlar



Şekil 7. Algoritmaların kroA200 Test Problemi İçin Elde Ettiği Sonuçlar



Şekil 8. Algoritmaların euclid300 Test Problemi İçin Elde Ettiği Sonuçlar

Şekil 4-8'deki grafikler incelendiğinde, feromon yaklaşımının klasik YAK algoritmasının yakınsama performansını önemli ölçüde geliştirdiği görülebilmektedir. Feromon entegre edilen YAK algoritmaları, KKS algoritması seviyesinde yakınsama hızı kazanmış ve YAK'ın keşif yeteneğiyle birlikte daha başarılı çözümler üretebilmişlerdir. Bu bağlamda klasik KKS, klasik YAK, fYAK ve gfYAK algoritmalarının seçilen problem çözümlerinden elde ettikleri, en iyi, en kötü sonuçlar ve 30'ar denemede elde edilen sonuçların hesaplanan ortalama ve standart sapma değerleri Tablo 2'de gösterilmektedir.

Tablo 2. Algoritmaların elde ettikleri sonuçlar

		eil51	**kroA100**	**kroA150**	**kroA200**	**euclid300**
KKS	*En İyi*	455	22839	27860	32785	47026
	En Kötü	460	26709	32011	33608	48454
	Ortalama	451	25083	28574	32994	47565
	Standart Sapma	5	369	658	783	948
YAK	*En İyi*	435	23647	28084	35798	48893
	En Kötü	457	25993	33453	38037	51278
	Ortalama	442	28993	29157	37715	49962
	Standart Sapma	4	427	703	904	1037
fYAK	*En İyi*	445	21917	27580	32964	47218
	En Kötü	464	24057	29857	33792	48704
	Ortalama	459	22694	29154	33107	47869
	Standart Sapma	9	274	599	814	862
gfYAK 1	*En İyi*	447	21856	27890	32671	46872
	En Kötü	468	24804	30074	33386	47939
	Ortalama	456	22470	28473	32819	47103
	Standart Sapma	10	266	603	577	814
gfYAK 2	*En İyi*	445	21904	27701	31779	46717
	En Kötü	464	25581	29940	32936	47747
	Ortalama	453	24707	28792	32631	46969
	Standart Sapma	9	278	586	558	746
gfYAK 3	*En İyi*	441	21746	27225	31469	46273
	En Kötü	458	24759	29753	32707	47037
	Ortalama	451	22879	28037	32435	46739
	Standart Sapma	7	225	556	495	676

Tablo 2'deki sonuçlara bakarak YAK algoritmasının küçük boyutlu problemler için daha başarılı çözümler ürettiği, problem boyutu arttığında çözüm başarısında azalma olduğu gözlemlenmektedir. Tablo 2'den feromon özelliği eklenen YAK'ın daha başarılı çözümler üretebildiği daha net görülmektedir. gfYAK modelinde feromon seviyesinin transfer fonksiyonlarıyla belirlenmesinin fYAK'ı daha da güçlendirdiği görülmektedir. Sonuçlar, çözüm ortalamaları ve standart sapma verileriyle birlikte değerlendirildiğinde tranfer fonksiyonlarıyla daha kararlı çözümler elde edilebildiği de söylenebilmektedir.

4. Sonuç

Optimizasyon problem çözümleri için, biyolojik yaşamdan ilham alan ve özellikle de sürü zekasını taklit eden metasezgisel yöntemlerin popülerliği gün geçtikçe artmaktadır. Koloni halinde yaşayan kuş, balık, balina, ateş böceği, kanguru gibi hayvanlar ile termit, karınca, arı, kelebek gibi mikroorganizmalar bu tür algoritmaların geliştirilmesine ilham kaynağı olmuşlardır. Çoğu zaman, algoritmaların vurgulayıcı bileşenleri, birlikte kullanılarak geliştirilen melez yöntemler, yalın algoritmadan daha başarılı olabilmektedir. Bu düşünce temelinde, YAK algoritmasına KKO'nun sömürü performansını güçlü kılan feromon salgısı eklenerek fYAK algoritması geliştirilmiştir. Çalışma kapsamında, fYAK algoritmasını daha verimli hale getirebilmek için, düğümler arası yakınlığı çözüm başarısıyla daha çok ilişkilendiren ikinci bir matris ve algoritmanın arama yönünü bu matristeki değerleri referans alarak hesaplayan transfer fonksiyonları kullanılmıştır.

Uygulama, farklı boyutlardaki GSP örnekleri üzerinde test edilmiş, algoritmanın yakınsama durumu ve elde edilen sonuçlar klasik YAK, klasik KKO ve fYAK algoritma sonuçlarıyla karşılaştırılmıştır. Buna göre, tercih edilen transfer fonksiyonlarına göre gfYAK yönteminin, fYAK performansını anlamlı seviyede geliştirdiği gözlemlenmiştir.

Kaynakça

Akay, B., & Karaboga, D. (2009). Parameter Tuning for the Artificial Bee Colony Algorithm (Vol. 5796, pp. 608–619). https://doi.org/10.1007/978-3-642-04441-0_53

Akay, B., & Karaboga, D. (2015). A survey on the applications of artificial bee colony in signal, image, and video processing. *Signal, Image and Video Processing*, *9*(4), 967–990. https://doi.org/10.1007/s11760-015-0758-4

Bansal, J. C., Sharma, H., & Jadon, S. S. (2013). Artificial bee colony algorithm: A survey. *International Journal of Advanced Intelligence Paradigms*, *5*(1–2), 123–159. https://doi.org/10.1504/IJAIP.2013.054681

Barnebau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm Intelligence From Natural to Artificial Systems*. New York: Oxford University Press.

Beni, G., & Wang, J. (1993). Swarm Intelligence in Cellular Robotic Systems. In P. Dario, G. Sandini, & Aebischer Patrick (Eds.), *Robots and Biological Systems: Towards a New Bionics?* (pp. 703–712). Springer Berlin Heidelberg. https://doi.org/https://doi.org/10.1007/978-3-642-58069-7_38

Dorigo, M, & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66. https://doi.org/10.1109/4235.585892

Dorigo, Marco, Maniezzo, V., & Colorni, A. (1991). *Positive feedback as a search strategy*. Milano, Italy.

Ekmekci, D. (2019a). A Pheromonal Artificial Bee Colony (pABC) Algorithm for Discrete Optimization Problems. *Applied Artificial Intelligence*, *33*(11), 935–950. https://doi.org/10.1080/08839514.2019.1661120

Ekmekci, D. (2019b). An Ant Colony Optimization Memorizing Better Solutions (ACO-MBS) for Traveling Salesman Problem. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)* (pp. 1–5). IEEE. https://doi.org/10.1109/ISMSIT.2019.8932768

Karaboga, D. (2005). *An Idea Based on Honey Bee Swarm for Numerical Optimization*. Kayseri, Turkey. Retrieved from https://www.researchgate.net/publication/255638348_An_Idea_Based_on_Honey_Bee_Swarm_for_Numerical_Optimization_Technical_Report_-_TR06

Kwang Mong Sim, & Weng Hong Sun. (2003). Ant colony optimization for routing and load-balancing: survey and new directions. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *33*(5), 560–572. https://doi.org/10.1109/TSMCA.2003.817391

Monteiro, M. S. R., Fontes, D. B. M. M., & Fontes, F. A. C. C. (2012). *Ant Colony Optimization: a literature survey*. *FEP Working Papers*. Retrieved from http://ideas.repec.org/p/por/fepwps/474.html