*8. Türkiye Ekonometri ve İstatistik Kongresi*
*24-25 Mayıs 2007 – İnönü Üniversitesi*
*Malatya*

# KARINCA KOLONİSİ ALGORİTMASI İLE GEZEN SATICI PROBLEMİNİN ÇÖZÜMÜ

Hasan SÖYLER[1] Timur KESKİNTÜRK[2]

**Özet:** Karınca kolonisi algoritması (KKA), gezgin satıcı ve benzer yapıdaki problemlerin çözümü için geliştirilen sezgisel bir yöntemdir. Koloniler halinde yaşayan karıncalar, yuvalarıyla yiyecek arasında en kısa yolu bulma kabiliyetine sahiptirler. Geçtikleri yollara bıraktıkları feromon denen izler sayesinde yollarını bulan karıncaların gerçek hayattaki bu davranışlarından yola çıkılarak geliştirilen algoritma ile simetrik ve asimetrik gezen satıcı problemlerinde (GSP) uygun ve iyi çözümler bulunmaktadır. Oluşturulan yapay karıncalar kullanılarak ve karıncaların belli kurallarla geçiş yaptığı yollarda yapay feromon güncellemesi yapılarak en kısa yol iterasyonlar boyunca araştırılmaktadır. Bu çalışmanın amacı, ekonomik ve askeri alanda oldukça önemli bir yere sahip lojistik-dağıtım konusunda geliştirilmiş olan karınca kolonisi algoritmasının tanıtılması, çalışma şeklinin ve prensiplerinin gösterilmesidir. Farklı karınca kolonisi algoritmalarından bazılarına da değinildiği makalenin sonunda, örnek bir GSP problemine yer verilmiş ve sonuçlar diğer yöntemlerin sonuçları ile karşılaştırılmıştır.

**Anahtar kelimeler:** Karınca kolonisi algoritması, gezen satıcı problemi, feromon güncellemesi, tur uzunluğu.

## SOLUTİON OF TRAVELLİNG SALESMAN PROBLEM WİTH ANT COLONY ALGORİTHM

**Abstract:** Ant Colony Algorithm is a heuristic model that is developed for solving traveling salesman type problems. Ants which live in colonies are capable of finding the shortest path between a food source and their nests. They deposit a certain amount of pheromone -a chemical substance- on their way, in order to exchange information about the path that should be followed. This behavior of real ants has inspired the Ant Colony Algorithm which can be used for deriving approximate and optimal solutions for Symmetric or Asymmetric traveling salesman problems (TSP). By using artificial ants and performing artificial pheromone update, the shortest path is searched during iterations. The aim of this study is to introduce Ant Colony Algorithm for logistic distribution, which is very important for the economic and military fields, and demonstrate its principals several types of Ant Colony Algorithms are mentioned at the end of the paper as well as an example of TSP and comparison of its solution among different methods.

**Keywords:** Ant Colony Algorithm, Travelling salesman problem, pheromone update, tour length

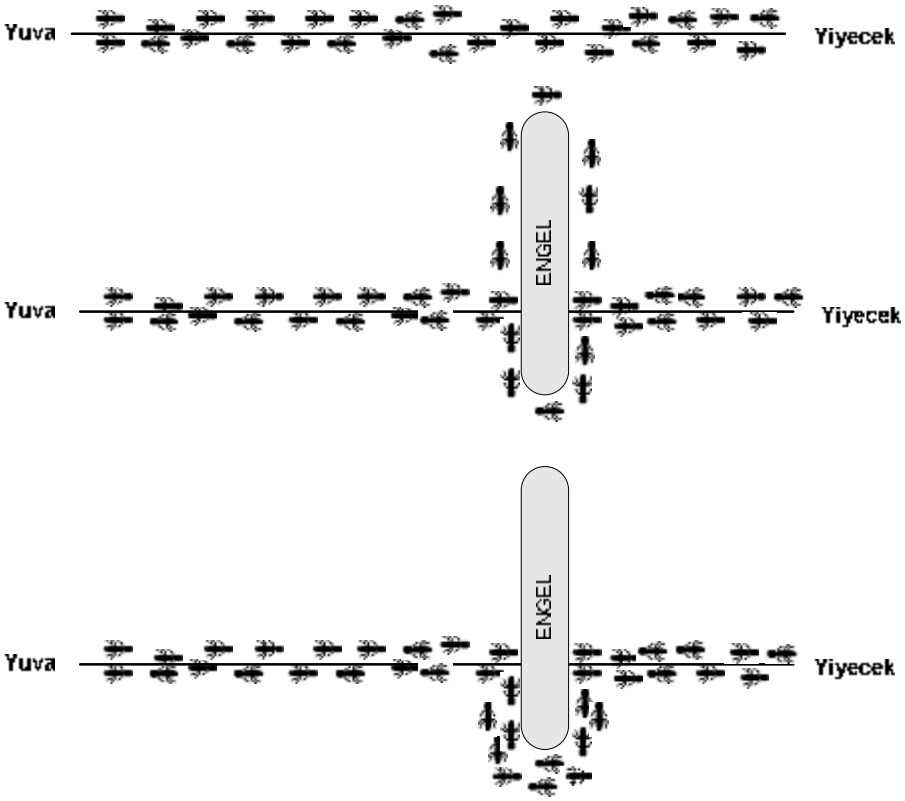
[1] İnönü Üniversitesi İİBF Ekonometri Bölümü Araştırma Görevlisi, hsoyler@inonu.edu.tr
[2] İstanbul Üniversitesi, İşletme Fakültesi, Araştırma Görevlisi, tkturk@istanbul.edu.tr

## 1.GİRİŞ

Bilim adamları, böcek davranışlarını inceleyerek başarılı optimizasyon algoritmaları geliştirmişlerdir. Bu teknikler birçok bilimsel alanda ve mühendislik problemlerinde başarıyla uygulanmıştır. Teknikler, statik problemlerdeki yüksek performanslarına ilaveten, dinamik özellik gösteren problemlerde de yüksek derecede esnekliğe sahiptirler [Bonabeau,2000]. Karınca algoritmaları ilk olarak Dorigo ve meslektaşları tarafından; gezgin satıcı problemi (GSP) ve kuadratik atama (QAP) gibi zor optimizasyon problemlerinin çözümü için geliştirilmiştir. Optimizasyon problemlerinin çözümü amacıyla karınca algoritmaları üzerine birçok çalışma devam etmektedir. Bu çalışma alanlarından bazıları, komünikasyon ağlarının belirlenmesi, grafik renklendirme, iş çizelgeleme vs.dir [Dorigo,1999].

Karıncalar, yiyecek kaynaklarından yuvalarına en kısa yolu görme duyularını kullanmadan bulma yeteneğine sahiptirler. Aynı zamanda, çevredeki değişime adapte olma yetenekleri vardır. Dış etkenler sonucu takip ettikleri mevcut yol artık en kısa yol değilse, yeni en kısa yolu bulabilmektedirler [Dorigo,1997].

ŞEKİL 1'de de görüldüğü gibi karıncalar, başlangıçta düz bir hattı takip etmekte ve bu esnada feromon olarak adlandırılan bir maddeyi yol güzergahına bırakarak kendilerinden sonra gelen karıncaların yollarını bulmalarını kolaylaştırmaktadırlar. Önlerine bir engel konulduğunda feromonları takip edemediklerinden, karıncalar gidebilecekleri iki yoldan birini öncelikle rastsal olarak seçmektedirler. Kısa olan yoldan birim zamandaki geçiş daha fazla olacağından bırakılan feromon miktarı da daha fazla olur. Buna bağlı olarak, zaman içerisinde kısa olan yolu tercih eden karıncaların sayısında artış olur. Belli bir süre sonra tüm karıncalar kısa yolu tercih ederler.




**ŞEKİL 1. Gerçek karıncaların en kısa yolu bulması**

Başta rastsal hareket eden karıncaların izleri kontrol ederek yüksek olasılılıkla izlerin yoğun olduğu yönü takip etmesi otokatalitik bir davranış şeklidir ve karıncaların karşılıklı etkileşiminde sinerjik bir etki vardır. Algoritma, karınca kolonilerinden esinlenerek geliştirildiğinden sisteme, karınca sistemi (KS), algoritma ise karınca kolonileri algoritması (KKA) olarak adlandırılır. Karınca kolonileri optimizasyon problemlerinde kullanılır. Karınca sistemindeki karıncalar doğal karıncalardan farklıdır. Hafızaya sahiptirler, tamamen kör değildirler ve zamanın kesikli olduğu bir çevrede yaşarlar [Dorigo,1991].

## 2. KARINCA KOLONİSİ ALGORİTMASI

Yapay karıncalardan oluşan Karınca Kolonisi Algoritması, yapay feromon izlerinin güncelleştirilmesiyle tekrarlanan bir yapıya sahiptir. Algoritmanın çalışma sürecinde, karıncalar tarafından güncellenen feromon izleriyle iyi bir çözümün bulunması için bilgi oluşturulmakta ve her iterasyonda bu bilgiler güncellenmektedir. Karınca kolonisi algoritmasına ait döngü ŞEKİL 2' deki gibidir.

Algoritmanın çalışma sürecinde temel işlemler, yapay karıncaların turları sonunda geçmiş oldukları yolların feromon miktarlarının arttırılması, belirli bir oranda feromon buharlaşmasının gerçekleştirilmesi, en iyi çözümün bulunması, buna bağlı olarak global feromon güncellemesinin yapılması ve karıncaların yenilenen bu feromon miktarlarına bağlı olarak yeni turlarını gerçekleştirmeleridir. Tüm bu işlemlere ait hesaplama yöntemleri ayrıntılarıyla alt başlıklarda verilmiştir.

Adım 1 : Başlangıç feromon değerleri belirlenir.
Adım 2 : Karıncalar her düğüme rastsal olarak yerleştirilir.
Adım 3: Her karınca, sonraki şehri denklemde verilen lokal arama olasılığına bağlı olarak seçmek suretiyle turunu tamamlar.
Adım 4: Her karınca tarafından katedilen yolların uzunluğunu hesaplanır ve lokal feromon güncellemesi yapılır.
Adım 5: En iyi çözüm hesaplanır ve global feromon yenilemesinde kullanılır.
Adım 6: Maksimum iterasyon sayısı yada yeterlilik kriteri sağlanana kadar Adım 2' ye gidilir.

**ŞEKİL 2:Karınca Kolonisi Algoritması**

### 2.1 Geçiş Kuralı

KKA' da bir tur esnasında, i noktasında bulunan k karıncası için, sonraki j noktasını seçerken iki alternatif yol söz konusudur. İlk alternatif, gidebileceği yollar içersinden feromon miktarlarına bağlı olarak hesaplanan seçim değerlerinden maksimum olanını seçmesidir. Genellikle bu yolla tercih yapma olasılığı ($q_0$) %90 olarak belirlenmektedir. İkinci alternatifte ise yollardaki feromon miktarı göz önüne alınarak oluşturulan olasılık dağılımına bağlı olarak yollar seçilir.

Aşağıda bu geçiş kuralına göre i noktasında bulunan k karıncasının u adet alternatif noktadan hangisine gideceğinin belirlendiği formül görülmektedir:

$$j = \max_{u \in J_k(i)} \left\{ \left[\tau(i,u)\right]^{\alpha} \times \left[\eta(i,u)\right]^{\beta} \right\} \quad \text{eğer} \quad q \le q_0 \qquad (1)$$

Burada $t(i,u)$, $(i,u)$ hattındaki feromon izidir. $h(i,u)=1/d(i,u)$ i noktasından u noktasına uzaklığın tersidir. $J_k(i)$ i noktasındaki k karıncası tarafından henüz gidilmemiş şehirleri temsil etmektedir. $\beta$ $(b>0)$ feromonun güncellenmesinde, uzaklığın göreli önemliliğini belirleyen parametredir. $q_0$ $(0 \le q_0 \le 1)$ çözüm uzayını araştırmanın göreli önemliliğini gösteren parametredir.

$q \ge q_0$ olması durumunda ise ikinci geçiş kuralı uygulanır. Bu kurala göre gidilecek bir sonraki nokta hesaplanan seçim değerlerine bağlı olarak rastsal olarak seçilmektedir. Dolayısıyla feromon miktarının daha yoğun olduğu yolların seçilme olasılığı daha fazla olacaktır. Gidilebilecek yolların seçilme olasılıkları aşağıdaki formülle hesaplanmaktadır:

$$p_k(i,j)=\begin{cases}\dfrac{[\tau(i,j)]^{\alpha}\times[\eta(i,j)]^{\beta}}{\sum_{u\in j_k(i)}[\tau(i,u)]^{\alpha}\times[\eta(i,u)]^{\beta}} & \text{eğer } j\in J_k(i)\\ 0 & \text{Diğer durumlarda}\end{cases} \qquad (2)$$

Turlar esnasında hem sezgisel bilgiden ve hem de feromon bilgisinden faydalanılır.

### 2.2 Feromon güncellemesi:

Tüm karıncalar turlarını tamamladıktan sonra feromon miktarları güncellenmektedir. İlk olarak tüm yollardaki feromonlar, belirlenen oranda (buharlaşma oranı) buharlaştırılmaktadır. Daha sonra karıncaların geçiş yapmış oldukları yollardaki feromon miktarları, o yolu kullanan karıncanın toplam yol uzunluğuyla ters orantılı olarak arttırılmaktadır [Stützle, 2000]. Böylelikle daha kısa yola sahip karıncaların kullandıkları yollardaki feromon miktarları daha fazla artış göstermektedir. Feromon güncellemesi iki yolla yapılabilmektedir. Bunlar Lokal feromon güncellemesi ve Global Feromon güncellemesidir.

#### 2.2.1 Lokal Feromon Güncellemesi

Tüm karıncalar turlarını tamamladıktan sonra, eski feromon miktarları belli bir oranda buharlaştırılır, her bir karıncanın turu boyunca geçiş yapmış olduğu yollarda belli bir miktarda feromon artışı sağlanır. Bu işlemler aşağıdaki formüle göre yapılmaktadır:

$$t_{ij}(t+1)=(1-r)\,t_{ij}(t)+\sum_{k=1}^{m}\Delta t_{ij}^{k}(t+1) \qquad (3)$$

burada $t_{ij}(t)$ başlangıç feromon düzeyidir ve $r$, $(0 \le r \le 1)$ feromon buharlaşma parametresidir. Lokal güncelleme kuralı turları dinamik olarak değiştirerek geçiş yapılan yolları cazip hale getirir. Karıncalar farklı yollardan giderken, yüksek bir olasılıkla bunlardan biri önceki çözümlerden daha kısa bir yoldan giderek çözümü iyileştirir. $\Delta t_{ij}^{k}(t+1)$, değişik karınca kolonisi algoritmalarında farklı yöntemlerle hesaplanmaktadır. En çok kullanılan lokal feromon güncelleme formülü aşağıdaki gibidir:

$$\Delta\tau_{ij}^{k}(t+1) = \begin{cases} 1/L^{k}(t+1) & \text{k karıncası (i, j)} \\ & \text{yolunu kullanmışsa,} \\ 0 & \text{diğer durumlarda} \end{cases} \quad (4)$$

$L^k(t+1)$ k karıncasının toplam tur uzunluğudur.

### 2.2.2 Global Feromon Güncellemesi

Global feromon güncellemesi, tüm karıncalar turlarını tamamladıktan sonra yapılır. Karıncaların her birinin toplam yol uzunlukları hesaplandıktan sonra en kısa yolu kullanan karınca bulunur. Bu karıncanın geçiş yapmış olduğu yollardaki feromon miktarları aşağıdaki formüle göre arttırılmaktadır:

$$\tau_{ij}(t+1) = (1-\rho)\,\tau_{ij}(t) + \Delta\tau_{ij}^{k}(t+1) \quad (5)$$

$$\Delta\tau_{ij}(t+1) = \begin{cases} \dfrac{1}{L_{best}(t+1)} & \text{(i, j) en iyi tura ait ise} \\ 0 & \text{diğer durumlarda} \end{cases} \quad (6)$$

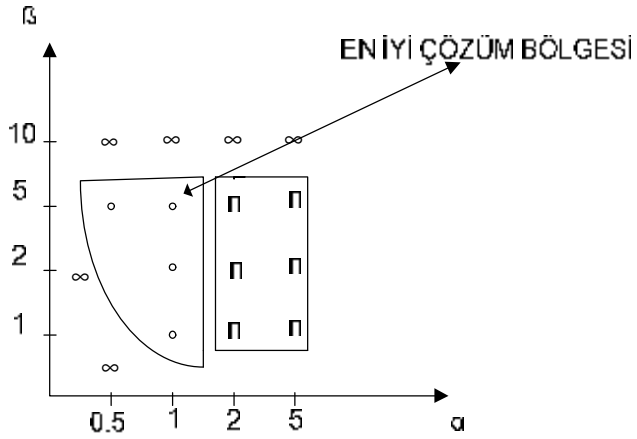
$L_{best}(t+1)$ geçerli iterasyonda bulunan en iyi turun uzunluğudur [Ying, 2004].

### 2.3 Optimum Karınca Sayısı

Karınca sayısının arttırılması çözümde iyileşmeye neden olur. Fakat hesaplamaları arttırdığı için karınca sayısının fazla arttırılması işlem zamanlarının uzamasına neden olur. GSP problemlerinde yapılan denemeler sonucunda karınca sayısının şehir sayısına eşit seçilmesinin uygun olacağı sonucuna varılmıştır. Karınca sayısı, problem büyüklüğüne ve uygulama alanına bağlı olarak değişir [Dorigo, 1991].

### 2.4 Parametre Değerleri

α değeri, ilgili yolun feromon miktarının önemini belirlemektedir. α değerinin yüksek olması feromonun yoğun olduğu yolların seçilme olasılığını arttırmaktadır. $\beta$ değeri ise yol uzunluklarının, bir sonraki noktanın seçimindeki etkisini belirlemektedir. $\beta$ değeri arttıkça tesadüfilik artmaktadır. ŞEKİL 3' te α ve $\beta$ parametrelerinin aldığı çeşitli değerler karşısında çözümün nasıl etkilendiği gösterilmiştir [Dorigo, 1996].




**ŞEKİL 3.Parametre seçimi**

## 3. KARINCA KOLONİSİ ALGORİTMALARI

Karınca kolonisi algoritması üzerinde, geliştirildiğinden bu yana birçok çalışma yapılmış ve bu çalışmalara sonucunda birçok farklı karınca kolonisi algoritması ortaya çıkmıştır. Bunlardan iki tanesi olan Rank temelli karınca kolonisi algoritması ve Max-min karınca sistemi bu bölümde kısaca anlatılmıştır.

### 3.1 Rank Temelli Karınca Kolonisi Algoritması

Rank temelli karınca koloni algoritmasında, sadece belli sayıdaki karıncaların geçiş yapmış olduğu yolların feromon izlerinin yenilenmesine izin verilir. Karıncalar tur uzunluklarına göre sıralanır ($L_1(t) \leq L_2(t) \leq \ldots L_m(t)$) ve belirlenen sayıdaki karıncanın geçiş yaptığı yollardaki feromon düzeyi tur uzunluklarına bağlı olarak yenilenir. Dolaysıyle global en iyi çözümün feromon güncelleme düzeyi en yüksektir. Rank temelli karınca koloni algoritmasında feromon güncelleme formülü aşağıdaki gibidir [Stützle, 1999]:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{r=1}^{w-1}(w-r)\Delta\tau_{ij}^{t}(t) + w\Delta\tau_{ij}^{best}(t) \qquad (7)$$

$$\Delta\tau_{ij}^{t}(t) = 1/L^{r}(t)\ ,\quad \Delta\tau_{ij}^{gb}(t) = 1/L^{best} \qquad (8)$$

### 3.2 Max-Min Karınca Sistemi (MMKS)

MMKS'nin temel özelliklerinden biri her iterasyonda sadece bir karıncanın feromon yenilemesine izin verilerek çözümün İyileştirilmesinin sağlanmasıdır. Bu karınca, bir önceki iterasyondaki en iyi çözümü veren karıncadır. İkinci özelliği, aramadaki dalgalanmayı önlemek için, feromon izlerinin sınırlı bir aralıkta olmasıdır [Stützle, 2000].

$[\tau_{min}, \tau_{max}]$ , $\tau_{min} \leq \tau \leq \tau_{max}$ (9)

Feromon izleri üst limitten başlatılır ve bu da başlangıçta yüksek oranda iyileşme sağlar. MMKS'de alt ve üst limitler uygun seçilmezse tüm karıncalar aynı yoldan gider ve bu da iyi bir çözümün bulunmasını engeller. Alt ve üst limitler aşağıdaki formüllerle hesaplanır (Stutzle,1999):

$$\tau_{max} = \frac{1}{\rho}\frac{1}{L^{best}}, \tau_{min} = \frac{\tau_{max}}{2n} \quad (10)$$

## 4. KARINCA KOLONİSİ ALGORİTMASININ GSP'YE UYGULANMASI

GSP, başlanan noktaya dönülmek üzere, n adet düğümün sadece bir kere ziyaret edilerek, toplam mesafe, maliyet veya sürenin minimizasyonunu sağlama problemi olarak tanımlanır [12]. Yolların uzunlukları $d_{ij}$ olmak üzere, simetrik GSP'de $d_{ij} = d_{ji}$, asimetrik GSP'de $d_{ij} \neq d_{ji}$ 'dir. Amaç uygun yollardan ilerleyerek tur uzunluğunu minimize etmektir.

GSP problemlerinin çözümüne yönelik birçok yöntem geliştirilmiştir. Dal-Sınır yöntemide, bütün alternatif rotalar kıyaslanarak içlerinden en iyi olanı seçilir. Ancak düğüm sayısı arttıkça, kıyaslanacak devre sayısı üstel olarak artmaktadır. Örneğin n=10 düğümlü yönlü olmayan bir şebekede (n -1) ! / 2 = 181.440 kadar farklı devre, aynı düğüm sayısına sahip yönlü bir şebekede ise (n -1) ! = 362.880 farklı devre mevcut olacaktır. Bu örnekten de anlaşılacağı gibi şebeke yönlü veya yönsüz olsun, düğüm sayısı arttıkça devrelerin tümünü kıyaslamak imkansız hale gelmektedir [Timor,2001]. Örnek uygulamamız düşünüldüğünde kıyaslanacak rota alternatifi sayısı 81 ! = 5,79712602074737E+120 olacaktır. Bunun dışında optimum çözümü garanti etmeyen yaklaşık çözüm yöntemleri mevcuttur. Bunlardan bazıları İlave Etme Algoritması (Insertion Algorithm), En Yakın Komşu Algoritması (Nearest Neighbor), İki Yol Değiştirme Algoritması (Two-way Exchange Improvement Heuristic)' dır. Ayrıca GSP problemlerinin çözümünde Tamsayılı Programlama ve Dinamik Programlama da kullanılmaktadır.

Karınca koloni algoritmasının GSP'ye uygulanmasında feromon izleri ve sezgisel bilgiler kullanılır. İz yoğunluğu $\tau_{ij}(t)$ algoritmanın adımları boyunca değiştirilen nümerik bilgilerdir.

Başlangıçta her m karınca rastsal olarak seçilen şehirlere yerleştirilir ve iteratif olarak her şehre geçiş kuralı uygulanır. i şehrindeki karınca henüz gidilmemiş j şehrini, iz yoğunluğu ( $\tau_{ij}(t)$ ) ve şehirler arasındaki uzunluğun fonksiyonuna bağlı bir olasılığa bağlı olarak seçer. Karınca büyük olasılıkla kendisine daha yakın olan şehri ve yüksek feromon izine sahip hattı seçer. Uygun bir çözümde her karınca tabu listesi olarak adlandırılan sınırlı bir hafızaya sahiptir. Hafıza henüz gidilmemiş şehirlere gidilmesini sağlar ve uygun çözümün sağlanmasını garanti eder. Tüm karıncalar bir turu tamamladıktan sonra feromonlar yenilenir. Başlangıçta çok düşük sabit bir feromon düzeyi alınır. [Stützle, 1999].

## 5. UYGULAMA

Karınca kolonisi algoritmasının ilk uygulaması gezgin satıcı problemi üzerine olmuştur. Bunun nedeni en kısa yol mantığının karınca kolonisi davranışlarıyla birebir uyumlu olması ve kolayca adapte edilebilir olmasıdır. Ayrıca kolayca anlaşılabilir olması ve çok karmaşık teknikler ve hesaplar gerektirmemesi de algoritmanın en çok uygulandığı problem olmasını sağlamıştır [Dorigo, 1999]. Geniş bir uygulama alanı olması ve bahsedilen avantajlarından dolayı bu çalışmada da karınca kolonisi algoritmasının GSP' ye uygulamasına yer verilmiştir.

Örnek problem, Türkiye'nin herhangi bir ilinden yola çıkılarak ve tüm şehirler ziyaret edilerek tekrar başlangıçtaki noktaya dönüş şeklinde düşünülmüştür. Gerçekleştirilecek böyle bir Türkiye turunun, en kısa şekilde nasıl yapılabileceği GSP problemi olarak ve karınca kolonisi algoritması ile belirlenmiştir.

Öncelikle 81 ilimize ait mesafeler tablosu oluşturulmuştur. Tüm illerimizin birbirlerine olan karayolu uzaklığı kilometre cinsinden tablolaştırılmıştır. Simetrik GSP problemi olduğundan bütün $d_{ij}$ ve $d_{ji}$ ler birbirlerine eşittir. 81 ilimize ait mesafeler tablosunun bir kısmı örnek olarak TABLO 1' de gösterilmiştir.

**TABLO 1. Mesafeler tablosundan örnek bir parça**

| | Adana | Adıyaman | Afyon | Ağrı | Amasya | Ankara |
|---|---|---|---|---|---|---|
| Adana | 0 | 330 | 573 | 966 | 612 | 490 |
| Adıyaman | 330 | 0 | 903 | 648 | 636 | 757 |
| Afyon | 573 | 903 | 0 | 1314 | 593 | 257 |
| Ağrı | 966 | 648 | 1314 | 0 | 734 | 1057 |
| Amasya | 612 | 636 | 593 | 734 | 0 | 336 |
| Ankara | 490 | 757 | 257 | 1057 | 336 | 0 |

Mesafeler tablosu oluşturulduktan sonra problemin çözümü için tasarlanan karınca kolonisi algoritması Microsoft Excel' de Visual Basic Macro diliyle yazılmıştır. Problemin çözümünde lokal feromon güncellemesiyle birlikte global feromon güncellemesi de kullanılmıştır. Bilinen algoritmalardan farklı olarak her bir iterasyondaki en uygun turu gerçekleştiren karınca bir sonraki iterasyona aktarılmış, böylelikle bulunan en iyi turun korunması sağlanmıştır. En iyi tur korunduğundan q değerinin düşmesinin avantaj sağlayacağı düşünülmüştür. Bununla aramanın hızlandırılması sağlanmıştır. Tüm iller plaka numaralarına göre 1'den 81' e kadar kodlandıktan sonra algoritma ile ilgili parametreler yapılan araştırmalar ve denemeler sonucunda TABLO 2' deki gibi belirlenmiştir.

**TABLO 2. Kullanılan parametreler**

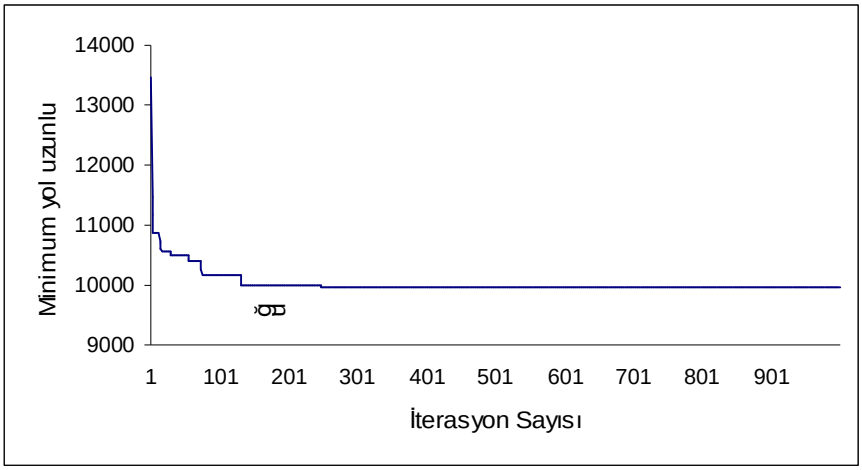
| Parametre | Değer |
|---|---|
| Global güncelleme paydası | 100 |
| Buharlaşma oranı | 0,1 |
| q değeri | 15 |
| Karınca sayısı | 81 |
| İterasyon sayısı | 1000 |
| Alfa | 0,80 |
| Beta | 4 |

Program belirlenen iterasyon sayısı kadar çalıştırılmış ve optimum sonuç olan **9954** değeri **241** inci iterasyonda bulunmuştur. Bulunan bu en iyi sonuca ait çözüm değerleri Tablo 3' te gösterilmiştir.

**TABLO 3. İllere göre en kısa Türkiye turu.**

| Çıkış noktası | | Varış noktası | Çıkış noktası | | Varış noktası | Çıkış noktası | | Varış noktası | Çıkış noktası | | Varış noktası |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Çorum | - | Yozgat | Edirne | - | Kırklareli | Rize | - | Artvin | Batman | - | Diyarbakır |
| Yozgat | - | Kırşehir | Kırklareli | - | Tekirdağ | Artvin | - | Ardahan | Diyarbakır | - | Mardin |
| Kırşehir | - | Kırıkkale | Tekirdağ | - | İstanbul | Ardahan | - | Kars | Mardin | - | Şanlıurfa |
| Kırıkkale | - | Çankırı | İstanbul | - | Kocaeli | Kars | - | Iğdır | Şanlıurfa | - | Adıyaman |
| Çankırı | - | Ankara | Kocaeli | - | Yalova | Iğdır | - | Ağrı | Adıyaman | - | K.Maraş |
| Ankara | - | Eskişehir | Yalova | - | Bursa | Ağrı | - | Erzurum | K.Maraş | - | Gaziantep |
| Eskişehir | - | Kütahya | Bursa | - | Bilecik | Erzurum | - | Bayburt | Gaziantep | - | Kilis |
| Kütahya | - | Afyon | Bilecik | - | Sakarya | Bayburt | - | Gümüşhane | Kilis | - | Hatay |
| Afyon | - | Uşak | Sakarya | - | Bolu | Gümüşhane | - | Erzincan | Hatay | - | Osmaniye |
| Uşak | - | Isparta | Bolu | - | Düzce | Erzincan | - | Tunceli | Osmaniye | - | Adana |
| Isparta | - | Burdur | Düzce | - | Zonguldak | Tunceli | - | Malatya | Adana | - | İçel |
| Burdur | - | Antalya | Zonguldak | - | Bartın | Malatya | - | Elazığ | İçel | - | Karaman |
| Antalya | - | Denizli | Bartın | - | Karabük | Elazığ | - | Bingöl | Karaman | - | Konya |
| Denizli | - | Muğla | Karabük | - | Kastamonu | Bingöl | - | Muş | Konya | - | Aksaray |
| Muğla | - | Aydın | Kastamonu | - | Sinop | Muş | - | Bitlis | Aksaray | - | Niğde |
| Aydın | - | İzmir | Sinop | - | Samsun | Bitlis | - | Van | Niğde | - | Nevşehir |
| İzmir | - | Manisa | Samsun | - | Ordu | Van | - | Hakkari | Nevşehir | - | Kayseri |
| Manisa | - | Balıkesir | Ordu | - | Giresun | Hakkari | - | Şırnak | Kayseri | - | Sivas |
| Balıkesir | - | Çanakkale | Giresun | - | Trabzon | Şırnak | - | Siirt | Sivas | - | Tokat |
| Çanakkale | - | Edirne | Trabzon | - | Rize | Siirt | - | Batman | Tokat | - | Amasya |
| | | | | | | | | | Amasya | | Çorum |

Her bir iterasyona ait en kısa turları gösteren grafik ŞEKİL 4' te gösterilmiştir. Grafikten de anlaşılacağı üzere karınca kolonisi optimizasyonu uygulanan problemde ilk iterasyonlarda çözüm değerinde çok yüksek oranda bir iyileşme söz konusudur. Daha sonra bu iyileşme hızı azalmaktadır.




**ŞEKİL 4. İterasyonlar boyunca minimum yol uzunluğu**

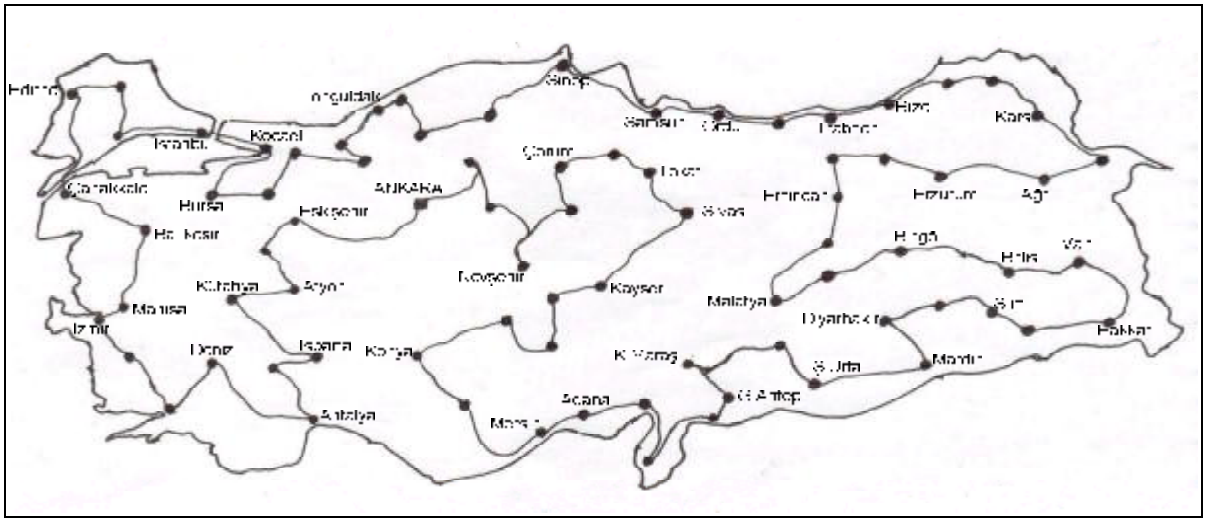
GSP problemlerinde karınca kolonisi algoritmasının, diğer sezgisel yöntemlere göre daha iyi ve daha hızlı sonuç verdiği bilinmektedir [Dorigo, 1997].

TABLO 4' te KKA ve diğer yöntemlerin sonuç ve süre karşılaştırmalarına yer verilmiştir.

**TABLO 4. KKA ve diğer yöntemlerin karşılaştırılması**

| Yöntem | Sonuç |
|---|---|
| KKA | 9954 |
| İlave Etme Algoritması | 12049 |
| En Yakın Komşu Algoritması | 13120 |
| İki Yol Değiştirme Algoritması | 11136 |

Çözüme ait güzergah ŞEKİL 5' de gösterilmiştir. En kısa yoldan Türkiye turu yapmak isteyen bir kişi şekilde belirtilen yolu takip etmek zorundadır. Başlangıç noktası olarak her bir il seçilebilir. Çünkü yol başlangıç noktasından başlayıp yine aynı yere döndüğünden tüm noktalar (iller) için de geri dönüş söz konusudur.




**ŞEKİL 5. Bulunan sonuca göre Türkiye turu**

## 6.SONUÇ

Dağıtım, günümüz ekonomik hayatında hem rekabet avantajı hem de maliyetler üzerindeki etkisi bakımından önemini arttırmaktadır. Maliyetler ve zaman açısından dağıtımın en uygun şekilde yapılabilmesi için geliştirilmiş birçok teknikten faydalanılmaktadır. Bunlardan en yenisi olan karınca kolonisi algoritması GSP problemlerine benzer durumlar için oldukça hızlı ve iyi sonuçlar vermektedir. İyi tasarlanmış ve uygulamaya yönelik kolaylıkları da bünyesinde barındıran, KKA ile geliştirilmiş bir dağıtım programı ile kurumların iyi ve hızlı sonuç alacağı muhakkaktır. Birçok alanda ve kurumda kullanılmaya başlayan KKA' nın çalışmadaki uygulamaya benzer turistik gezilerin planlanmasında, lojistik firmalarının ve ürünlerini belli noktalara ulaştırmak zorunda olan tüm işletmelerin dağıtım planlarında, yol yapım çalışmalarında vb. alanlarda başarıyla kullanılabileceği düşünülmektedir.

## KAYNAKÇA


Bonabeau E.; Dorigo M.; Theraulaz G.; "*Inspiration for optimization from social insect behavior*", **Nature** 2000;406, ss.39–42.

Dorigo M.; Maniezzo V.; Colorni A., **The Ant System: An Autocatalytic Optimizing Process**, Technical Report No. 91-016 Revised, Politecnico di Milano, Italy, 1991.

Dorigo M.; Maniezzo V.; Colorni A., "*The Ant System: Optimization by a Colony of Cooperating Agents*", **IEEE Transactions on Systems, Man, and Cybernetics-Part B**, 1996, 26(1), ss.29-41

Dorigo M.; Gambardella LM., "*Ant colonies for the traveling salesman problem*" **BioSystems**, 43,1997, ss.73–81.

Dorigo M.; Gambardella LM., "*Ant colony system: A cooperative learning approach to the traveling salesman problem*" **IEEE Transactions on Evolutionary Computation**, 1(1), 1997, ss53–66

Dorigo M.; Di Caro G.; Gambardella LM., "*Ant algorithms for discrete optimization*", **Artificial Life** 1999;5, ss.137–172.

Stützle T.; Dorigo M., "ACO Algorithms for the Traveling Salesman Problem" In K. Miettinen, M. Makela, P. Neittaanmaki, J. Periaux, editors, **Evolutionary Algorithms in Engineering and Computer Science**, Wiley, 1999.

Stützle,T.; Hoos,H.H., "*Max–min ant system*" **Future Generation Computer Systems**, 16, 2000, ss.889–914.

Ying K.; Liao C., "*An Ant Colony System For Permutation Flow-Shop Sequencing*", **Computers& Operations Research**, 31,: 791-801, 2004

Timor M., **Yöneylem Araştırması ve İşletmecilik Uygulamaları**, İ.Ü.İşletme Fakültesi Yayınları, 2001, ss. 148-149

Tsai C.F.; Tsai C.W.; Tseng C.C., "*A new hybrid heuristic approach for solving large traveling salesman problem*", **Information Sciences**, 2003, ss. 1-15

http://www.kgm.gov.tr/il1.asp (10.10.2004)