# Karınca Koloni Optimizasyonu (ACO)

**Doğadan Esinlenen Optimizasyon Algoritması (schwefel fonksiyonu)  
Hazırlayanlar: [Mehmet Ali ÜNER, Helin AKTAŞ]**

## 

# Giriş

Doğadan esinlenen optimizasyon algoritmaları, karmaşık problemlerin çözümünde ilham kaynağıdır.  
Karıncaların yiyecek arama davranışında kullandığı feromon izleri, kısa yolu bulma prensibine dayanır.  
Karıncalar arasındaki işbirliği ve iletişim mekanizması, optimizasyon problemlerine uyarlanmıştır.

Karınca Kolonisi Algoritması Tarihi

1959 yılında Fransız böcekbilimci Pierre-Paul Grasse karıncaların 'önemli uyarılar' olarak adlandırdığı duruma tepki verdiğini, karıncaların koordinasyon mekanizması ve kolonilerini gözlemlemiştir. 1989 yılında ise Gerardo Beni ve Jing Wang sürü zekası kavramını ortaya atmıştır. Bu bilgiler ışığında sürü zekası algoritmalarından biri olan, Karınca Koloni Algoritması ilk olarak 1992 yılında Marco Dorig tarafından doktora tezinde, en kısa yolun bulunmasında temel prensipleri ile ele alınmıştır. Bu algoritmaya Karınca Sistemi (Ant System - AS) adını vermiş, algoritmadaki parametrelerin uygunluk değerlerini saptamış, değişik boyutlardaki birçok gezgin satıcı problemlerinde deneyerek başarılı sonuçlar elde etmiştir. Bunun üzerine karınca kolonisi algoritmaları diğer araştırmacılar tarafından da kullanılmaya başlanmış ve günümüzde en iyileme problemlerinin çözümünde yaygın olarak kullanılan bir yapay zekâ tekniği haline gelmiştir.

• Karınca kolonisi algoritması (KKA) amacı, simetrik ve asimetrik gezgin satıcı ve benzer yapıdaki problemlere uygun ve iyi çözümler bulmaktır. Koloniler halinde yaşayan karıncalar, yuvalarıyla yiyecek arasında en kısa yolu bulma kabiliyetine sahiptir. Geçtikleri yollara bıraktıkları feromon adı verilen kimyasal sıvı sayesinde yollarını bulan karıncaların, gerçek hayattaki bu davranışlarından yola çıkılarak geliştirilmiştir.

• Oluşturulan yapay karıncalar kullanılarak ve karıncaların belli kurallarla geçiş yaptığı yollarda yapay feromon güncellemesi yapılarak en kısa yol iterasyonlar boyunca araştırılmaktadır.

Bu algoritmada kolonilerin benzetimi ile değil yapay karınca kolonilerinin bir optimizasyon aracı olarak kullanılmasıyla ilgilenilmektedir.

KKA:

➤ Zeki davranış biçimi sergiler.

➤ Kombinasyonel optimizasyon gösterir.

➤ Sezgisel ve olasılık tabanlıdır.

➤ Adaptif yapıya sahiptir.

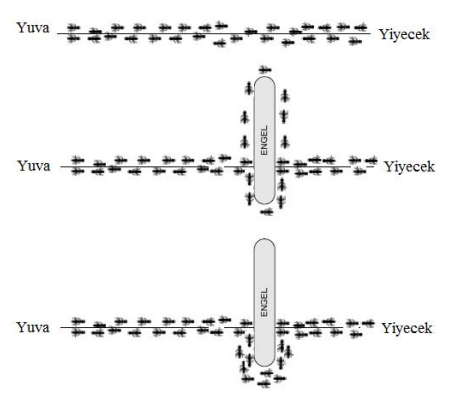
➤ Pozitif ve negatif geri besleme oluşturur.

Doğa, problemlerin çözümünde etkili stratejiler sunar. Karıncaların yiyecek arama davranışlarında şu özellikler dikkat çeker:

- \*\*Feromon izleri\*\*: Yolları işaretleyerek kısa yolları tercih ederler.

-\*\*Adaptasyon\*\*: Değişen çevre koşullarına uyum sağlarlar.

## Karınca Hareketleri ve Yiyecek Arama Mekanizması



• Karıncalar yiyecek kaynağı ve yuva arasına feromon izi bırakır.

• Karıncaların izleyeceği yol üzerine yerleştirilen bir engel izi böler ve bir kısım karınca yukarıdaki yolu bir kısmı ise aşağıdakini seçer. Seçim tümüyle rassaldır.

• Karıncalar neredeyse sabit bir hızda yürüdükleri için aşağıdaki ve kısa olan yolu seçen karıncalar turlarını daha çabuk bitirirler. Kısa yolda feromon birikmesi daha fazla olur ve karıncaların çoğu feromon izlerini takip ederek kısa yolu kullanır.

## Karınca kolonisi Algoritmasının Adımları

**1. Adım:** Karıncaların yönlerini bulabilmesi için kullandıkları feromon sıvılarına ait başlangıç feromon sıvı değeri belirlenir.

**2. Adım:** Karıncalar her farklı noktaya rastgele yerleştirilir.

**3. Adım:** Karıncalar, sonraki hedeflerine olasılık denklemlerine bağlı olacak şekilde turlarını tamamlar.

**4. Adım:** Karıncaların aldıkları yollar ve buna ait feromon sıvı miktarları hesaplanır, sonrasında yeni lokal feromon sıvı miktarları oluşturularak ilgili bilgi güncellenir.

**5. Adım:** En iyi çözüme ait yol hesaplanır ve global feromon güncellenmesinde kullanılır.

**6. Adım:** Maksimum iterasyon sayısı yada yeterlilik kriteri sağlanana kadar Adım 2'ye gidilir.

## Karınca Kolonisi Optimizasyonu Avantajları ve Dezavantajları

**Avantajları**

• Eşzamanlı olarak çalışabilir.

• İyi çözümleri hızlı bir şekilde keşfedebilir.

• Uyum sağlar ve dinamiktir.

• Hedefe yakın çözümler verme garantisi sunar.

• Gezgin satıcı problemi ve benzeri problemlerde etkili çözümler sunar.

**Dezavantajları**

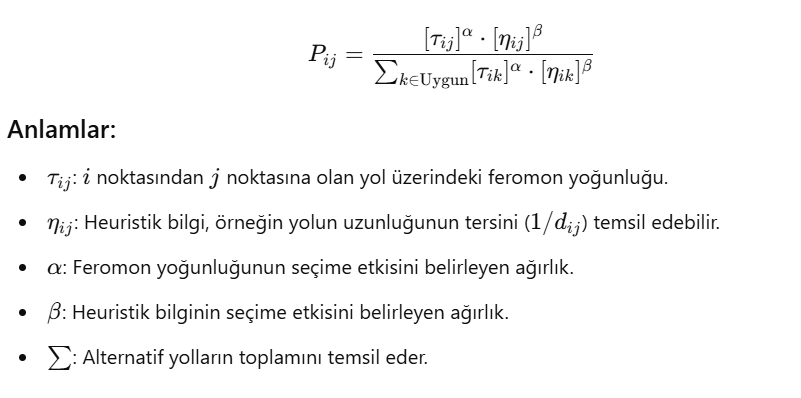
• Olasılık dağılımı her bir yineleme için dağılır.

• Teorik olarak analiz zordur.

• Hedefe ulaşmak için çok fazla yineleme gerektirebilir.

• Rastgele karar dizilerinden oluşur.

## KKA’NIN Bileşenleri



## Feromon Güncelleme Formülü

## 

## KKA Uygulama Alanları

- Gezgin Satıcı Problemi (TSP)  
- Rota Planlama  
- Zaman Çizelgeleme Problemleri  
- Ağ Optimizasyonu  
- Robotik ve Otonom Araçlar  
- Veri Madenciliği ve Makine Öğrenmesi

## Karınca Kolonisi Algoritması Parametreleri

➤ **Karınca Sayısı:** Kolonide kaç tane karınca olacağını belirler. Karınca sayısının arttırılması çözümde iyileşme sağlar fakat sayı arttıkça işlem zamanlaması uzar.

➤ **İterasyon Sayısı:** Arama işleminin kaç iterasyon (adım) gerçekleşeceğini belirler. Koloni sayısı olarak da adlandırılır. Koloni sayısının çok büyük olması programın çalışma zamanını uzatır.

➤ **Feromon Kuvvetlendirme Oranı (α):** Düğümler arasındaki feromon miktarının önem derecesini belirler. a değerinin yüksek olması feromonun yoğun olduğu yolların seçilme olasılığını arttırırken tesadüfiliği azaltmaktadır.

➤ **Sezgisellik Kuvvetlendirme Oranı (β):** Düğümler arasındaki mesafenin önem derecesini belirler. ẞ değeri arttıkça bir sonraki yolun seçiminde tesadüfilik artmaktadır. ẞ'nın düşük olması ise alternatif çözümlerin araştırılması ihtimalini azaltır.

➤ **Feromon Buharlaşma Oranı (ρ):** Her iterasyon sonunda düğümler arasındaki feromonların hangi oranda buharlaşacağını belirler. Genellikle (0,1) aralığında ve sıfıra yakın bir değer olarak belirlenir. Bu oranın çok yüksek olması en iyi Çözümün bulunmasında fazla zaman gerektirirken çok düşük olması algoritmanın yerel optimumlara takılmasına sebep olur.

➤ **Po değeri:** En iyi çözümün sonraki iterasyonlara aktarılması olasılığını belirler. Gezgin satıcı problemlerinde genellikle %90 olarak alınmıştır. Mevcut karıncaların %90 oranla en iyi çözümü sağlayan yolu takip etmesi, %10 olasılıkla feromon izlerine bağlı olarak yolunu belirlemesidir.

## Örnek Problem: Gezgin Satıcı Problemi

\*\*Tanım\*\*: Bir satıcının belirli sayıda şehri ziyaret ederek en kısa rotayı bulması.  
  
KKA ile çözüm adımları:  
1. Karıncalar rastgele rotalar oluşturur.  
2. Feromonlar, kısa yollar üzerinde yoğunlaşır.  
3. İterasyonlar sonucunda en kısa rota belirlenir.

## Karınca Kolonisi Optimizasyonu Akış Şeması

KKO parametrelerini belirle. Adım 1

Olasılık dağılımını kullanarak çözüm oluştur. (Feromon izi ve Randomizasyon)

Adım 2

Yerel feromonları güncelle.

Adım 3

Hayır

Tüm karıncalar hedefe ulaştı mı?

Adım 4

Evet

En uygun yolun uzunluğunu hesapla ve yalnızca en uygun yolun feromon seviyesini güncelle.

Adım 5

Hayır

Adım 6

Sona erme koşulu (mak. İterasyon) sağlanıyor mu?

Evet

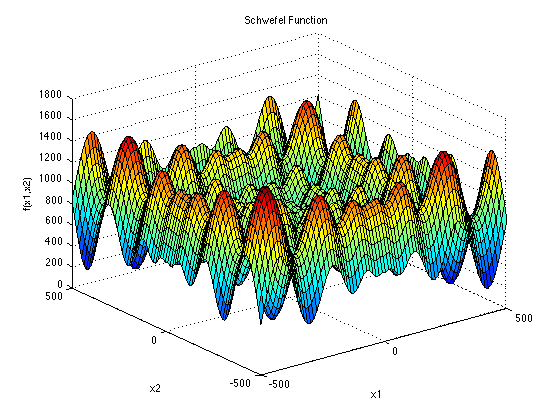
Çıktı en yüksek feromon seviyesini içeren çözümü gösterir.

# Karınca Kolonisi Algoritması'nda bir tur esnasında i noktasında bulunan k karıncası için, sonraki j noktasını seçerken iki alternatif yol söz konusudur.

# 1. Alternatif: Gidebileceği yollar içerisinden feromon miktarlarına bağlı olarak hesaplanan seçim değerlerinden maksimum olanını seçmesidir. Genellikle bu yolu tercih etme olasılığı %90 olarak belirlenmektedir.

# 2. Alternatif: Yollardaki feromon miktarı göz önüne alınarak oluşturulan olasılık dağılımına bağlı olarak yolların seçilmesidir.

## Karınca Kolonisi Algoritması ile Schwefel Fonksiyonu Optimizasyonu



schwef2.png (292×63)

 **f(x):**

* Bu, fonksiyonun değerini ifade eder. Amaç, bu fonksiyonun minimum değerini bulmaktır.

 **d**:

* Problemdeki boyut sayısını ifade eder. Örneğin, 2 boyutlu bir problemde d=2 olur.

 **xi**:

* Her bir değişkenin değeridir (i=1,2,…,d).
* Değişkenlerin genellikle [−500,500]] aralığında olduğu kabul edilir.

 **418.9829d**:

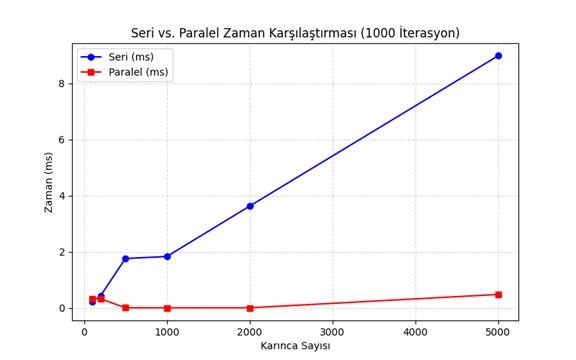
* Bu, fonksiyonun sabit bir kısmıdır. Problem boyutu arttıkça bu değer artar.

 **−∑i=1d​xi​sin(|xi|​​):**

* Bu, her bir değişken için hesaplanan bir terimdir ve toplamı alınır.

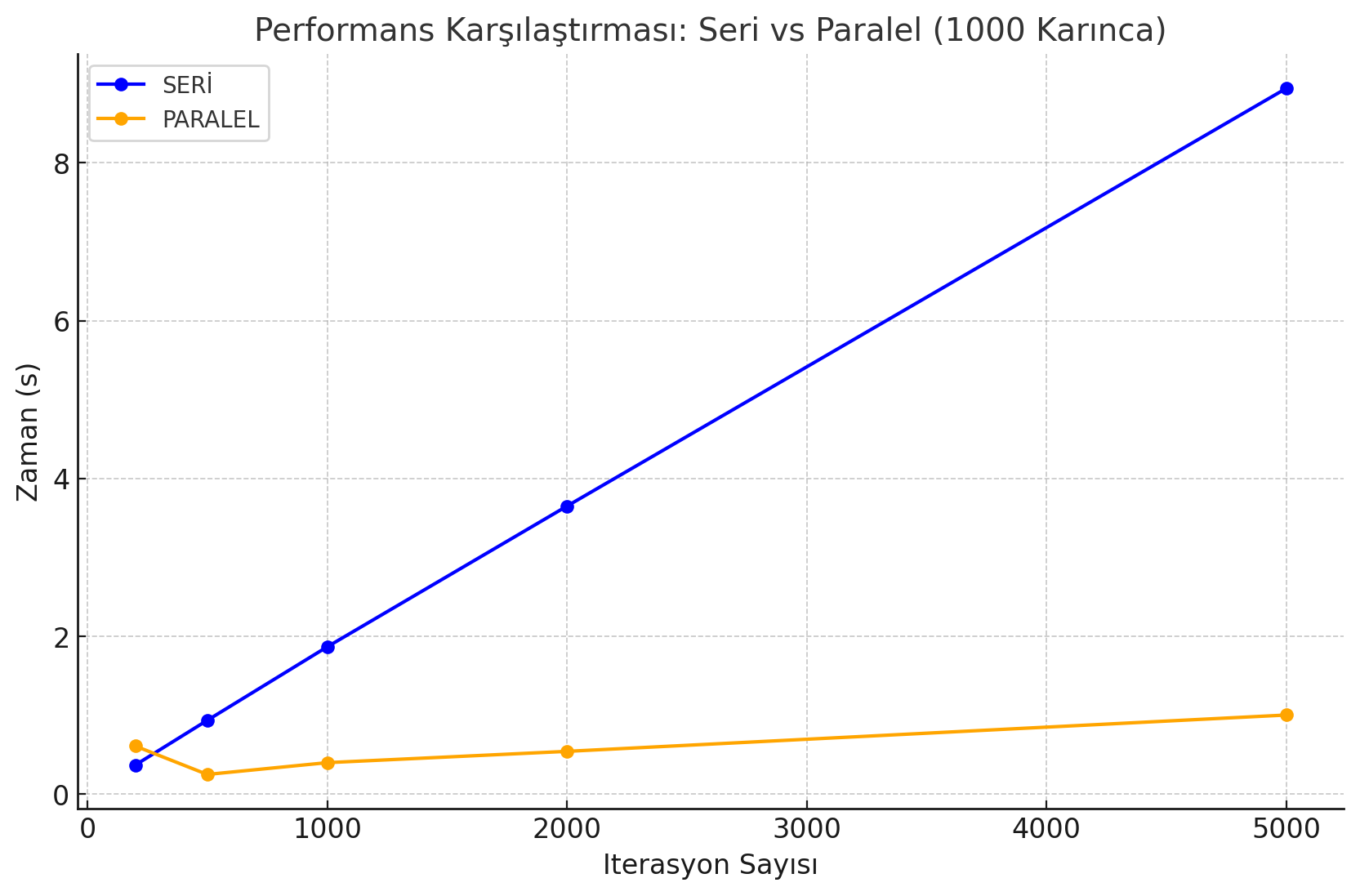
## 1000 Iterasyon için program çıktı tablosu:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **karınca Sayısı** | **100** | **200** | **500** | **1000** | **2000** | **5000** |
| **Seri (s)** | 0.220086 | 0.429335 | 1.75734 | 1.82749 | 3.62301 | 8.97577 |
| **Paralel(s)** | 0.324554 | 0.321255 | 0.00297151 | 0.00141577 | 0.00141577 | 0.477164 |



## 1000 Karınca için program çıktı tablosu:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Iterasyon Sayısı** | **200** | **500** | **1000** | **2000** | **5000** |
| **Seri (s)** | 0.368863 | 0.935179 | 1.86628 | 3.64995 | 8.94504 |
| **Paralel(s)** | 0.609287 | 0.249614 | 0.398938 | 0.542105 | 1.00265 |

****

## Sonuç ve Değerlendirme

Karınca Koloni Optimizasyonu (ACO), doğadan esinlenen etkili bir optimizasyon yöntemidir. Paralel yapı ve adaptasyon özellikleri sayesinde karmaşık problemler üzerinde başarılı sonuçlar vermektedir. Çok çeşitli alanlarda uygulanabilir bir çözüm yöntemidir.