Harris Hawks Optimizasyon Algoritması

151320201045 - Berat Çavdar
151320201097 - Fatih Yıldırım
151320211139 - İzzet Şişman

HHO NEDİR?

Harris Hawks Optimizasyonu (HHO), doğada Harris şahinlerinin işbirliği ve av takip tarzından esinlenerek geliştirilen bir sürü tabanlı, gradyan olmayan bir optimizasyon algoritmasıdır. HHO'nun ana özellikleri şunlardır: Esnek bir yapıya sahip olması, "şaşırtıcı atılım" ilkesini kullanması, dinamik parametreleri ayarlayabilmesi, farklı keşif mekanizmalarını içermesi, kısa atlama tabanlı sömürüyü desteklemesi, progressive seçim şemasıyla çalışması, rastgele atlama gücünü dengelemesi ve adaptif bileşenlerle zorlukları ele alması. HHO, birçok optimizasyon probleminde başarılı sonuçlar verir.

Harris Şahinleri ve Avlanma Stratejileri

- Harris şahinleri, avlanırken işbirliği yaparlar. Bir grup şahin, avın üzerine dalarken diğerleri yukarıda bekler.
- Bu strateji, avın kaçmasını engellemek ve daha etkili bir şekilde avlanmak için kullanılır.

HHO'nun Çalışma Mantığı

- Popülasyon, tasarım parametrelerini temsil eden bireylerden oluşur.
- Her birey, bir çözümün bir noktasını temsil eder.
- HŞO, bireylerin konumlarını güncelleyerek en iyi çözümü arar.

HHO'NUN ADIMLARI

- Başlangıçta rastgele konumlarla bir populasyon oluşturulur.
- Her iterasyonda:
- En iyi çözüm ve uygunluk değeri güncellenir.
- Harris şahinlerinin hareketi taklit edilir:
- Eşik değerleri kullanılarak konumlar güncellenir.
- Sınırlar kontrol edilir.
- En iyi çözüm ve uygunluk değeri güncellenir.

Basit örnek

(10 sayı arasından en büyüğü bulmak)

1 -) Başlangıçta, 10 adet rastgele sayı seçin

$x1, x2, \ldots, x10$

2 -) Her şahini bir diğerine doğru hareket ettirin

$xi = xi + \text{rastgele adım}$

3 -) Bütün sayıları deneyene ya da belirlenen iterasyon sayısına kadar tekrarla ve her adımda en büyük sayıyı güncelle.

Kodlar ve Açıklaması

```python
import numpy as np

def fitness_function(parameters):
    # Redüktörün ağırlığını hesapla (tasarım parametrelerine bağlı)
    # Örneğin: ağırlık = w1 * p1 + w2 * p2 + ...
    # Burada tasarım parametrelerini kullanarak ağırlığı hesaplayın
    # Daha karmaşık bir modeliniz varsa bu fonksiyonu uygun şekilde güncelleyin
    # Örneğin, gerilim sınırları, yüzey gerilimleri, şaft eğilmesi vb. dikkate alınabilir
    # Ağırlığı minimize etmek için uygun bir formül kullanın
    # Döndürülen değer daha düşük olmalıdır (minimize edilmelidir)

    # Örnek olarak:
    weight = np.dot(parameters, weights)

    return weight
```

Kodlar ve Açıklaması

```python
def hho_optimization(dim, n, max_iter):
    for t in range(max_iter):
        for i in range(n):
            fitness = fitness_function(X[i])
            if fitness < best_fitness:
                best_fitness = fitness
                best_solution = X[i]
            # Harris Hawks'ın hareketi
            E0 = 2 * np.random.rand() - 1
            E1 = 2 * np.random.rand() - 1
            E2 = 2 * np.random.rand() - 1
            # Konum güncellemesi
            X[i] = X[i] + E0 * (X[i] - best_solution) + E1 * (X[i] - X[np.ran
            # Sınırları kontrol et
            X[i] = np.clip(X[i], 0, 1)  # Örneğin, 0 ile 1 arasında sınırla
    return best_solution, best_fitness
```

Başlangıçta rastgele konumlarla bir kuş sürüsü oluşturulur.
Her iterasyonda, en iyi uygunluğa sahip kuşun konumu güncellenir.
Diğer kuşlar, en iyi kuşun konumuna doğru hareket eder ve bu şekilde arama alanını keşfederler.
Sınırları kontrol ederek konumları güncelleyin (örneğin, 0 ile 1 arasında sınırlayın).

Kodlar ve Açıklaması

```
# Ağırlıkları normalize et
dim = 15  # Tasarım parametre sayısı
weights = np.random.rand(dim)
weights /= np.sum(weights)

# Örnek kullanım
# dim = 15
n = 100   # Kuş sayısı
max_iter = 100

best_solution, best_fitness = hho_optimization(dim, n, max_iter)
print("En iyi çözüm:", best_solution)
print("En iyi uygunluk değeri:", best_fitness)
print("Ağırlıklar:", weights)
```

Optimizasyon süreci belirli bir iterasyon sayısı boyunca devam eder.
Sonuç olarak, en iyi çözüm ve uygunluk değeri elde edilir.

Örnek Sonuç

```
En iyi çözüm: [0.         0.         0.         0.         0.         1.
 0.         1.         0.20691534 1.         1.         1.
 0.         0.         1.        ]
En iyi uygunluk değeri: 0.0036364763515016967
Ağırlıklar: [0.04811151 0.08180035 0.06654755 0.02292192 0.02680605 0.10226437
 0.15981674 0.11362241 0.04795262 0.05725185 0.03991591 0.08427626
 0.0696762  0.01269292 0.06634335]
```

En İyi Çözüm: Bu dizi, tasarım parametrelerini içerir. Her bir değer, redüktör tasarımının belirli bir özelliğini temsil eder.
Örneğin, diş sayısı, şaft uzunluğu, yüzey kalitesi, dişli çapı ve güç aktarım kapasitesi gibi parametreler bu çözümde yer alır.
En İyi Uygunluk Değeri: Bu değer, HHO algoritması tarafından hesaplanan en iyi çözümün uygunluk değeridir.
Daha düşük bir uygunluk değeri, tasarımın daha iyi olduğunu gösterir. Bu durumda, tasarımın ağırlığı minimize edilmiştir.
Ağırlıklar: Bu dizi, tasarım parametrelerini ağırlıklandırmak için kullanılır.
Rastgele oluşturulmuş ve toplamı 1'e normalize edilmiştir.

Teşekkürler!