ENDÜSTRİ MÜHENDİSLİĞİNDE SEÇİLMİŞ KONULAR I

SEZGİSEL OPTİMİZASYON KONU ANLATIMI

1307210092 KAAN KAYSERILI
1307191431 EMRE GÖZÜOTLU

İçindekiler

Giriş 2
1. Sezgisel Optimizasyonun Temelleri 3
1.1 Optimizasyon Nedir? 3
1.2 Sezgisel Optimizasyonun Temel Kavramları 3
1.3 Sezgisel Optimizasyon Yöntemlerinin Sınıflandırılması 3
2. Sezgisel Optimizasyon Yöntemleri 4
2.1. Yapay Arı Kolonisi Algoritması (ABC) 4
2.2. Karınca Kolonisi Optimizasyonu (ACO) 6
2.3. Kuş Sürüsü Optimizasyonu (PSO) 7
2.4. Yarasaların Ekolasyonu Algoritması (BAT) 9
2.5. Gravitasyonel Arama Algoritması (GSA) 10
2.6. Ateşböceği Algoritması (FA) 12
2.7. Harmony Search Algorithm (HSA) 14
2.8. Cuckoo Search Algorithm (CSA) 15
2.9. Sinep Algoritması (FA) 17
2.10. Memetic Algoritmalar (MA) 18
2.11. Tabu Arama Algoritması (TA) 20
2.12. Genetik Algoritma (GA) 22
2.13. Yapay Sinir Ağları (ANN) 23
2.14. Derin Öğrenme Tabanlı Sezgisel Algoritmalar 25
2.15. Çok Hedefli Sezgisel Algoritmalar 26
2.16. Hibrit Sezgisel Algoritmalar 27
2.17. Arama ve Kurtarma Optimizasyon Algoritması (AKOA) 29
3. Genetik Algoritma ile n Vezir Problemi Uygulaması 30
KAYNAKÇA 35

Giriş

Optimizasyon, günümüz endüstriyel ve bilimsel çalışmalarında temel bir bileşen olarak kabul edilir. Bir sistem veya sürecin en iyi durumunu belirlemek ve bu duruma ulaşmak için kullanılan bir dizi teknik ve metodolojiyi içerir. Ancak, gerçek dünyadaki karmaşık problemler genellikle analitik yöntemlerle çözülemez ve bu noktada sezgisel optimizasyon devreye girer.

Sezgisel optimizasyon, bilgisayar algoritmalarının doğal süreçlerden esinlenerek karmaşık problemleri çözmek için kullanıldığı bir alanı ifade eder. Bu alanda, biyolojiden, fizikten, metaforlardan ve diğer alanlardan esinlenen yöntemler kullanılarak, problemlerin çözümüne yönelik çeşitli yaklaşımlar geliştirilir. Bu yaklaşımlar, problem alanına ve problemin karmaşıklığına bağlı olarak farklılık gösterir.

Bu rapor, sezgisel optimizasyonun temel kavramlarını ve yöntemlerini ele alarak, bu alandaki önemi ve uygulama alanlarını açıklamayı amaçlamaktadır. Ayrıca, endüstri mühendisliği bağlamında sezgisel optimizasyonun rolüne odaklanarak, bu alandaki eğilimleri ve gelecekteki potansiyeli değerlendirmeyi hedeflemektedir.

Sezgisel optimizasyonun, karmaşık problemleri çözmek için güçlü bir araç olarak kullanılması, endüstri mühendislerinin karşılaştığı zorlukları aşmalarına ve verimliliği artırmalarına yardımcı olabilir. Bu rapor, sezgisel optimizasyonun teorik temellerini anlamak ve pratik uygulamalarını değerlendirmek için bir kaynak olarak hizmet etmeyi amaçlamaktadır.

1. Sezgisel Optimizasyonun Temelleri

1.1 Optimizasyon Nedir?

Optimizasyon, bir sistemin veya sürecin belirli bir hedef veya kriter altında en iyi duruma getirilmesi sürecidir. Bu hedef genellikle maksimum kar elde etme, minimum maliyet veya en iyi performans gibi belirli bir kriter olabilir. Sezgisel optimizasyon, bu tür optimizasyon problemlerini çözmek için doğal süreçlerden esinlenen algoritmaları kullanır.

1.2 Sezgisel Optimizasyonun Temel Kavramları

- Popülasyon: Bir optimizasyon algoritmasında, olası çözümleri temsil eden bireylerin kümesidir. Her birey, potansiyel bir çözümü ifade eder.

- Fonksiyonel Değerlendirme: Bir çözümün belirli bir hedef fonksiyonu altında ne kadar iyi olduğunun değerlendirilmesidir. Bu fonksiyon genellikle optimize edilmek istenen kriteri temsil eder.

- Yeni Çözümler Üretme: Sezgisel optimizasyon algoritmaları, mevcut çözümleri kullanarak yeni çözümler üretirler. Bu yeni çözümler, mevcut çözümlerden türetilerek veya rastgele oluşturularak elde edilebilir.

- Seçilim ve Evrim: Popülasyondaki çözümler, belirli bir kriter kullanılarak seçilir ve belirli bir yöntemle değiştirilir. Bu süreç, daha iyi çözümlerin elde edilmesini ve popülasyonun zamanla iyileşmesini sağlar.

1.3 Sezgisel Optimizasyon Yöntemlerinin Sınıflandırılması

Sezgisel optimizasyon yöntemleri genellikle kullanılan temel yaklaşımlara göre sınıflandırılabilir. Bu sınıflandırma şunları içerir:

- **Doğal Süreçlerden İlham Alan Yöntemler:** Genetik algoritma gibi algoritmalar, doğal seleksiyon ve genetik varyasyon kavramlarından esinlenir.
- **Fiziksel Süreçlerden İlham Alan Yöntemler:** Simüle edilmiş tavlama gibi algoritmalar, fizikteki termodinamik süreçlerden esinlenir.
- **Toplumsal Davranıştan İlham Alan Yöntemler:** Parçacık sürü optimizasyonu gibi algoritmalar, toplum içindeki bireylerin davranışlarından esinlenir.

2. Sezgisel Optimizasyon Yöntemleri

Sezgisel optimizasyonun temel yöntemleri, çeşitli doğal süreçlerden esinlenen algoritmaları içerir. Bu yöntemler, genellikle belirli bir problem alanına veya problemin karmaşıklığına göre seçilir ve uygulanır.

2.1. Yapay Arı Kolonisi Algoritması (ABC)

Yapay Arı Kolonisi Algoritması (ABC), bal arılarının besin arama davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2005 yılında Derviş Karaboğa tarafından geliştirilmiştir.

ABC Algoritmasının Temel Prensipleri:

- **Sürü Zekâ:** ABC algoritması, bal arılarının koloniler halinde çalışarak karmaşık problemleri çözme yeteneğini taklit eder.
- **Üç Arı Türü:** Algoritmada üç tür arı bulunur:
    - **Görevli Arılar:** Mevcut besin kaynaklarını (çözümleri) geliştirmeye çalışırlar.
    - **Gözcü Arılar:** Yeni besin kaynakları (çözümler) aramakla sorumludurlar.
    - **Kaşif Arılar:** Rastgele yeni besin kaynakları (çözümler) ararlar.
- **Besin Kaynağı Kalitesi:** Besin kaynaklarının (çözümlerin) kalitesi, nektar miktarı ile temsil edilir. Daha fazla nektar, daha iyi bir çözüm anlamına gelir.
- **Bilgi Paylaşımı:** Arılar, besin kaynaklarının (çözümlerin) konumları ve kaliteleri hakkındaki bilgileri kovan içerisindeki diğer arılarla paylaşır.

ABC Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Görevli Arı Aşaması:** Her görevli arı, rastgele seçtiği bir besin kaynağına (çözüme) gider ve onu geliştirmeye çalışır.

3. **Gözcü Arı Aşaması:** Gözcü arılar, kovan içerisindeki diğer arılardan besin kaynaklarının (çözümlerin) konumları ve kaliteleri hakkındaki bilgileri toplar.
4. **Kaşif Arı Aşaması:** Kaşif arılar, rastgele yeni besin kaynakları (çözümler) arar.
5. **Hafıza Güncelleme:** Yeni ve daha iyi besin kaynakları (çözümler) bulunursa, hafızadaki eski besin kaynakları (çözümler) ile değiştirilir.
6. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

ABC Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

ABC Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

ABC Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.2. Karınca Kolonisi Optimizasyonu (ACO)

Karınca Kolonisi Optimizasyonu (ACO), karıncaların yiyecek arama davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 1997 yılında Marco Dorigo tarafından geliştirilmiştir.

ACO Algoritmasının Temel Prensipleri:

- **Sürü Zekâ:** ACO algoritması, karıncaların koloniler halinde çalışarak karmaşık problemleri çözme yeteneğini taklit eder.
- **Feromon İzleri:** Karıncalar, yiyecek kaynaklarına (çözümlere) gidiş-dönüş yollarında feromon izleri bırakırlar.
- **Feromon Yoğunluğu:** Feromon izinin yoğunluğu, yiyecek kaynağının (çözümün) kalitesini gösterir.
- **Olasılıklı Seçim:** Karıncalar, bir sonraki hamlelerini yaparken feromon izlerinin yoğunluğuna göre olasılıklı seçim yaparlar.
- **Buharlaşma:** Feromon izleri zamanla buharlaşır ve bu da yeni yolların keşfedilmesine yardımcı olur.

ACO Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Feromon İzleri Oluşturma:** Her çözüm için bir feromon izi oluşturulur.
3. **Çözüm Oluşturma:** Her karınca, feromon izlerini takip ederek bir çözüm oluşturur.
4. **Yerel Arama:** Karıncalar, oluşturdukları çözümleri geliştirmeye çalışır.
5. **Feromon Güncelleme:** Karıncalar, oluşturdukları ve geliştirdikleri çözümlere göre feromon izlerini günceller.
6. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

ACO Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

ACO Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

ACO Algoritmasının Uygulama Alanları:

- Seyahat satıcı problemleri
- Araç rotalama problemleri
- Zaman çizelgeleme problemleri
- Üretim planlama
- Grafik boyama problemleri
- Telekominikasyon ağ optimizasyonu

2.3. Kuş Sürüsü Optimizasyonu (PSO)

Kuş Sürüsü Optimizasyonu (PSO), kuş sürülerinin yiyecek arama davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 1995 yılında James Kennedy ve Russell Eberhart tarafından geliştirilmiştir.

PSO Algoritmasının Temel Prensipleri:

- **Sürü Zeka:** PSO algoritması, kuş sürülerinin koloniler halinde çalışarak karmaşık problemleri çözme yeteneğini taklit eder.
- **Konum ve Hız:** Her kuşun bir konumu (çözümü) ve bir hızı (çözümdeki değişimi gösteren vektör) vardır.

- **En İyi Küresel Konum:** Sürüdeki tüm kuşlar tarafından bilinen en iyi çözüm.
- **En İyi Kişisel Konum:** Her kuşun geçmişte bulduğu en iyi çözüm.
- **Hız Güncelleme:** Kuşlar, en iyi küresel konuma ve en iyi kişisel konumlarına göre hızlarını günceller.
- **Konum Güncelleme:** Kuşlar, hızlarını kullanarak konumlarını günceller.

PSO Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Hız ve Konum Güncelleme:** Her kuşun hızı ve konumu güncellenir.
3. **En İyi Küresel Konum Belirleme:** Sürüdeki tüm kuşlar tarafından bilinen en iyi çözüm belirlenir.
4. **En İyi Kişisel Konum Belirleme:** Her kuşun geçmişte bulduğu en iyi çözüm belirlenir.
5. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

PSO Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

PSO Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

PSO Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi

- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.4. Yarasaların Ekolasyonu Algoritması (BAT)

Yarasaların Ekolasyonu Algoritması (BAT), yarasaların av arama ve engellerden kaçınma davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2010 yılında Xin-She Yang tarafından geliştirilmiştir.

BAT Algoritmasının Temel Prensipleri:

- **Ekolasyon:** Yarasalar, avlarını ve engelleri bulmak için yüksek frekanslı ses dalgaları (ultrason) gönderir ve geri dönen yankıları analiz ederler.
- **Sesi Yükseltme ve Düşürme:** Yarasalar, avlarına yaklaştıkça ses frekanslarını yükseltir ve avdan uzaklaştıkça düşürürler.
- **Atalet:** Yarasalar, bir sonraki hamlelerini yaparken önceki konumlarını ve hızlarını da dikkate alırlar.
- **En İyi Çözüm:** Sürüdeki tüm yarasalar tarafından bilinen en iyi çözüm.

BAT Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Sesi Yükseltme ve Düşürme:** Her yarasaya bir frekans ve bir atalet değeri atanır.
3. **Çözüm Oluşturma:** Her yarasa, frekansını ve ataletini kullanarak bir çözüm oluşturur.
4. **En İyi Çözüm Belirleme:** Sürüdeki tüm yarasalar tarafından bilinen en iyi çözüm belirlenir.

5. **Sesi Yükseltme ve Düşürme:** Yarasalar, en iyi çözüme göre frekanslarını ve ataletlerini günceller.
6. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

BAT Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

BAT Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

BAT Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.5. Gravitasyonel Arama Algoritması (GSA)

Gravitasyonel Arama Algoritması (GSA), Newton'un evrensel kütleçekim yasasından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2009 yılında Rashedi, Erol ve Hosseinzadeh tarafından geliştirilmiştir.

GSA Algoritmasının Temel Prensipleri:

- **Kütleçekim:** GSA algoritmasında, her bir çözüm bir cisim olarak temsil edilir ve cisimler arasındaki kütleçekim kuvveti, çözümlerin birbirine çekimini gösterir.
- **Ağır Nesneler:** Daha iyi çözümler, daha büyük kütleye sahip cisimler olarak temsil edilir.
- **Hareket:** Cisimler, kütleçekim kuvvetinin etkisiyle birbirlerine doğru hareket eder.
- **En İyi Çözüm:** En iyi çözüm, en büyük kütleye sahip cisimdir.

GSA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Kütlelerin Hesaplanması:** Her bir çözümün kütlesi, fitness değerine göre hesaplanır.
3. **Hız Güncelleme:** Cisimlerin hızları, kütleçekim kuvvetinin etkisiyle güncellenir.
4. **Konum Güncelleme:** Cisimler, hızlarını kullanarak konumlarını günceller.
5. **En İyi Çözüm Belirleme:** En iyi çözüm, en büyük kütleye sahip cisim olarak belirlenir.
6. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

GSA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

GSA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

GSA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.6. Ateşböceği Algoritması (FA)

Ateşböceği Algoritması (FA), ateşböceklerinin ışık üretme ve birbirleriyle iletişim kurma davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2008 yılında Xin-She Yang tarafından geliştirilmiştir.

FA Algoritmasının Temel Prensipleri:

- **Işık Yoğunluğu:** Ateşböcekleri, ışık yoğunluğunu kullanarak birbirleriyle iletişim kurarlar.
- **Çekicilik:** Daha parlak ışık, daha çekici bir etkiye sahiptir.
- **Mesafe:** Ateşböcekleri arasındaki çekim, aralarındaki mesafeyle ters orantılıdır.
- **En İyi Çözüm:** En parlak ışık, en iyi çözümü temsil eder.

FA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (çözümler kümesi) oluşturulur.
2. **Işık Yoğunluğunun Hesaplanması:** Her bir çözümün ışık yoğunluğu, fitness değerine göre hesaplanır.
3. **Hareket:** Ateşböcekleri, daha parlak ışığa doğru hareket eder.
4. **Çözüm Güncelleme:** Ateşböcekleri, hareket ettikleri yere göre çözümlerini günceller.
5. **En İyi Çözüm Belirleme:** En parlak ışık, en iyi çözüm olarak belirlenir.
6. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

FA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

FA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

FA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.7. Harmony Search Algorithm (HSA)

Harmony Search Algorithm (HSA), müzik doğaçlama ve en iyi armoninin peşinden koşma fikrinden ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2007 yılında Zong Woo Geem tarafından geliştirilmiştir.

HSA Algoritmasının Temel Prensipleri:

- **Müzisyenler:** HSA algoritmasında, her bir çözüm bir müzisyen tarafından temsil edilir.
- **Enstrümanlar:** Her bir değişken, bir enstrüman tarafından temsil edilir.
- **Notlar:** Her bir değişkenin alabileceği değerler, bir enstrümandaki notalar tarafından temsil edilir.
- **Armoni:** Bir çözümdeki tüm değişkenlerin değerlerinin kombinasyonu, bir armoni olarak temsil edilir.
- **En İyi Armoni:** En iyi çözüm, en iyi armoniyi temsil eder.

HSA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (müzisyenler kümesi) oluşturulur.
2. **Harmonilerin Oluşturulması:** Her bir müzisyen, rastgele seçtiği notalardan oluşan bir armoni oluşturur.
3. **Harmonilerin Değerlendirilmesi:** Her bir armoninin uyumu (fitness değeri) hesaplanır.
4. **Hafıza Güncelleme:** En iyi armoni hafızaya kaydedilir.
5. **Doğaçlama:** Müzisyenler, hafızadaki en iyi armoniden ilham alarak yeni armoniler oluşturur.
6. **Pürüzsüzleştirme:** Yeni armoniler, rastgele seçilen notalarla değiştirilerek iyileştirilir.
7. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

HSA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

HSA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

HSA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.8. Cuckoo Search Algorithm (CSA)

Cuckoo Search Algorithm (CSA), guguk kuşlarının yumurtalarını diğer kuşların yuvalarına bırakma davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2009 yılında Xin-She Yang tarafından geliştirilmiştir.

CSA Algoritmasının Temel Prensipleri:

- **Yumurtlama:** Guguk kuşları, rastgele yuvalara yumurtlar.
- **Yuvadan Atma:** Yuvadaki diğer yumurtaları fark eden kuşlar, yuvadan atar.
- **En İyi Yuva:** En iyi çözümü içeren yuva, en iyi yuva olarak adlandırılır.

CSA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (yuvalar kümesi) oluşturulur.
2. **Yumurtlama:** Her bir guguk kuşu, rastgele bir yuvaya bir yumurta bırakır.
3. **Yuvadan Atma:** Yuvadaki diğer yumurtaları fark eden kuşlar, yuvadan atar.
4. **Yumurtaların Değerlendirilmesi:** Her bir yumurtanın (çözümün) fitness değeri hesaplanır.
5. **Lévy Uçuşu:** Guguk kuşları, Lévy uçuşu kullanarak yeni yuvalar arar.
6. **En İyi Yuva Belirleme:** En iyi çözümü içeren yuva, en iyi yuva olarak belirlenir.
7. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

CSA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

CSA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

CSA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.9. Sinep Algoritması (FA)

Sinep Algoritması (FA), sincap kolonilerinin yiyecek arama ve saklama davranışlarından ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 2007 yılında Xin-She Yang tarafından geliştirilmiştir.

FA Algoritmasının Temel Prensipleri:

- **Yiyecek Arama:** Sincaplar, yiyecek bulmak için ormanda dolaşırlar.
- **Yiyecek Saklama:** Sincaplar, buldukları yiyecekleri saklamak için yuvalar ve depolar kullanırlar.
- **Unutma:** Sincaplar, zamanla yiyeceklerin yerini unutabilirler.
- **En İyi Yiyecek:** En iyi çözümü temsil eden yiyecek.

FA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (sincaplar kümesi) oluşturulur.
2. **Yiyecek Arama:** Sincaplar, rastgele arama ve Lévy uçuşu kullanarak yiyecek ararlar.
3. **Yiyecek Saklama:** Sincaplar, buldukları yiyecekleri hafızalarına ve yuvalarına saklarlar.
4. **Unutma:** Sincaplar, zamanla yiyeceklerin yerini unutabilirler ve bu yiyecekler kaybolur.
5. **Yiyeceklerin Değerlendirilmesi:** Her bir yiyeceğin (çözümün) fitness değeri hesaplanır.
6. **En İyi Yiyecek Belirleme:** En iyi çözümü temsil eden yiyecek, en iyi yiyecek olarak belirlenir.
7. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

FA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

FA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

FA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.10. Memetic Algoritmalar (MA)

Memetic Algoritmalar (MA), evrimsel algoritmaların ve yerel arama tekniklerinin bir kombinasyonunu kullanan meta-sezgisel algoritmalardır. Bu algoritmalar, evrimsel algoritmaların küresel arama yeteneğini ve yerel arama tekniklerinin lokal arama yeteneğini birleştirerek optimizasyon problemlerini çözmeyi amaçlar.

MA Algoritmalarının Temel Prensipleri:

- **Evrimsel Algoritmalar:** Genetik algoritmalar, partikül sürü optimizasyonu ve diferansiyel evrim gibi algoritmalar.
- **Yerel Arama Teknikleri:** Tepe tırmanışı, bencil arama ve tabu arama gibi teknikler.
- **Mem:** Memetic algoritmalarda, her birey bir kromozom ve bir memden oluşur. Kromozom, evrimsel algoritma tarafından optimize edilirken, mem yerel arama tarafından optimize edilir.

MA Algoritmalarının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (bireyler kümesi) oluşturulur.
2. **Değerlendirme:** Her bir bireyin fitness değeri hesaplanır.
3. **Ebeveyn Seçimi:** Evrimsel algoritma, ebeveynleri seçmek için kullanılır.
4. **Çaprazlama:** Ebeveynlerden yeni bireyler oluşturmak için çaprazlama kullanılır.
5. **Mutasyon:** Yeni bireylerin çeşitliliğini artırmak için mutasyon kullanılır.
6. **Yerel Arama:** Her bireyin memu, yerel arama teknikleri ile optimize edilir.
7. **Seçim:** Yeni popülasyon oluşturmak için en iyi bireyler seçilir.
8. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

MA Algoritmalarının Avantajları:

- Evrimsel algoritmaların küresel arama yeteneği ve yerel arama tekniklerinin lokal arama yeteneğini birleştirir.
- Diğer meta-sezgisel algoritmalara göre daha hızlı ve daha iyi çözümler üretebilir.
- Karmaşık problemlerde optimal çözümü bulma şansını artırır.

MA Algoritmalarının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha karmaşıktır.
- Daha fazla parametreye sahip olduğu için ayarlanması daha zordur.

MA Algoritmalarının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.11. Tabu Arama Algoritması (TA)

Tabu arama algoritması (TA), kombinatoryal optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 1986 yılında Fred Glover tarafından geliştirilmiştir. TA algoritması, lokal optimumlara takılmadan optimal veya yakın optimal çözümler bulmaya çalışır.

TA Algoritmasının Temel Prensipleri:

- **Komşu Çözümler:** Bir çözümün komşuları, o çözümden tek bir hamle ile ulaşılabilen çözümlerdir.
- **Tabu Listesi:** Son zamanlarda ziyaret edilen çözümleri içeren bir listedir.
- **Tabu Kriteri:** Bir çözümün tabu listesine eklenip eklenmeyeceğini belirleyen kriterdir.
- **Aspiration Kriteri:** Tabu listede olmasına rağmen bir çözümün kabul edilebileceği kriterdir.

TA Algoritmasının Adımları:

1. **Başlangıç Çözümü:** Başlangıç çözümü rastgele veya problemle ilgili bilgilere dayalı olarak oluşturulur.
2. **Komşu Çözümler:** Başlangıç çözümünün komşuları bulunur.
3. **Komşu Çözümlerin Değerlendirilmesi:** Komşu çözümlerin fitness değerleri hesaplanır.

4. **Tabu Listesi Kontrolü:** Komşu çözümler tabu listesinde olup olmadığı kontrol edilir.
5. **En İyi Komşu Çözümün Seçimi:** Tabu listesinde olmayan ve aspiration kriterini sağlayan en iyi komşu çözüm seçilir.
6. **Hareket:** Seçilen komşu çözüme hareket edilir.
7. **Tabu Listesi Güncelleme:** Tabu listesine yeni çözümler eklenir ve eski çözümler silinir.
8. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

TA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

TA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

TA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.12. Genetik Algoritma (GA)

Genetik Algoritma (GA), canlılardaki genetik kodlama ve evrimsel süreçten ilham alan ve optimizasyon problemlerini çözmek için kullanılan bir meta-sezgisel algoritmadır. 1975 yılında John Holland tarafından geliştirilmiştir.

GA Algoritmasının Temel Prensipleri:

- **Popülasyon:** Birbirinden farklı çözümlerden oluşan bir topluluktur.
- **Kromozom:** Bir çözümü temsil eden bir veri dizisidir.
- **Gen:** Kromozomu oluşturan her bir elemandır.
- **Uygunluk Değeri:** Bir çözümün ne kadar iyi olduğunu gösteren bir değerdir.
- **Çaprazlama:** İki kromozomdan yeni kromozomlar oluşturma işlemidir.
- **Mutasyon:** Bir kromozomdaki genlerin rastgele değiştirilmesi işlemidir.

GA Algoritmasının Adımları:

1. **Popülasyon Oluşturma:** Rastgele bir başlangıç popülasyonu (kromozomlar kümesi) oluşturulur.
2. **Uygunluk Değerlendirilmesi:** Her bir kromozomun uygunluk değeri hesaplanır.
3. **Ebeveyn Seçimi:** Üreme için ebeveyn kromozomlar seçilir.
4. **Çaprazlama:** Ebeveyn kromozomlardan yeni kromozomlar oluşturmak için çaprazlama kullanılır.
5. **Mutasyon:** Yeni kromozomların çeşitliliğini artırmak için mutasyon kullanılır.
6. **Yeni Popülasyon Oluşturma:** Yeni kromozomlar kullanılarak yeni bir popülasyon oluşturulur.
7. **Durdurma Kriterine Ulaşana Kadar Tekrarla:** Yukarıdaki adımlar, durdurma kriterine (örneğin, maksimum iterasyon sayısı) ulaşana kadar tekrarlanır.

GA Algoritmasının Avantajları:

- Basit ve anlaşılması kolay bir algoritmadır.
- Hızlı ve etkili bir şekilde çözüm üretebilir.
- Farklı optimizasyon problemlerine uygulanabilir.
- Birçok parametresi olmadığı için ayarlanması kolaydır.

GA Algoritmasının Dezavantajları:

- Diğer meta-sezgisel algoritmalara göre daha az hassas olabilir.
- Karmaşık problemlerde optimal çözümü bulmakta zorlanabilir.
- Büyük boyutlu problemlerde hesaplama maliyeti yüksek olabilir.

GA Algoritmasının Uygulama Alanları:

- Mühendislik tasarımı
- Makine öğrenmesi
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Robotik

2.13. Yapay Sinir Ağları (ANN)

Yapay sinir ağları (ANN), insan beyninin sinir ağlarından ilham alan ve karmaşık problemleri çözmek için kullanılan bir hesaplama modelidir. 1940'larda Warren McCulloch ve Walter Pitts tarafından geliştirilmiştir. ANN'ler, insan beynindeki nöronlara benzer şekilde çalışan ve birbirine bağlı basit işlem birimlerinden oluşur.

ANN'lerin Temel Prensipleri:

- **Nöronlar:** ANN'lerin temel işlem birimleridir. Her nöron, bir dizi girişi işler ve tek bir çıkış üretir.
- **Ağırlıklar:** Nöronlar arasındaki bağlantıları temsil eder. Her bağlantının bir ağırlığı vardır ve bu ağırlık, bir nöronun diğer nöronun çıkışını ne kadar etkilediğini belirler.
- **Aktivasyon Fonksiyonu:** Bir nöronun çıkışını hesaplamak için kullanılır.
- **Öğrenme:** ANN'ler, örnekler üzerinden öğrenerek çalışır. Öğrenme sırasında, ağırlıklar ayarlanır ve bu sayede ANN, problemin çözümünü öğrenir.

ANN'lerin Türleri:

- **Yatay iletimli sinir ağları:** En yaygın ANN türüdür. Bu ağlarda, nöronlar katmanlar halinde düzenlenmiştir ve her katman bir sonraki katmana bilgi gönderir.
- **Yinelemeli sinir ağları:** Bu ağlarda, nöronlar birbiriyle bağlantılıdır ve bilgi geriye doğru akabilir.
- **Konvolüsyonel sinir ağları:** Görüntü işleme gibi görevler için özel olarak tasarlanmış ANN'lerdir.

ANN'lerin Uygulama Alanları:

- Görüntü işleme
- Konuşma tanıma
- Makine çevirisi
- Tahminleme
- Sınıflandırma
- Kontrol

ANN'lerin Avantajları:

- Karmaşık problemleri çözme yeteneğine sahiptir.
- İnsan beynine benzer şekilde öğrenme ve uyum sağlama yeteneğine sahiptir.
- Büyük veri kümelerinden öğrenmek için kullanılabilir.

ANN'lerin Dezavantajları:

- Yorumlamak zor olabilir.
- Eğitmek için çok fazla veriye ihtiyaç duyabilir.
- Hesaplama açısından pahalı olabilir.

2.14. Derin Öğrenme Tabanlı Sezgisel Algoritmalar

Derin öğrenme tabanlı sezgisel algoritmalar, yapay sinir ağlarını (ANN) ve sezgisel algoritmaları birleştiren bir grup algoritmadır. Bu algoritmalar, karmaşık optimizasyon problemlerini çözmek için derin öğrenmenin gücünü sezgisel algoritmaların esnekliğiyle birleştirir.

Derin Öğrenme Tabanlı Sezgisel Algoritmaların Türleri:

- **Derin pekiştirmeli öğrenme:** Bu algoritmalar, bir ajan ve çevresi arasındaki etkileşimi modellemek için derin sinir ağlarını kullanır. Ajan, çevreyle etkileşime girerek ve aldığı ödülleri maksimize etmeye çalışarak öğrenir.
- **Derin evrimsel algoritmalar:** Bu algoritmalar, popülasyonları temsil etmek için derin sinir ağlarını kullanır. Popülasyon, seçilim, çaprazlama ve mutasyon gibi işlemler yoluyla evrimleşir.
- **Derin Tabu Arama:** Bu algoritmalar, tabu listesini yönetmek için derin sinir ağlarını kullanır. Derin sinir ağı, hangi çözümlerin tabu listesine ekleneceğini ve hangilerinin çıkarılacağını belirler.

Derin Öğrenme Tabanlı Sezgisel Algoritmaların Avantajları:

- Karmaşık problemleri çözme yeteneğine sahiptir.
- Derin öğrenmenin gücünü sezgisel algoritmaların esnekliğiyle birleştirir.
- Büyük veri kümelerinden öğrenmek için kullanılabilir.

Derin Öğrenme Tabanlı Sezgisel Algoritmaların Dezavantajları:

- Yorumlamak zor olabilir.
- Eğitmek için çok fazla veriye ihtiyaç duyabilir.
- Hesaplama açısından pahalı olabilir.

2.15. Çok Hedefli Sezgisel Algoritmalar

Çok hedefli sezgisel algoritmalar (ÇHSA), birden fazla hedefi optimize etmek için kullanılan bir algoritma sınıfıdır. Bu algoritmalar, geleneksel sezgisel algoritmalara benzer şekilde çalışır, ancak birden fazla hedefe göre çözüm aramayı optimize ederler.

ÇHSA'ların Temel Prensipleri:

- **Hedefler:** Optimize edilecek birden fazla hedef fonksiyonu vardır.
- **Pareto Optimal Çözümler:** Hiçbir hedefin değerini bozmadan başka bir hedefin değerini iyileştirmek mümkün olmayan çözümlerdir.
- **Hakimiyet:** Bir çözümün, başka bir çözümden en az bir hedefte daha iyi olması ve diğer tüm hedeflerde eşit veya daha iyi olması durumudur.
- **Çeşitlilik:** Popülasyonda farklı Pareto optimal çözümlerin temsil edilmesi.

ÇHSA'ların Türleri:

- **Genetik algoritmalar:** En yaygın kullanılan ÇHSA türüdür. Bu algoritmalar, popülasyonu evrimleştirmek için seçilim, çaprazlama ve mutasyon gibi operatörleri kullanır.
- **Parçacık sürüsü optimizasyonu:** Bu algoritmalar, bir parçacık sürüsünün davranışını simüle ederek çalışır. Parçacıklar, en iyi çözümü bulmak için birbirleriyle etkileşime girerler.
- **Karınca kolonisi optimizasyonu:** Bu algoritmalar, karıncaların yiyecek bulmak için kullandıkları iletişim ve iş birliği mekanizmalarını simüle ederek çalışır.

ÇHSA'ların Uygulama Alanları:

- Mühendislik tasarımı
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Tedarik zinciri optimizasyonu

ÇHSA'ların Avantajları:

- Birden fazla hedefi optimize etme yeteneğine sahiptir.
- Pareto optimal çözümler kümesi bulma imkanı sunar.
- Farklı optimizasyon problemlerine uygulanabilir.

ÇHSA'ların Dezavantajları:

- Tek hedefli algoritmalara göre daha karmaşık olabilir.
- Hesaplama açısından daha pahalı olabilir.
- Pareto optimal çözümler kümesi arasında seçim yapmak zor olabilir.

2.16. Hibrit Sezgisel Algoritmalar

Hibrit sezgisel algoritmalar (HSA), birden fazla sezgisel algoritmayı birleştiren ve karmaşık optimizasyon problemlerini çözmek için kullanılan algoritmalardır. Bu algoritmalar, farklı algoritmaların en iyi özelliklerini birleştirerek daha iyi performans elde etmeyi amaçlar.

HSA'ların Temel Prensipleri:

- **Farklı sezgisel algoritmaların kombinasyonu:** HSA'lar, genetik algoritmalar, tabu arama, benzetilmiş tavlama gibi farklı sezgisel algoritmaları birleştirir.
- **Güçlü yönlerin birleştirilmesi:** HSA'lar, farklı algoritmaların güçlü yönlerini birleştirerek daha iyi bir arama yeteneği ve daha hızlı yakınsama elde etmeyi amaçlar.

- **Zayıflıkların giderilmesi:** HSA'lar, farklı algoritmaların zayıf yönlerini birbirleriyle telafi ederek daha sağlam ve tutarlı bir performans elde etmeyi amaçlar.

HSA'ların Türleri:

- **Paralel HSA'lar:** Farklı algoritmalar paralel olarak çalıştırılır ve bilgi alışverişi yapar.
- **Seri HSA'lar:** Farklı algoritmalar sıralı olarak çalıştırılır ve her algoritma bir sonraki algoritmaya bilgi aktarır.
- **Hiyerarşik HSA'lar:** Farklı algoritmalar farklı arama katmanlarında çalıştırılır.

HSA'ların Uygulama Alanları:

- Mühendislik tasarımı
- Finansal optimizasyon
- Üretim planlama
- Enerji optimizasyonu
- Tedarik zinciri optimizasyonu

HSA'ların Avantajları:

- Farklı algoritmaların en iyi özelliklerini birleştirir.
- Daha iyi performans elde etme imkânı sunar.
- Karmaşık problemleri çözmek için daha uygundur.

HSA'ların Dezavantajları:

- Tasarım ve uygulama karmaşık olabilir.
- Farklı algoritmaların parametrelerini ayarlamak zor olabilir.
- Hesaplama açısından daha pahalı olabilir.

2.17. Arama ve Kurtarma Optimizasyon Algoritması (AKOA)

AKOA, Dr. Hatice Erol Aksoy tarafından 2020 yılında geliştirilmiş bir sezgisel optimizasyon algoritmasıdır. Bu algoritma, insan davranışlarından ilham alarak tasarlanmıştır ve karmaşık ve çok boyutlu optimizasyon problemlerini çözmek için kullanılmaktadır.

AKOA'nın Temel Prensipleri:

- **Araştırma:** Arama ekibi, en iyi çözümü için arama sahasını keşfeder.
- **Kurtarma:** Arama ekibi, en iyi çözümü bulduktan sonra onu kurtarır ve diğer çözümlerle karşılaştırır.
- **Yerel Arama:** Arama ekibi, en iyi çözümün yakınında daha iyi çözümler bulmak için yerel arama yapar.
- **İletişim:** Arama ekibi, bilgi alışverişi yapmak ve koordinasyonu artırmak için birbirleriyle iletişim kurar.

AKOA'nın Aşamaları:

1. **Başlangıç:** Arama ekibi rastgele bir çözüm üretir.
2. **Değerlendirme:** Arama ekibi, çözümün değerini hesaplar.
3. **Hafıza:** Arama ekibi, en iyi çözümü hafızasına kaydeder.
4. **Komşu Çözümler:** Arama ekibi, mevcut çözümün komşu çözümlerini üretir.
5. **Hareket:** Arama ekibi, daha iyi bir çözüm bulmak için komşu çözümlere hareket eder.
6. **Durma Kriteri:** Durma kriteri karşılanana kadar 2-5 arasındaki adımlar tekrarlanır.

AKOA'nın Avantajları:

- Basit ve anlaşılır bir yapıya sahiptir.
- Farklı optimizasyon problemlerine uyarlanabilir.
- Hızlı ve etkili bir şekilde çözüm bulma imkânı sunar.
- Yerel optimallere takılma olasılığı düşüktür.

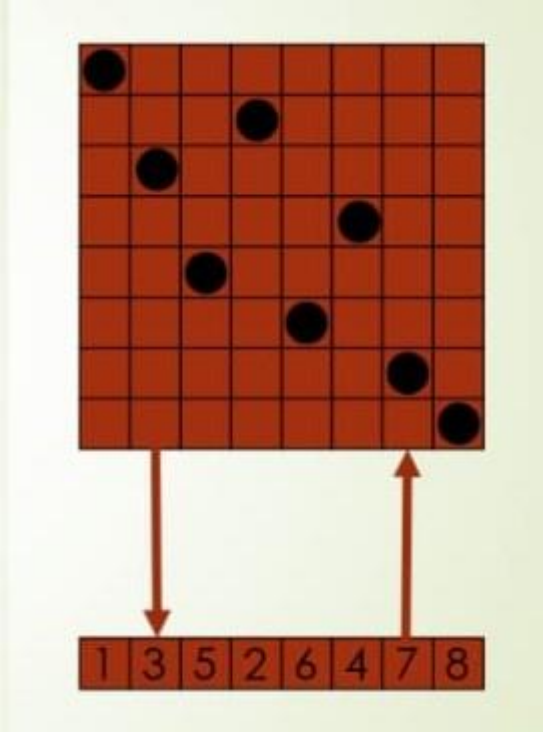
AKOA'nın Dezavantajları:

- Karmaşık problemlerde çözüm bulmak için uzun sürebilir.
- Parametre ayarı problemin özelliğine göre yapılmalıdır.
- Diğer sezgisel algoritmalarla karşılaştırıldığında daha yeni bir algoritmadır ve daha fazla araştırmaya ihtiyaç duymaktadır.

3. Genetik Algoritma ile n Vezir Problemi Uygulaması

8 Vezir Bulmacası, 8x8'lik bir satranç tahtasına 8 adet vezirin hiçbiri olağan vezir hamleleriyle birbirini alamayacak biçimde yerleştirmesi sorunudur. Her bir vezirin konumunun diğer bir vezire saldırmasına engel olması için hiçbir vezir başka bir vezirle aynı satıra, aynı kolona ya da aynı köşegene yerleştirilemez. 8 Vezir Bulmacası daha genel olan n Vezir Bulmacası'nın özel bir durumudur.

n Vezir Bulmacası, n ≥ 4 için n×n boyutunda bir satranç tahtasına n adet vezirin birbirini alamayacak biçimde yerleştirilmesi sorunudur.



Problemin kısıtları:

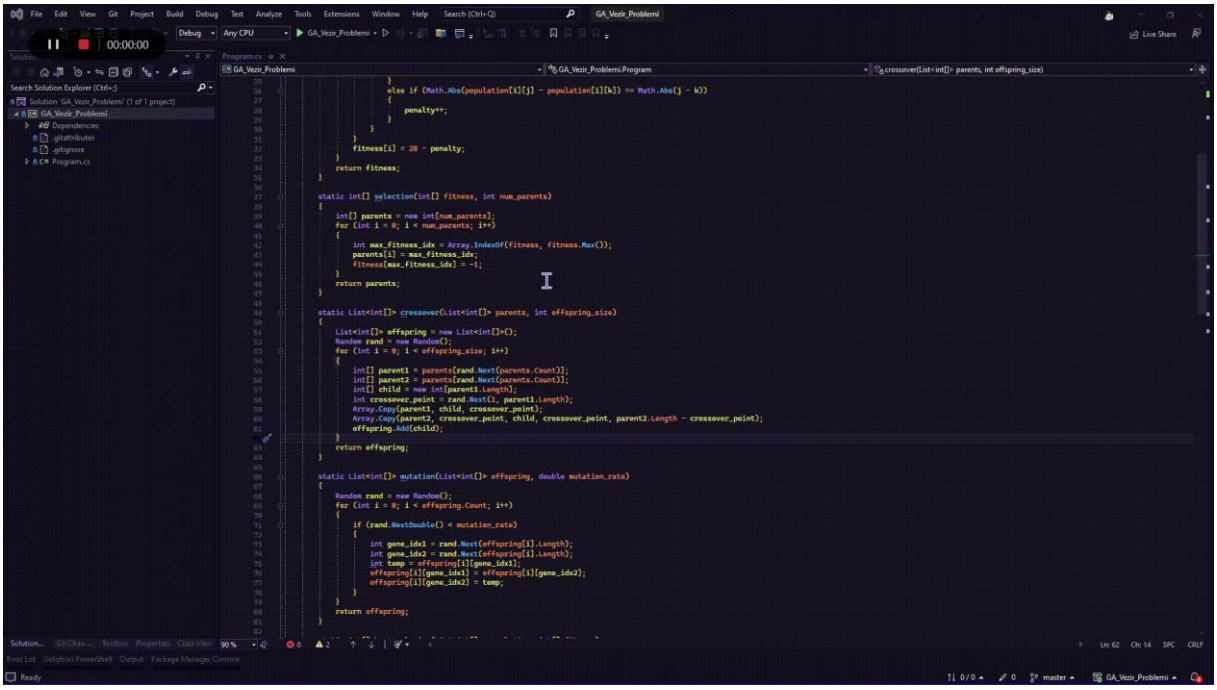
- Bir 8x8 satranç tahtası üzerinde, 8 vezirin yerleştirilmesi gerekmektedir.
- Her vezir, aynı satır, sütun veya diyagonalde başka bir vezir olmamalıdır.
- Amaç, tüm vezirlerin birbirini tehdit etmediği, yani tüm kısıtları sağlayan bir yerleşim bulmaktır.
- Bir vezir icin ceza, tehdit ettigi vezir sayisidir.
- Popülasyon büyüklüğü: 10
- Caprazlama Oranı: %60 (Permutasyon)
- Mutasyon Oranı: %1 (Insert veya Swap)
- Strateji: Popülasyondaki en kötüyü yer degiştirir.
- Iterasyon Sayisi: 10

Genetik algoritma kullanarak vezir problemi çözümü:

- Başlangıçta, popülasyon, rasgele 8 vezir yerleşimi içerecek şekilde oluşturulur.
- Fitness fonksiyonu, her bir vezirin tehdit ettiği diğer vezir sayısını hesaplar ve toplam tehdit sayısını döndürür. Cezalandırma fonksiyonu, tehdit sayısının tersi olarak hesaplanır. Yani, daha az tehdit edilen vezirler daha yüksek puan alır.
- Seçim işlemi, fitness değerine göre turnuva seçimi kullanarak yapılır. Popülasyonun en iyi çözümleri seçilir ve ebeveyn olarak atanır.
- Çaprazlama işlemi, bir ebeveyn çiftinin rastgele seçilen noktalarından bölünür ve çocuklar oluşturulur. Bu örnekte, permütasyon çaprazlama yöntemi kullanılır.
- Mutasyon işlemi, popülasyonun belirli bir yüzdesinde, rastgele bir yerde veya iki vezirin yerini değiştirerek yapılır.
- Yeni popülasyon, ebeveynler ve çocuklar arasından seçilerek oluşturulur. Seçim işlemi, fitness değerine göre turnuva seçimi kullanarak yapılır.
- Popülasyon, belirli bir iterasyon sayısı veya belirli bir süre boyunca değişmediğinde durur.

Tabu arama kullanarak vezir problemi çözümü:

- Başlangıçta, popülasyon, rasgele 8 vezir yerleşimi içerecek şekilde oluşturulur.
- Fitness fonksiyonu, her bir vezirin tehdit ettiği diğer vezir sayısını hesaplar ve toplam tehdit sayısını döndürür. Cezalandırma fonksiyonu, tehdit sayısının tersi olarak hesaplanır. Yani, daha az tehdit edilen vezirler daha yüksek puan alır.
- Başlangıçta, en iyi çözüm ve çözüme ait fitness değeri kaydedilir.
- Tabu Listesi, mevcut çözümü değiştirmeye yönelik kısıtlamalar getirerek, daha önce denenmiş fakat daha kötü çözümlere geri dönüşü engeller. Bu sayede, arama alanında daha etkili bir keşif yaparak optimum çözüme daha hızlı ulaşılabilir.
- Her iterasyonda, en iyi çözüm ve çözüme ait fitness değeri kaydedilir.
- Yeni çözümler oluşturulur ve fitness değerleri hesaplanır.
- Tabu Listesi kullanılarak, mevcut çözüme dönüş yapmamak koşulu ile en iyi yeni çözüm seçilir.
- Popülasyonun belirli bir yüzdesi, rasgele mutasyona uğrar.
- Popülasyonun belirli bir yüzdesi, en kötü çözümleri yer değiştirmek için seçilir ve yer değiştirme işlemi yapılır.
- Popülasyon, belirli bir iterasyon sayısı veya belirli bir süre boyunca değişmediğinde durur.



Kodlamada her bir adım aşağıda açıklanmıştır:

- fitness(population) fonksiyonu
- Popülasyon içindeki her bir bireyin uygunluğunu hesaplar ve uygunluk değerlerini bir dizi olarak döndürür.
- Uygunluk, her bir vezirin diğer vezirlerle çakışıp çakışmadığını kontrol etmek için hesaplanır. Her bir çift vezir arasındaki yatay, dikey ve çapraz farkları kontrol edilir. Herhangi bir çakışma varsa, uygunluk değeri azaltılır.
- selection(fitness, num_parents) fonksiyonu
- Fitness değerleri ile birlikte popülasyondan num_parents sayıda en iyi bireyleri seçer.
- En iyi bireyler, fitness değerlerine göre seçilir.
- crossover(parents, offspring_size) fonksiyonu
- Belirtilen sayıda yeni bireyler üretir.
- Üretim için rastgele seçilen iki ebeveyn arasında crossover yapılır. Crossover noktası rastgele belirlenir ve yeni birey, her bir ebeveynin genlerinin bir kısmını alır.
- mutation(offspring, mutation_rate) fonksiyonu
- Yeni bireylerin bazı genlerinde değişiklik yapar.
- Her bir bireyin her bir geninde, belirtilen mutasyon oranına göre değişiklik yapılır. Bu, rastgele seçilen iki genin yerlerinin değiştirilmesiyle yapılır.
- best_solution(population, fitness) fonksiyonu
- Popülasyondaki en iyi bireyin değerini döndürür.
- Main fonksiyonu
- Programın ana işlevi.
- İlk olarak, popülasyon boyutu, çaprazlama oranı, mutasyon oranı ve iterasyon sayısı gibi parametreler ayarlanır.
- Ardından, rastgele bir popülasyon yaratılır ve belirtilen sayıda iterasyon yapılır.
- Her bir iterasyonda, fitness değerleri hesaplanır ve ebeveynler seçilir.

- Crossover adımı: crossover metodunu çağırarak, seçilen ebeveynlerden yeni çocuklar oluşturuyoruz. Offspring listesine eklenen yeni çocukları tutuyoruz.
- Mutasyon adımı: mutation metodunu çağırarak, offspring listesindeki bireylerin genlerinde mutasyon yapıyoruz.
- Yeni populasyon oluşturma adımı: population listesini temizleyerek, ebeveynleri ve çocukları ekliyoruz.
- En iyi çözüm adımı: best_solution metodunu çağırarak, yeni populasyon için en iyi çözümü buluyoruz.
- Son olarak, Console.WriteLine() yöntemini kullanarak her iterasyonda en iyi çözümü yazdırıyoruz.

İterasyonlar devam ederken, her iterasyonda en iyi çözüm geliştirilir ve ekrana yazdırılır. Bu şekilde, tüm iterasyonlar tamamlandığında, en iyi çözüm bulunur.

KAYNAKÇA

https://www.deeplearningbook.org/

https://dergipark.org.tr/tr/download/article-file/1366974

https://dergipark.org.tr/tr/download/article-file/193945

https://dergipark.org.tr/tr/download/article-file/929485

https://arxiv.org/abs/2203.04898

https://arxiv.org/abs/1004.0250

https://arxiv.org/abs/2203.04898

https://arxiv.org/abs/2005.14373

https://arxiv.org/abs/2103.11818

https://ieeexplore.ieee.org/document/6608204

https://users.encs.concordia.ca/~chvatal/8queens.html

https://www.aiai.ed.ac.uk/~gwickler/eightqueens.html

https://ieeexplore.ieee.org/document/10348342