Benzetim Tavlama (Simulated Annealing)(SA)

Dilaver ŞAHİN
Fırat Üniversitesi Adli Bilişim Mühendisliği

- ## Benzetim Tavlama Nedir?
- **Tanım**: Benzetim tavlama, yüksek enerji durumundan düşük enerji durumuna geçiş yaparak en uygun çözümü bulmaya çalışan bir optimizasyon tekniğidir.
- **Kökeni**: Metalurji sürecinde bir metalin ısıtılması ve yavaşça soğutulması fikrine dayanır. Bu süreç, metalin iç yapısının en düşük enerji seviyesine ulaşmasını sağlar.
- ## Benzetim Tavlamanın Temel Bileşenleri
- **Enerji Fonksiyonu (Objective Function)**: Optimizasyon probleminin hedef fonksiyonudur. Minimize veya maksimize edilmek istenir.
- **Sıcaklık (Temperature)**: Çözüm arama sürecinin kontrol parametresidir. Yüksek sıcaklıklarda daha büyük adımlar atılabilirken, düşük sıcaklıklarda ince ayar yapılır.
- **Soğuma Programı (Cooling Schedule)**: Sıcaklığın zamanla nasıl düşürüleceğini belirleyen kurallardır. Genellikle geometrik veya logaritmik olarak azaltılır.
- **Geçiş Olasılığı (Transition Probability)**: Mevcut çözümden yeni bir çözüme geçişin kabul edilip edilmeyeceğini belirler. Yüksek sıcaklıklarda kötü çözümler de kabul edilebilirken, düşük sıcaklıklarda sadece iyi çözümler kabul edilir.
- ## Algoritmanın Adımları
- **Başlangıç Durumu**: Rastgele bir başlangıç çözümü seçilir ve başlangıç sıcaklığı belirlenir.
- **Iterasyon**:
    - Yeni bir çözüm oluşturulur (komşu çözüm).
    - Enerji farkı hesaplanır (ΔE).
    - ΔE < 0 ise yeni çözüm kabul edilir; ΔE >= 0 ise kabul olasılığı hesaplanır (e^(-ΔE/T)).
- **Soğutma**: Sıcaklık, soğuma programına göre düşürülür.
- **Durdurma Kriteri**: Sıcaklık yeterince düşükse veya belirli bir iterasyon sayısına ulaşılmışsa algoritma durdurulur.
- ## Benzetim Tavlamanın Uygulama Alanları
- **Mühendislik**: Karmaşık sistemlerin optimizasyonu (örneğin, elektronik devre tasarımı).
- **Bilgisayar Bilimleri**: Gezgin Satıcı Problemi (TSP), çizelgeleme problemleri.
- **Fizik ve Kimya**: Molekül yapılandırmaları ve enerji minimizasyonu.
- **Ekonomi**: Portföy optimizasyonu.
- ## Benzetim Tavlamanın Avantajları ve Dezavantajları

Avantajlar:

- Küresel optimizasyon yapabilme yeteneği.
- Yerel minimumlardan kaçınma yeteneği.
- Basit ve geniş bir uygulama yelpazesi.

Dezavantajlar:

- Uygulama süresi genellikle uzundur.
- Parametre seçimleri (başlangıç sıcaklığı, soğuma programı) çok öneme sahiptir ve deneysel olarak belirlenmesi gerekebilir.

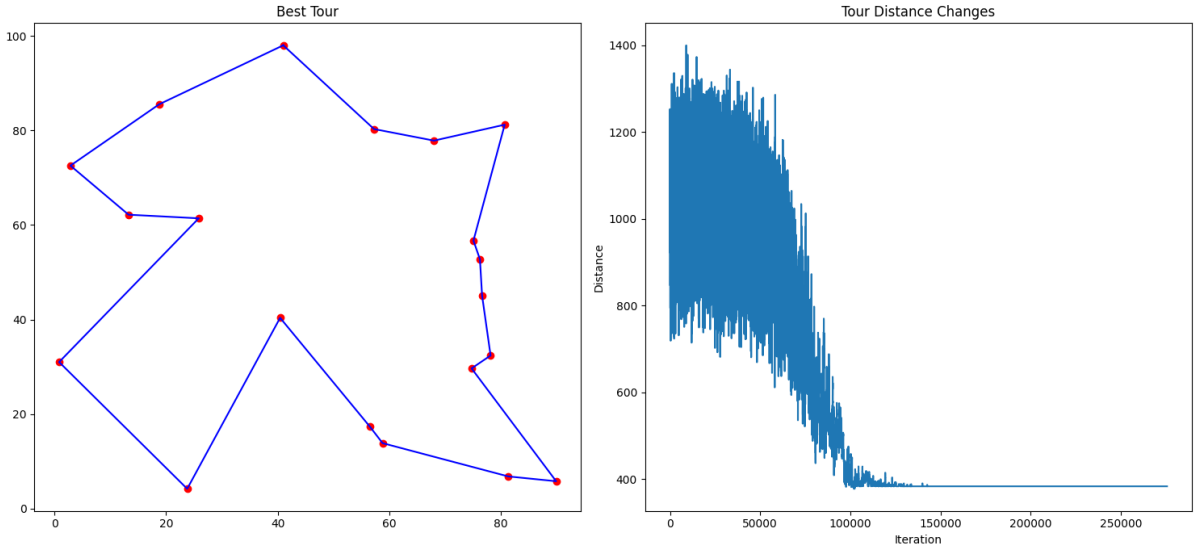
Örnek Uygulama

Bir örnekle açıklamak gerekirse, bir gezgin satıcı problemini çözmek için benzetim tavlama algoritması şu adımları takip edebilir:

- Rastgele bir şehir turu oluşturulur.
- Rastgele seçilen iki şehrin yerleri değiştirilir ve yeni turun toplam uzunluğu hesaplanır.
- Yeni tur, enerji fonksiyonuna göre değerlendirilir ve belirlenen kurallara göre kabul edilir veya reddedilir.
- Bu işlem, sıcaklık düşürülerek ve durdurma kriterine ulaşılana kadar tekrarlanır.

Sonuç

Benzetim tavlama, özellikle karmaşık optimizasyon problemleri için güçlü bir araçtır. Doğru uygulandığında, birçok farklı alanda başarılı çözümler üretebilir. Ancak, parametre ayarları ve algoritmanın doğru yapılandırılması önemlidir.



Açıklamalar:

- **Fonksiyon Tanımı:**
  - `simulated_annealing`: Bu fonksiyon, Travelling Salesman Problem (TSP) için Simulated Annealing algoritmasını uygular.
- **Parametreler:**
  - `cities`: Şehirlerin koordinatlarını içeren bir numpy dizisi. Her şehir, x ve y koordinatlarıyla tanımlanır.
  - `initial_temperature`: Algoritmanın başlangıç sıcaklığı. Bu, algoritmanın başındaki kabul edilebilirlik kriterini belirler.
  - `cooling_rate`: Sıcaklığın her iterasyonda ne kadar azalacağını belirten soğuma oranı. Çarpılarak eklendiği için 1 den küçük 0dan büyük olur.
  - `stopping_temperature`: Algoritmanın çalışmayı bırakacağı sıcaklık eşiği. Bu eşiğin altına düşüldüğünde algoritma durur.
- **Değişkenler:**
  - `n`: Şehir sayısı.
  - `best_tour`: Şu ana kadar bulunan en iyi tur, şehir indekslerinin bir listesi olarak temsil edilir.
  - `best_distance`: Şu ana kadar bulunan en kısa tur uzunluğu.
  - `distances`: Her iterasyonda bulunan en iyi tur uzunluklarını saklayan bir liste.
  - `temperature`: Şu anki sıcaklık değeri.
  - `iteration`: Döngü sayacı.
- **Algoritma:**
  - Algoritma, sıcaklık belirtilen eşik değerin üstünde olduğu sürece çalışır. İç döngü, her sıcaklık değerinde birkaç iterasyon yaparak iki şehir arasında değişim yapar.
  - Değişim sonrası yeni turun uzunluğu hesaplanır ve eski tur ile karşılaştırılır.
  - Yeni tur, eski tura göre daha kısa veya belirli bir olasılık ile kabul edilebilir.
  - Sıcaklık her iterasyonda `cooling_rate` ile azaltılır ve ilerleme çubuğu güncellenir.
- **Sonuçlar:**
  - `best_tour`: En iyi bulunan tur.
  - `best_distance`: En kısa tur uzunluğu.
  - `distances`: Her iterasyonda bulunan en iyi tur uzunluklarını içeren liste.

- **Kütüphanelerin İçeri Aktarılması:**

  - `numpy`, `matplotlib.pyplot`, `logging`, `datetime`, ve `tqdm` kütüphaneleri kullanılıyor. `tqdm` kütüphanesi, simülasyon sürecini görsel olarak izlemeyi sağlayan bir ilerleme çubuğu ekler.

- **Renkli Terminal Çıktıları:**

  - `print_green` ve `input_green` fonksiyonları, terminalde renkli çıktı ve giriş sağlar.

- **Parametreler ve Kullanıcı Girdileri:**

  - Kullanıcıdan şehir sayısı, başlangıç sıcaklığı, soğuma oranı gibi parametreler alınır.

- **Şehirlerin Oluşturulması:**
    - `generate_cities` fonksiyonu, belirtilen sayıda şehir oluşturur ve bunları rastgele koordinatlarla yerleştirir.
- **Benzetim Tavlama Algoritması:**
    - `simulated_annealing` fonksiyonu, şehirler arasında en kısa turu bulmak için benzetim tavlama algoritmasını uygular. İlerleme çubuğu, simülasyonun ilerleyişini gösterir.
- **Mesafe Hesaplama:**
    - `distance` fonksiyonu, iki şehir arasındaki Euclidean mesafeyi hesaplar.
- **Simülasyon ve Sonuçların Gösterimi:**
    - Şehirler ve en iyi tur görsel olarak `matplotlib` ile gösterilir. Ayrıca, mesafe değişimlerinin grafiği çizilir.
- **Loglama ve Dosya Kaydetme:**
    - Loglar `applog.log` dosyasına kaydedilir ve sonuçlar, tarih ve saat içeren bir dosya adıyla `results_YYYY-MM-DD_HH-MM-SS.png` olarak kaydedilir.
- **Hata Yönetimi:**
    - Kod çalışırken bir hata oluşursa, hata bilgisi log dosyasına yazılır ve kullanıcıya bir hata mesajı gösterilir.