Algoritma Analizi ve Tasarımı Dersi

2022-2023 Bahar Dönemi

**TSP PROBLEMİ ÖDEV RAPORU**

Teslim Tarihi: 11.04.2023

Öğrenci Adı: İlayda DİNÇBİLEK

Öğrenci Numarası: 212802040

1. Giriş

Bu proje, TCP probleminin optimal çözümünü bulmak için geliştirilen bir algoritma ile farklı boyutlardaki beş farklı dosya üzerinde çalışarak, her bir dosyanın optimal maliyet değerini çıkarmayı amaçlamaktadır.

Ödev için geliştirilen algoritmanın giriş verileri, bir koordinat dosyasıdır. Bu dosya, TSP probleminde verilen şehirlerin koordinatlarını içerir. Çıkış verileri ise en kısa turu oluşturan şehirlerin sırasını gösteren bir dizi ve bu turun uzunluğudur.

1. Geliştirme ve Çalıştırma Ortamları

Ödev Python ile visual studio code uygulamasında geliştirilmiştir.

1. Algoritma Akışı (pseudo code)

1. Kordinatdosyasiokuma(kordinatdosyasi) fonksiyonunu tanımla:

a. kordinatçiftleri adında bir boş liste oluştur

b. kordinatdosyasi dosyasını aç

c. Her satır için:

i. satırdaki x ve y koordinatlarına ayrıştır ve onları kordinatçiftleri listesine ekle

d. kordinatçiftleri listesini bir NumPy dizisine dönüştür ve dizi döndür

2. Mesafehesaplama(birincikordinat, ikincikordinat) fonksiyonunu tanımla:

a. İki kordinat çifti arasındaki mesafeyi hesapla

b. Mesafeyi döndür

3. Enyakinkomşualg(kordinatçiftleri, başlangiç=0) fonksiyonunu tanımla:

a. Şehir sayısını hesapla

b. Gezilmemiş şehirleri başlangıçta tüm şehirler olarak ayarla

c. Başlangıç şehrini bulunduğu şehir olarak ayarla

d. Yolu bulunduğu şehir ile başlat ve başlangıç şehrini yolun başına ekle

e. Gezilmemiş şehirler boş olana kadar:

i. En yakın komşuyu bul ve yolun sonuna ekle

ii. Bulunan şehri bulunduğu şehir olarak ayarla

iii. Bulunan şehri gezilmemiş şehirler listesinden kaldır

f. Yolu bir NumPy dizisine dönüştür ve dizi döndür

4. Yoluzunluğu(kordinatçiftleri, yol) fonksiyonunu tanımla:

a. Toplam yol uzunluğunu hesaplamak için sıfır olarak başlat

b. Yolun her çiftinde gezinerek:

i. İki şehir arasındaki mesafeyi hesapla

ii. Bu mesafeyi toplam yol uzunluğuna ekle

c. Toplam yol uzunluğunu döndür

5. Başlangıç zamanını kaydet

6. Verilen koordinat dosyasından kordinatları oku numpy array olarak ata

7. En yakın komşu algoritmasını kullanarak optimal bir yolu hesapla

8. Optimal yolu ve uzunluğunu yazdır

9. Bitiş zamanını al

10. Geçen süreyi yazdır

1. Sonuç

* TSP (Travelling Salesman Problem) problemi, bir satıcı veya işçinin, belirli bir sayıda şehri en kısa yolculuk mesafesiyle ziyaret etmesi gerektiği durumları modelleyen bir optimizasyon problemidir. Problemin amacı, satıcının her şehri sadece bir kez ziyaret ederek başlangıç noktasına dönmesi için gereken en kısa rota yolunu bulmaktır.
* TSP problemi, lojistik, turizm, elektronik, üretim planlaması, telekomünikasyon ve hava taşımacılığı gibi birçok alanda kullanılmaktadır. Örneğin, bir turizm acentesi, turistlerin en az zaman ve maliyetle en çok sayıda turistik yerleri gezebileceği bir rota planlamak için TSP problemini kullanabilir. Elektronik üreticileri, bir devre kartındaki parçaların en az miktarda tel kullanarak birbirlerine nasıl bağlanabileceğini belirlemek için TSP problemini kullanabilirler. Lojistik şirketleri, birçok şehirdeki müşterilerine en hızlı ve en ekonomik şekilde nasıl hizmet verebilecekleri için TSP problemini kullanabilirler. Lojistik şirketleri, tedarik zinciri yönetimi ve rotalama problemleri ile ilgilenen birçok endüstri, TSP probleminin çözümüne ihtiyaç duyar. Ayrıca, mikroçip üreticileri, DNA dizilimi analizleri ve diğer bilimsel araştırmalar da TSP probleminin çözümüne ihtiyaç duyan diğer örneklerdir.
* 5, 124, 1000, 5915, 11849 boyutlu dosyalar için algoritmanın boyut-çalışma zaman grafiği

|  |  |
| --- | --- |
| Dosya Boyutu(byte) | Çalışma Zamanı(s) |
| 5 | 0.00098s |
| 124 | 0.014909s |
| 1000 | 0.91049s |
| 5915 | 32.59609s |
| 11849 | 130.9681s |

* Bu tablo, TSP probleminin çözümü için kullanılan en yakın komşu algoritmasının performansını dosya boyutu ve çalışma zamanı açısından gösteriyor.
* Tablodan görülebileceği gibi, dosya boyutu arttıkça algoritmanın çalışma süresi de artmaktadır. Bu, algoritmanın işlem yükünün artması ve hesaplama maliyetinin artmasıyla açıklanabilir. Örneğin, 5 boyutlu bir problem için algoritma sadece 0.00098 saniyede çalışırken, 11849 boyutlu bir problem için çalışma süresi 130.9681 saniyeye kadar çıkmıştır.

NOT: Grafik için geçerli süreler şu özelliklerde kaydedilmiştir;

* İşlemci: AMD Ryzen 7 4800H with Radeon Graphics, 2901 Mhz, 8 Çekirdek, 16 Mantıksal İşlemci
* Ram :

Yüklü Fiziksel Bellek (RAM) 8,00 GB

Toplam Fiziksel Bellek 7,42 GB

Kullanılabilir Fiziksel Bellek 0,99 GB

Toplam Sanal Bellek 15,8 GB

Kullanılabilir Sanal Bellek 2,95 GB

* Disk: SAMSUNG MZVLQ512HBLU-00B00