ALGORİTMA ANALİZİ ÖDEV RAPORU

DİNAMİK PROGRAMLAMA

Grup 1 – Mine Elif Karslıgil

MÜDAFER KAYMAK

20011093

Mudafer.kaymak@std.yildiz.edu.tr

Video Linki : https://youtu.be/P7qcfEAZIjU

Problem Tanımı

Bu ödevde, kullanıcıdan alınan iki farklı uzunluktaki string üzerinde çalışan bir algoritma geliştirilmesi istenmektedir. Algoritma, verilen iki string içerisinde bulunan varsa eğer birden fazla en uzun sekansı belirlemeli ve bu ortak ifadeleri ekrana yazdırmalıdır. Kullanıcıdan alınan string'ler üzerinde gezinme ve karşılaştırma işlemlerini içeren algoritma, aynı uzunluğa sahip tüm en uzun ifadeleri belirleyerek kullanıcıya sunmalıdır.

Problemin Çözümü

Çözüm kullanıcıdan alınan iki farklı uzunluktaki string arasındaki en uzun ortak dizi problemini çözmeyi hedeflemektedir. İlk olarak, kullanıcıdan giriş alındıktan sonra dinamik programlama yaklaşımı kullanılarak bir matris oluşturulur. Bu matris, her iki string'in karakterlerini karşılaştırarak en uzun ortak diziyi bulmaya yönelik adımları içerir. Ayrıca, harfin sekans içinde seçilip seçilmediğini belirlemek için ek bir matris oluşturulur.

Her satır üzerindeki işlemler tamamlandığında, oluşturulan matrisler ekrana basılır. Ardından, tüm satırlar işlendikten sonra en uzun ortak dizinin boyutu belirlenir ve kullanıcıya sunulur. Eğer birden fazla en uzun ortak dizi bulunuyorsa, bu ifadeler de ekrana yazdırılır. Bu adımlar, problemi çözmek için sistematik bir yaklaşım sunarak algoritmanın adımlarını açıklar ve kullanıcının işlemi anlamasına yardımcı olur.

Karşılaşılan Zorluklar

Bu problemde karşılaşılan en büyük zorluk birden fazla LCS olduğunda yazdırılması kısmıydı. Recursive bir şekilde yazıldığında matriste birbirinden ayrılan yolların tekrar bir araya gelme ihtimali olduğundan dolayı aynı kelimelerin tekrar tekrar çıktı ekranında görülmesi sorunu vardı. Bunun için ek bir dizi tutularak yazdırılan kelimeler o diziye atandı. Yeni bir çıktı yazdırılırken diziden kontrol yapılarak daha önce yazılıp yazılmadığı öğrenildi.

Karmaşıklık Analizi

fillMatrix fonksiyonu kodda bulunan en karmaşık ve önem sırasında en yüksekte bulunan fonksiyondur bu fonksiton dinamik programlama matrisini dolduran bir işlemler kümesini içerir. İki adet for döngüsü içerisinde, her bir hücreye yapılan işlemler sabit sayıda adım içerdiği için toplam karmaşıklık O(m * n) olacaktır (burada m ve n, iki string'in uzunluklarıdır).

Ekran görüntüleri

1. Örnek

```
Enter the first String = MAERBPHCAPPBA

Enter the second String = AMRRERCHAZBZA


1. row of first matrix = 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1. row of second matrix = 0 0 0 0 0 0 0 0 0 0 0 0 0 0

2. row of first matrix = 0 0 1 1 1 1 1 1 1 1 1 1 1 1
2. row of second matrix = 0 2 3 0 0 0 0 0 0 0 0 0 0 0

3. row of first matrix = 0 1 1 1 1 1 1 1 2 2 2 2 2 2
3. row of second matrix = 0 3 2 2 2 2 2 2 2 3 0 0 0 3

4. row of first matrix = 0 1 1 1 1 2 2 2 2 2 2 2 2 2
4. row of second matrix = 0 1 2 2 2 3 0 0 0 2 2 2 2 2

5. row of first matrix = 0 1 1 2 2 2 3 3 3 3 3 3 3 3
5. row of second matrix = 0 1 2 3 3 2 3 0 0 0 0 0 0 0

6. row of first matrix = 0 1 1 2 2 2 3 3 3 3 3 4 4 4
6. row of second matrix = 0 1 2 1 2 2 1 2 2 2 2 3 0 0

7. row of first matrix = 0 1 1 2 2 2 3 3 3 3 3 4 4 4
7. row of second matrix = 0 1 2 1 2 2 1 2 2 2 2 1 2 2

8. row of first matrix = 0 1 1 2 2 2 3 3 4 4 4 4 4 4
8. row of second matrix = 0 1 2 1 2 2 1 2 3 0 0 2 2 2

9. row of first matrix = 0 1 1 2 2 2 3 4 4 4 4 4 4 4
9. row of second matrix = 0 1 2 1 2 2 1 3 2 2 2 2 2 2

10. row of first matrix = 0 1 1 2 2 2 3 4 4 5 5 5 5 5
10. row of second matrix = 0 3 2 1 2 2 1 1 2 3 0 0 0 3

11. row of first matrix = 0 1 1 2 2 2 3 4 4 5 5 5 5 5
11. row of second matrix = 0 1 2 1 2 2 1 1 2 1 2 2 2 2

12. row of first matrix = 0 1 1 2 2 2 3 4 4 5 5 5 5 5
12. row of second matrix = 0 1 2 1 2 2 1 1 2 1 2 2 2 2

13. row of first matrix = 0 1 1 2 2 2 3 4 4 5 5 6 6 6
13. row of second matrix = 0 1 2 1 2 2 1 1 2 1 2 3 0 0

14. row of first matrix = 0 1 1 2 2 2 3 4 4 5 5 6 6 7
14. row of second matrix = 0 3 2 1 2 2 1 1 2 3 2 1 2 3

Press Enter to see matrices and LCS!
```

```
Second Matrix
(0->skipX || 1->skipY || 2->skipX and skipY || 3->leftCross)
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 3 0 0 0 0 0 0 0 0 0 0 0
0 3 2 2 2 2 2 2 2 3 0 0 0 3
0 1 2 2 2 3 0 0 0 2 2 2 2 2
0 1 2 3 3 2 3 0 0 0 0 0 0 0
0 1 2 1 2 2 1 2 2 2 2 3 0 0
0 1 2 1 2 2 1 2 2 2 2 1 2 2
0 1 2 1 2 2 1 2 3 0 0 2 2 2
0 1 2 1 2 2 1 3 2 2 2 2 2 2
0 3 2 1 2 2 1 1 2 3 0 0 0 3
0 1 2 1 2 2 1 1 2 1 2 2 2 2
0 1 2 1 2 2 1 1 2 1 2 2 2 2
0 1 2 1 2 2 1 1 2 1 2 3 0 0
0 3 2 1 2 2 1 1 2 3 2 1 2 3

First Matrix
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1 1
0 1 1 1 1 1 1 1 1 2 2 2 2 2
0 1 1 1 1 2 2 2 2 2 2 2 2 2
0 1 1 2 2 2 3 3 3 3 3 3 3 3
0 1 1 2 2 2 3 3 3 3 3 4 4 4
0 1 1 2 2 2 3 3 3 3 3 4 4 4
0 1 1 2 2 2 3 3 4 4 4 4 4 4
0 1 1 2 2 2 3 4 4 4 4 4 4 4
0 1 1 2 2 2 3 4 4 5 5 5 5 5
0 1 1 2 2 2 3 4 4 5 5 5 5 5
0 1 1 2 2 2 3 4 4 5 5 5 5 5
0 1 1 2 2 2 3 4 4 5 5 6 6 6
0 1 1 2 2 2 3 4 4 5 5 6 6 7

Longest Common Subsequence Length = 7
Longest Common Subsequences
AERCABA
MERCABA
AERHABA
MERHABA
```

2. Örnek

```
Enter the first String = RSBECGREEAT

Enter the second String = KRESLGECRFET


1. row of first matrix = 0 0 0 0 0 0 0 0 0 0 0 0 0
1. row of second matrix = 0 0 0 0 0 0 0 0 0 0 0 0 0

2. row of first matrix = 0 0 1 1 1 1 1 1 1 1 1 1 1
2. row of second matrix = 0 2 3 0 0 0 0 0 0 3 0 0 0

3. row of first matrix = 0 0 1 1 2 2 2 2 2 2 2 2 2
3. row of second matrix = 0 2 1 2 3 0 0 0 0 0 0 0 0

4. row of first matrix = 0 0 1 1 2 2 2 2 2 2 2 2 2
4. row of second matrix = 0 2 1 2 1 2 2 2 2 2 2 2 2

5. row of first matrix = 0 0 1 2 2 2 2 3 3 3 3 3 3
5. row of second matrix = 0 2 1 3 2 2 2 3 0 0 0 3 0

6. row of first matrix = 0 0 1 2 2 2 2 3 4 4 4 4 4
6. row of second matrix = 0 2 1 1 2 2 2 1 3 0 0 0 0

7. row of first matrix = 0 0 1 2 2 2 3 3 4 4 4 4 4
7. row of second matrix = 0 2 1 1 2 2 3 2 1 2 2 2 2

8. row of first matrix = 0 0 1 2 2 2 3 3 4 5 5 5 5
8. row of second matrix = 0 2 3 1 2 2 1 2 1 3 0 0 0

9. row of first matrix = 0 0 1 2 2 2 3 4 4 5 5 6 6
9. row of second matrix = 0 2 1 3 2 2 1 3 2 1 2 3 0

10. row of first matrix = 0 0 1 2 2 2 3 4 4 5 5 6 6
10. row of second matrix = 0 2 1 3 2 2 1 3 2 1 2 3 2

11. row of first matrix = 0 0 1 2 2 2 3 4 4 5 5 6 6
11. row of second matrix = 0 2 1 1 2 2 1 1 2 1 2 1 2

12. row of first matrix = 0 0 1 2 2 2 3 4 4 5 5 6 7
12. row of second matrix = 0 2 1 1 2 2 1 1 2 1 2 1 3
```

```
Second Matrix
(0->skipX || 1->skipY || 2->skipX and skipY || 3->leftCross)
0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 3 0 0 0 0 0 0 3 0 0 0
0 2 1 2 3 0 0 0 0 0 0 0 0
0 2 1 2 1 2 2 2 2 2 2 2 2
0 2 1 3 2 2 2 3 0 0 0 3 0
0 2 1 1 2 2 2 1 3 0 0 0 0
0 2 1 1 2 2 3 2 1 2 2 2 2
0 2 3 1 2 2 1 2 1 3 0 0 0
0 2 1 3 2 2 1 3 2 1 2 3 0
0 2 1 3 2 2 1 3 2 1 2 3 2
0 2 1 1 2 2 1 1 2 1 2 1 2
0 2 1 1 2 2 1 1 2 1 2 1 3

First Matrix
0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 1 1 1 1 1 1
0 0 1 1 2 2 2 2 2 2 2 2 2
0 0 1 1 2 2 2 2 2 2 2 2 2
0 0 1 2 2 2 2 3 3 3 3 3 3
0 0 1 2 2 2 2 3 4 4 4 4 4
0 0 1 2 2 2 3 3 4 4 4 4 4
0 0 1 2 2 2 3 3 4 5 5 5 5
0 0 1 2 2 2 3 4 4 5 5 6 6
0 0 1 2 2 2 3 4 4 5 5 6 6
0 0 1 2 2 2 3 4 4 5 5 6 6
0 0 1 2 2 2 3 4 4 5 5 6 7

Longest Common Subsequence Length = 7
Longest Common Subsequences
RSECRET
```

3. Örnek

```
Enter the first String = ABCDZB

Enter the second String = BACEDAB


1. row of first matrix = 0 0 0 0 0 0 0 0
1. row of second matrix = 0 0 0 0 0 0 0 0

2. row of first matrix = 0 0 1 1 1 1 1 1
2. row of second matrix = 0 2 3 0 0 0 3 0

3. row of first matrix = 0 1 1 1 1 1 1 2
3. row of second matrix = 0 3 2 2 2 2 2 3

4. row of first matrix = 0 1 1 2 2 2 2 2
4. row of second matrix = 0 1 2 3 0 0 0 2

5. row of first matrix = 0 1 1 2 2 3 3 3
5. row of second matrix = 0 1 2 1 2 3 0 0

6. row of first matrix = 0 1 1 2 2 3 3 3
6. row of second matrix = 0 1 2 1 2 1 2 2

7. row of first matrix = 0 1 1 2 2 3 3 4
7. row of second matrix = 0 3 2 1 2 1 2 3

Press Enter to see matrices and LCS!
```

```
Second Matrix
(0->skipX || 1->skipY || 2->skipX and skipY || 3->leftCross)
0 0 0 0 0 0 0 0
0 2 3 0 0 0 3 0
0 3 2 2 2 2 2 3
0 1 2 3 0 0 0 2
0 1 2 1 2 3 0 0
0 1 2 1 2 1 2 2
0 3 2 1 2 1 2 3

First Matrix
0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 1
0 1 1 1 1 1 1 2
0 1 1 2 2 2 2 2
0 1 1 2 2 3 3 3
0 1 1 2 2 3 3 3
0 1 1 2 2 3 3 4

Longest Common Subsequence Length = 4
Longest Common Subsequences
BCDB
ACDB
```