DİNAMİK PROGRAMLAMA

Bütün parametrelerin önceden bilinmesi gerekiyor.

Binom Sabitlerinin Hesaplanması

$$\binom{n}{k} \begin{cases} 1 & k = n \quad veya \quad k = 0 \; ise \\ \binom{n-k}{k-1} + \binom{n-1}{k} & 0 < k < n \end{cases}$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$ // Çok hızlı artan fonksiyon

$$n! \sim \sqrt{2\pi} \cdot n^{(\frac{n+1}{2})} e^{-n}$$ // Stirling Formülü

$$\lim_{n \to \infty} \frac{n!}{\sqrt{2\pi} \cdot n^{(\frac{n+1}{2})} e^{-n}} = 1$$ // Alt sınırı bulmamızı sağlayacak

$$\underbrace{\frac{n!}{(n-2)!2!} \geq \frac{2^h}{\sqrt{\pi n}}}_{\text{üssel artış}}$$

En iyi durum arada 2 farkın olduğu durum. Burada en iyi durumu düşünüyoruz $\Omega(2^n)$

LCS (En uzun ortak alt katar problemi)
2 string ver ortak katar bulmaya çalış

```
LCS(A,B)
for i=0 to m
        len(i,0)
for j=0 to n
        len(0,j)=0
for i=1 to m
        for j=1 to n
                if a_i=b_j then
                        len(i,j)=len(i-1,j-1)+1
                        prev(i.j)='↖'
                end
                else if len(i-1,j)≥len(i,j-1)
                        len(i,j)=len(i-1,j)
                        prev(i,j)='↑'
                end
                else
                        len(i,j)=len(i,j-1)
                        prev(i,j)='←'
                end
                return len,prev
```

Örnek
president , providence

		0	1	2	3	4	5	6	7	8	9	10
		0	**p**	**r**	**o**	**v**	**i**	**d**	**e**	**n**	**c**	**e**
0	**0**	0	0	0	0	0	0	0	0	0	0	0
1	**p**	0	1↖	1←	1←	1←	1←	1←	1←	1←	1←	1←
2	**r**	0	1↑	2↖	2←	2←	2←	2←	2←	2←	2←	2←
3	**e**	0	1↑	2↑	2↑	2↑	2↑	2↑	3↖	3←	3←	3↖
4	**s**	0	1↑	2↑	2↑	2↑	2↑	2↑	↑	↑	↑	↑
5	**i**	0	1↑	2↑	2↑	2↑	3↖	3←	3←	3←	3←	3←
6	**d**	0	1↑	2↑	2↑	2↑	3↑	4↖	4←	4←	4←	4←
7	**e**	0	1↑	2↑	2↑	2↑	3↑	4↑	5↖	5←	5←	5←
8	**n**	0	1↑	2↑	2↑	2↑	3↑	4↑	5↑	6↖	6←	6←
9	**t**	0	1↑	2↑	2↑	2↑	3↑	4↑	5↑	6↑	6↑	6↑

Okları takip edip sadece çaprazları alıyoruz ; **priden**

Dinamik yaklaşım kullanmadan
BruteForce (Kaba Kuvvet) $O(2^n m)$

Dinamik Programlama kullanarak
O(m.n) Eşit olup olmadığını bilmediğimiz için, eşit olursa $O(n^2)$

MATRİS ZİNCİR ÇARPIMI

M1 M2 M3 matrislerinin çarpımı istense

M1 2x10 **M2** 10x2 **M3** 2x10

$$\underbrace{\underbrace{(M1 x M2)}_{40\ çarpma}\ x M3}_{40\ çarpma}$$

+----------

80 çarpma

$$M1 x \underbrace{(M2 x M3)}_{200\ çarpma}$$

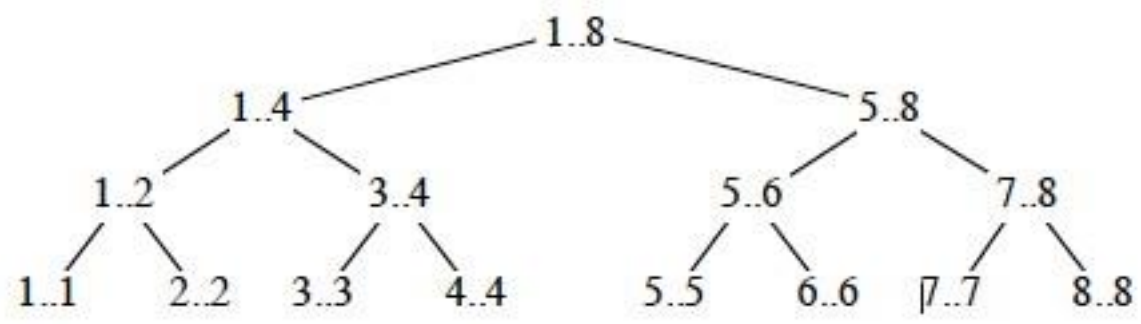
$$\underbrace{\qquad\qquad}_{200\ çarpma}$$

+----------

400 çarpma

Çarpma sırası önemli Optimizasyon problemi

2 matris 1 yol
3 matris 2 yol
4 matris ⋮



Özyineleme (Matris Zincir Çarpımı)

```
int MatrisSırasi(int p[], int i, int j)
    if (i == j) return 0;
        int k ;
        int min = INT_MAX;
        int count;
    for (k = i; k < j; k++)
    {
     count = MatrisSırası(p, i, k) + MatrisSırası(p, k+1, j) + p[i-1]* p[k]* p[j]
    if (count < min)
        min = count
    }
return min
```

Analiz (Matris Zincir Çarpımı)

n Adet matris var

$$\underbrace{(M_1 \; x \; M_2 \; x \; M_3 \cdots M_k)}_{f(k)\, yol\ var} \cdot \underbrace{(M_{k+1} \; x \; M_{k+2} \cdots M_n)}_{f(n-k)\, yol\ \text{var}}$$

$$\sum_{k=1}^{n-1} f(k) \cdot f(n-k)$$

$$\frac{1}{n}\binom{2n-2}{n-1} \approx \frac{4^n}{4\sqrt{n} \cdot n^{3/2}}$$

Alt Sınır Ω

$$\Omega\left(\frac{4^n}{\sqrt{n}}\right)$$

Üssel ifade
Özyinelemeli yaklaşım

Dinamik Yaklaşım (Matris Zincir Çarpımı)

```
int MatrisSirasi(int P[], int n)
{
    int m[n][n];              // 1 matris gelince 0 dönüyor
    for (i = 1; i < n; i++)
     m[i][i] = 0;
        for (l = 2; l < n; l++)          // l zincir uzunluğu
            for (i = 1; i < n-1; i++)
            {
             j = i + l - 1
             m[i][j] = INT_MAX
                for (k = i; k < j; k++)
                 q = m[i][k] + m[k+1][j] + p[i-1] · p[k] · p[j]
                 if (q < m[i][j])
                     m[i][j] = q
                }
            }
        }
        return m[1][n-1];
```

Analiz

$m[i][j]$

n=3 olduğunu düşünürsek

$i=1 \quad k=1 \quad j=2$

$m[1][2] = \min(m[1][1] + m[2][2] + p_0 \cdot p_1 \cdot p_2)$

$i=2 \quad k=2 \quad j=3$

$m[2][3] = \min(m[2][2] + m[3][3] + p_1 \cdot p_2 \cdot p_3)$

$1 \le \underbrace{k}_{\text{sol taraftaki sayıyı ifade ediyor}} < \underbrace{3}_{n=3}$

$1 \le k < 6$ olduğunu varsayalım

$$\min \begin{cases} m[1][1] + m[2][6] + p_0 \cdot p_1 \cdot p_6 \\ m[1][2] + m[3][6] + p_0 \cdot p_2 \cdot p_6 \\ m[1][3] + m[4][6] + p_0 \cdot p_3 \cdot p_6 \\ m[1][4] + m[5][6] + p_0 \cdot p_4 \cdot p_6 \\ m[1][5] + m[6][6] + p_0 \cdot p_5 \cdot p_6 \end{cases}$$

Döngüye dönüştürüldüğünde daha hızlı çalıştığını görüyoruz **$O(n^3)$**

Graf üzerinde
All pairs Shortes Path (Bütün eşler arası en kısa yol) Problemi

$G = (V, E)$

Ağırlıklı ve yönlü graf olması gerek
V: Düğümler
E: Kenarlar

$p = < \underbrace{V_0, V_1, V_2, \cdots V_k}_{\text{üzerinden geçmiş olduğu düğümler}} >$

Maliyeti: Bir yolu oluşturan kenarların toplamıdır

$$w(p) = \sum_{p=1}^{k} w(V_{i-1}, V_j)$$

Mesela U'dan V'ye en kısa yolun ağırlığı

$$\delta(U,V) \begin{cases} \min\{w(p)\} & U'dan\, V'ye\, yol \text{ var } ise \\ \infty & Yol\ yok\ ise \end{cases}$$

$$\delta(U,V) = w(p)$$

(eksi) - değer içermiyor

1.Yaklaşım
BruteForce / Kaba Kuvvet

Genel graflar için böyle bir şey yaparsam

$C(V^2E)$ En iyi durumda E=V olur BigO'ya göre; $O(V^3)$

2.Yaklaşım
Floyd-Warshall / Dinamik Yaklaşım

Kaynaktan hedefe gidene kadar bütün düğümler $\{1, 2, 3, \ldots k\}$

$d_{i,j}^{(k)} \Rightarrow$ i düğümünden j düğümüne ***en kısa yol***

k aradaki düğüm sayısını ifade ediyor

k=0 ise arada hiç düğüm yok

$d_{i,j}^{(k=0)} = w_{i,j}$

$w_{i,j}$ i den j ye aradaki ağırlık

$$d_{i,j}^{k} = \begin{cases} w_{i,j} & k = 0 \\ \min(d_{i,j}, d_{i,k} + d_{k,j}) & k > 0 \end{cases}$$

k>0 en az bir düğüm

$n \leftarrow rows$

$D^0 \leftarrow w$

Kaç tane satır var

rows[w]

for $k = 1$ *to* n

  for $i = 1$ *to* n

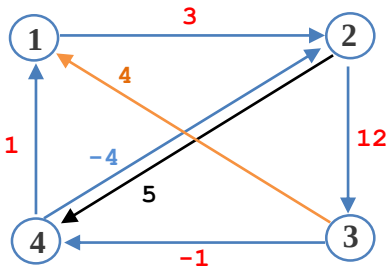
    for $j = 1$ *to* n

$$d_{i,j}^{k} \leftarrow \min(\underbrace{d_{i,j}^{k-1}, d_{i,k}^{k-1} + d_{k,j}^{k-1}}_{\text{iki boyutlu dizi gibi düşün}})$$

return D^n

$O(V^3)$ daha iyileştirilmiş

Örnek



D^0 Birbiri arasında düğüm yokken

$$D^0 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & \infty & \infty \\ \infty & 0 & 12 & 5 \\ 4 & \infty & 0 & -1 \\ 1 & -4 & \infty & 0 \end{bmatrix} \end{array}$$

D^1 En fazla bir düğüm varsa // *arada ve daha kısa ise*

$$D^1 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 15 & 8 \\ 6 & 0 & 12 & 5 \\ 0 & -5 & 0 & -1 \\ 1 & -4 & 8 & 0 \end{bmatrix} \end{array}$$

D^3 En fazla iki düğüm varsa

$$D^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 15 & 8 \\ 6 & 0 & 12 & 5 \\ 0 & -5 & 0 & -1 \\ 1 & -4 & 8 & 0 \end{bmatrix} \end{array}$$

D^3 En fazla üç düğüm varsa

$$D^3 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 15 & 8 \\ 6 & 0 & 12 & 5 \\ 0 & -5 & 0 & -1 \\ 1 & -4 & 8 & 0 \end{bmatrix} \end{array}$$

Algoritmaya göre D^4 'de bulunabilir fakat D^3 ile aynı oalcaktır.