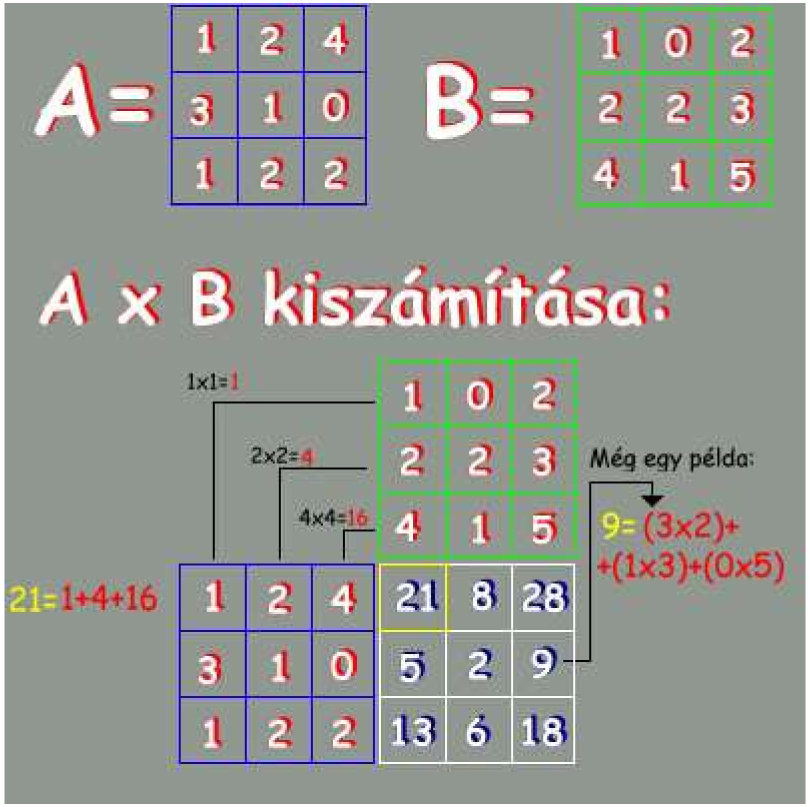
Haladó informatikai algoritmusok

Mátrixok véges sorozatainak szorzása



Dinamikus programozás jellemzői

- a feladatot részfeladatokra való osztással oldja meg
- akkor alkalmazható, ha a részproblémák nem függetlenek, azaz közös részproblémáik vannak
- minden egyes részfeladatot és annak minden részfeladatát pontosan egyszer oldja meg
- elkerüli az ismételt számítást, mivel a részfeladatok eredményeit tároljuk

Dinamikus programozás jellemzői

Egy dinamikus programozási algoritmus kifejlesztése 4 lépésre bontható fel:

1. Jellemezzük az optimális megoldás szerkezetét.
2. Rekurzív módon definiáljuk az optimális megoldás értékét.
3. Kiszámítjuk az optimális megoldás értékét alulról felfelé történő módon.
4. A kiszámított információk alapján megszerkesztünk egy optimális megoldást.

Feladat

Legyenek $A_1, A_2, \ldots, A_n$ adott mátrixok.
Számítsuk ki a mátrixok $A_1 A_2 \ldots A_n$ szorzatát.
Definíció: Mátrixok egy szorzata *teljesen zárójelezett*, ha

- vagy egyetlen mátrix,
- vagy két zárójelbe tett teljesen zárójelezett mátrixszorzat szorzata.

Megjegyzés: A mátrixok szorzása asszociatív művelet.

Példa:

(A_1, A_2, A_3, A_4) sorozat lehetséges zárójelezései:

$$(A_1(A_2(A_3 A_4)))$$
$$(A_1((A_2A_3)A_4))$$
$$((A_1A_2)(A_3A_4))$$
$$(((A_1A_2)A_3)A_4)$$

Két mátrix összeszorzásának algoritmusa

MÁTRIXSZORZÁS(A,B)

if $oszlop[A] \neq sor[B]$

  then error "nem összeillő dimenzió"

  else for $i \leftarrow 1$ **to** $sor[A]$

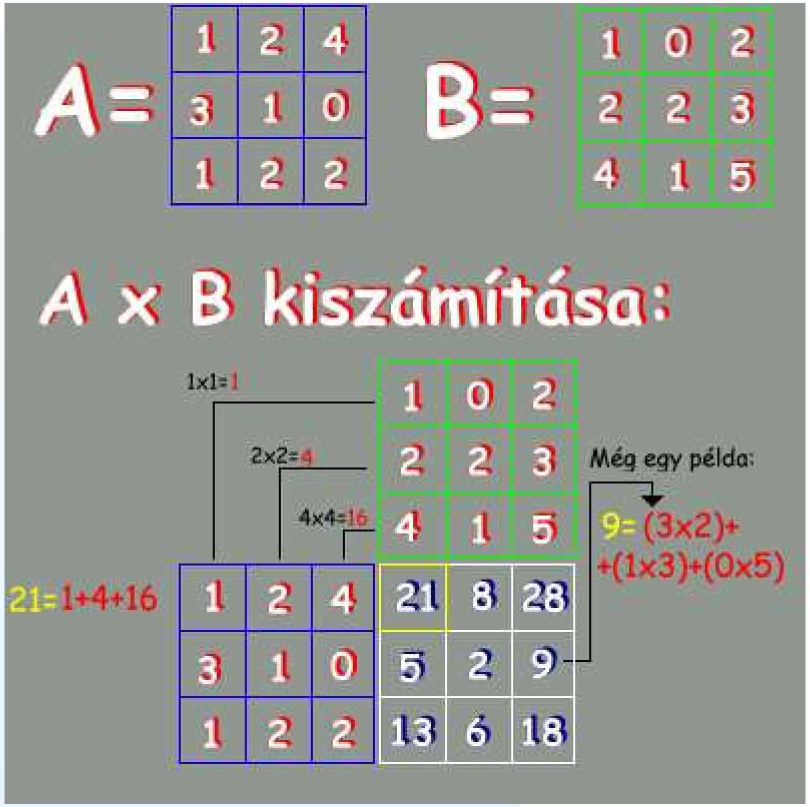
    do for $j \leftarrow 1$ **to** $oszlop[B]$

      do $C[i,j] \leftarrow 0$

        for $k \leftarrow 1$ **to** $oszlop[A]$

          do $C[i,j] \leftarrow C[i,j] + A[i,k] * B[k,j]$

    return C



Szorzatmátrix mérete:

$$A_{p\times q} * B_{q\times r} = C_{p\times r}$$

Számítási idő: pqr

Véges sok mátrix összeszorzásának problémája

Legyen $(A_1, \ldots, A_n)$ mátrixok egy adott véges sorozata, ahol az A_i mátrix mérete $p_{i-1} \times p_i$ $(i = 1, \ldots, n)$.

Keressük az $A_1 A_2 \ldots A_n$ sorozat azon **teljes zárójelezését**, mely minimalizálja a szorzat kiszámításához szükséges skalár szorzások számát.

Cél: a szorzás optimális sorrendjének megállapítása, a szorzást nem végezzük el.

Példa: A_1 10x100, A_2 100x5, A_3 5x50

költség$((A_1A_2)A_3)$=(10x100x5)+(10x5x50)=7500

költség$(A_1(A_2A_3))$=(100x5x50)+(10x100x50)=75000

Zárójelezések vizsgálata

Zárójelezések száma: Legyen $P(n)$ az n mátrix zárójelezéseinek száma.
$P(1) = 1$
Tegyük fel, hogy ha $n \geq 2$, akkor a sorozatot a k-dik és a $k + 1$-dik eleme között szétvágva a két részsorozatra a zárójelezések száma egymástól függetlenül meghatározható $(k = 1, \ldots, n - 1)$
Rekurzív formula:

$$P(n) = \begin{cases} 1, \text{ ha } n = 1, \\ \sum_{k=1}^{n-1} P(k)P(n-k), \text{ ha } n \geq 2 \end{cases}$$

A megoldás nagyságrendje: $\Omega(2^n)$
Az összes lehetséges zárójelezés megvizsgálása ("nyers erő" módszerrel) nem hatékony!

1. Az optimális zárójelezés szerkezete

Optimális megoldás a részfeladatok optimális megoldásából:
Keressük azt az alkalmas optimális részstruktúrát, mely felhasználható a probléma optimális megoldásának létrehozásához a részproblémák optimális megoldásaiból.

Az optimális zárójelezés szerkezete:
Jelölje $A_{i..j}$ az $A_i A_{i+1} \ldots A_j$ szorzat eredményét.
Ha $i < j$, akkor az optimális zárójelezés két részre vágja a sorozatot valamely A_k és A_{k+1} mátrix között $(i \leq k < j)$

Az $A_{i..j}$ mátrixot megkapjuk, ha összeszorozzuk az $A_{i..k}$ és $A_{k+1..j}$ mátrixokat;

$$koltseg(A_{i..j}) =$$
$$koltseg(A_{i..k}) + koltseg(A_{k+1..j}) + koltseg(A_{i..k}A_{k+1..j})$$

A részsorozatok zárójelezésének is optimálisnak kell lennie!

2. Rekurzív megoldás

Részprobléma: $A_i A_{i+1} \ldots A_j$ szorzat optimális zárójelezése, ahol $1 \leq i \leq j \leq n$.
Legyen $m[i,j]$ az $A_{i..j}$ mátrix számításához szükséges skalár szorzások minimális száma.
Az $A_{i..k} A_{k+1..j}$ szorzat számításának költsége a korábbiak alapján: $p_{i-1} p_k p_j$
$m[i,j]$ **számítása:**

$$m[i,j] = \begin{cases} 0, \text{ ha } i = j, \\ min_{i \leq k < j}\{m[i,k] + m[k+1,j] + p_{i-1} p_k p_j\}, \text{ ha } i < j \end{cases}$$

Megjegyzés: A k értékét ugyan nem ismerjük, de csak $j - i$ féle lehet: $k = i, i+1, \ldots, j-1$, ezekből választjuk a legjobbat.

Optimális költség számítása

A rekurzív algoritmus időigénye exponenciális, nem jobb, mint az összes zárójelezés vizsgálata, viszont a részfeladatok száma alacsony:
$\Rightarrow$ ahány i, j pár kielégíti az $1 \leq i \leq j \leq n$ feltételt:
$n(n+1)/2 = \Theta(n^2)$

(az algoritmus egy-egy részfeladattal többször is foglalkozhat a rekurziós fa különböző ágaiban)

3. Alulról felfelé történő megközelítés

A táblázat kitöltésénél fontos a sorrend:
$j - i + 1$ db mátrix $m[i, j]$ összeszorzási költségének számításához csak $j - i + 1$-nél rövidebb szorzatokat használunk:
$k = i, i + 1, \ldots, j - 1$ esetén

- $A_{i..k}$ mátrix $k - i + 1 < j - i + 1$ mátrix szorzata,
- $A_{k+1..j}$ mátrix $j - k < j - i + 1$ mátrix szorzata.

$\Rightarrow$ az m tömböt a szorzatok növekvő hossza szerint kell kitölteni.
Legyen $s[i, j]$ az a k index, melynél az optimális zárójelezés az $A_{i..j}$ szorzatot kettévágja.

Algoritmus:

MÁTRIX-SZORZÁS-SORREND(p)

```
n ← hossz[p] − 1
for i ← 1 to n
    do m[i, i] ← 0
for l ← 2 to n
    do for i ← 1 to n − l + 1
        do j ← i + l − 1
        m[i, j] ← ∞
        for k ← i to j − 1
            do q ← m[i, k] + m[k + 1, j] + p_{i−1} p_k p_j
               if q < m[i, j]
                 then m[i, j] ← q
                 s[i, j] ← k
return m és s
```

Input: $p = < p_0, p_1, \ldots, p_n >$, rendre a $p_{i-1} \times p_i$ dimenziójú A_i mátrixok méretei

Végrehajtás

- először az $m[i,i] = 0\ (i = 1, 2, \ldots, n)$ hozzárendelés végrehajtása, azaz az 1 hosszú sorozatok számítása,
- majd az $m[i,j]$ -re vonatkozó rekurzív egyenlet segítségével az $m[i, i+1]\ (i = 1, 2, \ldots, n-1)$ értékek meghatározása, azaz az $l = 2$ hosszú sorozatok számítása,
- az $m[i, i+2]\ (i = 1, 2, \ldots, n-2)$ értékek meghatározása, azaz az $l = 3$ hosszú sorozatok számítása,
- stb.

$m[i,j]$ csak a korábban már meghatározott $m[i,k]$ és $m[k+1,j]$ elemektől függ.
Futásidő: $O(n^3)$, mert l, i, k ciklusváltozók mindegyike legfeljebb n értéket vehet fel. (valójában $\Omega(n^3)$)

Az optimális megoldás előállítása

Az s tömb felhasználásával: $s[i,j]$ az a k index, amely után az $A_{i..j}$ sorozatot vágjuk.

Algoritmus (pszeudokód):

OPTIMÁLIS-ZÁRÓJELEZÉS-NYOMTATÁSA(s, i, j)

1. **if** $j = i$
2. **then print** $"A"i$
3. **else print** $"("$
4. OPTIMÁLIS-ZÁRÓJELEZÉS-NYOMTATÁSA$(s, i, s[i,j])$
5. OPTIMÁLIS-ZÁRÓJELEZÉS-NYOMTATÁSA$(s, s[i,j]+1, j)$
6. print $")"$

Kezdeti hívás:
OPTIMÁLIS-ZÁRÓJELEZÉS-NYOMTATÁSA$(s, 1, n)$

Feladat: $\underset{3\times 2}{A_1}\ \overset{2\times 1}{A_2}\ \underset{1\times 2}{A_3}$ optimális zárójelezése!

$p = \langle \underset{p_0}{3}, \underset{p_1}{2}, \underset{p_2}{1}, \underset{p_3}{2} \rangle \qquad n=3$

$m[1,1]=0 \quad m[2,2]=0 \quad m[3,3]=0 \Longleftarrow$ Az 1 hosszú ($l=1$) sorozatok számítása $\emptyset$ költségű.

$\boxed{l=2}$ hosszú sorozatok számítása

$\boxed{i=1}$

$\boxed{j}=1+2-1=\boxed{2} \qquad m[1,2]:=\infty$

$\boxed{k=1}$

$q := m[1,1]+m[2,2]+p_0p_1p_2 =$

$=0+0+3\cdot 2\cdot 1 = 6$ $\quad (A_1 \uparrow A_2)$ kiszámítása $\uparrow$ 1. után vágunk

$6 \overset{?}{<} m[1,2]$ ✓

$m[1,2]:=6 \quad (q)$

$s[1,2]:=1 \quad (k)$

$\boxed{i=2} \dashrightarrow n-l+1 = 3-2+1 = 2$

$\boxed{j}=2+2-1=\boxed{3}$ a mátrix hányadik sorának hányadik eleme

$m[2,3]:=\infty$

$\boxed{k=2}$

$q := m[2,2]+m[3,3]+p_1p_2p_3 =$

$=0+0+2\cdot 1\cdot 2 = 4$

$4 \overset{?}{<} m[2,3]$ ✓

$m[2,3]:=4$

$s[2,3]:=2$

s mátrix

$$\begin{bmatrix} & 1 & \\ & & 2 \\ & & \end{bmatrix}$$

m mátrix

$$\begin{bmatrix} 0 & \cancel{\infty}\ 6 & \infty \\ & 0 & \cancel{\infty}\ 4 \\ & & 0 \end{bmatrix}$$

$\boxed{l=3}$ hosszú sorozatok

$\boxed{i=1}$ $j = 1+3-1 = 3$

$m[1,3] := \infty$

$\boxed{k=1}$ $A_1 (A_2 A_3)$

$q := m[1,1] + m[2,3] + p_0 p_1 p_3 =$

$= 0 + 4 + 3 \cdot 2 \cdot 2 = 16$

$16 \overset{?}{<} m[1,3]$ ✓

$m[1,3] := 16$

$s[1,3] := 1$

$\boxed{k=2}$ $(A_1 A_2) A_3$

$q := m[1,2] + m[3,3] + p_0 p_2 p_3 =$

$= 6 + 0 + 3 \cdot 1 \cdot 2 = 12$

$12 \overset{?}{<} m[1,3]$ ✓

$m[1,3] := 12$

$s[1,3] := 2$

s mátrix

$$\begin{bmatrix} & 1 & \cancel{1}\ 2 \\ & & 2 \\ & & \end{bmatrix}$$

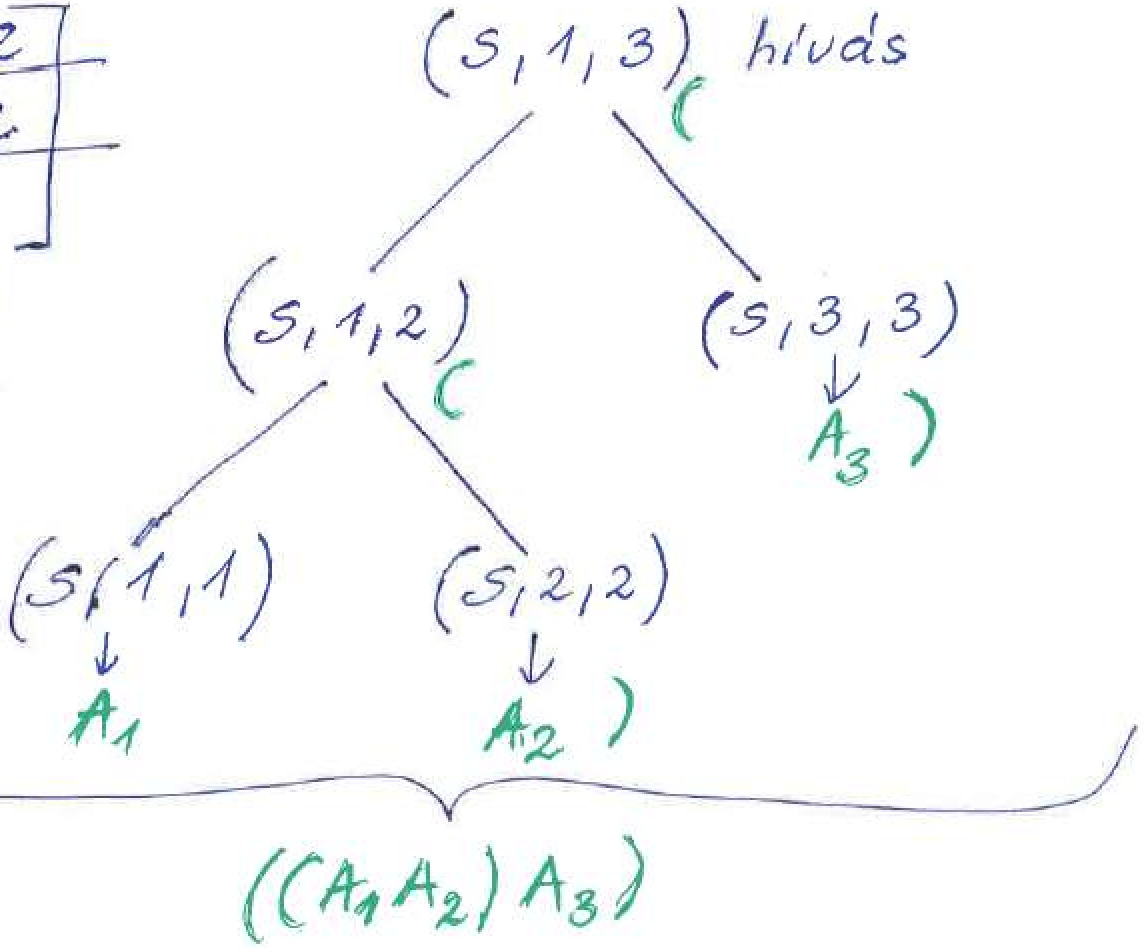
m mátrix

$$\begin{bmatrix} 0 & 6 & \cancel{16}\ \boxed{12} \\ & 0 & 4 \\ & & 0 \end{bmatrix}$$

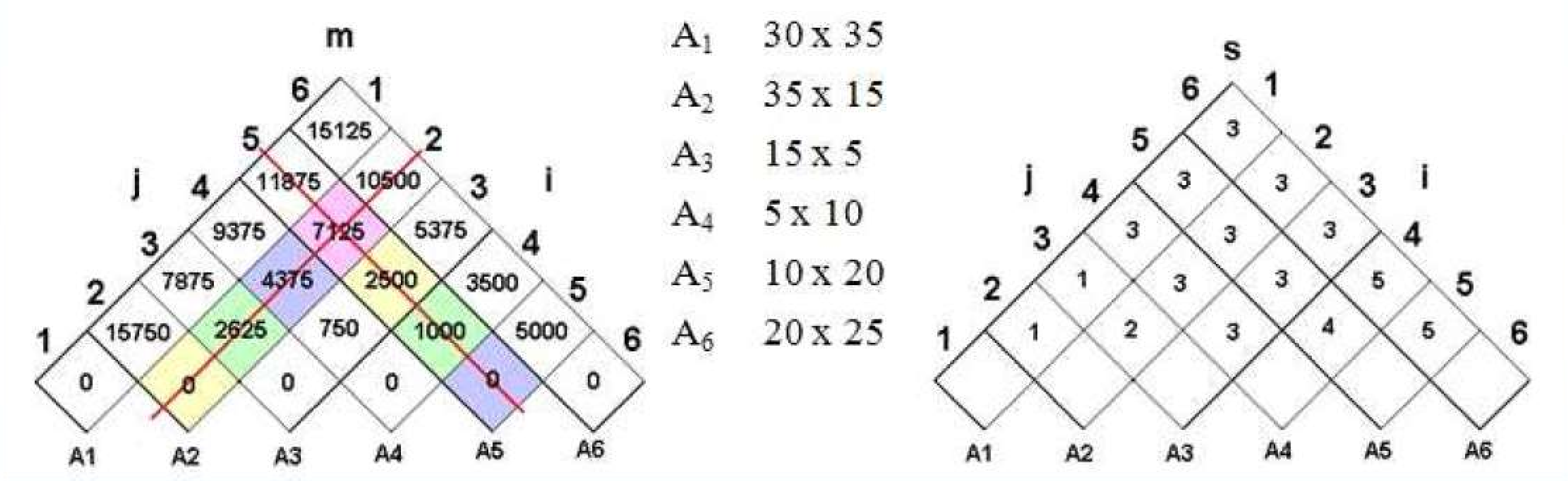
A 3 mátrix összeszorzásához minimálisan 12 skalár szorzás szükséges.

s mátrix

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & & 1 & 2 \\ 2 & & & 2 \\ 3 & & & \end{array}$$



Példa



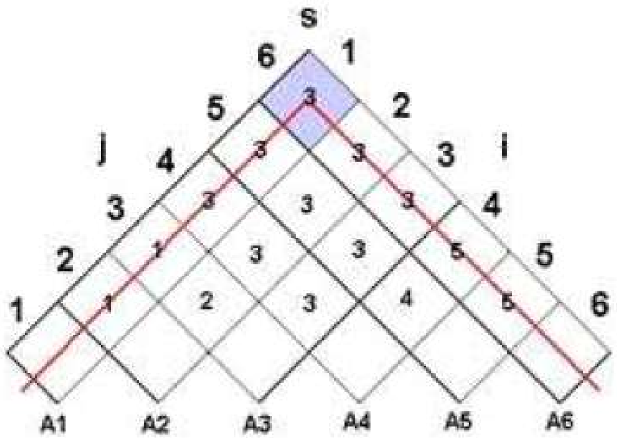
Pl.: $A_{2..5}$ részsorozat optimális összeszorzása: $p_0 = 30$, $p_1 = 35$, $p_2 = 15$, $p_3 = 5$, $p_4 = 10$, $p_5 = 20$, $p_6 = 25$

$$m[2,5] \quad = \quad min \begin{cases} m[2,2] + m[3,5] + p_1 p_2 p_5 = 0 + 2500 + 35 \times 15 \times 20 = 13000 \\ m[2,3] + m[4,5] + p_1 p_3 p_5 = 2625 + 1000 + 35 \times 5 \times 20 = \mathbf{7125} \\ m[2,4] + m[4,5] + p_1 p_4 p_5 = 4375 + 0 + 35 \times 10 \times 20 = 11375 \end{cases}$$

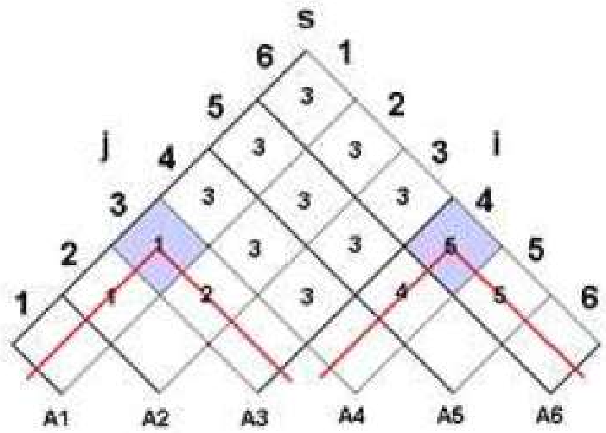
A hat mátrix összeszorzásához minimálisan $m[1,6] = 15125$ skalár szorzás kell.

A zárójelezés meghatározása rekurzívan

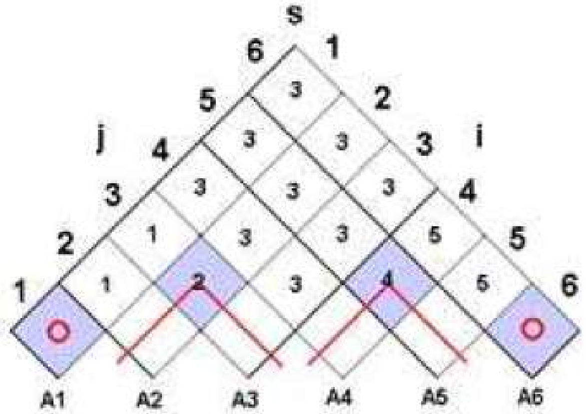
$A_{1..6} = (\, A_1\ A_2\ A_3\ A_4\ A_5\ A_6\,)$

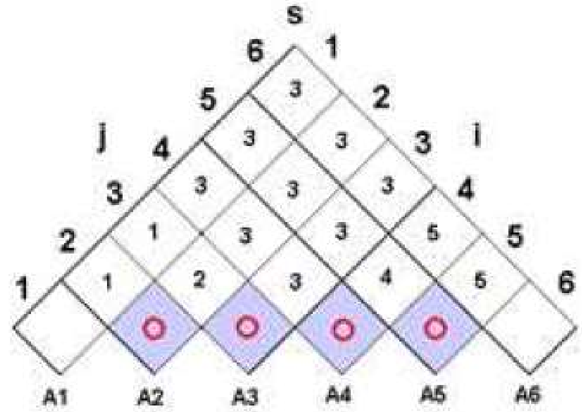


$A_{1..3}A_{4..6} = ((A_1\ A_2\ A_3)\ (A_4\ A_5\ A_6))$



$A_1A_{2..3}A_{4..5}A_6 = ((A_1\ (A_2\ A_3\ ))\ ((A_4\ A_5)\ A_6))$





$A_1A_2A_3A_4A_5A_6 = ((A_1\ (A_2\ A_3))\ ((A_4\ A_5)\ A_6)) = ((A_1\ (A_2\ A_3))\ ((A_4\ A_5)\ A_6))$

Az $A_{1..n}$ szorzat optimális kiszámítása

Az utolsó mátrixszorzás:

$$A_{1..s[1,n]}A_{s[1,n]+1..n}$$

A korábbi mátrixszorzások számítása rekurzívan:

- $A_{1..s[1,n]}$ számításakor vágás $s[1, s[1, n]]$ mögött
- $A_{s[1,n]+1..n}$ számításakor vágás $s[s[1, n] + 1, n]$ előtt

Rekurzív hívások sorrendje:

$[1, 6] \Rightarrow$
$([1, 3]([1, 1] \rightarrow A1 \rightarrow [2, 3]([2, 2] \rightarrow A2 \rightarrow [3, 3] \rightarrow A3 \rightarrow))$
$[4, 6]([4, 5]([4, 4] \rightarrow A4 \rightarrow [5, 5] \rightarrow A5 \rightarrow)[6, 6] \rightarrow A6 \rightarrow))$

Optimális zárójelezés:

$$((A1(A2A3))((A4A5)A6))$$