Mátrixszorzás

Ha egy $i \times j$ méretű mátrixot és egy $j \times k$ méretű mátrixot szorzunk össze a skalár műveletek száma ijk.
A mátrixok szorzása asszociatív, az elvégzendő skalár műveletek száma függ a zárójelezéstől.
Példa:
Legyenek A_1, A_2, A_3 méretei 2×3, 3×4, 4×5. Ekkor:
- $(A_1A_2)A_3$ 12+40=52 műveletet hajt végre
- $A_1(A_2A_3)$ pedig 60+30=90 műveletet.

A mátrixszorzás probléma feladata egy adott szorzás optimális zárójelezésének megtalálása. Az input:
$A_1, A_2, \ldots A_n$, ahol A_i mérete $p_{i-1} \times p_i$.

Dinamikus programozás I

Részprobléma: $A_i \dots A_j$ optimális zárójelezése minden i, j párra, a megoldás értéke legyen $m[i, j]$. Nyilvánvaló, hogy $m[i, i] = 0$.
Rekurzív összefüggés: Ha a szorzásnál az első zárójelpár hátsó zárójele az A_k után kerül, akkor a költség: $m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j$. Ezen lehetőségek közül választjuk a legjobbat, így ha $i < j$, akkor $m[i, j] = \min_{i \le k < j} \{m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j\}$.

Dinamikus programozás II

Táblázatkitöltés: m[i,j]-hez használjuk az m[i,k] és m[k+1,j] értékeket, ezeknek kell meglenni az m[i,j] érték számításánál. Így a helyes kitöltési sorrend átlónként megy (elsőként a j=i, értékek, aztán j=i+1, majd j=i+2 és így tovább).
A megoldás meghatározását feljegyzéses módszerrel oldjuk meg, S[i,j]-ben feljegyezzük, hogy mi volt az optimális döntés m[i,j] számításakor.

Pszeudo kód

for $i = 1$ to n
    $m[i, i] = 0$
for $l = 2$ to n
    for $i = 1$ to $n - l + 1$
        $j := i + l - 1$
        $m[i, j] := \infty$
        for $k = i$ to $j - 1$
            $q := m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j$
            If $q < m[i, j]$
                Then $m[i, j] := q$
                $S[i, j] := k$

Megoldás meghatározása

A fenti a szakasz kitölti az m és S táblázatokat, a kiíratás S alapján egy rekurzív algoritmussal megtehető.

KIIR(i,j)

```
If j>i
Then Print "("
     KIIR(i,S[i,j])
     KIIR(S[i,j]+1,j)
     Print ")"
     Else Print "A_i"
```

Leghosszabb közös részsorozat

Leghosszabb közös részsorozat (LKR)

Egy sorozat, akkor részsorozata egy másiknak, ha abból elemeinek elhagyásával megkapható. A feladat két sorozat $X = (x_1, \ldots, x_m)$ és $Y = (y_1, \ldots, y_n)$ leghosszabb közös részsorozatának meghatározása.
A továbbiakban X_i az X sorozat i hosszú prefixét jelöli $X_i = (x_1, \ldots, x_i)$ és hasonlóan jelöljük a prefixeket az Y és Z sorozatokra is.

Leghosszabb közös részsorozat

Lemma: Legyen $X = (x_1, \ldots, x_m)$ és $Y = (y_1, \ldots, y_n)$ két sorozat és $Z = (z_1, \ldots, z_k)$ ezek LKR-je. Ekkor:
- Ha $x_m = y_n$, akkor $z_k = x_m = y_n$ és Z_{k-1} az X_{m-1} és Y_{n-1} sorozatok egy LKR-je.
- Ha $x_m \neq y_n$, akkor Z az X_{m-1} és Y vagy az X és Y_{n-1} sorozatok egy LKR-je.

Dinamikus programozás

Megoldás dinamikus programozással:
Részprobléma: X_i és Y_j LKR-je. Az LKR hossza legyen c[i,j].
Nyilvánvalóan c[0,j]=c[i, 0]= 0.
Rekurzív összefüggés: A lemma alapján

$$c[i,j] = \begin{cases} 0, & \text{ha } i = 0 \text{ vagy } j = 0, \\ c[i-1, j-1] + 1, & \text{ha } x_i = y_j, \\ \max\{c[i-1,j], c[i,j-1]\} & \text{egyébként,} \end{cases}$$

Dinamikus programozás

Táblázatkitöltés: c[i,j]-hez használjuk a c[i,j-1] és c[i-1,j] értékeket, ezeknek kell meglenni a c[i,j] érték számításánál. Így a helyes kitöltési sorrend soronként minden sorban a nagyobb j érték felé.
A megoldás meghatározását feljegyzéses módszerrel oldjuk meg, S[i,j]-ben feljegyezzük, hogy mi volt az optimális döntés c[i,j] számításakor.

Pszeudo kód

Pszeudokód:

for i=0 to m
    $c[i, 0] = 0$
for j=1 to n
    $c[0, j] = 0$
for i=1 to m
    for j=1 to n
        if $x_i = y_j$
            then $c[i, j] := c[i - 1, j - 1] + 1$
            $S[i, j] := 2$
        else if $c[i - 1, j] \geq c[i, j - 1]$
            then $c[i, j] := c[i - 1, j]$
            $S[i, j] := 1$
        else $c[i, j] := c[i, j - 1]$
            $S[i, j] := 0$

Megoldás meghatározása

Ez a szakasz kitölti a c és S táblázatokat, a kiíratás S alapján egy rekurzív algoritmussal megtehető.

```
KIIR(i,j)
If i=0 or j=0
    Then return
If S[i,j]=2
    Then KIIR(i-1,j-1)
    Print "x_i"
Else if S[i,j]=1
    Then KIIR(i-1,j)
Else KIIR(i,j-1)
```