Fontes principais

1. Cormem T. H.; Leiserson C. E.; Rivest R.: Stein C.. *Introduction to Algorithms*, 3^a edição, MIT Press, 2009
2. Análise de algoritmo - IME/USP (prof. Paulo Feofiloff) `http://www.ime.usp.br/~pf/analise_de_algoritmos`

Programação dinâmica

Programação dinâmica

Quando se aplica:

- ▷ Problemas de otimização
- ▷ Solução consiste em uma sequência de decisões

Programação dinâmica

Características:

▷ **Existência de subestrutura ótima**: as soluções ótimas do problema contém soluções ótimas dos subproblemas

▷ **Fórmula de recorrência**: descreve a relação entre as soluções ótimas dos subproblemas.

▷ **Sobreposição de subproblemas**: a solução ótima de um subproblema é usada várias vezes mas **calculada uma vez**!

Programação dinâmica

Características:

▷ **Soluções de subproblemas são armazenadas em tabelas**, logo é preciso que um número total de subproblemas que precisam ser resolvidos seja pequeno (polinomial no tamanho da entrada)

▷ **Estratégia bottom-up** (diferente de divisão e conquista que é top down) e possui muitas sequências de decisões.

▷ "Não existe"prova de corretude

Cálculo fatorial

Cálculo fatorial

solução recursiva natural

fatorial(n)

se $n = 0$

  então retorne 1

  senão

    retorne $n * \text{fatorial}(n - 1)$

Como é uma solução com programação dinâmica?

Cálculo fatorial (PD)

fatorial(n)

se $n = 0$
    então retorne 1
    senão
        se $(fat[n]! = 0)$ $\triangleright$ caso já calculado
            então retorne $fat[n]$
            senão
                retorne $fat[n] = n *$ fatorial$(n-1)$

Multiplicação de matrizes

Multiplicação de matrizes

Problema: Multiplicar n matrizes realizando o número mínimo de operações.

$$M = M_1 \times M_2 \times \ldots M_n$$

Multiplicação de matrizes

Exemplo: $(A_1)_{10\times 100}$, $(A_2)_{100\times 5}$ e $(A_3)_{5\times 50}$

Quantas multiplicações são realizadas nas sequencias abaixo?

$$
\begin{aligned}
A &= (A_1 \times A_2) \times A_3 \\
A &= A_1 \times (A_2 \times A_3)
\end{aligned}
$$

Multiplicação de matrizes

Exemplo: $(A_1)_{10\times100}$, $(A_2)_{100\times5}$ e $(A_3)_{5\times50}$

Número de multiplicações:

$$\begin{aligned} A &= (A_1 \times A_2) \times A_3 \\ &= (10 \times 100 \times 5) + (10 \times 5 \times 50) \\ &= 7500 \text{ operações} \end{aligned}$$

$$\begin{aligned} A &= A_1 \times (A_2 \times A_3) \\ &= (100 \times 5 \times 50) + (10 \times 100 \times 50) \\ &= 75000 \text{ operações} \end{aligned}$$

Multiplicação de matrizes

Exemplo: $(A_1)_{10\times 100}$, $(A_2)_{100\times 5}$ e $(A_3)_{5\times 50}$

Número de multiplicações:

$$\begin{aligned} A &= (A_1 \times A_2) \times A_3 \\ &= 7500 \text{ operações} \end{aligned}$$

$$\begin{aligned} A &= A_1 \times (A_2 \times A_3) \\ &= 75000 \text{ operações} \end{aligned}$$

Conclusão: A ordem das multiplicações faz **muita** diferença

Multiplicação de matrizes

Como determinar a sequência ótima de multiplicação?

Multiplicação de matrizes

Como determinar a sequência ótima de multiplicação?

▷ Algoritmo força bruta é "impraticável": existem $\Omega(2^n)$ possibilidades (número de Catalão)

Aplicando programação dinâmica:

▷ É possível determinar uma sequência ótima eficiente (polinomial)

Multiplicação de matrizes

Para resolver este problema, tudo que precisamos é saber qual o melhor índice k tal que

$$A = (A_1 \times A_2 \times \cdots A_k) \cdot (A_{k+1} \times A_{k+2} \times \cdots A_n)$$

onde k varia de 1 a $n - 1$.

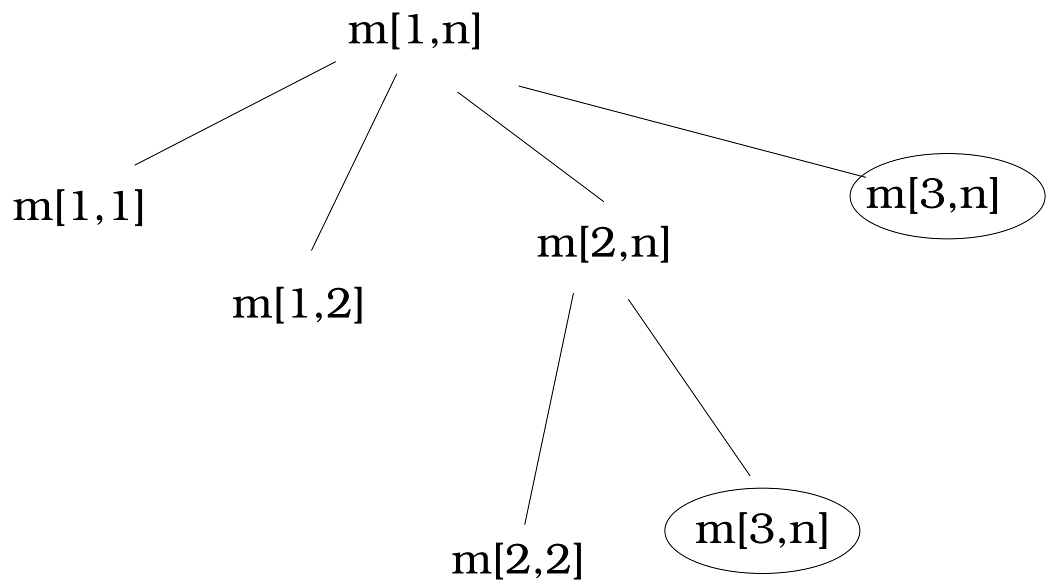
Multiplicação de matrizes

Denotemos por $m[i,j]$ o número mínimo de multiplicações necessárias para multiplicar $A_i \cdots A_j$, $i \leq j$. Temos que:

$$m[i,j] = \begin{cases} 0 & \text{, se } i = j \\ min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\} & \text{, se } i < j \end{cases}$$

onde $p_0, p_1, \cdots p_n$ determinam as dimensões das matrizes: cada A_i tem dimensão $p_{i-1} \times p_i$.

Multiplicação de matrizes



O algoritmo recursivo para calcular $m[i, j]$ por esta fórmula seria excessivamente lenta (exponencial)

Multiplicação de matrizes

Com a programação dinâmica é possível construir uma tabela para armazenar os valores de $m[i,j]$ de forma *bottom-up*, usando memoização.

1. $m[i,i] = 0$, para $1 \leq i \leq n$.
2. $m[i,i+1]$ são calculados, para $1 \leq i \leq n-1$.
3. $m[i,i+2]$ são calculados, para $1 \leq i \leq n-2$.
4. e assim por diante

Multiplicação de matrizes

Exemplo: $A = A_1 \times A_2 \times A_3 \times A_4$ com $p = \{200, 2, 30, 20, 5\}$

$$A \quad = A_1 \quad \times A_2 \quad \times A_3 \quad \times A_4$$

Multiplicação de matrizes

Exemplo: $A = A_1 \times A_2 \times A_3 \times A_4$ com $p = \{200, 2, 30, 20, 5\}$

$$\begin{array}{ll} A & = A_1 \quad \times A_2 \quad \times A_3 \quad\quad \times A_4 \\ & (200 \times 2)(2 \times 30)(30 \times 20)(20 \times 5) \end{array}$$

Multiplicação de matrizes

Exemplo: $A = A_1 \times A_2 \times A_3 \times A_4$ com $p = \{200, 2, 30, 20, 5\}$

$$
\begin{array}{ll}
A & = A_1 \quad \times A_2 \quad \times A_3 \quad\quad \times A_4 \\
& (200 \times 2)(2 \times 30)(30 \times 20)(20 \times 5) \\
& (p_0 \times p_1)(p_1 \times p_2)(p_2 \times p_3)(p_3 \times p_4)
\end{array}
$$

Multiplicação de matrizes

Exemplo: $A = A_1 \times A_2 \times A_3 \times A_4$ com $p = \{200, 2, 30, 20, 5\}$

$$\begin{array}{lllll} A & = A_1 & \times A_2 & \times A_3 & \times A_4 \\ & (200 \times 2) & (2 \times 30) & (30 \times 20) & (20 \times 5) \\ & (p_0 \times p_1) & (p_1 \times p_2) & (p_2 \times p_3) & (p_3 \times p_4) \end{array}$$

$m[1,1] = m[2,2] = m[3,3] = m[4,4] = 0$

Multiplicação de matrizes

Exemplo: $A = A_1 \times A_2 \times A_3 \times A_4$ com $p = \{200, 2, 30, 20, 5\}$

$$\begin{array}{ll} A & = A_1 \quad\quad \times A_2 \quad\quad \times A_3 \quad\quad\quad \times A_4 \\ & (200 \times 2)(2 \times 30)(30 \times 20)(20 \times 5) \\ & (p_0 \times p_1)(p_1 \times p_2)(p_2 \times p_3)(p_3 \times p_4) \end{array}$$

$m[1,1] = m[2,2] = m[3,3] = m[4,4] = 0$

$m[1,2] = 12000, \quad m[2,3] = 1200, \quad m[3,4] = 3000$

Multiplicação de matrizes

$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}$, para $i \cdots j-1$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,3] = min\{m[1,k] + m[k+1,3] + p_0p_kp_3\}, k = 1,2$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,3] = min\{m[1,k] + m[k+1,3] + p_0p_kp_3\}, k = 1,2$$

$$m[1,3] \quad = min\{m[1,1] + m[2,3] + p_0p_1p_3, m[1,2] + m[3,3] + p_0p_2p_3\}$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,3] = min\{m[1,k] + m[k+1,3] + p_0p_kp_3\}, k = 1,2$$

$$\begin{aligned} m[1,3] &= min\{m[1,1] + m[2,3] + p_0p_1p_3, m[1,2] + m[3,3] + p_0p_2p_3\} \\ &= min\{0 + 1200 + 8000, 12000 + 0 + 120000\} \end{aligned}$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,3] = min\{m[1,k] + m[k+1,3] + p_0p_kp_3\}, k = 1,2$$

$$\begin{aligned} m[1,3] &= min\{m[1,1] + m[2,3] + p_0p_1p_3, m[1,2] + m[3,3] + p_0p_2p_3\} \\ &= min\{0 + 1200 + 8000, 12000 + 0 + 120000\} \\ &= 9200 \quad (A_1 \times (A_2 \times A_3)) \end{aligned}$$

Multiplicação de matrizes

$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}$, para $i \cdots j-1$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[2,4] = min\{m[2,k] + m[k+1,4] + p_1p_kp_4\}, k = 2,3$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[2,4] = min\{m[2,k] + m[k+1,4] + p_1p_kp_4\}, k = 2,3$$

$$\begin{aligned} m[2,4] &= min\{m[2,2] + m[3,4] + p_1p_2p_4, m[2,3] + m[4,4] + p_1p_3p_4\} \\ &= min\{0 + 3000 + 300, 1200 + 0 + 200\} \\ &= 1400 \quad ((A_2 \times A_3) \times A_4) \end{aligned}$$

Multiplicação de matrizes

$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}$, para $i \cdots j-1$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,4] = min\{m[1,k] + m[k+1,4] + p_0p_kp_4\}, k = 1,2,3$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j\}, k = i \cdots j-1$$

$$m[1,4] = min\{m[1,k] + m[k+1,4] + p_0p_kp_4\}, k = 1,2,3$$

$$\begin{aligned} m[1,4] &= min\{m[1,1] + m[2,4] + p_0p_1p_4, \\ &\qquad m[1,2] + m[3,4] + p_0p_2p_4, \\ &\qquad m[1,3] + m[4,4] + p_0p_3p_4\} \\ &= min\{0 + 1400 + 2000, \\ &\qquad 12000 + 3000 + 30000, \\ &\qquad 9200 + 0 + 20000\} \\ &= 3400 \quad (A_1 \times ((A_2 \times A_3) \times A_4)) \end{aligned}$$

Multiplicação de matrizes

$$m[i,j] = min\{m[i,k] + m[k+1,j] + p_{i-1}p_k p_j\}, k = i \cdots j-1$$

$$m[1,4] = min\{m[1,k] + m[k+1,4] + p_0 p_k p_4\}, k = 1,2,3$$

$$\begin{aligned}
m[1,4] &= min\{m[1,1] + m[2,4] + p_0p_1p_4, \\
&\qquad m[1,2] + m[3,4] + p_0p_2p_4, \\
&\qquad m[1,3] + m[4,4] + p_0p_3p_4\} \\
&= min\{0 + 1400 + 2000, \\
&\qquad 12000 + 3000 + 30000, \\
&\qquad 9200 + 0 + 20000\} \\
&= 3400 \quad (A_1 \times ((A_2 \times A_3) \times A_4))
\end{aligned}$$

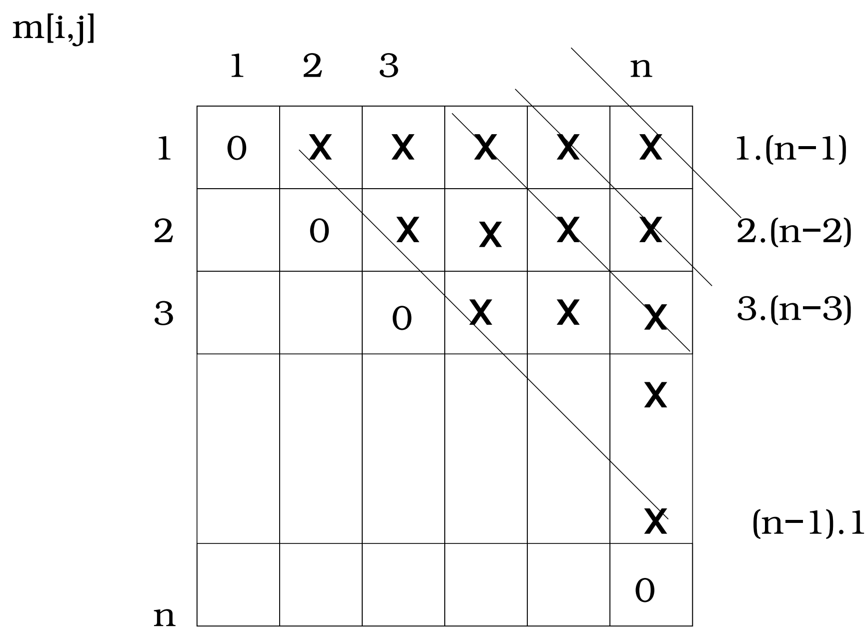
Solução ótima: $(A_1 \times ((A_2 \times A_3) \times A_4))$

Exercícios

1) Aplique o algoritmo para multiplicar 5 matrizes, onde $p = \{30, 35, 15, 5, 10, 20\}$

2) Apresente uma solução para a variante do problema de multiplicação de matrizes em que o objetivo é maximizar o número de operações.

Multiplicação de matrizes



$$\sum_{i=1}^{n-1} i \cdot (n-i) = O(n^3)$$

Multiplicação de matrizes

algoritmo: ordem-cadeia-matriz()

entrada: um vetor $p[0..n]$ das dimensões das matrizes, e n o número de matrizes

saída: $m[1, n]$ (solução ótima)

Multiplicação de matrizes

ordem-cadeia-matriz(p, n)

para $i = 1$ **até** n **faça** $m[i,i] = 0$
para $l = 2$ **até** n **faça**
    para $i = 1$ **até** $n - l + 1$ **faça**
        $j = i + l - 1$
        $m[i,j] = \infty$
        para $k = i$ **até** $j - 1$ **faça**
            $q = m[i,k] + m[k+1,j] + p[i-1]p[k]p[j]$
            se $q < m[i,j]$ **então**
                $m[i,j] = q$
                $s[i,j] = k$ $\triangleright$ usado para recuperar a
                $\triangleright$ parentização ótima

Multiplicação de matrizes

imprime-parenteses-otimo(A, s, i, j)

se $i = j$
    então imprime A_i
    senão
        imprime "("
        imprime-parenteses-otimo$(A, s, i, s[i, j])$
        imprime-parenteses-otimo$(A, s, s[i, j] + 1, j)$
        imprime ")"

Multiplicação de matrizes

calcula-produto(A, s, i, j)

se $i = j$
  então retorne A_i
  senão
    $X =$ calcula-produto$(A, s, i, s[i, j])$
    $Y =$ calcula-produto$(A, s, s[i, j] + 1, j)$
    retorne multiplica(X,Y)

Subsequência Comum Máxima (SCM)

Subsequência Comum Máxima (SCM)

Problema: Dadas duas sequências X e Y determinar a maior subsequência comum de X e Y.

Exemplo: X = ATCTGAT e Y = TGCATTA

Subsequência Comum Máxima (SCM)

Problema: Dadas duas sequências X e Y determinar a maior subsequência comum de X e Y.

Exemplo: X = ATCTGAT e Y = TGCATTA

SCM(X,Y) = TCTA

Subsequência Comum Máxima (SCM)

Problema: Dadas duas sequências X e Y determinar a maior subsequência comum de X e Y.

Exemplo: X = ATCTGAT e Y = TGCATTA

SCM(X,Y) = TCTA

Visualmente temos:

```
AT_C_TGAT
 | | | |
_TGCAT_A_
```

Subsequência Comum Máxima (SCM)

Seja $c[i,j]$ o comprimento da sequencia comum máxima dos prefixos X_i e Y_j. Se $|X| = m$ e $|Y| = n$, então $c[m,n]$ é o valor **ótimo**.

Subestrutura ótima

Seja $Z = z_1 \cdots z_k$ a subsequência comum máxima de $X = x_1 \cdots x_m$ e $Y = y_1 \cdots y_n$, temos:

- ▷ Se $x_m = y_n$, então $z_k = x_m = y_n$ e $Z_{k-1} = SCM(X_{m-1}, Y_{n-1})$
- ▷ Se $x_m \neq y_n$, então $z_k \neq x_m$ e $Z = SCM(X_{m-1}, Y)$
- ▷ Se $x_m \neq y_n$, então $z_k \neq y_n$ e $Z = SCM(X, Y_{n-1})$

Fórmula de recorrência:

$$c[i,j] = \begin{cases} 0 & \text{, se } i = 0 \text{ ou } j = 0 \\ c[i-1, j-1] + 1 & \text{, se } i,j > 0 \text{ e } x_i = y_j \\ max\{c[i-1, j], c[i, j-1]\} & \text{, se } i,j > 0 \text{ e } x_i \neq y_j \end{cases}$$

SCM(X, Y, m, n, c, b)

para $i = 0$ **até** m **faça** $c[i, 0] = 0$
para $j = 0$ **até** n **faça** $c[0, j] = 0$
para $i = 1$ **até** m **faça**
    para $j = 0$ **até** n **faça**
        se $x_i == y_i$ **então**
            $c[i, j] = c[i-1, j-1] + 1$
            $b[i, j] = '' \nwarrow ''$
        senão
            se $c[i, j-1] > c[i-1, j]$ **então**
                $c[i, j] = c[i, j-1]$
                $b[i, j] = '' \leftarrow ''$
            senão
                $c[i, j] = c[i-1, j]$
                $b[i, j] = '' \uparrow ''$
retorne $(c[m, n], b)$

Subsequência Comum Máxima (SCM)

Ex.: $X = ABCBDAB$ e $Y = BDCABA$, $m = 8$, $n = 7$

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B	0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D	0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A	0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B	0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

Subsequência Comum Máxima (SCM)

Note que:

▷ A complexidade do algoritmo é $O(mn)$

▷ O algoritmo não encontra a subsequência comum de comprimento máximo, ele determina qual é o valor do comprimento máximo.

▷ Com a matriz b preenchida é fácil encontrar a subsequência comum máxima.

Para recuperar a SCM, temos:

```
recupera-SCM(b, X, Y, i, j)
se i = j então retorne
se b[i, j] = "↖" então
    recupera-SCM(b, X, Y, i − 1, j − 1)
    imprime(X_i)
senão
    se b[i, j] = "↑" então
        recupera-SCM(b, X, Y, i − 1, j)
    senão
        recupera-SCM(b, X, Y, i, j − 1)
```

Subsequência Comum Máxima (SCM)

Perceba que:

$\triangleright$ Dado b é possível determinar a subsequência comum máxima em tempo $O(m+n)$ no pior caso.
$\triangleright$ A complexidade de tempo e de espaço do algoritmo de programação dinâmica para o problema de subsequência comum máxima é $O(mn)$
$\triangleright$ A matriz b é dispensável, pois podemos economizar memória recuperando a solução a partir da matriz c.

Subsequência Comum Máxima (SCM)

Caso não exista interesse em determinar a subsequência comum máxima, mas apenas seu comprimento, é possível reduzi o gasto de memória para $O(min\{m, n\})$, basta registrar apenas a linha da matriz sendo preenchida e a anterior.

Obrigado