

Programación dinámica: un último ejemplo

Dado: matrices $A_{m \times n}$, $B_{n \times r}$, $C_{r \times s}$ de números enteros

Para calcular $A_{m \times n} \cdot B_{n \times r}$ el algoritmo usual realiza $m \cdot n \cdot r$ multiplicaciones de números enteros

Programación dinámica: un último ejemplo

Dado: matrices $A_{m \times n}$, $B_{n \times r}$, $C_{r \times s}$ de números enteros

Para calcular $A_{m \times n} \cdot B_{n \times r}$ el algoritmo usual realiza $m \cdot n \cdot r$ multiplicaciones de números enteros

¿Cuántas multiplicaciones de enteros realiza el algoritmo usual para calcular $A \cdot B \cdot C$?

Programación dinámica: un último ejemplo

Dado: matrices $A_{m \times n}$, $B_{n \times r}$, $C_{r \times s}$ de números enteros

Para calcular $A_{m \times n} \cdot B_{n \times r}$ el algoritmo usual realiza $m \cdot n \cdot r$ multiplicaciones de números enteros

¿Cuántas multiplicaciones de enteros realiza el algoritmo usual para calcular $A \cdot B \cdot C$?

Como la multiplicación de matrices es asociativa esto depende de si calculamos $(A \cdot B) \cdot C$ or $A \cdot (B \cdot C)$

- Calcular $(A \cdot B) \cdot C$ requiere de $m \cdot n \cdot r + m \cdot r \cdot s$ multiplicaciones de enteros, mientras que calcular $A \cdot (B \cdot C)$ requiere de $m \cdot n \cdot s + n \cdot r \cdot s$ multiplicaciones de números enteros

Programación dinámica: un último ejemplo

Los números $m \cdot n \cdot r + m \cdot r \cdot s$ y $m \cdot n \cdot s + n \cdot r \cdot s$ pueden ser distintos.

- ▶ En particular dependiendo de los valores de m y s

Programación dinámica: un último ejemplo

Los números $m \cdot n \cdot r + m \cdot r \cdot s$ y $m \cdot n \cdot s + n \cdot r \cdot s$ pueden ser distintos.

- En particular dependiendo de los valores de m y s

Ejemplo

Sea $m = 1000$ y $n = r = s = 2$ tenemos que

$$m \cdot n \cdot r + m \cdot r \cdot s = 8000$$

$$m \cdot n \cdot s + n \cdot r \cdot s = 4008$$

En este caso nos conviene calcular $A \cdot (B \cdot C)$

Programación dinámica: un último ejemplo

Problema a resolver: Dada una secuencia de matrices $A_1, A_2, \dots, A_\ell$ ($\ell \geq 1$), determinar el número mínimo de multiplicaciones de números enteros que debe realizar el algoritmo usual para calcular $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$

Programación dinámica: un último ejemplo

Problema a resolver: Dada una secuencia de matrices $A_1, A_2, \dots, A_\ell$ ($\ell \geq 1$), determinar el número mínimo de multiplicaciones de números enteros que debe realizar el algoritmo usual para calcular $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$

- ▶ Se debe decidir como colocar paréntesis en la expresión $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$ al utilizar la asociatividad de la multiplicación de matrices
- ▶ Suponemos que las matrices tienen los ordenes adecuados para que $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$ sea calculable

Programación dinámica: un último ejemplo

Problema a resolver: Dada una secuencia de matrices $A_1, A_2, \dots, A_\ell$ ($\ell \geq 1$), determinar el número mínimo de multiplicaciones de números enteros que debe realizar el algoritmo usual para calcular $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$

- ▶ Se debe decidir como colocar paréntesis en la expresión $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$ al utilizar la asociatividad de la multiplicación de matrices
- ▶ Suponemos que las matrices tienen los ordenes adecuados para que $A_1 \cdot A_2 \cdot \dots \cdot A_\ell$ sea calculable

Vamos a resolver este problema usando programación dinámica.

Calculando la multiplicación de una secuencia de matrices

Dada una matriz A , usamos $\text{filas}(A)$ y $\text{col}(A)$ para denotar el número de filas y columnas de A

Calculando la multiplicación de una secuencia de matrices

Dada una matriz A , usamos $\text{filas}(A)$ y $\text{col}(A)$ para denotar el número de filas y columnas de A

Considere una lista de matrices $[A_1, \dots, A_\ell]$ tales que $\ell \geq 1$ y $\text{col}(A_i) = \text{filas}(A_{i+1})$ para $i \in \{1, \dots, \ell - 1\}$

Calculando la multiplicación de una secuencia de matrices

Dada una matriz A , usamos $\text{filas}(A)$ y $\text{col}(A)$ para denotar el número de filas y columnas de A

Considere una lista de matrices $[A_1, \dots, A_\ell]$ tales que $\ell \geq 1$ y $\text{col}(A_i) = \text{filas}(A_{i+1})$ para $i \in \{1, \dots, \ell - 1\}$

Queremos definir una función **NumMult** $([A_1, \dots, A_\ell])$ que retorna el número mínimo de multiplicaciones que debe realizar el algoritmo usual para calcular $A_1 \cdot \dots \cdot A_\ell$

Calculando la multiplicación de una secuencia de matrices

Dada una matriz A , usamos $\text{filas}(A)$ y $\text{col}(A)$ para denotar el número de filas y columnas de A

Considere una lista de matrices $[A_1, \dots, A_\ell]$ tales que $\ell \geq 1$ y $\text{col}(A_i) = \text{filas}(A_{i+1})$ para $i \in \{1, \dots, \ell - 1\}$

Queremos definir una función **NumMult** $([A_1, \dots, A_\ell])$ que retorna el número mínimo de multiplicaciones que debe realizar el algoritmo usual para calcular $A_1 \cdot \dots \cdot A_\ell$

¿Puede ser **NumMult** implementada utilizando programación dinámica?

¿Podemos utilizar programación dinámica?

¿Satisface el problema el principio de optimalidad de sub-problemas?

¿Podemos utilizar programación dinámica?

¿Satisface el problema el principio de optimalidad de sub-problemas?

► Sí

¿Podemos utilizar programación dinámica?

¿Satisface el problema el principio de optimalidad de sub-problemas?

► Sí

Dado que se satisface el principio de optimalidad de sub-problemas obtenemos la siguiente definición recursiva de **NumMult**:

$$\text{NumMult}([A_1, \dots, A_\ell]) = \begin{cases} 0 & \ell = 1 \\ \min_{1 \leq i \leq \ell-1} \{ \text{NumMult}([A_1, \dots, A_i]) + \\ \quad \text{NumMult}([A_{i+1}, \dots, A_\ell]) + \\ \quad \text{filas}(A_1) \cdot \text{col}(A_i) \cdot \text{col}(A_\ell) \} & \ell > 1 \end{cases}$$

¿Podemos utilizar programación dinámica?

¿Se satisface el requisito de tener un número pequeño (polinomial) de sub-problemas?

¿Podemos utilizar programación dinámica?

¿Se satisface el requisito de tener un número pequeño (polinomial) de sub-problemas?

- ▶ Sí, las llamadas posibles son de la forma **NumMult**($[A_i, \dots, A_j]$) con $1 \leq i \leq j \leq \ell$

¿Podemos utilizar programación dinámica?

¿Se satisface el requisito de tener un número pequeño (polinomial) de sub-problemas?

- ▶ Sí, las llamadas posibles son de la forma **NumMult**($[A_i, \dots, A_j]$) con $1 \leq i \leq j \leq \ell$
- ▶ Tenemos entonces $\frac{\ell \cdot (\ell + 1)}{2}$ llamadas a la función **NumMult**

¿Podemos utilizar programación dinámica?

¿Se satisface el requisito de tener un número pequeño (polinomial) de sub-problemas?

- ▶ Sí, las llamadas posibles son de la forma **NumMult**($[A_i, \dots, A_j]$) con $1 \leq i \leq j \leq \ell$
- ▶ Tenemos entonces $\frac{\ell \cdot (\ell + 1)}{2}$ llamadas a la función **NumMult**

Una última pregunta: ¿cómo calculamos **NumMult**($[A_1, \dots, A_\ell]$) de manera eficiente?

Un enfoque bottom-up para la evaluación

Ejercicio

Construya un algoritmo para calcular una tabla que contiene los valores **NumMult** $([A_i, \dots, A_j])$ para $1 \leq i \leq j \leq \ell$

- ▶ Para analizar la complejidad del algoritmo considere como operaciones básicas acceder a una celda de la tabla (para leer o escribir), acceder a $\text{filas}(A_i)$ y $\text{col}(A_i)$, y sumar, multiplicar y comparar números enteros
- ▶ Muestre entonces que el algoritmo para construir la tabla en el peor caso es $O(\ell^3)$

Un enfoque bottom-up para la evaluación

Ejercicio

Construya un algoritmo para calcular una tabla que contiene los valores **NumMult** $([A_i, \dots, A_j])$ para $1 \leq i \leq j \leq \ell$

- ▶ Para analizar la complejidad del algoritmo considere como operaciones básicas acceder a una celda de la tabla (para leer o escribir), acceder a $\text{filas}(A_i)$ y $\text{col}(A_i)$, y sumar, multiplicar y comparar números enteros
- ▶ Muestre entonces que el algoritmo para construir la tabla en el peor caso es $O(\ell^3)$

Corolario

*Utilizando programación dinámica es posible construir un algoritmo para calcular **NumMult** $([A_1, \dots, A_\ell])$ que en el peor caso es $O(\ell^3)$*

Dos preguntas finales

¿Cómo podemos modificar el enfoque anterior para retornar una forma óptima para calcular $A_1 \cdot \dots \cdot A_\ell$?

Dos preguntas finales

¿Cómo podemos modificar el enfoque anterior para retornar una forma óptima para calcular $A_1 \cdot \dots \cdot A_\ell$?

- ▶ ¿Qué debemos agregar a la tabla para poder obtener esto?

Dos preguntas finales

¿Cómo podemos modificar el enfoque anterior para retornar una forma óptima para calcular $A_1 \cdot \dots \cdot A_\ell$?

- ▶ ¿Qué debemos agregar a la tabla para poder obtener esto?

El algoritmo usual no es el algoritmo más eficiente para multiplicar dos matrices de números enteros.

Dos preguntas finales

¿Cómo podemos modificar el enfoque anterior para retornar una forma óptima para calcular $A_1 \cdot \dots \cdot A_\ell$?

- ▶ ¿Qué debemos agregar a la tabla para poder obtener esto?

El algoritmo usual no es el algoritmo más eficiente para multiplicar dos matrices de números enteros.

- ▶ Dado un algoritmo $\mathcal{A}$ para calcular la multiplicación de dos matrices, ¿puede ser utilizado el enfoque anterior para obtener una forma óptima para calcular $A_1 \cdot \dots \cdot A_\ell$?

Dos preguntas finales

Suponga que para multiplicar $A_{m \times n}$ con $B_{n \times p}$ el algoritmo $\mathcal{A}$ realiza $f(m, n, p)$ multiplicaciones de números enteros.

Dos preguntas finales

Suponga que para multiplicar $A_{m \times n}$ con $B_{n \times p}$ el algoritmo $\mathcal{A}$ realiza $f(m, n, p)$ multiplicaciones de números enteros.

El enfoque descrito en las transparencias anteriores puede ser utilizado considerando la siguiente definición recursiva de **NumMult**:

$$\text{NumMult}([A_1, \dots, A_\ell]) = \begin{cases} 0 & \ell = 1 \\ \min_{1 \leq i \leq \ell-1} \{ \text{NumMult}([A_1, \dots, A_i]) + \\ \quad \text{NumMult}([A_{i+1}, \dots, A_\ell]) + \\ \quad f(\text{filas}(A_1), \text{col}(A_i), \text{col}(A_\ell)) \} & \ell > 1 \end{cases}$$