



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

Algoritmos Voraces

Profesor: Juan Francisco Diaz

jdiaz@eisc.univalle.edu.co

August 11, 2002



Page 1 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

1. Generalidades

- Continuamos con el objetivo de buscar buenas soluciones para problemas de optimización.
- La programación dinámica no es siempre “barata”.
- Un algoritmo voraz, en lugar de “pensar” mucho el camino a seguir, toma decisiones con rapidez sobre vistas locales: toma decisiones óptimas localmente, esperando que esto le lleve a una solución óptima global.
- Un algoritmo voraz no encuentra siempre una solución óptima, pero muchas veces la logra.



Page 2 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

2. El Problema de Selección de Actividades

- Suponga que se cuenta con un conjunto $S = \{a_1, \dots, a_n\}$, de **actividades** que necesitan usar un recurso que no puede ser usado sino por una actividad a la vez.

Cada actividad a_i tiene un tiempo inicial s_i y final f_i asociados, tal que $s_i \leq f_i$.

Las actividades i y j son **compatibles** si $[s_i, f_i] \cap [s_j, f_j] = \phi$.

- El problema de selección de actividades:

entrada: $S = \{a_1, \dots, a_n\}$ s_i, f_i $1 \leq i \leq n$

salida: $A \subseteq S$: todas las actividades de A son mutuamente compatibles, y A es el máximo posible.



Page 3 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

- Consideremos el siguiente algoritmo para resolver este problema.

Se supone que $f_1 \leq f_2 \leq \dots \leq f_n$ (si no, se ordenan primero), y que s y f son representadas por arreglos.

Selección_Actividades_Voraz (s, f)

```
1  $n \leftarrow \text{length}[s]$ 
2  $A \leftarrow \{a_1\}$ 
3  $j \leftarrow 1$ 
4 for  $i \leftarrow 2$  to  $n$ 
    $A \subseteq \{a_1, \dots, a_j, \dots, a_{i-1}\}, a_j \in A, a_{j+1}, a_{i-1} \notin A, j < i,$ 
    $A$  es una solución óptima para  $S' = \{a_1, \dots, a_m\}$ 
5 do if  $s_i \geq f_j$ 
6     then  $A \leftarrow A \cup \{a_i\}$ 
7          $j \leftarrow i$ 
8 return  $A$ 
```

Si $|S| = n$, el algoritmo funciona en tiempo $\mathcal{O}(n)$ (suponiendo que las actividades están ordenadas por tiempo final).

En cada paso se agrega a A la actividad que termine más pronto, que sea compatible con las escogidas.

Esto es una decisión **Voraz**, pues no depende de solucionar primero unos subproblemas.



Page 4 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

- Es correcto el algoritmo anterior? Es decir, siempre encuentra una solución óptima?

Este es un análisis que **siempre** se debe hacer, pues se puede estar ganando en eficiencia, pero perdiendo en corrección!!

Teorema: El algoritmo **Selección-Actividades-Voraz** produce una solución del máximo tamaño posible, para el problema de selección de actividades.

Demostración:

i) Existen soluciones óptimas que incluyen a_1 , es decir, si $A \subseteq S$ es una solución óptima entonces existe otra (o la misma) que incluye a_1

ii) Si A' es una solución óptima para $S' = \{a_i \in S : s_i \geq f_1\}$ entonces $A = A' \cup \{a_1\}$

es una solución óptima para S .



Page 5 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

3. Características de las Estrategias Voraces

- Voraz \Rightarrow
 - ★ Serie de buenas decisiones rápidas!!
 - ★ Se escoge la mejor decisión según criterios locales.
 - ★ No siempre se encuentra el óptimo, pero puede darse.
 - Cuando utilizar esta técnica?
 - ★ No hay receta, pero si el problema exhibe.
 - Propiedad de escogencia Voraz
 - Subestructura óptima
- Entonces la técnica voraz puede funcionar.



Page 6 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

• Propiedad de escogencia Voraz

- ★ Una solución óptima se puede hallar a partir de soluciones óptimas locales.
- ★ En programación dinámica, también se toman decisiones, pero ellas dependen de conocer las soluciones a subproblemas.
En cambio, con una técnica voraz, las decisiones se toman con la información que se tenga en el momento, y esas decisiones determinan los subproblemas que se deberán resolver.
- ★ Clave:
 - Demostrar que una solución óptima global se puede hallar a partir de decisiones óptimas locales.



Page 7 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

★ Estrategia

- Demostrar que existe al menos una solución óptima que contiene el resultado de la primera decisión “Voraz”.
 - Demostrar que una vez tomada una decisión “Voraz”, lo que queda por hacer es buscar una solución óptima, con el mismo algoritmo, para un subproblema.
 - Entonces, por inducción, se tendría que el algoritmo “Voraz” produce una solución óptima global.
-
- **Estuctura óptima:** Misma propiedad expuesta para utilizar la técnica de programación dinámica.



Page 8 of 30

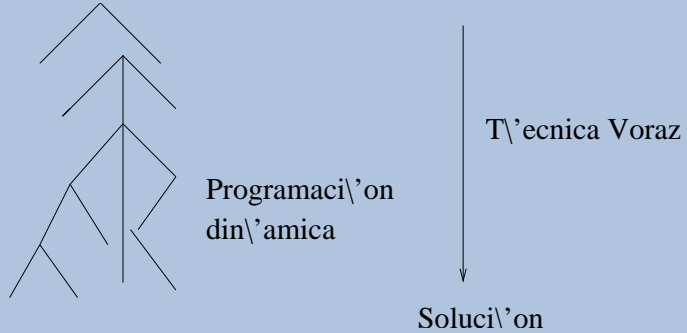
Full Screen

Quit



- Generalidades
- El Problema de...
- Características de...**
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...
- Algoritmo de...

• Técnicas Voraces v.s. Programación dinámica

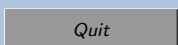
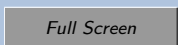


- ★ Tienen en común: estructura óptima
⇒ puede llevar a errores

Navigation controls: back, forward, page 9 of 30, Full Screen, Quit



- Generalidades
- El Problema de...
- Características de...
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...
- Algoritmo de...



★ Consideremos los problemas:

○ **morral 0-1**

entrada : n items y para cada item i , dos valores:

v_i : valor del item i (entero)

w_i : peso del item i (entero)

w : Máximo peso que resiste el morral

salida : un subconjunto de items que, juntos, no sobrepasan el peso que resiste el morral, y de máximo valor

○ **morral fraccional**

entrada : igual el anterior

salida : igual, pero en el morral se pueden meter fracciones de un item

○ Ambos problemas tienen la propiedad de subestructura óptima.

morral 0-1 :

Si $A = \{i_1, \dots, i_k\}$ es una solución óptima para la entrada $\{1, \dots, n\}, v_i, w_i, w$ entonces $A' = \{i_1, \dots, i_{j-1}, i_{j+1}, \dots, i_k\}$ es solución óptima para la entrada $I' = I_{\{i_j\}}$ y $w = w' - w_j, w'_j = w_j$

morral-fraccional:

Si $A = \{(p_1, i_1), \dots, (p_k, i_k)\}$ es una solución óptima para $\{1, \dots, n\}, w$, entonces $A' = \{(p_1, i_1), \dots, (p_j \cdot p_1 i_j), \dots, (p_k, i_k)\}$ es una solución óptima $I' = I, w' = w - p, w'_j = w_{i_j} - p, w'_k = w_k$ si $k \neq j$ $v'_k = v_k$



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 11 of 30

Full Screen

Quit

- o Pero solo uno de los dos problemas tiene la propiedad de escogencia voraz: **el morral fraccional**.

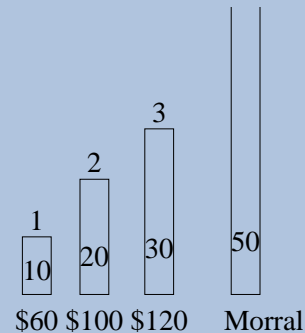
idea:

-Ordenar decrecientemente los items por $\frac{v_i}{w_i}$, valor por libra de peso.

-Escoger lo más que se pueda, que quepa en el morral del primer item, luego del segundo, etc, hasta llenar el morral.

- o Sin embargo, esta estrategia no funciona para **morral 0-1**:

La solución dada por la estrategia sería:





- Generalidades
- El Problema de...
- Características de...**
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...
- Algoritmo de...

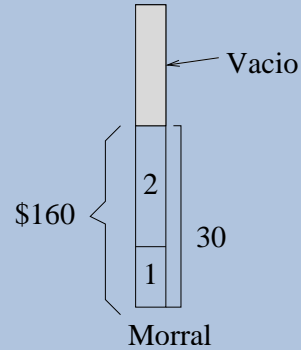


Page 12 of 30

Full Screen

Quit

pero la mejor solución es:





Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

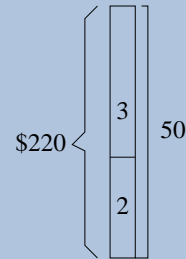


Page 13 of 30

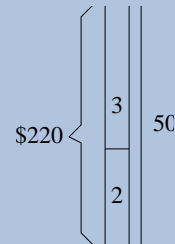
Full Screen

Quit

- En el caso del morral fraccional la solución sería: 220 Que si es óptima.
- En el caso del **morral 0-1** la decisión correcta no se puede tener hasta no analizar si es mejor meter un item, o dejarlo por fuera, y para este se necesita resolver subproblemas antes de tomar una decisión: \Rightarrow Prog. Dinámica.



Morral



Morral



4. Ejemplos de Algoritmos Voraces y Programación Dinámica en Algoritmos sobre Grafos

- Dados un grafo dirigido $G = (V, E)$, y una función de costo sobre los arcos, $C : E \rightarrow \mathbb{R}$, se define el costo de un camino $p = \langle v_0, \dots, v_k \rangle$ como la suma de los costos de los arcos que lo componen:

$$c(\langle v_0, v_1, \dots, v_k \rangle) = \sum_{i=1}^k c(v_{i-1}, v_i)$$

Así mismo se define el costo del camino más corto entre un nodo u y un nodo v como:

$$\delta(u, v) = \begin{cases} \min\{c(p) : u \xrightarrow{p} v\} & \text{Si existe un camino de } u \text{ a } v \\ \infty & \text{sino} \end{cases}$$

Un **camino más corto** entre u y v se define entonces como un camino p entre u y v tal que: $c(p) = \delta(u, v)$.

Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 14 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

- En este curso consideramos 2 problemas:

- ★ **Camino más corto 1 Origen-muchos destinos CMC1**

entrada: $G = (V, E)$, dirigido
 $s \in V$

Salida: Un camino más corto de s a v , para todos $u \in v$

- ★ **Camino más corto entre todo par de nodos: CMC2**

Entada $G = (V, E)$, dirigido

Salida para cada par de nodos $u, v \in V$, el camino más corto entre u y v .



Page 15 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

5. Un Algoritmo voraz para CMC1: Algoritmo de Dijkstra

- Una restricción $c : E \rightarrow R^+$.
- La respuesta se dará en un arreglo d donde se calculará δ , y un arreglo π que permite encontrar un camino más corto.

Dijkstra (G, c, s)

▷ Inicializar d y π

1 **for** $v \in V$

2 **do** $d[v] \leftarrow \infty$

3 $\pi[v] \leftarrow NIL$

4 $d[s] \leftarrow 0$

▷ Inicializar S y Q :

5 $S \leftarrow \phi$

6 $Q \leftarrow V$

▷ Invariante: $\forall v \in S : d[v] = \delta(s, v)$ $S \cup Q = V$

▷ $\forall v \in Q : d[v] = \delta_{G_s}(s, v) \geq \delta(s, v)$

7 **while** $Q \neq \phi$

8 **do** $u \leftarrow v \in Q : d[v]$ es minimo ▷ $d[u] = \delta(s, u)$

9 $S \leftarrow S \cup \{u\}$

10 **for** $v \in Adj[u]$

11 **do if** $d[v] > d[u] + c(u, v)$

12 **then** $d[v] \leftarrow d[u] + c(u, v)$

13 $\pi[v] \leftarrow u$

▷ $Q = \phi$

▷ $S = V$

▷ $\forall v \in V : d[v] = \delta(s, v)$



Page 16 of 30

Full Screen

Quit



Universidad
del Valle

Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 17 of 30

Full Screen

Quit

- Corrección del algoritmo de Dijkstra
 - ★ La clave para demostrar la corrección del algoritmo está en demostrar que cuando en la línea 8 se escoge un nodo u para ingresar a S , este ya cumple con la condición para ello: $d[u] = \delta(s, u)$
 - ★ Al comenzar el **while** (línea 7) la condición invariante es cierta.
 - ★ Al terminar el **while**, la condición invariante unida al hecho que $Q = \phi$, implica que el cálculo es correcto.
 - ★ Falta probar que el invariante se mantiene:
 1. La ecogencia en la línea 8 es adecuada.
 2. La actualización hecha en las líneas 12 y 13 es adecuada.



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

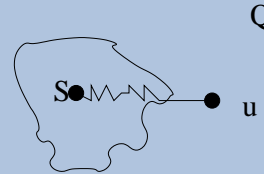


Page 18 of 30

Full Screen

Quit

Prueba de 1:



Como el invariante es cierto cada que inicia el ciclo entonces:

$$d[u] = \delta_{G_s}(s, u) \geq \delta(s, u)$$

Vamos a probar que, para ese u en particular

$$\delta(s, u) \geq \delta_{G_s}(s, u)$$

De alli se concluirá que $\delta_{G_s}(s, u) = \delta(s, u)$.

Suponga que $\delta(s, u) < \delta_{G_s}(s, u)$

Entonces existe un camino p de s a u cuyo costo es inferior a $\delta_{G_s}(s, u)$:

$$c(p) < \delta_{G_s}(s, u)$$

Sea v el primer nodo en p , que no está en S (es decir que está en Q).

Como el invariante se cumple, se tiene que

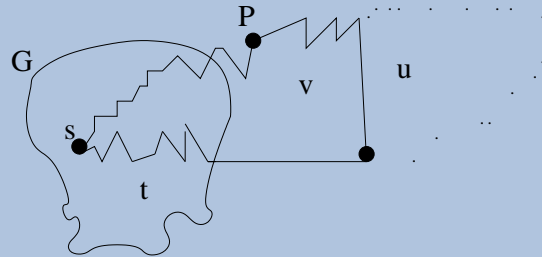
$$d[v] = \delta G_s(s, u)$$

y por la escogencia del u , se tiene que

$$d[u] = \delta G_s(s, u) \leq \delta G_s(s, v) = d[v]$$

Entonces $c(p) > \delta G_s(s, v) \geq \delta G_s(s, u)$ Contradicción!! por tanto $\delta(s, u) \geq \delta G_s(s, u)$

prueba de 2: sencillo





- Generalidades
- El Problema de...
- Características de...
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...
- Algoritmo de...

• Análisis del algoritmo

$$\star T(n) = |V| + |V|.(\text{costo Escoger } U, \text{ linea 8}) + E.(\text{costo de actualizar})$$

- o Si d es un arreglo de $|V|$ elementos

Costo Escoger $u : |Q| \leq |V|$

$$\Rightarrow \mathcal{O}(|V|^2 + |E|) = \mathcal{O}(|V|^2)$$

- o Si d es un heap de $|V|$ elementos:

costo Escoger $u = \log(|Q|) \leq \log(|V|)$

costo de actualizar: $\log(|Q|) \leq \log(|V|)$

$$\Rightarrow \mathcal{O}((|V| + |E|)\log(|V|))$$

$$= \mathcal{O}(|E|\log(V))$$

si todos los vertices son alcanzables desde el origen.

Sirve, realmente si $|E| = \mathcal{O}(|V|)$

Sino, es más costoso que la primera opción!!

- o Incluso se puede lograr

$$\mathcal{O}(|V|\log(|V|) + |E|)$$



Page 20 of 30

Full Screen

Quit



6. El problema CMC2: Programación Dinámica

- Suponemos que la función de costo está representada por una matriz c :

$$c_{ij} = \begin{cases} 0 & \text{si } i=j \\ c_{(ij)} & \text{si } (i, j) \in E, i \neq j \\ \infty & \text{si } i \neq j, (i, j) \notin E \end{cases}$$

La salida estará representada por 2 matrices:

$$D_{ij} = \delta(i, j)$$

$$\Pi_{ij} = \begin{cases} \text{Nul} & \text{si } i = j \text{ o no hay camino entre } i \text{ y } j \\ k, & k \text{ es un predecesor de } j \text{ es un camino mas corto de } i \text{ a } j \end{cases}$$



Page 21 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2: ...

Algoritmo de...

- **Camino más corto y multiplicación de matrices:** *CMC2* por programación dinámica.

★ Estructura de una solución óptima:
Sea p un camino más corto de i a j .

Entonces p es de la forma $i \xrightarrow{p'} k \rightarrow j$, donde
 p' es un camino más corto entre i y k y
 $\delta(i, j) = \delta(i, k) + c_{kj}$



Page 22 of 30

Full Screen

Quit



- Generalidades
- El Problema de...
- Características de...
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...**
- Algoritmo de...



★ Definición recursiva de una solución:

Sea $d_{ij}^{(m)}$ = “ Camino más corto de i a j que utiliza m arcos a lo sumo ”

$$\text{Entonces } d_{ij}^{(0)} = \begin{cases} 0 & i = j \\ \infty & i \neq j \end{cases}$$

$$d_{ij}^{(m)} = \min(d_{ij}^{(m-1)}, \min_{1 \leq k \leq n} \{d_{ik}^{(m-1)} + c_{kj}\})$$

$$= \min_{1 \leq k \leq n} \{d_{ik}^{(m-1)} + c_{kj}\}$$

Entonces cómo calcular $\delta(i, j)$?

Si no existen ciclos negativos en el grafo: \Rightarrow los CMC son simples.

$$\delta(i, j) = d_{ij}^{(n-1)} = d_{ij}^{(n)} = \dots$$



- ★ Cálculo de una solución óptima
La idea es calcular $D^{(1)}, \dots, D^{(n-1)} = D$.
Observe que $D^{(1)} = C$
primero calculemos $D^{(i+1)}$ a partir de $D^{(i)}$

Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 24 of 30

Full Screen

Quit



Generalidades
El Problema de...
Características de...
Ejemplos de...
Un Algoritmo voraz...
El problema CMC2:...
Algoritmo de...



Page 25 of 30

Full Screen

Quit

★ Extienda_caminos_ más_ cortos (D,C)

```
1  $n \leftarrow \text{filas}[D]$ 
 $\triangleright D' \Rightarrow (d_{ij})$  será la respuesta
    $D' = (d''_{ij})$  sea una matriz de  $n \times n$ 
2 for  $i \leftarrow 1$  to  $n$ 
3   do for  $j \leftarrow 1$  to  $n$ 
4     do  $d'_{ij} \leftarrow \infty$ 
5       for  $k \leftarrow 1$  to  $n$ 
6         do  $d'_{ij} \leftarrow \min(d'_{ij}, d_{ik} + c_{kj})$ 
7 return  $D'$ 
```

El algoritmo anterior es de orden $\mathcal{O}(|V|^3)$



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

Observe que esto no es más que multiplicación de matrices D, C donde

$\Sigma \leftarrow MIN$

$$C_{ij} = \sum_{k=1}^n a_{ik} \times b_{kj}$$

$$a' \leftarrow d^{(m-1)} + w' \leftarrow b + c \leftarrow d^{(m)}$$

$$\sum_{k=1}^n a_{ik} b_{kj} \leftarrow MIN_{i \leq k \leq n} \{ (d_i k^{(m-1)} + C_{kj}) \}$$



Page 26 of 30

Full Screen

Quit



CMC2-lento (C)

```
1  $n \leftarrow \text{filas}[C]$ 
2  $D^{(1)} \leftarrow C$ 
3 for  $m \leftarrow 2$  to  $n - 1$ 
4   do  $D^{(m)} \leftarrow \text{Extienda\_Caminos\_más\_cortos}(D^{(m-1)}, C)$ 
5 return  $D^{(n-1)}$ 
 $\mathcal{O}(|V| \cdot |V|^3) = \mathcal{O}(|V|^4)$ 
```

¿Cómo agilizarlo?
 $D^n = (D^{n/2})^2$

Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 27 of 30

Full Screen

Quit



Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...

7. Algoritmo de Strassen para Multiplicación de Matrices

- Idea: Dividir y Conquistar

- Sea $A, B \in \mathcal{U}_{n \times n}$, $n = 2^k$

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} B = \begin{pmatrix} e & g \\ f & h \end{pmatrix}, a, b, c, d, e, f, g, h, \in \mathcal{U}_{n/2 \times n/2}$$

$$\Rightarrow AB = \begin{pmatrix} r & s \\ t & u \end{pmatrix} \text{ donde}$$

$$r = ae + bf$$

$$s = ag + bh$$

$$t = ce + df$$

$$u = cg + dh$$

$$\Rightarrow T(n) = 8T(n/2) + \theta(n^2)$$

Por teorema maestro

$$T(n) = \theta(n^3)$$

y no se mejora el algoritmo clásico de multiplicación de matrices.



Page 28 of 30

Full Screen

Quit



- Generalidades
- El Problema de...
- Características de...
- Ejemplos de...
- Un Algoritmo voraz...
- El problema CMC2:...
- Algoritmo de...

- Strassen descubrió otra manera de dividir y conquistar:

$$T(n) = 7T(n/2) + \theta(n^2)$$

cuya solución es $T(n) = \theta(n^{\lg 7}) = \theta(n^{2.81})$

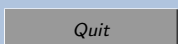
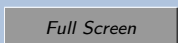
que es mejor que $\theta(n^3)$

El método de Strassen en

- ★ Dividir A y B como antes.
- ★ Calcular 14 matrices de $n/2 \times n/2$ con $\theta(n^2)$ operaciones: (Adiciones y Sustracciones)
 $A_1, B_1, \dots, A_7, B_7$
- ★ Calcular C , a partir de P_1, \dots, P_7 , en $\theta(n^2)$ operaciones.

$A_1 = a$	$B_1 = g - h$	$P_1 = ag - ah$	$T(n/2) + \theta(n^2)$
$A_2 = a + b$	$B_2 = h$	$P_2 = ah + bh$	$T(n/2) + \theta(n^2)$
$A_3 = c + d$	$B_3 = e$	$P_3 = ce + de$	$T(n/2) + \theta(n^2)$
$A_4 = d$	$B_4 = f - e$	$P_4 = df - de$	$T(n/2) + \theta(n^2)$
$A_5 = a + d$	$B_5 = e + h$	$P_5 = ae + ah + de + dh$	$T(n/2) + \theta(n^2)$
$A_6 = b - d$	$B_6 = f + h$	$P_6 = bf + bh - df - dh$	$T(n/2) + \theta(n^2)$
$A_7 = a - c$	$B_7 = e + g$	$P_7 = ae + ag - ce - cg$	$T(n/2) + \theta(n^2)$
			$\frac{7T(n/2) + \theta(n^2)}{}$

Sea





Generalidades

El Problema de...

Características de...

Ejemplos de...

Un Algoritmo voraz...

El problema CMC2:...

Algoritmo de...



Page 30 of 30

Full Screen

Quit

$$\begin{aligned} r &= P_5 + P_4 - P_2 + P_6 & \theta(n^2) \\ s &= P_1 + P_2 & \theta(n^2) \\ t &= P_3 + P_4 & \theta(n^2) \\ u &= P_5 + P_1 - P_3 - P_7 & \frac{\theta(n^2)}{\theta(n^2)} \end{aligned}$$

$$\text{Total } T(n) = 7T(n/2) + \theta(n^2)$$

- Este algoritmo en la práctica solo es útil si las matrices son grandes ($n > 45$) y densos.
- Usando técnicas más avanzadas se han encontrado mejores algoritmos. De hecho hay uno de orden $\theta(n^{2.376})$
- Sea MM el problema de multiplicar matrices y $T_{MM}(n)$ el tiempo de ejecución asociado a un algoritmo A , cualquiera que resuelva el problema. Se tiene que:

$$\omega(n^2) \leq T_{MM}(n) \leq \mathcal{O}(n^{2.376})$$

No se conoce una cota exacta para este problema.