Tema 2.

La Descomposición QR. Algoritmos paralelos basados en reflexiones de Householder.

Estabilidad numérica del algoritmo de Householder

The roundoff properties associated with Householder matrices are very favorable. Wilkinson (AEP, pp. 152-62) shows that house produces a Householder vector $\hat{v}$ very near the exact v. If $\hat{P} = I - 2\hat{v}\hat{v}^T/\hat{v}^T\hat{v}$ then

$$\| \hat{P} - P \|_2 = O(u)$$

Moreover, the computed updates with $\hat{P}$ are close to the exact updates with P :

$$fl(\hat{P}A) = P(A+E) \qquad \| E \|_2 = O(u)\| A \|_2)$$

$$fl(A\hat{P}) = (A+E)P \qquad \| E \|_2 = O(u)\| A \|_2)$$

If $A=Q_1R_1$

the corresponding result for the Householder approach is

$$\hat{Q}_1^T\hat{Q}_1 = I + E_H \qquad \| E_H \|_2 \approx u .$$

Estabilidad numérica del algoritmo de Givens

Las propiedades de error de las rotaciones de Givens son tan buenas como las de Householder.

We offer some remarks about the propagation of roundoff error in algorithms that involve sequences of Householder/Givens updates. To be precise, suppose $A = A_0 \in \mathbb{R}^{m\times n}$ is given and that matrices $A_1, \ldots, A_p = B$ are generated via the formula

$$A_k = fl(\hat{Q}_k A_{k-1} \hat{Z}_k) \qquad k = 1{:}p .$$

Assume that the above Householder and Givens algorithms are used for both the generation and application of the $\hat{Q}_k$ and $\hat{Z}_k$. Let Q_k and Z_k be the orthogonal matrices that would be produced in the absence of roundoff. It can be shown that

$$B = (Q_p \cdots Q_1)(A + E)(Z_1 \cdots Z_p), \tag{5.1.11}$$

where $\| E \|_2 \leq cu \| A \|_2$ and c is a constant that depends mildly on n, m, and p. In plain English, B is an exact orthogonal update of a matrix near to A.

Descomposición QR basada en Givens: Casos especiales

Reducción a la forma triangular superior de una matriz de Hessenberg superior

$$Q^T \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ & x & x & x \\ & & x & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & x & x \\ ① & x & x & x \\ & ② & x & x \\ & & ③ & x \end{bmatrix}$$

Coste: $3n^2$ flops

Reducción a la forma triangular superior de una matriz tridiagonal

$$Q^T \begin{bmatrix} x & x & & \\ x & x & x & \\ & x & x & x \\ & & x & x \end{bmatrix} \rightarrow \begin{bmatrix} x & x & y & \\ ① & x & x & y \\ & ② & x & x \\ & & ③ & x \end{bmatrix}$$

Coste: 25n flops

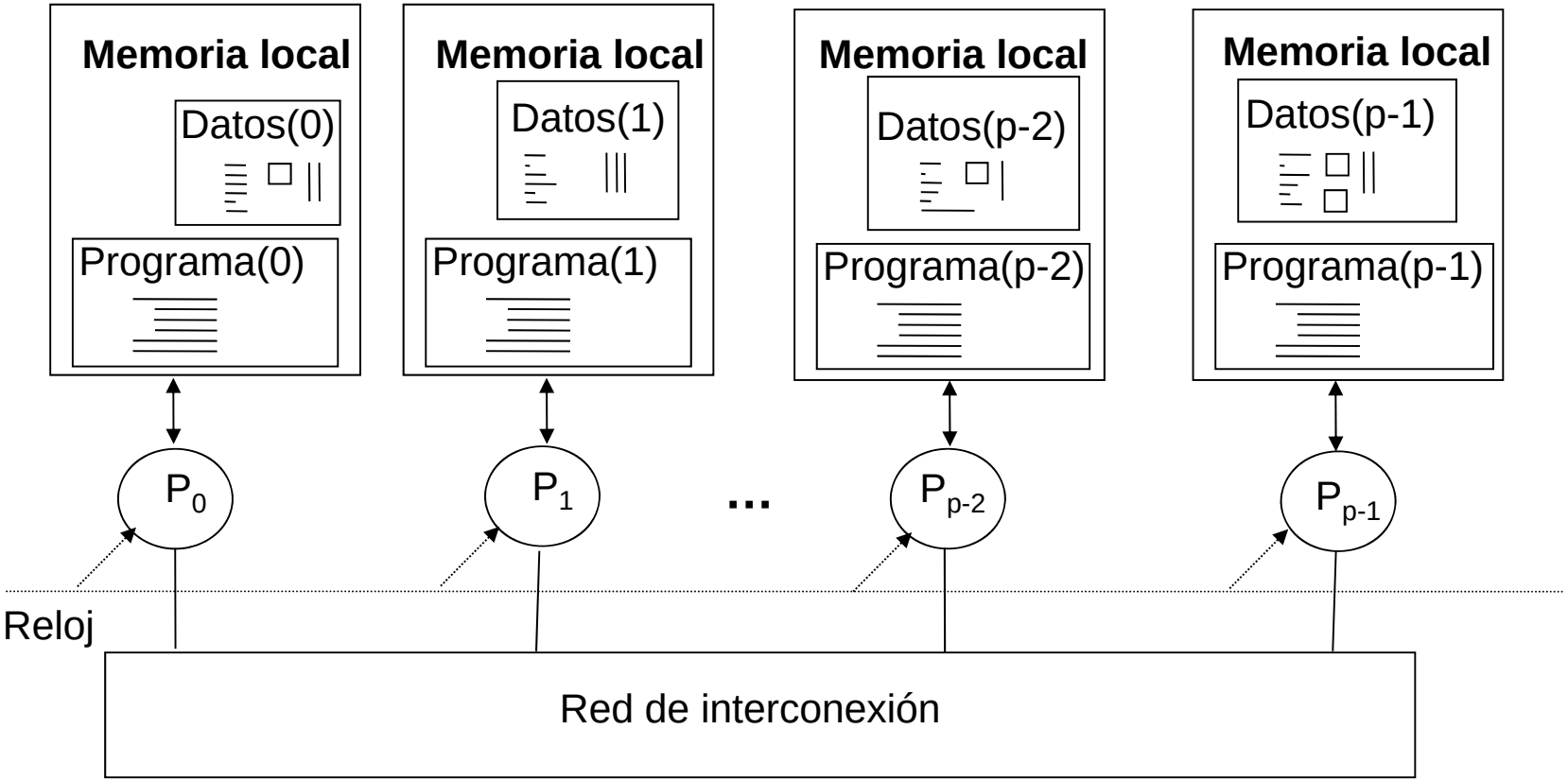
Reducción a la forma triangular inferior de una matriz tridiagonal

$$\begin{bmatrix} x & x & & \\ x & x & x & \\ & x & x & x \\ & & x & x \end{bmatrix} Q^T \rightarrow \begin{bmatrix} x & ① & & \\ x & x & ② & \\ y & x & x & ③ \\ & y & x & x \end{bmatrix}$$

Coste: 25n flops

Paralelización del Algoritmo de Householder
en el Modelo de Paso de Mensajes

Objetivo:
Diseñar un algoritmo paralelo óptimo para triangularizar una matriz en un multicomputador



Características:

Entorno de paso de mensajes (MPI)

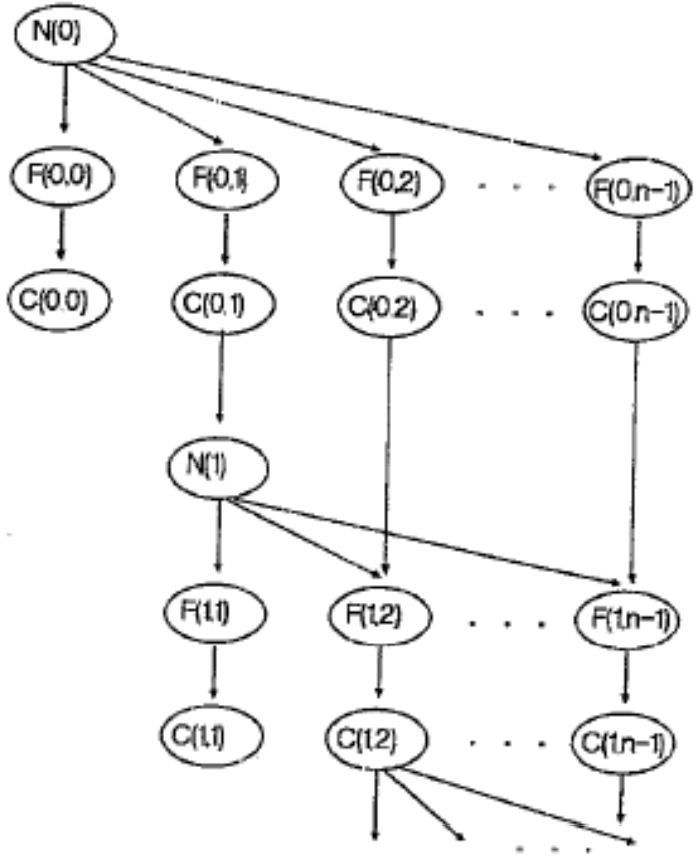
Se conoce el tiempo de ejecución de 1 Flop (operación elemental en coma flotante:+,-,*,:)

Enviar un mensaje de N palabras desde un nodo a otro (o a otros) tiene un coste de $N\tau+\beta$, con τ y β conocidos ($\tau=t_w$ **y** $\beta=t_s$)

$$t_p \cong t_A + t_C - t_{sol} \cong t_A + t_C$$

Grafo de dependencias

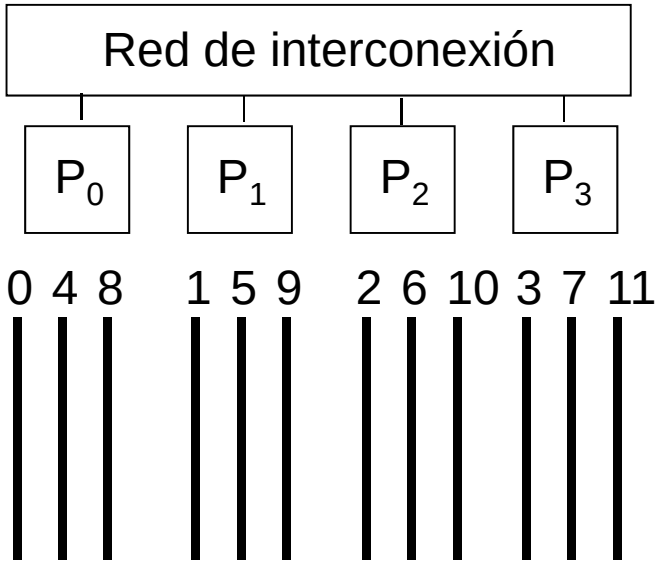
N(j)=Norm(j); F(j,k)=Fact(j,k); C(j,k)=Cmod(j,k)



Distribución cíclica: suponiendo n=k*p

$$Columna\ j \rightarrow P_{jMODp}$$

$$\text{Pr ocesador } i \rightarrow \text{ Columnas } rp+i, r = 0,1,...,(n/p)-1$$



Idea básica del algoritmo

Para cada columna

1. Calcular los parámetros de la transformación de Householder (sólo el procesador que la contiene)
2. Difundir los parámetros al resto de procesadores
3. Actualizar las restantes columnas (cada procesador sólo las que contiene)

Algoritmo paralelo

EN PARALELO: PARA pr=0,1,…,p-1

En P_{pr}:

PARA j=0,1,…,n-1

SI jMODp=pr (*columna j pertenece a P_{pr} *)

ENTONCES

Norm(j)

Difunde parámetros (* v_j, β_j*)

EN OTRO CASO

Espera parámetros

FIN SI

PARA k=j,j+1,…,n-1

SI kMODp=pr (*columna k pertenece a P_{pr} *)

ENTONCES

Fact(j,k)

Cmod(j,k)

FIN SI

FIN PARA

FIN PARA

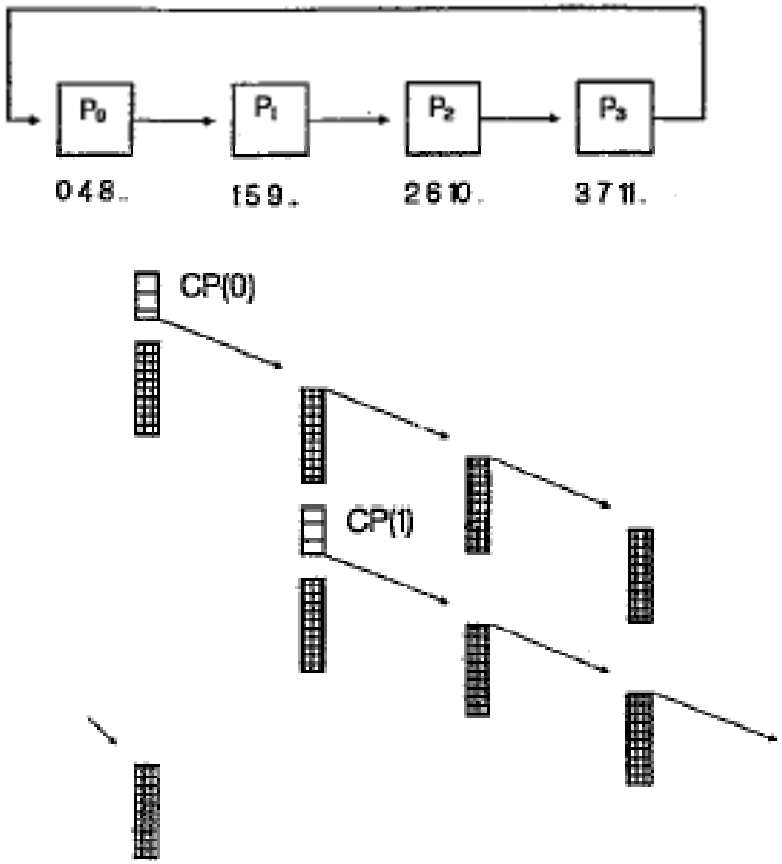
FIN PARA

Coste:

$$t_A = (2n^2 / p)(m - n/3)\ Flops$$

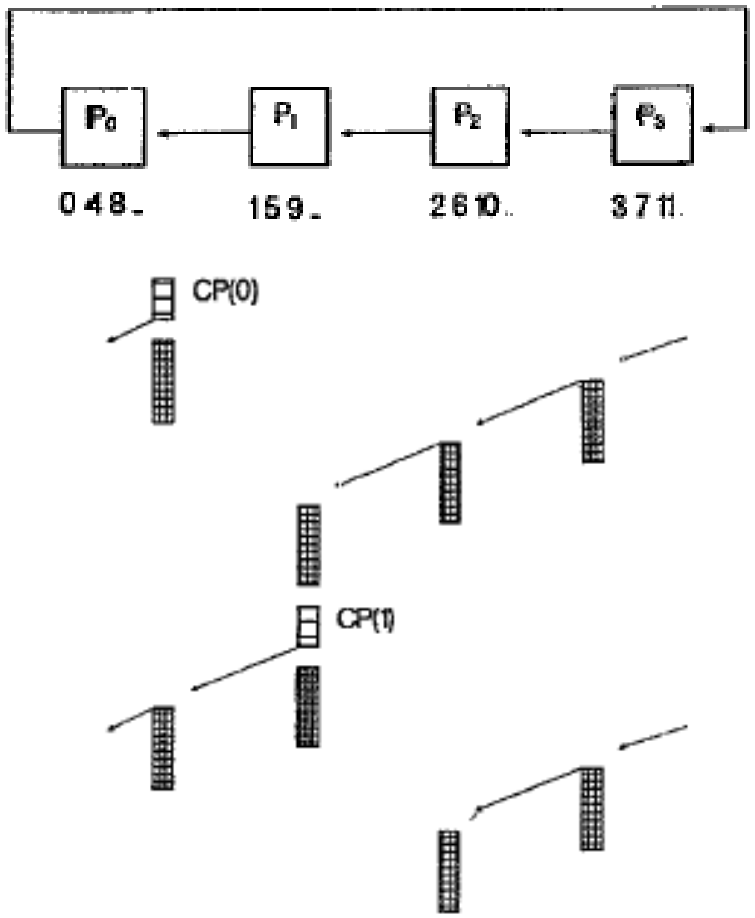
$$t_C = p\left[(mn - n^2/2)\tau_{DIF} + n\beta_{DIF}\right]$$

Algoritmo sobre un anillo unidireccional



Algoritmo asíncrono, con solapamiento de comunicaciones y operaciones aritméticas

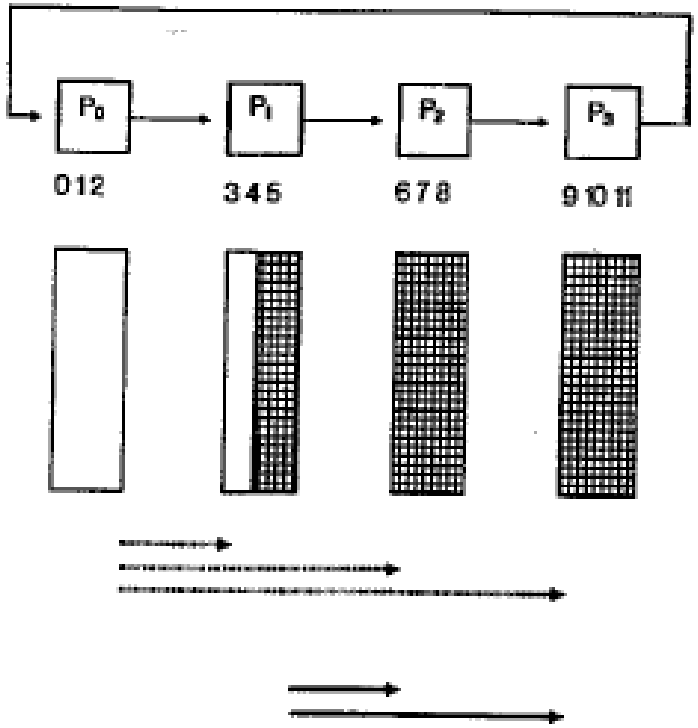
Algoritmo sobre un anillo unidireccional



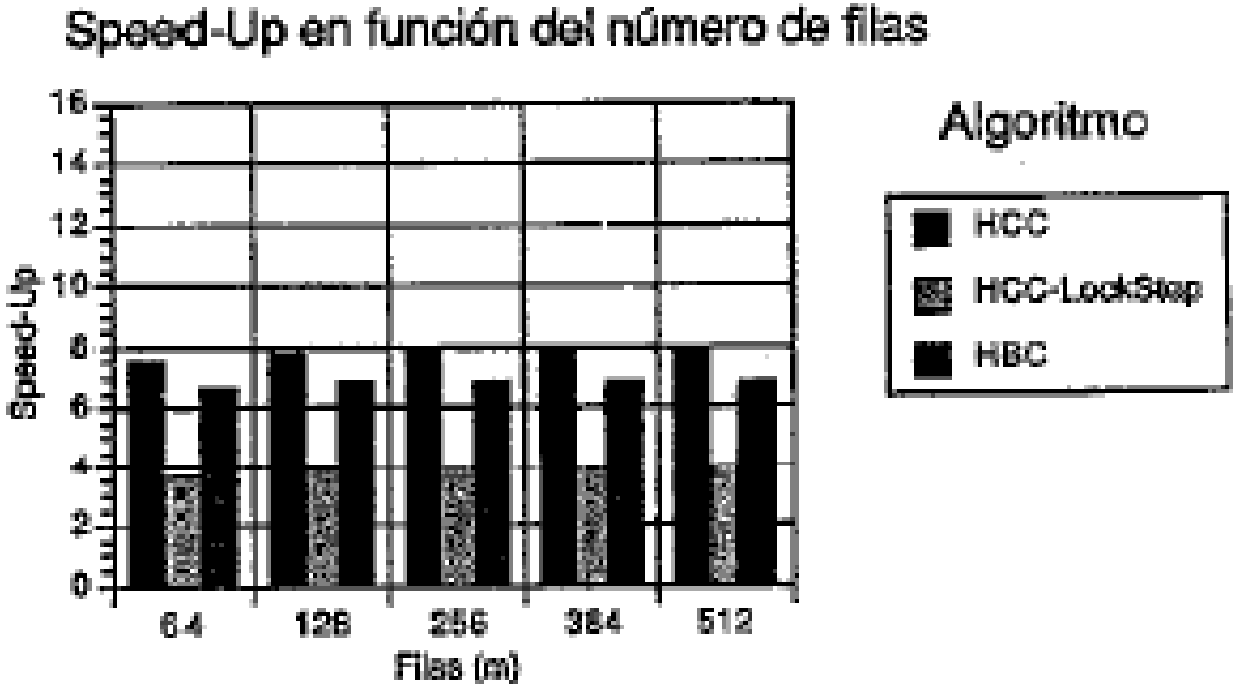
Algoritmo síncrono (lock-step), sin solapamiento de comunicaciones y operaciones aritméticas

Algoritmo con distribución por bloques de columnas consecutivas

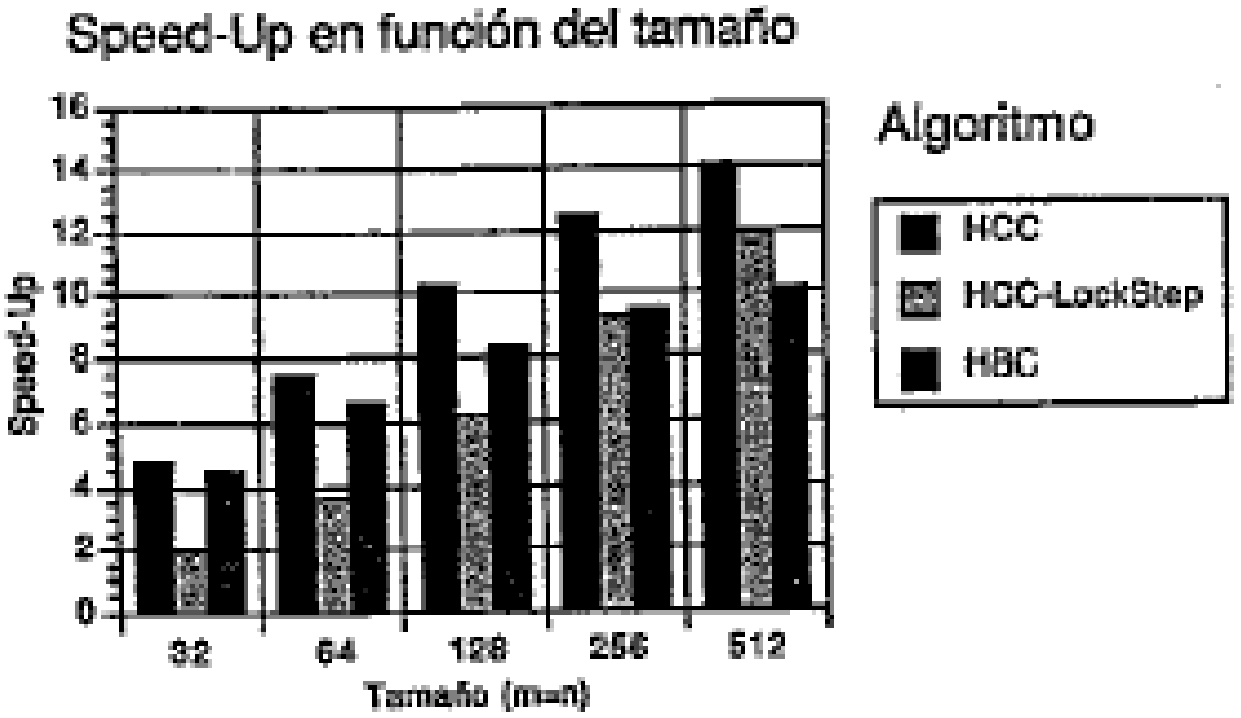
- Distribución cíclica: tiende a equilibrar la carga
- Distribución por bloques de columnas consecutivas: tiende a disminuir el tiempo de comunicación



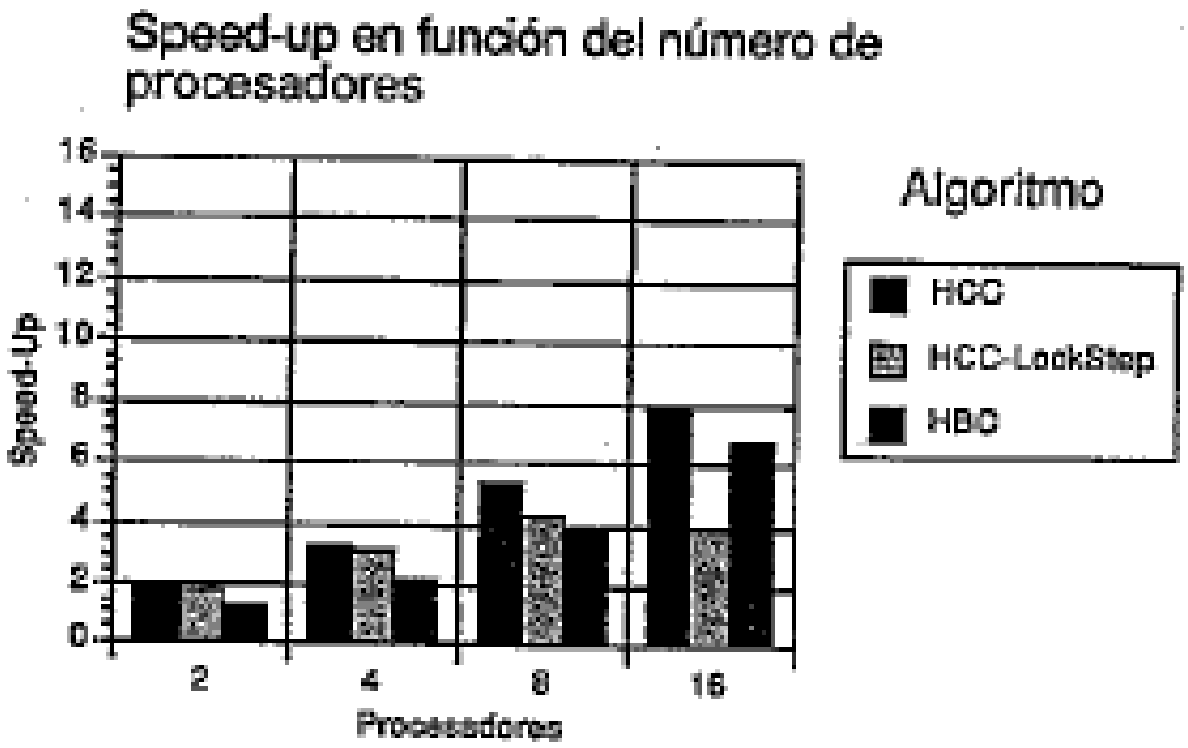
MATRIZ RECTANGULAR mx64
16 PROCESADORES



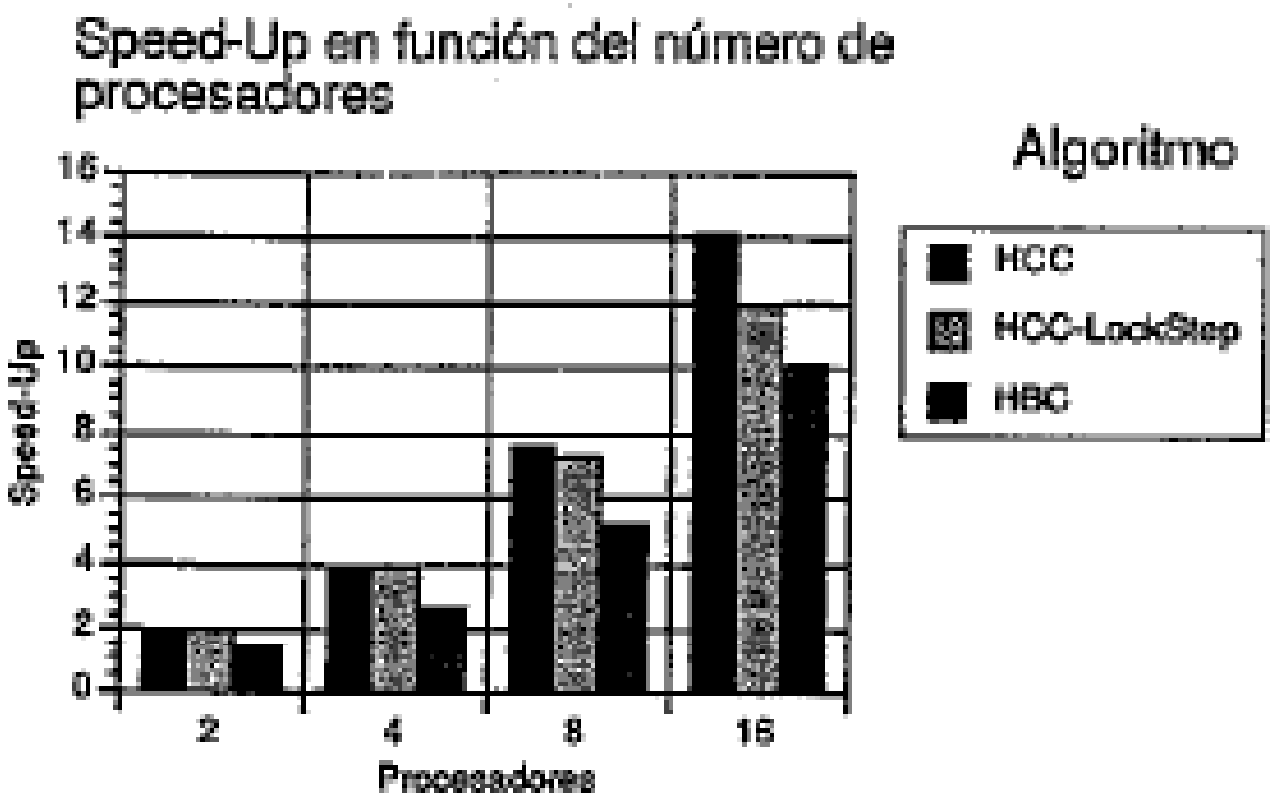
MATRIZ CUADRADA nxn
16 PROCESADORES



MATRIZ RECTANGULAR 512x64



MATRIZ CUADRADA 512X512



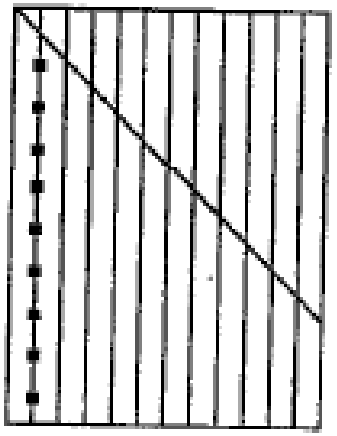
Otros aspectos del Algoritmo Paralelo

Algoritmo general

```
PARA j=0,1,…,n-1
    CalculaParámetros(j)
    PARA k=j,j+1,…,n-1
        ActualizarColumna(j,k)
```

Eliminación Gaussiana

Parámetros=multiplicadores



Actualización:

$$A_k = A_k - a_{jk} * A_j$$

$$\left| \quad \right| = \left| \; - a_{jk} * \right|$$

Algoritmo paralelo general

EN PARALELO: PARA pr=0,1,…,p-1

En P_{pr}:

PARA j=0,1,…,n-1

SI jMODp=pr (*columna j pertenece a P_{pr} *)

ENTONCES

CalcularParámetros(j)

Difunde parámetros (* , *)

EN OTRO CASO

Espera parámetros

FIN SI

PARA k=j,j+1,…,n-1

SI kMODp=pr (*columna k pertenece a P_{pr} *)

ENTONCES

ActualizarColumna(j,k)

FIN SI

FIN PARA

FIN PARA

FIN PARA

Organización por bloques

$$P_1 = I - \tau_1 z_1 z_1^T ; P_2 = I - \tau_2 z_2 z_2^T ; A = P_1 P_2 R_{(2)}$$

$$R_{(2)} = \begin{bmatrix} r_{11} & r_{12} & \bar{a}_{13} & \bar{a}_{14} \\ z_{21} & r_{22} & \bar{a}_{23} & \bar{a}_{24} \\ z_{31} & z_{32} & \bar{a}_{33} & \bar{a}_{34} \\ z_{41} & z_{42} & \bar{a}_{43} & \bar{a}_{44} \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ z_{21} & 1 \\ z_{31} & z_{32} \\ z_{41} & z_{42} \end{bmatrix} = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \quad \bar{A} = \begin{bmatrix} \bar{a}_{13} & \bar{a}_{14} \\ \bar{a}_{23} & \bar{a}_{24} \\ \bar{a}_{31} & \bar{a}_{34} \\ \bar{a}_{41} & \bar{a}_{44} \end{bmatrix} = \begin{bmatrix} \bar{A}_1 \\ \bar{A}_2 \end{bmatrix}$$

$$P_1 P_2 = I - ZTZ^T = I - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} T \begin{bmatrix} Z_1^T & Z_2^T \end{bmatrix}, \text{con } T = \begin{bmatrix} t_{11} & t_{12} \\ 0 & t_{22} \end{bmatrix}$$

Organización por bloques

Caso de 2 columnas

- $P_1 P_2 = I - \tau_1 (z_1 z_1^T) - \tau_2 (z_2 z_2^T) + \tau_1 \tau_2 (z_1^T z_2)(z_1 z_2^T)$

- $$I - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} T \begin{bmatrix} Z_1^T & Z_2^T \end{bmatrix} = I - \begin{bmatrix} z_1 & z_2 \end{bmatrix} T \begin{bmatrix} z_1^T \\ z_2^T \end{bmatrix} =$$

$$I - \begin{bmatrix} t_{11} z_1 + t_{21} z_2 & t_{12} z_1 + t_{22} z_2 \end{bmatrix} \begin{bmatrix} z_1^T \\ z_2^T \end{bmatrix} =$$

$$= I - t_{11} z_1 z_1^T - t_{21} z_2 z_1^T - t_{12} z_1 z_2^T - t_{22} z_2 z_2^T$$

$$\text{con } T = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix}, Z = \begin{bmatrix} z_1 & z_2 \end{bmatrix} \qquad \Rightarrow t_{21} = 0$$

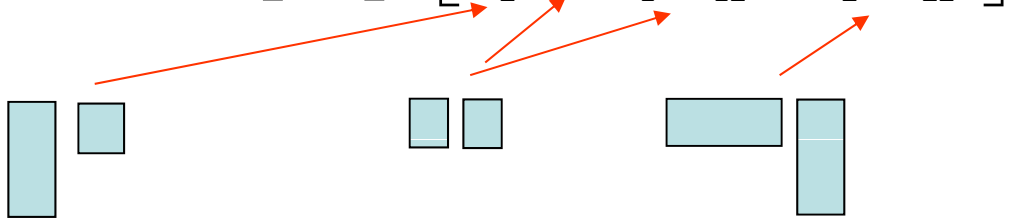
Organización por bloques

$$P_1 = I - \tau_1 z_1 z_1^T ; P_2 = I - \tau_2 z_2 z_2^T ; A = P_1 P_2 R_{(2)}$$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad R_{(2)} = \begin{bmatrix} \overline{R} & \overline{A} \end{bmatrix}$$

$$\overline{A} = (P_1 P_2)^T \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix} = (I - ZTZ^T)^T \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix} = (I - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} T^T \begin{bmatrix} Z_1^T & Z_2^T \end{bmatrix}) \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix}$$

$$\overline{A} = \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix} - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} T^T (Z_1^T A_{12} + Z_2^T A_{22}) = \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix} - \begin{bmatrix} Z_1 T^T (Z_1^T A_{12} + Z_2^T A_{22}) \\ Z_2 T^T (Z_1^T A_{12} + Z_2^T A_{22}) \end{bmatrix}$$

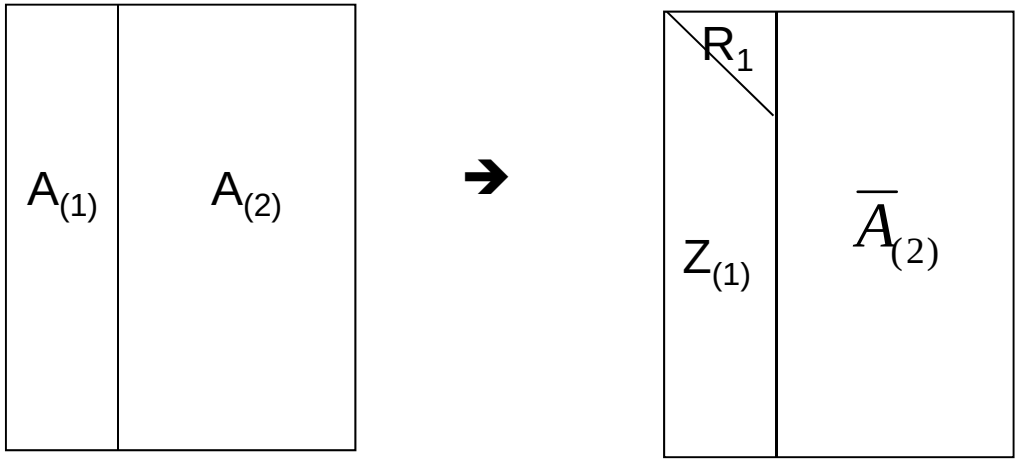


Organización por bloques

Caso general

$$P_1 = I - \tau_1 z_1 z_1^T ; P_2 = I - \tau_2 z_2 z_2^T ; ... ; P_k = I - \tau_k z_k z_k^T$$

$$A = P_1 P_2 ... P_k R_{(2)}$$

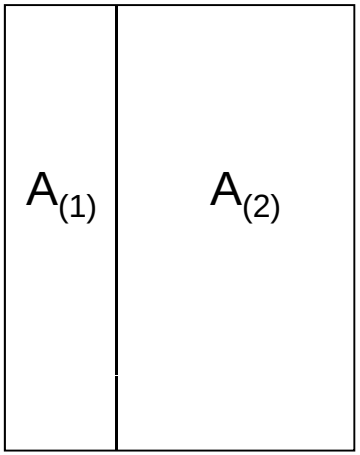


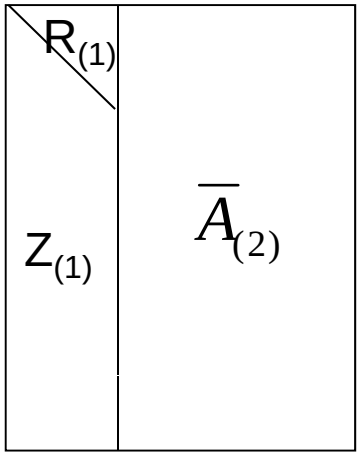
$$P_1 P_2 ... P_r = I - ZTZ^T = I - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} T \begin{bmatrix} Z_1^T & Z_2^T \end{bmatrix}, \text{ con } T = \begin{bmatrix} t_{11} & t_{12} & ... & t_{1k} \\ & t_{22} & ... & t_{2k} \\ & & ... & . \\ & & & t_{kk} \end{bmatrix}$$

Organización por bloques

1. Aplicar la triangularización de Householder a $A_{(1)}$: *Calcular* $Z_{(1)}, R_{(1)}, T_{(1)}$

2. Actualizar $A_{(2)} : \overline{A}_{(2)} = \left(I - Z_{(1)} * T_{(1)} * {Z_{(1)}}^{T}\right)^{T} * A_{(2)}$



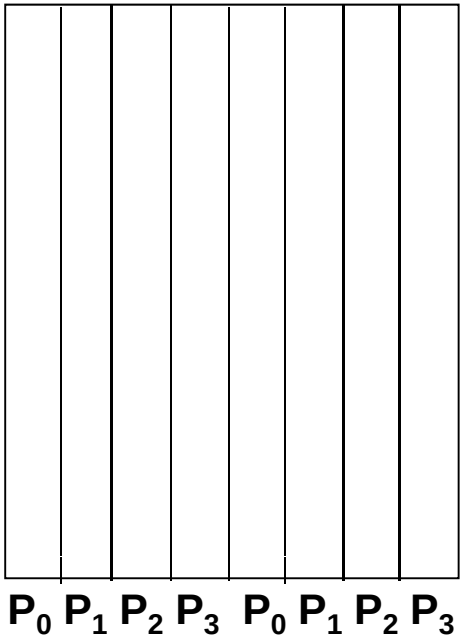


Las operaciones de actualización se pueden hacer utilizando BLAS3 ya que son productos matrizXmatriz

Algoritmo de triangularización de Householder orientado por bloques en el modelo de memoria distribuida

$$nb = n/tb, \quad A_j \rightarrow P_{jMODp},$$

$$A = \left[A_0, A_1, \ldots, A_{nb-1}\right] \in \Re^{mxn}$$



Para $pr = 0,1,\ldots, p-1$

En P_{pr} :

Para $j = 0,1,\ldots, nb-1$

Si $jMODp = pr$

Triangulariza A_j y *calcula* Z_j, T_j

Difunde Z_j, T_j

en otro caso

Espera Z_j, T_j

finsi

Para $k = j+1, j+2,\ldots, nb-1$

Si $kMODp = pr$

Actualiza $A_k = \left[I - Z_j T_j Z_j^T\right] A_k$

finsi

finpara

finpara

Prestaciones del algoritmo

$$T_p = (2 + \frac{1}{2tb})\left[\frac{n^2}{p}\left(m - \frac{n}{3}\right)\right] Flop_{BLAS3} + 2ntb\left(m - \frac{n}{2}\right) Flop + n(m - \frac{n}{2})\tau_{DIF} + \left(\frac{n}{tb}\right)\beta_{DIF}$$

$$\frac{dT_p}{dtb} = 0 \Rightarrow tb = \sqrt{\frac{n\left(m - \frac{n}{3}\right) Flop_{BLAS3} + 2p\beta_{DIF}}{4p\left(m - \frac{n}{2}\right) Flop}}$$