

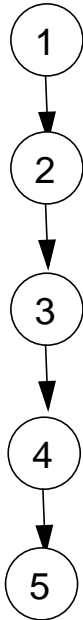
Aspectos Esenciales en la Programación Paralela

- Ejecución paralela versus ejecución secuencial
- Granularidad de las tareas
- Estilos de paralelismo
- Planificación
- Impacto de la decisiones de diseño en las características del algoritmo
- Ejemplo sencillo
- Sistema triangular de ecuaciones en un modelo de variables compartidas
- Sistema triangular de ecuaciones en un modelo de paso de mensajes
- Sistema triangular de ecuaciones en HPF



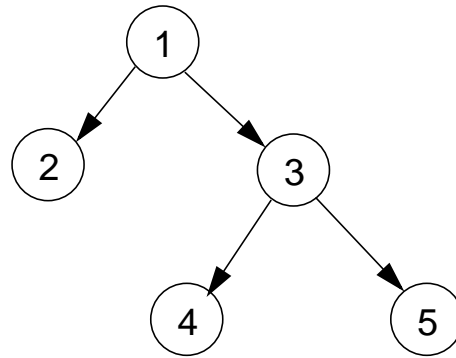
Ejecución paralela versus ejecución secuencial

Ejecución Secuencial



Ejecución Paralela

Tareas



Arcos: comunicación/sincronización entre tareas

Overhead del paralelismo:

$$\text{Tiempo paralelo} = \frac{\text{Tiempo secuencial}}{\text{Número de procesadores}} + \text{Overhead del paralelismo}$$

Causas del overhead del paralelismo:

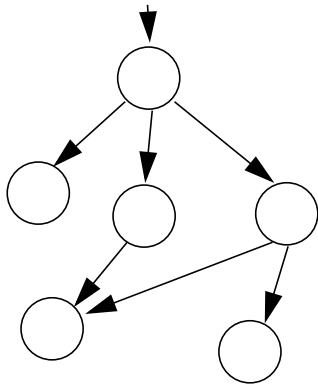
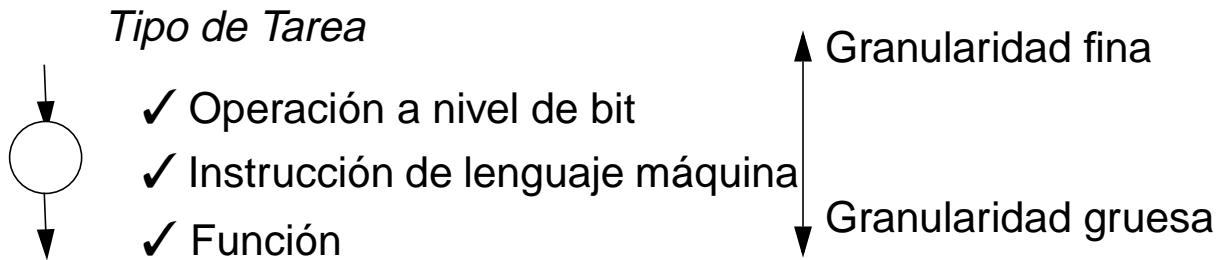
- ✓ Desequilibrio de carga
- ✓ Coste de comunicación/sincronización

Aspectos esenciales en la minimización del overhead:

- ✓ ¿Cómo realizar la descomposición en tareas?
- ✓ ¿Cuándo realizar la descomposición?
- ✓ ¿Cuándo y dónde ejecutar cada tarea?

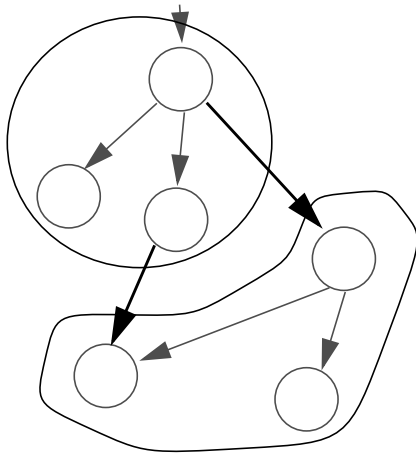
¿Cómo realizar la descomposición en tareas?

□ Granularidad de las Tareas



Granularidad fina

- ✓ Más comunicación entre tareas
- ✓ Mayor nivel de paralelismo



Granularidad gruesa

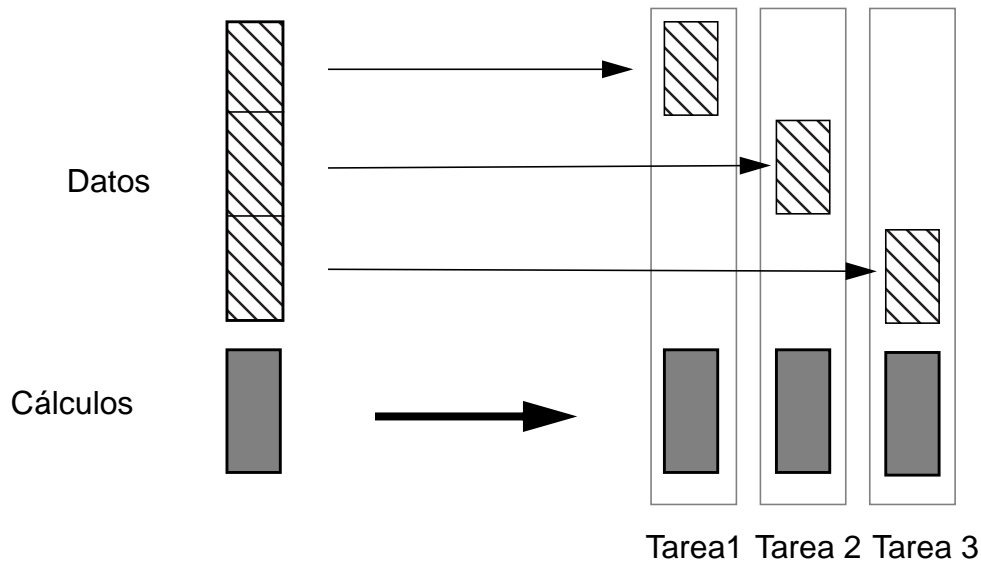
- ✓ Menos comunicación entre tareas
- ✓ Menor nivel de paralelismo

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Paralelismo de datos

Los datos se descomponen en bloques. Todas las tareas realizan los mismos cálculos, cada una con su propio bloque de datos.

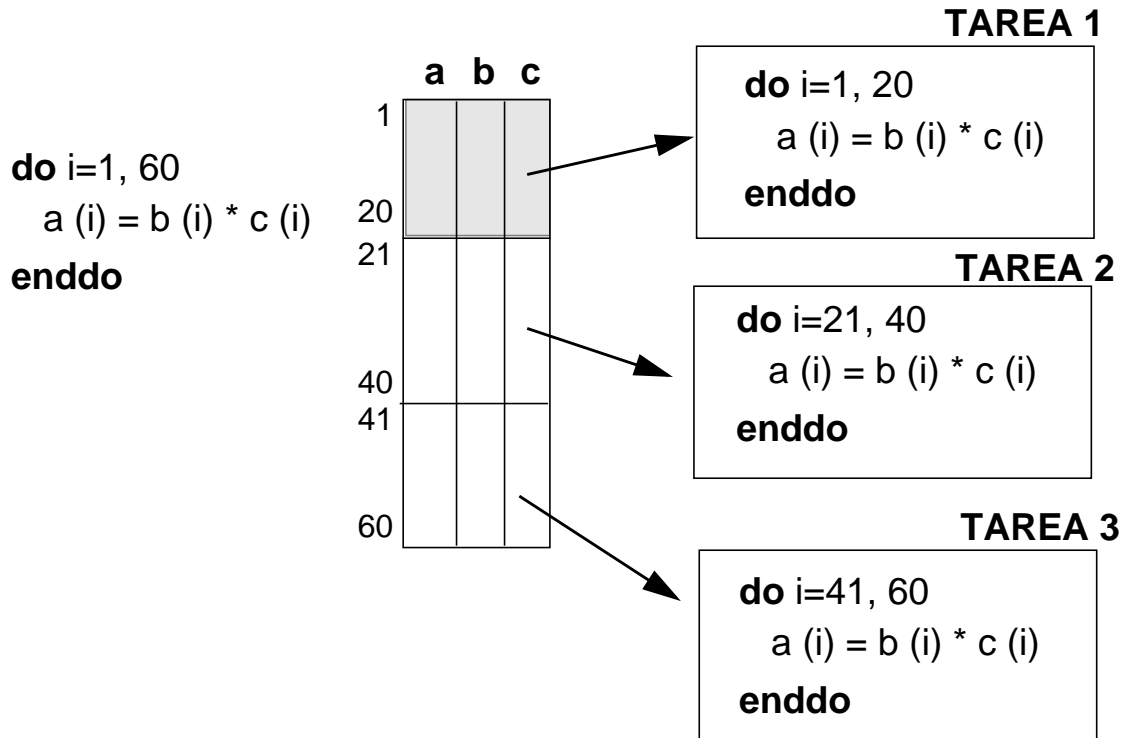


- ✓ El nivel de paralelismo es proporcional a la cantidad de datos
- ✓ Es adecuado para cálculo científico, en el que hay involucradas estructuras de datos grandes y regulares (vectores, matrices, etc.)

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Paralelismo de datos: Un ejemplo

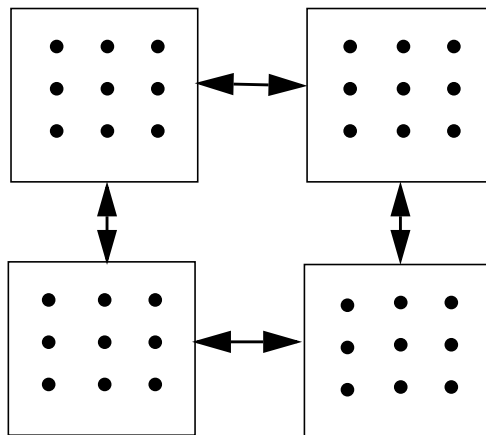
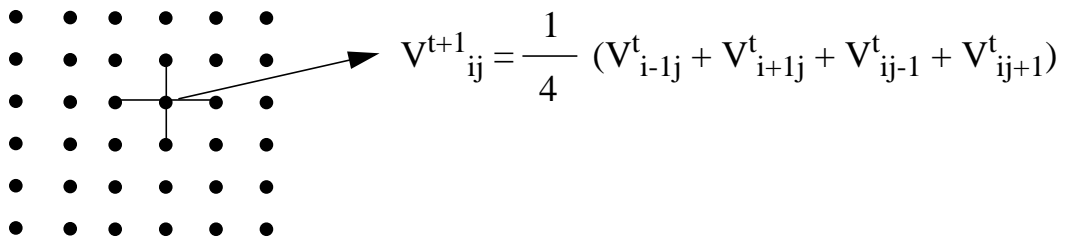


¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Paralelismo de datos: Otro ejemplo

Método iterativo de Jacobi para resolver la ecuación de Laplace



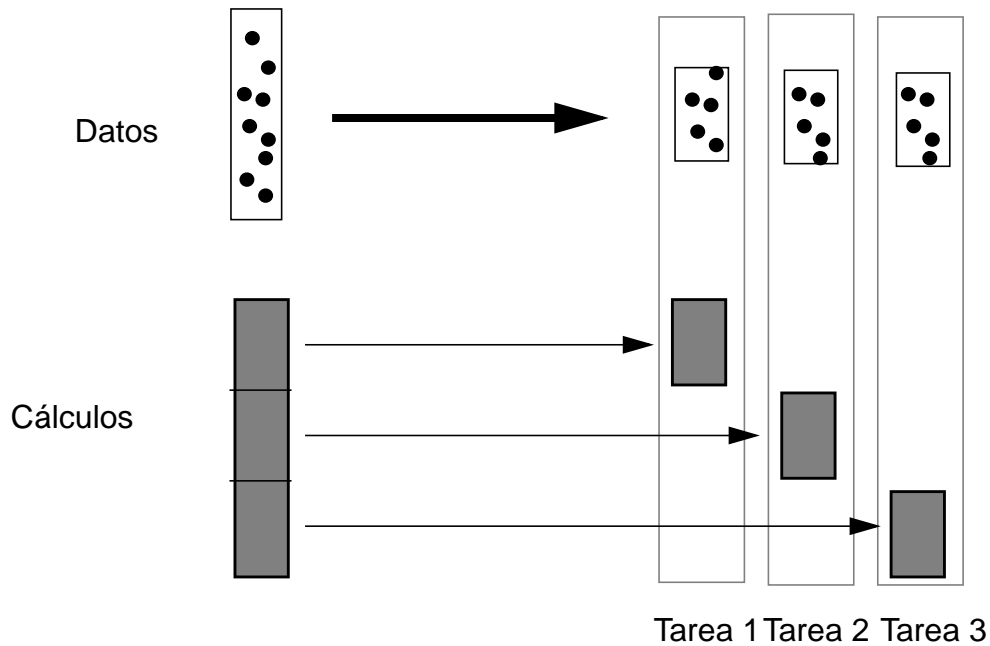
✓ Usualmente se obtienen grafos de tareas regulares

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Paralelismo de función

Los cálculos se descomponen en tareas (funciones). Cada tarea usa el subconjunto de los datos que necesita.

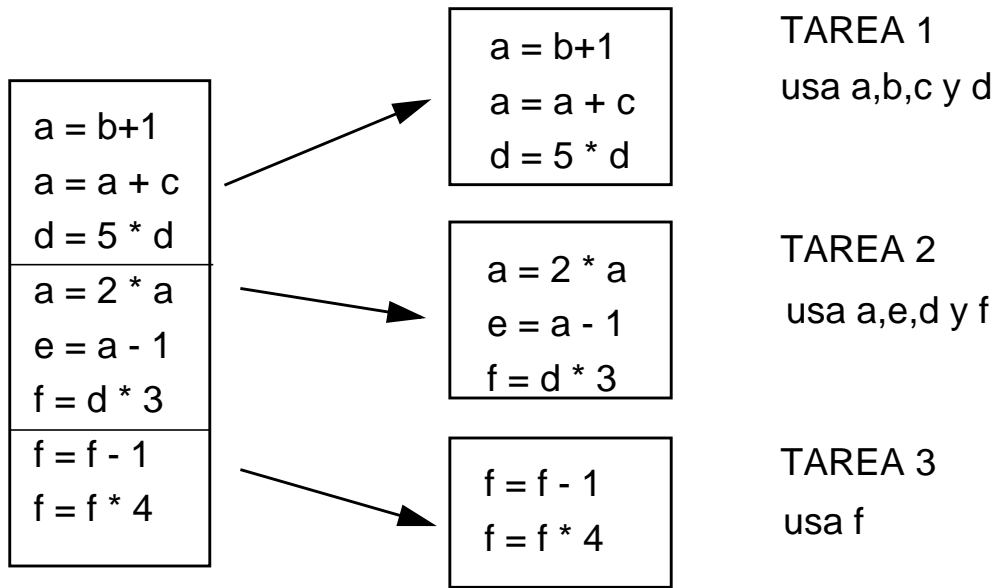


- ✓ El nivel de paralelismo es limitado
- ✓ Es adecuado para cálculo no regular

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Paralelismo de función: Un ejemplo



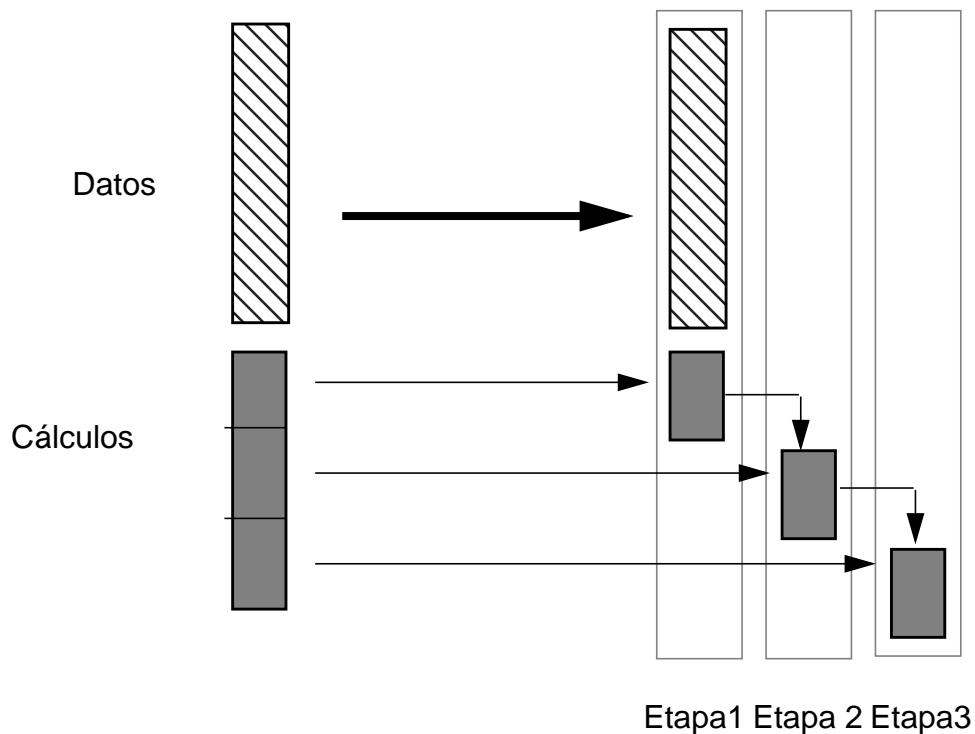
✓ Usualmente se obtienen grafos de tareas no regulares

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Segmentación

Una serie de cálculos deben realizarse sobre todos los elementos de una estructura de datos. Los cálculos se descomponen en tareas (etapas).

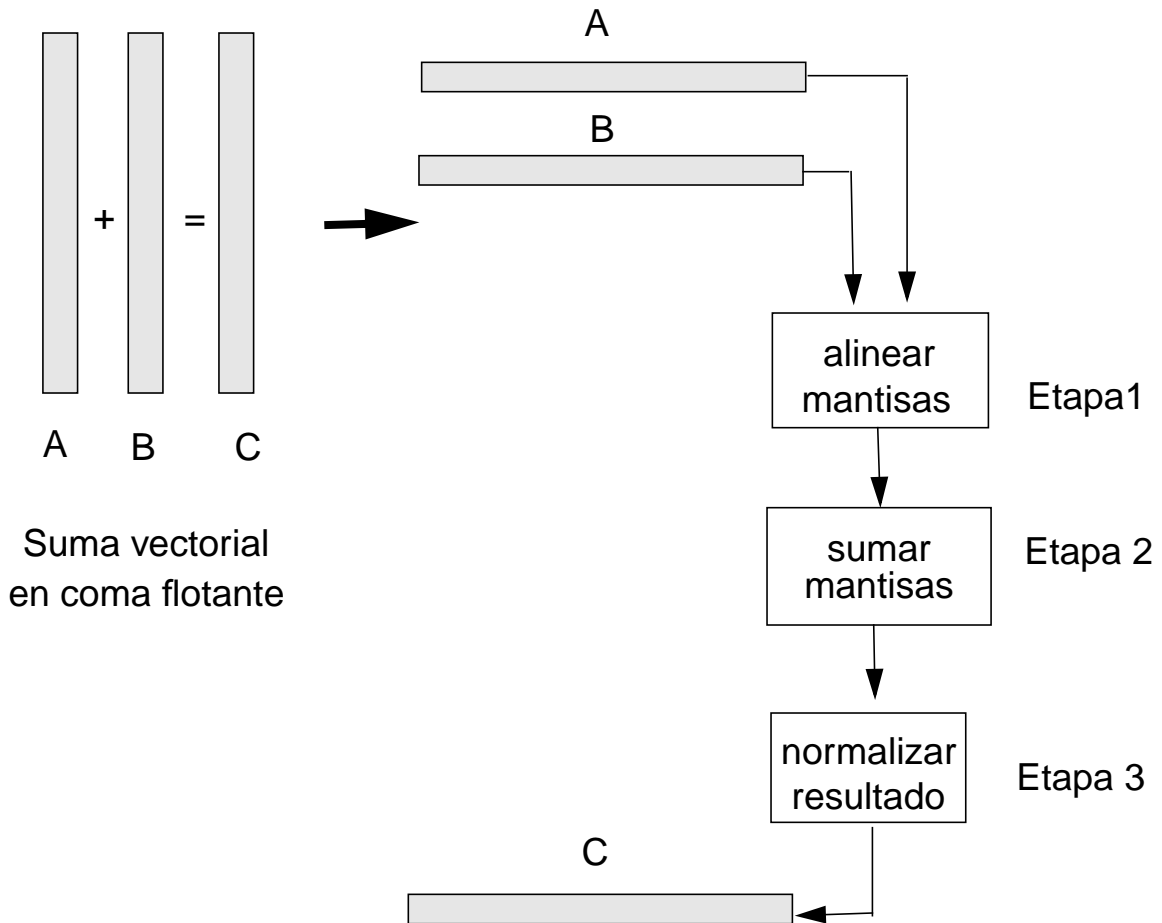


- ✓ El nivel de paralelismo es limitado
- ✓ Es adecuado para cálculo regular

¿Cómo realizar la descomposición en tareas?

□ Estilos de Paralelismo

Segmentación: Un ejemplo



¿Cuándo realizar la descomposición en tareas?

□ Descomposición estática

La descomposición es decidida por el programador o por el compilador.

doall i = 1, 60	cobegin
a (i) = b (i)*c (i)	a;
enddoall	b;
	c;
	coend

□ Descomposición dinámica

La descomposición se decide en tiempo de ejecución.

a (X,Y,Z) :- b (X,Z), c (Z,Y)

En tiempo de ejecución se decide si b y c se ejecutan en paralelo (en caso de que Z ya esté instanciado) o secuencialmente (si Z es todavía desconocido).



¿Cuándo y dónde se ejecutan las tareas?

Planificación

Planificación Estática

La asignación de cálculos a tareas se realiza en tiempo de COMPILACION.

No ocasiona sobrecarga en tiempo de ejecución.

Planificación Dinámica

La asignación de cálculos a tareas se realiza en tiempo de EJECUCION.

Mejor equilibrio de carga.



¿Cuándo y dónde se ejecutan las tareas?

Planificación

Ejemplo: Implementación de DOALL en un sistema con memoria compartida y con P procesadores

```
Doall i=1, N
  trabajo (i)
enddo
```

□ Planificación Estática

Se crean P tareas. La tarea q ejecuta el siguiente código:

```
Do local_i = (q-1)N/P+1, qN/P
  trabajo (local_i)
enddo
```

□ Planificación Dinámica

Se crean P tareas. La tarea q ejecuta el siguiente código:

```
local_i := i
i := i+1 Acceso Exclusivo
While local_i < N do
  trabajo (local_i)
  local_i := i
  i := i+1 Acceso Exclusivo
endwhile
```



Relación granularidad-planificación

En caso de planificación dinámica

Granularidad fina

- ✓ Mejor equilibrio de carga
- ✓ Menor localidad en los accesos a los datos (cociente entre el número de cálculos realizados y el número de datos usados)

Granularidad gruesa

- ✓ Peor equilibrio de carga
- ✓ Mayor localidad en los accesos a los datos



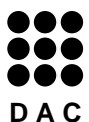
Impacto de las decisiones de diseño en las características del algoritmo

❑ Decisiones

- ✓ Asignación de cálculos y datos a procesadores
- ✓ Granularidad
- ✓ Ordenación local

❑ Características del algoritmo

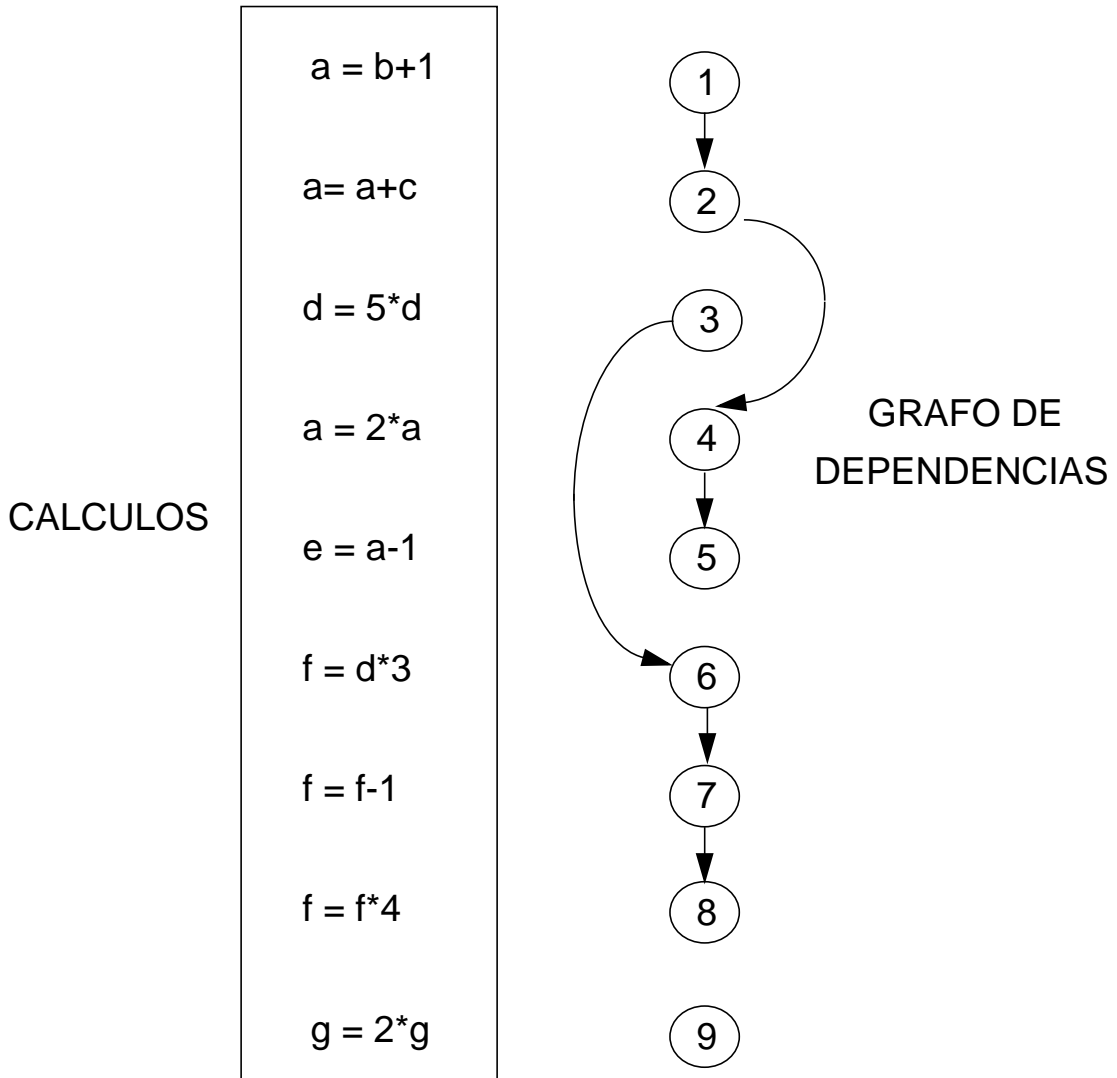
- ✓ Necesidades y topología de comunicación/sincronización
- ✓ Facilidad de codificación
- ✓ Equilibrio espacial de la carga
- ✓ Equilibrio temporal de la carga
- ✓ Localidad de los accesos a los datos



Un ejemplo sencillo

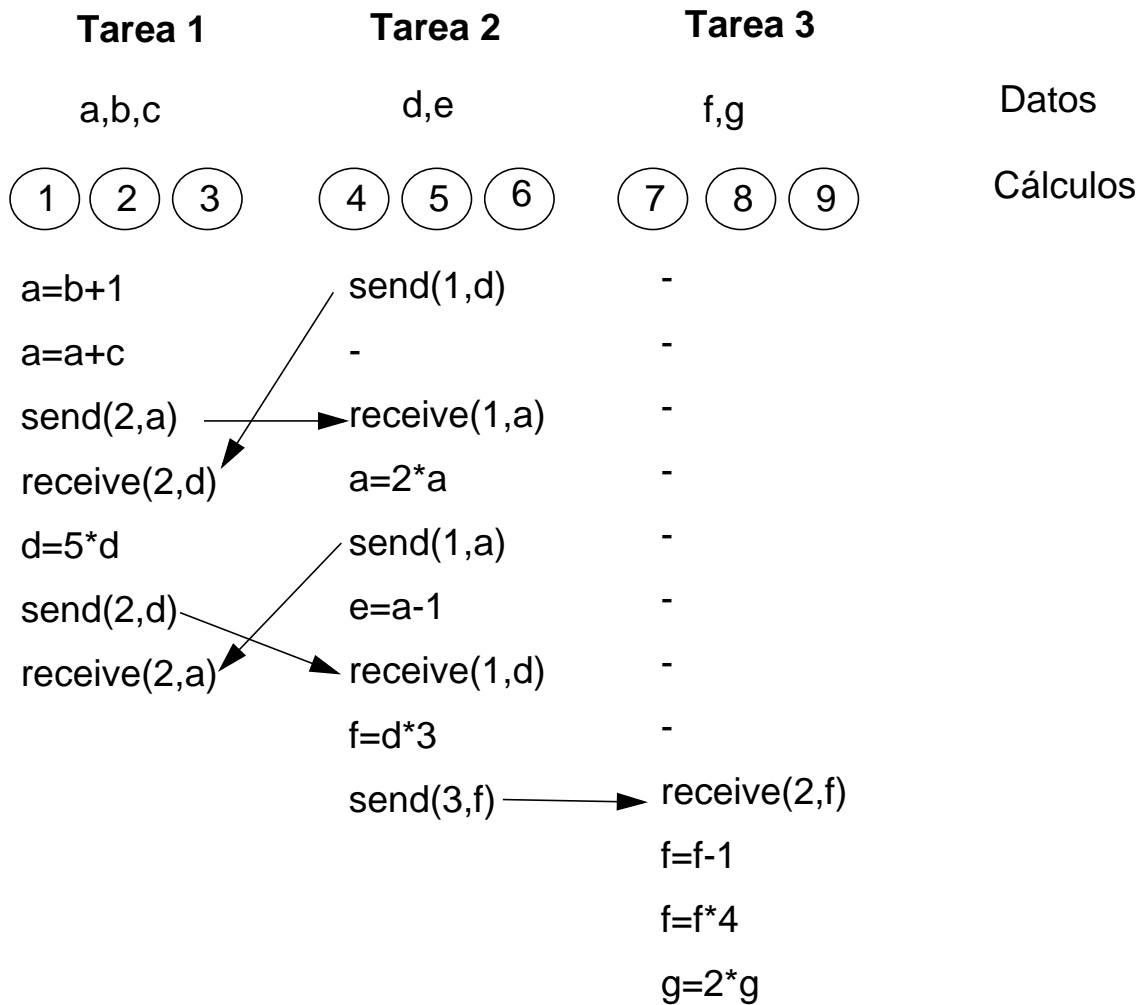
a	d	g
b	e	
c	f	

DATOS



Un ejemplo sencillo

Paralelismo de Función



La asignación de datos y cálculos a procesadores determina:

Equilibrio espacial: equilibrio en el número de operaciones asignadas a cada tarea

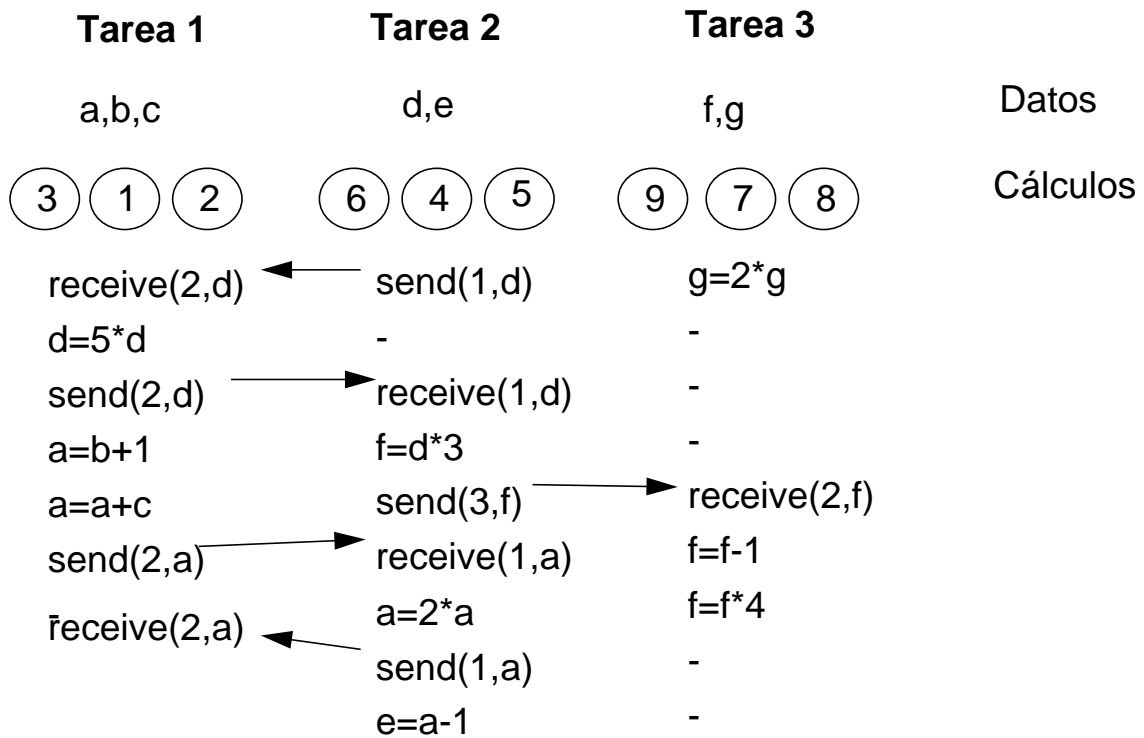
Necesidades de comunicación

Topología de comunicación



Un ejemplo sencillo

Un nuevo orden local



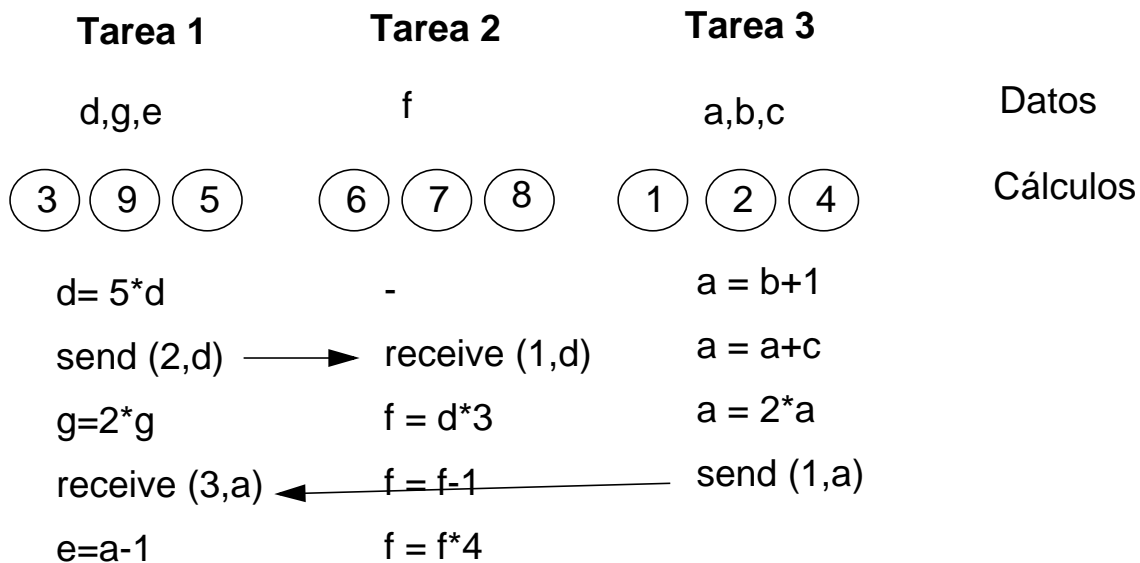
La ordenación local determina:

Equilibrio temporal: equilibrio en los periodos de inactividad forzada



Un ejemplo sencillo

Una nueva descomposición y orden local



Un ejemplo más complejo

□ Resolución de un sistema triangular de ecuaciones

$$\begin{bmatrix} a_{11} & & & \\ a_{21} & a_{22} & & \\ a_{31} & a_{32} & a_{33} & \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

$$x_1 = \frac{b_1}{a_{11}}$$

$$x_2 = \frac{b_2 - a_{21} * x_1}{a_{22}}$$

$$x_3 = \frac{b_3 - a_{31} * x_1 - a_{32} * x_2}{a_{33}}$$

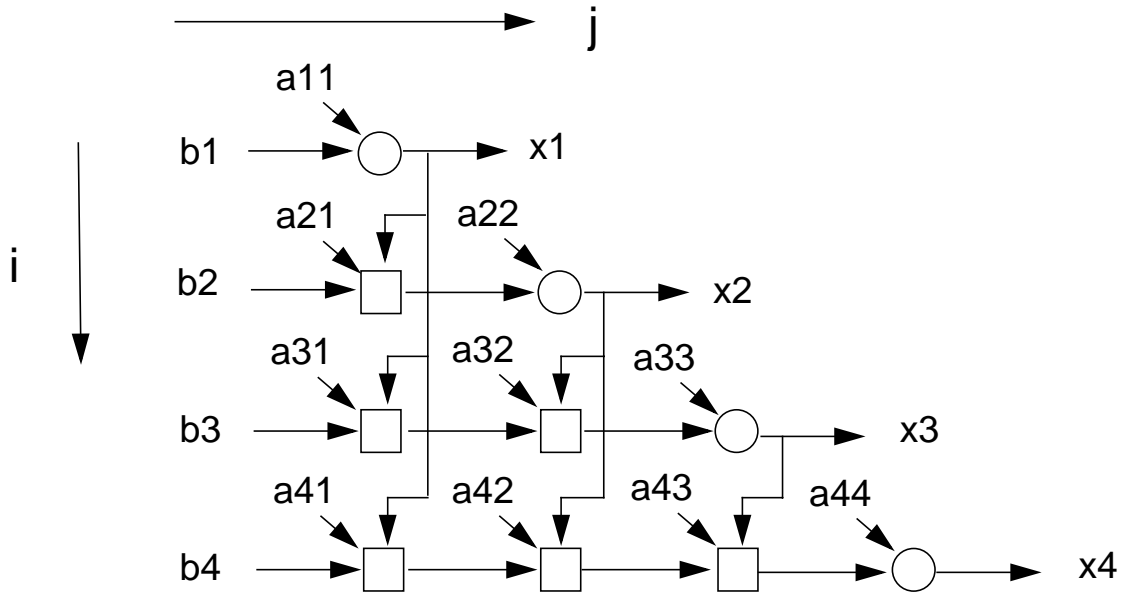
$$x_4 = \frac{b_4 - a_{41} * x_1 - a_{42} * x_2 - a_{43} * x_3}{a_{44}}$$

```
do i=1,N
  do j=1, i-1
    bi = bi - aij * xj
  enddo
  xi = bi / aii
enddo
```

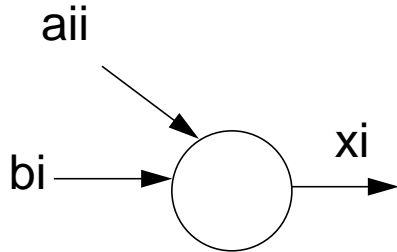
Un ejemplo más complejo

Resolución de un sistema triangular de ecuaciones

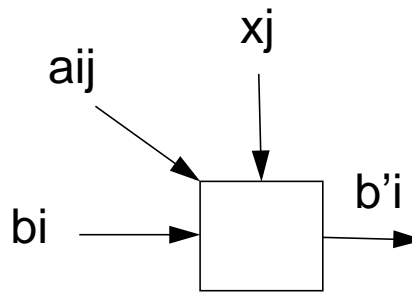
Grafo de Dependencias



Operaciones Elementales



$$x_i = \frac{b_i}{a_{ii}}$$



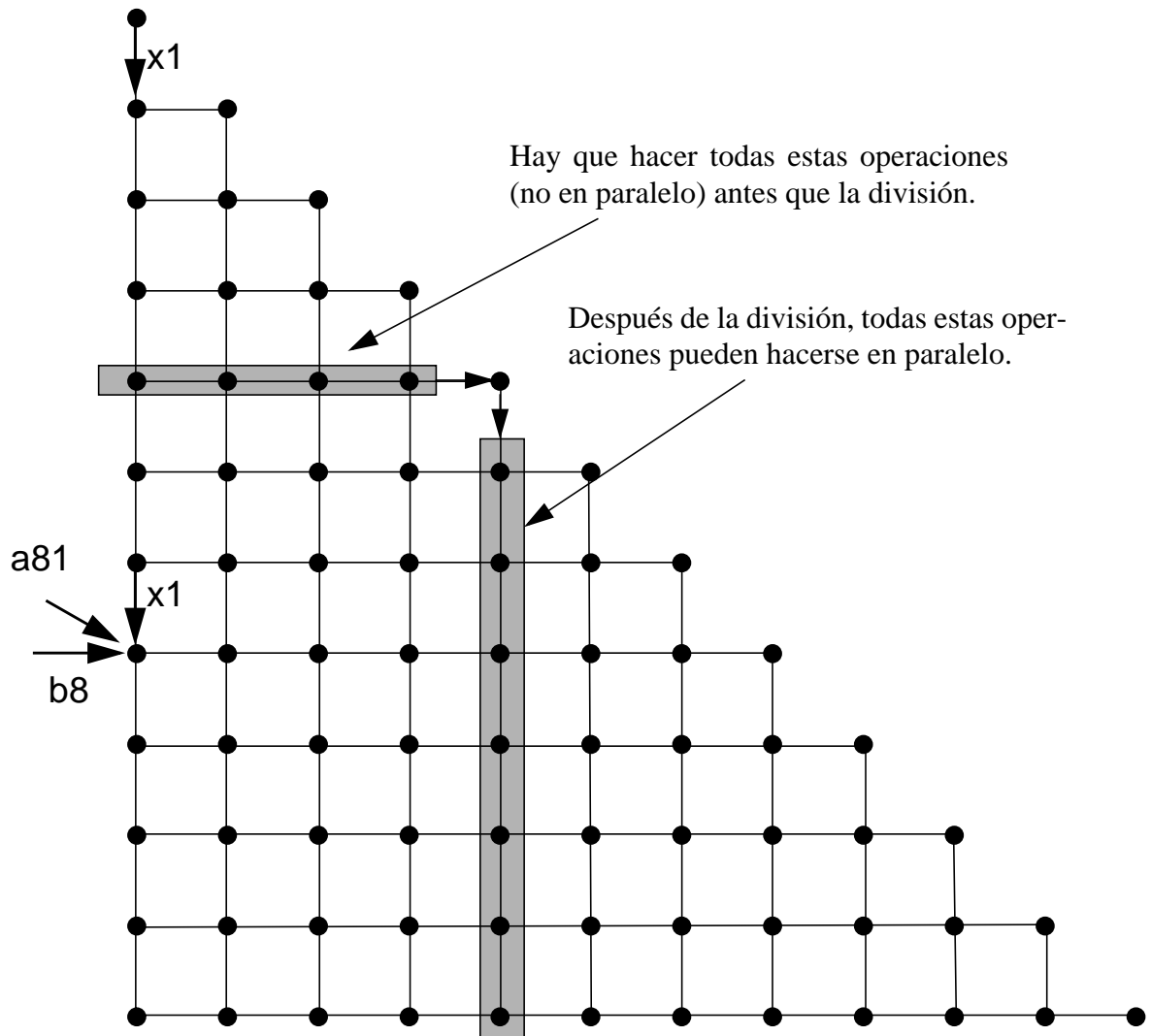
$$b'_i = b_i - a_{ij} * x_j$$



Un ejemplo más complejo

□ Resolución de un sistema triangular de ecuaciones

Grafo de Dependencias



Un ejemplo más complejo

□ Caracterización del Equilibrio de Carga (E_p)

Asumiendo un coste 0 de comunicación/sincronización:

$$E_p = \frac{T_1}{pT_p}$$

siendo:

T_1 el tiempo de ejecución en un procesador

T_p el tiempo de ejecución en p procesadores

E_p es un número entre 0 y 1. Cuanto más cercano a 1, mejor equilibrio de carga.

□ Caracterización de la Localidad (L)

$$L = \frac{1}{n} \sum_{i=1}^n \frac{O_i}{D_i}$$

siendo:

n el número de tareas

O_i el número de operaciones elementales (sumas, productos) realizadas en la tarea i

D_i número de datos involucrados en la tarea i



Un ejemplo más complejo

□ Caracterización del Coste de Sincronización (S)

Consideramos el coste de la sincronización en la ejecución de un bucle *DOALL* $i=1, N$, usando p procesadores, con un modelo de planificación dinámica:

$$S = pT_c + \frac{N}{p}T_s + T_j$$

siendo:

T_c el coste de creación de una tarea

T_s el coste de sincronización para que cada procesador obtenga la siguiente tarea

T_j el coste de la operación de sincronización al acabar todas las tareas

□ Caracterización del Coste de Comunicación (C)

C será el número total de mensajes del algoritmo

□ Caracterización del Tiempo de Cálculo (T_o)

T_o el coste de una operación elemental



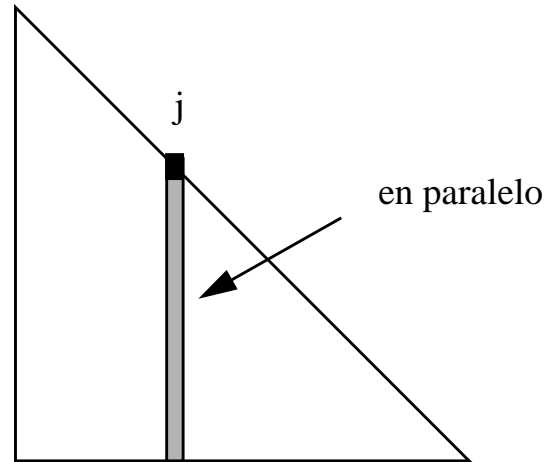
Un ejemplo más complejo

Algoritmos para un modelo de variables compartidas

ALGORITMO 1

```

do j=1,N
  xj = bj/aj,j
  doall i = j+1, N
    bi = bi-ai,j*xj
  enddo
enddo
    
```



1												
2	6											
3	7	11										
4	8	12	15									
5	9	13	16	19								
2	10	14	17	20	23							
3	7	12	18	21	24	26						
4	8	13	16	22	25	27	29					
5	9	14	17	20	24	28	30	32				
2	7	12	18	21	25	27	31	33	34			
3	8	13	16	20	24	28	30	33	35	36		
4	9	14	17	21	25	27	30	33	35	37	38	

$N=12$

$p=3$

$E_p=0.68$

$$T_p = \sum_{i=1}^N \left(1 + \left\lceil \frac{N-i}{p} \right\rceil \right) T_o \approx$$

$$\approx N \left(1 + \frac{N-1}{2p} \right) T_o \quad \text{si } N \gg 0$$

$$E_p = \frac{N(N+1)}{2pN \left(1 + \frac{N-1}{2p} \right)} \rightarrow 1 \quad \text{si } N \rightarrow \infty$$



Un ejemplo más complejo

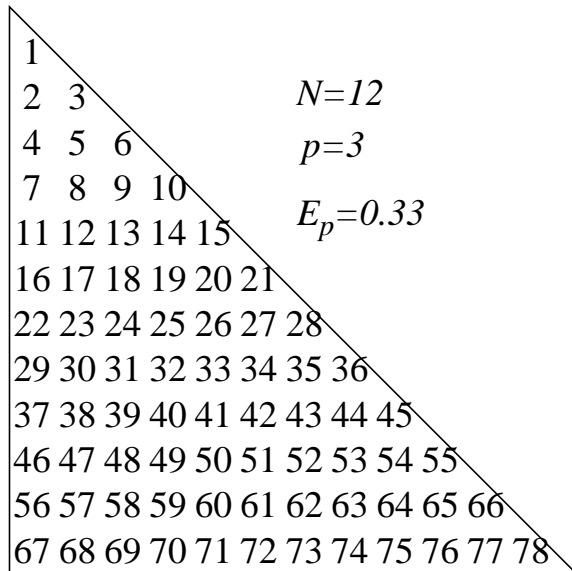
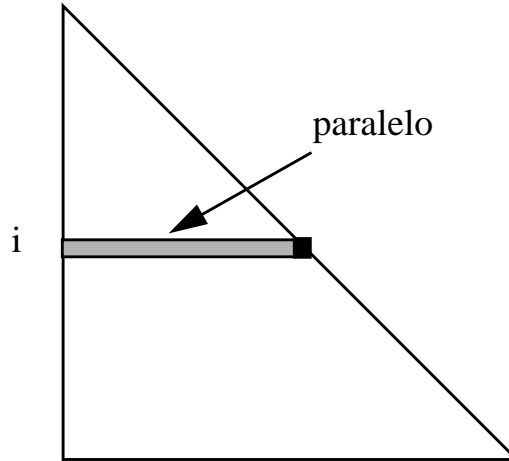
□ Algoritmos para un modelo de variables compartidas

ALGORITMO 2

```

signal (s)
do i=1,N
  doall j = 1, i-1
    wait (s)
     $bi = bi - ai,j * xj$ 
    signal (s)
  enddo
   $xi = bi / ai,i$ 
enddo

```



$$T_p = \frac{N(N+1)}{2} T_o$$

$$E_p = \frac{1}{p}$$



Un ejemplo más complejo

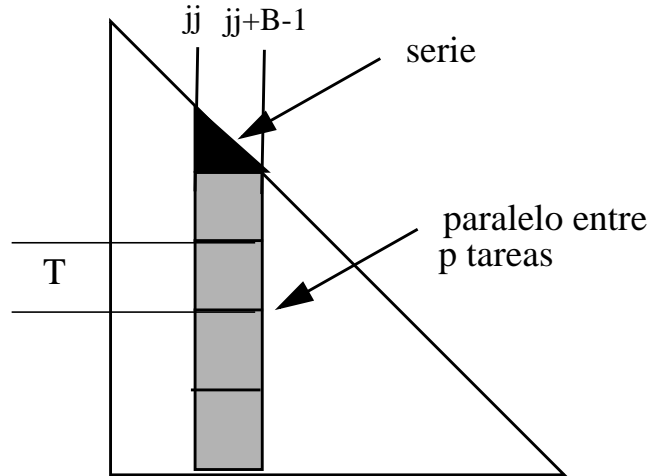
□ Algoritmos para un modelo de variables compartidas

ALGORITMO 3

```

do jj=1,N,B
  do j=jj, jj+B-1
    xj = bj/aj,j
    do i=j+1, jj+B-1
      bi = bi - ai,j*xj
    enddo
  enddo
  T = (N-(jj+B)+1)/p
  doall ii= jj+B,N,T
    do j=jj, jj+B-1
      do i=ii, ii+T-1
        bi = bi-ai,j*xj
      enddo
    enddo
  enddo
enddo

```



1												$N=12$
2	3											$p=3$
4	5	6										$B=3$
7	8	9	16									$E_p=0.62$
10	11	12	17	18								
13	14	15	19	20	21							
7	8	9	22	23	24	28						
10	11	12	25	26	27	29	30					
13	14	15	22	23	24	31	32	33				
7	8	9	25	26	27	34	35	36	37			
10	11	12	22	23	24	34	35	36	38	39		
13	14	15	25	26	27	34	35	36	40	41	42	

$$T_p = \sum_{i=1}^{N/B} \left(\frac{B^2 + B}{2} + \left\lceil \frac{N - iB}{p} \right\rceil B \right) T_o \approx \frac{N}{2} \left(B + 1 + \frac{N - B}{p} \right) T_o \quad \text{si } N \gg 0$$

$$E_p = \frac{N + 1}{B(p - 1) + p + N} \rightarrow 1 \text{ si } N \rightarrow \infty$$



Un ejemplo más complejo

- Algoritmos para un modelo de variables compartidas

Localidad en el algoritmo 3

Coste de sincronización para al algoritmo 3

$$C = \sum_{i=1}^{N/B} (pT_c + T_s + T_j) = \frac{N}{B}(pT_c + T_s + T_j)$$



Un ejemplo más complejo

□ Algoritmos para un modelo de variables compartidas

Cálculo del nivel óptimo de granularidad (B_{opt})

Ignoramos efecto de la jerarquía de memorias en el tiempo de cálculo

$$T = \frac{N(N + p + B(p - 1))}{2p} T_o + \frac{N}{B} (pT_c + T_s + T_j)$$

$$T'(B) = \left(\frac{N}{2} - \frac{N}{2p} \right) T_o - \frac{N}{B^2} (pT_c + T_s + T_j)$$

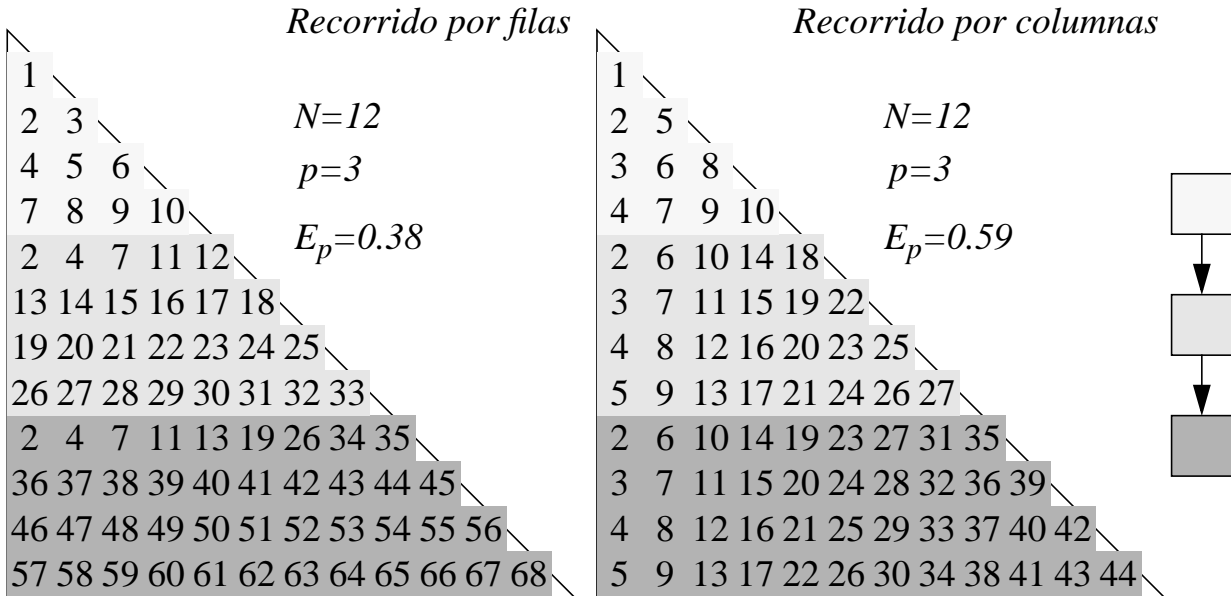
$$T'(B) = 0 \Rightarrow B = \sqrt{\frac{2p(pT_c + T_s + T_j)}{(p - 1)T_o}} = B_{opt}$$

Un ejemplo más complejo

❑ Algoritmos para un modelo de paso de mensajes

ALGORITMO 1

Asignación de datos y cálculos: Bloques de filas

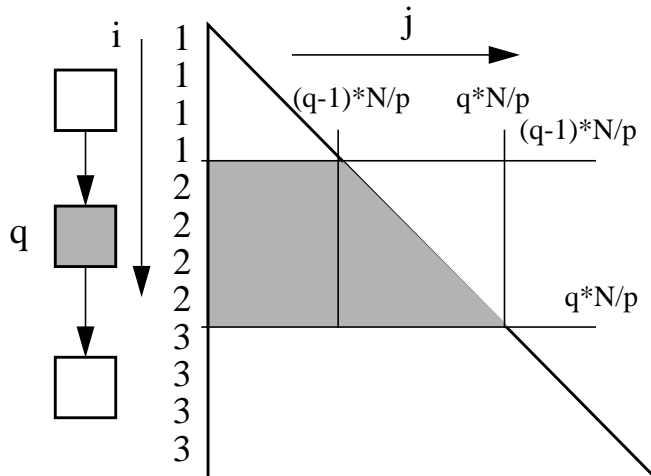


Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

ALGORITMO 1

Código recorrido por columnas



Código del procesador q ($q=1,\dots,p$)

```

do j=1, (q-1)*N/p
  receive (q-1, xj)
  if q < p then send (q+1,xj)
  do i= (q-1)*N/p + 1, q*N/p
    bi = bi - ai,j*xj
  enddo
enddo
do j=(q-1)*N/p+1, q*N/p
  xj = bj/aj,j
  if q < p then send (q+1,xj)
  do i= j + 1, q*N/p
    bi = bi - ai,j*xj
  enddo
enddo

```

Código del procesador q ($q=1,\dots,p$)
(accesos locales)

```

do j=1, (q-1)*N/p
  receive (q-1, xj)
  if q < p then send (q+1,xj)
  ii = 1
  do i= (q-1)*N/p + 1, q*N/p
    bii = bii - aii,j*xj
    ii = ii+1
  enddo
enddo
jj = 1
do j=(q-1)*N/p+1, q*N/p
  xj = bj,j/aj,j
  if q < p then send (q+1,xj)
  ii = jj+1
  do i= j + 1, q*N/p
    bii = bii - aii,j*xj
    ii = ii+1
  enddo
  jj = jj+1
enddo

```

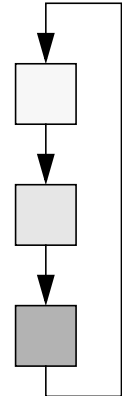
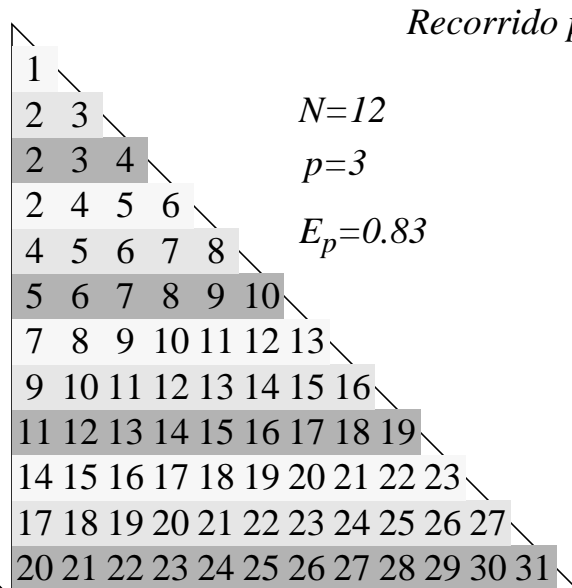
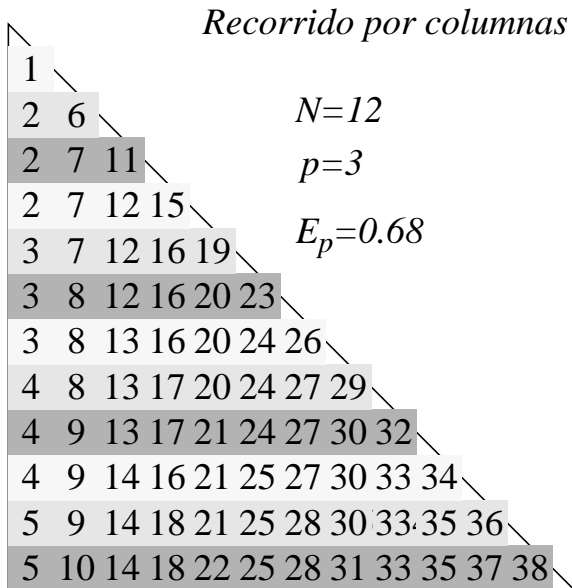


Un ejemplo más complejo

❑ Algoritmos para un modelo de paso de mensajes

ALGORITMO 2

Asignación de datos y cálculos: Cíclico por filas

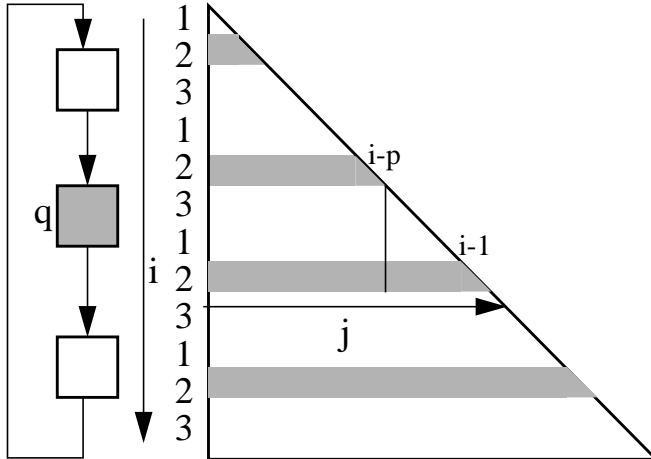


Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

ALGORITMO 2

Código recorrido por filas



Código del procesador q ($q=1,\dots,p$)

```

do i=q, N, p
  do j=1, i-p
    bi = bi - ai,j*xj
  enddo
  do j=i-p+1, i-1
    receive ((q-1) mod p, xj)
    if j <> i-p + 1
      then send ((q+1) mod p, xj)
    endif
    bi = bi - ai,j*xj
  enddo
  xi = bi/ai,i
  if i < N then send ((q+1) mod p, xi)
endif
enddo

```

*Código del procesador q ($q=1,\dots,p$)
(accesos locales)*

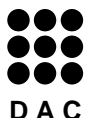
```

ii = 1
do i=q, N, p
  do j=1, i-p
    bii = bii - aii,j*xj
  enddo
  do j=i-p+1, i-1
    receive ((q-1) mod p, xj)
    if j <> i-p + 1
      then send ((q+1) mod p, xj)
    endif
    bii = bii - aii,j*xj
  enddo
  xi = bii/aii,i
  if i < N then send ((q+1) mod p, xi)
endif
  ii = ii+1
enddo

```



CEPBA



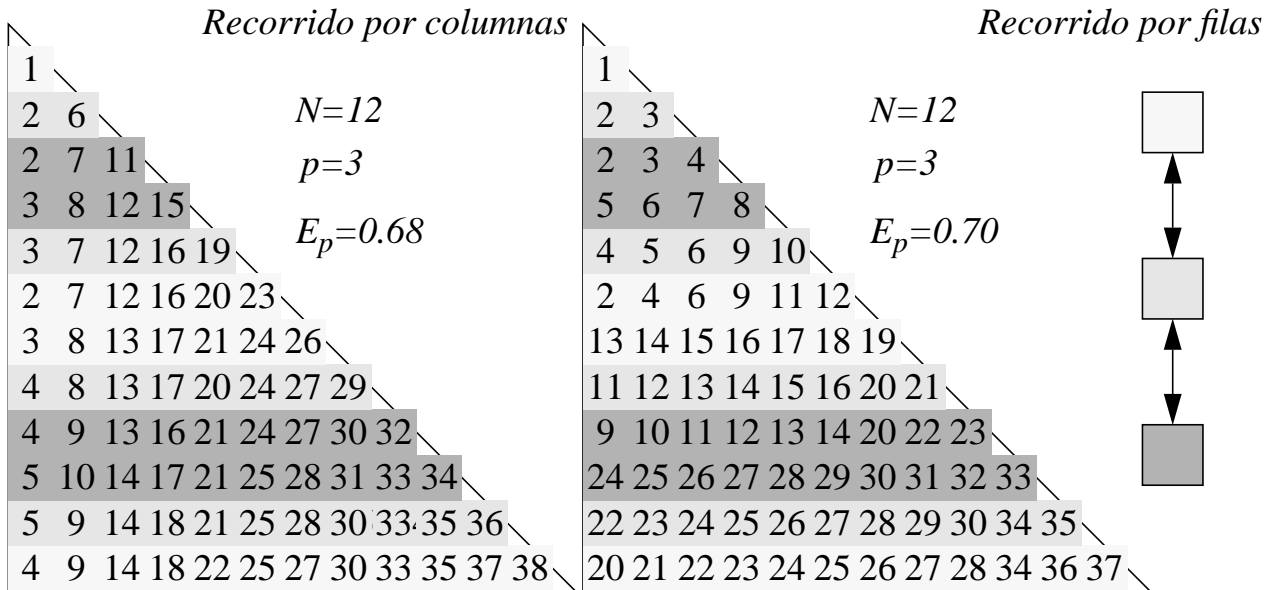
D A C

Un ejemplo más complejo

❑ Algoritmos para un modelo de paso de mensajes

ALGORITMO 3

Asignación de datos y cálculos: Plegado por filas



Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

Análisis del equilibrio de carga (E_p):

Algoritmo 1 (Bloques de filas, recorrido por columnas)

$$E_p = \frac{(N+1)/2}{N\left(1 - \frac{1}{2p}\right) + \frac{p}{N}(p-1) + \frac{1}{2}} \rightarrow \frac{p}{2p-1}$$

Algoritmo 2 (Ciclico por filas, recorrido por filas)

$$E_p = \frac{(N^2 + N)/2}{p\left(\frac{N^2}{2p} + \frac{N}{2} + p - 1\right)} \rightarrow 1$$

Algoritmo 3 (Plegado por filas, recorrido por filas)

$$E_p = \frac{(N+1)/2}{\frac{3p}{2} + \frac{N}{2} - 1} \rightarrow 1$$

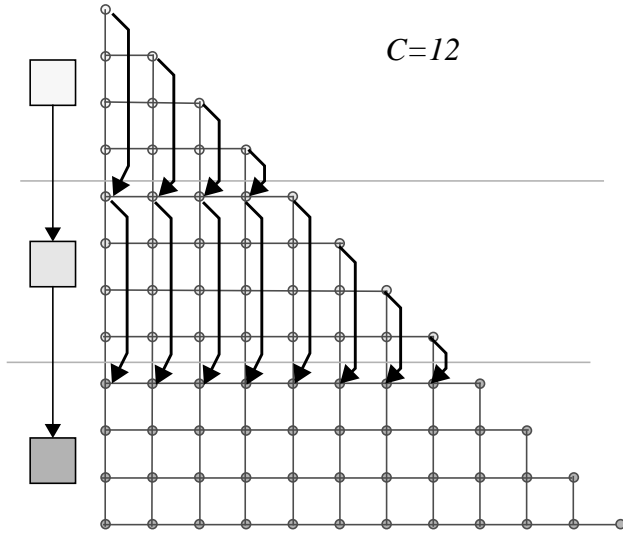


Un ejemplo más complejo

❑ Algoritmos para un modelo de paso de mensajes

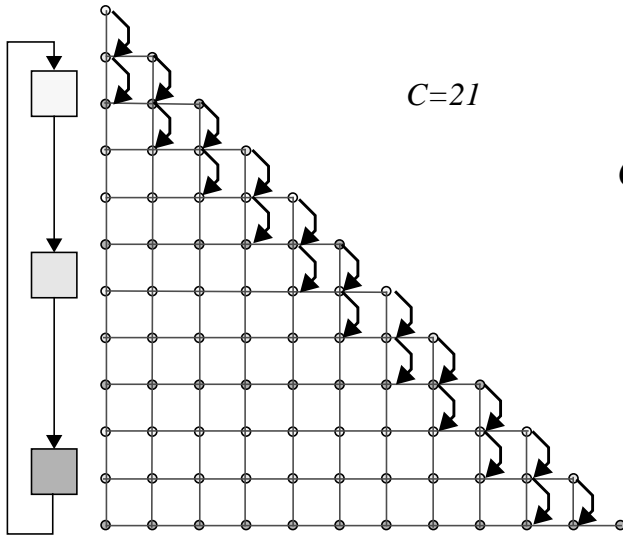
Análisis del coste de comunicación (C):

Algoritmo 1 (Bloques de filas, recorrido por columnas)



$$C = \sum_{i=1}^{p-1} i \frac{N}{p} = \frac{N}{2}(p-1)$$

Algoritmo 2 (Cíclico por filas, recorrido por filas)



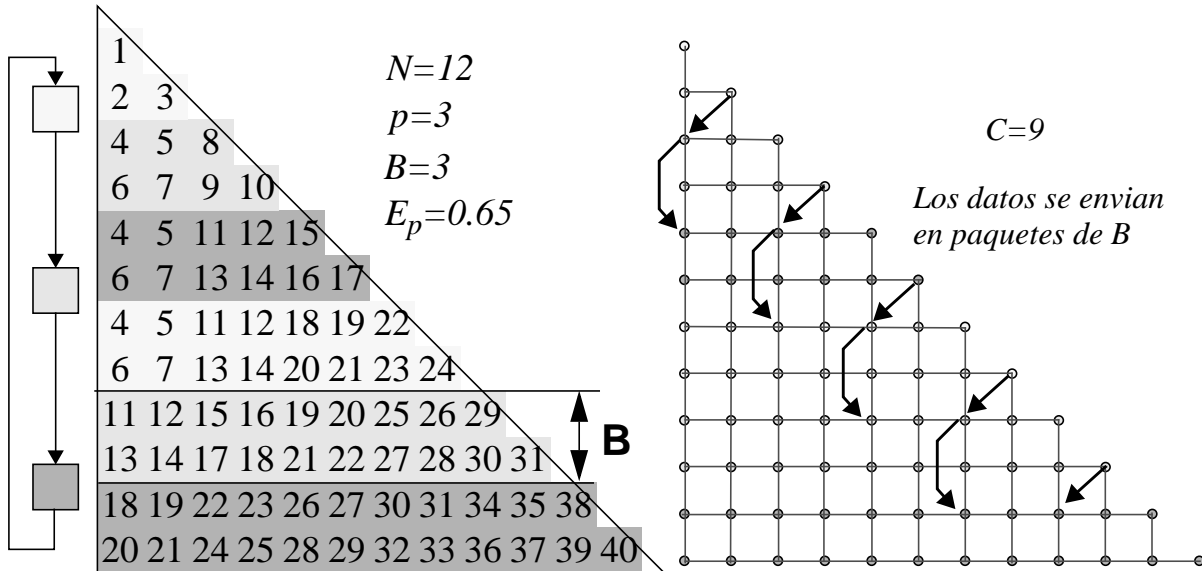
$$C = (N - p + 1)(p - 1) \sum_{i=1}^{p-2} i = N(p - 1) - \frac{p^2 - p}{2}$$

Un ejemplo más complejo

❑ Algoritmos para un modelo de paso de mensajes

ALGORITMO 4

Asignación de datos y cálculos: Cíclico por bloques de B filas

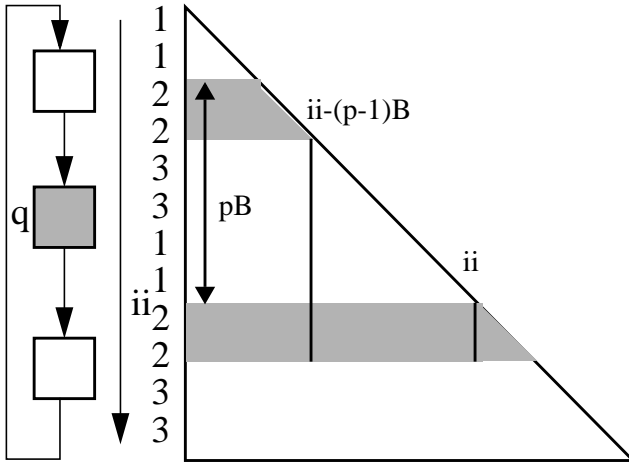


Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

ALGORITMO 4

Código



Código del procesador q (q=1,...,p)

```

do ii = (q-1)B + 1, N, p*B
  do j = 1, ii - (p-1)B - 1
    do i = ii, ii+B - 1
      bi = bi - aij * xj
    enddo
  do jj = ii - (p-1)B, ii-B, B
    receive ((q-1) mod p, xjj : jj + B - 1)
    if jj <> ii - (p-1)B
      then send ((q+1) mod p, xjj : jj+B-1)
    endif
    do j = jj, jj+B-1
      do i = ii, ii+B-1
        bi = bi - aij*xj
      enddo
    enddo
  do j = ii, ii+B-1
    xj = bj/ajj
    do i = j+1, ii+B-1
      bi = bi + aij*xj
    enddo
  enddo
  if ii < N-B+1
    then send ((q+1) mod p, xii : ii+B-1)
  endif
enddo

```

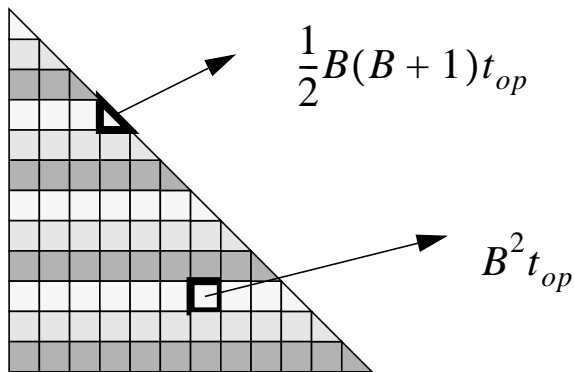
Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

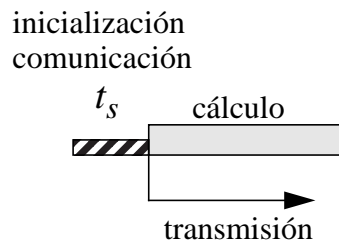
ALGORITMO 4

Cálculo del valor óptimo de B

Coste de las operaciones aritméticas



Comunicaciones asíncronas con solapamiento cálculo/ comunicación



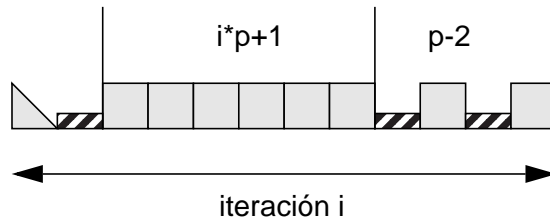
Un ejemplo más complejo

□ Algoritmos para un modelo de paso de mensajes

ALGORITMO 4

Cálculo del valor óptimo de B

El nodo p hace $\bar{N} = N/(pB)$ iteraciones. En la iteración i hace:



$$T(B) = \sum_{i=1}^{\bar{N}} \left[\frac{B(B+1)t_{op}}{2} + (ip+1)B^2t_{op} + (p-2)(t_s + B^2t_{op}) \right] =$$

$$XB + \frac{Y}{B} + Z$$

siendo:

$$X = \frac{(3p-1)Nt_{op}}{2p} \quad Y = \frac{(p-2)Nt_s}{p} \quad Z = \frac{N(N+1)t_{op}}{2p}$$

$$T'(B) = X - \frac{Y}{B^2}$$

$$T'(B) = 0 \Rightarrow B = \sqrt{\frac{Y}{X}} = \sqrt{\frac{(2p-4)t_s}{(3p-1)t_{op}}}$$



Un ejemplo más complejo

□ Programación en HPF

```
Real*8          a(N,N), x(N), b(N)
TEMPLATE        t(N)
ALIGN           a(i,j), t(i)
ALIGN           b(i), t(i)
DISTRIBUTE (CYCLC) t(N)
```

```
do j=1,N
  xj = bj/aj,j
  doall i = j+1, N
    bi = bi-ai,j*xj
  enddo
enddo
```

