Capítulo 3

Métodos bidireccionales

Se considera de nuevo el problema de resolver el sistema

$$A\boldsymbol{x} = \boldsymbol{d}, \tag{3.1}$$

donde A es la matriz tridiagonal dada por (3.2), estrictamente diagonal dominante e irreducible y

$$\boldsymbol{d} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

es el vector de términos independientes.

3.1 Factorización LDU de matrices tridiagonales

Una matriz tridiagonal A de tamaño $n \times n$,

$$A = \begin{bmatrix} a_1 & b_1 & & & \\ c_2 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & c_{n-1} & a_{n-1} & b_{n-1} \\ & & & c_n & a_n \end{bmatrix}, \tag{3.2}$$

se puede descomponer como producto de tres matrices

$$A = LDU \tag{3.3}$$

donde D es una matriz diagonal y las matrices L y U son bidiagonales con la característica particular de que su diagonal principal está formada por unos; más explícitamente

$$A = \begin{bmatrix} 1 & & & & \\ l_2 & 1 & & & \\ & \ddots & \ddots & & \\ & & l_{n-1} & 1 & \\ & & & l_n & 1 \end{bmatrix} \begin{bmatrix} e_1 & & & & \\ & e_2 & & & \\ & & \ddots & & \\ & & & e_{n-1} & \\ & & & & e_n \end{bmatrix} \begin{bmatrix} 1 & u_1 & & & \\ & 1 & u_2 & & \\ & & \ddots & \ddots & \\ & & & 1 & u_{n-1} \\ & & & & 1 \end{bmatrix},$$

véase Golub y Van Loan [50].

Una vez obtenida la factorización (3.3) se puede resolver el sistema de ecuaciones lineales (3.1) en dos pasos: en el primero, se resuelve el sistema bidiagonal $LD\boldsymbol{z} = \boldsymbol{d}$, obteniéndose así el vector auxiliar $\boldsymbol{z}$; en el segundo, se obtiene la solución final del sistema (3.1) resolviendo el sistema bidiagonal $U\boldsymbol{x} = \boldsymbol{z}$.

El producto LDU viene dado por

$$LDU = \begin{bmatrix} e_1 & u_1e_1 & & & & \\ l_2e_1 & u_1l_2e_1+e_2 & u_2e_2 & & & \\ & l_3e_2 & u_2l_3e_2+e_3 & u_3e_3 & & \\ & \ddots & \ddots & \ddots & & \\ & & l_{n-1}e_{n-2} & u_{n-2}l_{n-1}e_{n-2}+e_{n-1} & u_{n-1}e_{n-1} & \\ & & & l_ne_{n-1} & u_{n-1}l_ne_{n-1}+e_n \end{bmatrix},$$

relacionando los elementos de A con los del producto LDU se tiene

$$a_1 = e_1,$$

y para $i = 2, 3, \ldots, n$,

$$c_i = l_ie_{i-1}, \tag{3.4}$$

$$a_i = u_{i-1}l_ie_{i-1} + e_i, \tag{3.5}$$

$$b_{i-1} = u_{i-1}e_{i-1}. \tag{3.6}$$

De las expresiones (3.4) y (3.6) se tiene

$$l_i = \frac{c_i}{e_{i-1}}, \qquad u_{i-1} = \frac{b_{i-1}}{e_{i-1}}, \quad \text{para} \quad i = 2, 3, \ldots, n, \tag{3.7}$$

y sustituyendo la expresión (3.7) en la expresión (3.5) se obtiene

$$e_i = a_i - \frac{b_{i-1}}{e_{i-1}}\frac{c_i}{e_{i-1}}e_{i-1}, \quad \text{para} \quad i = 2, 3, \ldots, n.$$

Por tanto, los elementos de la diagonal principal de D pueden obtenerse de forma sencilla mediante la siguiente relación de recurrencia

$$\begin{aligned} e_1 &= a_1, \\ e_i &= a_i - c_i\frac{b_{i-1}}{e_{i-1}}, \quad \text{para} \quad i = 2, 3, \ldots, n, \end{aligned}$$

y los de las matrices L y U mediante las recurrencias (3.7).

3.2 Método bidireccional para dos procesadores

3.2.1 Descripción del método

Aunque a primera vista la factorización (3.3) parece de naturaleza secuencial, es posible encontrar un paralelismo de grado dos en los cálculos (véase Ortega [91]), es decir, se pueden utilizar dos procesadores para obtener una factorización no exactamente igual que la anterior, pero sí equivalente en cuanto a complejidad. La idea original aparece en la eliminación Gaussiana bidireccional (*two-sided Gaussian elimination*) introducida por Babuska [4].

Para paralelizar la factorización (3.3), se supone además que $n = 2q$ para algún $q \geq 1$. Se considera la matriz de coeficientes particionada en bloques del siguiente modo

$$A = \left[\begin{array}{ccccc|ccccc} a_1 & b_1 & & & & & & & & \\ c_2 & a_2 & b_2 & & & & & & & \\ & \ddots & \ddots & \ddots & & & & & & \\ & & c_q & a_q & & b_q & & & & \\ \hline & & & c_{q+1} & & a_{q+1} & b_{q+1} & & & \\ & & & & & \ddots & \ddots & \ddots & & \\ & & & & & & c_{n-1} & a_{n-1} & b_{n-1} & \\ & & & & & & & c_n & a_n & \end{array}\right].$$

Sean las matrices

$$M = \left[\begin{array}{ccccc|ccccc} 1 & & & & & & & & & \\ l_2 & 1 & & & & & & & & \\ & l_3 & 1 & & & & & & & \\ & & \ddots & \ddots & & & & & & \\ & & & l_q & 1 & & & & & \\ \hline & & & & l_{q+1} & 1 & u_{q+1} & & & \\ & & & & & & 1 & u_{q+2} & & \\ & & & & & & & 1 & \ddots & \\ & & & & & & & & \ddots & u_{n-1} \\ & & & & & & & & & 1 \end{array}\right],$$

$$D = \left[\begin{array}{ccccc|ccccc} e_1 & & & & & & & & & \\ & e_2 & & & & & & & & \\ & & e_3 & & & & & & & \\ & & & \ddots & & & & & & \\ & & & & e_q & & & & & \\ \hline & & & & & e_{q+1} & & & & \\ & & & & & & e_{q+2} & & & \\ & & & & & & & \ddots & & \\ & & & & & & & & e_{n-1} & \\ & & & & & & & & & e_n \end{array}\right],$$

y

$$V = \left[\begin{array}{ccccc|ccccc} 1 & u_1 & & & & & & & & \\ & 1 & u_2 & & & & & & & \\ & & 1 & \ddots & & & & & & \\ & & & \ddots & u_{q-1} & & & & & \\ & & & & 1 & u_q & & & & \\ \hline & & & & & 1 & & & & \\ & & & & & l_{q+2} & 1 & & & \\ & & & & & & l_{q+3} & \ddots & & \\ & & & & & & & & \ddots & \\ & & & & & & & & l_n & 1 \end{array}\right],$$

tales que

$$A = MDV. \tag{3.8}$$

Si se procede como en la factorización LDU de la matriz A, los elementos de M, D y V pueden calcularse mediante las siguientes relaciones

- Sea $e_1 = a_1$.
- Para $i = 2, 3, \ldots, q$, sea

$$u_{i-1} = \frac{b_{i-1}}{e_{i-1}}; \quad l_i = \frac{c_i}{e_{i-1}} \tag{3.9}$$

  y calcúlese

$$\begin{aligned} e_i &= a_i - l_i e_{i-1} u_{i-1} \\ &= a_i - c_i u_{i-1}. \end{aligned} \tag{3.10}$$

- Sea

$$u_q = \frac{b_q}{e_q}, \quad l_{q+1} = \frac{c_{q+1}}{e_q}.$$

- Sea $e_n = a_n$ y calcúlese

$$l_n = \frac{c_n}{e_n}, \quad u_{n-1} = \frac{b_{n-1}}{e_n}. \tag{3.11}$$

- Para $i = n-1, n-2, \ldots, q+2$, calcúlese

$$\begin{aligned} e_i &= a_i - l_{i+1} e_{i+1} u_i \\ &= a_i - l_{i+1} b_i. \end{aligned} \tag{3.12}$$

  y sea

$$l_i = \frac{c_i}{e_i}; \quad u_{i-1} = \frac{b_{i-1}}{e_i}. \tag{3.13}$$

- Finalmente, calcúlese

$$\begin{aligned} e_{q+1} &= a_{q+1} - l_{q+1} e_q u_q - l_{q+2} e_{q+2} u_{q+2} \\ &= a_{q+1} - c_{q+1} u_q - l_{q+2} b_{q+1}. \end{aligned} \tag{3.14}$$

Para realizar la implementación de los cálculos anteriores en un ordenador con dos procesadores, se utiliza una técnica similar a la desarrollada por Van der Vorst [103] para matrices simétricas. Obsérvese que mediante las relaciones de recurrencia (3.10) y (3.12) se calculan todos los elementos de la diagonal principal de D excepto e_{q+1}, estos cálculos pueden realizarse perfectamente en paralelo (sin necesidad de comunicación de datos) ya que cada procesador contiene los elementos necesarios. Sin embargo, para calcular e_{q+1} (en el procesador P_1) mediante la relación (3.14) se necesita conocer el valor de e_q, calculado en el procesador P_0. Por lo tanto, es necesario un paso de comunicación para transmitir los datos necesarios desde el procesador P_0 al procesador P_1. En la figura 3.1 se muestran los cálculos en cada uno de los procesadores y las comunicaciones necesarias cuando $n = 8$.

Para resolver el sistema (3.1) usando la factorización (3.8), se necesita resolver en primer lugar el sistema $MD\boldsymbol{z} = \boldsymbol{d}$, para $\boldsymbol{z}$, y a continuación el sistema $V\boldsymbol{x} = \boldsymbol{z}$, para $\boldsymbol{x}$. Como consecuencia de la estructura especial de las matrices M y D, el vector

$$\boldsymbol{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$

$\boxed{P_0}$	$\boxed{P_1}$
$e_1 = a_1; \quad u_1 = \dfrac{b_1}{e_1}$	$e_8 = a_8; \quad l_8 = \dfrac{c_8}{e_8}$
$e_2 = a_2 - c_2u_1; \quad u_2 = \dfrac{b_2}{e_2}$	$e_7 = a_7 - l_8b_7; \quad l_7 = \dfrac{c_7}{e_7}$
$e_3 = a_3 - c_3u_2; \quad u_3 = \dfrac{b_3}{e_3}$	$e_6 = a_6 - l_7b_6; \quad l_6 = \dfrac{c_6}{e_6}$
$e_4 = a_4 - c_4u_3; \quad u_4 = \dfrac{b_4}{e_4}$	
$u_4 \quad \longrightarrow\longrightarrow\longrightarrow$	$\boxed{u_4}$
	$e_5 = a_5 - c_5\boxed{u_4} - l_6b_5$

Figura 3.1: *Cálculo en paralelo de los elementos de D para una matriz con $n = 8$.*

se puede obtener directamente de las expresiones

$$z_1 = \frac{d_1}{e_1}, \tag{3.15}$$

$$z_i = \frac{d_i - c_i z_{i-1}}{e_i}, \quad \text{para } i = 2, 3, \ldots, q, \tag{3.16}$$

$$z_n = \frac{d_n}{e_n}, \tag{3.17}$$

$$z_i = \frac{d_i - b_i z_{i+1}}{e_i}, \quad \text{para } i = n-1, n-2, \ldots, q+2, \tag{3.18}$$

$$z_{q+1} = \frac{d_{q+1} - c_{q+1}z_q - b_{q+1}z_{q+2}}{e_{q+1}}. \tag{3.19}$$

Como consecuencia de la estructura de la matriz V, al resolver $V\boldsymbol{x} = \boldsymbol{z}$ se obtiene que

$$x_{q+1} = z_{q+1}$$

y por tanto

$$x_i = z_i - u_i x_{i+1}, \quad \text{para } i = q, q-1, \ldots, 1, \tag{3.20}$$

$$x_i = z_i - l_i x_{i-1}, \quad \text{para } i = q+2, \ldots, n. \tag{3.21}$$

$\boxed{P_0}$	$\boxed{P_1}$
$e_1 = a_1; \quad u_1 = \dfrac{b_1}{e_1}; \quad z_1 = \dfrac{d_1}{e_1}$	$e_8 = a_8; \quad l_8 = \dfrac{c_8}{e_8}; \quad z_8 = \dfrac{d_8}{e_8}$
$e_2 = a_2 - c_2u_1; \quad u_2 = \dfrac{b_2}{e_2}; \quad z_2 = \dfrac{d_2 - c_2z_1}{e_2}$	$e_7 = a_7 - l_8b_7; \quad l_7 = \dfrac{c_7}{e_7}; \quad z_7 = \dfrac{d_7 - b_7z_8}{e_7}$
$e_3 = a_3 - c_3u_2; \quad u_3 = \dfrac{b_3}{e_3}; \quad z_3 = \dfrac{d_3 - c_3z_2}{e_3}$	$e_6 = a_6 - l_7b_6; \quad l_6 = \dfrac{c_6}{e_6}; \quad z_6 = \dfrac{d_6 - b_6z_7}{e_6}$
$e_4 = a_4 - c_4u_3; \quad u_4 = \dfrac{b_4}{e_4}; \quad z_4 = \dfrac{d_4 - c_4z_3}{e_4}$	
$u_4, \quad z_4 \longrightarrow\longrightarrow\longrightarrow$	$\boxed{u_4}, \boxed{z_4}$
$\boxed{z_6}, \boxed{l_6}$	$\longleftarrow\longleftarrow\longleftarrow \quad z_6, \quad l_6$
$e_5 = a_5 - c_5u_4 - \boxed{l_6}b_5$	$e_5 = a_5 - c_5\boxed{u_4} - l_6b_5$
$x_5 = z_5 = \dfrac{d_5 - c_5z_4 - b_5\boxed{z_6}}{e_5}$	$z_5 = \dfrac{d_5 - c_5\boxed{z_4} - b_5z_6}{e_5}$
$x_4 = z_4 - u_4x_5$	
$x_3 = z_3 - u_3x_4$	$x_6 = z_6 - l_6x_5$
$x_2 = z_2 - u_2x_3$	$x_7 = z_7 - l_7x_6$
$x_1 = z_1 - u_1x_2$	$x_8 = z_8 - l_8x_7$

Figura 3.2: *Cálculo en paralelo de la solución del sistema (2.4) para una matriz con* $n = 8$.

Debe observarse que todos los cálculos expresados en las relaciones (3.15)–(3.21) pueden desarrollarse en paralelo en dos procesadores excepto el del elemento centrales z_{q+1}. Todas las comunicaciones que deben realizarse están orientadas precisamente al cálculo de este elemento.

En la figura 3.2 se muestran los cálculos realizados en cada procesador para obtener la solución del sistema (3.1) cuando $n = 8$.

Ejemplo 3.1 *Sea el sistema* $A\boldsymbol{x} = \boldsymbol{d}$*, donde*

$$A = \begin{bmatrix} 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 \end{bmatrix} \quad y \quad \boldsymbol{d} = \begin{bmatrix} 67 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 65 \end{bmatrix}.$$

Puesto que cada término independiente se ha obtenido como suma de los coeficientes de su ecuación, la solución del sistema tiene todas sus componentes iguales a 1.

Si se aplican las relaciones (3.10)–(3.14) se obtiene

$$M = \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{66} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{66}{4357} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{66}{4357} & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & -\frac{66}{4357} & 1 & \frac{66}{4357} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & \frac{66}{4357} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{1}{66} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right],$$

$$D = \left[\begin{array}{cccc|cccc} 66 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{4357}{66} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{4357}{66} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{4357}{66} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & \frac{2179}{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{4357}{66} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{4357}{66} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 66 \end{array}\right],$$

$$V = \left[\begin{array}{cccc|cccc} 1 & \frac{1}{66} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \frac{66}{4357} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \frac{66}{4357} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{66}{4357} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{66}{4357} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{66}{4357} & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{66} & 1 \end{array}\right].$$

Utilizando las expresiones (3.15)-(3.19) se obtiene la solución del sistema $MD\boldsymbol{z} = \boldsymbol{d}$,

$$\boldsymbol{z} = \begin{bmatrix} \frac{67}{66} \\ \frac{4423}{4357} \\ \frac{4423}{4357} \\ \frac{4423}{4357} \\ 1 \\ \frac{4291}{4357} \\ \frac{4291}{4357} \\ \frac{65}{66} \end{bmatrix}.$$

Finalmente, mediante las relaciones (3.20) y (3.21) se obtiene la solución de $V\boldsymbol{x} = \boldsymbol{z}$ *que coincide con la del sistema* $A\boldsymbol{x} = \boldsymbol{d}$;

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

■

3.2.2 Algoritmo BSP para dos procesadores

Todo lo expuesto en la subsección 3.2.1 sugiere el siguiente algoritmo BSP para dos procesadores. Se supone que la matriz A y el vector de términos independientes $\boldsymbol{d}$ están

inicialmente almacenados en el procesador principal P_0.

Algoritmo 3.1 Algoritmo BSP para dos procesadores

Superpaso 1

El procesador P_0 envía a P_1 los elementos a_i, b_i, c_i, d_i, para $i = q+1, q+2, \ldots, n$, excepto $b_n = 0$.

Superpaso 2

Cálculo de los elementos e_i y z_i.

- En el procesador P_0,
  - Sea $e_1 = a_1$ y calcúlese z_1 mediante la expresión (3.15).
  - Para $i = 2, 3, \ldots, q$, calcúlese u_{i-1}, e_i y z_i usando las expresiones (3.9), (3.10) y (3.16).
  - El procesador P_0 envía al procesador P_1 los elementos u_q y z_q.
- En el procesador P_1,
  - Sea $e_n = a_n$ y calcúlense l_n, z_n utilizando las expresiones (3.11) y (3.17).
  - Para $i = n-1, n-2, \ldots, q+2$, calcúlense e_i, l_i y z_i usando las expresiones (3.12) y (3.13) y (3.18).
  - El procesador P_1 envía a P_0 los elementos l_{q+2} y z_{q+2}.

Superpaso 3

Cálculo de los elementos e_{q+1}, z_{q+1} y de la solución.

- Ambos procesadores calculan e_{q+1} y z_{q+1} de acuerdo con (3.14) y (3.19), respectivamente. Sea $x_{q+1} = z_{q+1}$, entonces
  - En el procesador P_0 se calcula x_i, para $i = q, q-1, \ldots, 1$, usando la expresión (3.20).

  - En el procesador P_1 se calcula x_i, para $i = q+2, q+3, \ldots, n$, usando la expresión (3.21).

- El procesador P_1 envía al procesador P_0 las componentes x_i, para $i = q+2, q+3, \ldots, n$, del vector solución $\boldsymbol{x}$.

El coste computacional del algoritmo 3.1 viene dado por la suma de los costes individuales de cada uno de los superpasos que lo integran.

Coste del superpaso 1. En este primer superpaso sólo hay comunicación, por lo que el coste aritmético es cero. El coste de comunicación viene determinado por el envío desde el procesador principal a P_1 de $4q - 1$ elementos, correspondientes a la matriz de coeficientes y el vector de términos independientes. Por tanto, el coste total de este superpaso es

$$(2n - 1)g + l.$$

Coste del superpaso 2. El procesador P_0 realiza una división para calcular z_1, una división para calcular u_i, con $i = 1, 2, \ldots, q$, dos operaciones para el cálculo de d_i y tres operaciones para el cálculo de z_i, con $i = 2, 3, \ldots, q$; en total $6q-4$ operaciones. Mientras tanto, el procesador P_1 realiza $6q-10$ (seis menos) operaciones, que corresponden a las $q-1$ divisiones necesarias para calcular l_i, con $i = n, n-1, \ldots, q+2$, una división para el cálculo de z_n, 2 operaciones para calcular d_i y 3 para calcular z_i, con $i = n-1, n-2, \ldots, q+2$.

El procesador principal envía dos elementos a P_1 y éste a su vez comunica dos a P_0. Luego el coste de comunicación es $2g$. En consecuencia, el coste del superpaso es

$$3n - 4 + 2g + l.$$

Coste del superpaso 3. En este superpaso se deben contabilizar las operaciones realizadas por el procesador principal ya que P_1 realiza dos operaciones menos. Para calcular los elementos e_{q+1} y z_{q+1} se requieren 9 operaciones y para el cálculo de x_i, con $i = 1, 2, \ldots, q$, se precisan $2q$ operaciones, luego el coste aritmético es $2q + 9$.

En cuanto al coste de comunicación, el procesador P_1 envía al procesador principal las componentes $x_{q+2}, x_{q+3}, \ldots, x_n$ de la solución $\boldsymbol{x}$, lo que supone un coste de $(q-1)g$. Se

tiene por tanto que el coste de este superpaso es

$$n + 9 + \left(\frac{n}{2} - 1\right) g + l.$$

Sumando los costes se obtiene que el coste del algoritmo 3.1 es

$$4n + 5 + \frac{5}{2}ng + 3l. \tag{3.22}$$

3.3 Método bidireccional para un número par de procesadores

En esta sección se presenta una generalización para p procesadores del método estudiado en la sección 3.2, basada en algunos de los resultados obtenidos en el método de las particiones superpuestas que se ha estudiado en el capítulo 2. Se supone que tanto el número de procesadores, p, como el orden, n, de la matriz de coeficientes es par; asimismo se supone que existe un número natural k tal que $k = \dfrac{n}{p}$.

3.3.1 Descripción del método

Sea m el valor obtenido al aplicar la expresión (2.14), esto es

$$m = \left\lceil \frac{1}{\log \delta^{-1}} \log \frac{\epsilon(1 - \delta^{-2})}{\mu} \right\rceil, \tag{3.23}$$

donde ϵ es el máximo error permitido al resolver el sistema (3.1) mediante el método de las particiones superpuestas y

$$\mu = \frac{\max\limits_{1 \le i \le n} \left\{ \left| \dfrac{d_i}{a_i} \right| \right\}}{1 - \delta^{-1}},$$

es una cota superior de la norma infinito de la solución del mismo. Se define

$$m' = 2 \left\lceil \frac{m}{2} \right\rceil. \tag{3.24}$$

Se considera el sistema (3.1) particionado en $\frac{p}{2}$ bloques de tamaño $2k \times 2k$ como sigue

$$\begin{bmatrix} A_1 & B_1 & & & \\ C_2 & A_2 & B_2 & & \\ & \ddots & \ddots & \ddots & \\ & & C_{\frac{p}{2}-1} & A_{\frac{p}{2}-1} & B_{\frac{p}{2}-1} \\ & & & C_{\frac{p}{2}} & A_{\frac{p}{2}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{\frac{p}{2}-1} \\ \boldsymbol{x}_{\frac{p}{2}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \vdots \\ \boldsymbol{d}_{\frac{p}{2}-1} \\ \boldsymbol{d}_{\frac{p}{2}} \end{bmatrix} \tag{3.25}$$

y una nueva partición en bloques (basada en la anterior) añadiendo $2m$ filas (respectivamente, componentes) a cada uno de los bloques intermedios de la matriz de coeficientes (respectivamente, vector de términos independientes) y m' filas (respectivamente, componentes) a los bloques extremos de A (respectivamente, $\boldsymbol{d}$). Los nuevos bloques están solapados y contienen un número par de filas, de este modo se obtiene un conjunto de subsistemas (los centrales con $2k+2m$ ecuaciones e incógnitas y los extremos con $2k+m'$ ecuaciones e incógnitas)

$$\hat{A}_i \hat{\boldsymbol{x}}_i = \hat{\boldsymbol{d}}_i, \quad i = 1, 2, \ldots, \frac{p}{2}, \tag{3.26}$$

donde

- $\hat{A}_1$ es la submatriz de A de tamaño $(2k+m') \times (2k+m')$ formada por las filas y columnas $1, 2, \ldots, 2k+m'$ y $\hat{\boldsymbol{d}}_1$ es el vector formado por las componentes $1, 2, \ldots, 2k+m'$ de $\boldsymbol{d}$;
- $\hat{A}_i$, para $i = 2, 3, \ldots, \frac{p}{2} - 1$, es la submatriz de A de tamaño $(2k+2m) \times (2k+2m)$ formada por las filas y columnas $2(i-1)k-m+1, 2(i-1)k-m+2, \ldots, 2ik+m$ y $\hat{\boldsymbol{d}}_i$ es el vector formado por las componentes $2(i-1)k-m+1, 2(i-1)k-m+2, \ldots, 2ik+m$ de $\boldsymbol{d}$;
- $\hat{A}_{\frac{p}{2}}$ es la submatriz de A de tamaño $(2k+m') \times (2k+m')$ formada por las filas y columnas $n-2k-m'+1, n-2k-m'+2, \ldots, n$ y, finalmente, $\hat{\boldsymbol{d}}_{p-1}$ es el vector formado por las componentes $n-2k-m'+1, n-2k-m'+2, \ldots, n$, de $\boldsymbol{d}$.

Estos subsistemas son resueltos aplicando el algoritmo 3.1 en cada par de procesadores (P_{2i-2}, P_{2i-1}), para $i = 1, 2, \ldots, \frac{p}{2}$.

El método consta de tres fases: en la primera, se comunican datos desde el procesador principal al resto; en la segunda, se aplica el algoritmo 3.1 para resolver el sistema (3.26) en cada par de procesadores (P_{2i-2}, P_{2i-1}), con $i = 1, 2, \ldots, \dfrac{p}{2}$; finalmente, en la tercera fase, cada procesador comunica sus soluciones parciales al procesador principal. A continuación se describen cada una de esas fases.

Fase 1

Como se ha comentado anteriormente, cada par de procesadores recibe los bloques $\hat{A}_i$ y $\hat{\boldsymbol{d}}_i$ de los subsistemas (3.26). En un único superpaso, cada procesador recibe los datos necesarios para ejecutar el algoritmo 3.1 en la siguiente fase, el número total de datos recibidos por el procesador j, para $j = 1, 2, \ldots, p-1$, es $4t(j) - 1$, con

$$t(j) = \begin{cases} k + \frac{m'}{2}, & \text{si } j = 1, p-2, p-1, \\ k + m, & \text{en otro caso.} \end{cases} \tag{3.27}$$

La primera fila de A y $\boldsymbol{d}$ que el procesador j, para $j = 1, 2, \ldots, p-1$, debe recibir del principal es $\phi_1(j)$, con

$$\phi_1(j) = \begin{cases} t(j) + 1, & \text{si } j = 1, \\ 2ik - m + 1, & \text{si } j = 2i, \quad \text{con } i = 1, 2, \ldots, \dfrac{p}{2} - 2, \\ (2i+1)k + 1, & \text{si } j = 2i+1, \text{ con } i = 1, 2, \ldots, \dfrac{p}{2} - 2, \\ n - 2t(j) + 1, & \text{si } j = p-2, \\ n - t(j) + 1, & \text{si } j = p-1; \end{cases} \tag{3.28}$$

nótese que si $p = 4$ la expresión anterior se limita a

$$\phi_1(j) = \begin{cases} t(j) + 1, & \text{si } j = 1, \\ n - 2t(j) + 1, & \text{si } j = p-2, \\ n - t(j) + 1, & \text{si } j = p-1. \end{cases}$$

Ejemplo 3.2 *Para $p = 8$, $n = 40$ y $m = 3$, la partición (3.25) tiene la forma*

$$\begin{bmatrix} A_1 & B_1 & & \\ C_2 & A_2 & B_2 & \\ & C_3 & A_3 & B_3 \\ & & C_4 & A_4 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \\ \boldsymbol{x}_4 \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \boldsymbol{d}_3 \\ \boldsymbol{d}_4, \end{bmatrix} \tag{3.29}$$

$m' = 4$ *y*

$$t(j) = \begin{cases} 7, & \textit{si } j = 1, 6, 7 \\ 8, & \textit{en otro caso.} \end{cases}$$

Este último valor representa el número total de filas que cada procesador debe recibir de la matriz de coeficientes y del vector de términos independientes.

Al calcular la función ϕ_1, usando la expresión (3.28), se obtiene

$$\begin{array}{lllllllll} \phi_1(1) & = & 8, & \phi_1(2) & = & 8, & \phi_1(3) & = & 16, \\ \phi_1(4) & = & 18, & \phi_1(5) & = & 26, & \phi_1(6) & = & 27, \\ \phi_1(7) & = & 34, & & & & & & \end{array}$$

lo que significa que el procesador P_1 recibe los elementos de A y $\boldsymbol{d}$ situados desde la fila 8 a la 14 excepto b_{14}, P_2 recibe los elementos situados desde la fila 8 a la 15 excepto c_8, P_3 recibe los elementos situados desde la fila 16 a la 23 excepto b_{23}, P_4 recibe los elementos situados desde la fila 18 a la 25 excepto c_{18}, P_5 recibe los elementos situados desde la fila 26 a la 33 excepto b_{33}, P_6 recibe los elementos situados desde la fila 27 a la 33 excepto c_{27} y, finalmente, P_7 recibe los elementos situados desde la fila 34 a la 40. ■

Fase 2

En esta fase cada par de procesadores (P_{2i-2}, P_{2i-1}), para $i = 1, 2, \ldots, \dfrac{p}{2}$, ejecutan el algoritmo 3.1 (excepto la primera y última comunicación) con los elementos que han recibido en la fase 1. Obsérvese que, en este caso, los procesadores pares ejecutan las mismas operaciones y comunicaciones que el procesador P_0 en el algoritmo 3.1, mientras

que los procesadores impares realizan las mismas operaciones y comunicaciones que el procesador P_1 en el algoritmo 3.1.

Fase 3

Tras la ejecución del algoritmo 3.1 en la fase 2, cada procesador j, para $j = 1, 2, \ldots, p-1$, ha obtenido un vector solución parcial de $t(j)$ componentes (la solución parcial de P_0 tiene $k + \frac{m'}{2}$ componentes). El objetivo de esta fase es determinar qué componentes de su solución parcial debe enviar cada procesador al principal a fin de obtener la solución general del sistema (3.1).

Ejemplo 3.3 *Continuando con el ejemplo 3.2, cuando se aplica la fase 2 del algoritmo 3.1 los procesadores P_2 a P_6 calculan un vector solución parcial de 8 componentes mientras que la solución parcial de los procesadores P_0, P_1, P_6 y P_7 tiene 7 componentes. De cada uno de los cuatro bloques de la partición (3.29) se eligen 10 componentes como sigue: de $\boldsymbol{x}_1$ las 10 primeras componentes, de $\boldsymbol{x}_4$ las 10 últimas y para el resto las 10 componentes centrales. Obsérvese que parte de las componentes de $\boldsymbol{x}_i$, para $i = 1, 2, 3, 4$, están en el procesador P_{2i-2} y parte en el procesador P_{2i-1}, se necesita pues determinar qué componentes se eligen de cada procesador.* ■

Todas las componentes del vector solución parcial del primer y último procesador forman parte de la solución general. Se deben tomar las k últimas componentes en los procesadores pares, excepto en el último procesador par en el que se eligen las $k - \frac{m'}{2}$ últimas componentes, y las k primeras componentes en los procesadores impares, excepto en el primer procesador impar en el que se toman las primeras $k - \frac{m'}{2}$ componentes. En la figura 3.3 se muestra, para $p = 8$, la formación de la solución del sistema (3.1) a partir de las soluciones parciales en cada procesador.

3.3.2 Algoritmos BSP para un número par de procesadores

El siguiente algoritmo BSP implementa el nuevo método paralelo para resolver sistemas tridiagonales, basado en las tres fases que se han descrito anteriormente.

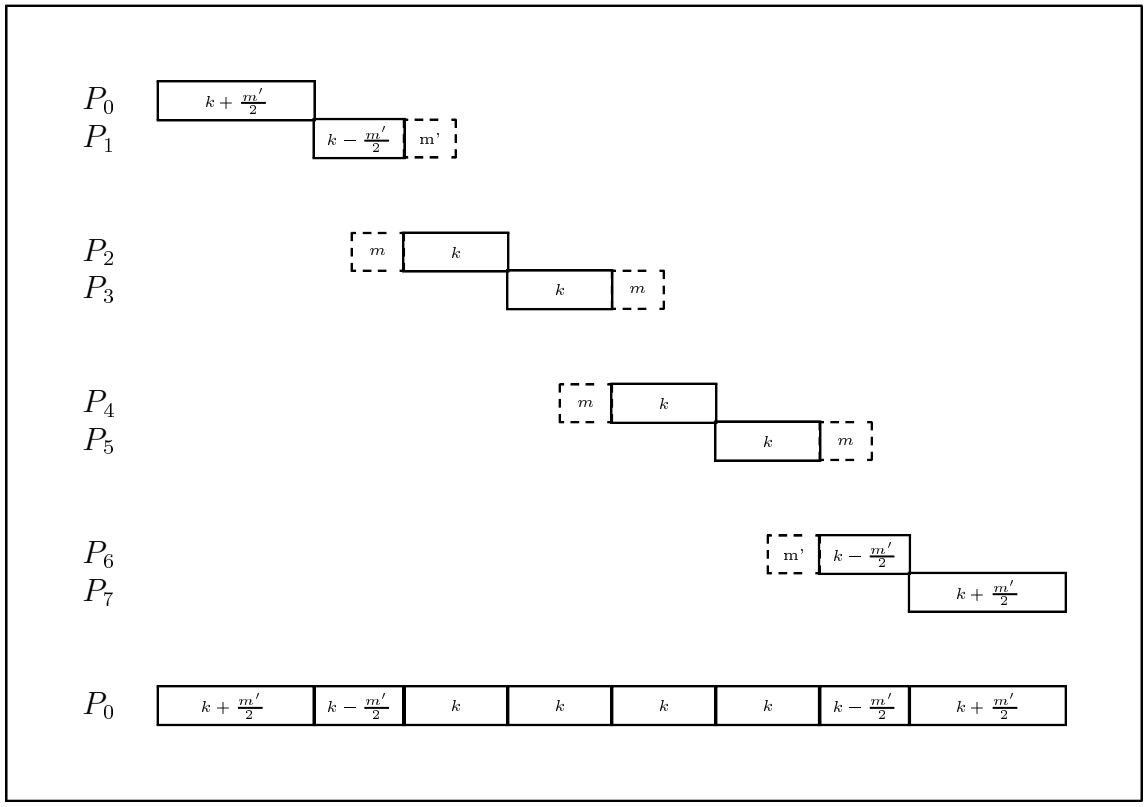


Figura 3.3: *Obtención de la solución general a partir de las soluciones parciales para* $p = 8$.

Algoritmo 3.2 Algoritmo BSP para resolver sistemas tridiagonales para p procesadores.

Superpaso 1

Inicio.

- El procesador principal calcula $k = \dfrac{n}{p}$, m de acuerdo con (3.23) y m' mediante (3.24).
- Cada procesador recibe desde el procesador principal los correspondientes elementos de A y $\boldsymbol{d}$ de acuerdo con la función ϕ_1 definida en (3.28), esto es
  - P_0 envía a P_1 la fila j-ésima de $\hat{A}_1$ y $\hat{\boldsymbol{d}}_1$, para $j = k+\dfrac{m'}{2}+1, k+\dfrac{m'}{2}+2, \ldots, 2k+m'$.
  - Para $i = 1, 2, \ldots, \dfrac{p}{2} - 2$.

- $\diamond$ P_0 envía a P_{2i} la fila j-ésima de $\hat{A}_i$ y $\hat{\boldsymbol{d}}_i$, para $j = 1, 2, \ldots, k+m$.
- $\diamond$ P_0 envía a P_{2i+1} la fila j-ésima de $\hat{A}_i$ y $\hat{\boldsymbol{d}}_i$, para $j = k+m+1, k+m+2, \ldots, 2k+2m$.

$\circ$ P_0 envía a P_{p-2} y P_{p-1} la fila j-ésima de $\hat{A}_{\frac{p}{2}}$ y $\hat{\boldsymbol{d}}_{\frac{p}{2}}$, para $j = 1, 2, \ldots, k + \dfrac{m'}{2}$ y $j = k + \dfrac{m'}{2} + 1, k + \dfrac{m'}{2} + 2, \ldots, 2k + m'$, respectivamente.

- El procesador principal envía m al resto de procesadores.

Superpaso 2

- Para $i = 1, p-2, p-1$, el procesador P_i calcula m'.
- Para $i = 1, 2, \ldots, \dfrac{p}{2}$, los procesadores (P_{2i-2}, P_{2i-1}) resuelven el subsistema $\hat{A}_i \hat{\boldsymbol{x}}_i = \hat{d}_i$ ejecutando los superpasos 2 y 3 del algoritmo 3.1, excepto la comunicación final de soluciones del final del superpaso 3.
- Comunicación de soluciones parciales.

  $\circ$ P_1 comunica a P_0 las componentes $k + \dfrac{m'}{2} + 1, k + \dfrac{m'}{2} + 2, \ldots, 2k$ de $\hat{\boldsymbol{x}}_1$.

  $\circ$ Para $i = 1, 2, \ldots, \dfrac{p}{2} - 2$

  - $\diamond$ P_{2i} comunica a P_0 las componentes $m+1, m+2, \ldots, m+k$ de $\hat{\boldsymbol{x}}_{i+1}$.
  - $\diamond$ P_{2i+1} comunica a P_0 las componentes $m+k+1, m+k+2, \ldots, m+2k$ de $\hat{\boldsymbol{x}}_{i+1}$.

  $\circ$ P_{p-2} comunica a P_0 las componentes $m'+1, m'+2, \ldots, k + \dfrac{m'}{2}$ de $\hat{\boldsymbol{x}}_{\frac{p}{2}}$.

  $\circ$ P_{p-1} comunica a P_0 las componentes $k + \dfrac{m'}{2} + 1, k + \dfrac{m'}{2} + 2, \ldots, 2k + m'$ de $\hat{\boldsymbol{x}}_{\frac{p}{2}}$.

- El procesador principal forma la solución $\boldsymbol{x}$ del sistema (3.1) haciendo:

  $\circ$ Para $j = 1, 2, \ldots, 2k$, las componentes j-ésima y $(n-2k+j)$-ésima de $\boldsymbol{x}$ igual a las componentes j-ésima y $(m'+j)$-ésima de $\hat{\boldsymbol{x}}_1$ y $\hat{\boldsymbol{x}}_{\frac{p}{2}}$, respectivamente.

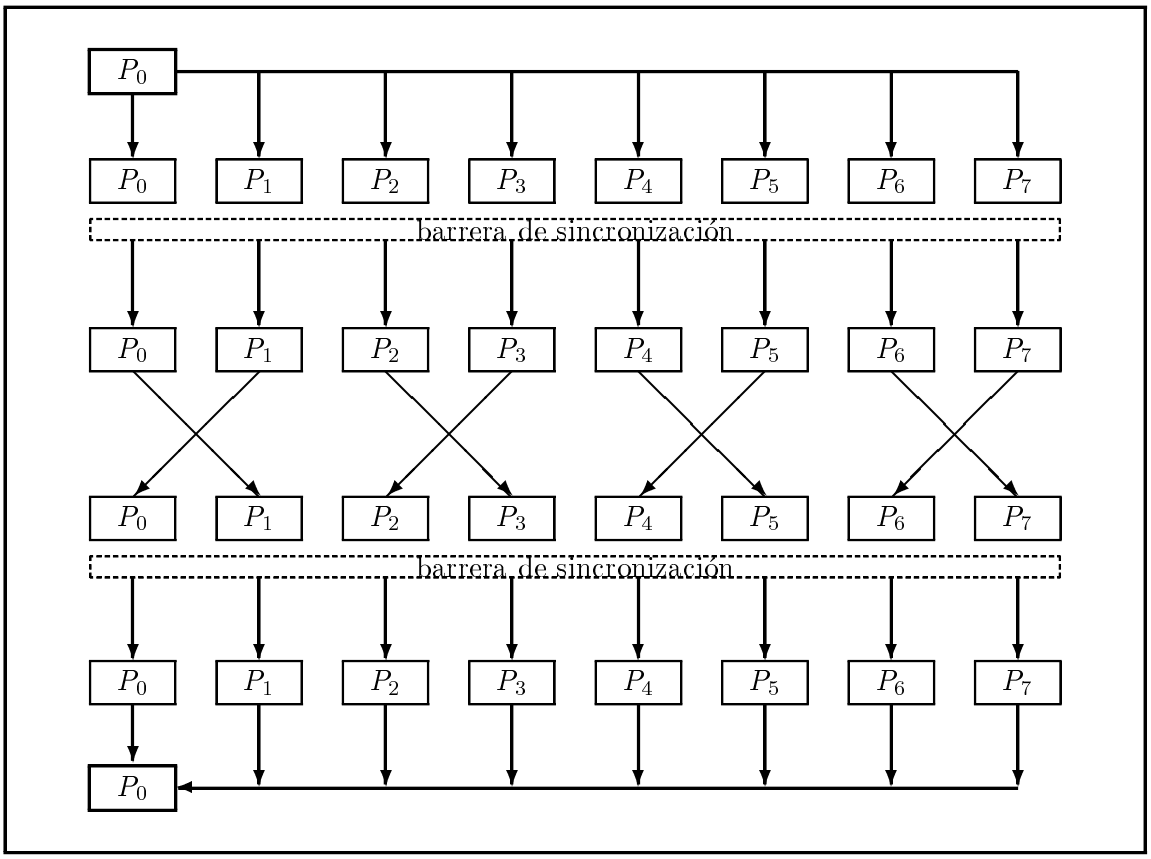


Figura 3.4: *Esquema de comunicación para* 8 *procesadores del método descrito en el algoritmo 3.2.*

- Para $i = 1, 2, \ldots, \dfrac{p}{2} - 2$ y $j = 1, 2, \ldots, 2k$, la componente $(2ik + j)$-ésima de $\boldsymbol{x}$ igual a la componente $(m + j)$-ésima de $\hat{\boldsymbol{x}}_{i+1}$.

Obsérvese que el algoritmo 3.2 contiene tres barreras de sincronización ya que en el superpaso 2 se ejecutan los superpasos 2 y 3 del algoritmo 3.1. En la figura 3.4 se muestra un ejemplo de ejecución del algoritmo para $p = 8$; obsérvese el especial patrón de comunicaciones necesario: cada par de procesadores comunican entre sí en el superpaso central.

A continuación se obtiene el coste computacional del algoritmo 3.2.

Coste del superpaso 1. Para calcular m, a partir de los parámetros δ y μ, se necesitan $3n+12$ operaciones (véase la expresión (2.19) en la página 77). El cálculo de k requiere una operación y el de m' dos. En consecuencia, el coste aritmético del superpaso es $3n+15$.

Para obtener el coste de comunicación de este superpaso, debe tenerse en cuenta que el procesador principal envía $4(k+m)-1$ elementos al procesador P_i, para $i=2,3,\ldots,p-3$, y envía $4\left(k+\frac{m'}{2}\right)-1$ elementos al procesador P_j, para $j=1,p-2,p-3$, además envía m a todos los procesadores. Como en el peor de los casos $m'=m+1$, en total envía $(p-4)(4k+4m-1)+3(4k+2m+1)+p-1$ elementos, luego el coste de comunicación es $(4n-4k-10m+4mp+6)g$.

El coste computacional del superpaso es, por tanto,

$$3n+15+(4n-4k-10m+4mp+6)g+l. \tag{3.30}$$

Coste del superpaso 2. Los procesadores que más operaciones realizan son P_i con $i=2,3,\ldots,p-3$. Cada pareja de procesadores resuelve un sistema de tamaño $2(k+m)\times 2(k+m)$, en consecuencia el coste aritmético del superpaso 2 del algoritmo 3.1 es $3[2(k+m)]-4$ y el del superpaso 3 es $2(k+m)+9$ (véase la expresión (3.22)). El coste aritmético de este superpaso es, por tanto, $8k+8m+5$.

En este superpaso se pueden considerar dos etapas de comunicación. La primera contiene la comunicación del superpaso 2 del algoritmo 3.1, cada pareja de procesadores comunican entre sí 2 elementos. La segunda está relacionada con la obtención del vector solución final en el procesador principal, este recibe en total $n-\left(k+\frac{m'}{2}\right)$ elementos del resto de procesadores. El coste total de comunicación de este superpaso es $\left(n-k-\frac{m}{2}+\frac{3}{2}\right)g$, considerando que en el peor de los casos $m'=m+1$.

El coste global de este superpaso es

$$8k+8m+5+\left(n-k-\frac{m}{2}+\frac{3}{2}\right)g+2l. \tag{3.31}$$

Sumando las expresiones (3.30) y (3.31) se obtiene el coste computacional del método descrito por el algoritmo 3.2

$$3n+8k+8m+20+\left(5n-5k-\frac{21}{2}m+4mp+\frac{15}{2}\right)g+3l.$$

Ejemplo 3.4 *Sea el sistema* $A\boldsymbol{x} = \boldsymbol{d}$, *de* 12 *ecuaciones lineales, donde*

$$A = \begin{bmatrix} 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 \end{bmatrix} \quad y \quad \boldsymbol{d} = \begin{bmatrix} 67 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 65 \end{bmatrix}.$$

Puesto que cada término independiente se ha obtenido como suma de los coeficientes de su ecuación, la solución del sistema tiene sus doce componentes iguales a 1.

A continuación se va a resolver el sistema aplicando el algoritmo 3.2 para 6 *procesadores.*

La diagonal dominanza de A *vale*

$$\delta = \min\left\{\frac{67}{1}, \frac{66}{1+1}, \cdots, \frac{66}{1+1}, \frac{65}{1}\right\} = 33.$$

Una cota superior de la norma infinito de la solución del sistema viene dada por

$$\mu = \frac{\max\left\{\frac{67}{66}, \frac{66}{66}, \cdots, \frac{66}{66}, \frac{65}{66}\right\}}{1 - 33^{-1}} = \frac{67}{64} = 1.0469.$$

Si se toma un error $\epsilon = 10^{-3}$, *se obtiene para* m *el siguiente valor*

$$m = \left\lceil \frac{1}{\log 33^{-1}} \log \frac{10^{-3}(1 - 33^{-2})}{\frac{67}{64}} \right\rceil = \lceil 1.989 \rceil = 2,$$

en consecuencia $m' = 2$.

De las expresiones (3.27) y (3.28) se obtiene

$$\begin{aligned} t(1) &= 3, \quad \phi_1(1) = 4, \\ t(2) &= 4, \quad \phi_1(2) = 3, \\ t(3) &= 4, \quad \phi_1(3) = 7, \\ t(4) &= 3, \quad \phi_1(4) = 7, \\ t(5) &= 3, \quad \phi_1(5) = 10, \end{aligned}$$

esto significa que el procesador P_1 *recibe de* P_0 *los elementos de* A *y* $\boldsymbol{d}$ *situados desde la fila* 4 *hasta la fila* 6 *excepto el elemento* b_6, P_2 *desde la fila* 3 *hasta la fila* 6 *excepto el elemento* c_3, P_3 *desde la fila* 7 *hasta la fila* 10 *excepto el elemento* b_{10}, P_4 *desde la fila* 7 *hasta la fila* 9 *excepto el elemento* c_7 *y* P_5 *desde la fila* 10 *hasta la fila* 12 *excepto el elemento* b_{12}.

Los procesadores P_0 *y* P_1 *aplican los superpasos 2 y 3 del algoritmo 3.1 al subsistema*

$$\begin{bmatrix} 66 & 1 & 0 & 0 & 0 & 0 \\ -1 & 66 & 1 & 0 & 0 & 0 \\ 0 & -1 & 66 & 1 & 0 & 0 \\ 0 & 0 & -1 & 66 & 1 & 0 \\ 0 & 0 & 0 & -1 & 66 & 1 \\ 0 & 0 & 0 & 0 & -1 & 66 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \\ \hat{x}_4 \\ \hat{x}_5 \\ \hat{x}_6 \end{bmatrix} = \begin{bmatrix} 67 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \end{bmatrix},$$

obteniendo como solución del mismo

$$\hat{\boldsymbol{x}}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.9998 \\ 1.0151 \end{bmatrix}.$$

Los procesadores P_2 y P_3 aplican los superpasos 2 y 3 del algoritmo 3.1 al subsistema

$$\begin{bmatrix} 66 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 66 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 66 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 66 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 66 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 66 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 66 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 66 \end{bmatrix} \begin{bmatrix} \hat{x}_3 \\ \hat{x}_4 \\ \hat{x}_5 \\ \hat{x}_6 \\ \hat{x}_7 \\ \hat{x}_8 \\ \hat{x}_9 \\ \hat{x}_{10} \end{bmatrix} = \begin{bmatrix} 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 66 \end{bmatrix},$$

obteniendo como solución del mismo

$$\hat{\boldsymbol{x}}_2 = \begin{bmatrix} 0.9849 \\ 0.9998 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0.9998 \\ 1.0151 \end{bmatrix}.$$

Los procesadores P_4 y P_5 aplican los superpasos 2 y 3 del algoritmo 3.1 al subsistema

$$\begin{bmatrix} 66 & 1 & 0 & 0 & 0 & 0 \\ -1 & 66 & 1 & 0 & 0 & 0 \\ 0 & -1 & 66 & 1 & 0 & 0 \\ 0 & 0 & -1 & 66 & 1 & 0 \\ 0 & 0 & 0 & -1 & 66 & 1 \\ 0 & 0 & 0 & 0 & -1 & 66 \end{bmatrix} \begin{bmatrix} \hat{x}_7 \\ \hat{x}_8 \\ \hat{x}_9 \\ \hat{x}_{10} \\ \hat{x}_{11} \\ \hat{x}_{12} \end{bmatrix} = \begin{bmatrix} 66 \\ 66 \\ 66 \\ 66 \\ 66 \\ 65 \end{bmatrix},$$

obteniendo como solución del mismo

$$\hat{\boldsymbol{x}}_3 = \begin{bmatrix} 0.9849 \\ 0.9998 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

En la siguiente tabla se muestra la solución parcial de cada procesador.

Procesador	*Solución parcial*			
P_0	1	1	1	—
P_1	1	0.9998	1.0151	—
P_2	0.9849	0.9998	1	1
P_3	1	1	0.9998	1.0151
P_4	0.9849	0.9998	1	—
P_5	1	1	1	—

Finalmente, el procesador principal obtiene la solución general a partir de las soluciones parciales, tomando las tres componentes del vector solución parcial en P_0, la primera en P_1, la tercera y cuarta componentes de la solución parcial en P_2, las dos primeras en P_3, la tercera en P_4 y las tres componentes de la solución parcial en P_5. ■

En el capítulo 2 se definieron, para $i = 0, 1, \ldots, p-1$,

$$\delta_i = \min_{ik+1 \le j \le (i+1)k} \left\{ \frac{|a_j|}{|c_j| + |b_j|} \right\}, \tag{3.32}$$

$$\mu_i = \max_{ik+1 \le j \le (i+1)k} \left\{ \frac{|d_j|}{|a_j|} \right\}, \tag{3.33}$$

entonces

$$\delta = \min_{0\le i\le p-1}\{\delta_i\}, \tag{3.34}$$

$$\mu = \frac{\max_{0\le i\le p-1}\{\mu_i\}}{1-\delta^{-1}}. \tag{3.35}$$

Al igual que se hizo en el capítulo 2, el cálculo de los parámetros δ y μ se puede realizar en paralelo. En este caso, el procesador principal envía a cada procesador P_i, para $i = 1, 2, \ldots, p-1$, las filas $(ik+j)$-ésismas, con $j = 1, 2, \ldots, k$, del sistema de manera que éste pueda calcular los parámetros δ_i y μ_i de acuerdo con las expresiones (3.32) y (3.33).

Una vez calculado por P_0 el valor de m, debe comunicar a los otros procesadores el resto de elementos del sistema (3.1) que se necesitan para que cada pareja de procesadores pueda resolver el subsistema de (3.26) que le corresponda. Todos los procesadores deben recibir m filas excepto P_1 y P_{p-2} que deben recibir m' filas y P_{p-1} que debe recibir $\dfrac{m'}{2}$ filas. La primera fila de A y $\boldsymbol{d}$ que el procesador P_j, para $j = 1, 2, \ldots, p-1$, debe recibir del principal es $\phi_2(j)$, con

$$\phi_2(j) = \begin{cases} 2k+1, & \text{si } j = 1, \\ 2ik - m + 1, & \text{si } j = 2i \quad \text{con } i = 1, 2, \ldots, \frac{p}{2}-2, \\ 2(i+1)k+1, & \text{si } j = 2i+1 \text{ con } i = 1, 2, \ldots, \frac{p}{2}-2, \\ n - 2k - m' + 1, & \text{si } j = p-2, \\ n - k - \dfrac{m'}{2} + 1, & \text{si } j = p-1. \end{cases} \tag{3.36}$$

El siguiente algoritmo BSP modifica el algoritmo 3.2 para realizar el cálculo de δ y μ en paralelo.

Algoritmo 3.3 Algoritmo BSP para p procesadores con cálculo de δ y μ en paralelo

Superpaso 1

- El procesador principal P_0 calcula $k = \dfrac{n}{p}$.

- Para $i = 1, 2, \ldots, p-1$, el procesador P_0 envía a P_i los elementos c_{ik+j}, a_{ik+j}, b_{ik+j} y d_{ik+j}, con $j = 1, 2, \ldots, k$ excepto $b_n = 0$.

Superpaso 2

- Para $i = 0, 1, \ldots, p-1$, P_i calcula δ_i y μ_i de acuerdo con las expresiones (3.32) y (3.33).
- Para $i = 1, 2, \ldots, p-1$, P_i envía a P_0 los valores de los parámetros δ_i y μ_i.

Superpaso 3

- El procesador P_0 calcula los parámetros δ, μ, m y m' haciendo uso de las expresiones (3.34), (3.35), (3.23) y (3.24).
- Para $i = 1, 2, \ldots, p-1$, el procesador principal P_0 envía a P_i los correspondientes elementos de A y d, de acuerdo con la función ϕ_2 definida en (3.36), esto es
  - P_0 envía a P_1 los elementos c_j, a_j, b_j y d_j, con $j = 2k+1, 2k+2, \ldots, 2k+m'$, excepto $b_{2k+m'}$.
  - Para $i = 1, 2, \ldots, \dfrac{p}{2} - 2$
    - P_0 envía a P_{2i} los elementos c_j, a_j, b_j y d_j, con $j = 2ik - m + 1$, $2ik - m + 2, \ldots, 2ik$, excepto $c_{2ik-m+1}$.
    - P_0 envía a P_{2i+1} los elementos c_j, a_j, b_j y d_j, con $j = 2(i+1)k + 1$, $2(i+1)k + 2, \ldots, 2(i+1)k + m$, excepto $b_{2(i+1)k+m}$.
  - P_0 envía a P_{p-2} los elementos c_j, a_j, b_j y d_j, con $j = n - 2k - m' + 1$, $n - 2k - m' + 2, \ldots, n - 2k$, excepto $c_{n-2k-m'+1}$.
  - P_0 envía a P_{p-1} los elementos c_j, a_j, b_j y d_j, con $j = n - k - \dfrac{m'}{2} + 1$, $n - k - \dfrac{m'}{2} + 2, \ldots, n - k$.
- El procesador principal envía m al resto de procesadores.

Superpaso 4

Como el superpaso 2 del algoritmo 3.2.

Al repartir el cálculo de δ y μ entre p procesadores, el número de operaciones baja de $3n+3$ a $3k+3$, luego el coste aritmético pasa de $3n+8k+8m+20$ a $11k+8m+20$; se han introducido dos superpasos más, por lo que el coste de sincronización aumenta de $3l$ a $5l$. La comunicación se ha incrementado en $2(p-1)$ elementos (que corresponden al envío desde los procesadores remotos a P_0 de los parámetros δ_i y μ_i) por un lado y $4m'$ elementos (correspondientes a las $\frac{m'}{2}$ filas enviadas a P_1 que no serán necesarias para resolver el subsistema $\hat{A}_1\hat{x}_1=\hat{d}_1$ y las $\frac{m'}{2}$ filas enviadas a P_{p-2} no necesarias para la resolución del subsistema $\hat{A}_{\frac{p}{2}}\hat{x}_{\frac{p}{2}}=\hat{d}_{\frac{p}{2}}$) por otro; en consecuencia, el coste de comunicación vale

$$\begin{aligned}&\left[\left(5n-5k-\frac{21}{2}m+4mp+\frac{15}{2}\right)+(4m+4+2p-2)\right]g\\ =\;&\left(5n-5k-\frac{13}{2}m+4mp+2p+\frac{19}{2}\right)g,\end{aligned}$$

donde se ha considerado que en el peor de los casos $m'=m+1$.

A continuación se obtiene con detalle el coste del algoritmo 3.3.

Coste del superpaso 1. Como se ha visto en el capítulo 2 (véase la expresión (2.25) de la página 80), el coste de este superpaso es

$$1+(4n-4k-1)g+l. \tag{3.37}$$

Coste del superpaso 2. En el capítulo 2 se obtuvo el coste de este superpaso (véase (2.26), página 80), que es

$$3k+(2p-2)g+l. \tag{3.38}$$

Coste del superpaso 3. El procesador principal debe realizar tres operaciones para calcular μ, nueve operaciones para obtener el valor de m y dos operaciones para calcular

m'; en total 14 operaciones. En este superpaso, envía el valor de m a todos los procesadores, envía $4m'-1$ elementos a cada uno de los procesadores P_1 y P_{p-2}, envía $2m'$ elementos al procesador P_{p-1} y envía $4m-1$ elementos al procesador P_i, con $i = 2, 3, \ldots, p-3$, por tanto el coste de comunicación es $[p-1+2(4m'-1)+2m'+(4m-1)(p-4)]\,g$. Como en el peor de los casos $m' = m+1$, se tiene que el coste del superpaso 3 es

$$14 + (-6m + 4mp + 11)g + l. \tag{3.39}$$

Coste del superpaso 4. Puesto que este superpaso coincide con el superpaso 2 del algoritmo 3.2, su coste viene dado por la expresión (3.31).

Sumando las expresiones (3.37), (3.38),(3.39) y (3.31) se obtiene el coste del algoritmo 3.3

$$11k + 8m + 20 + \left(5n - 5k - \frac{13}{2}m + 4mp + 2p + \frac{19}{2}\right) g + 5l. \tag{3.40}$$

3.4 Resultados numéricos

En esta sección se analizan los tiempos previstos teóricamente y los tiempos experimentales de los algoritmos 3.1, 3.2 y 3.3 (a los que se referenciará en las tablas y figuras como TW, TW1 y TW2 respectivamente). Las pruebas experimentales se han realizado en el IBM SP2 y el *cluster* de PC's cuyas características se han descrito en la subsección 1.5.3. Por comodidad, en la tabla 3.1 se repiten los parámetros obtenidos para esas máquinas que se muestran en la tabla 1.1.

El tiempo teórico se ha obtenido considerando que el tamaño de bloque de los mensajes que se comunican entre los procesadores es $k = \dfrac{n}{p}$ y que el coste de comunicación de una palabra de 32 bits es

$$g(k) = \left(\frac{n_{\frac{1}{2}}}{k} + 1\right) g_\infty,$$

además se ha tomado $m = 5$. Los algoritmos 3.1 (TW), 3.2(TW1) y 3.3(TW2) han sido implementados en Fortran usando la versión v1.3 de la librería BSPLib, se ha generando el sistema (3.1) obteniendo aleatoriamente los elementos de la matriz de coeficientes A

s	p	l	g	$n_{\frac{1}{2}}$
45	1	423	2.3	26
	2	3294	9.5	25
	4	5366	12.4	25
	6	8164	12.5	25

(a) *IBM SP2 switch*

s	p	l	g	$n_{\frac{1}{2}}$
45	1	423	2.3	8
	2	20235	709.7	3
	4	54163	1362.6	9
	6	121958	3211.2	9

(b) *IBM SP2 ethernet*

s	p	l	g	$n_{\frac{1}{2}}$
16.4	1	23	0.2	22
	2	2556	6.9	5
	4	5152	7.4	4
	6	7538	6.8	4

(c) *Cluster de PC's*

Tabla 3.1: *Valores de parámetros BSP.*

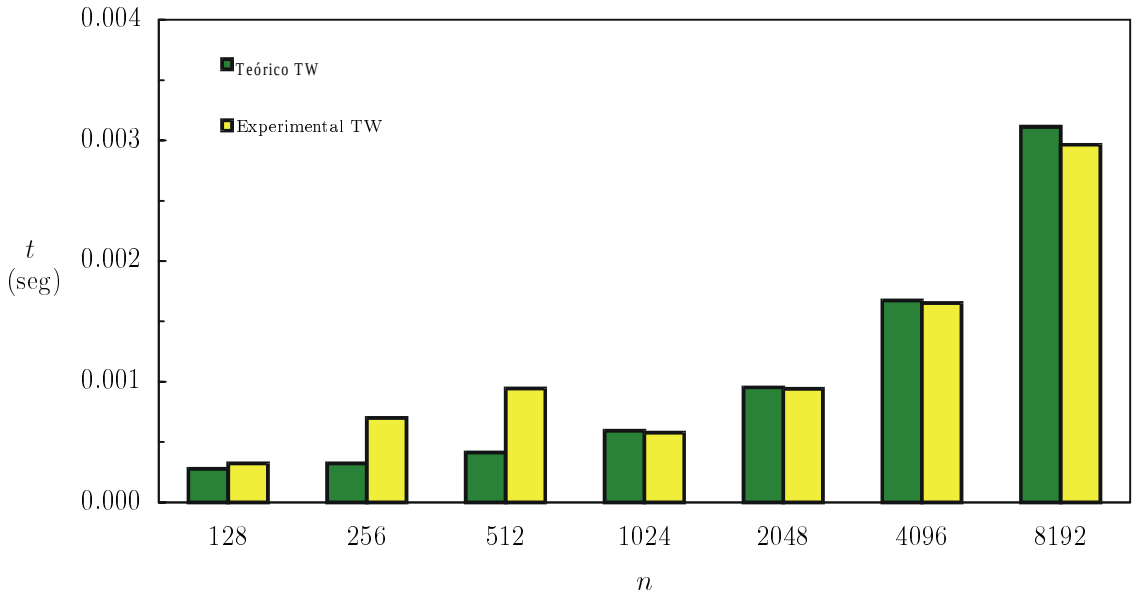
y, para simplificar, eligiendo el vector de términos independientes $\boldsymbol{d}$ de manera que la solución sea $\boldsymbol{x} = [1, 1, \ldots, 1]^T$. A la elección aleatoria de los coeficientes se le ha añadido la restricción de que $m = 5$ (véase la tabla 2.1 en la página 70).

En la tabla 3.2 y la figura 3.5 se muestran los tiempos teóricos y experimentales, medidos en segundos, del algoritmo 3.1 (TW) en el IBM SP2 con 2 procesadores utilizando *switch* y en las tablas 3.3, 3.4 y figuras 3.6, 3.7 se muestran los tiempos teóricos y experimentales, medidos en segundos, de las algoritmos 3.2(TW1) y 3.3(TW2) en el IBM SP2 para 4 y 6 procesadores utilizando *switch*.

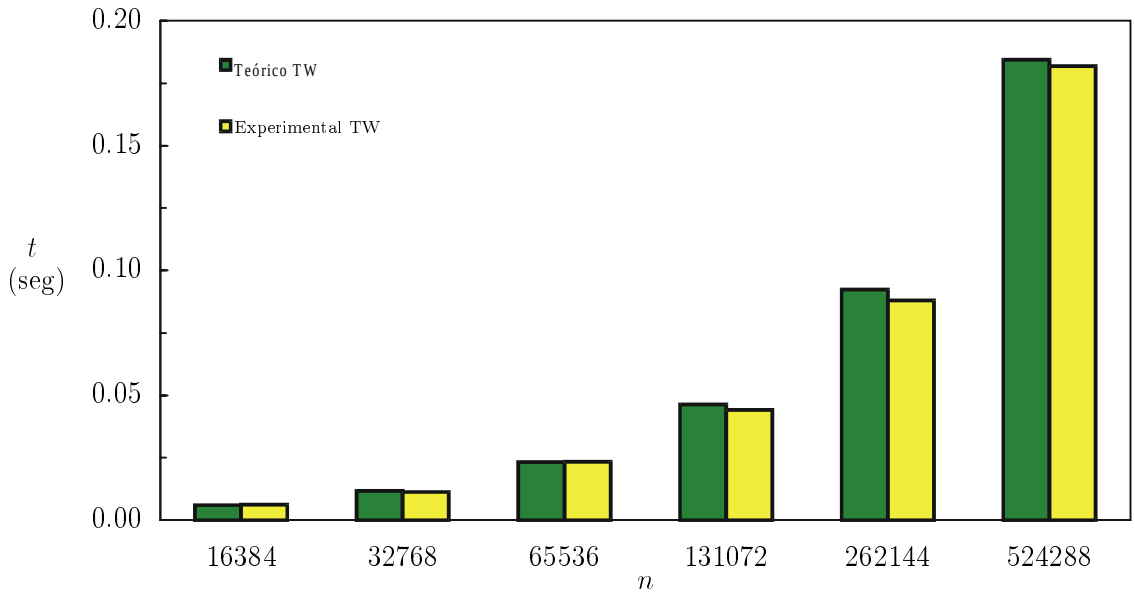
En la tabla 3.5 y la figura 3.8 se muestran los tiempos teóricos y experimentales, medidos en segundos, del algoritmo 3.1 (TW) en el IBM SP2 con 2 procesadores utilizando *ethernet* y en las tablas 3.6, 3.7 y figuras 3.9, 3.10 se muestran los tiempos teóricos y experimentales, medidos en segundos, de las algoritmos 3.2(TW1) y 3.3(TW2) en el IBM SP2 para 4 y 6 procesadores utilizando *ethernet*.

IBM SP2 2 procesadores *switch*		
	TW	
n	Teórico	Experimental
128	0.0003	0.0003
256	0.0003	0.0007
512	0.0004	0.0009
1024	0.0006	0.0006
2048	0.0010	0.0009
4096	0.0017	0.0017
8192	0.0031	0.0030
16384	0.0060	0.0062
32768	0.0117	0.0113
65536	0.0233	0.0234
131072	0.0463	0.0441
262144	0.0923	0.0880
524288	0.1845	0.1818

Tabla 3.2: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW), medidos en un IBM SP2 con 2 procesadores interconectados mediante switch, para* $128 \leq n \leq 524288$.



(a) $128 \leq n \leq 8192$

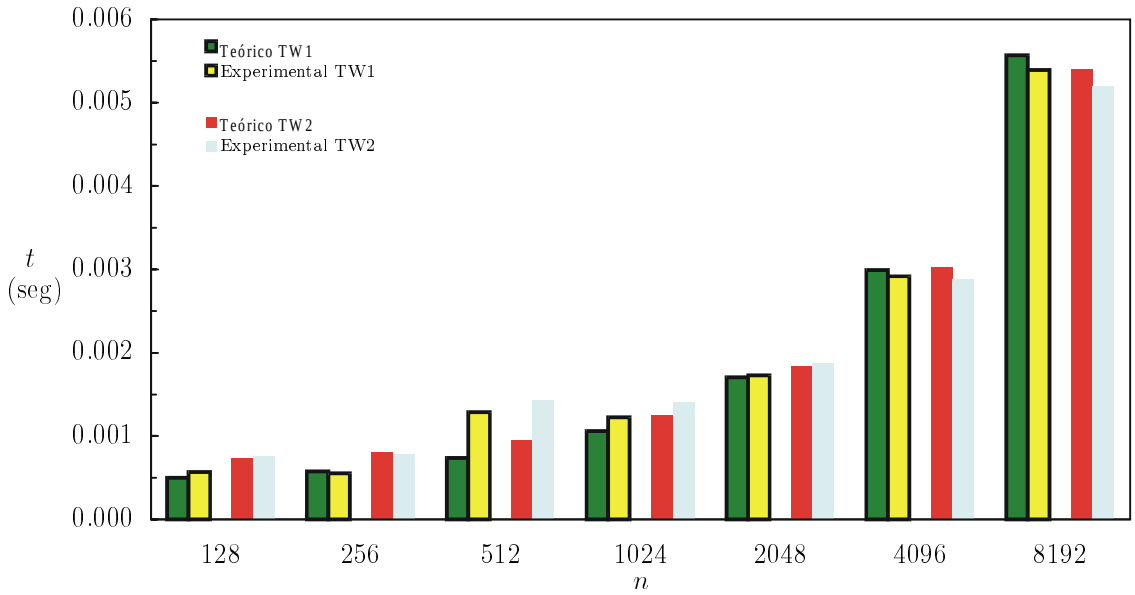


(b) $16384 \leq n \leq 524288$

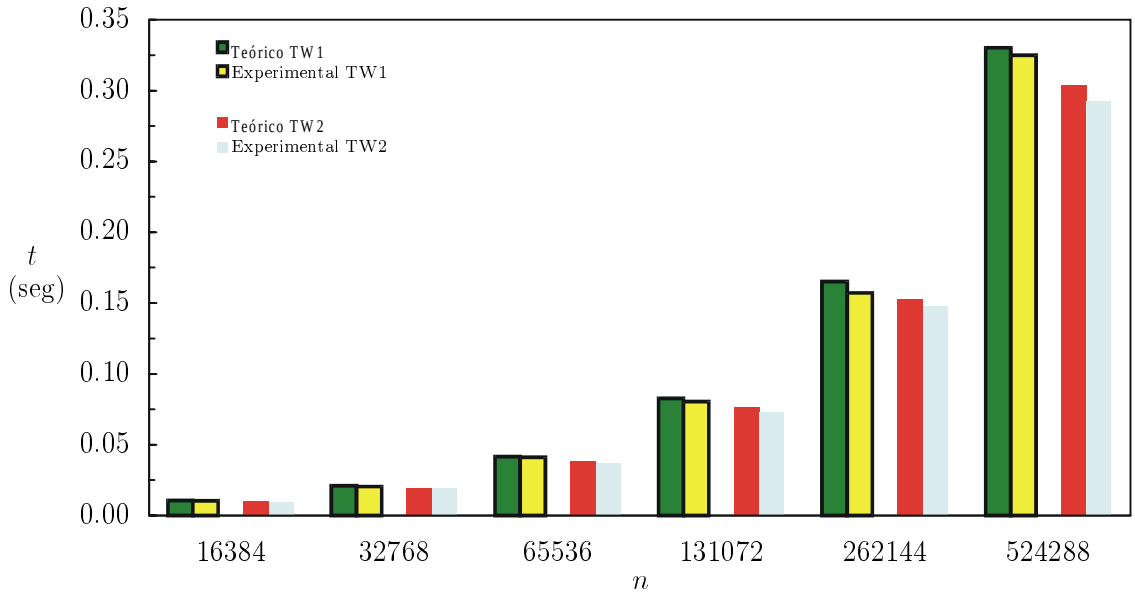
Figura 3.5: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW) en un IBM SP2 con 2 procesadores interconectados mediante switch.*

IBM SP2 4 procesadores *switch*				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
128	0.0005	0.0006	0.0007	0.0008
256	0.0006	0.0006	0.0008	0.0008
512	0.0007	0.0013	0.0010	0.0014
1024	0.0011	0.0012	0.0013	0.0014
2048	0.0017	0.0017	0.0018	0.0019
4096	0.0030	0.0029	0.0030	0.0029
8192	0.0056	0.0054	0.0054	0.0052
16384	0.0107	0.0105	0.0101	0.0098
32768	0.0210	0.0206	0.0196	0.0193
65536	0.0416	0.0413	0.0386	0.0372
131072	0.0829	0.0805	0.0766	0.0735
262144	0.1653	0.1572	0.1525	0.1478
524288	0.3302	0.3249	0.3043	0.2928

Tabla 3.3: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2), medidos en un IBM SP2 con 4 procesadores interconectados mediante switch, para* $128 \leq n \leq 524288$.



(a) $128 \leq n \leq 8192$

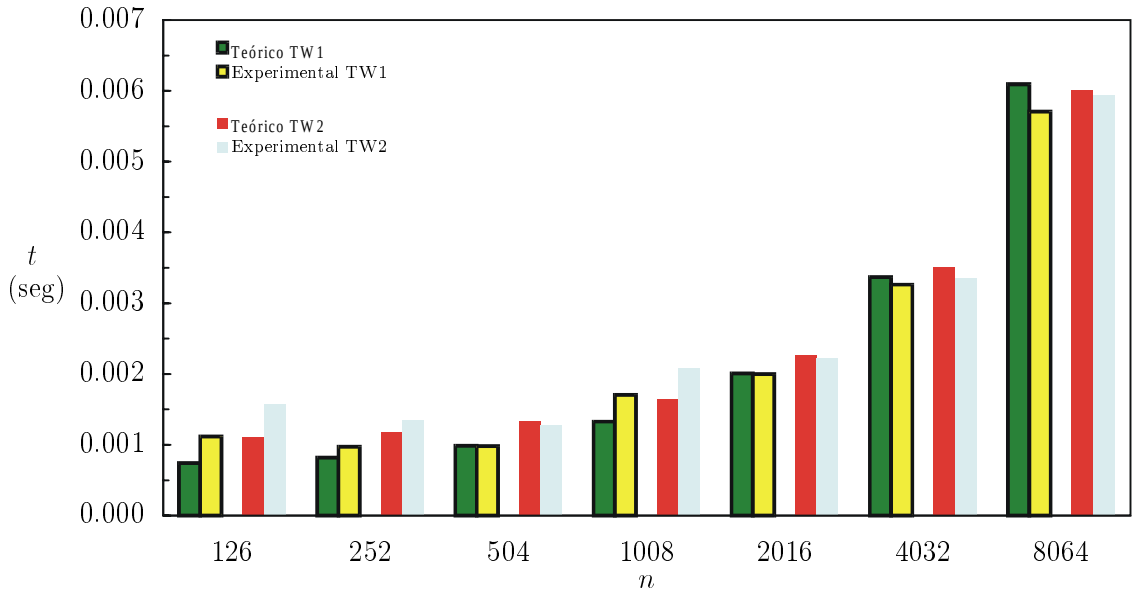


(b) $16384 \leq n \leq 524288$

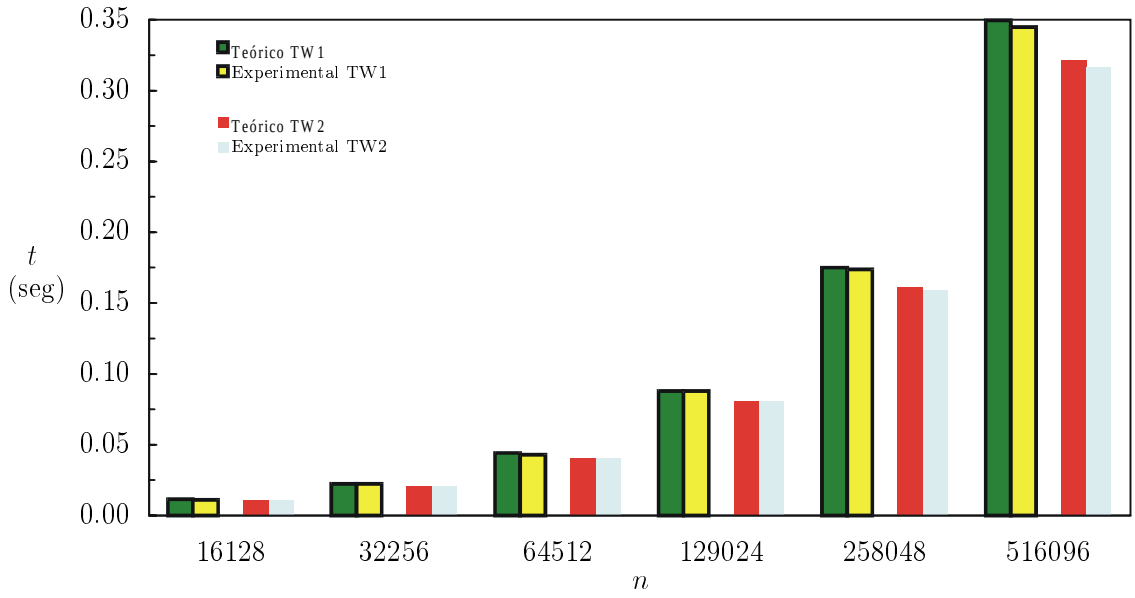
Figura 3.6: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con 4 procesadores interconectados mediante switch.*

IBM SP2 6 procesadores *switch*				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
126	0.0007	0.0011	0.0011	0.0016
252	0.0008	0.0010	0.0012	0.0013
504	0.0010	0.0010	0.0013	0.0013
1008	0.0013	0.0017	0.0016	0.0021
2016	0.0020	0.0020	0.0023	0.0022
4032	0.0034	0.0033	0.0035	0.0034
8064	0.0061	0.0057	0.0060	0.0059
16128	0.0115	0.0112	0.0110	0.0107
32256	0.0225	0.0224	0.0210	0.0207
64512	0.0443	0.0429	0.0410	0.0409
129024	0.0879	0.0878	0.0811	0.0810
258048	0.1751	0.1739	0.1612	0.1594
516096	0.3496	0.3450	0.3213	0.3168

Tabla 3.4: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2), medidos en un IBM SP2 con* 6 *procesadores interconectados mediante switch, para* $126 \leq n \leq 516096$.



(a) $126 \leq n \leq 8064$

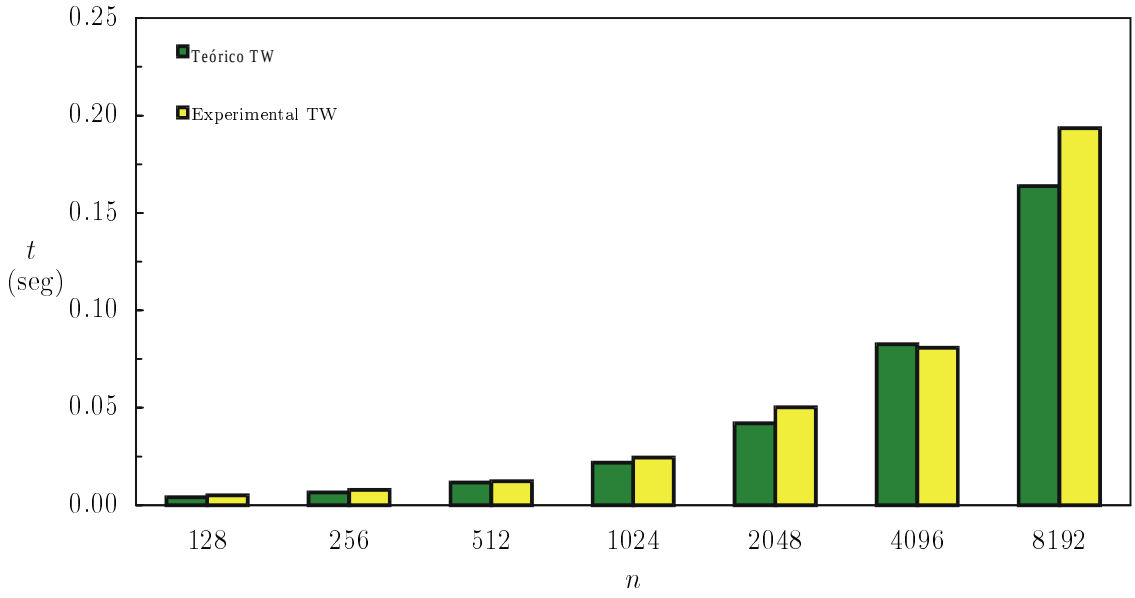


(b) $16128 \leq n \leq 516096$

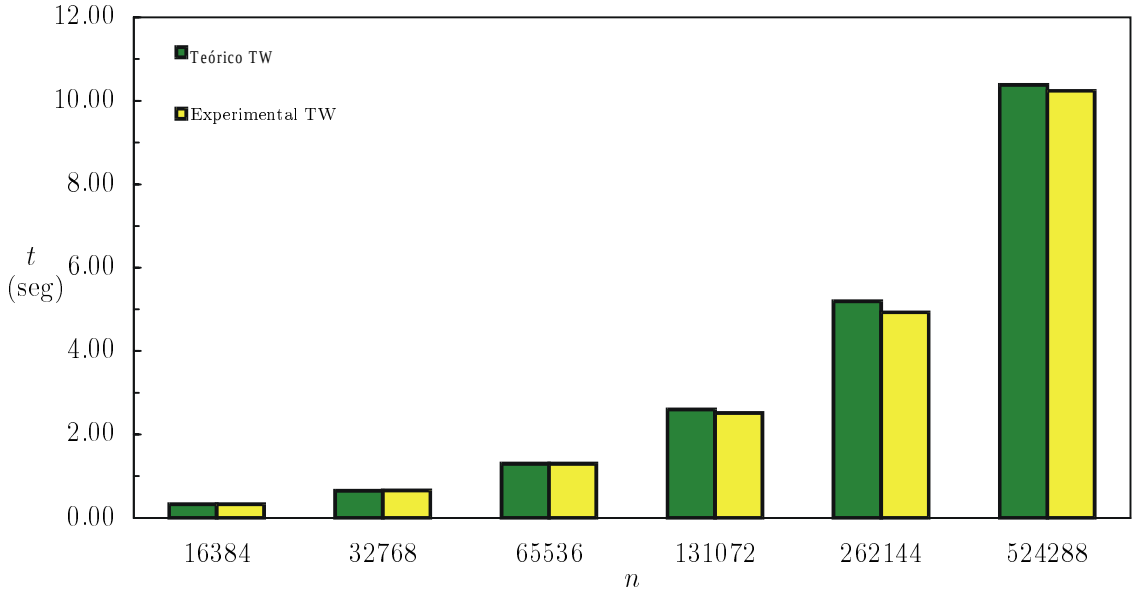
Figura 3.7: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con 6 procesadores interconectados mediante switch.*

IBM SP2 2 procesadores *ethernet*		
	TW	
n	Teórico	Experimental
128	0.0040	0.0051
256	0.0065	0.0078
512	0.0116	0.0122
1024	0.0217	0.0245
2048	0.0420	0.0503
4096	0.0826	0.0807
8192	0.1637	0.1935
16384	0.3259	0.3267
32768	0.6503	0.6570
65536	1.2992	1.2962
131072	2.5969	2.5170
262144	5.1923	4.9264
524288	10.3831	10.2384

Tabla 3.5: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW), medidos en un IBM SP2 con 2 procesadores interconectados mediante ethernet, para* $128 \leq n \leq 524288$.



(a) $128 \leq n \leq 8192$

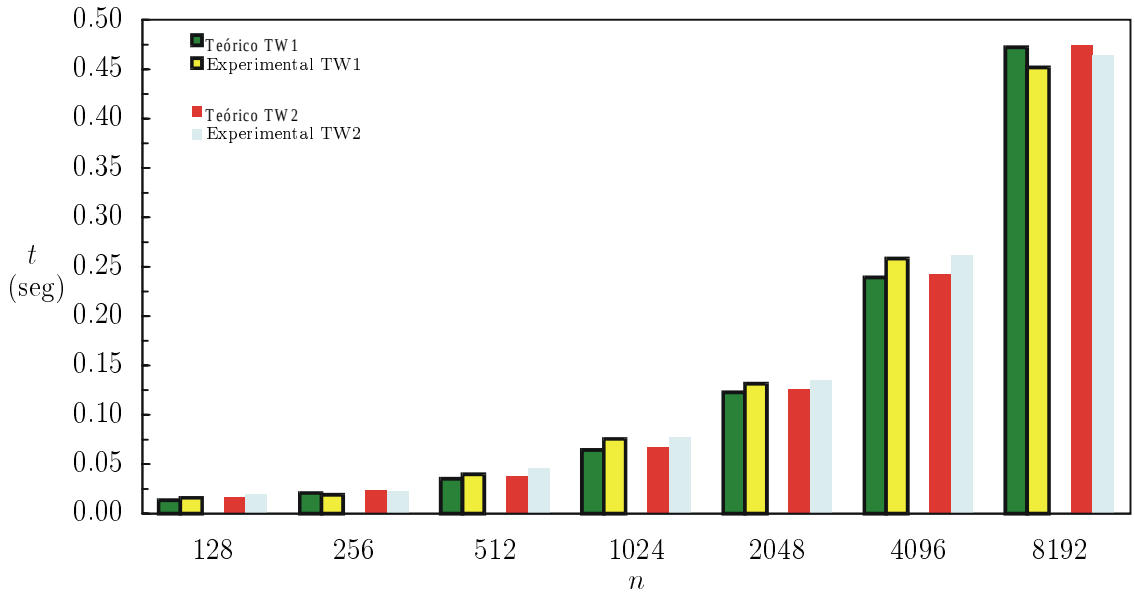


(b) $16384 \leq n \leq 524288$

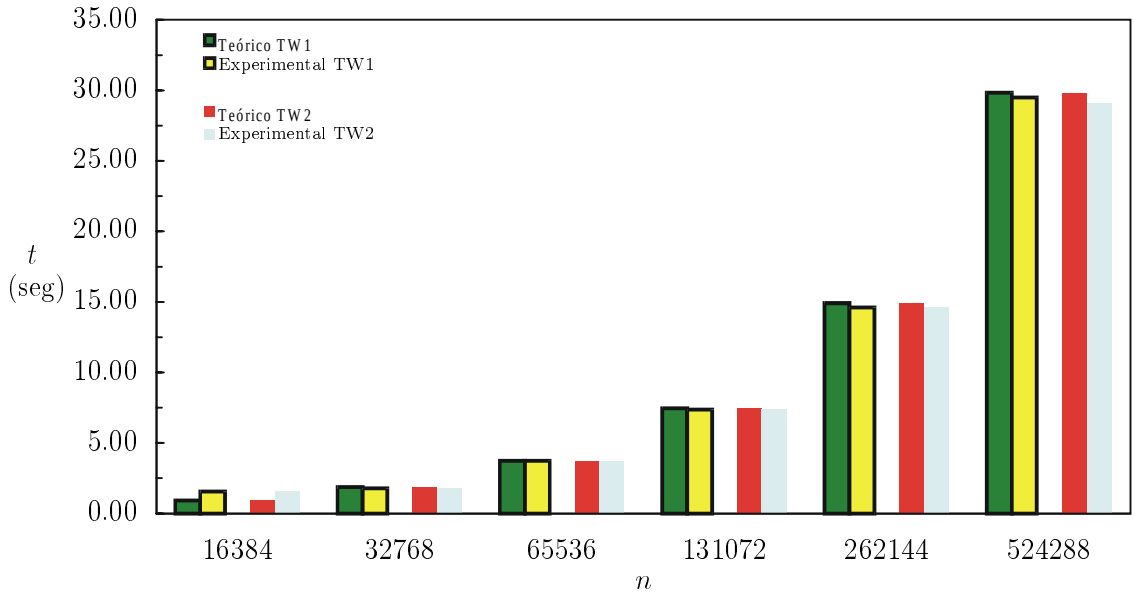
Figura 3.8: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW) en un IBM SP2 con 2 procesadores interconectados mediante ethernet.*

IBM SP2 4 procesadores *ethernet*				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
128	0.0136	0.0159	0.0167	0.0199
256	0.0208	0.0192	0.0238	0.0223
512	0.0353	0.0397	0.0383	0.0457
1024	0.0645	0.0756	0.0673	0.0776
2048	0.1227	0.1317	0.1255	0.1353
4096	0.2392	0.2581	0.2419	0.2614
8192	0.4722	0.4516	0.4747	0.4645
16384	0.9382	1.5570	0.9403	1.5768
32768	1.8703	1.7849	1.8716	1.8026
65536	3.7344	3.7286	3.7340	3.7290
131072	7.4625	7.3648	7.4589	7.4057
262144	14.9188	14.6156	14.9087	14.6577
524288	29.8315	29.5015	29.8082	29.0884

Tabla 3.6: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2), medidos en un IBM SP2 con 4 procesadores interconectados mediante ethernet, para* $128 \leq n \leq 524288$.



(a) $128 \leq n \leq 8192$

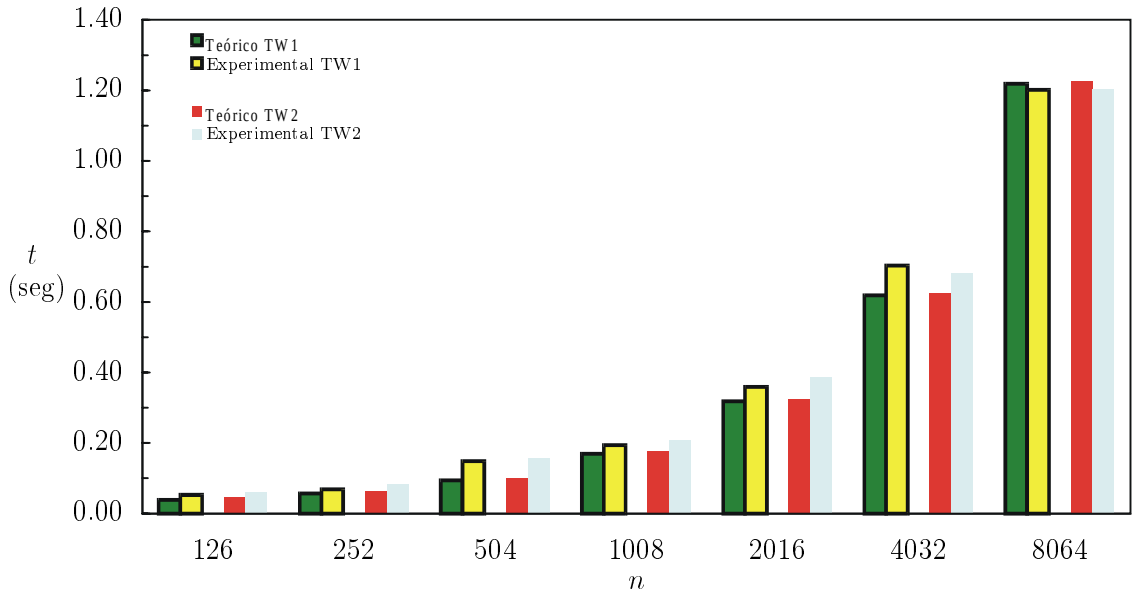


(b) $16384 \leq n \leq 524288$

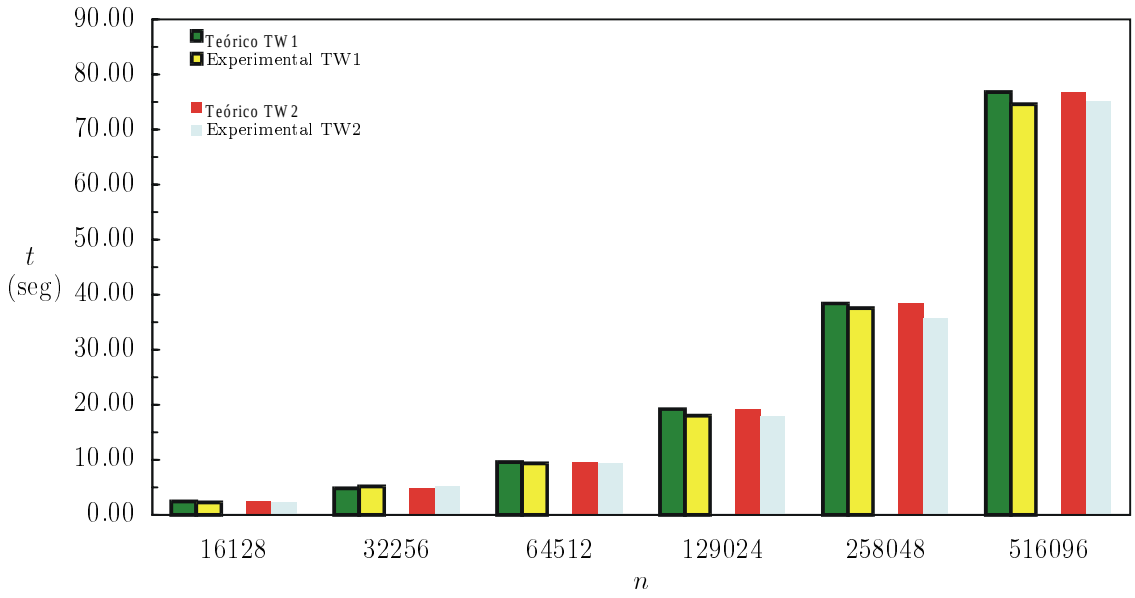
Figura 3.9: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con 4 procesadores interconectados mediante ethernet.*

IBM SP2 6 procesadores *switch*				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
126	0.0387	0.0530	0.0462	0.0613
252	0.0569	0.0693	0.0640	0.0826
504	0.0941	0.1485	0.1011	0.1578
1008	0.1689	0.1939	0.1758	0.2076
2016	0.3188	0.3595	0.3256	0.3873
4032	0.6187	0.7034	0.6253	0.6810
8064	1.2185	1.2018	1.2249	1.2027
16128	2.4181	2.2668	2.4241	2.2816
32256	4.8173	5.1613	4.8224	5.2240
64512	9.6158	9.3562	9.6191	9.4004
129024	19.2128	17.9982	19.2125	17.9316
258048	38.4068	37.5757	38.3993	35.7255
516096	76.7948	74.5992	76.7730	75.1931

Tabla 3.7: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2), medidos en un IBM SP2 con 6 procesadores interconectados mediante ethernet, para* $126 \leq n \leq 516096$.



(a) $126 \leq n \leq 8064$



(b) $16128 \leq n \leq 516096$

Figura 3.10: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con 6 procesadores interconectados mediante ethernet.*

En la tabla 3.8 y la figura 3.11 se muestran los tiempos teóricos y experimentales, medidos en segundos, del algoritmo 3.1 (TW) en el *cluster* de PC's para 2 procesadores y en las tablas 3.9, 3.10 y figuras 3.12, 3.13 se muestran los tiempos teóricos y experimentales, medidos en segundos, de las algoritmos 3.2(TW1) y 3.3(TW2) en el *cluster* de PC's para 4 y 6 procesadores.

Los tiempos han sido obtenidos para diferentes tamaños de la matriz de coeficientes, en el IBM SP2 el tamaño varía desde 128 a 524288 para 2 y 4 procesadores y desde 126 a 516096 para 6 procesadores; en el *cluster* de PC's el tamaño de la matriz de coeficientes varía desde 128 a 65536 para 2 y 4 procesadores y desde 126 a 64512 para 6 procesadores.

En las figuras 3.14, 3.15 y 3.16 se muestra el porcentaje de desviación del tiempo experimental con respecto al teórico en función del tamaño n del sistema.

Las mayores diferencias entre el tiempo previsto y el experimental se obtienen en los sistemas de tamaño pequeño en los que, por lo general, suele ser mayor el tiempo experimental que el teórico. A medida que aumenta el tamaño de sistema el tiempo obtenido experimentalmente se ajusta mejor al esperado, para tamaños grandes la desviación (salvo alguna excepción) oscila entre el 0% y el 5%.

En las tablas 3.11, 3.12 y 3.13 se muestra el algoritmo más rápido (teórica y experimentalmente) entre los algoritmos 3.2 (TW1) y 3.3 (TW2), en cada una de las máquinas para los distintos tamaños de sistema. Se observa que en el IBM SP2 utilizando *switch* es mejor calcular m en paralelo (esto es, el algoritmo 3.3) para valores de $n > 4100$. En el IBM SP2 utilizando *ethernet* y en el *cluster* de PC's es mejor el algoritmo 3.3 para tamaños de sistema mayores, por encima de 50000 ecuaciones.

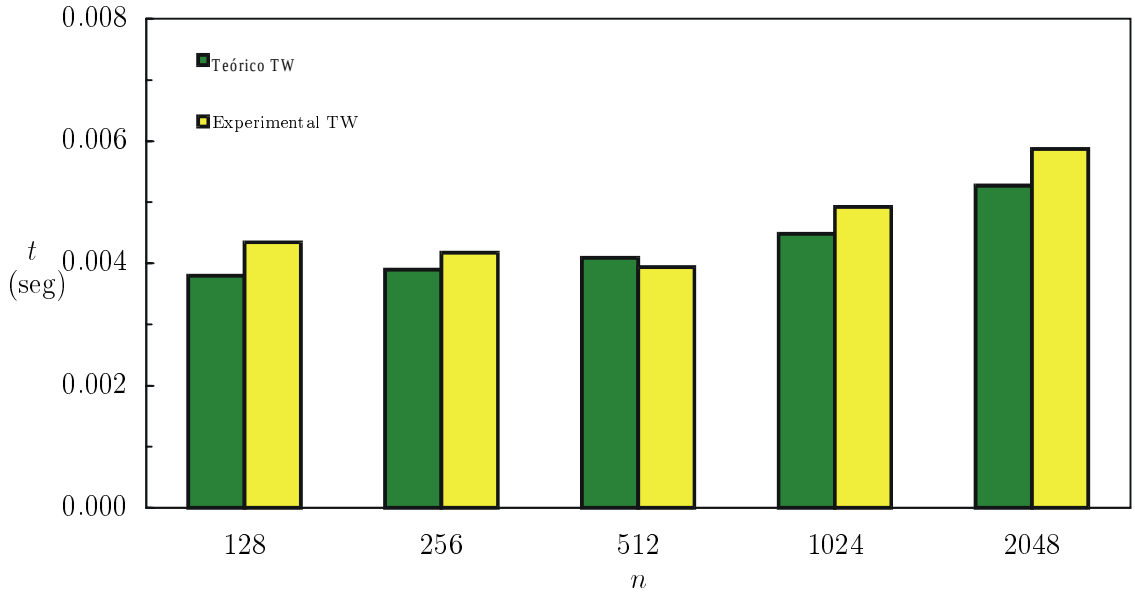
En la tabla 3.14 se muestra el *speed-up* y la eficiencia de los algoritmos 3.1 (TW), 3.2(TW1) y 3.3(TW2) con respecto al método de eliminación de Gauss para sistemas tridiagonales, que es el mejor algoritmo secuencial para la resolución de sistemas tridiagonales, considerando los tiempos teóricos de los sistemas de tamaño máximo de los que se han obtenido resultados experimentales en cada una de las máquinas.

La eficiencia que se obtiene no es muy buena, y en caso del IBM SP2 con *ethernet* es pésima; sin duda un factor determinante es el valor de g. Para otras máquinas con menor valor de g se obtiene mejor eficiencia, como es el caso de un CRAY T3D o un CRAY T3E. Por comodidad, en la tabla 3.15 se repiten los valores de los parámetros de estas

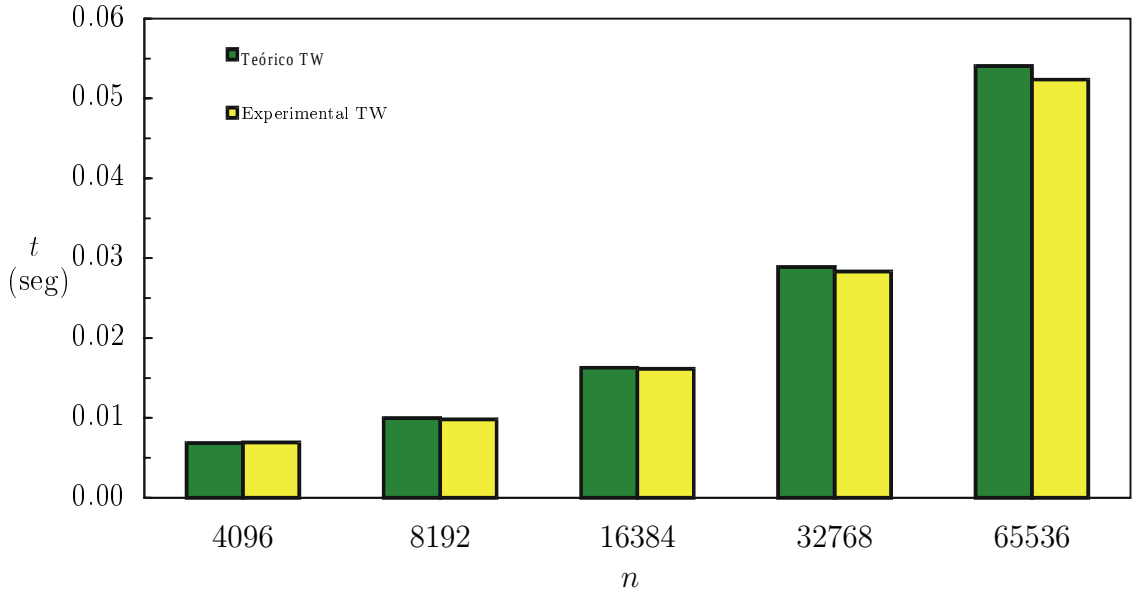
Cluster de PC's 2 procesadores		
	TW	
n	Teórico	Experimental
128	0.0038	0.0043
256	0.0039	0.0042
512	0.0041	0.0039
1024	0.0045	0.0049
2048	0.0053	0.0059
4096	0.0068	0.0069
8192	0.0100	0.0098
16384	0.0163	0.0161
32768	0.0289	0.0283
65536	0.0541	0.0523

Tabla 3.8: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW) medidos en un cluster de PC's, para 2 procesadores y* $128 \leq n \leq 65536$.

máquinas que se muestran en la tabla 2.16, para las mismas se obtiene el *speed-up* y la eficiencia que figura en la tabla 3.16 considerando el mismo tamaño de sistema que para la tabla 3.14.



(a) $128 \leq n \leq 2048$

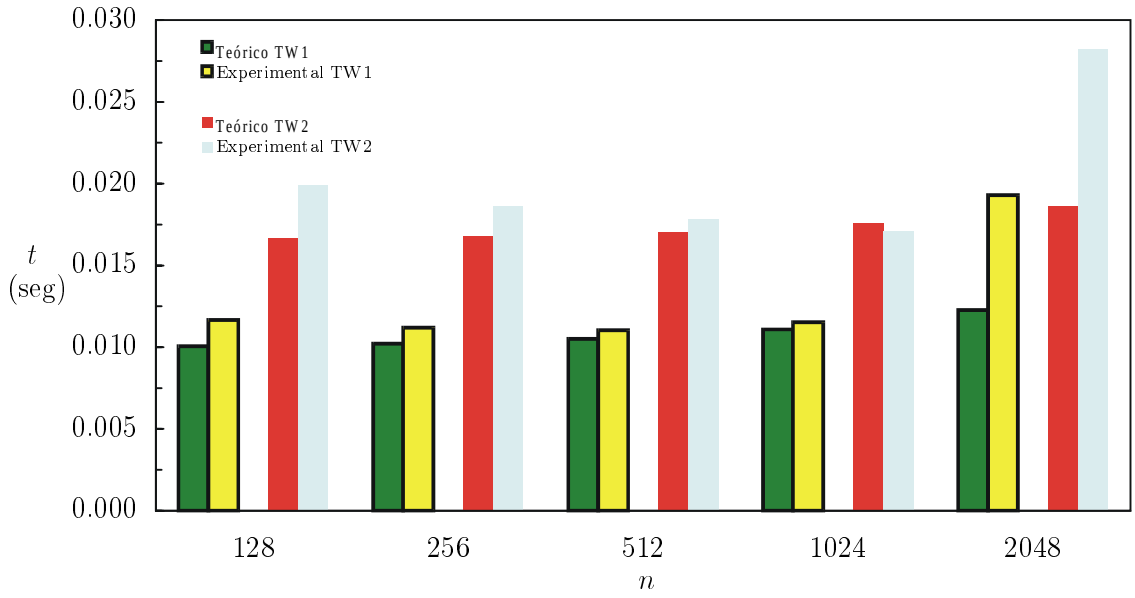


(b) $4096 \leq n \leq 65536$

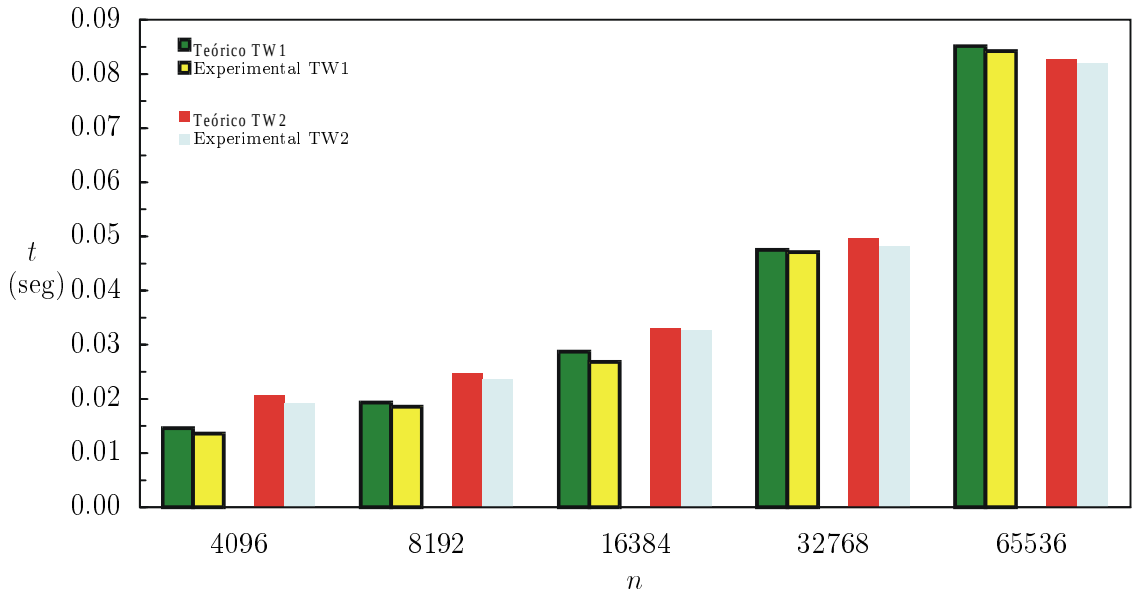
Figura 3.11: *Tiempos teóricos y experimentales del algoritmo 3.1 (TW) en un cluster de PC's para 2 procesadores.*

Cluster de PC's 4 procesadores				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
128	0.0101	0.0116	0.0166	0.0199
256	0.0102	0.0112	0.0168	0.0186
512	0.0105	0.0110	0.0170	0.0178
1024	0.0111	0.0115	0.0175	0.0171
2048	0.0123	0.0193	0.0186	0.0282
4096	0.0146	0.0136	0.0206	0.0192
8192	0.0193	0.0186	0.0248	0.0237
16384	0.0287	0.0268	0.0331	0.0326
32768	0.0475	0.0471	0.0496	0.0481
65536	0.0851	0.0842	0.0828	0.0820

Tabla 3.9: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) medidos en un cluster de PC's, para* 4 *procesadores y* $128 \leq n \leq 65536$.



(a) $128 \leq n \leq 2048$

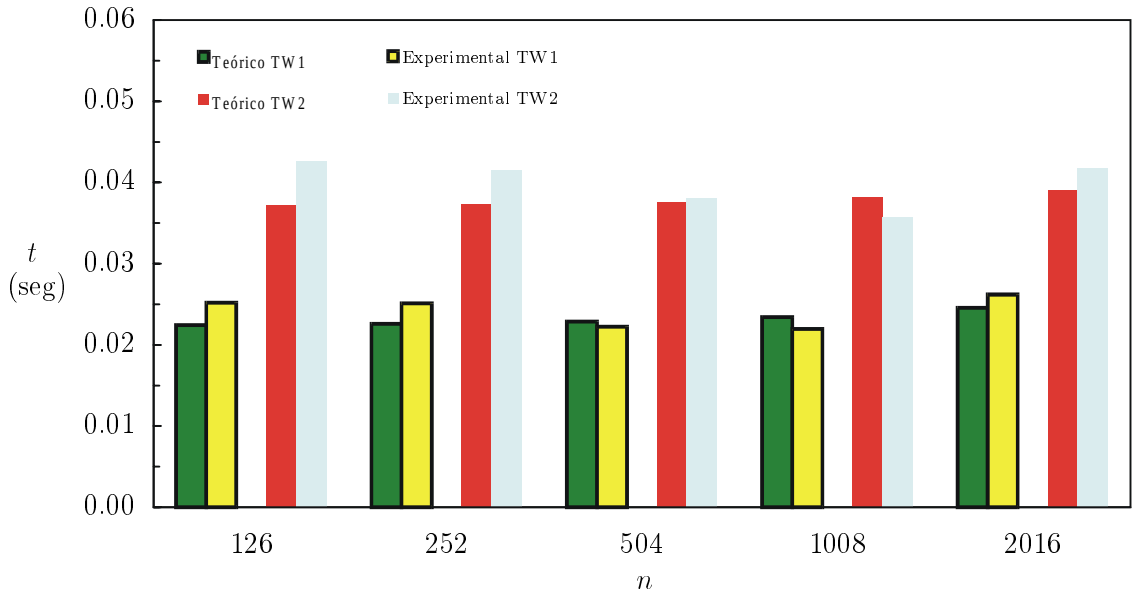


(b) $4096 \leq n \leq 65536$

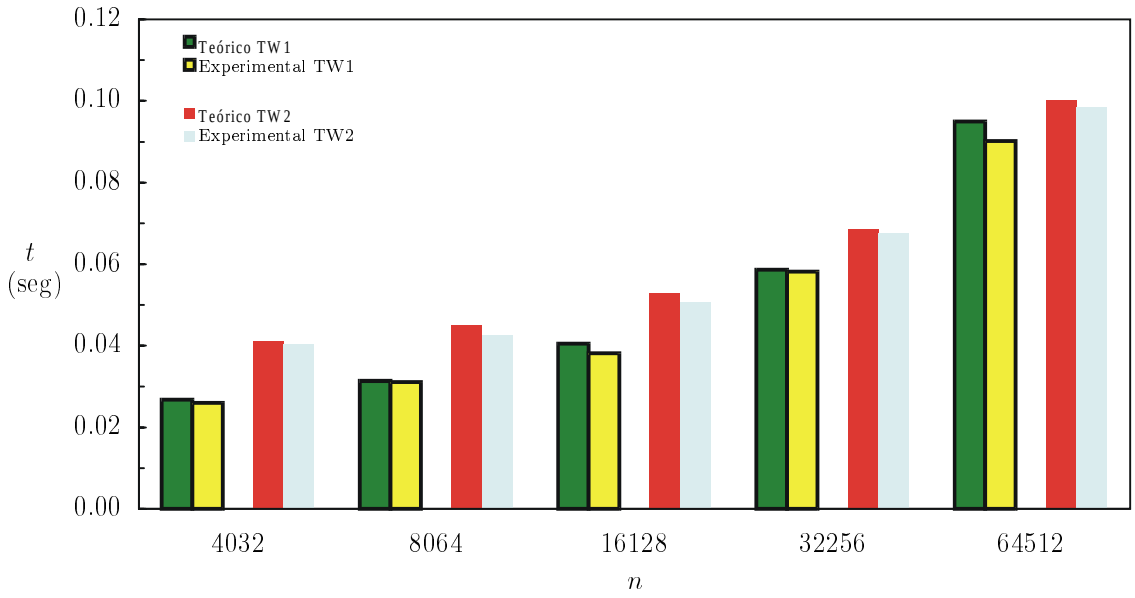
Figura 3.12: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un cluster de PC's para 4 procesadores.*

Cluster de PC's 6 procesadores				
	TW1		TW2	
n	Teórico	Experimental	Teórico	Experimental
126	0.0224	0.0252	0.0373	0.0426
252	0.0226	0.0251	0.0374	0.0416
504	0.0229	0.0223	0.0376	0.0381
1008	0.0234	0.0220	0.0381	0.0357
2016	0.0246	0.0262	0.0391	0.0417
4032	0.0268	0.0260	0.0411	0.0403
8064	0.0314	0.0311	0.0450	0.0425
16128	0.0405	0.0382	0.0529	0.0506
32256	0.0586	0.0582	0.0686	0.0675
64512	0.0950	0.0902	0.1000	0.0984

Tabla 3.10: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) medidos en un cluster de PC's, para* 6 *procesadores y* $126 \leq n \leq 64512$.

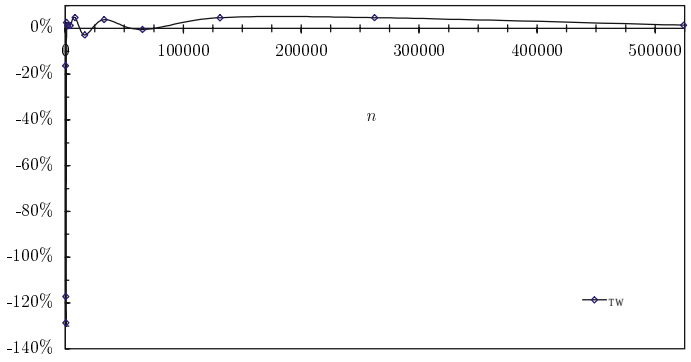


(a) $126 \leq n \leq 2016$

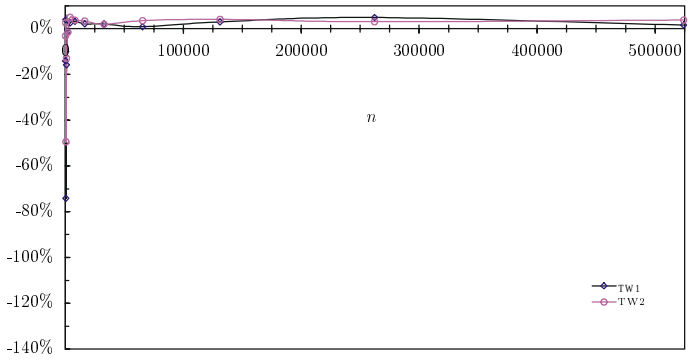


(b) $4032 \leq n \leq 64512$

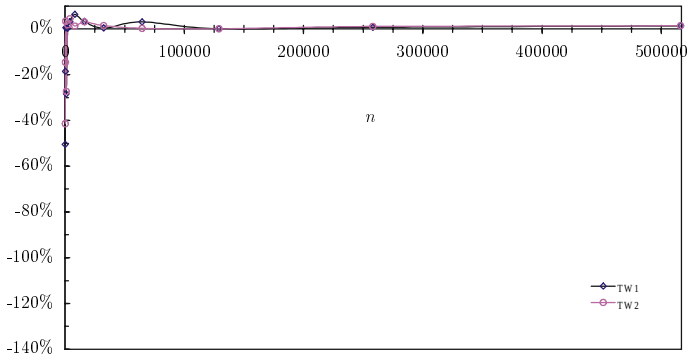
Figura 3.13: *Tiempos teóricos y experimentales de los algoritmos 3.2 (TW1) y 3.3 (TW2) en un cluster de PC's para* 6 *procesadores.*



(a) 2 *procesadores*

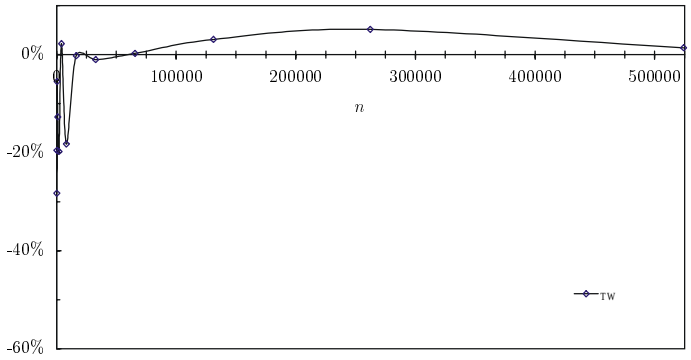


(b) 4 *procesadores*

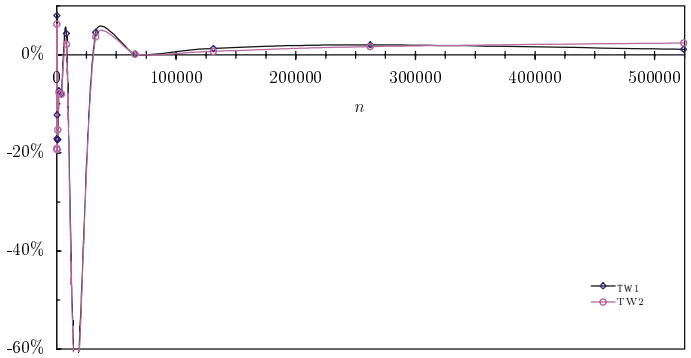


(c) 6 *procesadores*

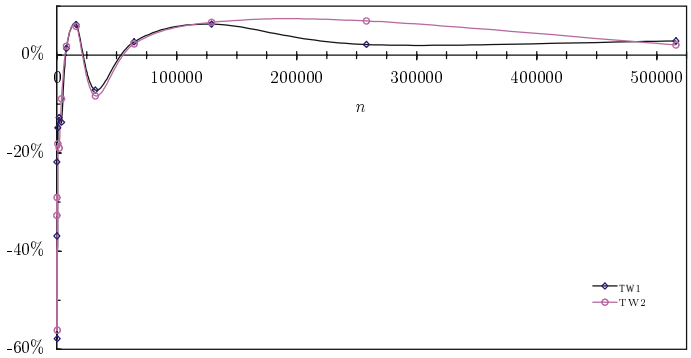
Figura 3.14: *Diferencias porcentuales entre los tiempos teóricos y experimentales de los algoritmos 3.1 (TW), 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con switch.*



(a) 2 *procesadores*

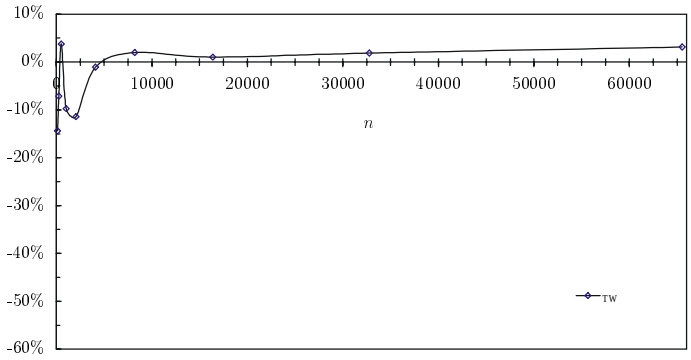


(b) 4 *procesadores*

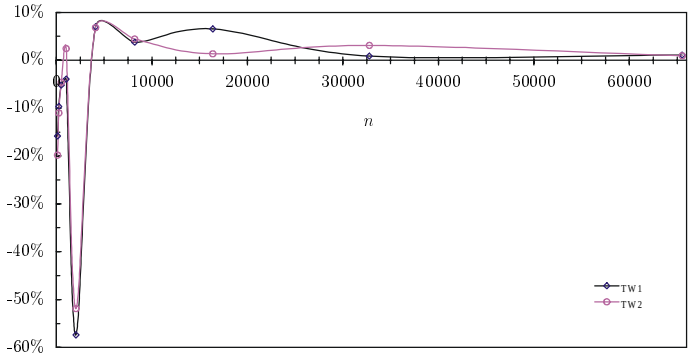


(c) 6 *procesadores*

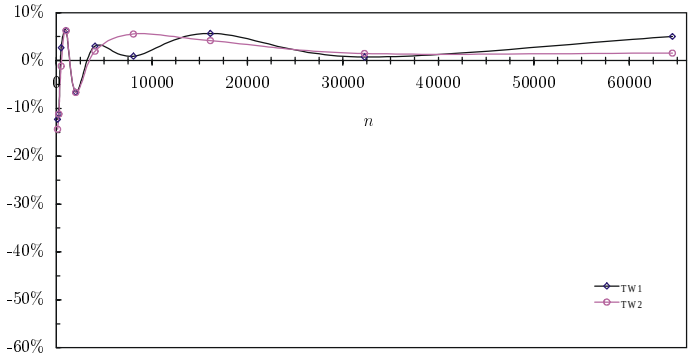
Figura 3.15: *Diferencias porcentuales entre los tiempos teóricos y experimentales de los algoritmos 3.1 (TW), 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 con ethernet.*



(a) 2 *procesadores*



(b) 4 *procesadores*



(c) 6 *procesadores*

Figura 3.16: *Diferencias porcentuales entre los tiempos teóricos y experimentales de los algoritmos 3.1 (TW), 3.2 (TW1) y 3.3 (TW2) en un cluster de PC's.*

IBM SP2 *switch*					
	4 procesadores			6 procesadores	
n	Teór.	Exper.	n	Teór.	Exper.
128	TW1	TW1	126	TW1	TW1
256	TW1	TW1	252	TW1	TW1
512	TW1	TW1	504	TW1	TW1
1024	TW1	TW1	1008	TW1	TW1
2048	TW1	TW1	2016	TW1	TW1
4096	TW1	TW2	4032	TW1	TW1
8192	TW2	TW2	8064	TW2	TW1
16384	TW2	TW2	16128	TW2	TW2
32768	TW2	TW2	32256	TW2	TW2
65536	TW2	TW2	64512	TW2	TW2
131072	TW2	TW2	129024	TW2	TW2
262144	TW2	TW2	258048	TW2	TW2
524288	TW2	TW2	516096	TW2	TW2

Tabla 3.11: *Algoritmo más rápido (teórica y experimentalmente) entre los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 switch.*

IBM SP2 *switch*					
	4 procesadores			6 procesadores	
n	Teór.	Exper.	n	Teór.	Exper.
128	TW1	TW1	126	TW1	TW1
256	TW1	TW1	252	TW1	TW1
512	TW1	TW1	504	TW1	TW1
1024	TW1	TW1	1008	TW1	TW1
2048	TW1	TW1	2016	TW1	TW1
4096	TW1	TW1	4032	TW1	TW2
8192	TW1	TW1	8064	TW1	TW1
16384	TW1	TW1	16128	TW1	TW1
32768	TW1	TW1	32256	TW1	TW1
65536	TW2	TW1	64512	TW1	TW1
131072	TW2	TW1	129024	TW2	TW2
262144	TW2	TW1	258048	TW2	TW2
524288	TW2	TW2	516096	TW2	TW1

Tabla 3.12: *Algoritmo más rápido (teórica y experimentalmente) entre los algoritmos 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 ethernet.*

Cluster de PC's					
	4 procesadores			6 procesadores	
n	Teór.	Exper.	n	Teór.	Exper.
128	TW1	TW1	126	TW1	TW1
256	TW1	TW1	252	TW1	TW1
512	TW1	TW1	504	TW1	TW1
1024	TW1	TW1	1008	TW1	TW1
2048	TW1	TW1	2016	TW1	TW1
4096	TW1	TW1	4032	TW1	TW1
8192	TW1	TW1	8064	TW1	TW1
16384	TW1	TW1	16128	TW1	TW1
32768	TW1	TW1	32256	TW1	TW1
65536	TW2	TW2	64512	TW1	TW1

Tabla 3.13: *Algoritmo más rápido (teórica y experimentalmente) entre los algoritmos 3.2 (TW1) y 3.3 (TW2) en un cluster de PC's.*

IBM SP2 *switch*				
	TW o TW1		TW2	
p	S_p	E_p	S_p	E_p
2	0.51	25.26%		
4	0.28	7.06%	0.31	7.66%
6	0.26	4.37%	0.29	4.76%

(a) *IBM SP2 switch*

IBM SP2 *ethernet*				
	TW o TW1		TW2	
p	S_p	E_p	S_p	E_p
2	0.01	0.45%		
4	0.00	0.08%	0.00	0.08%
6	0.00	0.02%	0.00	0.02%

(b) *IBM SP2 ethernet*

Cluster de PC's				
	TW o TW1		TW2	
p	S_p	E_p	S_p	E_p
2	0.59	29.50%		
4	0.37	9.36%	0.39	9.63%
6	0.33	5.51%	0.31	5.23%

(c) *Cluster de PC's*

Tabla 3.14: *Speed-up (S_p) y eficiencia (E_p) de los algoritmos 3.1 (TW), 3.2 (TW1) y 3.3 (TW2) en un IBM SP2 y un cluster de PC's.*

s	p	l	g	$n_{\frac{1}{2}}$
12	1	68	0.3	94
	2	164	0.7	71
	4	168	0.7	66
	8	175	0.8	59
	16	181	0.9	61
	32	201	1.1	28
	64	148	1	27
	128	301	1.1	20
	256	387	1.2	15

(a) *CRAY T3D*

s	p	l	g	$n_{\frac{1}{2}}$
46.7	1	86	2.12	9
	2	269	0.87	33
	4	357	0.87	40
	8	506	0.81	40
	16	751	1.04	38
	32	1252	1.31	45

(b) *CRAY T3E*

Tabla 3.15: *Valores de parámetros BSP.*

CRAY T3D				
	TW o TW1		TW2	
p	S_p	E_p	S_p	E_p
2	1.96	97.84%		
4	1.27	31.67%	1.97	49.20%
8	1.39	17.38%	2.56	31.96%
16	1.42	8.91%	2.85	17.84%
32	1.35	4.22%	2.65	8.29%
64	1.43	2.23%	3.03	4.73%
128	1.38	1.08%	2.82	2.21%
256	1.32	0.52%	2.61	1.02%

(a) *CRAY T3D*

CRAY T3E				
	TW o TW1		TW2	
p	S_p	E_p	S_p	E_p
2	1.57	78.60%		
4	1.17	29.27%	1.73	43.32%
8	1.39	17.31%	2.54	31.72%
16	1.35	8.41%	2.55	15.95%
32	1.22	7.60%	2.12	13.25%

(b) *CRAY T3E*

Tabla 3.16: *Speed-up (S_p) y eficiencia (E_p) de los algoritmos 3.1 (TW), 3.2 (TW1) y 3.3 (TW2) en un CRAY T3D y un CRAY T3E.*