A parallel method for solving pentadiagonal systems of linear equations

Ivan G. Ivanov[a1] **and Chris Walshaw**[b]

[a] *Faculty of Mathematics and Informatics, University of Shoumen, Shoumen, Bulgaria*

[b] *School of Computing and Mathematical Sciences, University of Greenwich, London, England*

Abstract

A new parallel approach for solving a pentadiagonal linear system is presented. The parallel partition method for this system and the TW parallel partition method on a chain of P processors are introduced and discussed. The result of this algorithm is a reduced pentadiagonal linear system of order $P-2$ compared with a system of order $2P-2$ for the parallel partition method. More importantly the new method involves only half the number of communications startups than the parallel partition method (and other standard parallel methods) and hence is a far more efficient parallel algorithm.

Keywords. parallel algorithm, linear system, pentadiagonal matrix, block tridiagonal matrix

1. Introduction

1.1. The parallel solution of pentadiagonal systems

Many problems in mathematics and applied science require the solution of linear systems having pentadiagonal coefficient matrices, for example the solution of certain partial differential equations, spline approximation, etc. [6]. This kind of linear system is a special class of narrow banded linear systems. There are various parallel methods for computing the solution of general narrow banded linear systems: the divide and conquer algorithm (dca) [4, 8]; the single-width separator algorithm (swsa) [5, 7, 10, 16]; the double-width separator algorithm (dwsa) [5, 16]. These methods have a common structure and three phases: factorization, solution of the reduced system and back substitution. The solution of the reduced system is the crucial step of these algorithms [3] because the system is solved sequentially.

In addition, in many applications the given narrow banded linear system has a special banded structure, e.g. periodically banded [2]. Furthermore, there is the method of Sameh [12, 13] which does not use communication in the reduction phase and the result is a pentadiagonal linear system. For instance in spline approximation a banded circulant linear system is obtained [6]. In other cases we have to solve a block pentadiagonal system [1, 6].

Here we describe three parallel methods for solving a general pentadiagonal linear system

[1]This work was carried out while the author was visiting the University of Greenwich, London, funded by a Tempus grant

- the TW matrix factorization method designed for pairs of parallel computers (section 2)
- the parallel Partition Algorithm for Pentadiagonal Systems ($PAPS$) designed for a chain of processors (section 3)
- the TW Partition Algorithm for Pentadiagonal Systems ($TWPAPS$). This method combines the two previous methods and works for a chain of $\frac{P}{2}$ pairs of processors i.e. for P processors (section 4)

These three parallel methods are modifications of a previous algorithm introduced and discussed by Walshaw and Farr [14, 15] for a tridiagonal matrix system. The third method, $TWPAPS$, solves the reduced pentadiagonal system in parallel. Finally, we compare the algorithms - dca, swsa, dwsa, Walshaw's Partition Algorithm for Tridiagonal Systems ($PATS$) referred to as $LUTW$ in [14] and the TW Partition Algorithm for Tridiagonal Systems ($TWPATS$) referred to as $TWTW$ in [14] and the $TWPAPS$ and $PAPS$ algorithms. We discuss a numerical implementation and the results from numerical experiments.

1.2. Complexity analysis

In our paper we use the following machine dependent parameters [11, 14]

- t_a - time for one floating point operation: $+,\ -,\ \times,\ \div$
- s_c - start-up time for each inter-processor communication
- t_c - time to transmit one number.

For matrices with multiple right hand sides or which remain constant throughout many iterations much work can be saved by generating a matrix decomposition once and storing it. We use square brackets notation, [ ], to denote operations which need only be carried out once for constant matrices rather than at every iteration. For multiple right hand sides the costs that are not in square brackets are those required for each right hand side.

2. A TW matrix factorization algorithm for pentadiagonal matrices

2.1. The method

In this section we describe the TW matrix factorization for a pentadiagonal system on a pair of processors. The letters TW are an acronym for Two-Way and refer to the manner in which factorisation and back substitution take place in two directions at once (top-down and bottom-up). We consider the pentadiagonal linear system

$$Ax = d \tag{1}$$

where A is a $n \times n$ matrix and $x,\ d$ are vectors $(n \geq 5)$ of the type

$$A = \begin{pmatrix} b_1 & c_1 & h_1 & & & & & & \\ a_2 & b_2 & c_2 & h_2 & & & & & \\ f_3 & a_3 & . & . & . & & 0 & & \\ & . & . & . & . & . & & & \\ & & . & . & . & . & . & & \\ & & & . & . & . & . & . & \\ & 0 & & & . & . & . & . & h_{n-2} \\ & & & & & . & . & . & c_{n-1} \\ & & & & & & f_n & a_n & b_n \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad d = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}.$$

and we assume A is strongly diagonal dominant.

Let $q = \frac{n}{2}$. Using the LU-decomposition, the matrix A will be factorised into the matrix product TW, where

$$T = \begin{pmatrix} u_1 & 0 & 0 & & & & & & & & \\ s_2 & u_2 & 0 & 0 & & & & & 0 & & \\ f_3 & . & . & . & . & & & & & & \\ & . & . & . & . & . & & & & & \\ & & f_q & s_q & u_q & 0 & 0 & & & & \\ & & & 0 & 0 & u_{q+1} & s_{q+1} & h_{q+1} & & & \\ & & & & . & . & . & . & . & & \\ & 0 & & & & . & . & . & . & h_{n-2} \\ & & & & & & . & . & . & s_{n-1} \\ & & & & & & & 0 & 0 & u_n \end{pmatrix}$$

and

$$W = \begin{pmatrix} 1 & v_1 & w_1 & & & & & & & \\ 0 & 1 & v_2 & w_2 & & & & & & \\ 0 & . & . & . & w_{q-2} & & 0 & & & \\ & . & . & . & v_{q-1} & w_{q-1} & & & & \\ & & 0 & 0 & 1 & v_q & w_q & & & \\ & & & w_{q+1} & v_{q+1} & 1 & 0 & 0 & & \\ & & & & w_{q+2} & v_{q+2} & 1 & & . & \\ & 0 & & & & w_{q+3} & . & . & . & 0 \\ & & & & & & . & . & . & 0 \\ & & & & & & & w_n & v_n & 1 \end{pmatrix}.$$

Hence, we compute the elements of the matrices T and W by

$$u_1 = b_1, \quad v_1 = \frac{c_1}{u_1}, \quad w_1 = \frac{h_1}{u_1},$$
$$s_2 = a_2, \quad u_2 = b_2 - s_2 v_1, \quad v_2 = \frac{c_2 - s_2 w_1}{u_2}, \quad w_2 = \frac{h_2}{u_2},$$

$$\left.\begin{cases} s_i = a_i - f_i v_{i-2}, \\ u_i = b_i - s_i v_{i-1} - f_i w_{i-2}, \\ v_i = \frac{c_i - s_i w_{i-1}}{u_i}, \\ w_i = \frac{h_i}{u_i}, \end{cases}\right\} i = 3, 4, \ldots, q = \tfrac{n}{2}$$

and

$$\begin{aligned}
& u_n = b_n, \quad v_n = \frac{a_n}{u_n}, \quad w_n = \frac{f_n}{u_n}, \\
& s_{n-1} = a_{n-1}, \quad u_{n-1} = b_{n-1} - s_{n-1} v_n, \\
& v_{n-1} = \frac{c_{n-1} - s_{n-1} w_n}{u_{n-1}}, \quad w_{n-1} = \frac{f_{n-1}}{u_{n-1}}, \\
& \left.\begin{cases} s_i = a_i - h_i v_{i+2}, \\ u_i = b_i - s_i v_{i+1} - h_i w_{i+2}, \\ v_i = \frac{c_i - s_i w_{i+1}}{u_i}, \\ w_i = \frac{f_i}{u_i}. \end{cases}\right\} i = n-2, n-3, \ldots, q+1
\end{aligned}$$

Now, we must solve two linear systems

$$Wx = z \tag{2}$$

and

$$Tz = d.$$

Then in the upper and lower partitions

$$\begin{aligned}
z_1 &= \frac{d_1}{u_1}, \quad z_2 = \frac{d_2 - s_2 z_1}{u_2}, \\
z_i &= \frac{d_i - s_i z_{i-1} - f_i z_{i-2}}{u_i}, \quad i = 3, 4, \ldots, q
\end{aligned}$$

and

$$\begin{aligned}
z_n &= \frac{d_n}{u_n}, \quad z_{n-1} = \frac{d_{n-1} - s_{n-1} z_n}{u_{n-1}}, \\
z_i &= \frac{d_i - s_i z_{i+1} - h_i z_{i+2}}{u_i}, \quad i = n-2, n-3, \ldots, q+1.
\end{aligned}$$

From the linear system (2) the four central equations are

$$\begin{aligned}
x_{q-1} + v_{q-1} x_q + w_{q-1} x_{q+1} &= z_{q-1} \\
x_q + v_q x_{q+1} + w_q x_{q+2} &= z_q \\
w_{q+1} x_{q-1} + v_{q+1} x_q + x_{q+1} &= z_{q+1} \\
w_{q+2} x_q + v_{q+2} x_{q+1} + x_{q+2} &= z_{q+2}
\end{aligned}$$

The partitions must exchange information for solving these central equations. The numbers z_{q-1}, z_q and the numbers for dependent matrices v_{q-1}, w_{q-1}, v_q, w_q must be passed to the lower partition. The numbers z_{q+1}, z_{q+2} and the numbers for dependent matrices w_{q+1}, v_{q+1}, w_{q+2}, v_{q+2} must be passed to the upper partition.

This system is equivalent to the following system

$$\begin{aligned}
x_{q-1} + v_{q-1}x_q + w_{q-1}x_{q+1} &= z_{q-1} \\
w_{q+2}x_q + v_{q+2}x_{q+1} + x_{q+2} &= z_{q+2} \\
(1 - w_q w_{q+2})x_q + (v_q - w_q v_{q+2})x_{q+1} &= z_q - w_q z_{q+2} \\
(v_{q+1} - v_{q-1}w_{q+1})x_q + (1 - w_{q-1}w_{q+1})x_{q+1} &= z_{q+1} - w_{q+1}z_{q-1}
\end{aligned}$$

We use Kramer's rule for solving the last two equations of this system and we obtain

$$x_q = \frac{\Delta_1}{\Delta}, \quad x_{q+1} = \frac{\Delta_2}{\Delta}, \tag{3}$$

where

$$\begin{aligned}
\Delta &= \gamma_1\gamma_2 - \delta_1\delta_2, \\
\Delta_1 &= \hat{z}_q\gamma_2 - \hat{z}_{q+1}\delta_1, \\
\Delta_2 &= \hat{z}_{q+1}\gamma_1 - \hat{z}_q\delta_2
\end{aligned}$$

and

$$\begin{aligned}
&\gamma_1 = 1 - w_q w_{q+2}, \quad \gamma_2 = 1 - w_{q-1}w_{q+1}, \\
&\delta_1 = v_q - w_q v_{q+2}, \quad \delta_2 = v_{q+1} - v_{q-1}w_{q+1}, \\
&\hat{z}_q = z_q - w_q z_{q+2}, \quad \hat{z}_{q+1} = z_{q+1} - w_{q+1}z_{q-1}.
\end{aligned}$$

Then x_{q-1}, x_{q+2} can be calculated from

$$\begin{aligned}
x_{q-1} &= z_{q-1} - v_{q-1}x_q - w_{q-1}x_{q+1} \\
x_{q+2} &= z_{q+2} - w_{q+2}x_q - v_{q+2}x_{q+1}.
\end{aligned}$$

Finally, we can compute unknowns $\{x_i\}$

$$\begin{aligned}
x_i &= z_i - v_i x_{i+1} - w_i x_{i+2}, \quad i = q-2, q-3, \ldots, 1, \\
x_i &= z_i - v_i x_{i-1} - w_i x_{i-2}, \quad i = q+3, q+4, \ldots, n.
\end{aligned}$$

The arithmetic costs of this algorithms are

- for the decomposition $[10q - 12]$
- for the inwards sweep $5q + 6$
- for the central equation $12 + [11]$
- for the outwards sweep $4q - 8$

We consider a case of two processors. The algorithm requires 1 communication start-up and the transmission of $2 + [4]$ numbers (for example : z_{q-1}, z_q and v_{q-1}, w_{q-1}, v_q, w_q to the lower partition) . Hence the total cost for two processors is

$$(9q + 10 + [10q - 1])t_a + 1s_c + (2 + [4])t_c.$$

The corresponding cost of the sequential LU algorithm is $(9n-12+[10n-17])\ t_a$ [9] .

2.2. A case of $\frac{n}{2}$ processors

Now consider a case of $\frac{n}{2}$ processors (again A is of size $n \times n$) and let each be assigned to two of the variables. Using the same TW algorithm there are $\frac{q}{2}-1$ communications start-ups for the inward sweep, $\frac{q}{2}-1$ for the outward sweep and one for the central exchange. The total number of communications start-ups is then $q-1$. For each communication on the inward sweep and central exchange $2+[4]$ numbers are transmitted. The numbers are $z_{i-1},\ z_{i-2}$ and $v_{i-2},\ v_{i-1},\ w_{i-2},\ w_{i-1}$ (except for the first step of the sweep). On the outward sweep 2 numbers are transmitted, $x_{i-1},\ x_i$.

Hence the total cost for $\frac{n}{2}$ processors is

$$(9q+10+[10q-1])t_a+(q-1)\ s_c+(2q-2+[2q])\ t_c. \tag{4}$$

2.3. Stability of the algorithm

We suppose A is a diagonally dominant matrix. Then A is nonsingular and an LU factorization of A exists and it is stable. Our TW algorithm makes two LU decompositions and thus they are well defined. A possible problem arises in the division by Δ in equation (3). The following lemma proves the correctness of the division given certain conditions on the elements of the matrix A.

Lemma 1. We assume the elements of the matrix A satisfy

$$\begin{aligned}
&b_1 \geq |c_1|+|h_1|, \quad b_2 \geq |c_2|+|h_2|+2|a_2| \\
&b_i \geq |c_i|+|h_i|+2|a_i|+3|f_i|, \quad i=3,4,\ldots,q \\
&b_n \geq |a_n|+|f_n|, \quad b_{n-1} \geq |c_{n-1}|+|f_{n-1}|+2|a_{n-1}| \\
&b_i \geq |c_i|+|f_i|+2|a_i|+3|h_i|, \quad i=n-2,n-3,\ldots,q+1
\end{aligned}$$

then

$$u_i \geq 0, \quad |v_i| \leq 1, \quad |w_i| \leq 1, \quad |v_i|+|w_i| \leq 1, \quad i=1,2,\ldots,n.$$

Proof. Obviously $u_1=b_1 \geq 0$. We have

$$\begin{aligned}
|c_1| \leq b_1, \quad |v_1| = \frac{|c_1|}{b_1} \leq 1, \\
|h_1| \leq b_1, \quad |w_1| = \frac{|h_1|}{b_1} \leq 1.
\end{aligned}$$

Suppose, for induction

$$u_k \geq 0, \quad |v_k| \leq 1, \quad |w_k| \leq 1,$$

for $k=1,2,\ldots,i-1<q$.

Then

$$\begin{aligned}
u_i &= b_i - s_i v_{i-1} - f_i w_{i-2} \\
&= b_i - (a_i - f_i v_{i-2}) v_{i-1} - f_i w_{i-2} \\
&= b_i - a_i v_{i-1} + f_i v_{i-2} v_{i-1} - f_i w_{i-2} \\
&= b_i - |\alpha|.
\end{aligned}$$

Therefore

$$\begin{aligned}
|\alpha| &= |a_i v_{i-1} - f_i v_{i-2} v_{i-1} + f_i w_{i-2}| \\
&\le |a_i| + 2|f_i| \\
&\le b_i.
\end{aligned}$$

For v_i

$$\begin{aligned}
|v_i| &= \frac{|c_i - s_i w_{i-1}|}{u_i}, \\
&= \frac{|c_i - (a_i - f_i v_{i-2}) w_{i-1}|}{u_i}, \\
&= \frac{|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}|}{u_i}.
\end{aligned}$$

In order to satisfy $|v_i| \le 1$, it is necessary that

$$|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}| \le u_i$$

and hence

$$|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}| \le b_i - |\alpha|.$$

Then

$$|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}| + |\alpha| \le |c_i| + 2|a_i| + 3|f_i| \le b_i.$$

But last inequality is true which follows from condition of the lemma.
For w_i

$$|w_i| = \frac{|h_i|}{u_i} = \frac{|h_i|}{b_i - |\alpha|}.$$

Then

$$|h_i| + |\alpha| \le |c_i| + |a_i| + 2|f_i| \le b_i.$$

Furthermore

$$|v_i| + |w_i| = \frac{|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}|}{u_i} + \frac{|h_i|}{u_i}$$

and

$$
\begin{aligned}
|c_i - a_i w_{i-1} + f_i v_{i-2} w_{i-1}| + |h_i| + |\alpha| &\leq |c_i| + 2|a_i| + 3|f_i| + |h_i| \\
&\leq b_i, \quad i = 3, 4, \ldots
\end{aligned}
$$

Consequently

$$
u_i \geq 0, \quad |v_i| \leq 1, \quad |w_i| \leq 1, \quad |v_i| + |w_i| \leq 1, \quad i = 1, 2, \ldots, q.
$$

The proof of the lemma is analogous in the case $i = n, n-1, \ldots, q+1$.

3. The partition algorithm for pentadiagonal matrices

We now consider the partition algorithm for pentadiagonal matrices. The solution of the linear system (1) on an array of P processors is discussed. We obtain P pentadiagonal subsystems $\{\alpha^i,\ i = 1, \ldots, P\}$ of order m ($m \geq 5$) and $2(P-1)$ boundary equations $\{\beta^i,\ i = 1, \ldots, P-1\}$. Here the integer m is defined so that $n = mP + 2(P-1)$. For simplicity it is assumed that n and P are such that m exists.

We can rewrite the vector $x^T = \{x_i\}$ in the following way

$$
\begin{aligned}
x^T &= (x_1^1, \ldots, x_m^1, x^{2,1}, x^{2,2}, x_1^2, \ldots, x_m^2, \ldots, x^{P,1}, x^{P,2}, x_1^P, \ldots, x_m^P) \\
x^T &= ((\mathbf{x}^1)^T, (\hat{\mathbf{x}}^2)^T, (\mathbf{x}^2)^T, \ldots, (\hat{\mathbf{x}}^P)^T, (\mathbf{x}^P)^T).
\end{aligned}
$$

The vectors $b^T = \{b_i\},\ a^T = \{a_i\},\ f^T = \{f_i\},\ c^T = \{c_i\},\ h^T = \{h_i\}$, $d^T = \{d_i\},\ i = 1, \ldots, n$ are partitioned in the same way. Thus the system (1) can be written of the form

$$
(\alpha^1) \qquad P^1 \mathbf{x}^1 + D^1 \hat{\mathbf{x}}^2 = \mathbf{d}^1
$$

$$
(\alpha^i) \qquad Q^i \hat{\mathbf{x}}^i + P^i \mathbf{x}^i + D^i \hat{\mathbf{x}}^{i+1} = \mathbf{d}^i, \quad i = 2, \ldots, P-1
$$

$$
(\alpha^P) \qquad Q^P \hat{\mathbf{x}}^P + P^P \mathbf{x}^P = \mathbf{d}^P
$$

where

$$
P^i = \begin{pmatrix}
b_1^i & c_1^i & h_1^i & & & & & \\
a_2^i & . & . & . & & & & \\
f_3^i & . & . & . & . & 0 & & \\
 & . & . & . & . & . & & \\
 & & . & . & . & . & . & \\
 & & & . & . & . & . & . \\
 & 0 & & & . & . & . & h_{m-2}^i \\
 & & & & & . & . & c_{m-1}^i \\
 & & & & & f_m^i & a_m^i & b_m^i
\end{pmatrix},
$$

$$Q^i = \begin{pmatrix} f_1^i & a_1^i \\ 0 & f_2^i \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix}, \quad D^i = \begin{pmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ h_{m-1}^i & 0 \\ c_m^i & h_m^i \end{pmatrix}.$$

The method has three phases

- phase 1 - this is a local reduction to eliminate off-diagonals in the subsystems
- phase 2 - this is the global solution of the reduced block matrix for the boundary variables
- phase 3 - the local back-substitution for the subsystem variables

For the first phase each processor i is assigned the pentadiagonal system (α^i). The unknown vector $\hat{\mathbf{x}}^i = \begin{pmatrix} x^{i,1} \\ x^{i,2} \end{pmatrix}$ denotes the boundary variables of the boundary equations

$$(\beta^i) \qquad Q_B^i \begin{pmatrix} x_{m-1}^{i-1} \\ x_m^{i-1} \end{pmatrix} + P_B^i \hat{\mathbf{x}}^i + D_B^i \begin{pmatrix} x_1^i \\ x_2^i \end{pmatrix} = \hat{\mathbf{d}}^i,$$

where

$$Q_B^i = \begin{pmatrix} f^{i,1} & a^{i,1} \\ 0 & f^{i,2} \end{pmatrix}, \quad D_B^i = \begin{pmatrix} h^{i,1} & 0 \\ c^{i,2} & h^{i,2} \end{pmatrix}, \quad P_B^i = \begin{pmatrix} b^{i,1} & c^{i,1} \\ a^{i,2} & b^{i,2} \end{pmatrix}.$$

We use the equation (α^{i-1}) for computing x_{m-1}^{i-1}, x_m^{i-1} and the equation (α^i) for computing x_1^i, x_2^i in the first phase. Each processor solves the following linear systems

$$\begin{aligned} P^i \mathbf{w}^i &= \mathbf{d}^i, \\ P^i R^i &= Q^i, \\ P^i S^i &= D^i. \end{aligned}$$

In fact there are 5 linear systems with the same coefficient matrix P^i. We have

$$\begin{aligned} \mathbf{w}^i &= (P^i)^{-1} \mathbf{d}^i, \\ R^i &= (P^i)^{-1} Q^i, \\ S^i &= (P^i)^{-1} D^i. \end{aligned}$$

and from equation (α^i) obtain

$$((\alpha^i)') \qquad \mathbf{x}^i = \mathbf{w}^i - R^i \hat{\mathbf{x}}^i - S^i \hat{\mathbf{x}}^{i+1}.$$

The unknowns $x^{i,1}$, $x^{i,2}$, $x^{i+1,1}$, $x^{i+1,2}$ can be computed from the boundary equations

$$-Q_B^i E R^{i-1} \hat{\mathbf{x}}^{i-1} + (P_B^i - Q_B^i E S^{i-1} - D_B^i F R^i) \hat{\mathbf{x}}^i - D_B^i F S^i \hat{\mathbf{x}}^{i+1} = \hat{\mathbf{d}}^i - Q_B^i E \mathbf{w}^{i-1} - D_B^i F \mathbf{w}^i,$$

where $i = 2, \dots, P$, and $E,\ F$ are $2 \times n$ matrices of the type

$$E = \begin{pmatrix} 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & 0 & 0 & 1 \end{pmatrix}, \quad F = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \end{pmatrix}.$$

We obtain a new tridiagonal block system with 2×2 block matrices

$$My = q, \tag{5}$$

where

$$M = \begin{pmatrix} \tilde{B}^2 & \tilde{C}^2 & & & & \\ \tilde{A}^3 & \cdot & \cdot & & & \\ & \cdot & \cdot & \cdot & 0 & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & \cdot & \cdot \\ & 0 & & & \cdot & \cdot & \tilde{C}^{P-1} \\ & & & & & \tilde{A}^P & \tilde{B}^P \end{pmatrix}$$

and

$$y = \begin{pmatrix} \hat{\mathbf{x}}^2 \\ \hat{\mathbf{x}}^3 \\ \vdots \\ \hat{\mathbf{x}}^P \end{pmatrix}, \quad q = \begin{pmatrix} \tilde{d}^2 \\ \tilde{d}^3 \\ \vdots \\ \tilde{d}^P \end{pmatrix}.$$

The coefficients of the system are given by

$$\begin{aligned} \tilde{A}^i &= -Q_B^i E R^{i-1}, && i = 3, \dots, P, \\ \tilde{B}^i &= P_B^i - Q_B^i E S^{i-1} - D_B^i F R^i, && i = 2, \dots, P, \\ \tilde{C}^i &= -D_B^i F S^i, && i = 2, \dots, P-1, \\ \tilde{d}^i &= \hat{\mathbf{d}}^i - Q_B^i E \mathbf{w}^{i-1} - D_B^i F \mathbf{w}^i, && i = 2, \dots, P. \end{aligned}$$

The second phase is the computation of the coefficients and right hand side and solution of this reduced tridiagonal block system for the boundary variables. The cost is

- [8] arithmetic operations for computing $\tilde{A}^i$ and the transmission of 4 numbers of ER^{i-2} (the last two rows)
- [8] arithmetic operation for computing $\tilde{C}^i$ and not transmission
- [24] arithmetic operation for computing $\tilde{B}^i$ and the transmission of 4 numbers of ES^{i-1} (the first two rows)
- 12 arithmetic operation for computing $\tilde{d}^i$ and the transmission of 2 numbers $w_{m-1}^{i-1},\ w_m^{i-1}$

The calculation of the reduced block matrix and right hand side requires $12+[40]$ arithmetic operations per processor and the transmission of $2+[8]$ numbers. Hence the first processor requires $12+[32]$ operations for computing $\tilde{B}^2, \tilde{C}^2$ and $\tilde{d}^2$ and the transmission $2+[4]$ numbers. In the same way the processor P requires $12+[32]$ operations and the transmission $2+[8]$ numbers. However, we assume that these two processors need the same quantity of arithmetic operations as the other processors.

Thus the total cost for computing the coefficients of (5) is

$$(12+[40])\ t_a + (2+[8])\ t_c. \tag{6}$$

Next we have to solve this system (5). This can be done in various ways [3]. Here we propose a new approach for the solution of this special block tridiagonal system (5). First we reduce the obtained matrix M to a pentadiagonal matrix. We compute

$$\Lambda M \Sigma\ \Sigma^{-1}\ y = \Lambda q,$$

where $\Lambda,\ \Sigma$ are diagonal matrices of the type

$$\begin{aligned} \Lambda &= diag[I_2, \Lambda_3, \ldots, \Lambda_P], \\ \Sigma &= diag[I_2, \Sigma_3, \ldots, \Sigma_P], \end{aligned}$$

I_2 is the 2×2 identity matrix and

$$\Lambda_j = \begin{pmatrix} 1 & 0 \\ \lambda_j & 1 \end{pmatrix}, \quad \Sigma_j = \begin{pmatrix} 1 & \sigma_j \\ 0 & 1 \end{pmatrix}, \quad j = 3, \ldots, P.$$

We receive a new block tridiagonal system

$$\hat{M}\ \hat{y} = f, \tag{7}$$

where

$$\hat{M} = \begin{pmatrix} \hat{B}^2 & \hat{C}^2 & & & & & \\ \hat{A}^3 & \cdot & \cdot & & & & \\ & \cdot & \cdot & \cdot & & 0 & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & \cdot \\ & 0 & & & & \cdot & \cdot & \hat{C}^{P-1} \\ & & & & & & \hat{A}^P & \hat{B}^P \end{pmatrix}$$

and

$$\hat{y} = \Sigma^{-1} y, \quad f^T = (\Lambda q)^T = (f_2^T, f_3^T, \ldots, f_P^T).$$

For the block-elements of this system we obtain

$$\begin{array}{lll} \hat{A}^3 = \Lambda_3\ \tilde{A}^3, & \hat{A}^j = \Lambda_j\ \tilde{A}^j\ \Sigma_{j-1}, & j = 4, \ldots, P, \\ \hat{B}^2 = \tilde{B}^2, & \hat{B}^j = \Lambda_j\ \tilde{B}^j\ \Sigma_j, & j = 3, \ldots, P, \\ \hat{C}^2 = \tilde{C}^2\ \Sigma_3, & \hat{C}^j = \Lambda_j\ \tilde{C}^j\ \Sigma_{j+1}, & j = 3, \ldots, P-1, \\ f_2 = \tilde{d}_2, & f_j = \Lambda_j\ \tilde{d}^j, & j = 3, \ldots, P. \end{array}$$

If $\hat{A}^j = (\hat{a}^j_{ik}),\ \hat{B}^j = (\hat{b}^j_{ik}),\ \hat{C}^j = (\hat{c}^j_{ik})$ we choose $\lambda_j,\ \sigma_j,\ j = 3, \ldots, P$ so that

$$\hat{a}^j_{21} = 0, \quad \hat{c}^j_{12} = 0. \tag{8}$$

From (8) we find

$$\begin{aligned} \lambda_j &= -\frac{\tilde{a}^j_{21}}{\tilde{a}^j_{11}},\ j = 3, \ldots, P, \\ \sigma_{j+1} &= -\frac{\tilde{c}^j_{12}}{\tilde{c}^j_{11}},\ j = 3, \ldots, P-1. \end{aligned}$$

If $\tilde{a}^s_{11} = 0$ or $\tilde{c}^s_{11} = 0$ for $2 \leq s \leq P-1$ then we can choose

$$\Lambda_s = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \text{or} \quad \Sigma_{s+1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

This choice leads to the exchange the rows or columns of the matrix M.

The calculation of the new coefficients requires :

- [4] arithmetic operations for computing $\hat{A}^j$ and the transmission of [1] number (σ_{j-1})
- [8] arithmetic operations for computing $\hat{B}^j$
- [6] arithmetic operations for computing $\hat{C}^j$ and the transmission of [1] number (σ_{j+1})
- 2 arithmetic operations for computing f_j

per processor. The total cost for computing block elements of $\hat{M}$ and right hand side is

$$(2 + [18])\ t_a + 2\ t_c. \tag{9}$$

Then the total cost from (6) and (9) is

$$(14 + [58])\ t_a + (4 + [8])\ t_c \tag{10}$$

for computing the coefficients of the system (7).

Now, we solve our pentadiagonal linear system (7) using the TW matrix factorization described in section 2. Since we have to solve the system with $2(P-1)$ equations, we shall use $P-1$ processors, i.e. each processor assigned two of the variables. Here $q = P-1$ and we have total cost

$$(9q + 10 + [10q - 1])t_a + (q-1)\ s_c + (2q - 2 + [2q])\ t_c. \tag{11}$$

When we solve the linear system (7) we have to compute the solution $y = \Sigma\hat{y}$ of the system (5). For this 2 arithmetic operations per processor are required and the cost is $2\ t_a$.

The total cost on second phase from (10) , (11) and computing y is

$$(9q + 26 + [10q - 57])\ t_a + (q - 1)\ s_c + (2q + 2 + [2q + 8])\ t_c.$$

In the third phase each processor has the value of the boundary variables. Then these values are substituted into $(\alpha)'$ to obtain the solution.

The arithmetic costs of phase 1 and 3 of this algorithm are

- phase 1 - $[L^i, U^i]$ calculation $[10m - 17]$
- phase 1 - $\mathbf{w}^i$ calculation $9m - 12$
- phase 1 - $[R^i]$ calculation $[16m - 21]$
- phase 1 - $[S^i]$ calculation $[6m - 6]$
- phase 3 - $\mathbf{x}^i$ calculation $8m$

There are a few notes:

1. The solution of the systems $P^i R^i = Q^i$ and $P^i S^i = D^i$ uses LU decomposition of the matrix P^i. It depends on the implementation of the LU decomposition and uses either forward or backward substitution. Since Q^i and D^i are right hand sides of these systems which have a special type, the computation of R^i and S^i includes fewer arithmetic operations than for a full right hand part such as $\mathbf{w}^i$.

2. Processor 1 need not compute R^1 and processor P need not compute S^P. Hence the cost of phase 3 for these two processors is $5m$.

Thus the total cost of vector operations in phase 1 and 3 is

$$(17m - 12 + [32m - 44])t_a.$$

Thus the total cost of vector operations in phase 1, 2 and 3 is

$$(17m + 9q + 14 + [32m + 10q - 101])\ t_a + (q - 1)\ s_c + (2q + 2 + [2q + 8])\ t_c. \qquad (12)$$

Then in terms of n and P the operations can be summarised as $P - 2$ communications with $17\frac{n}{P} + 9P$ arithmetic operations for constant matrices and $49\frac{n}{P} + 19P$ arithmetic operations for time-dependent matrices.

4. The parallel TW partition algorithm for pentadiagonal matrices

In this section we combine the methods of section 2 and 3 and we obtain a new more efficient algorithm for solving a pentadiagonal system on P processors.

Firstly we consider the partition algorithm for $\frac{P}{2}$ processors. Each processor considers the subsystem (α^i), $i = 1, 2, \ldots, \frac{P}{2}$ and makes the LU decomposition of phase 1. Thus it is possible to assign each subsystem (α^i) to two processors and use the TW factorization from section 2 on phase 1 instead of LU decomposition for solving each subsystem (α^i).

So the first phase of this algorithm is to execute TW factorization of the subsystem (α^i). Then in the second phase we have to solve the reduced block tridiagonal system of order $2(\frac{P}{2}-1)$ instead of block tridiagonal system of order $2(P-1)$.

Now we describe the TW parallel partition algorithm. We consider the solution of the pentadiagonal system (1) on an array of P processors. Assume that P is even and that $q=\frac{P}{2}$. We define q pentadiagonal subsystems of order $2m$, $(2m \geq 5)$ $\{\alpha^i,\ i=2,4,6,\ldots,P\}$ and $q-1$ boundary equations $\{\beta^i,\ i=4,6\ldots,P\}$. Here the integer m is defined so that $n=2mq+2(q-1)$. For the system (1) we obtain

$$(\alpha^2) \qquad P^2\mathbf{x}^2+D^2\hat{\mathbf{x}}^4=\mathbf{d}^2$$

$$(\alpha^i) \qquad Q^i\hat{\mathbf{x}}^i+P^i\mathbf{x}^i+D^i\hat{\mathbf{x}}^{i+2}=\mathbf{d}^i, \quad i=4,6,\ldots,P-2$$

$$(\alpha^P) \qquad Q^P\hat{\mathbf{x}}^P+P^P\mathbf{x}^P=\mathbf{d}^P.$$

Then each subsystem (α^i) is assigned to two processors $i-1$ and i, $i=2,4,\ldots,P$, so that each of these pairs of processors store the system (α^i).

This system is distributed by assigning the first m equations to processor $i-1$ and the second m equations to processor i. Again we receive the boundary equations

$$(\beta^i) \qquad Q_B^i\begin{pmatrix} x_{2m-1}^{i-2} \\ x_{2m}^{i-2}\end{pmatrix}+P_B^i\hat{\mathbf{x}}^i+D_B^i\begin{pmatrix} x_1^i \\ x_2^i\end{pmatrix}=\hat{\mathbf{d}}^i, \quad i=4,6,\ldots,P,$$

and $(\hat{x}^i)^T=(x^{i,1},x^{i,2})$ are boundary variables between the subsystems (α^{i-2}) and (α^i).

Phases 1 and 3 of this algorithm are the same as phases 1 and 3 of the partition algorithm (section 3). The calculation of $\mathbf{w}^i, R^i, S^i$ is necessary for each pair of processors using a TW matrix factorisation and one communication exchange.

In phase 2 the reduced block tridiagonal system for boundary variables is solved. This system is a $q-1(=\frac{P}{2}-1)$ block system or $2q-2(=P-2)$ linear system. For solving this system we follow phase 2 of the partition algorithm. First we reduce the block system into the pentadiagonal linear system. Secondly the TW matrix factorization for pentadiagonal matrix is applied. In this case each boundary equation is stored on both sides of the inter-processor interface and the network can be divided into two independent parts : the even numbered processors and the odd numbered processors. Each group of processors has a copy of the reduced matrix. Both odd and even systems can be solved with a TW factorization. Then the value of boundary variables $\hat{x}^2$ and $\hat{x}^P$ must be transmitted to processors 1 and P respectively to start phase 3.

The arithmetic costs of phase 1 and 3 of this algorithm are

- phase 1 - the arithmetic operations for $[T^i, W^i]$ are $[10m-1]$ for $i-1$ and i processors
- phase 1 - the arithmetic operations for $\mathbf{w}^i$ are $9m+6$ for $i-1$ and i processors
- phase 1 - the arithmetic operations for $[R^i]$ are $[16m-25]$ and $[6m-12]$ for $i-1$ and i processors respectively

- phase 1 - the arithmetic operations for $[S^i]$ are $[6m-12]$ and $[16m-25]$ for $i-1$ and i processors respectively
- phase 3 - the arithmetic operations for $\mathbf{x}^i$ are $8m$

In phase 1 there is one communication exchange of $2+[6]$ numbers, $2+[4]$ numbers the computation of $\mathbf{w}^i, R^i, S^i$ and extra [2] numbers for the communication of the fill-ins for the pentadiagonal matrix. The total cost of phases 1 and 3 is

$$(17m+6+[32m-38])\ t_a+1\ s_c+(2+[4])\ t_c. \tag{13}$$

In phase 2, the calculation of the reduced matrix $\tilde{M}$ and right hand side f requires the same quantity of arithmetic operations and transmission of numbers as in phase 2 of partition algorithm, i.e.

$$(14+[58])\ t_a+(4+[8])\ t_c. \tag{14}$$

Now, we compute the cost for solving pentadiagonal linear system with $P-2$ equations. We use $\frac{P}{2}-1$ processors . Then from (11) and $k=\frac{P}{2}-1$ we have

$$(9k+10+[10k-1])t_a+(k-1)\ s_c+(2k-2+[2k])\ t_c. \tag{15}$$

At the end of phase 2, $\mathbf{x}^2$ and $\mathbf{x}^{P-2}$ must be transmitted to processor 1 and P respectively, a cost of

$$1\ s_c+2\ t_c. \tag{16}$$

Then from (13) - (16), we have

$$(17m+9k+30+[32m+10k+19])t_a+(k+1)\ s_c+(2k+6+[2k+12])\ t_c. \tag{17}$$

Finally the operation counts of this method can be defined as $\frac{P}{2}$ communications with $49\frac{n}{P}+\frac{19}{2}P$ arithmetic operations for time-dependent matrices or $17\frac{n}{P}+\frac{9}{2}P$ operations for constant matrices.

There are a few advantages for the parallel TW partition algorithm

1. The operations cost in (12) is reduced to that in (17).

2. In the case of a chain of processors the number of communications start-ups can be reduced to $\frac{P}{2}$.

3. The operation cost is reduced to $49\frac{n}{P}+\frac{19}{2}P$ arithmetic operations.

5. Discussion and numerical results

Arbenz and Gander [3] have discussed the three methods - dca, swsa, dwsa. They have considered these methods under the assumption the reduced system is solved by block Gaussian elimination [9] or by block cyclic reduction [9]. They have given parallel complexities of both cases. We consider the parallel complexity only the case of block Gaussian elimination used in second phase of these methods. Arbenz denotes the parallel complexity C^{par}_{dca}, C^{par}_{swsa}, C^{par}_{dwsa} for these methods and the complexity of Gaussian elimination on serial computers as C_{Gauss} and computes

$$
\begin{aligned}
C^{par}_{dca}(k) &\approx (8k^2+7k-1)\frac{n}{P}+(p-1)\frac{k}{6}(28k^2+45k-1)\\
C^{par}_{swsa}(k) &\approx (8k^2+10k-1)\frac{n}{P}+(p-1)\frac{k}{6}(28k^2+27k-7)\\
C^{par}_{dwsa}(k) &\approx (16k^2+10k-1)\frac{n}{P}+(p-1)\frac{2k}{3}(13k^2+12k+5)\\
C_{Gauss}(k) &\approx (2k^2+5k+1)\ n
\end{aligned}
$$

where k is lower half-bandwidth and upper half-bandwidth.

The following tables give the theoretical complexity for the various algorithms in the case of a tridiagonal system and in the case of a pentadiagonal system.

algorithm	complexity $k=1$	speedup	communications
Gauss	$8n$	-	-
$PATS$	$17\frac{n}{P}+4P$	$\frac{C_{Gauss}}{C_{PATS}}=\frac{P}{\frac{17}{8}+\frac{1}{2}\frac{P^2}{n}}$	$P-1$
$TWPATS$	$17\frac{n}{P}+2P$	$\frac{C_{Gauss}}{C_{TWPATS}}=\frac{P}{\frac{17}{8}+\frac{1}{4}\frac{P^2}{n}}$	$\frac{P}{2}+1$
dca	$14\frac{n}{P}+12(P-1)$	$\frac{C_{Gauss}}{C^{par}_{dca}(1)}=\frac{P}{\frac{14}{8}+\frac{12}{8}\frac{P^2-P}{n}}$	$2P$
swsa	$17\frac{n}{P}+8(P-1)$	$\frac{C_{Gauss}}{C^{par}_{swsa}(1)}=\frac{P}{\frac{17}{8}+\frac{P^2-P}{n}}$	$2P$
dwsa	$25\frac{n}{P}+20(P-1)$	$\frac{C_{Gauss}}{C^{par}_{dwsa}(1)}=\frac{P}{\frac{25}{8}+\frac{20}{8}\frac{P^2-P}{n}}$	$2P$

Table 1. Complexity for various algorithms of a tridiagonal system

algorithm	complexity $k=2$	speedup	communications
Gauss	$19n$	-	-
$PAPS$	$49\frac{n}{P}+19P$	$\frac{C_{Gauss}}{C_{PAPS}}=\frac{P}{\frac{49}{19}+\frac{P^2}{n}}$	$P-2$
$TWPAPS$	$49\frac{n}{P}+\frac{19}{2}P$	$\frac{C_{Gauss}}{C_{TWPAPS}}=\frac{P}{\frac{49}{19}+\frac{1}{2}\frac{P^2}{n}}$	$\frac{P}{2}$
dca	$45\frac{n}{P}+67(P-1)$	$\frac{C_{Gauss}}{C^{par}_{dca}(2)}=\frac{P}{\frac{45}{19}+\frac{67}{19}\frac{P^2-P}{n}}$	$2P$
swsa	$51\frac{n}{P}+55(P-1)$	$\frac{C_{Gauss}}{C^{par}_{swsa}(2)}=\frac{P}{\frac{51}{19}+\frac{55}{19}\frac{P^2-P}{n}}$	$2P$
dwsa	$83\frac{n}{P}+27(P-1)$	$\frac{C_{Gauss}}{C^{par}_{dwsa}(2)}=\frac{P}{\frac{83}{19}+\frac{27}{19}\frac{P^2-P}{n}}$	$2P$

Table 2. Complexity for various algorithms of a pentadiagonal system

From the tables we see that the TW approach for solving tridiagonal and pentadiagonal linear systems is faster in terms of arithmetic complexity than the double-width separator algorithm (dwsa) and about the same as the single-width separator algorithm (swsa). The divide and conquer algorithm (dca) is slightly faster than the $TWPATS$ and $TWPAPS$ algorithms. However, the main advantage of the TW approach is that uses fewer communications than other algorithms. $TWPAPS$ algorithm solves the reduced block tridiagonal system in the second phase using TW parallel algorithm described in section 2.

Both algorithms, the partition algorithm for pentadiagonal systems $PAPS$ and the algorithm $TWPAPS$ have been implemented. We carried out numerical experiments on a cluster of workstations at Greenwich University. The code was written in *FORTRAN 77* using the Message Passing Interface (MPI) communications library.

The results are divided into different cases for different number of processors. In case of 2 processors the $TWPAPS$ algorithm is the TW decomposition without boundary variables (section 2). For 4 and 8 processors the $TWPAPS$ algorithm is faster than $PAPS$ algorithm. The following tables show results from numerical experiments.

sequential computer				2 processors					
		$PAPS$	$TWPAPS$	$PAPS$			$TWPAPS$		
m	n	time	time	m	n	time	m	n	time
10000	20000	0.36	0.36	10000	20002	0.61	10000	20000	0.27
20000	40000	0.75	0.72	20000	40002	1.22	20000	40000	0.55
30000	60000	1.11	1.10	30000	60002	1.82	30000	60000	0.77
40000	80000	1.66	1.56	40000	80002	2.43	40000	80000	1.07

Table 3. Times in seconds for $PAPS$ and $TWPAPS$ for sequential computer and 2 processors.

4 processors					
$PAPS$			$TWPAPS$		
m	n	time	m	n	time
5000	20006	0.55	5000	20002	0.44
10000	40006	1.02	10000	40002	0.89
15000	60006	1.54	15000	60002	1.24
20000	80006	1.99	20000	80002	1.65

Table 4. Times in seconds for $PAPS$ and $TWPAPS$ for 4 processors.

8 processors					
PAPS			*TWPAPS*		
m	n	time	m	n	time
2500	20014	0.45	2500	20006	0.42
5000	40014	0.74	5000	40006	0.64
7500	60014	1.06	7500	60006	0.87
10000	80014	1.38	10000	80006	1.15

Table 5. Times in seconds for *PAPS* and *TWPAPS* for 8 processors.

6. Conclusion

From the theoretic complexities in Tables 1 and 2 we see that the main advantage in using the TW approach (i.e. solving the block systems on $\frac{p}{2}$ pairs of processors) is that the number of communication startups is halved. On many parallel machines these startups can be a considerable overhead and this is borne out in the parallel timing results (Tables 3, 4 and 5). Of course, on systems where the computational costs dominate or for very large values of n then the complexity tables suggest that the divide and conquer algorithm (dca) will be the most efficient parallel algorithm. However, in other circumstances, where the number of communication startups does play an important role, the results suggest that *TWPAPS* will be fastest algorithm.

References

[1] P. Amodio and M. Paprzycki, A cyclic reduction approach to the numerical solution of boundary value ODEs, *SIAM J. Sci. Comput.* 18 (1997) 56-68.

[2] P. Arbenz and M. Hegland, The stable parallel solution of general narrow banded linear systems, Technical Report, Department of Informatics, ETH Zurich (1994).

[3] P. Arbenz and W. Gander, A survey of direct parallel algorithms for banded linear systems, Technical Report 194-221.ps, Department of Informatics, ETH Zurich (1994).

[4] S. Bondineli, Divide and conquer: a parallel algorithm for a solution of a tridiagonal linear system of equations, *Parallel Computing* 17 (1991) 419-434.

[5] J. Conroy, Parallel algorithms for the solution of narrow banded systems, *Applied Numerical Mathematics* 5 (1989) 409-421.

[6] M. Chen, On the solution of circulant linear systems, *SIAM J. Numer. Analisys* 24 (1987) 668-683.

[7] J. Dongarra and L. Johnsson, Solving banded systems on a parallel processor, *Parallel Computing* 5 (1987) 219-246.

[8] J. Dongarra and A. Sameh, On some parallel banded solvers. *Parallel Computing* 1 (1984) 223-235.

[9] G. Golub and C. Van Loan, *Matrix Computations* (The John Hopkins University Press, Baltimore, 1989).

[10] S. Johnsson, Solving narrow banded systems on ensemble architectures, *ACM Transactions on Mathematical Software* 11 (1985) 271-288.

[11] S. Johnsson, Solving tridiagonal systems on ensemble architectures, *SIAM J. Sci. Stat. Comput.* 8 (1987) 354-392.

[12] D. Lawrie and A. Sameh, The computation and communication complexity of a parallel banded system solver, *ACM Transactions on Mathematical Software* 10 (1984) 185-195.

[13] A. Sameh and D Kuck, On stable parallel linear system solvers, *Journal of the Association for Computing Machinery* 25 (1978) 81-91.

[14] C. Walshaw and S. Farr, A two-way parallel partition method for solving tridiagonal systems, Research Report Series 93.25, School of Computer Studies, University of Leeds (1993).

[15] C. Walshaw, Diagonal dominance in the parallel partition method for tridiagonal systems, *SIAM J. Matrix Anal. Appl.* 16 (1995) 1086-1099.

[16] S. Wright. Parallel algorithms for banded linear systems. *SIAM J. Sci. Stat. Comput.* 12 (1991) 824-842.