AMODIO, P.; PAPRZYCKI, M.

Recent Advances in the Parallel Solution to Almost Block Diagonal Systems

In this paper we briefly summarize some of the most important algorithms for the parallel solution of Almost Block Diagonal linear systems. Then, we consider a parallel algorithm, based on the cyclic reduction, which seems to be quite competitive, especially when systems with additional boundary blocks are considered. Numerical tests carried out on a distributed memory parallel computer are reported and analysed.

1. Historical overview

Since the late 1970's, a number of publications appeared studying solution methods for Almost Block Diagonal (ABD) linear systems. ABD systems arise in various mathematical applications such as Chebyshev spectral decomposition on rectangular domains, orthogonal spline collocation for elliptic problems and, most importantly for the purpose of this paper, various discretizations of boundary value ordinary differential equations (BVP ODE's). The initial, single processor approaches can be traced first, to the SOLVEBLOCK package by de Boor and Weiss [6], and second, to the alternate row and column elimination algorithm due to Varah [12], later studied by Diaz et.al. [7] and finally implemented using level 3 BLAS primitives by Paprzycki and Gladwell [11].

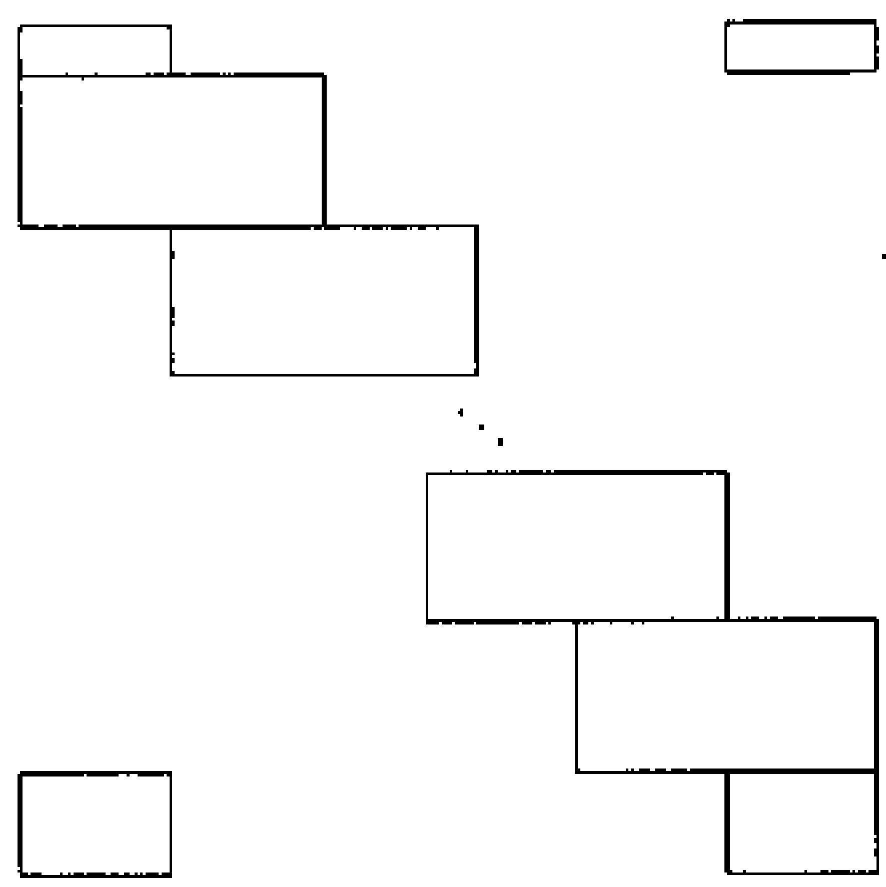

Figure 1: *Structure of an ABD matrix with additional corner blocks.*

As far as parallel solution of ABD systems is concerned, a number of approaches have been developed. It was observed that there are two basic parameters influencing the possible solution methods. When the size of each block is large and the total number of blocks is relatively small or medium (as in the case of spectral discretizations) the BLAS based approach from [11] can be applied (where parallelism is introduced inside the BLAS kernels). It will be relatively successful on shared memory computers [8]. When a number of blocks is large and their individual sizes are small, tearing-type methods can be applied. Ascher and Chan [5] and Jackson and Pancer [9] have formed normal equations and suggested applying methods for block tridiagonal systems. A different tearing-type algorithm has been proposed by Paprzycki and Gladwell [11], where the tearing process is applied to the whole ABD system; it has been later improved by Amodio and Paprzycki [3]. Yet another tearing approach was proposed by K. Wright [13] who applies tearing to the block-bidiagonal system obtained by ignoring boundary conditions (BC's), and reintroduces the BC related blocks only in the final step of the solution process. While the method used in [11, 3] can be applied on the message passing as well as shared memory computers, the method used in [13] can be used only on shared memory computers.

All these algorithms deal with the solution of the ABD system arising from the discretization of BVP ODE's wit *separated* boundary conditions. In case of *non – separated* boundary conditions additional corner blocks appea (see Fig. 1). S. Wright introduced a method similar to the approach of K. Wright that can also deal with thes additional blocks [14,15] and can be used on shared memory as well as message passing parallel computers. Th aim of this paper is to present a different, cyclic-reduction based approach to the solution of ABD systems wit separated as well as non-separated boundary conditions. In Section 2, the proposed algorithm is summarized whil in Section 3, the results of numerical experiments are presented and discussed.

2. The cyclic reduction approach

The cyclic reduction algorithm is one of the most interesting algorithms for the solution of tridiagonal and bloc tridiagonal linear systems on parallel computers (see, for example, [1]). Several implementations have been propose and have been used to optimize a parallel solution on different computer architectures. To derive a generalization c cyclic reduction algorithm for the factorization of ABD matrices let us express the ABD matrix M in the followin form (see Fig. 1):

$$M = \begin{pmatrix} A_{2,0} & D_{2,0} & & & & & & & C_{1,0} & B_{1,0} \\ D_{1,1} & A_{1,1} & B_{2,1} & C_{2,1} & & & & & & \\ C_{1,1} & B_{1,1} & A_{2,1} & D_{2,1} & & & & & & \\ & & D_{1,2} & A_{1,2} & B_{2,2} & C_{2,2} & & & & \\ & & C_{1,2} & B_{1,2} & A_{2,2} & D_{2,2} & & & & \\ & & & & & \ddots & & & & \\ & & & & & & D_{1,m} & A_{1,m} & B_{2,m} & C_{2,m} \\ & & & & & & C_{1,m} & B_{1,m} & A_{2,m} & D_{2,m} \\ B_{2,m+1} & C_{2,m+1} & & & & & & & D_{1,m+1} & A_{1,m+1} \end{pmatrix}, \tag{1}$$

where blocks $A_{i,j}$ are square and any generic block j in Fig. 1 (apart the corner blocks) is decomposed in the following form:

$$\begin{pmatrix} D_{1,j} & A_{1,j} & B_{2,j} & C_{2,j} \\ C_{1,j} & B_{1,j} & A_{2,j} & D_{2,j} \end{pmatrix}.$$

To emphasize a block tridiagonal structure, let us now consider the following decomposition of (1):

$$\begin{pmatrix} A_0 & B_0 & & C_0 \\ C_1 & A_1 & \ddots & \\ & \ddots & \ddots & B_{m-1} \\ B_m & & C_m & A_m \end{pmatrix}, \tag{2}$$

where

$$C_i = \begin{pmatrix} C_{1,i} & B_{1,i} \\ O & O \end{pmatrix} \quad A_i = \begin{pmatrix} A_{2,i} & D_{2,i} \\ D_{1,i+1} & A_{1,i+1} \end{pmatrix}, \quad B_i = \begin{pmatrix} O & O \\ B_{2,i+1} & C_{2,i+1} \end{pmatrix}.$$

We may now apply the odd-even cyclic reduction (similar to that proposed in [2]) to the matrix (2). In order to preserve the sparsity structure, we only require that the first and the last row of (2) must be treated as even rows (the first row is considered as row 0). For example, assuming m is even, the first step of reduction reduces matrix (2) to the following one which has half the number of blocks:

$$\begin{pmatrix} \tilde{A}_0 & \tilde{B}_0 & & & C_0 \\ \tilde{C}_2 & \tilde{A}_2 & \tilde{B}_2 & & \\ & \tilde{C}_4 & \tilde{A}_4 & \ddots & \\ & & \ddots & \ddots & \tilde{B}_{m-2} \\ B_m & & & \tilde{C}_m & \tilde{A}_m \end{pmatrix}, \tag{3}$$

$$\tilde{A}_0 = A_0 - B_0 A_1^{-1} C_1, \qquad \tilde{A}_m = A_m - C_m A_{m-1}^{-1} B_{m-1},$$
$$\tilde{A}_{2i} = A_{2i} - B_{2i} A_{2i+1}^{-1} C_{2i+1} - C_{2i} A_{2i-1}^{-1} B_{2i-1}, \qquad \text{for } i = 1, \ldots, m/2 - 1,$$
$$\tilde{C}_{2i} = -C_{2i} A_{2i-1}^{-1} C_{2i-1}, \qquad \text{for } i = 1, \ldots, m/2,$$
$$\tilde{B}_{2i} = -B_{2i} A_{2i+1}^{-1} B_{2i+1}, \qquad \text{for } i = 0, \ldots, m/2 - 1.$$

Note that during the factorization of blocks A_i row pivoting can be applied. Blocks B_i and C_i have some null rows, and thus $B_{2i}A_{2i+1}^{-1}$ and $C_{2i}A_{2i-1}^{-1}$ have zeroes in the same rows and therefore the blocks $\tilde{C}_{2i}$ and $\tilde{B}_{2i}$ maintain the same sparsity structure as the corresponding C_{2i} and B_{2i}. Moreover, observe that in the matrix (3) the blocks B_m and C_0 as well as rows corresponding to the first and the last row of (1) remain unchanged.

The same approach may be repeated and applied to (3) and after $\log_2 m$ steps a 2×2 block matrix (or a 4×4 block matrix if expressed in terms of $A_{j,i}$, $B_{j,i}$, $C_{j,i}$ and $D_{j,i}$) is obtained and factorized using Gaussian Elimination with partial pivoting.

3. Numerical tests

We have studied the efficiency of the parallel algorithm presented in the previous section (we will refer to it as ABDCR) on ABD linear systems arising from the discretization of some known boundary value problems, and compared to the single-processor performance of SOLVEBLOCK. Both algorithms have been coded in Fortran and executed on a MicroWay Multiputer with 16 processors, each one with 1Mb of local memory.

We have considered the following three test problems:

- problem 1: system of two first order BVP's – blocks of size 2×2 (example 1 from Ascher and Chan [5])
- problem 2: system of three first order BVP's – blocks of size 3×3 (problem 1A from Wright [15])
- problem 3: system of five first order BVP's – blocks of size 5×5 (problem 4A from Wright [15])

In all cases, we considered these problems with separated boundary conditions in order to allow the application of SOLVEBLOCK which is tailored for ABD matrices without additional corner blocks. In a separate experiment we have applied the new algorithm to problems 1-3 with non-separated boundary conditions and obtained the same timings. It should be also mentioned that no numerical instabilities have been observed (see also [4] for more details).

Tables 1 and 2 contain the execution times for the considered solvers. Table 1 is devoted to single-processor execution times for $m = 32, 128, 512$ blocks. Table 2 contains parallel execution times for the ABDCR solver for $m = 32, 128, 512$ blocks and for $p = 4, 16$ processors.

Table 1: scalar execution times

	SOLVEBLOCK			ABDCR		
problem	$m = 32$	$m = 128$	$m = 512$	$m = 32$	$m = 128$	$m = 512$
1	397	1572	6282	801	3174	12656
2	717	2838	11321	1311	5149	20504
3	2249	8919	35604	2901	11388	45321

Table 2: parallel execution times of the ABDCR solver

	4 processors			16 processors		
problem	$m = 32$	$m = 128$	$m = 512$	$m = 32$	$m = 128$	$m = 512$
1	318	913	3283	391	545	1132
2	492	1456	5310	530	771	1737
3	1031	3166	11671	943	1480	3608

It can be observed that SOLVEBLOCK outperforms ABDCR in all cases. This can be explained by the fact that the arithmetical complexity of the proposed algorithm is slightly higher (see [4]). It should be also observed that (as expected) the ratio of times remains relatively constant for the increasing value of m. For a given number of processors the speed-up (calculated against the single processor performance of ABDCR) increases as the value of

m increases and reaches 3.88 on 4 processors and 12.56 on 16 processors. At the same time the speed-up calculat against SOLVEBLOCK reaches 3.05 on 4 processors and 9.86 on 16 processors.

4. Conclusion

There exist a number of different parallel aproaches toward the solution of ABD linear systems. We have present one of them that is relatively competitive in terms of its parallel performance as well as numerical properties. Wh is needed now is a comparative study of all existing approaches on a large number of problems and on a variety parallel architectures.

5. References

1 Amodio, P., Brugnano, L., Politi, T.: Parallel factorizations for tridiagonal matrices, SIAM J. Numer. Anal. : (1993), 813–823.

2 Amodio, P., Mazzia, F.: Backward error analysis of cyclic reduction for the solution of tridiagonal systems, Math. Com 62 (1994), 601–617.

3 Amodio, P., Paprzycki, M.: Parallel solution of almost block diagonal systems on a Hypercube, Linear Algebra Appl (in press).

4 Amodio, P., Paprzycki, M.: A cyclic reduction approach to the numerical solution of boundary value ODE's, Report 19/95 of the Dipartimento di Matematica, Università di Bari, Bari, Italy.

5 Ascher, U., M., Chan, S., Y., P.: On the parallel methods for boundary value ODE's, Computing 46 (1991), 1-17.

6 De Boor, C., Weiss, R.: SOLVEBLOCK: A package for solving almost block diagonal linear systems, ACM Trans. Mat Software 6 (1980), 80–87.

7 Diaz, J., C., Fairweather, G., Keast, P.: Fortran packages for solving certain almost block diagonal linear systems l modified alternate row and column elimination, ACM Trans. Math. Software 9 (1983), 358–375.

8 Gladwell, I., Paprzycki, M.: Parallel solution of almost block diagonal systems on the CRAY Y-MP using level 3 BLA J. Comput. Appl. Math. 45 (1993), 181-189.

9 Jackson, K., R., Pancer, R., N.: The parallel solution of ABD systems arising in numerical methods for BVPs for ODE Technical report n. 255/91 of the Department of Computer Science, University of Toronto, Toronto, Ontario, Cananda.

10 Kreiss, H., O., Nichols, K., Brown, D.: Numerical methods for stiff two-point boundary value problems, SIAM Numer. Anal. 23 (1986), 325-368.

11 Paprzycki, M., Gladwell, I.: Solving almost block diagonal systems on parallel computers, Parallel Comput. 17 (1991 133-153.

12 Varah, J., M.: Alternate row and column elimination for solving certain linear systems, SIAM J. Numer. Anal. 1 (1976), 71–75.

13 Wright, K.: Parallel treatment of block-bidiagonal matrices in the solution of ordinary differential boundary valu problems, J. Comput. Appl. Math. 45 (1993), 191-200.

14 Wright, S.: Stable parallel algorithms for two-point boundary value problems, SIAM J. Sci. Statist. Comput. 13 (1992 742-764.

15 Wright, S.: Stable parallel elimination for boundary value ODE's, Numer. Math. 67 (1994), 521–536.

Addresses: Pierluigi Amodio, Università di Bari, Dipartimento di Matematica, Via Orabona 4, I-70125 Bari Italy.

Marcin Paprzycki, University of Texas of the Permian Basin, Department of Mathematics and Computer Science, Odessa, Texas, TX 79762, USA.