**General Mathematics Vol. 18, No. 1 (2010), 25–32**

# Remarks on the cyclic reduction method [1]

Ioana Chiorean

**Abstract**

A tridiagonal system of equations appears often by discretization of differential equations. Several methods for solving such a system are known. The aim of this paper is to compare, from computational time point of view, the cyclic reduction method and an alternative, which uses recurrence relations, by using parallel calculus.

**2000 Mathematics Subject Classification:** 65F05, 65M06, 65Y05.

**Key words and phrases:** Numerical analysis, differential equation, parallel calculus.

[1] *Received 05 June, 2009*

*Accepted for publication (in revised form) 15 September, 2009*

# 1 Introduction

A tridiagonal system of equations has the following form:

$$\text{(1)} \qquad \begin{bmatrix} b_1 & c_1 & & & & 0 \\ a_2 & b_2 & c_2 & & & \\ & a_3 & b_3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & c_{n-1} \\ 0 & & & & a_n & b_n \end{bmatrix}$$

or

$$\text{(2)} \qquad a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i, \quad i = \overline{1, n}$$

with $a_1 = 0$ and $c_n = 0$.

Due to the fact that such a system appears very often in all sort of problems, there are many approaches for solving it, both with serial and parallel computers. It is well known the Thomas algorithm, which generates the solution after $O(n)$ operations (see [2]), but also algorithms which generate logarithmical time of execution (see [1]), which use different parallel techniques, as cyclic even-odd reduction.

In what follows, we take into account the Thomas algorithm and show that its coefficient can be computed also in logarithmical time.

# 2 Thomas Algorithm

It is also called "the tridiagonal matrix algorithm".

Starting with a system of type (1), the method has the following steps:

**Step 1.** Modify the coefficients of system (1) according with formulas:

$$c'_i = \begin{cases} \dfrac{c_1}{b_1}, & i = 1 \\ \\ \dfrac{c_i}{b_i - c'_{i-1}a_i}, & i = 2, 3, \ldots, n-1 \end{cases} \tag{3}$$

$$d'_i = \begin{cases} \dfrac{d_1}{b_1}, & i = 1 \\ \\ \dfrac{d_i - d'_{i-1}a_i}{b_i - c'_{i-1}a_i}, & i = 2, 3, \ldots, n \end{cases} \tag{4}$$

**Step 2.** The solution is obtained by back substitution:

$$x_n = d'_n$$

$$x_i = d'_i - c'_i x_{i+1},\ i = n-1, n-2, \ldots, 1$$

It is clear that the solution of our system is obtained after $O(n)$ operations.

# 3 The Cyclic Even-Odd Reduction Method

Another possibility for solving a system of type (1) is given by a technique called Cyclic Even-Odd Reduction.

In the general case, considering a first order recurrence

$$x_i = a_i + b_i x_{i-1} \tag{5}$$

where $a_i$ and $b_i$ are known $(i = \overline{1,n})$ and $x_1, \ldots, x_n$ are to be calculated, with $x_0 = \alpha$, given, we set $a_0 = \alpha$ and $a_i = 0$ for $i < 0$ and $b_i = 0$ for $i \leq 0$. Then (5) will also hold for $i \leq 0$, with $x_i = 0$ for $i < 0$.

In (3) we replace $i$ by $i-1$ and we get:

$$x_i = a_i + b_i(a_{i-1} + b_{i-1}x_{i-2}) = a_i^{(1)} + b_i^{(1)}x_{i-2} \tag{6}$$

Then, in (6) we replace $i$ by $i-2$ and we get

$$x_i = a_i^{(1)} + b_i^{(1)}(a_{i-2}^{(1)} + b_{i-2}^{(1)}x_{i-4}) = a_i^{(2)} + b_i^{(2)}x_{i-4} \tag{7}$$

Continuing in this way, $x_i$ can be expressed successively in terms on $x_{i-1}$, $x_{i-2}$, $x_{i-4}$, $x_{i-8}, \ldots$ and after at most $k$ steps, where $k = \lceil \log_2 n \rceil$, the index $i-2^k$ will be negative or zero, and therefore $b_{i-2^k}^{(k)} = 0$, then $x_i = a_{i-2^k}^{(k)}$.

How can we apply this method to solve a system of type (1)? Like in [3], consider the three equations involving $x_i$:

$$a_{i-1}x_{i-2} + b_{i-1}x_{i-1} + c_{i-1}x_i = y_{i-1} \tag{8}$$

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = y_i \tag{9}$$

$$a_{i+1}x_i + b_{i+1}x_{i+1} + c_{i+1}x_{i+2} = y_{i+1} \tag{10}$$

By subtracting suitable multiples of equations (8) and (10) from (9), we can obtain an equation of the form

$$a_i^{(1)}x_{i-2} + b_i^{(1)}x_i + c_i^{(1)}x_{i+2} = y_i^{(1)} \tag{11}$$

By similar manipulation of three equations of the form (11) we obtain

$$a_i^{(2)}x_{i-4} + b_i^{(2)}x_i + c_i^{(2)}x_{i+4} = y_i^{(2)} \tag{12}$$

and so on. Setting $x_i = a_i = c_i = y_i = 0$ and $b_i = 1$ when $i$ is outside the relevant range, after $k = \lceil \log_2 n \rceil$ steps, we reach the equation:

$$b_i^k x_i = y_i^{(k)} \tag{13}$$

giving the value $x_i$.

# 4 Thomas Algorithm revisited

Let's consider, again, the system (1) and the determinant of the matrix

$$(14)\qquad \begin{vmatrix} b_1 & c_1 & & & & 0 \\ a_2 & b_2 & c_2 & & & \\ & a_3 & b_3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & c_{n-1} \\ 0 & & & & a_n & b_n \end{vmatrix}$$

Then

$$\begin{aligned} \det{}_1 &= b_1 \\ \det{}_2 &= \begin{vmatrix} b_1 & c_1 \\ a_2 & b_2 \end{vmatrix} = b_2 b_1 - a_2 c_1 = b_2 \det{}_1 - \alpha_1 \det{}_0, \\ & \qquad\qquad\qquad \text{for } \det{}_0 = 1 \text{ and } \alpha_1 = a_2 c_1 \\ \det{}_3 &= \begin{vmatrix} b_1 & c_1 & 0 \\ a_2 & b_2 & c_2 \\ 0 & a_3 & b_3 \end{vmatrix} = b_3 \det{}_2 - \beta_2 \det{}_1 \text{ with } \beta_2 = a_3 c_2 b_1 \\ &\vdots \end{aligned}$$

and so on.

Then

$$(15)\qquad \det{}_n = b_n \det{}_{n-1} - \alpha_n \det{}_{n-1},$$

with $\alpha_n$ apriori computed.

Having these observations in mind, we may observe that formulas (3) becomes:

$$
c'_i = \begin{cases} c'_1 = \dfrac{c_1}{\det_1}, & i = 1 \\[2ex] c'_i = \dfrac{c_i \det_{i-1}}{\det_i}, & i = 2, 3, \ldots, n-1 \end{cases} \tag{16}
$$

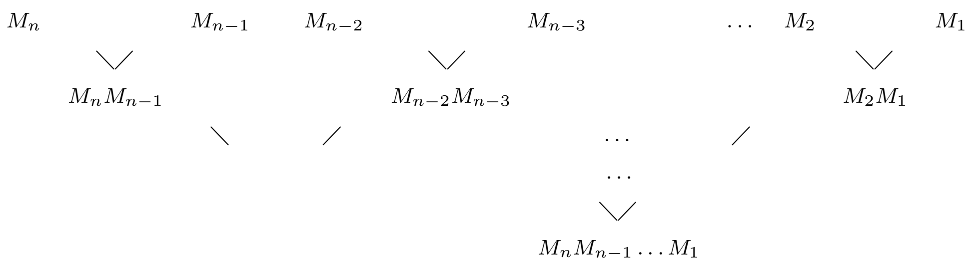
Formula (15) is a linear recurrence with two terms which can be perform, in parallel, in the following way:

$$
\begin{aligned}
\begin{bmatrix} \det_n \\ \det_{n-1} \end{bmatrix} &= \begin{bmatrix} b_n & \alpha_n \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \det_{n-1} \\ \det_{n-2} \end{bmatrix} \\
&= \begin{bmatrix} b_n & \alpha_n \\ 1 & 0 \end{bmatrix} \begin{bmatrix} b_{n-1} & \alpha_{n-1} \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \det_{n-2} \\ \det_{n-3} \end{bmatrix} = \cdots = \\
&= \begin{bmatrix} b_n & \alpha_n \\ 1 & 0 \end{bmatrix} \begin{bmatrix} b_{n-1} & \alpha_{n-1} \\ 1 & 0 \end{bmatrix} \cdots \begin{bmatrix} b_1 & \alpha_1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \det_1 \\ \det_0 \end{bmatrix}
\end{aligned} \tag{17}
$$

So, at every moment of time, we may compute $\det_i$, $i = \overline{1, n}$, according with relation (17). The computation takes place only at matricial level, in the following way: if we denote

$$
M_i = \begin{bmatrix} b_i & \alpha_i \\ 1 & 0 \end{bmatrix}, \quad i = \overline{1, n},
$$

the matrix multiplication in (17) may be performed on a binary tree network with $2^n - 1$ processors:




After $\lceil \log_2 n \rceil$ parallel steps, the final product will be in the root. The parallel technique used here is called "double recursive technique" (see [3]).

Concerning the coefficients $d_i'$, $i = \overline{1,n}$ in (4), a similar result may be obtained, by considering the following determinants:

$$\det_1' = b_1$$

$$\det_2' = \begin{vmatrix} b_1 & d_1 \\ a_2 & d_2 \end{vmatrix}$$

$$\det_3' = \begin{vmatrix} b_1 & c_1 & d_1 \\ a_2 & b_2 & d_2 \\ 0 & a_3 & d_3 \end{vmatrix}$$

$$\vdots$$

$$\det_n' = \begin{vmatrix} b_1 & c_1 & \dots & \dots & 0 & d_1 \\ a_2 & b_2 & c_2 & \dots & 0 & d_2 \\ 0 & a_3 & b_3 & \dots & 0 & d_3 \\ \vdots & & & \ddots & & \\ 0 & 0 & \dots & \dots & b_n & d_n \end{vmatrix}$$

Then

$$d_i' = \begin{cases} d_1' = \dfrac{d_1}{\det_1}, & i = 1 \\ \\ d_i' = \dfrac{\det_i'}{\det_i}, & i = 2, 3, \ldots, n \end{cases}$$

## 5 Conclusions

In this paper, we give an anternative to the cyclic even-odd reduction scheme for solving a tridiagonal system of equations, by using Thomas algorithm and the double recursive technique. Our approach generates also an execution time of order $O(\log_2 n)$.

## References

[1] I. Chiorean, *Calcul paralel*, Ed. Microinformatica, 1995.

[2] S.D. Conte, C. de Boor, *Elementary Numerical Analysis*, Mc Graw-Hill, New York, 1972.

[3] J.J. Modi, *Parallel Algorithms and Matrix Computation*, Clarendon Press, 1988.

**Ioana Chiorean**
Babeş-Bolyai University
Department of Applied Mathematics
Kogălniceanu 1, Cluj-Napoca, Romania
e-mail: robica_chiorean@yahoo.com