ACKNOWLEDGMENTS

The authors would like to thank Reviewer G for improving the article, and also Profs. I.F. Blake and M.Z. Wang for their useful comments during the manuscript preparation.

This work was supported, in part, by the University of Waterloo under a start-up grant and, in part, by the Canadian Institute for Telecommunications Research under the NCE program. A preliminary version of part of this article was presented at the 16th Biennial Symposium on Communications, Kingston, Ontario, Canada. [1]

REFERENCES

[1] M.A. Hasan and V.K. Bhargava, "A VLSI architecture for a low complexity rate-adaptive Reed-Solomon encoder," *Proc. 16th Biennial Symp. Communications*, Kingston, Ontario, pp. 331–334, May 1992.

[2] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*. Cambridge: Cambridge Univ. Press, 1986.

[3] E.R. Berlekamp, "Bit-serial Reed-Solomon encoder," *IEEE Trans. Information Theory*, vol. 28, pp. 869–874, Nov. 1982.

[4] M.Z. Wang and I.F. Blake, "Bit serial multiplication in finite fields," *SIAM J. Disc. Math.*, vol. 3, pp. 140–148, Feb. 1990.

[5] M.A. Hasan and V.K. Bhargava, "Division and bit-serial multiplication over GF(q^m)," *IEEE Proc. Part E*, vol. 139, pp. 230–236, May 1992.

[6] I.-S. Hsu, I.S. Reed, T.K. Truong, K. Wang, C.S. Yeh, and L.J. Deutsch, "The VLSI implementation of a Reed-Solomon encoder using Berlekamp's bit-serial multiplier algorithm," *IEEE Trans. Computers*, vol. 33, pp. 906–911, Oct. 1984.

[7] N. Zierler and J. Brilluart, "On primitive trinomials (mod 2)," *Inform. and Contr.*, vol. 13, pp. 541–554, 1968.

Parallel Minimal Norm Method for Tridiagonal Linear Systems

E. Dekker and L. Dekker

***Abstract*—Based on the parallel minimal norm method an algorithm is derived to solve tridiagonal linear systems with a high degree of parallelism. No conditions need to be posed with respect to the system. Experiments indicate that the numerical stability of the algorithm is similar to Gaussian elimination with partial pivoting.**

***Index Terms*—Parallel algorithms, parallel minimal norm method, tridiagonal linear systems, row-oriented orthogonalization, structural orthogonality.**

I. INTRODUCTION

Large tridiagonal linear systems arise in many fields of numerical computation. New algorithms for these systems have been developed that take advantage of the architecture of parallel computers. Ortega [9] gives an overview. Parallel algorithms include Cyclic Reduction [8], Wang's partition method [13], Recursive Doubling [10] and the methods of Sun [11] and Bondeli [2]. At this moment parallel machines are available that consist of many thousands of processors ([6], [12]). As in the near future the number of processors will increase even more, the degree of parallelism becomes a dominant property of algorithms. The algorithm presented here possesses a high degree of parallelism.

The parallel minimal norm method [3] is based on an inner product equation representation of a linear system. Each equation is an inner product of a known vector and the unknown solution that must satisfy a right-hand side. By orthogonalizing the inner product equations the solution is determined. With tridiagonal linear systems structural orthogonality is used such that significant arithmetic savings are achieved and a high degree of parallelism is available. The algorithm is based on a *divide-and-conquer* strategy [1].

II. ALGORITHM

A tridiagonal linear system in inner product notation looks like

$$\left.\begin{aligned}(\underline{a}_1, \underline{x}) &= b_1,\\ (\underline{a}_2, \underline{x}) &= b_2,\\ (\underline{a}_3, \underline{x}) &= b_3,\\ &\vdots\\ (\underline{a}_n, \underline{x}) &= b_n;\end{aligned}\right\} \text{ with } \left\{\begin{aligned}\underline{a}_1^T &= (a_{11}, a_{12} \quad\quad\quad\quad\quad\quad\quad\quad\quad\quad),\\ \underline{a}_2^T &= (a_{21}, a_{22}, a_{23} \quad\quad\quad\quad\quad\quad\quad),\\ \underline{a}_3^T &= (\quad a_{32}, a_{33}, a_{34} \quad\quad\quad\quad\quad\quad),\\ &\vdots \quad\quad\quad\quad\quad\quad\quad\quad\quad\ddots\\ \underline{a}_n^T &= (\quad\quad\quad\quad\quad\quad\quad\quad a_{n,n-1}, a_{nn});\end{aligned}\right. \tag{1}$$

for $\underline{a}_1, \ldots, \underline{a}_n, \underline{x}, \underline{b} \in \mathcal{R}^n$. The structural orthogonality of the vectors satisfies $(\underline{a}_i, \underline{a}_j) = 0$, for $|i - j| \geq 3$. Therefore the system is divided into three sets of equations. Kamath [7] applies the same approach to introduce parallelism in block-tridiagonal linear systems. The size of the system is assumed to be $n = 3.2^m$ with $m \in N$. The initial set S consists of all n inner product equations, or $S = \{1, 2, 3, \ldots, n\}$. The following division is applied.

Manuscript received July 31, 1992; revised Nov. 20, 1994.

E. Dekker is with the Faculty of Applied Physics, Technical University of Delft, Lorentzweg 1, 2628 CJ Delft, The Netherlands.

L. Dekker is with the Faculty of Technical Mathematics and Computer Science, Technical University of Delft, Mekelweg 4, 2628 CD Delft, The Netherlands.

IEEECS Log Number C95058.

$$
\begin{aligned}
&S = \{1, 2, 3, \ldots, n\},\\
&\textbf{divide } S \rightarrow S_1, S_2, S_3;\\
&S_1 := \{2, 5, 8, \ldots, n-1\},\\
&S_2 := \{1, 4, 7, \ldots, n-2\},\\
&S_3 := \{3, 6, 9, \ldots, n\}.
\end{aligned} \tag{2}
$$

Consequently the three sets consist of the following inner product equations.

$$
\begin{array}{lll}
S_1: & (\underline{a}_2, \underline{x}) = b_2, & S_2: \quad (\underline{a}_1, \underline{x}) = b_1, \qquad S_3: \quad (\underline{a}_3, \underline{x}) = b_3,\\
& (\underline{a}_5, \underline{x}) = b_5, & \qquad\quad (\underline{a}_4, \underline{x}) = b_4, \qquad\quad (\underline{a}_6, \underline{x}) = b_6,\\
& \vdots & \qquad\quad \vdots \qquad\qquad\quad \vdots\\
& (\underline{a}_{n-1}, \underline{x}) = b_{n-1}; & \qquad (\underline{a}_{n-2}, \underline{x}) = b_{n-2}; \qquad (\underline{a}_n, \underline{x}) = b_n.
\end{array} \tag{3}
$$

The number of equations within each set equals $n_i = \frac{n}{3}$ for $i = 1, 2, 3$.

In order to be relieved from the absolute indices of the inner product equations, the symbolic notation is used. The index s_{ij} refers to the jth inner product equation in set S_i. In this way the notation can be kept simple.

The nonzero structures of the vectors of the inner product equations of the three sets are

$$
\begin{array}{lll}
S_1: & & \\
\underline{a}_{s_{11}}^T & = & (*, *, * \qquad\qquad\quad),\\
\underline{a}_{s_{12}}^T & = & (\qquad *, *, * \qquad),\\
\qquad\vdots & & \qquad\qquad\quad \ddots\\
\underline{a}_{s_{1n_1}}^T & = & (\qquad\qquad\quad *, *, *);\\
S_2: & & \\
\underline{a}_{s_{21}}^T & = & (*, * \qquad\quad),\\
\underline{a}_{s_{22}}^T & = & (\quad *, *, * \qquad),\\
\qquad\vdots & & \qquad\quad \ddots\\
\underline{a}_{s_{2n_2}}^T & = & (\qquad\quad *, *, *);\\
S_3: & & \\
\underline{a}_{s_{31}}^T & = & (*, *, * \qquad\quad),\\
\underline{a}_{s_{32}}^T & = & (\qquad *, *, * \qquad),\\
\qquad\vdots & & \qquad\quad \ddots\\
\underline{a}_{s_{3n_3}}^T & = & (\qquad\quad *, *).
\end{array} \tag{4}
$$

Within each set the vectors of the inner product equations are orthogonal. However this is not the case for vectors of different sets. So orthogonalization is needed between the sets. Based on symmetry considerations the sets S_2 and S_3 are merged with respect to S_1.

Each vector in S_2 and S_3 only overlaps with the two neighboring vectors in S_1. Exceptions are the first vector in S_2 and the last vector in S_3 which overlap with only one vector in S_1. The following equations show the mergings of S_2 and S_3. The mergence of S_2 is represented by **merge**$(S_2|S_1)$ in a compact notation.

$$
\underline{a}_{s_{21}} := \underline{a}_{s_{21}} - \frac{\left(\underline{a}_{s_{21}}, \underline{a}_{s_{11}}\right)}{\left\|\underline{a}_{s_{11}}\right\|^2}\underline{a}_{s_{11}}, \tag{5}
$$

$$
\underline{a}_{s_{2j}} := \underline{a}_{s_{2j}} - \frac{\left(\underline{a}_{s_{2j}}, \underline{a}_{s_{1,j-1}}\right)}{\left\|\underline{a}_{s_{1,j-1}}\right\|^2}\underline{a}_{s_{1,j-1}} - \frac{\left(\underline{a}_{s_{2j}}, \underline{a}_{s_{1j}}\right)}{\left\|\underline{a}_{s_{1j}}\right\|^2}\underline{a}_{s_{1j}}, \text{ for } j = 2, 3, \ldots, n_2; \tag{6}
$$

$$
b_{s_{21}} := b_{s_{21}} - \frac{\left(\underline{a}_{s_{21}}, \underline{a}_{s_{11}}\right)}{\left\|\underline{a}_{s_{11}}\right\|^2}b_{s_{11}}, \tag{7}
$$

$$
b_{s_{2j}} := b_{s_{2j}} - \frac{\left(\underline{a}_{s_{2j}}, \underline{a}_{s_{1,j-1}}\right)}{\left\|\underline{a}_{s_{1,j-1}}\right\|^2}b_{s_{1,j-1}} - \frac{\left(\underline{a}_{s_{2j}}, \underline{a}_{s_{1j}}\right)}{\left\|\underline{a}_{s_{1j}}\right\|^2}b_{s_{1j}}, \text{ for } j = 2, 3, \ldots, n_2. \tag{8}
$$

The mergence of S_3 is denoted by **merge**$(S_3|S_1)$ in compact notation.

$$
\underline{a}_{s_{3j}} := \underline{a}_{s_{3j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{1j}}\right)}{\left\|\underline{a}_{s_{1j}}\right\|^2}\underline{a}_{s_{1j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{1,j+1}}\right)}{\left\|\underline{a}_{s_{1,j+1}}\right\|^2}\underline{a}_{s_{1,j+1}}, \text{ for } j = 1, 2, \ldots, n_3 - 1; \tag{9}
$$

$$
\underline{a}_{s_{3n_3}} := \underline{a}_{s_{3n_3}} - \frac{\left(\underline{a}_{s_{3n_3}}, \underline{a}_{s_{1n_1}}\right)}{\left\|\underline{a}_{s_{1n_1}}\right\|^2}\underline{a}_{s_{1n_1}}, \tag{10}
$$

$$
b_{s_{3j}} := b_{s_{3j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{1j}}\right)}{\left\|\underline{a}_{s_{1j}}\right\|^2}b_{s_{1j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{1,j+1}}\right)}{\left\|\underline{a}_{s_{1,j+1}}\right\|^2}b_{s_{1,j+1}}, \text{ for } j = 1, 2, \ldots, n_3 - 1; \tag{11}
$$

$$
b_{s_{3n_3}} := b_{s_{3n_3}} - \frac{\left(\underline{a}_{s_{3n_3}}, \underline{a}_{s_{1n_1}}\right)}{\left\|\underline{a}_{s_{1n_1}}\right\|^2}b_{s_{1n_1}}. \tag{12}
$$

The resulting vectors of the inner product equations of S_2 and S_3 are now orthogonal to those of S_1. The nonzero patterns of the vectors of S_2 and S_3 are

$$
\begin{array}{lll}
S_2: & \underline{a}_{s_{21}}^T & = (*, *, * \qquad\qquad\quad),\\
& \underline{a}_{s_{22}}^T & = (*, *, *, *, *, * \qquad),\\
& \qquad\vdots & \qquad\qquad\quad \ddots\\
& \underline{a}_{s_{2n_2}}^T & = (\qquad\qquad\quad *, *, *, *, *, *);\\
S_3: & \underline{a}_{s_{31}}^T & = (*, *, *, *, *, * \qquad\quad),\\
& \underline{a}_{s_{32}}^T & = (\qquad *, *, *, *, *, * \qquad),\\
& \qquad\vdots & \qquad\qquad\quad \ddots\\
& \underline{a}_{s_{3n_3}}^T & = (\qquad\qquad\quad *, *, *).
\end{array} \tag{13}
$$

The average bandwidth of the vectors is doubled with respect to the original vectors. The structural properties of the vectors of both sets satisfy $(\underline{a}_{s_{ij}}, \underline{a}_{s_{ik}}) = 0$, for $|j - k| \geq 2$. Therefore for both sets a division into two sets becomes obvious. The set S_2 is divided into a new S_2 and a new S_4. The indices s_{2j} with j even are contained in the new S_2. The indices s_{2j} with j odd move to S_4. For set S_3 the division is reverse. The new S_3 contains the indices s_{3k} with k odd and the new S_5 contains the indices s_{3k} with k even.

$$
\begin{aligned}
&S_2 = \{s_{21}, s_{22}, \ldots, s_{2n_2}\}, \qquad S_3 = \{s_{31}, s_{32}, \ldots, s_{3n_3}\},\\
&\textbf{divide } S_2 \rightarrow S_2, S_4; \qquad\quad \textbf{divide } S_3 \rightarrow S_3, S_5;\\
&S_2 := \{s_{22}, s_{24}, \ldots, s_{2n_2}\}, \qquad S_3 := \{s_{31}, s_{33}, \ldots, s_{3,n_3-1}\},\\
&S_4 := \{s_{21}, s_{23}, \ldots, s_{2,n_2-1}\}; \quad S_5 := \{s_{32}, s_{34}, \ldots, s_{3n_3}\}.
\end{aligned} \tag{14}
$$

After the divisions the number of equations in S_2, S_3, S_4, and S_5 is halved, it satisfies $n_i = \frac{n}{6}$ for $i = 2, 3, 4, 5$. The nonzero patterns of the vectors of the new S_2 and S_3 are identical.

$$S_i:\ \begin{array}{rcl} \underline{a}_{s_{i1}}^T &=& (*,*,*,*,*,* \qquad\qquad\qquad\qquad\qquad\qquad\quad), \\ \underline{a}_{s_{i2}}^T &=& (\qquad\qquad *,*,*,*,*,* \qquad\qquad\qquad\quad), \\ \vdots & & \qquad\qquad\qquad\qquad\qquad\qquad\quad \ddots \qquad\qquad \text{for i = 2, 3.} \\ \underline{a}_{s_{in_i}}^T &=& (\qquad\qquad\qquad\qquad\qquad\qquad *,*,*,*,*,*), \end{array} \tag{15}$$

The nonzero patterns of the vectors of S_4 and S_5 look like

$$S_4:\ \begin{array}{rcl} \underline{a}_{s_{41}}^T &=& (*,*,* \qquad\qquad\qquad\qquad\qquad\qquad\quad), \\ \underline{a}_{s_{42}}^T &=& (\qquad\quad *,*,*,*,*,* \qquad\qquad\qquad\quad), \\ \vdots & & \qquad\qquad\qquad\qquad\qquad\qquad\quad \ddots \\ \underline{a}_{s_{4n_4}}^T &=& (\qquad\qquad\qquad\qquad\qquad\quad *,*,*,*,*,* \ ); \end{array} \tag{16}$$

$$S_5:\ \begin{array}{rcl} \underline{a}_{s_{51}}^T &=& (\ *,*,*,*,*,*,* \qquad\qquad\qquad\qquad\quad), \\ \underline{a}_{s_{52}}^T &=& (\qquad\qquad\qquad\quad *,*,*,*,*,* \qquad\qquad\quad), \\ \vdots & & \qquad\qquad\qquad\qquad\qquad\qquad\quad \ddots \\ \underline{a}_{s_{5n_5}}^T &=& (\qquad\qquad\qquad\qquad\qquad\qquad\quad *,*,*). \end{array} \tag{17}$$

The vectors within S_2 are orthogonal. This holds for S_3 as well. However corresponding vectors in S_2 and S_3 are not orthogonal. Therefore a *mergence* has to be done. The merging of S_3 with respect to S_2 involves only one orthogonalization per inner product equation. The nonzero pattern of S_3 does not change. In compact notation this mergence is **merge**$(S_3|S_2)$.

$$\underline{a}_{s_{3j}} := \underline{a}_{s_{3j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{2j}}\right)}{\left\|\underline{a}_{s_{2j}}\right\|^2}\underline{a}_{s_{2j}}, \text{ for } j = 1, 2, \ldots, n_3; \tag{18}$$

$$b_{s_{3j}} := b_{s_{3j}} - \frac{\left(\underline{a}_{s_{3j}}, \underline{a}_{s_{2j}}\right)}{\left\|\underline{a}_{s_{2j}}\right\|^2}b_{s_{2j}}, \text{ for } j = 1, 2, \ldots, n_3. \tag{19}$$

Again there exists symmetry between the nonzero patterns of S_4 and S_5 and those of S_2 and S_3. Therefore a similar procedure can be followed as with **merge**$(S_2|S_1)$ and **merge**$(S_3|S_1)$. The only difference is that the mergings of S_4 and S_5 are not with respect to one but with respect to two sets. In compact notation the mergence of S_4 is **merge**$(S_4|S_2, S_3)$.

$$\underline{a}_{s_{41}} := \underline{a}_{s_{41}} - \sum_{i=2}^{3}\frac{\left(\underline{a}_{s_{41}}, \underline{a}_{s_{i1}}\right)}{\left\|\underline{a}_{s_{i1}}\right\|^2}\underline{a}_{s_{i1}}, \tag{20}$$

$$\underline{a}_{s_{4j}} := \underline{a}_{s_{4j}} - \sum_{i=2}^{3}\left[\frac{\left(\underline{a}_{s_{4j}}, \underline{a}_{s_{i,j-1}}\right)}{\left\|\underline{a}_{s_{i,j-1}}\right\|^2}\underline{a}_{s_{i,j-1}} + \frac{\left(\underline{a}_{s_{4j}}, \underline{a}_{s_{ij}}\right)}{\left\|\underline{a}_{s_{ij}}\right\|^2}\underline{a}_{s_{ij}}\right], \text{ for } j = 2, 3, \ldots, n_4; \tag{21}$$

$$b_{s_{41}} := b_{s_{41}} - \sum_{i=2}^{3}\frac{\left(\underline{a}_{s_{41}}, \underline{a}_{s_{i1}}\right)}{\left\|\underline{a}_{s_{i1}}\right\|^2}b_{s_{i1}}, \tag{22}$$

$$b_{s_{4j}} := b_{s_{4j}} - \sum_{i=2}^{3}\left[\frac{\left(\underline{a}_{s_{4j}}, \underline{a}_{s_{i,j-1}}\right)}{\left\|\underline{a}_{s_{i,j-1}}\right\|^2}b_{s_{i,j-1}} + \frac{\left(\underline{a}_{s_{4j}}, \underline{a}_{s_{ij}}\right)}{\left\|\underline{a}_{s_{ij}}\right\|^2}b_{s_{ij}}\right], \text{ for } j = 2, 3, \ldots, n_4. \tag{23}$$

In compact notation the mergence of S_5 is **merge**$(S_5|S_2, S_3)$.

$$\underline{a}_{s_{5j}} := \underline{a}_{s_{5j}} - \sum_{i=2}^{3}\left[\frac{\left(\underline{a}_{s_{5j}}, \underline{a}_{s_{ij}}\right)}{\left\|\underline{a}_{s_{ij}}\right\|^2}\underline{a}_{s_{ij}} + \frac{\left(\underline{a}_{s_{5j}}, \underline{a}_{s_{i,j+1}}\right)}{\left\|\underline{a}_{s_{i,j+1}}\right\|^2}\underline{a}_{s_{i,j+1}}\right], \text{ for } j = 1, 2, \ldots, n_5 - 1; \tag{24}$$

$$\underline{a}_{s_{5n_5}} := \underline{a}_{s_{5n_5}} - \sum_{i=2}^{3}\frac{\left(\underline{a}_{s_{5n_5}}, \underline{a}_{s_{in_5}}\right)}{\left\|\underline{a}_{s_{in_5}}\right\|^2}\underline{a}_{s_{in_5}}, \tag{25}$$

$$b_{s_{5j}} := b_{s_{5j}} - \sum_{i=2}^{3}\left[\frac{\left(\underline{a}_{s_{5j}}, \underline{a}_{s_{ij}}\right)}{\left\|\underline{a}_{s_{ij}}\right\|^2}b_{s_{ij}} + \frac{\left(\underline{a}_{s_{5j}}, \underline{a}_{s_{i,j+1}}\right)}{\left\|\underline{a}_{s_{i,j+1}}\right\|^2}b_{s_{i,j+1}}\right], \text{ for } j = 1, 2, \ldots, n_5 - 1; \tag{26}$$

$$b_{s_{5n_5}} := b_{s_{5n_5}} - \sum_{i=2}^{3}\frac{\left(\underline{a}_{s_{5n_5}}, \underline{a}_{s_{in_5}}\right)}{\left\|\underline{a}_{s_{in_5}}\right\|}b_{s_{in_5}}. \tag{27}$$

S_2 and S_3 are now orthogonal to S_4 and S_5. The nonzero structures of S_4 and S_5 have an analogous symmetry as those of S_2 and S_3 after mergence with respect to S_1 (13).

$$S_4:\ \begin{array}{rcl} \underline{a}_{s_{41}}^T &=& (*,*,*,*,*,* \qquad\qquad\qquad\qquad\qquad\qquad\quad), \\ \underline{a}_{s_{42}}^T &=& (*,*,*,*,*,*,*,*,*,*,*,*,* \qquad\qquad\quad), \\ \vdots & & \qquad\qquad\qquad\qquad\qquad\qquad\quad \ddots \\ \underline{a}_{s_{4n_4}}^T &=& (\qquad\qquad\qquad\qquad\qquad\quad *,*,*,*,*,*,*,*,*,*,*,*,*); \end{array} \tag{28}$$

$$S_5:\ \begin{array}{rcl} \underline{a}_{s_{51}}^T &=& (*,*,*,*,*,*,*,*,*,*,*,*,* \qquad\qquad\qquad\quad), \\ \underline{a}_{s_{52}}^T &=& (\qquad\qquad *,*,*,*,*,*,*,*,*,*,*,*,* \qquad\qquad\quad), \\ \vdots & & \qquad\qquad\qquad\qquad\qquad\qquad\quad \ddots \\ \underline{a}_{s_{5n_5}}^T &=& (\qquad\qquad\qquad\qquad\qquad\qquad\quad *,*,*,*,*,*). \end{array} \tag{29}$$

Therefore the previous procedure of the divisions and the subsequent mergences is generalized. In compact notation each step is given by

divide $S_{2i} \rightarrow S_{2i}, S_{2i+2}$; **divide** $S_{2i+1} \rightarrow S_{2i+1}, S_{2i+3}$

merge$\left(S_{2i+1}|S_{2i}\right)$

merge$\left(S_{2i+2}|S_{2i}, S_{2i+1}\right)$; **merge**$\left(S_{2i+3}|S_{2i}, S_{2i+1}\right)$

The procedure has to be applied $\log_2(\frac{n}{3})$ times. The assumption about the size of the system n was made in order to maintain the active sets at equal size during each stage of the computation. The total number of sets is $n_s = 2\log_2(\frac{n}{3}) + 3$. The last two sets are orthogonal to all other sets. Both sets consist of only one equation with a full vector. The vectors are not yet orthogonal, therefore the last set is merged with respect to the last but one set (*in compact notation* **merge**$(S_{n_s}|S_{n_s-1})$).

To each orthogonal set belongs a minimal norm solution which is the sum of the minimal norm solutions of its orthogonal inner product equations.

$$\underline{x}_i = \sum_{j=1}^{n_i}\frac{b_{s_{ij}}}{\left\|\underline{a}_{s_{ij}}\right\|^2}\underline{a}_{s_{ij}}, \text{ for } i = 1, 2, \ldots, n_s. \tag{30}$$

This is referred to as **solve**(S_i) in compact notation. Finally the solu-

tion of the complete system, i.e., (1), is found as the sum of the minimal norm solutions of all sets.

$$\underline{x} = \sum_{i=1}^{n_s} \underline{x}_i . \tag{31}$$

In Fig. 1a the algorithm is presented in compact notation.

init: **divide** $S \rightarrow S_1, S_2, S_3$
merge$(S_2|S_1)$; **merge**$(S_3|S_1)$
loop: $i = 1, 2, \ldots, log_2(\frac{n}{3})$
divide $S_{2i} \rightarrow S_{2i}, S_{2i+2}$
divide $S_{2i+1} \rightarrow S_{2i+1}, S_{2i+3}$
merge$(S_{2i+1}|S_{2i})$
merge$(S_{2i+2}|S_{2i}, S_{2i+2})$
merge$(S_{2i+3}|S_{2i}, S_{2i+1})$
end: **merge**$(S_{n_s}|S_{n_s-1})$
solve$(S_1); \ldots;$**solve**(S_{n_s})
$\mathbf{x} := \mathbf{x}_1 + \cdots + \mathbf{x}_{n_s}$

Fig. 1a. Standard algorithm.

init: **divide** $S \rightarrow S_1, S_2, S_3$
merge$(S_2|S_1)$; **merge**$(S_3|S_1)$
solve(S_1)
$\mathbf{x} := \mathbf{x}_1$
loop: $i = 1, 2, \ldots, log_2(\frac{n}{3})$
divide$S_{2i} \rightarrow S_{2i}, S_{2i+2}$
divide $S_{2i+1} \rightarrow S_{2i+1}, S_{2i+3}$
merge$(S_{2i+1}|S_{2i})$
merge$(S_{2i+2}|S_{2i}, S_{2i+1})$
merge$(S_{2i+3}|S_{2i}, S_{2i+1})$
solve(S_{2i}); **solve**(S_{2i+1})
$\mathbf{x} := \mathbf{x} + \mathbf{x}_{2i} + \mathbf{x}_{2i+1}$
end: **merge**$(S_{n_s}|S_{n_s-1})$
solve(S_{n_s-1}); **solve**(S_{n_s})
$\mathbf{x} := \mathbf{x} + \mathbf{x}_{n_s-1} + \mathbf{x}_{n_s}$

Fig. 1b. Modified algorithm.

III. Memory Requirements

Within an orthogonal set the vectors do not overlap. The nonzero patterns of the consecutive vectors exactly succeed each other. One n element register can be used to store all vectors of a set. Consequently $2\log_2(\frac{n}{3})+3$ registers are needed for the complete algorithm. The solution and the right-hand sides require two n element registers. This brings the total amount of storage to $2n\log_2(\frac{n}{3})+5n$.

A simple modification of the algorithm (Fig. 1b) significantly reduces the memory requirements. At the start of each stage consecutive vectors in S_{2i} and S_{2i+1} overlap with half the bandwidth. After the divisions there exist four sets where the consecutive vectors within each set do not overlap. Thus four n element registers are needed for the four sets. The **merge**$(S_{2i+1}|S_{2i})$ does not change the nonzero pattern of S_{2i+1}, thus no additional storage is required. The **merge**$(S_{2i+2}|S_{2i}, S_{2i+1})$ and **merge**$(S_{2i+3}|S_{2i}, S_{2i+1})$ almost double the nonzero elements in the vectors of S_{2i+2} and S_{2i+3}. Since these vectors were stored consecutively, no vector can be overwritten without destroying data of other vectors in the same register that have not been processed yet. Therefore the new vectors of both sets must be stored in four new n element registers. Since S_{2i} and S_{2i+1} are now orthogonal with respect to all other sets, the minimal norm solutions of both sets can be determined and the sets can be disposed of because they are no longer needed. The two solutions are used to update the sum of the minimal norm solutions of the previous orthogonal sets. The memory needed for S_{2i} and S_{2i+1} and the old S_{2i+2} and S_{2i+3} can be used for sets that are created at the next stage of computation. Thus eight n element registers are needed for all sets at each stage. The intermediate solution and the right-hand sides require two additional registers. The total amount of storage is $10n$, which is of the same order as Gaussian elimination which requires $4n$ memory locations.

IV. Operation Count

Only floating point operations contribute to the operation count. In the modified algorithm only the **merge** and **solve** parts and the summation of the minimal norm solutions of the sets involve floating point operations. The **divide** part consists of indexed memory references only. Each **merge** part consists of orthogonalizations of one inner product equation with respect to another. An inner product orthogonalization is given by

$$\underline{a}_i := \underline{a}_i - \frac{(\underline{a}_i, \underline{a}_j)}{\|\underline{a}_j\|^2}\underline{a}_j, \tag{32}$$

$$b_i := b_i - \frac{(\underline{a}_i, \underline{a}_j)}{\|\underline{a}_j\|^2} b_j . \tag{33}$$

Suppose that $\underline{a}_i$ and $\underline{a}_j$ consist of p respectively q nonzero elements and their overlap is equal to r nonzero elements. If the number of floating point operations of an orthogonalization is represented by $ort(q, r)$, it satisfies $ort(q, r) = 3(q + r) + 1$. The **solve** part consists of the computation of the minimal norm solution of an orthogonal set. The minimal norm solution of the jth inner product equation of the set S_i is

$$\underline{x}_{s_{ij}} = \frac{b_{s_{ij}}}{\|\underline{a}_{s_{ij}}\|^2}\underline{a}_{s_{ij}} . \tag{34}$$

If $\underline{a}_{s_{ij}}$ consists of p nonzero elements, the computation requires $3p$ floating point operations. Since there is no overlap between the vectors within a set, the computation of the minimal norm solution of all inner product equations within the set requires $n_i.3.(\frac{n}{n_i}) = 3n$ flops. In Table I the operation counts are shown. Summation of all contributions gives the operation count of the modified algorithm.

$$(56n-4)\log_2\left(\frac{n}{3}\right) - 8\cdot\frac{n}{3} + 92. \tag{35}$$

V. Parallel Processing

The amount of parallelism within each **merge**$(S_i|S_j)$ is proportional to the number of inner product equations that belong to both sets. As the size of a set is halved after a division—the consecutive set sizes

TABLE I
OPERATION COUNT OF THE DIFFERENT PARTS OF THE MODIFIED ALGORITM

Part	Number of floating point operations
divide $S \rightarrow S_1, S_2, S_3$	–
merge$(S_2\|S_1)$; **merge**$(S_3\|S_1)$	$2\{(\frac{n}{3}-1)ort(3,1)+\frac{n}{3}ort(3,2)\}$
solve(S_1)	$3n$
$\mathbf{x} := \mathbf{x}_1$	–
divide $S_{2i} \rightarrow S_{2i}, S_{2i+2}$	–
divide $S_{2i+1} \rightarrow S_{2i+1}, S_{2i+3}$	–
merge$(S_{2i+1}\|S_{2i})$	$\frac{n}{3.2^i}ort(3.2^i, 3.2^i)$
merge$(S_{2i+2}\|S_{2i}, S_{2i+1})$	$(2\frac{n}{3.2^i}-1)\{ort(3.2^i, 3.2^{i-1})+ort(3.2^i, 3.2^i)\}$
merge$(S_{2i+3}\|S_{2i}, S_{2i+1})$	$(2\frac{n}{3.2^i}-1)\{ort(3.2^i, 3.2^{i-1})+ort(3.2^i, 3.2^i)\}$
solve(S_{2i}); **solve**(S_{2i+1})	$6n$
$\mathbf{x} := \mathbf{x} + \mathbf{x}_{2i} + \mathbf{x}_{2i+1}$	$2n$
merge$(S_{n_s}\|S_{n_s-1})$	$ort(n, n)$
solve(S_{n_s-1}); **solve**(S_{n_s})	$6n$
$\mathbf{x} := \mathbf{x} + \mathbf{x}_{n_s-1} + \mathbf{x}_{n_s}$	$2n$

are $\frac{n}{3}, \frac{n}{6}, \frac{n}{12}, \ldots, 1$ —the degree of parallelism decreases during the computation. However the nonzero patterns of the vectors of consecutive pairs of sets double in size. The consecutive vector sizes are $3, 6, 12, \ldots, \frac{n}{2}, n$. Because each inner product equation orthogonalization consists of dot, square norm and saxpy computations, parallelization is possible as well here. The saxpy computation can be performed in parallel without any interaction. The dot and square norm computations require interaction when they are performed in parallel. The influence of this interaction on performance and speedup highly depends upon the architectural features of the parallel computer such as interconnection network and communication latency. Because the vector size increases, the degree of parallelism increases as well. The total degree of the parallelism is the product of both components; it roughly remains constant during computation and it approaches $O(n)$.

VI. RESULTS AND CONCLUSIONS

The modified algorithm was implemented in FORTRAN. It was run on the Convex C3820 and the CRAY Y-MP4/464. Tridiagonal systems with (a_{i-i}, a_i, a_{i+1}) equal to $(1, -2, 1)$, $(1, -1, 1)$, $(2, 1, 2)$, $(1, 2, -2)$, $(1, 0.01, -1.01)$ and $n \leq 3.2^{16}$ were tested. Note that several of them do not belong to the symmetric, positive definite or diagonal dominant class. The error was compared with the error of Gaussian elimination with partial pivoting. No significant differences were observed. These measurements indicate a favorable error behavior.

In Table II the total number of operations, the average degree of parallelism and the parallel time complexity are shown for several algorithms. No communication overhead is assumed. The parallel time complexity of Recursive Doubling and Cyclic Reduction equals $O(\log_2 n)$. Wang's partition method has a higher order of the parallel time complexity.

Bondeli's DAC algorithm which is based on the same partioning as Wang's method has a smaller total operation count. Communication overheads depend on the architecture and they affect the parallel properties of the algorithms. The limited processor version of Recursive Doubling ([4]) and Sun's PPH algorithm include communication overheads for hypercube architectures and they are only efficient when the number of processors is not too large. The communication overheads were not investigated for the parallel minimal norm method. Although the algorithm is costly in absolute terms in comparison with other algorithms, the degree of interaction is limited and the communication overheads should be small. Therefore the algorithm should compete well with other methods.

TABLE II
CHARACTERISTICS OF SEVERAL ALGORITHMS

Algorithm	Operation count	Av. DOP	Parallel time complexity
Gaussian Elimination	$8n$	$O(1)$	$O(n)$
Cyclic Reduction	$17n$	$O(n/\log_2 n)$	$O(\log_2 n)$
Recursive Doubling	$17n \log_2 n$	$O(n)$	$O(\log_2 n)$
Parallel Minimal Norm	$56n \log_2 n$	$\approx O(n)$	$\approx O(\log_2 n)$
Wang's method $(p = \sqrt{\frac{n}{2}})$	$21n - 12p - 8\frac{n}{p}$	$O(\sqrt{n})$	$O(\sqrt{n})$

All existing parallel algorithms impose restrictions on the kind of linear systems to be solved, such as symmetry, positive definiteness and/or diagonal dominance to guarantee numerical stability. For general tridiagonal linear systems pivoting is required, which destroys the parallel properties of the algorithms. The experiments indicate that no such requirements are needed for the parallel minimal norm method. This robustness will be advantageous in a parallel environment. Further research has to be done, especially implementations on local memory architectures with a large number of processors (MPPs) will be of interest.

ACKNOWLEDGMENTS

The authors would like to thank the Stichting Nationale Computer Faciliteiten for the grant that was awarded to perform experiments on the CRAY Y-MP4/464.

REFERENCES

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974, pp. 60-67.

[2] S. Bondeli, "Divide and conquer: A parallel algorithm for the solution of a tridiagonal linear system of equations," *Parallel Computing*, vol. 17, no. 4-5, pp. 419-434, July 1991.

[3] E. Dekker and L. Dekker, "Parallel minimal norm method for direct solving of linear algebraic equations," *J. Systems Analysis and Modeling Simulation*, vol. 6, no. 9, pp. 643-657, Sept. 1989.

[4] Ö. Eğecioğlu, C.K. Koc, and A.J. Laub, "A recursive doubling algorithm for solution of tridiagonal systems on hypercube multiprocessors," *J. Comp. Appl. Math.*, vol. 27, pp. 95-108, 1989.

[5] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Baltimore: Johns Hopkins Univ. Press, 1983, pp. 92-98.

[6] Intel Corporation. *Paragon XP/S Product Overview*. Beaverton, Ore.: Intel Corporation, 1991.

[7] C. Kamath and A. Sameh, "A projection method for solving nonsymmetric linear systems on multiprocessors," *Parallel Computing*, vol. 9, no. 3, pp. 291-312, Feb. 1989.

[8] J.J. Lambiotte and R.G. Voigt, "The solution of tridiagonal linear systems on the CDC STAR-100 computer," *ACM Trans. Math. Soft.*, vol. 1, no. 4, pp. 308-329, Dec. 1975.

[9] J.M. Ortega and R.G. Voigt, "Solution of partial differential equations on vector and parallel computers," *SIAM Rev.*, vol. 27, no. 2, pp. 149-240, June 1985.

[10] H.S. Stone, "An efficient parallel algorithm for the solution of a tridiagonal linear system of equations," *J. ACM*, vol. 20, no. 1, pp. 87-95, Jan. 1973.

[11] X.H. Sun, H. Zhang, and L.M. Ni, "Efficient tridiagonal solvers on multicomputers," *IEEE Trans. Computers*, vol. 41, no. 3, pp. 286-296, Mar. 1992.

[12] Thinking Machines Corporation. *The Connection Machine CM-5 Technical Summary*. Cambridge, Mass.: Thinking Machines Corporation, 1992.

[13] H.H. Wang, "A parallel method for tridiagonal equations," *ACM Trans. Math. Soft.*, vol. 7, no. 2, pp. 170-183, June 1981.