COMPUTING THE CHOLESKY FACTORIZATION USING A SYSTOLIC ARCHITECTURE

Richard P. Brent*
Franklin T. Luk**

TR 82-521
September 1982

Department of Computer Science
Upson Hall
Cornell University
Ithaca, New York 14853

*Department of Computer Science, Australian National University, Canberra, A.C.T. 2600, Australia

**Supported in part by U.S. Army Research Office under grant DAAG 29-79-C0124 and in part by the Mathematical Sciences Research Centre and the Centre for Mathematical Analysis at the Australian National University.

Abstract

This note concerns the computation of the Cholesky factorization of a symmetric and positive definite matrix on a systolic array. We use the special properties of the matrix to simplify the algorithm and the corresponding architecture given by Kung and Leiserson.

Key Words and Phrases

Cholesky factorization, symmetric and positive definite matrix, band matrix, parallel algorithms, systolic arrays, VLSI.

1. Introduction

In [3] Kung and Leiserson present a systolic architecture for computing the LU-factorization of a square matrix by Gaussian elimination without pivoting. They remark that their architecture is applicable to matrices that are symmetric and positive definite. In this note we show how one may use these special matrix properties to simplify their presentation. We shall discuss both the Cholesky factorization and its square-root-free variant.

The QR factorization of matrices which are not necessarily positive definite may also be performed on a systolic array, as shown in [1, 2].

2. Architectures

The Cholesky factorization of a symmetric and positive definite matrix A, viz. $A = LL^T$, may be evaluated according to the following recurrences. We access only the lower triangular elements of A and so $i \geq j$.

$$a_{ij}^{(1)} = a_{ij} ,$$

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \ell_{ik}\, \ell_{jk} , \quad \text{for } k = 1,2,\ldots,j-1,$$

$$\ell_{ij} = \begin{cases} \sqrt{a_{ij}^{(j)}} & \text{if } i = j , \\ a_{ij}^{(j)}\, \ell_{jj}^{-1} & \text{if } i > j . \end{cases}$$

We use the same idea as [3] to pipeline these recurrences on a hex-connected processor array. It is assumed that A is a band matrix. We present a global view of this pipelined computation in Figure 2 for a heptadiagonal matrix. The processor array is constructed as follows. The hexagonal processors below the upper boundary are the standard type inner product step processors (cf. Figure 1(a) and [3]). The processor at the top, denoted by a half-circle, computes the square root of its input and passes (i) the result northwards and (ii) the reciprocal of the result in the southwest direction (cf. Figure 1(c)). The other processors on the upper boundary are again inner product step processors, but they have been rotated clockwise by 120 degrees. The operations performed by the pentagonal processors on the right side boundary are depicted in Figure 1(b). All the remarks made in [3] are applicable to this architecture, but the number of processors required is almost halved because we take advantage of symmetry. The bottleneck of this array is the top processor which computes a square root and a reciprocal.

(a)

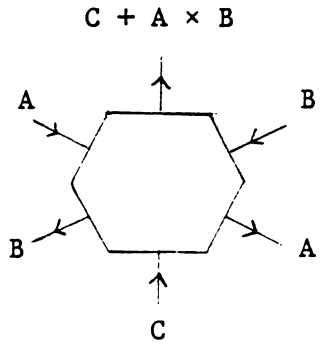
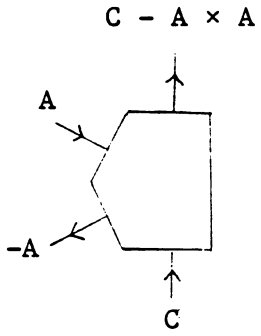
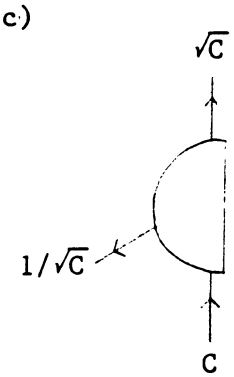


Figure 1. Operations of the processors.

(b)



(c)



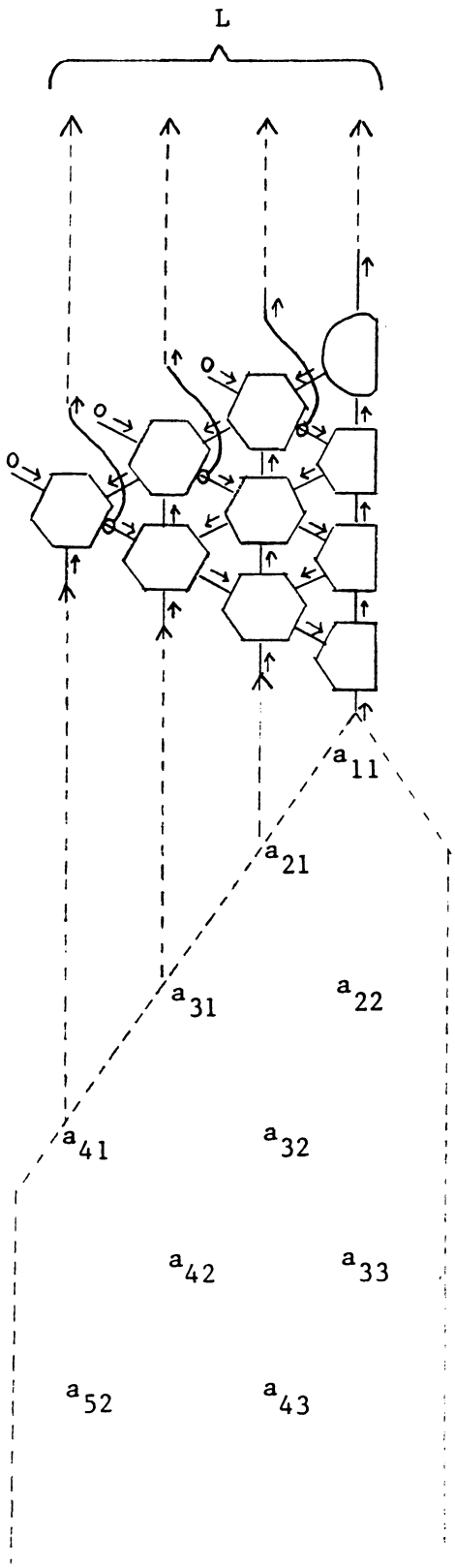


Figure 2. A processor array for pipelining the Cholesky factorization of a heptadiagonal matrix.

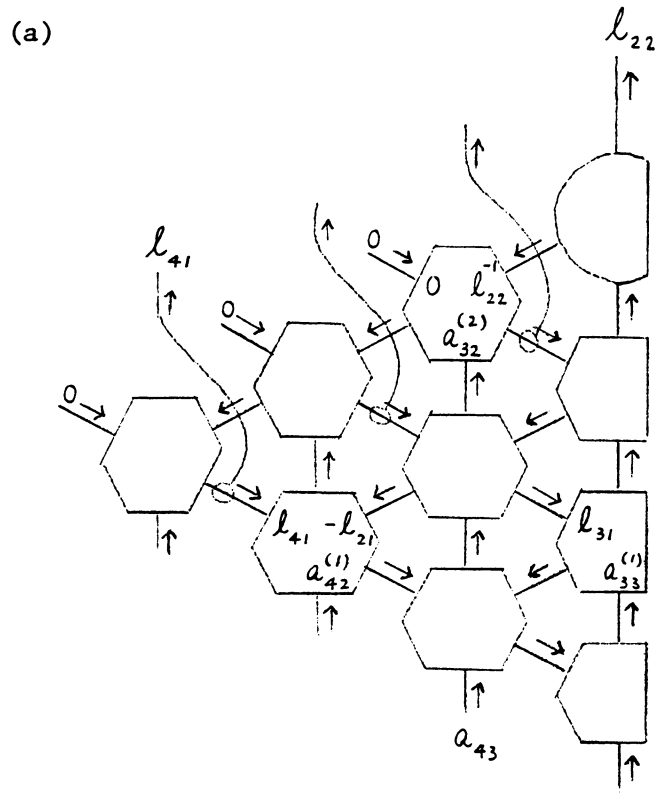
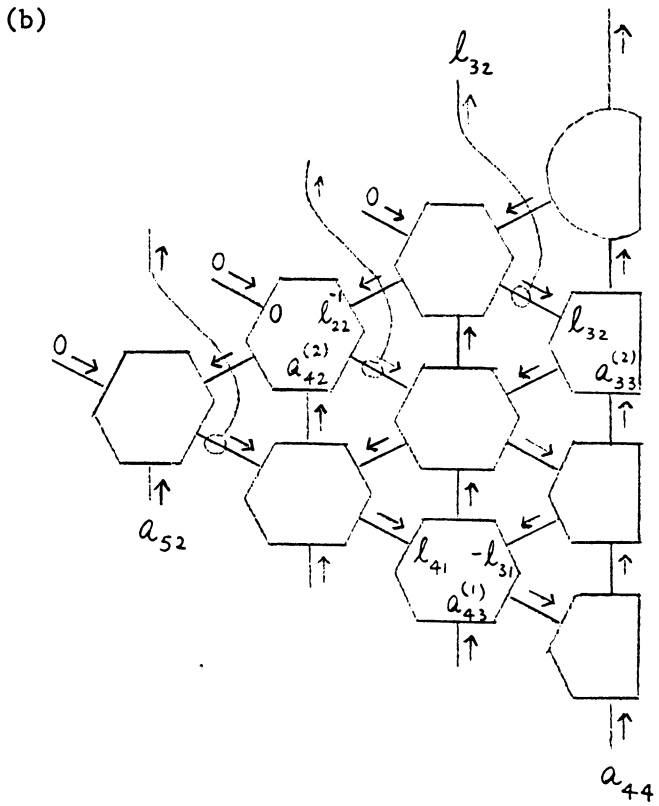


Figure 3. Four steps during the Cholesky factorization shown in Figure 2.



(c)

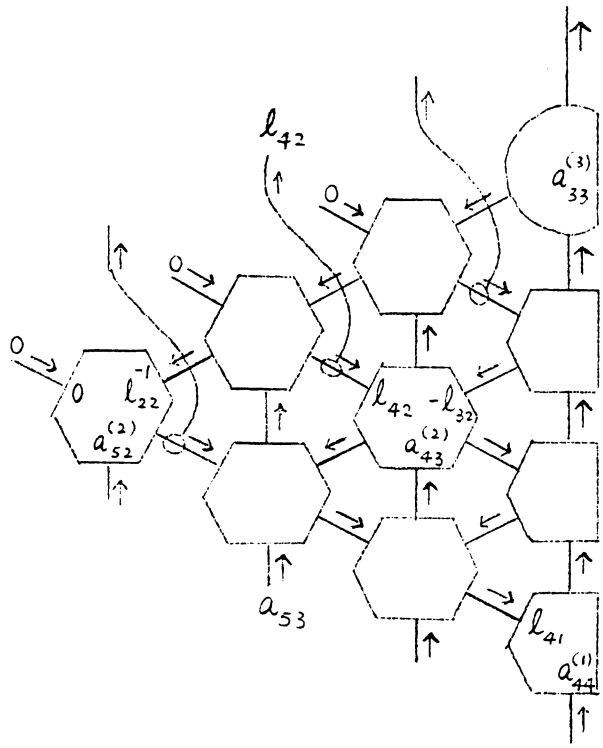
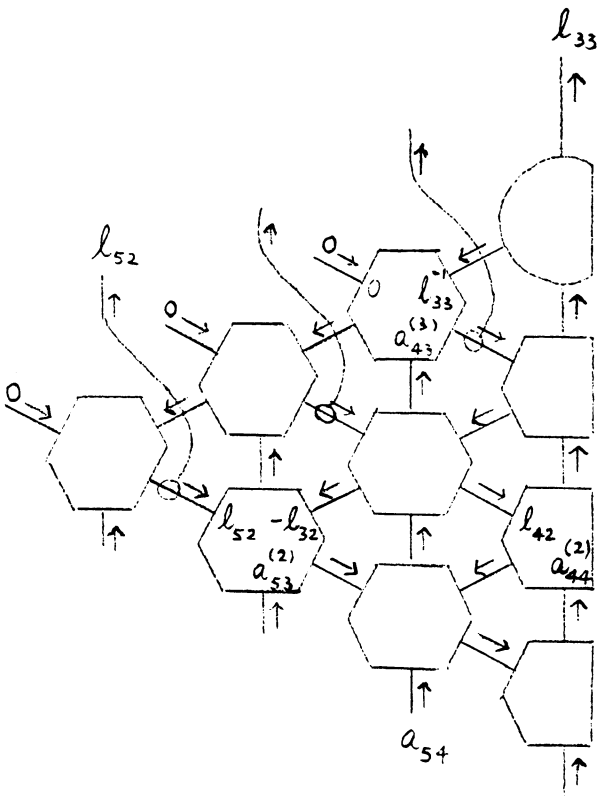


Figure 3 (contd)

(d)



It is therefore worthwhile to avoid the square roots. We give here the recurrences (where $i \geq j$) for computing the LDL^T-factorization of A :

$$a_{ij}^{(1)} = a_{ij} \ ,$$

$$\text{For } k = 1,2,\ldots j-1, \quad a_{ij}^{(k+1)} = \begin{cases} a_{ij}^{(k)} - a_{ik}^{(k)} \, \ell_{jk} & \text{if } i > j \ , \\ a_{ii}^{(k)} - (a_{ik}^{(k)})^2 \, d_k^{-1} & \text{if } i = j \ , \end{cases}$$

$$d_j = a_{jj}^{(j)} \ ,$$

$$\ell_{ij} = a_{ij}^{(j)} \, d_j^{-1} \ .$$

A corresponding systolic architecture can be constructed. The principal ideas are illustrated in Figures 4-6. The price we pay is a slightly higher communication requirement between processors along the right boundary (compare Figures 2 and 5).

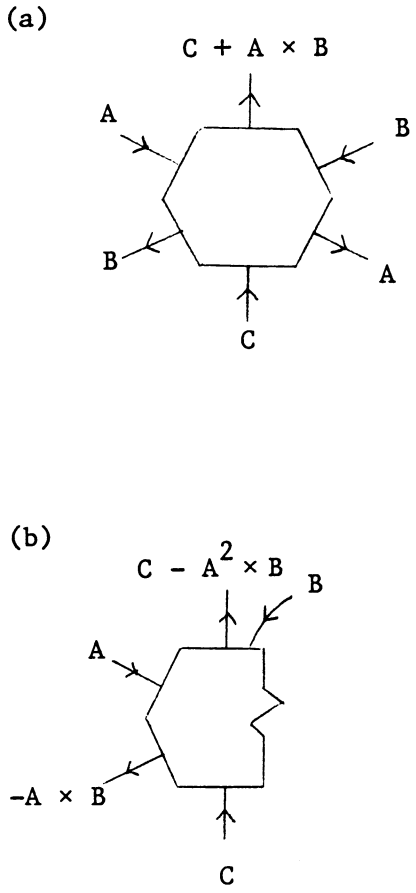


Figure 4. Operations of the processors.

(d)

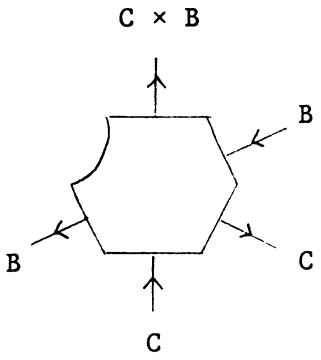
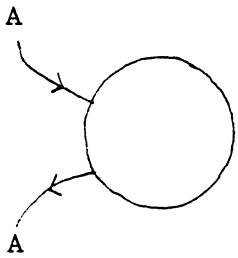


Figure 4 (contd)

(e)



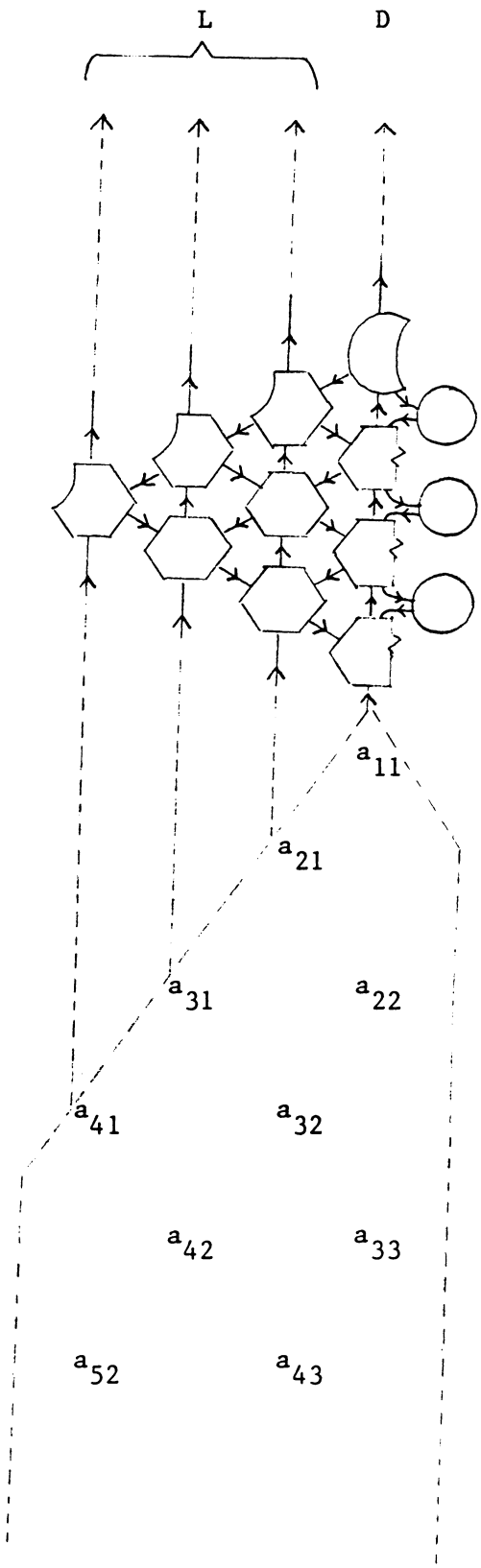


Figure 5. A processor array for pipelining the LDL^T-factorization of a heptadiagonal matrix.

(a)

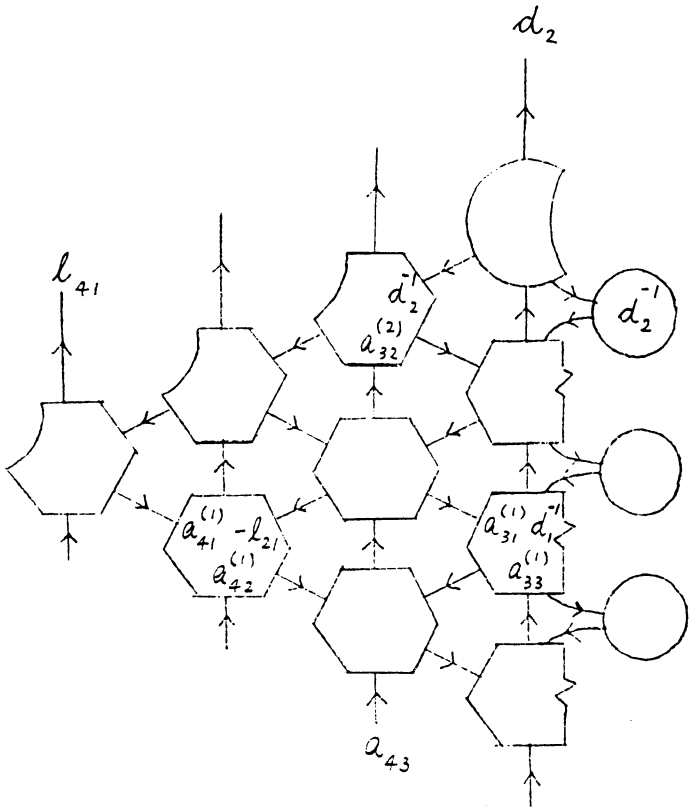
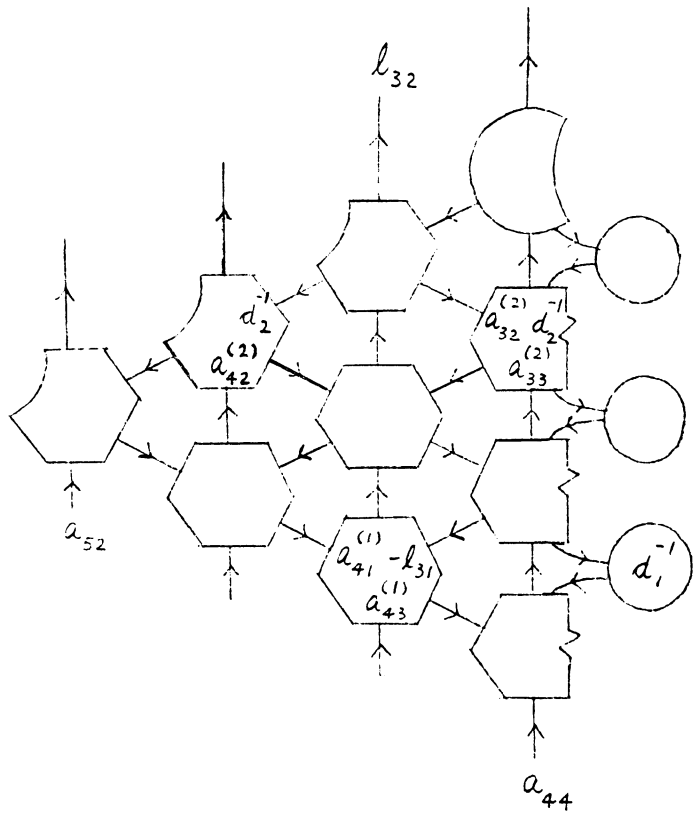


Figure 6. Four steps during the LDL^T-factorization shown in Figure 5.

(b)



(c)

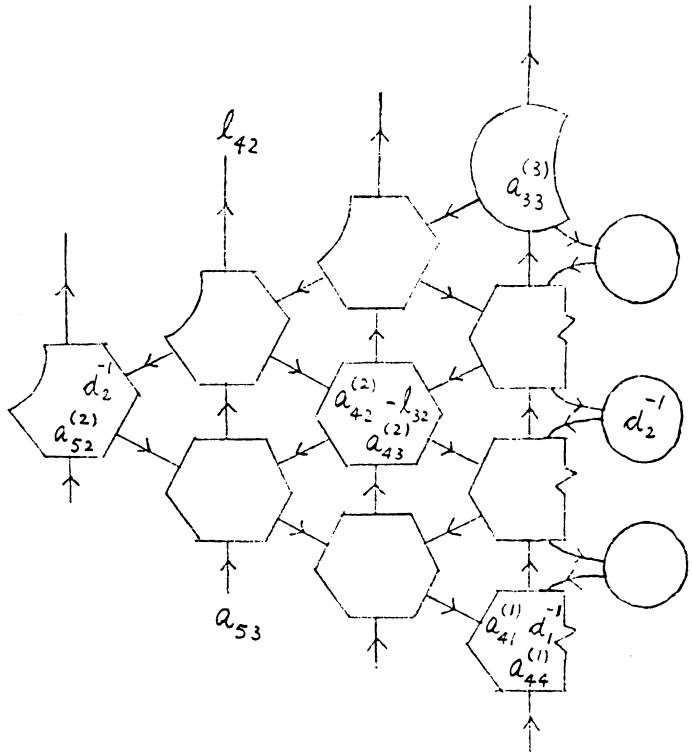
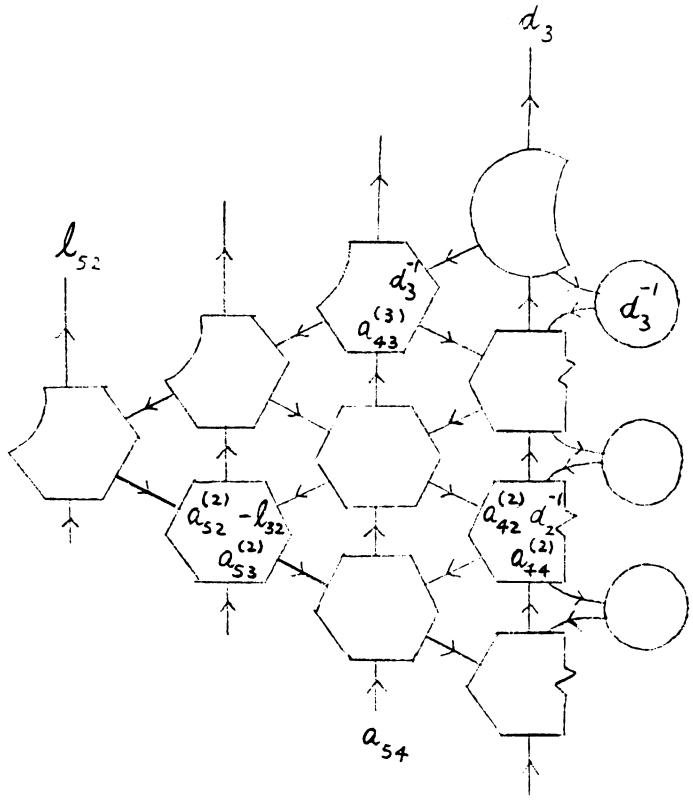


Figure 6 (contd)

(b)



References:

1 A. Bojanczyk, R.P. Brent and H.T. Kung, Numerically stable solution of dense systems of linear equations using mesh-connected processors, to appear in SIAM J. Sci. and Stat. Computing. Also available as Report TR-CS-81-01, Dept. of Computer Science, ANU, May 1981.

2. W.M. Gentleman and H.T. Kung, Matrix triangularization by systolic arrays, Proc. SPIE Symposium, Vol. 298, Real-Time Signal Processing IV, Society of Photo-optical Instrumentation Engineers, August 1981.

3. H.T. Kung and C.E. Leiserson, Algorithms for VLSI processor arrays, in Introduction to VLSI Systems (by C. Mead and L. Conway), Addison-Wesley, Reading, Massachusetts, 1980, pp. 271-292.