Computers & Structures Vol. 17, No. 4, pp. 621–622, 1983
Printed in Great Britain.

0045-7949/83/100621–02$03.00/0

TECHNICAL NOTE

THE OPTIMAL ORDERING OF TREE NETWORKS†

G. C. EVERSTINE and E. H. CUTHILL
David Taylor Naval Ship R & D Center, Bethesda, MD 20084, U.S.A.

(*Received* 20 *July* 1982; *received for publication* 17 *September* 1982)

INTRODUCTION

A recent paper by Martin and Alarcon[1] presented an algorithm for re-ordering the nodes of a tree structure to effect a more efficient solution of the linear algebraic equations which arise in applications such as piping analysis. The algorithm was claimed to be "optimal", although criteria for assessing optimality were not given. The purposes of this note are to discuss several concepts of optimal ordering (both in general and, in particular, for trees) and to relate the Martin-Alarcon algorithm to some other algorithms which preceded it.

OPTIMAL ORDERINGS AND THE MARTIN-ALARCON ALGORITHM

Since the nodes of a network are re-ordered prior to solving a set of algebraic equations by Gaussian elimination or by performing a triangular factorization, an ordering is considered optimal if it minimizes either computer storage or solution time (or some combination of these two quantities) for a particular implementation of a solution algorithm.

From a storage point of view, the optimal ordering could be that which minimizes fill[2]. (If the symmetric matrix A is to be factored, the "fill" is defined as the number of nonzero matrix elements in either factor of A in positions in which A has zero elements.) However, it does little good to minimize fill unless the factorization routine to be used can exploit such a sequence by operating only on nonzeros in the matrix. Thus, as a practical matter, the optimal ordering is that which minimizes storage or solution time for the particular factorization routine being used.

Most application programs involving the solution of large sparse systems of equations are based, not on minimum storage requirements (the number of anticipated nonzeros in the factors of a matrix), but on either diagonal band storage or profile or wavefront storage. In diagonal band storage, all elements on the matrix diagonal and in as many bands as are needed parallel to that diagonal are stored. In profile/wavefront storage, one stores, for each row, all elements from the first nonzero to the diagonal, or the corresponding elements in the transpose of the matrix. Thus, depending on the factorization routine used, one may want to order the nodes of a network to minimize fill, bandwidth, average bandwidth (or profile), maximum wavefront, or root-mean-square wavefront[3, 4].

A tree network, the type of network considered by Martin and Alarcon[1], is particularly nice since a sequence resulting in zero fill can be obtained merely by sequencing the nodes such that, at each step of the elimination process, a "leaf" of the tree is chosen next[2, 3].

Martin and Alarcon do not state their criteria for optimality but it is evidently not minimum fill, since the only parameters they mention are maximum semi-bandwidth and average semi-bandwidth. However, their algorithm is not optimal under these criteria either, because the Martin-Alarcon algorithm orders the nodes using a scheme closely related to the (forward) Cuthill-McKee scheme[5], which is known to be not optimal (in any common sense) for trees[3]. Both the reverse Cuthill-McKee ordering and similarly the reverse Martin–Alarcon ordering result in the optimal zero fill for trees[3] but not optimal bandwidth or profile in general.

There are two main differences between the Martin–Alarcon (MA) algorithm and the forward Cuthill-McKee (CM) scheme. First, MA always uses the first node of the first branch as the new starting node. CM selects instead as potential starting nodes those nodes of low connectivity which can root a graph of minimal width. The practical effect of this difference is that CM selects the best sequence from several considered.

Given a starting node, both MA and CM order nodes by levels, where the first level is the single starting node. Each subsequent level consists of all unnumbered nodes adjacent to the nodes of the previous level. The second difference then between MA and CM is that CM, unlike MA, sequences nodes within a given level by choosing first those nodes adjacent to *lower* numbered nodes of the previous level in order of increasing connectivity. The practical effect of this difference is to reduce the likelihood of the lowest numbered node of one level being adjacent to the highest numbered node of the next level. (Precisely this situation occurs in Martin and Alarcon's sample network[1] with new nodes 20 and 27. The interchange, for example, of 27 and 26 in that network would reduce the nodal semi-bandwidth for the network from 8 to 7.)

Because the MA sequence is so closely related to the CM sequence, the average bandwidth (or profile) obtained could probably be reduced by reversing the ordering (i.e. using the sequence $N, N-1, \ldots, 1$ rather than $1, 2, \ldots, N$). It was observed by George[6] and proved by Liu and Sherman[7] for the CM sequence that such a reversal, which preserves matrix bandwidth, often reduces the profile and can never increase it.

The average bandwidth, as used here[3, 4] and by most other writers, has a non-standard definition in Martin and Alarcon's work[1]. Their definition, which must be deduced from their Fortran listing, is that the average bandwidth is the average of the maximum differences, *in absolute value*, of all connected node pair labels. Thus, the larger of the "left bandwidth" and "right bandwidth" is used for each matrix row. The more conventional definition considers, for each matrix row, only the number of matrix columns from the first nonzero in the row to the diagonal, inclusive. This latter definition, unlike the former, has practical value since the sum of all the "row semi-bandwidths" (the matrix profile) is a measure of the storage requirements for a profile (or variable band) factorization algorithm.

EXAMPLE

A concrete example can be used to demonstrate that the Martin–Alarcon (MA) algorithm[1] is not optimal using either maximum or average matrix semi-bandwidth as the criterion. We consider Martin and Alarcon's example[1], a tree network (Fig. 2 in their reference) having 32 nodes and nine branches. We resequenced this network using both the reverse Cuthill-McKee (RCM)[5] and the Gibbs-Poole-Stockmeyer (GPS)[8] algorithms. RCM and GPS are general purpose resequencing algorithms known to be both effective and fast[4, 9]. The specific implementations of RCM and GPS used for this test were those

†Dedicated to the memory of James M. McKee (1942–1983).

Table 1. Comparison of Martin–Alarcon (MA) algorithm with reverse Martin–Alarcon (RMA), reverse Cuthill–McKee (RCM), and Gibbs–Poole–Stockmeyer (GPS) algorithms for tree network of Fig. 2 in Ref. [1]

	Maximum Nodal Semi-Bandwidth	Average Nodal Semi-Bandwidth
Original Ordering	26	4.97
After MA [1]	8	3.84
After RMA	8	3.09
After RCM [5]	7	2.84
After GPS [8]	5	2.75

appearing in the BANDIT computer program (version 9)[10–12], a widely-used re-ordering preprocessor to the NASTRAN structural analysis computer program. For RCM, which selects a new sequence from several considered, the resequencing was performed twice, once using bandwidth as the reduction criterion and once using profile (average bandwidth). (For this network, RCM selected the same new sequence both times.) The results of the test are shown in Table 1, where we have used the conventional definition[3, 4], rather than Martin and Alarcon's definition, for average semi-bandwidth. We also show in the table the further reduction in average semi-bandwidth which can be achieved for this network by reversing the MA sequence. For this network, GPS clearly obtains the best sequence, i.e. the smallest values of bandwidth and average bandwidth.

REFERENCES

1. A. F. Martin and E. Alarcon, A contribution to the optimal numbering of tree structures. *Comput. Structures* **15**(3), 283–290 (1982).
2. S. Parter, The use of linear graphs in Gauss elimination. *SIAM Rev.* **3**(2), 119–130 (1961).
3. E. Cuthill, Several strategies for reducing the bandwidth of matrices, *Sparse Matrices and Their Applications* (Edited by D. J. Rose and W. A. Willoughby), pp. 157–166. Plenum Press, New York (1972).
4. G. C. Everstine, A comparison of three resequencing algorithms for the reduction of matrix profile and wavefront. *Int. J. Numer. Meth. Engng* **14**(6), 837–853 (1979).
5. E. Cuthill and J. M. McKee, Reducing the bandwidth of sparse symmetric matrices. *Proc. 24th Nat. Conf., Assn. for Computing Machinery*, pp. 157–172. ACM Pub. P69, New York (1969).
6. A. George, Computer implementation of the finite element method, Ph.D. dissertation, *Tech. Rep. STAN-CS*-71-208, Computer Sci. Dept., Stanford Univ., Stanford, California (1971).
7. W.-H. Liu and A. H. Sherman, Comparative analysis of the Cuthill–McKee and the reverse Cuthill–McKee ordering algorithms for sparse matrices. *SIAM J. Numer. Anal.* **13**(2), 198–213 (1976).
8. N. E. Gibbs, W. G. Poole, Jr., and P. K. Stockmeyer, An algorithm for reducing the bandwidth and profile of a sparse matrix. *SIAM J. Numer. Anal.* **13**(2), 236–250 (1976).
9. N. E. Gibbs, W. G. Poole, Jr., and P. K. Stockmeyer, A comparison of several bandwidth and profile reduction algorithms. *ACM Trans. Math. Software* **2**(4), 322–330 (1976).
10. G. C. Everstine, The BANDIT computer program for the reduction of matrix bandwidth for NASTRAN, *Rep.* 3827, Naval Ship Research and Development Center, Bethesda, Maryland (1972).
11. G. C. Everstine, Recent improvements to BANDIT, *NASTRAN: Users' Experiences*, NASA TM X-3278, pp. 511–521. National Aeronautics and Space Admin., Washington, DC (1975).
12. G. C. Everstine, BANDIT user's guide, *Rep. TM*-184-77-03, David W. Taylor Naval Ship Research and Development Center, Bethesda, Maryland (1977).