ON METHODS FOR ORDERING SPARSE MATRICES IN CIRCUIT SIMULATION

Gunther Reißig[*]

Massachusetts Institute of Technology, Room 66-363
77 Massachusetts Avenue
Cambridge, MA 02139, U.S.A.

ABSTRACT

Recently proposed methods for ordering sparse symmetric matrices are discussed and their performance is compared with that of the Minimum Degree and the Minimum Local Fill algorithms. It is shown that these methods applied to symmetrized modified nodal analysis matrices yield orderings significantly better than those obtained from the Minimum Degree and Minimum Local Fill algorithms, in some cases at virtually no extra computational cost.

1. INTRODUCTION

Simulating an electrical circuit requires the solution of solve numerous linear equations of the form

$$Ax = b, \tag{1}$$

where A is a real or complex $n \times n$ matrix [1–3]. Although A is unsymmetric and indefinite, (1) is solved directly without pivoting for numerical accuracy. In fact, it is assumed that one can, from a numerical point of view, solve

$$(PAP^T)y = Pb \tag{2}$$

by forward elimination and back substitution for any $n \times n$ permutation matrix P and then set $x = P^T y$. That is, the diagonal entries may be chosen as pivots, in any order [3, 4].

Typically, A is very large and extremely sparse, i.e., only few of its entries are nonzero. Unfortunately, as the elimination phase progresses, new nonzeros are introduced:

Usually, the coefficient matrix PAP^T is successively overwritten by the matrix $L + U - \mathrm{id}_n$, where L and U are factors of PAP^T,

$$LU = PAP^T, \tag{3}$$

L being unit lower triangular and U being upper triangular, and id_n is the $n \times n$ identity matrix [3, 5, 6]. In general, the matrix $L + U - \mathrm{id}_n$ will have nonzeros at positions where the coefficient matrix PAP^T has zeros. Those newly created nonzeros are called *fill-ins*, and their total is called *fill* [7, 8].

The coefficient matrices of all linear equations to solve usually have the same zero-nonzero pattern. Hence, the same permutation matrix P is used throughout the simulation [3,5]. Since the amount of fill created, and hence, the computational complexity of solving equation (1), heavily depend on P, a deliberate choice of P is extremely important for the overall performance of circuit simulation software.

Various heuristics have been proposed, which aim at acceptable low, rather than minimum, fill [7], as minimizing the fill appears to be harder than solving the linear equations at hand without taking advantage of their sparsity [9].

This paper focuses on so-called *local* algorithms, which mimic Gaussian elimination based on the zero-nonzero pattern of the coefficient matrix alone, under the assumption that nonzeros do not accidentially cancel out. These methods determine a pivoting order by choosing, in each elimination step, a pivot that minimizes some scoring function [7].

MARKOWITZ was the first to propose heuristics that aim at the creation of low fill in Gaussian elimination [10]. In the *Minimum Local Fill (MF)* algorithm, the score of a diagonal entry is the number of fill-ins that would be created in the next elimination step if that entry was chosen as the next pivot. Although mentioning that strategy, MARKOWITZ recommended using the following scoring function instead, as its values are easier to obtain [10]:

Let c_i and r_i be the number of off-diagonal nonzeros in the ith column and row, respectively, of the zero-nonzero pattern that evolved from the preceding eliminations. Obviously, the product $c_i r_i$, which is called the *Markowitz product*, is an upper bound on the number of fill-ins introduced when the ith variable in the current scheme is eliminated next using the ith equation. To chose a pivot with minimum Markowitz product has become known as *Markowitz' algorithm*.

When applied to structurally symmetric matrices ($A_{i,j} \neq 0$ iff $A_{j,i} \neq 0$), Markowitz' algorithm is also called *Minimum Degree (MD)* algorithm [8] for reasons that become clear in Section 2.

A drawback of the MF algorithm is its immense computational cost, which may be two orders of magnitude higher than that of a version of the MD algorithm [11]. On the other hand, it has been found especially in circuit simulation that the MF algorithm saves at most 5% in factorization operations compared with Markowitz' algorithm [3].

Only recently, the MF algorithm as well as newly proposed variants of both the MF and MD algorithms have been found to yield significantly better orderings than the MD algorithm on certain test suites of symmetric matrices [11–15], and the running times of some of those algorithms are just in the order of those of the MD algorithm [11, 12, 14, 15].

Despite of the fact that the savings in fill of those variants heavily depend on the field of application and that these savings increase with increasing problem size [12], extensive tests and comparisons of those variants on matrices derived from circuits of a size representing today's simulation tasks are still missing.

E-Mail: gr@ieee.org. Work supported by Deutsche Forschungsgemeinschaft. Part of the results of this paper have been obtained while the author was with Infineon Technologies, München, Germany.

0-7803-6685-9/01/$10.00

The purpose of this paper is to report on such tests and to show that those new heuristics can be successfully applied to the modified nodal equations.

Our review of PARTER's interpretation of Gaussian elimination in terms of graphs [16] and the data structure underlying local symmetric ordering methods in Section 2 provides the basis for the definition of the recently proposed ordering heuristics in Section 3.

In Section 4, we compare the performance of the MD and MF algorithms and the algorithms from Section 3 on a test set of 15 matrices which have been obtained from symmetrizing Jacobians of modified nodal equations extracted from the circuit simulator TITAN [2] of Infineon Technologies in the following way [17]:

Initially, diagonal entries with Markowitz product zero are chosen as pivots, as many as possible. Since the corresponding elimination steps do not create any fill, the zero-nonzero pattern obtained after those steps is that of a submatrix $\widetilde{A}$ obtained from A by removing the pivot rows and columns. Now, a symmetric ordering method is applied to $|\widetilde{A}| + |\widetilde{A}|^T$, thereby completing the pivoting order for A. That way, much of the unsymmetry of A is removed by the initial steps of Markowitz' algorithm so that $\widetilde{A}$ will be symmetric or nearly so.

We obtain orderings that are, in terms of the resulting number of factorization operations, significantly better than those obtained from the MD and MF algorithms, in some cases at virtually no extra computational cost.

2. DATA STRUCTURE OF LOCAL SYMMETRIC ORDERING METHODS

Throughout this section, A is a structurally symmetric $n \times n$ matrix, i.e., $A_{i,j} \neq 0$ iff $A_{j,i} \neq 0$, for all i and j, $1 \leq i, j \leq n$.

2.1. Gaussian elimination in terms of graphs

A *graph* is an ordered pair (V, E) of a finite set V of *vertices* and a set E of *edges*, $E \subseteq \big\{\{v, w\} \subseteq V \mid v \neq w\big\}$.

Two vertices $v, w \in V$ are *adjacent* in the graph G, $G = (V, E)$, if $\{v, w\} \in E$. For $w \in V$ and $W \subseteq V$, $\mathrm{adj}_G(w)$ and $\mathrm{adj}_G(W)$ denote the *adjacent set* of w and W, respectively, i.e.,

$$\mathrm{adj}_G(w) = \big\{u \in V \mid \{u, w\} \in E\big\},$$
$$\mathrm{adj}_G(W) = \bigcup_{u \in W} \mathrm{adj}_G(u) \setminus W.$$

For $X \in V \cup \mathcal{P}(V)$, we denote the *degree of X in G* by $\deg_G(X)$,

$$\deg_G(X) = |\,\mathrm{adj}_G(X)|,$$

where $|\cdot|$ denotes cardinality, and $\mathcal{P}(V)$ is the *power set* of V.

A vertex $v \in V$ and an edge $e \in E$ are *incident* if $v \in e$.

The *graph of* A, denoted $\mathcal{G}(A)$, is the graph (V, E) defined by

$$V = \{1, \ldots, n\},$$
$$E = \big\{\{i, j\} \mid A_{i,j} \neq 0, i \neq j\big\}.$$

The *elimination graph* G_v is obtained by *eliminating* $v \in V$ *from* $G = (V, E)$, i.e., by removing v and its incident edges from G, and connecting all vertices previously adjacent to v [16], thereby creating a set $\mathrm{fill}_G(v)$ of new edges or *fill-ins*.

Choosing pivots down the diagonal in PAP^T for some $n \times n$ permutation matrix P corresponds to selecting vertices of $\mathcal{G}(A)$ in the order $\pi(1), \pi(2), \ldots, \pi(n)$ for some bijection $\pi : V \to V$, which we call the *pivoting order*. Thus, local ordering algorithms are equivalent to the selection of vertices as follows:

Input:	Graph $G = (V, E)$, scoring function s.
Step 1:	$S := \emptyset$.
Step 2:	Pick $v \in V \setminus S$ with $s(v, G) = \min_{w \in V \setminus S} s(w, G)$.
Step 3:	$S := S \cup \{v\}$, $\pi(|S|) := v$, $G := G_v$.
Step 4:	If $S \neq V$, goto Step 2.
Output:	Pivoting order π.

In particular, if the input graph G equals $\mathcal{G}(A)$, the above algorithm is the MD and the MF algorithm for $s(v, G) = \deg_G(v)$ and $s(v, G) = |\,\mathrm{fill}_G(v)|$, respectively.

2.2. Clique representations of elimination graphs

A *clique* is a graph in which any two distinct vertices are adjacent. The vertex set of a clique will also be called a clique.

A set $\mathcal{C} \subseteq \mathcal{P}(V)$ is called a *clique representation* of the graph $G = (V, E)$ if $E = \big\{\{v, w\} \subseteq C \mid C \in \mathcal{C}, v \neq w\big\}$. In other words, clique representations of $\mathcal{G}(A)$ correspond to coverings of the nonzeros of A by full symmetric minors. Note also that E is a (trivial) clique representation of G.

If $\mathcal{C}$ is a clique representation of G and $v \in V$, then the set $\mathcal{C}'$,

$$\mathcal{C}' = \{C \in \mathcal{C} \mid v \notin C\} \cup \{\mathrm{adj}_G(v)\} \tag{4}$$

is a clique representation of G_v.

2.3. Indistinguishable vertices

The vertices $v, w \in V$ are *indistinguishable* [8], $v \underset{G}{\sim} w$, if

$$\mathrm{adj}_G(v) \cup \{v\} = \mathrm{adj}_G(w) \cup \{w\}. \tag{5}$$

If $v \underset{G}{\sim} w$, then $\deg_G(v) = \deg_G(w)$ and $\mathrm{fill}_G(v) = \mathrm{fill}_G(w)$. Moreover, for any reasonable scoring function, it should suffice to determine the score of one of v and w only. Hence, we may maintain the quotient graph $G/\underset{G}{\sim}$ rather than G itself, where $G/\underset{G}{\sim} = (V/\underset{G}{\sim}, E')$ and

$$E' = \big\{\{[v]_{\underset{G}{\sim}}, [w]_{\underset{G}{\sim}}\} \mid \{v, w\} \in E, [v]_{\underset{G}{\sim}} \neq [w]_{\underset{G}{\sim}}\big\}.$$

3. IMPROVED SCORING FUNCTIONS

Let G be the current elimination graph, $G = (V, E)$, $\sim$ be the equivalence relation of indistinguishable vertices, and let $\mathcal{C}$ be a clique representation of $G/\sim$. For simplicity, we denote the class $[v]_\sim$ by $[v]$.

For each vertex $[v]$ of $G/\sim$, let $\kappa_{[v]}$ be the list of cliques containing $[v]$. Let $c_{[v]}$ be the number of cliques in that list that have been created by eliminating vertices and assume that those created cliques are located at the beginning of the list.

We define scoring functions in terms of $G/\sim$. For $W \subseteq V/\sim$, we define

$$\|W\| = \sum_{[w] \in W} |[w]|.$$

3.1. Bounds on the local fill

The scoring function of the MD algorithm represents an upper bound on the number of fill-ins introduced by eliminating a vertex from G, since

$$\vartheta : x \mapsto (x^2 - x)/2$$

is monotonic on the set of nonnegative integers and $|\operatorname{fill}_G(v)| \leq \vartheta(\deg_G(v))$ for all $v \in V$.

That bound may be improved since some of the $\vartheta(\deg_G(v))$ potential fill-in edges in G_v are edges of G that are easy to identify:

First, if v is indistinguishable from w in G, $v \neq w$, then eliminating v from G does not create any new edges adjacent to w in G_v. Hence, the *external degree* $\deg_G([v])$ *of* v represents an upper bound on $|\operatorname{fill}_G(v)|$,

$$|\operatorname{fill}_G(v)| \leq \vartheta(\deg_G([v])). \tag{6}$$

Taking the external degree as scoring function usually produces less overall fill than taking the degree [8].

Further, if $C \in \mathcal{C}$ is a clique, then the elimination of $[v]$ from G cannot create any new edge $\{[u],[w]\} \subseteq C$. Even if only those cliques are considered that contain $[v]$, the bound in (6) may be improved in several ways:

The *Approximate Minimum Local Fill (AMF)* algorithm of ROTHBERG and EISENSTAT uses the upper bound s_{AMF_0},

$$s_{AMF_0}([v]) = \vartheta(\deg_G([v])) - \begin{cases} 0 & \text{if } c_{[v]} = 0 \\ \vartheta(\|\kappa_{[v]}(1) \setminus \{[v]\}\|) & \text{otherwise} \end{cases}$$

as its scoring function, thereby taking into account the most recently eliminated clique only [15]. We will denote that scoring function that takes in account the largest, rather than the most recently created, clique, by s_{AMF_1}.

NG and RAGHAVAN propose to consider all cliques in $\kappa_{[v]}$ rather than just one [11].

3.2. Looking ahead

While local ordering algorithms usually consider the fill introduced in the next elimination step only, the concept of indistinguishability provides a simple means to look some steps ahead: Since the total number of fill-ins created when all vertices in $[v]$ are eliminated from G immediately upon each other is just $|\operatorname{fill}_G(v)|$, ROTHBERG and EISENSTAT consider dividing fill bounds by $|[v]|$ [15]. Their *(Approximate) Minimum Mean Local Fill ((A)MMF)* heuristics are based on the scoring functions s_{MMF^α} and $s_{AMMF_0^\alpha}$,

$$s_{MMF^\alpha}([v]) = |\operatorname{fill}_G(v)|/|[v]|^\alpha,$$
$$s_{AMMF_0^\alpha}([v]) = s_{AMF_0}([v])/|[v]|^\alpha$$

with $\alpha = 1$. According to [15], a version with $\alpha = 1/2$ "produced slightly better results".

We denote by $s_{AMMF_1^\alpha}$ that scoring function that results from application of the above trick to s_{AMF_1}.

3.3. Further variants

Among the various improvements of the MD and MF algorithms that we do not discuss in this paper are the tie breaking techniques of [12, 14], the *Modified Multiple Minimum Degree (MMMD)* algorithm of [11], and the *correction terms* of [11, 15].

4. COMPUTATIONAL RESULTS

We have implemented the MD and MF algorithms as well as their variants described in Section 3 in the language C as an extension to the ordering methods of the SPOOLES library [18], with several modifications to the original data structures.

In addition to the techniques described in Section 2, we apply further well-known techniques in our implementation, such as *element absorption*, *incomplete score update*, and *multiple elimination*, see [8].

The code was compiled with the cc compiler (Workshop Compilers 4.2 30 Oct 1996 C 4.2) using the options "-fast -fsimple=2 -xtarget=ultra -xarch=v8ultra" under SunOS Release 5.7 and run on one of the CPUs (sparcv9+vis, 400 MHz clock rate, 4 MB cache, 17.4 SPECint95, 25.7 SPECfp95) of a SUN Enterprise E4500 workstation with 6 Gbytes of memory.

We chose the *Multiple Minimum Degree (MMD)* algorithm [8], a version of the MD algorithm based on external degrees, which is probably the most widely used local symmetric ordering method today, as our reference algorithm.

For all other algorithms, we report ratios, i.e., quantities calculated for these algorithms divided by the corresponding quantities for the reference algorithm. We judge algorithms by comparing the geometric mean of the ratios calculated for the set of matrices.

Our primary measure for comparing ordering algorithms is the number $\sum_{i=1}^{n} c_i(1 + r_i)$ of factorization operations determined by the pivoting orders obtained, which represents divisions and multiplications, where c_i and r_i are the number of off-diagonal nonzeros in column and row i of the factors L and U, respectively, of PAP^T, and P is the permutation matrix corresponding to the pivoting order.

It is well known that ordering heuristics are sensitive to permutations of the rows and columns of the input matrices. Therefore, the arithmetic means over the quantities measured for 11 runs with different random orders for inserting the initial scores into the score heap [18] are the basis for subsequent comparisons.

Our test suite of input data consists of 15 symmetric matrices ($\widetilde{A}$ obtained from Jacobians of modified nodal equations as described in section 1). The matrices are listed in Tab. 1 under the names "bag" through "X" together with their characteristics and the performance of the MMD algorithm on them[1].

Computational results for some of the algorithms discussed earlier in this paper are presented in Tab. 2.

We first observe that the improvements of the local fill bound beyond the one represented by the external degree always lead to better pivoting orders. We also see that those algorithms that rely on the exact local fill yield significantly better pivoting orders than the others.

By comparing the results for the AMF_0 and AMMF_0^1 as well as those for the MMF^α and MF algorithms, we see that the looking ahead technique from Paragraph 3.2 is very useful.

The running times for those heuristics that are based on bounds on the local fill appear to be roughly the same, and those heuristics that are based on exact local fill counts are more than one order of magnitude slower.

After all, the $\text{MMF}^{1/2}$ algorithm leads to the fewest number of factorization operations, namely 31% less than the MMD algorithm. The $\text{AMMF}_1^{1/2}$ algorithm, which is the best among those

[1] More information on the structure of the Jacobians will be given in an extended version of this paper to be published elsewhere.

Matrix			MMD		
Name	n	m	o	m'	t/sec.
bag	143854	922936	$3.3 \cdot 10^7$	$2.0 \cdot 10^6$	7.42
cor	123118	659182	$1.7 \cdot 10^6$	$7.8 \cdot 10^5$	5.57
eng	4893	33993	$1.0 \cdot 10^6$	$8.3 \cdot 10^4$	0.16
jac	903	17967	$1.1 \cdot 10^6$	$3.6 \cdot 10^4$	0.05
m8	10464	197374	$3.0 \cdot 10^7$	$7.3 \cdot 10^5$	1.27
m14	15766	348968	$5.6 \cdot 10^7$	$1.3 \cdot 10^6$	2.44
m24	26120	613812	$3.9 \cdot 10^8$	$3.7 \cdot 10^6$	5.01
m40	156035	1789325	$3.4 \cdot 10^9$	$1.9 \cdot 10^7$	25.99
sei	3539	280785	$1.5 \cdot 10^8$	$7.6 \cdot 10^5$	0.51
buc	9751	53427	$8.7 \cdot 10^5$	$1.1 \cdot 10^5$	0.21
gue	86086	390074	$1.4 \cdot 10^7$	$7.9 \cdot 10^5$	2.27
te	59418	253584	$6.9 \cdot 10^5$	$3.4 \cdot 10^5$	1.33
tei	174702	804620	$4.6 \cdot 10^6$	$1.2 \cdot 10^6$	6.38
xch	10560	55852	$8.8 \cdot 10^5$	$1.2 \cdot 10^5$	0.23
X	16063	127887	$3.1 \cdot 10^6$	$2.4 \cdot 10^5$	0.46

Table 1: Test matrices and performance of the MMD algorithm. n and m are the number of rows and nonzeros, resp.. o, m' and t denote the number of factorization operations, the number of nonzeros including the fill, and the CPU time of the MMD algorithm. The values of o and m' are rounded to two decimal digits, those of t to a precision of 10^{-2}sec..

Algorithm	o	m'	t
AMF_0	0.95	0.98	1.3
AMMF_0^1	0.91	0.97	1.5
$\text{AMMF}_1^{1/2}$	0.85	0.95	1.4
MF	0.76	0.92	13
MMF^1	0.74	0.91	15
$\text{MMF}^{1/2}$	0.69	0.90	14

Table 2: Performance of ordering methods relative to the MMD algorithm. o, m' and t denote the number of factorization operations, the number of nonzeros including the fill, and the CPU time of the ordering algorithm. The reported values are geometric means over the quantities divided by those for the MMD algorithm rounded to two decimal digits.

based on bounds, still leads to 15% fewer operations than the MMD algorithm.

5. CONCLUSIONS

We have reviewed recently proposed local symmetric ordering methods, i.e., methods for obtaining pivoting orders for sparse symmetric matrices. It was shown that these methods require up to 31% fewer factorization operations than those obtained from the Minimum Degree algorithm, in some cases at virtually no extra computational cost, when applied to symmetrized Jacobians of modified nodal equations.

The performance of the combination from [17] of Markowitz' algorithm with symmetric methods has yet to be compared to that of unsymmetric methods such as Markowitz' algorithm. The results of such tests will be presented elsewhere.

6. ACKNOWLEDGMENT

I thank C. Ashcraft (Livermore, WA), G. Denk (München), F. Grund (Berlin), T. Klimpel (München), P. Raghavan (Knoxville, TN), and E. Rothberg (Mountain View, CA) for their kind support, valuable hints and comments.

7. REFERENCES

[1] L. O. Chua, C. A. Desoer, and E. S. Kuh, *Linear and Nonlinear Circuits*. McGraw–Hill, 1987.

[2] U. Feldmann, U. A. Wever, Q. Zheng, R. Schultz, and H. Wriedt, "Algorithms for modern circuit simulation," *Archiv für Elektronik und Übertragungstechnik (AEÜ)*, vol. 46, no. 4, pp. 274–285, 1992.

[3] L. W. Nagel, "SPICE 2: A computer program to simulate semiconductor circuits," Tech. Rep. ERL-M520, Univ. of Calif. Berkeley, Electronic Res. Lab., Berkeley, CA, 1975.

[4] I. N. Hajj, P. Yang, and T. N. Trick, "Avoiding zero pivots in the modified nodal approach," *IEEE Trans. Circuits and Systems*, vol. 28, no. 4, pp. 271–278, 1981.

[5] L. O. Chua and P.-M. Lin, *Computer–Aided Analysis of Electronic Circuits*. Englewood Cliffs, NJ: Prentice–Hall, 1975.

[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*. The John Hopkins Univ. Press, 2 ed., 1993.

[7] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct methods for sparse matrices*. Oxford University Press, 1986.

[8] A. George and J. W. Liu, "The evolution of the minimum degree ordering algorithm," *SIAM Rev.*, vol. 31, pp. 1–19, Mar. 1989.

[9] M. Yannakakis, "Computing the minimum fill-in is NP-complete," *SIAM J. Algebraic Discrete Methods*, vol. 2, no. 1, pp. 77–79, 1981.

[10] H. M. Markowitz, "The elimination form of the inverse and its application to linear programming," *Management Sci.*, vol. 3, pp. 255–269, 1957.

[11] E. G. Ng and P. Raghavan, "Performance of greedy ordering heuristics for sparse Cholesky factorization," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 4, pp. 902–914, 1999.

[12] I. A. Cavers, "Using deficiency measure for tiebreaking the minimum degree algorithm," Tech. Rep. 89-2, Dept. Comp. Sci., The Univ. Of British Columbia, Vancouver, British Columbia, Canada V6T 1W5, 10. Jan. 1989.

[13] I. J. Lustig, R. E. Marsten, and D. F. Shanno, "The interaction of algorithms and architectures for interior point methods.," in *Advances in optimization and parallel computing* (P. M. Pardalos, ed.), pp. 190–204, North-Holland, 1992.

[14] C. Mészáros, "Ordering heuristics in interior point LP methods," in *New trends in mathematical programming* (F. Giannessi, S. Komlósi, and T. Rapcśak, eds.), pp. 203–221, Boston, MA, U.S.A.: Kluwer Acad. Publ., 1998.

[15] E. Rothberg and S. C. Eisenstat, "Node selection strategies for bottom-up sparse matrix ordering," *SIAM J. Matrix Anal. Appl.*, vol. 19, pp. 682–695, July 1998.

[16] S. V. Parter, "The use of linear graphs in Gauss elimination," *SIAM Rev.*, vol. 3, pp. 119–130, 1961.

[17] G. Reißig and T. Klimpel, "Fill-In Minimierung in der Schaltkreissimulation." Erfindungsmeldung, Infineon Technologies, MP PTS, München, 28 Mar. 2000.

[18] "SPOOLES — An object oriented software library for solving sparse linear systems of equations." `netlib.bell-labs.com/netlib/linalg/spooles/`, 1999.