An algorithm for reducing the bandwidth of a matrix of symmetrical configuration

By G. G. Alway and D. W. Martin*

An algorithm is described which will reduce the bandwidth of a square matrix with symmetrically disposed coefficients, by permuting the rows and columns. A computer program for the algorithm has been used successfully to find the minimum bandwidth of matrices of order 30 or so arising in electrical and hydraulic networks, and for reducing substantially the bandwidth of matrices of order up to 250.

In many matrices, particularly those arising from the study of networks and frameworks, a large proportion of the coefficients are zeros. When the non-zero coefficients of such a "sparse" matrix are concentrated in a narrow "band" centred along the principal diagonal, the representation of the matrix in a computer or on paper is concise, and the arithmetic entailed in the solution of sets of linear equations associated with it is condensed. Unfortunately, sparse matrices often do not display such a compact band structure. At the same time it is common for the non-zero coefficients to occupy positions which are symmetrical about the principal diagonal. Considerable interest therefore attaches to the development of techniques for rearranging the non-zero coefficients of a matrix by permuting its rows and columns to provide the narrowest band possible, and yet preserve symmetry. (For matrices whose non-zero coefficients are located asymmetrically about the principal diagonal, one would permute rows and columns independently, and the problem of determining the narrowest band is then altogether more difficult. However, a reduction in bandwidth in this case might be obtained by inserting additional coefficients to produce a symmetrical array, applying the procedures of this paper, and then deleting the inserted coefficients from the compressed matrix so produced.)

For an $n \times n$ matrix with symmetrically disposed coefficients there are of course $n!$ permutations available, and when n is large the determination of a particular permutation is likely to prove a formidable task. However, the difficulty of finding a suitable permutation depends not solely on n, but also on the structure of the linear graph associated with the matrix. For example, when the matrix is such that the required permuted form contains no zero elements within the band, there exist only two corresponding permutations (one being the reverse of the other), and the techniques described in this paper will find one of them very efficiently, regardless of n. Conversely, experience suggests that the more permutations there are which give the narrowest band, the more difficult it is to find one of them.

In the absence of appropriate theorems from the theory of graphs, one approach to the problem is to construct a computer program to survey the $n!$ possible permutations, using such stringent conditions to dismiss unsuitable permutations that a comprehensive search can be completed in a reasonable time. This approach is the basis of the algorithm to be described here: it has proved successful in achieving the minimum bandwidth for matrices of order 30 or so, and in reducing substantially the bandwidth of matrices of order up to 250.

The algorithm presented was devised largely by experimenting with a prototype computer program, and modifying this to eliminate observed bottlenecks in its operation. It will be surprising and disappointing, therefore, if more extensive trials do not lead to further improvements of the algorithm. One purpose of this paper is to interest other workers in the extension of these ideas: the other is to present a tool which even in its present form has proved useful in the solution of practical problems.

Basic definitions and strategy

If working outwards in either direction from the principal diagonal we label $1, 2, \ldots b$ the parallel diagonals which must be traversed in order that only zero elements lie in the remaining diagonals, the number $2b + 1$ is the bandwidth of the matrix. We refer to b loosely as the "half-bandwidth" and define $b(l)$ to be the value of b for the matrix whose rows and columns are ordered according to the lth permutation in the *lexicographical sequence* of the numbers $1, 2, \ldots n$. (For example, when n is 3 the permutations corresponding to the 3! values of l are

l	1	2	3	4	5	6
permutation	123	132	213	231	312	321.)

We assume that the given matrix corresponds to the *identity permutation*, $123 \ldots n$, $l = 1$, and that the matrix is connected — since it would be prohibitively wasteful not to apply the algorithm to the connected submatrices.

The main part of the algorithm consists of a routine which starts with a given permutation in the sequence, and searches for the first permutation thereafter for which $b \leqslant m$, where m is a given integer: if no such permutation exists, the routine proceeds to the end of the sequence without success. In general the time

* *Mathematics Division, National Physical Laboratory, Teddington, Mddx.*

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

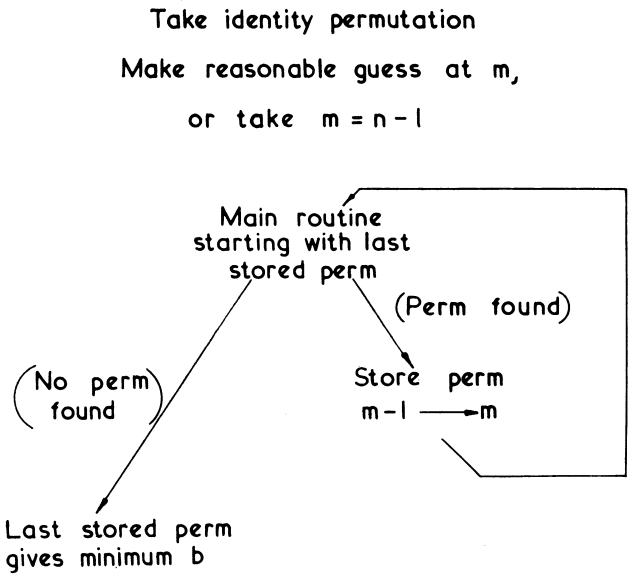


Fig. 1.—Outline flow diagram of algorithm

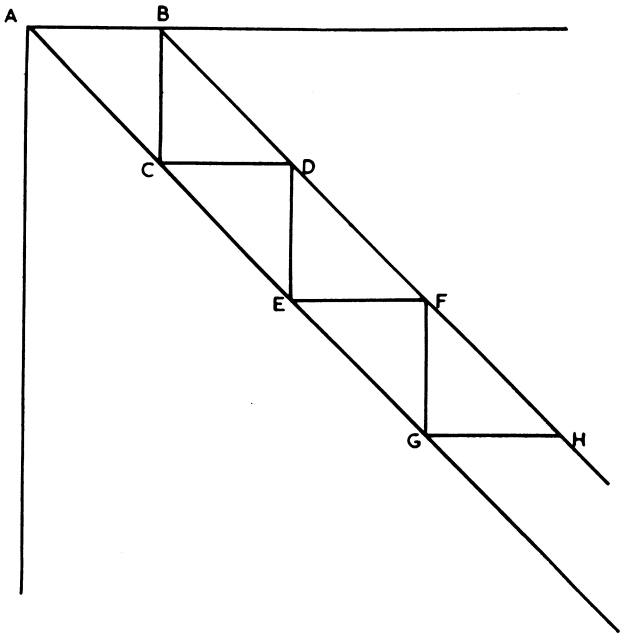


Fig. 2.—Constraints imposed by bandwidth on location of matrix coefficients

required for a search is larger the nearer m is to the minimum, but we have encountered matrices for which, with certain values of m, more time was taken to find a permutation giving $b \leqslant m$ than to find one giving $b \leqslant m - 1$ or $b \leqslant m - 2$. For the smaller matrices we have examined, no great loss of efficiency has resulted from using the routine in the simple manner described in **Fig. 1**.

The main routine is designed to examine together all those permutations which have the same initial s elements, and to determine, by means of two basic criteria Γ_1 and Γ_2, whether any of these $(n - s)!$ permutations have $b \leqslant m$. If, on the one hand, the routine can decide that all of these permutations have $b > m$, it proceeds to consider a subsequent set of permutations in the lexicographical sequence determined by other initial elements. Since in this sequence all permutations having the same initial elements occur together, rejection of permutations corresponding to some small value of s implies a large jump down the sequence. It is important, therefore, that the criteria Γ_1 and Γ_2 for accepting permutations should be as stringent as possible. (It will become clear in the next section that in concept Γ_1 is a sufficient criterion, but in fact it is not sufficiently stringent to be a practical tool by itself. Accordingly, it is supplemented in appropriate circumstances by Γ_2, and this has proved so stringent that the speed of the routine is increased by an order of magnitude.)

On the other hand, if the routine *cannot* decide that all of the $(n - s)!$ permutations have $b > m$, it increases s by 1 and proceeds to examine a subset of the permutations just considered. If s becomes $n - 1$, a permutation for which $b \leqslant m$ has been found.

The criteria Γ_1 and Γ_2

If the (i, j) coefficient of the matrix is non-zero, we say that the ith and jth rows of the matrix are *connected by one link*; if the (i, k) coefficient is zero, but the (i, j) and (j, k) coefficients are non-zero for one or more j, we say that the ith and kth rows are *connected by two links*, and so on. By considering all the connections of s rows, whose row numbers in the original matrix are $f_1, f_2, \ldots f_s$, to the remaining $n - s$ rows, criterion Γ_1 determines whether $f_1 f_2 \ldots f_s$ in that order can be the first s elements of a permutation for which $b \leqslant m$.

To illustrate this idea we consider the case $s = 1$ in detail. **Fig. 2** depicts the sort of sketch which the reader will need to construct repeatedly in familiarizing himself with the basic concepts of the algorithm. This sketch represents part of a matrix already in band form with half bandwidth m, in which the coefficients are assumed to occur at the nodes of a square lattice, e.g. at and between the points A, B, C, D, E, F, G, H where ACEG . . . is the principal diagonal and BDFH is the extreme diagonal of the band. Each horizontal line in the figure joins elements in a row of the matrix, and each vertical line joins elements in a column; and the hypotenuse and sides of each small triangle span $m + 1$ rows and/or columns.

The significance of the stepped line ABCDEFGH is that the non-zero super-diagonal coefficients which connect the first row of the matrix to other rows by just one link all lie on AB, and the rows so connected to the first row have their diagonal coefficients at C or between C and A. The coefficients which form connections of two or fewer links to the first row must all lie on or within the rhombus ABDC, and the rows so connected

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

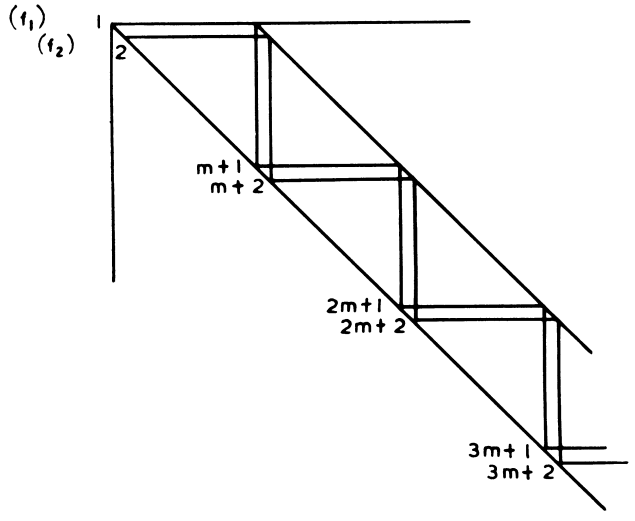


Fig. 3.—Sketch for applications of Γ_1 to f_1 and f_2

to the first row must have their diagonal elements at E or between E and A; and so on.

As can be seen from Fig. 2, it follows from the very definition of bandwidth that f_1 can be the first element of an acceptable permutation only if the number of rows which include

f_1 and rows connected to f_1 by one link $\leqslant m+1$

f_1 and rows connected to f_1 by two or fewer links $\leqslant 2m+1$

. . . .

f_1 and rows connected to f_1 by c or fewer links $\leqslant cm+1$.

The analogous situation for $s=2$ is sketched in **Fig. 3**, and we see that f_1 can be the first and f_2 the second element of an acceptable permutation only if the number of rows which include

f_1 and f_2 and the rows connected to f_1 by one link $\leqslant m+1$

f_1 and f_2 and the rows connected to f_1 by one link and the rows connected to f_2 by one link $\leqslant m+2$

f_1 and f_2 and the rows connected to f_1 by two or fewer links and the rows connected to f_2 by one link $\leqslant 2m+1$

f_1 and f_2 and the rows connected to f_1 by two or fewer links and the rows connected to f_2 by two or fewer links $\leqslant 2m+2$

and so on.

To keep the discussion concise we now represent the information on row connections by means of binary words or patterns whose individual bits correspond to the rows of the original matrix. The least-significant bit, denoted by π, corresponds to the first row, and the bit $2^{r-1}\pi$ to the rth row. We denote by $D(c,f)$ the word whose non-zero bits indicate the rows connected to row f by c or fewer links, so that the non-zero bits of $D(1,f)$, for example, indicate the non-zero coefficients in row f of the matrix.

We denote the number of non-zero bits in any word A by $q\{A\}$, so that $q\{D(c,f)\}$ is the number of rows of the matrix which are connected to f by c or fewer links. We denote by M the word whose non-zero bits indicate the s rows $f_1, f_2, \ldots f_s$; when $s=1$, M will contain the single non-zero bit $2^{f_1-1}\pi$. We use the conventional symbol $\vee$ to denote the logical operation "or".

The non-zero bits of $D(1,f_1)$ will include an entry corresponding to f_1 itself only if the (f_1,f_1) coefficient of the matrix is non-zero. However, when M is $2^{f_1-1}\pi$, "f_1 and rows connected to f_1 by one link" are bound to be covered by the word $M \vee D(1,f_1)$, and the number of these rows is $q\{M \vee D(1,f_1)\}$. Accordingly, the conditions that f_1 can be the first element of an acceptable permutation may now be written

$$\left.\begin{aligned} &q\{M \vee D(1,f_1)\} \leqslant m+1\\ &q\{M \vee D(2,f_1)\} \leqslant 2m+1\\ &\cdot\\ &\cdot\\ &\cdot\\ &q\{M \vee D(c,f_1)\} \leqslant cm+1 \text{ and so on.} \end{aligned}\right\} \quad (1)$$

Again f_1 and f_2 can be the first and second elements of an acceptable permutation only if

$$\left.\begin{aligned} &q\{M \vee D(1,f_1)\} \leqslant m+1 \quad \text{where } M \text{ is now } 2^{f_1-1}\pi + 2^{f_2-1}\pi\\ &q\{M \vee D(1,f_1) \vee D(1,f_2)\} \leqslant m+2\\ &q\{M \vee D(2,f_1) \vee D(1,f_2)\} \leqslant 2m+1\\ &q\{M \vee D(2,f_1) \vee D(2,f_2)\} \leqslant 2m+2 \end{aligned}\right\} \quad (2)$$

and so on.

As entries for the rows $f_3, f_4 \ldots$ are added to M, the gaps between the discrete groups of consecutive integers which constitute the right-hand sides of the inequalities analogous to (1) and (2) are gradually filled. Eventually we reach the position where $f_1, f_2, \ldots, f_m$ in that order can be the first m elements of an acceptable permutation only if

$$\left.\begin{aligned} &q\{M \vee D(1,f_1)\} \leqslant m+1 \quad \text{where } M \text{ is } 2^{f_1-1}\pi + 2^{f_2-1}\pi + \ldots + 2^{f_m-1}\pi\\ &q\{M \vee D(1,f_1) \vee D(1,f_2)\} \leqslant m+2\\ &\vdots\\ &q\{M \vee D(1,f_1) \vee D(1,f_2) \vee \ldots \vee D(1,f_m)\} \leqslant 2m\\ &q\{M \vee D(2,f_1) \vee D(1,f_2) \vee \ldots \vee D(1,f_m)\} \leqslant 2m+1\\ &q\{M \vee D(2,f_1) \vee D(2,f_2) \vee D(1,f_3) \vee \ldots \vee D(1,f_m)\} \leqslant 2m+2 \end{aligned}\right\} \quad (3)$$

and so on.

For $s > m$ the analogues of the inequalities (3) must hold with M having s non-zero bits but with Ds relating only to the m rows $f_{s-m+1}, f_{s-m+2}, \ldots f_s$, while the

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

right-hand sides take the values $s+1, s+2, \ldots n-1$. (As will be seen below, passage through other portions of the routine for smaller values of s will have ensured that the submatrix with rows $f_1, f_2, \ldots f_s$ has a half bandwidth not greater than m, and that the rows $f_1, f_2, \ldots f_{s-m}$ are not connected to any of the $n-s$ rows outside the submatrix.) If we denote by L_p the binary word which occurs within the braces on the left-hand side of the inequalities for successive values p of the right-hand side, and if we define c and d by the relations

$$1 \leqslant d \leqslant s, \quad s-m < d, \quad p-d = cm, \qquad (4)$$

(so that c is the least number of steps of length m back from p which brings us to an integer d which is not greater than s and positive—see **Fig. 4**), then

$$L_p = M \vee D(c, f_{s-m+1}) \vee D(c, f_{s-m+2}) \vee \ldots \vee D(c, f_d) \vee D(c-1, f_{d+1}) \vee \ldots \vee D(c-1, f_s). \qquad (5)$$

Here, as Fig. 4 indicates, connections of up to c links are considered for rows f_{s-m+1} to f_d, and of not more than $c-1$ links for rows f_{d+1} to f_s. Thus the non-zero bits of L_p indicate the rows which, by virtue of their connections to each other, must be placed among the first p rows of the matrix defined by the permutation currently under construction, for $p = s+1, s+2, \ldots n-1$ (or, when $s < m$, for those values of p for which $1 \leqslant p - cm \leqslant s, c = 1, 2, \ldots$). Clearly we require that $q\{L_p\} \leqslant p$, and this constitutes our criterion Γ_1.

If $q\{L_p\} > p$, $f_1 \ldots f_s$ cannot be the first s elements of an acceptable permutation, and the routine proceeds to consider subsequent permutations in the lexicographical list. If $q\{L_p\} < p$, the routine proceeds directly to the test for $p+1$, but when $q\{L_p\} = p$ the current situation must be examined more closely. The routine pursues different courses of action depending on whether $p = s+1$ or $p > s+1$.

In the former case the first test with Γ_1 on the rows $f_1, f_2, \ldots f_s$ shows not only that these may be the first s rows in the permuted matrix, but also that there must be a unique row which defines the $(s+1)$th element in the same permutation. This row corresponds to the single bit which is not common to M, which is composed of s non-zero bits, and to $L_{s+1} \equiv M \vee D(1, f_{s-m+1})$, which contains $s+1$ non-zero bits. Hence f_{s+1} is determined from $L - M$ without further tests, and the routine proceeds directly to apply Γ_1 to the elements $f_1, f_2, \ldots f_{s+1}$. (In fact this determination of f_{s+1} is the part of the routine referred to earlier which ensures that row f_{s-m+1} is connected only to rows which are included in $f_{s-2m+1}, f_{s-2m+2}, \ldots f_{s+1}$; and we know that row f_{s+1} will be linked to row f_{s-m+1} by a non-zero coefficient which lies on the edge of the band.) Moreover, the uniqueness of f_{s+1} means that if no acceptable permutation can be found having $f_1, f_2, \ldots f_{s+1}$ as its first $s+1$ elements, the next permutation in the sequence to be examined must correspond to a change of f_s or an earlier element. The special nature of f_{s+1} is recorded for subsequent reference by a binary indicator, W_{s+1}.

When $q\{L_p\} = p$ and $p > s+1$, we are not able immediately to determine any subsequent element of the current permutation uniquely, but we can apply a further criterion, Γ_2, to the rows which compose L_p. Γ_2 requires that the number of rows represented in L_p:

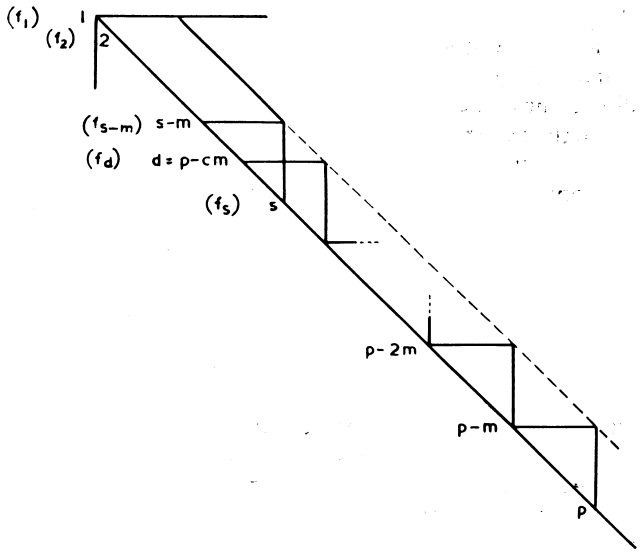


Fig. 4.—Sketch for an application of Γ_1 to $f_1, f_2, \ldots f_s$ $(s > m)$

which are connected by one link to rows not represented in $L_p \leqslant m$
which are connected by two or fewer links to rows not represented in $L_p \leqslant 2m$
. . . . . .
which are connected by $(c-1)$ or fewer links to rows not represented in $L_p \leqslant (c-1)m$ (6)

where c is defined in (4). If any of these inequalities is not satisfied, then the definition of bandwidth implies that $b > m$ for the current permutation, as with the earlier inequalities (1), (2) and (3). (In the case of c or fewer links in (6) equality will hold automatically since $q\{L_p\} = p$.)

To evaluate the numbers on the left-hand side of the inequalities (6) we consider the binary pattern L_{p+m}. By the definition of bandwidth, $L_{p+m} - L_p$ must include all those rows not represented in L_p which are connected by precisely one link to rows represented in L_p. Accordingly for each row represented in L_p, the chain of connection to a row not represented in L_p must involve at least one non-zero matrix coefficient which connects rows represented in $L_{p+m} - L_p$ directly to rows represented in L_p. Hence in the inequalities (6) we may replace "rows not represented in L_p" by "rows represented in $L_{p+m} - L_p$". In general the number of these latter rows will be much smaller than $n-p$, which is the total number of rows not represented in L_p, and considerable gains of speed result from using L_{p+m} in (6).

As a final comment on the ideas of this section, we remark that progression through the lexicographical list *can* be achieved without the use of Γ_2 and by restricting the application of Γ_1 to $p = s+1$. However, progress on these lines would be prohibitively slow.

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

Progression through the permutations

In the preceding section we described the basic criteria employed by the main routine in Fig. 1, but we did not discuss how the routine determines its detailed progress through the lexicographical sequence.

The permutation which defines the starting point for an application of the main routine can be specified by means of its first i elements $f_1, f_2, \ldots f_i$, on the assumption that the remaining elements are to be taken in their natural order. For the very first entry to the main routine with the identity permutation, we would specify $i = 1$ and $f_1 = 1$. In the event that the search had been interrupted at the point where all permutations beginning, say, 52 . . . had been considered, then the main routine would next enter the lexicographical list with $i = 2$, $f_1 = 5, f_2 = 3$, i.e. in the case $n = 6$, at the permutation 531246. Equally, if the routine has found the first permutation in the sequence for which $b \leqslant m$ and search is to proceed for a smaller bandwidth m^*, the initial permutation will be defined by $i = n - 1$ and corresponding elements $f_1, f_2 \ldots f_{n-1}$, since all previous permutations in the sequence will have already been excluded.

Within the routine we have an indicator g, whose definition is such that the given initial permutation determines the choice of trial elements f_s only as long as g exceeds zero. Accordingly g is given the initial value i, and is reduced by 1 each time an element of the given permutation is not rejected by criteria Γ_1 and Γ_2; and we reduce g directly to zero as soon as Γ_1 or Γ_2 call for departure from the given permutation.

If for any s, starting with s = 1, $q\{L_p\}$is found to be less than p for $p = s + 1, s + 2, \ldots n - 1$ in turn, then s is increased by 1 to s' and, depending on the state of the indicator g, the element to be taken as $f_{s'}$ is either that prescribed initially or that corresponding to the first permutation in the lexicographical list whose leading elements are $f_1, f_2, \ldots f_s$. For example, in the case referred to above with $n = 6$, we might start with $g = i = 2$ and $s = 1$ to test whether row 5 of the original matrix could be the first row in a permuted matrix having $m \leqslant 3$. If Γ_1 is not satisfied we reduce g to zero and test with $s = 1$ whether row 6 can be the first row; while if Γ_1 is satisfied we reduce g to 1 and test with $s = 2$ whether rows 5 and 3 of the original matrix can be rows 1 and 2 respectively in a permuted matrix. If Γ_1 is not satisfied we reduce g to zero and test with $s = 2$ whether rows 5 and 4 of the original matrix can be rows 1 and 2 respectively in a permuted matrix; and thereafter we test permutations beginning 56 (in the case of failure with Γ_1) or 541 (in the case of success). However, if Γ_1 is satisfied we reduce g to zero and test with $s = 3$ whether rows 5, 3 and 1 can be rows 1, 2 and 3 respectively in a permuted matrix, and thereafter test permutations beginning 532 or 5312.

In the examples discussed here Γ_1 could not be satisfied with $q\{L_{s+1}\} = s + 1$ because we did not consider a situation for which $s \geqslant m$. However, if we had started with $g = i = 4, f_1 = 5, f_2 = 3, f_3 = 2, f_4 = 1$, and we had sought a permutation for which $m \leqslant 2$, additional possibilities could have arisen. For example, let us suppose that 5 has been confirmed as a possible first element for a satisfactory permutation, and that we are testing 3 as a possible second element: thus we have $g = 3$, $s = 2$. If the test with $p = 3$ gives $q\{L_3\} = 3$, then we know that 5 and 3 can be acceptable first and second elements only in association with a unique third element corresponding to the non-zero bit in $L - M$. If this element is 2, we increase S to 3, reduce g to 2, make W_3 non-zero, and test whether 5, 3 and 2 are acceptable as the first, second and third elements respectively. However, if the bit in $L-M$ corresponds to row 1 in the original matrix, we cannot accept this row as the third row in a permuted matrix because permutations beginning 531 precede those beginning 532 in the lexicographical sequence: we therefore reduce g to zero and test with $s = 2$ whether permutations beginning 54 are acceptable. Yet again, if the bit in L—M corresponds to row 6, then we must pursue our tests with permutations starting 536, after reducing g to zero to mark our departure from the given initial permutation.

The general procedure to which these cases conform is the following. If, for any s, $q\{L_{s+1}\} = s + 1$, then s is increased by 1 to s' as before. If we are no longer required to heed a specified initial permutation, the binary indicator $W_{s'}$ is set non-zero, the non-zero bit in $L - M$ is added to M, and the routine proceeds directly to apply criterion Γ_1 to the augmented set of elements $f_1, f_2 \ldots f_{s'}$. However, if the indicator g exceeds zero we must confirm that acceptance of the row denoted by $L - M$ for $f_{s'}$ does not imply a regression to a permutation nearer the beginning of the lexicographical list than that specified initially. If the bit in $L - M$ is precisely that prescribed for $f_{s'}$, we set $W_{s'}$ non-zero, augment M and apply Γ_1 to $f_1, f_2, \ldots f_{s'}$; but if an *advance* down the list is implied we make g zero for future reference before so proceeding. On the other hand, if acceptance of the bit in $L - M$ for $f_{s'}$ implies *regression* to an earlier permutation, then we reduce s' by 1 to s, abandon the initial permutation, and test the first following permutation which the indicators $W_1, W_2, \ldots W_s$ show to be feasible. This permutation is determined as follows, and the same procedure is used whenever a jump down the lexicographical list is required because $q\{L_p\}$ exceeds p.

First, $2^{f_s - 1}\pi$ is removed from M to restore the value at the beginning of the cycle. If W_s is zero, we take as our new trial f_s the row-number corresponding to the first zero bit in M which follows the bit just rejected, and apply Γ_1 to the new $f_1, f_2, \ldots f_s$. If only non-zero bits follow the f_s rejected, or if W_s is 1, then we know that the next permutation in the sequence which should be considered must have f_{s-1} or an earlier element changed. Accordingly we reduce s by 1 to s'', and seek an alternative element to $f_{s''}$ by removing $2^{f_{s''} - 1}\pi$ from the current M, and repeating the steps just described as often as necessary. When s is reduced to zero, we know that no permutation exists with $b \leqslant m$.

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

Special operations used in the algorithm

To simplify the presentation of a concise flow diagram implementing the above ideas, we now define additional notation and logical operations which, together with arithmetical functions, have to be performed on the binary patterns and integers considered. In the computer it is, of course, essential that all operations be executed as speedily as possible.

Small letters are used throughout to denote integers and/or the stores which hold them, while capital letters denote binary patterns similarly. We assume that each binary pattern is $(n + 1)$ bits long, where n is the order of the matrix, and as before we denote by π the least-significant bit in every pattern . We denote the $(n + 1)$th bit, $2^n\pi$, by β when it is non-zero, and this will be seen in the flow diagram to serve as an indicator for changing elements in an unacceptable permutation. Likewise, it is convenient to combine with each binary integer f an additional bit to serve as the indicator W which must be set non-zero if the appropriate test with Γ_1 shows that $q\{L_{s+1}\} = s + 1$.

In addition to the logical "or" operation denoted by $\vee$, we require the logical "and", denoted by &. Also we assume that a facility exists for determining $q\{A\}$, the number of non-zero bits in the pattern A. Another function required is to find the integer r when the pattern $2^{r-1}\pi$ is given: r can be found as $1 + q(R)$, where $R = 2^{r-1}\pi +$ ones, that is, $R = 2^{r-1}\pi - \pi$.

Given a pattern A we need to determine its least-significant zero bit which is not less than $2^{r-1}\pi$, that is, the first gap in A which comes not before $2^{r-1}\pi$. We denote the pattern with a non-zero bit corresponding to the gap by $S(A, 2^{r-1}\pi)$; and so the pattern corresponding to the first gap which *follows* $2^{r-1}\pi$ will be $S(A, 2^r\pi)$. In fact, $S(A, 2^{r-1}\pi)$ can be obtained by the operations:

$$A + 2^{r-1}\pi \rightarrow B$$
$$B \;\&\; (-2^{r-1}\pi) \rightarrow B$$
$$B \;\&\; (-B) \rightarrow S(A, 2^{r-1}\pi).$$

The patterns $D(c,f)$, whose non-zero bits indicate the rows of the matrix which are connected to row f by c or fewer links, are required only for certain values of c and f. Accordingly it is economical to work out any pattern only when it is required in the main routine; but once a pattern has been obtained it should be stored, together with a record of the fact that it has been evaluated, in case it should be required again. Since whenever $D(c,f)$ is required $(c \geqslant 2)$, $D(c-1,f)$ and $D(c-2,f)$ will have been computed, we can obtain $D(c,f)$ from these as in **Fig. 5.** $D(1,f)$, whose non-zero bits correspond to the non-zero coefficients in row f of the original matrix, must be supplied to the program, of course; and we define $D(0,f)$ to be $D(1,f)$ & $2^{f-1}\pi$, that is, it consists of a single non-zero bit corresponding to the diagonal coefficient in row f (when this coefficient is non-zero).

Consideration of Fig. 4 and its analogue for $p - 1$

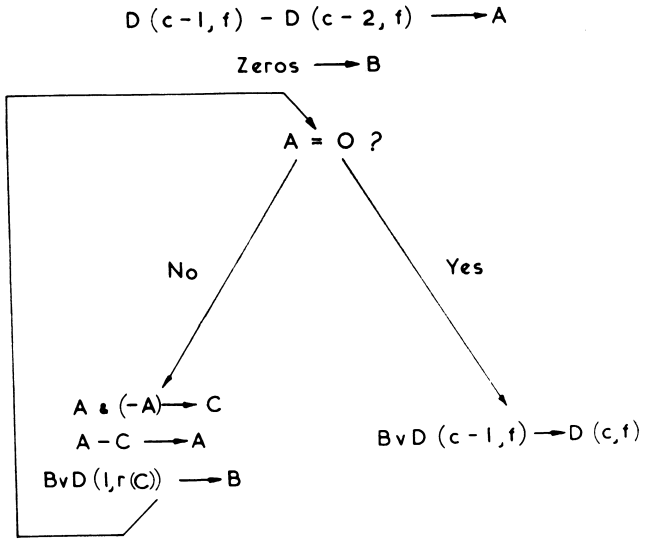


Fig. 5.—Determination of $D(c,f)$ from $D(c-1,f)$ and $D(c-2,f)$

shows that in all circumstances L_p satisfies the recurrence relation

$$L_p = L_{p-1} \vee D(c, fd),$$

where d is defined by equations (4). When no integer d can be found (i.e. for certain values of p when s is less than m), L_p is taken equal to L_{p-1}.

The flow diagram for the main routine is given in **Figs. 6**(*a*) and **6**(*b*). The seven returns from Fig. 6(*b*) to Fig. 6(*a*) are followed in the following circumstances:

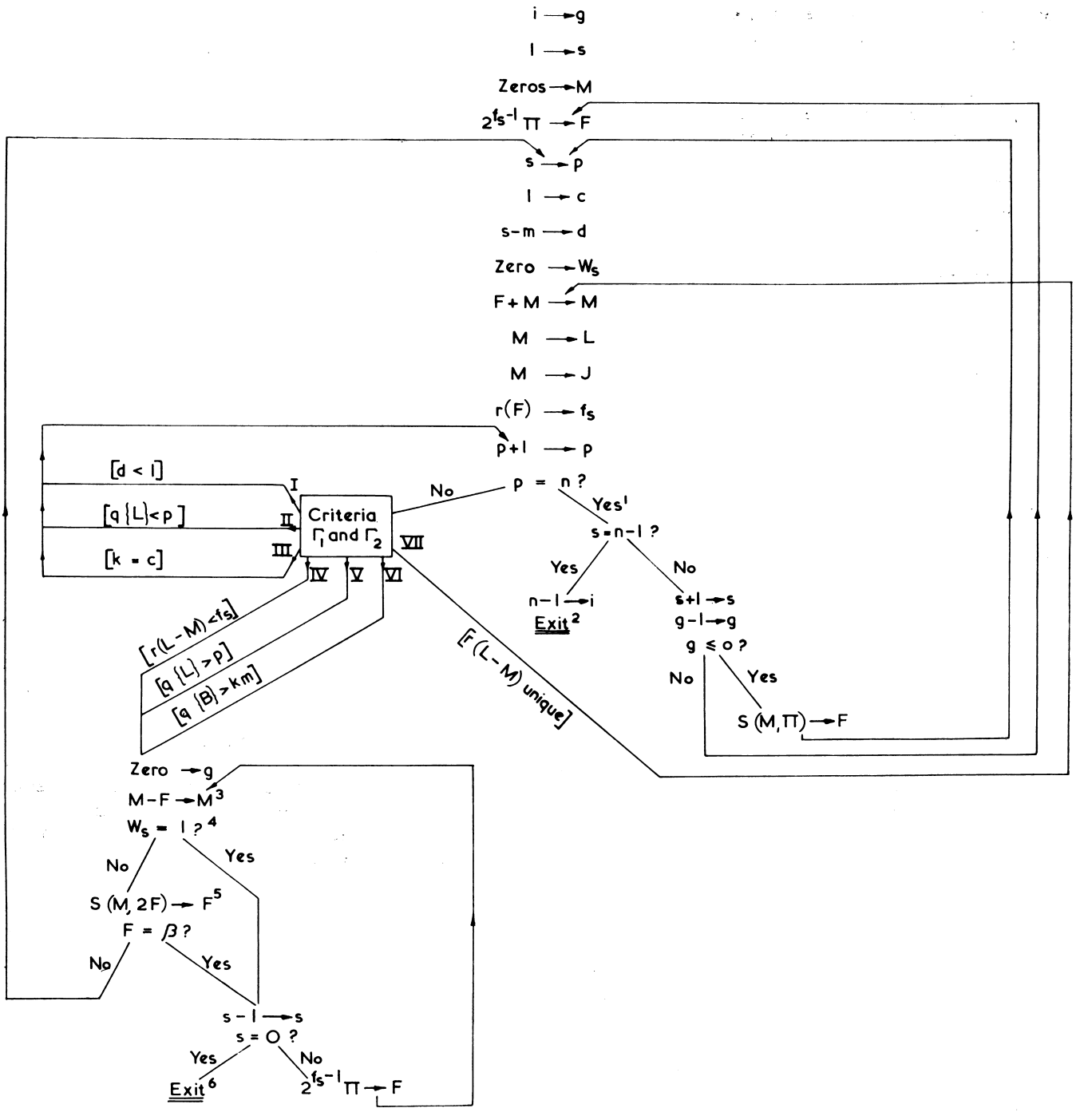
I. p does not have a value for which we can apply Γ_1. Therefore increase p by 1.
II. Γ_1 has provided no basis for action. Therefore increase p by 1 and apply Γ_1 to that case.
III. Γ_2 is bound to be satisfied. Therefore do not apply it but test with Γ_1 for the next value of p.
IV. Γ_1 has shown that acceptance of the unique element not common to L and M implies regression to a permutation earlier in the list than that given initially. Therefore depart from the initial permutation, by seeking change of last element but two (or earlier still).
V. Γ_1 has shown that $f_1, f_2, \ldots f_s$ cannot be first s elements in a permutation for which $b \leqslant m$. Therefore jump down list of permutations to the next which is worthy of consideration after that just rejected.
VI. Γ_2 has shown that $f_1, f_2, \ldots f_s$ cannot be first s elements, etc. (as in V).
VII. The unique element not common to L and M is acceptable as an addition to the group of elements which have not yet been rejected by Γ_1 and Γ_2.

Concluding remarks

In this final section we discuss the principal shortcomings of the algorithm as revealed by use of the computer program on practical problems. Our remarks

(*Continued on page* 271)

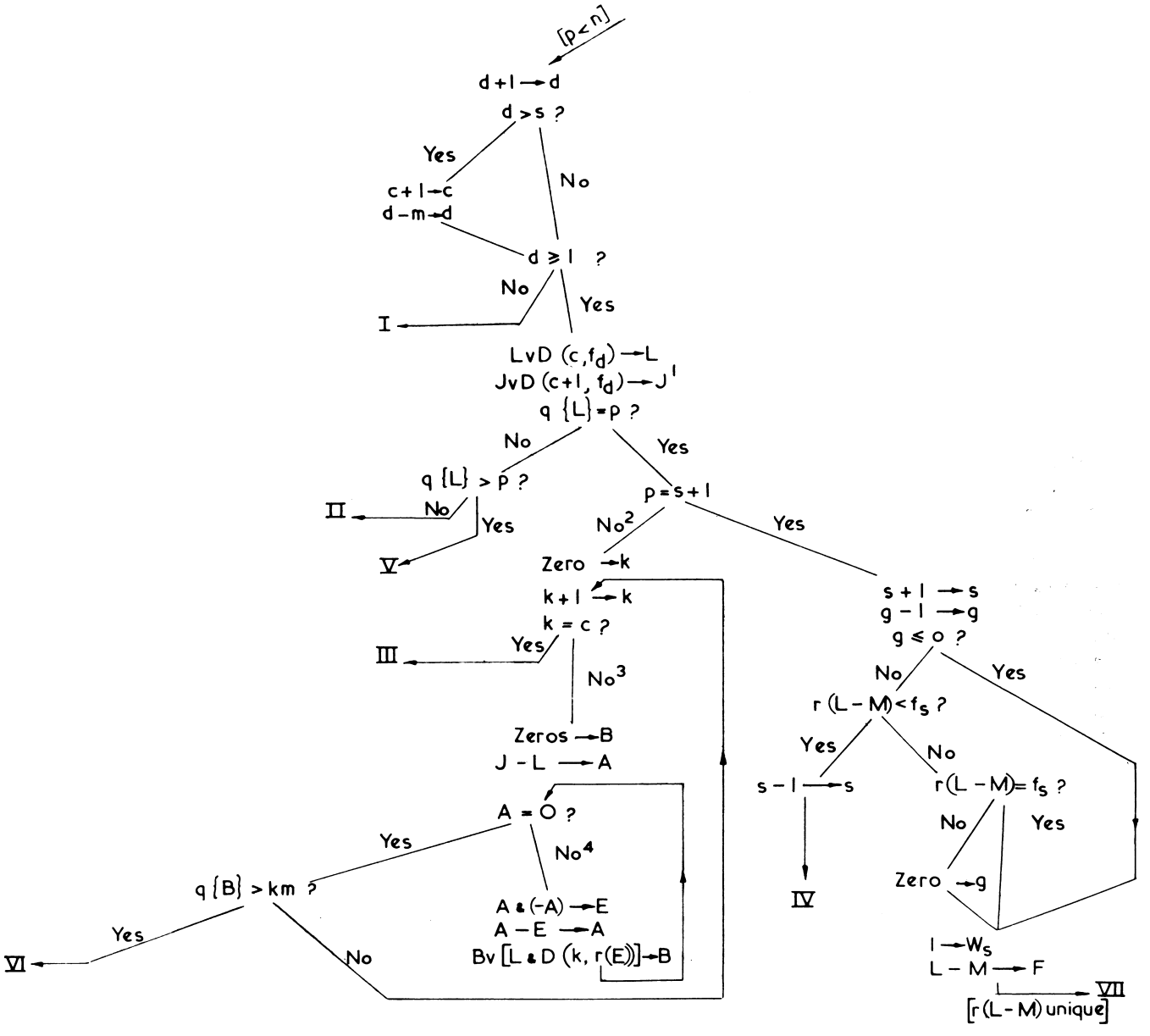
Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024



1. After n–s–1 tests Γ_1 and Γ_2 have not rejected $f_1, f_2, \ldots f_s$ as the first s elements in a permutation for which $b \leqslant m$.
2. Permutation for which $b \leqslant m$ has been found.
3. Restore original M.
4. Do we require a permutation determined by change of f_{s-1} or an earlier element?
5. First gap in M which follows f_s.
6. No permutation exists for which $b \leqslant m$.

Fig. 6(*a*) Flow diagram of main routine

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024



1. This gives incorrect J for $c = 1$ because J then is taken to be $M \vee D(2, f_d) \vee D(2, f_d - 1)$ . . . . only, whereas it should also include $D(1, f_d + 1)$, $D(1, f_d + 2)$ etc. However, the error is unimportant because J is not used for $c = 1$: see 3.
2. Entry to Γ_2.
3. This path is followed only when $c \geqslant 2$.
4. This loop removes rows in $J-L$ one by one and counts them.

Fig. 6(*b*).—Flow diagram of main routine: criteria Γ_1 and Γ_2

(*Continued from page* 269)

are divided between comments on the criteria Γ_1 and Γ_2, on the organization of the complete program, and on use of the lexicographical ordering.

In the first place, Γ_1 and Γ_2 both discriminate on L_p, which in general, as we have seen in (5), has the form

$$M \vee D(c, f_{s-m+1}) \vee D(c, f_{s-m+2}) \vee \ldots \vee D(c, f_d) \vee D(c-1, f_{d+1}) \vee \ldots \vee D(c-1, f_s).$$

Therefore, when Γ_1 or Γ_2 rejects those permutations for which the first s elements are $f_1, f_2, \ldots f_s$, then on the

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024

basis of the same information we should also reject every permutation which can be obtained by reordering $f_{s-m+1}, f_{s-m+2}, \ldots f_d$ among themselves, and by reordering $f_{d+1}, \ldots f_s$. Unfortunately the algorithm as it stands will labour through a number of these $m!/(s-d)!(m-s+d)!$ unacceptable permutations, regardless.

At the same time we have observed that, for the large matrices we have handled, the tests with Γ_1 and Γ_2 have been less useful for values of p near $n-1$ than for values near s. Thus considerable alteration of the first s elements accompanied tests with the smaller values of p, while Γ_1 and Γ_2 were satisfied without great difficulty for later values. There seems hope, therefore, that time might be saved by omitting the tests for large values of p, but unfortunately it is not clear how to choose the final value of p appropriate to any particular matrix.

Regarding the organization of the complete program, we stated when introducing Fig. 1 that we have obtained satisfactory results for small matrices by reducing m in steps of one until the minimum bandwidth was determined. (In the case of large matrices m was reduced until the time required to achieve a further reduction became exorbitant.) A more efficient procedure, anomalies apart, would be to approach the minimum bandwidth by a process of bisection as follows. We use two parameters m_1 and m_2 with initial values 1 and $n-1$, respectively. We take our first guess for m to be the integer halfway between m_1 and m_2 (or the larger of two middle integers) and enter the lexicographical list with the identity permutation. If the main routine finds a permutation for which $b \leqslant m$ we continue the exercise with $m_2 = m$; while if no satisfactory permutation is found we continue with $m_1 = m$. In place of Fig. 1 we proceed as in **Fig. 7.**

Nonetheless, unqualified application of even this program is unlikely to make the very best use of the main routine. For let us suppose we are examining a large matrix, or one for which the search through the permutations takes a prohibitively long time when m is near the minimum. When the program selects an m in this range, it will not improve upon its previous reduction of the halfbandwidth to m_2; whereas if the routine operated with $m = m_2 - a$, where a is some small integer, it might find a permutation for which $b \leqslant m_2 - a$ in a reasonable time. This cycle could then be repeated.

Lastly, we return to the prime strategy underlying the

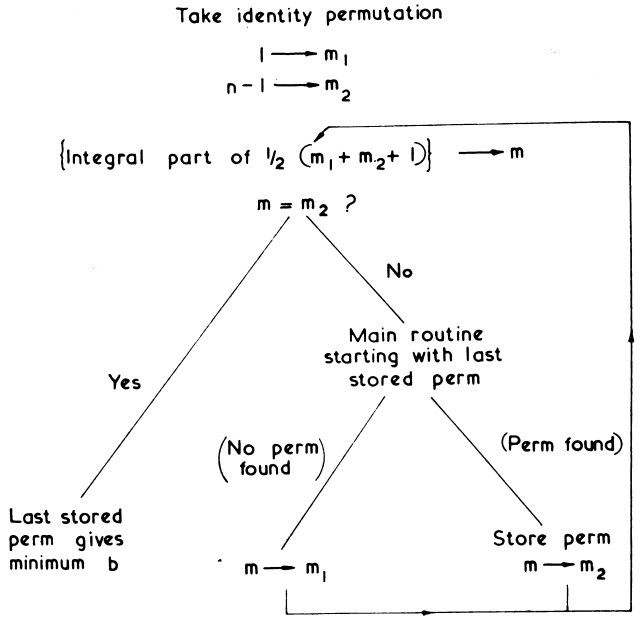


Fig. 7.—Improved use of the main routine

main routine, namely, that a search is made through all the possible permutations in a fixed order, without regard to the structure of the linear graph of the matrix. When the matrix is such that the minimum bandwidth is determined by the structure of a small sub-graph, the routine would waste a lot of time testing different permutations which involve reordering the rows and columns not in the corresponding submatrix. This means that, in general, for even quite small matrices of this type the routine would be useless. To overcome this defect further work is in progress on a method of avoiding the use of a fixed sequence of permutations, by having the configuration of the matrix decide which permutation shall be considered at any stage of the reduction.

Acknowledgement

The work described above has been carried out as part of the research programme of the National Physical Laboratory, and this paper is published by permission of the Director of the Laboratory.

Downloaded from https://academic.oup.com/comjnl/article/8/3/264/336112 by guest on 13 March 2024