Mansoura Bulletin, December 1978.

PARALLEL COMPUTATION OF LARGE SCALE SPARSE NETWORKS VIA TEARING

BY

Dr. E.H. Elkonyaly, Dr. R.M.K. El-Dewieny
and Dr. A.A.Hassan

ABSTRACT:

The concept of network diakoptics (tearing) is reviewed. The basic equations for this approach have been derived using circuit theorems. A method for tearing large scale networks called "dissection" is adopted. An algorithm based on this concept is derived.

The solution algorithm is well suited for parallel computation. Parallel processing offers solution time proportional to $\log_2 N$ comparod to N with serial computation. Potential advantages are explained. Possible improvements are also mentioned.

1. INTRODUCTION:

Solution of large sets of sparse simutaneous linear equations of the form $A \underline{x} = \underline{b}$ occurs frequently in various areas of engineering, particularly in electric power systems. As power system networks grow in size and complexity, the need for a superior solution technique is always and will be an open problem.

Diakoptics, or piecewise method of analysis, is one way to achieve this task. This method was originated by Kron[1] in the early 1950's. Kron's method was based on orthogonal network theory. The basic idea is to analyze an electrical (or mechanical) system (or a schematic topological model of it) by tearing it into a number of subdivisions (areas). This is achieved by removing some few elements or tearing some few nodes. The solution for the entire system is then obtained by analyzing each subdivision together with a set of equations to include the effect of the removed parts. Happ[2,3] and others[4] have recently extended and clarified the original obscure work of Kron.

In the 1960's, Tinnsy and his co-workers, in a series of publications[5,6,7], have introduced the concept of network sparsity into power systems' solution methods. They did actually improve the execution time and storage requirements for solving the entire interconnected system without tearing.

The outhors are lecturers at the Dept.of Elect. Power and Machines Engineering, Faculty of Engineering, University of El-Mansoura, El-Mansoura, Egypt.
Mansoura Bulletin, December 1978.

However, the recent developments in computer technology, i.e. the advant of parallel processors and the concept of on-line control[8] have once again enhanced the idea of solving networks by tearing. The purpose here is two-fold: improvement in the execution time and the efficient use of parallism in computing elements. This is particularly very important when the solution time is a critical factor, i.e. on-line applications and time consuming problems like stability and sensitivity analysis[9].

The paper goes along with this line of thoughts. First, the basic diakoptic equations are reviewed. Its potential disadvantages are discussed. Then, a tearing method to sectionalize the large system into pieces is introduced. The solution algorithm as well as the parallel computing layout are given. Potential advantages and possible improvements are also discussed.

2. Basic Diakoptic Equations and Network Bus Admittance Matrix:

The geometrical structure of a network can be described by replacing the network components by a linear graph. For a network with no mutual coupling between elements, the primitive impedance matrix, as well as the primitive admittance matrix, contain only diagonal elements. The connectivity between network branches and nodes are expressed in the incidence matrix $[A]$ [10].

The matrix $\underline{A}$ for the network of Fig.(1), can be partitioned into two submatrices A_r and A_t as follows:

$$[A] = [A_r \;\vdots\; A_t] \qquad \ldots\ldots\ldots(1)$$

where r stands for remaining elements;
t stands for torn elements.

Similarly, the primitive matrix y can be partitioned into

$$[y] = \begin{array}{|c|c|} \hline y_r & 0 \\ \hline 0 & y_t \\ \hline \end{array} \qquad \ldots\ldots\ldots(2)$$

Kirchhoff's current law and Kirchhoff's voltage law for the torn network take the following form:

$$v_r = A_r^T \underline{V} \qquad \ldots\ldots\ldots(3)$$

$$v_t = A_t^T \underline{V} \qquad \ldots\ldots\ldots(4)$$

$$A_r\, i_r + A_t\, i_t = \underline{I} \qquad \text{.....(5)}$$

The current voltage relations for elements are

$$i_r = y_r\, v_r = y_r\, A_r^T\, \underline{V} \qquad \text{.....(6)}$$

$$i_t = y_t\, v_t = y_t\, A_t^T\, \underline{V} \qquad \text{.....(7)}$$

where

v_r and v_t are voltage vector across elements;

i_r and i_t are current vector through elements;

T denotes transposition;

$\underline{V}$ is the vector of bus voltages; and

$\underline{I}$ is the vector of the injected bus currents of the complete network.

Substitution of (6) into (5) gives

$$A_r\, y_r\, A_r^T\, \underline{V} + A_t\, i_t = \underline{I}$$

or:

$$Y_r\, \underline{V} + A_t\, i_t = \underline{I} \qquad \text{.....(8)}$$

From (7):

$$A_t^T\, \underline{V} = y_t^{-1}\, i_t$$

$$= Z_t\, i_t$$

or:

$$A_t^T\, \underline{V} - Z_t\, i_t = 0 \qquad \text{.....(9)}$$

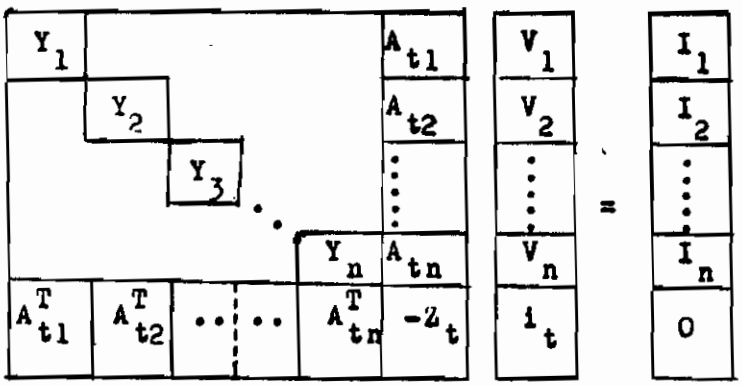
$$Z_t = y_t^{-1}$$

$$Y_r = A_r\, y_r\, A_r^T = \text{Y-bus for untorn network}$$

In matrix form, equations (8) and (9) are written as:

$$\begin{bmatrix} Y_r & A_t \\ A_t^T & -Z_t \end{bmatrix} \begin{bmatrix} \underline{V} \\ i_t \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix} \qquad \text{.....(10)}$$

Equations (10) can be generalized for n sections as follows



.....(11)

In its present form, equation (11) has been treated by many authors[11] and still getting much attention because of its special structure, i.e. the system is semi-coupled. Special technique for solving this system may result in a great saving in computational time. It should be mentioned, however, that using diakoptic approach to arrive at this form does not offer special advantage, from computational point of view, for most practical systems. One factor which makes the diakoptic algorithms less efficient is the large number of torn elements. It is desired, in general, to partition a system into several areas, so that parallel processing, i.e. solving the subsystems together at the same time, can efficiently be performed.

Partitioning a system into several areas means tearing many elements. This results in a large corner of the diakoptic equation matrix. This in turn results in lots of operations and less efficiency for diakoptic algorithms.

The diakoptic approach, however, can be made superior to the direct computational methods in a network containing one or more large loops[12]. Another unique feature is found in analizing certain non-linear networks[13].

The obvious remedy for the diakoptic method is a tearing method which offers the same special structure for the system equations and at the time reduce the corner blocks to a near minimum. The next section adopts what so called dissection strategy[14] to decompose the system into the suitable form for parallel processing and at the same time offers an efficient execution time.

• Network Tearing:

The direct solution of large scale networks has been improved by using Tinney's ordering schemes[6]. In short, those schemes are designed to explcit the sparsity structure of the bus and ordering nodes so that the number of generated elements during solution by Gaussian elimination is kept to a near

minimum. A special ordering subroutine capable of segregating networks by areas can really decompose the system. However, the logic to achieve this task has proved to be difficult. Moreover, the whole system has to be read into the computer memory to start segregation.

It has also acknowledged that matrix banding[15] can be used to partition the system. The purpose of matrix banding is to locate the non-zero elements near the major diagonal so that these entries can be stored in the computer memory in place of the entire matrix. However, a non-oriented matrix with scattered off-diagonal entries cannot be partitioned efficiently. Removal or renumbering of some nodes does not guarantee a partitioned matrix. A thorough and quick search process is therefore required to ensure true partitioning. A banded matrix facilitates partitioning without lengthy search. This method of partitioning loses its advantage with a continuosly structure-changing system like power networks.

An elegant and moreover efficient way of tearing can be obtained by adopting the so-called dissection (see the Appendix).This method possesses both speed and efficiency even if there is a substantial number of interconnecting subsections. A proposed dissection strategy for power system networks is illustrated in Fig.(2). By removing the nodes, interconnecting the subsections and numbering them last, we may achieve a very efficient overall solution strategy. The use of interactive graphics[15] during this dissection process can help identifying the tearing points.

The power of this method lies in the fact that identifying the initial subproblems exploits the physical structure of large scale systems. In electric power systems for example, a large power flow involves the interconnection of generators, loads and large network made up of areas connected by interties. Dissection through the interties actually performs tearing.

4. Equations Organization and Solution Algorithm:

The most crucial step in the analysis of power systems is the solution of a set of linear equations:

$$A\,x = b \qquad \text{.....(12)}$$

which arises in load flow, state estimation and stability analysis.

The proposed generalized dissection strategy decomposes the system of Fig. (2) into the following form.

$$\left[\begin{array}{c|c|c} Y_{11} & & Y_{13} \\ \hline & Y_{22} & Y_{23} \\ \hline Y_{31} & Y_{32} & Y_{33} \end{array}\right] \left[\begin{array}{c} X_1 \\ \hline X_2 \\ \hline X_3 \end{array}\right] = \left[\begin{array}{c} b_1 \\ \hline b_2 \\ \hline b_3 \end{array}\right] \qquad \text{.....(13)}$$

where

Y_{ii} is the Y-bus of area i with the boundary nodes removed (level 1 in Fig. 2), and

Y_{ij} is the Y-bus of boundary nodes.

It is noteworthy that, in the absence of mutual coupling between elements these Y_i's can be constracted by simple logical statements.

The solution of these equations can be obtained using LU factorization. However, for enhancing solution time, especially for on-line applications where time is a crucial factor, or in stability problems in which the solution time represents the most inefficient step in the solution sequence, parallel computation seems appropriate.

The following algorithm requires as many processor elements as many (areas + 1). The idea behind the proposed algorithm is:

Using equation (13)

$$Y_{11} X_1 + Y_{13} X_3 = b_1 \qquad \text{.....(14)}$$

Therefore

$$X_1 = Y_{11}^{-1} b_1 - Y_{11}^{-1} Y_{13} X_3 \qquad \text{.....(15)}$$

In general,

$$X_i = Y_{ii}^{-1} b_i - Y_{ii}^{-1} Y_{i,n+1} X_{n+1} \qquad \text{.....(16)}$$

where n is the number of subsystems.

For the lower border matrix of eqn. (13)

$$(Y_{33} - \sum_{i=1}^{2} Y_i') X_i = b_3 - \sum_{i=1}^{2} b_i' \qquad \text{.....(17)}$$

where $Y_i' = Y_{n+1,i} Y_{ii}^{-1} Y_{i,n-1}$

$b_i' = Y_{n+1,i} Y_{ii}^{-1} b_i$

Equation (17) has the general form

$$(Y_{n+1,\,n+1} - \sum_{i=1}^{n} Y_i') X_{n+1} = b_{n+1} - \sum_{i=1}^{n} b_i' \qquad \text{.....(18)}$$

To solve equations (16) and (18) efficiently, the algorithm proceeds as follows:

- Every processing element performs LU factors on one diagonal block together with its borders.
- The lower corner block is factored as follows.

$$Y_{n+1,n+1} = Y_{n+1,n+1} - \sum_{i=1}^{n} L_{n+1,i} U_{i,n+1} \qquad \text{.....(19)}$$

where $L_{n+1,i}$ and $U_{n+1,i}$ are triangular factors of the bordered blocks of the subsections. NOTE that, the product of equation(18) is not required since $Y_{n+1,n+1}$ can be updated using multipliers available in $L_{n+1,i}$ and $U_{i,n+1}$ from each elementry processor.

3- After factorization is completed, X_{n+1} is computed first and then X's are computed in parallel according to equation (16).

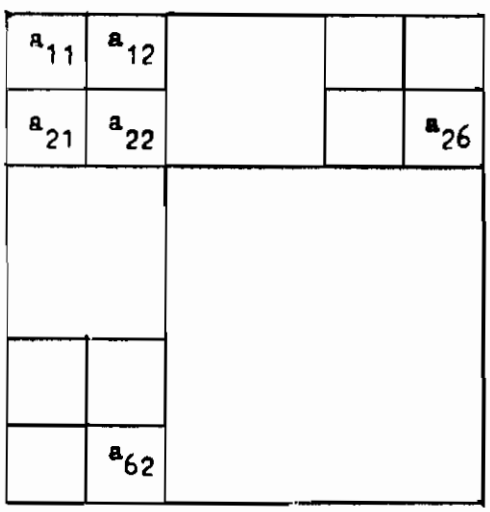
Careful data manipulation and use of sparsity techniques for storing and processing, can result in a very efficient implementation. The proposed parallel computation layout is illustrated in Fig.(3).

Example

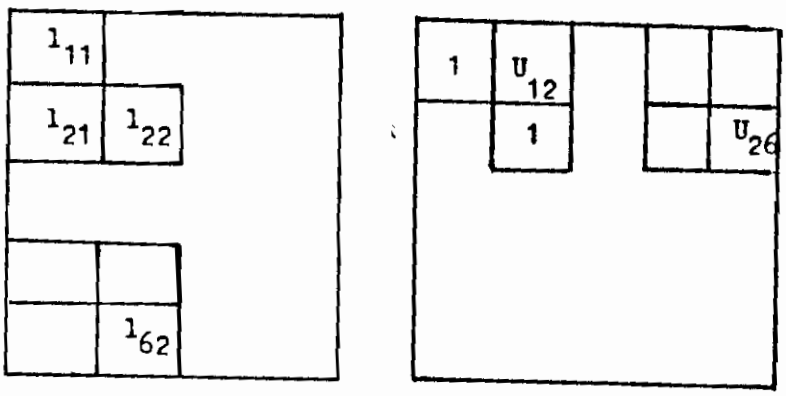
Consider the partitioned system which may be represented by matrix A, where

$$A = \begin{array}{|c|c|c|c|c|c|} \hline a_{11} & a_{12} & & & & \\ \hline a_{21} & a_{22} & & & & a_{26} \\ \hline & & a_{33} & a_{34} & & \\ \hline & & a_{43} & a_{44} & a_{45} & \\ \hline & & & a_{54} & a_{55} & a_{56} \\ \hline & a_{62} & & & a_{65} & a_{66} \\ \hline \end{array}$$

The processor for area 1 factors



into the following LU form

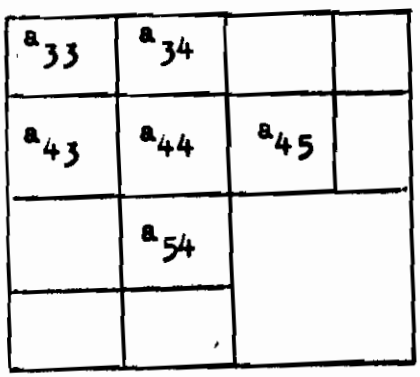


where $l_{11} = a_{11}$ $\quad l_{21} = a_{21}$

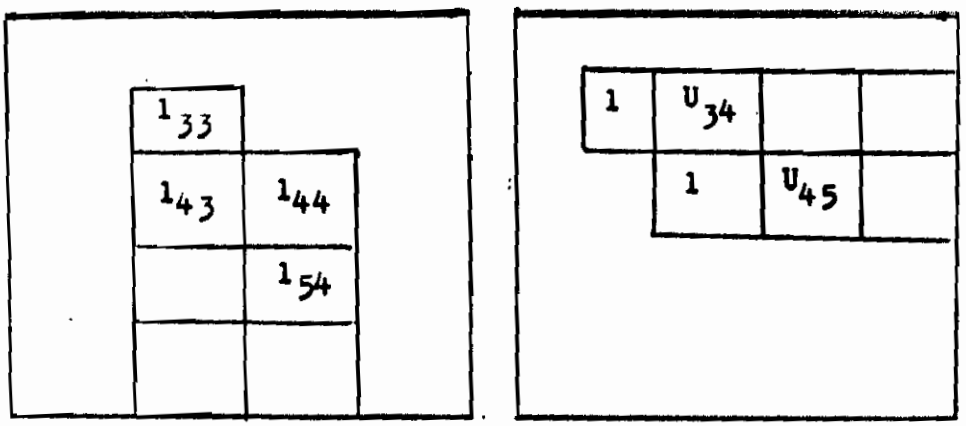
$U_{12} = \frac{a_{12}}{l_{11}}$ $\quad l_{22} = a_{22} - l_{21} U_{12}$

$U_{26} = \frac{a_{26}}{l_{22}}$ $\quad l_{62} = a_{62}$

The processor for area 2 factors



into the following LU form



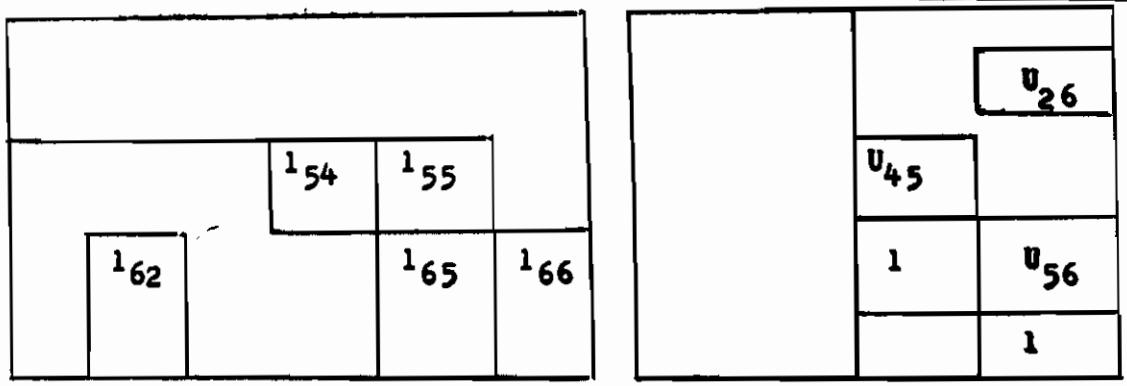
where $l_{33} = a_{33}$ $\quad l_{43} = a_{43}$

$U_{34} = \frac{a_{34}}{l_{33}}$ $\quad l_{44} = a_{44} - l_{43}\, U_{34}$

$l_{45} = a_{45} \,/\, l_{44}$

$l_{54} = a_{54}$

At the same time the processor for area 3 updates $\begin{vmatrix} a_{55} & a_{56} \\ a_{65} & a_{66} \end{vmatrix}$ to



where $l_{55} = a_{55} - l_{54}\, U_{45}$

$U_{56} = a_{56}/l_{55}$ $\quad l_{65} = a_{65}$

$l_{66} = a_{66} - l_{62}\, U_{26} - l_{65}\, U_{56}$

At this stage the factorization is completed. X_3 is calculated first, then the other two processors continue to compute X_1 and X_2 simultaneously.

5. DISCUSSION

As a matter of fact, serial processing is the same for area oriented and non-area oriented matrices with the exception that, for non-area oriented matrices serial processing(methods

not requiring explicit inversions) requires a few less operations. Therefore, area oriented matrices are advantageous if parallel processing is used. On the other hand, serial computing is not an efficient process for use on the new single instruction stream multiple data stream computers (SIMD)[15] that are capable of performing many simultaneous operations. The proposed algorithm seems appropriate for this type of computing machines.

From the point of view of speed, the standared solution time is proportional to (n+1), while the time for the proposed algorithm is proportional to $\log_2(n+1)$. There will be, however, some delay because of the broadcasting for updating the lower corner block. Nevertheless, careful programming practices will bring this problem to a minimum.

6. CONCLUTIONS:

A dissection tearing method offers the possibility of tearing a large scale network into subsections to enhance computation time. The method deals directly with the sparse Y-bus and does not deal with the network diakoptics equations.

An algorithm has been devised. The solution time has been enhanced to $\log_2(n+1)$ compared with (n+1) for serial computation. The method is well suited to the new generation of computers which offers the possibility of performing many simultaneous operations. It has been also mentioned that broad-casting and updating the border block may cause some execution delay.

The results of this paper stimulates other research topics including searches for other decomposing principles, analysis of numerical stability of these algorithms, minimal time for the solution and minimum parallism needed to solve them.

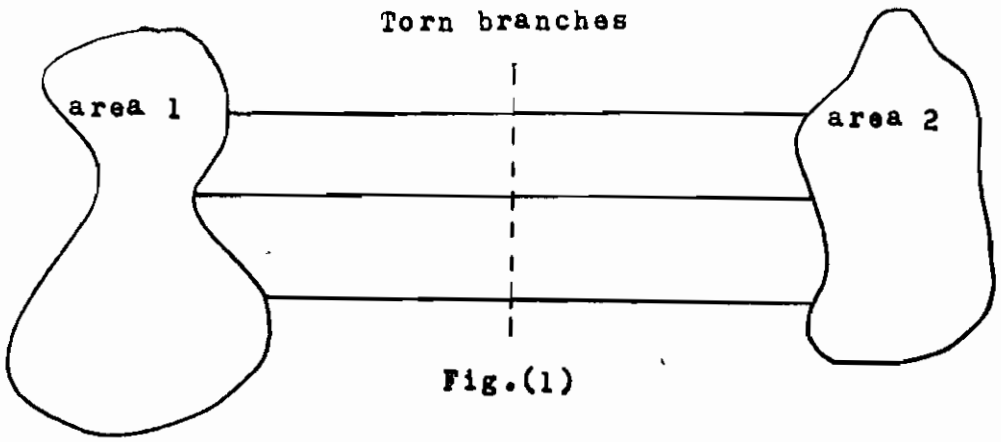


Fig.(1)

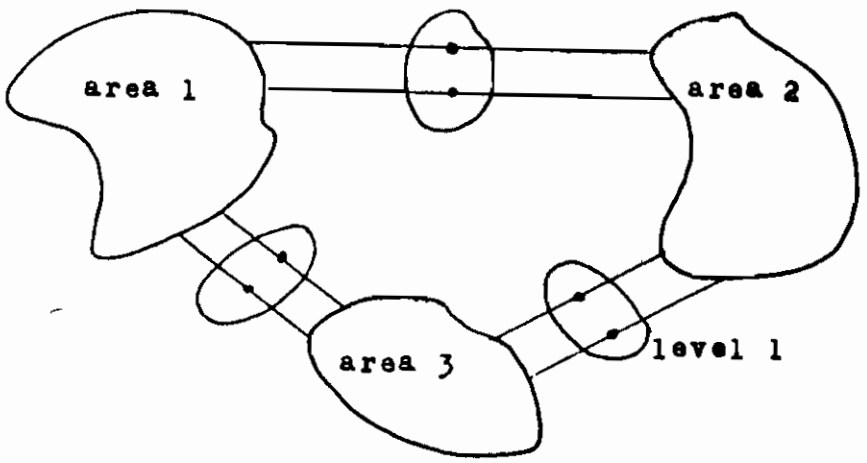
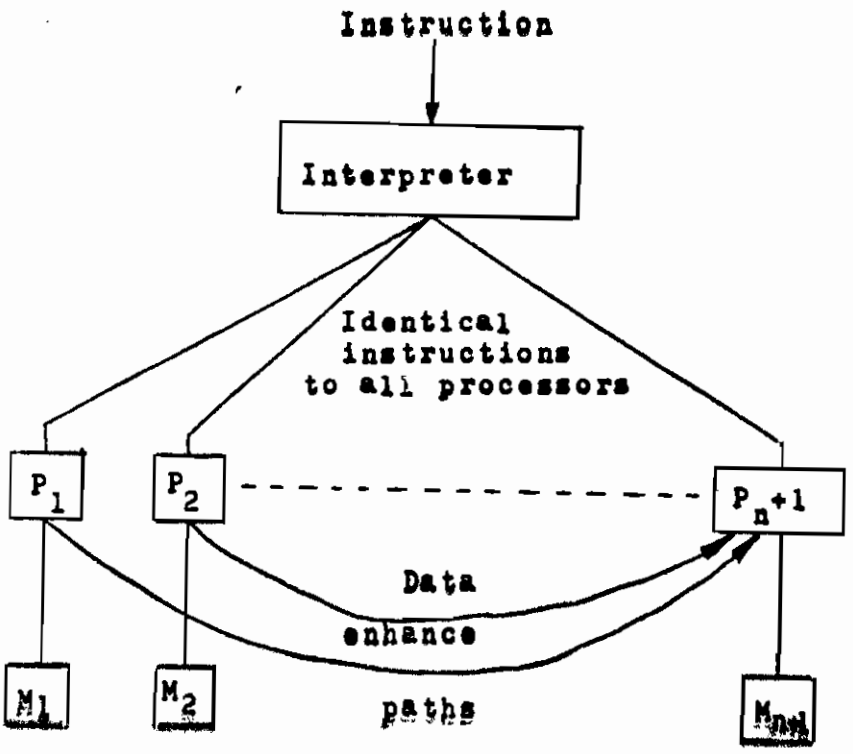


Fig.(2): Generalized dissection



APPENDIX

Dissection technique has been proposed to enhance computation in numerical algorithms of partial differential equations. The descritization process in case of a two dimensional problem results in the grid of Fig.a. The algorithm essentially sub-divides the grid and eliminates separately within the subdivisions before eliminating the subdivision boundaries. The elements ars eliminted in the order shown in the figure. The method as shown is called nested dissection.

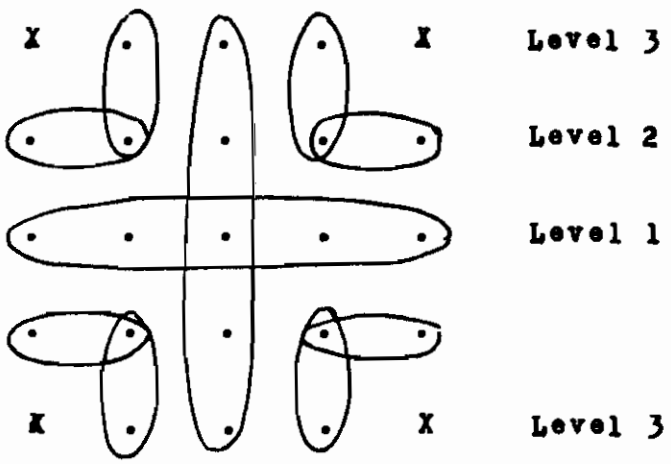


Fig. a. Nested dissection

Dissection, however, does not have to be nested to be useful. This is evident from Fig.b. It would be useful to first dissect along the lines shown and to eliminate independently with three regions (1) before completing the decomposition with region (2). In general, the best dissection will be done by examining the original structure as explained in the text.

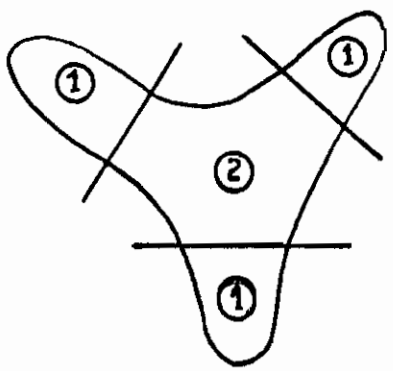


Fig. b, An irregularly shaped region

REFERENCES

1. Kron, G., "Diakoptics", McDonald, London, England, 1963.

2. Happ, H.H., "Diakoptics and Networks", Academic Press, New York, 1971.

3. Happ, H.H.& Young, C.C., "Tearing Algorithm for Large-Scale Network Programs", PICA Proceedings, pp 440-447, 1971.

4. Bhat, M.V. and Kesa Van, H.K., "Diakoptics Equations and Sparsity", IEEE PES Summer Meeting, Anaheim, California, July 14-19, 1974.

5. Sato, N. and Tinney, W.F., "Techniques for Exploiting the Sparsity of the Network Admittance Matrix", IEEE Trans., PAS-82, p 944, Dec. 1963.

6. Tinney, W.F. and Walker, J.W., "Direct Solution of Network Equations by Optimally Ordered Triangular Factorization", Proc. IEEE, pp 1801-1810, Nov. 1967.

7. Ogbuobiri, E.C., Tinney, W.F. and J. Walker, "Sparsity Directed Decomposition for Gaussian Elimination on Matrices", IEEE Trans., PAS-9, pp 141-150, 1970.

8. Dyliacco, T.E., "Real time Control of Power Systems", Proc. of IEEE, July 1974.

9. Jennings, "Matrix Computation for Engineering and Scientists" Book Adison Wisely, 1977.

10. Stagg, G. and El-Abiad, A.H., "Computer Methods in Power System Analysis" McGraw-Hill Book Company, New Yourk 1968.

11. "Circuits and Systems" Special issue, CAD-23, No. 12, IEEE Trans., 1976.

12. Alvarado, F.L., et al. "Sparsity in Diakoptic Algorithms" IEEE Trans, PAS-96, Sept. Oct. 1977.

13. Discussion of 12.

14. Duff, I.S., "A Survey of Aparse Matrix Research", Proc. of IEEE, Vol. 65, No. 4 April 1977.

15. Hobbs, L.C., et al., "Parallel Processor Systems, Technologies and Applications", Sparton Books, 1970.