Memoirs of the Faculty of Engineering, Okayama University, Vol.25, No. 2, pp.115-122, March 1991

Nested Dissection Method on Transputer

Takeo TANIGUCHI* & Hirotsugu IRIE**

(Received February 8 , 1991)

SYNOPSIS

Nested dissection method is an elimination method for a set of linear algebraic equations with minimum fill-ins. Physically it divides a domain into four subdomains, and each subdomain is again divided into four. This procedure is repeated till all nodes are included in some subdomains. Using this characteristic, the authors examine the efficiency of the method on the transputer.

1. INTRODUCTION

According to the development of the ability of computers, the size of problems treated in engineering fields has been growing. Then, faster solvers have been always required, and a number of effective solvers were proposed and applied for the actual problems. For the finite element users solvers for a large sparse set of linear equations have been required. Using the characteristics of the coefficient matrices, the sparse matrix technique is effectively introduced in the solvers and at present we find, for example, the band matrix method, the skyline method, and so on.

The efficiency of solvers can be judged by the ammount of necessary memories and also the CPU time required for the solution. Especially, the restriction of the memory size becomes important in recent years, when very large problems are treated.

---

* : Department of Engineering Science
** : Department of Civil Engineering

Any method based on the elimination necessarily requires additional memory for the storage of new nonzero entries called fill-in appearing during the forward elimination process. At present it is known that the nested dissection method generates the least number of fill-ins[1,2].

The elimination process used in the nested dissection method suggests us another important aspect, i.e. the parallel computation. At using the nested dissection method, physical domain is divided into subdomains, and the elimination of these subdomains can be treated independently. This characteristic obviously fits to the property of parallel machine.

In this paper the authors investigate the efficiency of the nested dissection method on a parallel machine. For this purpose they firstly explain the elimination procedure of the nested dissection method, discuss on the parallel coding of the method, and show the results of test problems. The results are compared with ones by other solvers.

2. HARDWARE AND SOFTWARE SYSTEM

The computer used for this investigation is a parallel machine called transputer with 2 processors, T-800. One of these processor is called the root processor, which governs another processor called subprocessor. Each processor has 1MB memory, and auxiliary 6MB memory is added to the root processor. The structure of this system prepares 8MB memory as a whole, but the subprocessor can store, at most, 1MB memory.

A microcomputer, NEC PC-9801RX, is used as I/O machine of the transputer, and a hard disk unit of 40 MB is attached to it. Operation system is MS-DOS. So, the user can treat whole system only using MS-DOS. For the computation, Parallel Fortran is used. Total hardware system is illustrated in Fig.1.

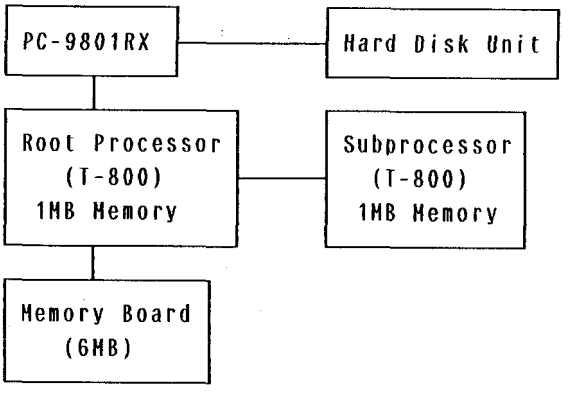

Fig. 1 Hardware System

For the efficient use of any parallel machine, the tasks loaded on each processor should be

equated. In above hardware system we can use two processors with 7MB and 1 MB, and the divided task for the machine is restricted by the subprocessor. That is, two processors can fully work when each processor subjects to the load less than 1MB.

3. ELIMINATION PROCESS FOR SPARSE SYSTEM

Let

$$Ax = b \qquad (1)$$

be a set of linear algebraic equations obtained by the application of the finite element method or the finite difference method.

Now, we consider on the elimination process for (1). Let a(i,j) be an entry of the matrix A. Then, the elimination of the k-th row of A replaces a(i,j) to a'(i,j).

$$a'(i,j) = a(i,j) - a(i,k)*a(k,j)/a(k,k) \qquad (2)$$

This relation shows that a(i,j) must be modified, if both of a(i,k) and a(k,j) are nonzeros. Then, the zero entry at a(i,j) becomes nonzero if the product a(i,k)*a(k,j) is nonzero. New nonzero entry a'(i,j) is called "fill-in".

The application of the theory of graph can clarify the generation of fill-in during the elimination ordering. At the presentation of the coefficient matrix A of (1) by a graph, we set firstly n nodes for expressing a matrix A(n*n), and indicate the existence of a nonzero entry a(i,j) of A by a line connecting a pair of nodes labelled i and j. Then, we can express the appearance of a fill-in also by a new line. Fig.2-a shows an example of a coefficient matirx, Fig.2-b presents its graph, and Fig.2-c shows the graph after the elimination of a node located at the centre. Dotted lines in Fig.2-c indicate the appearance of fill-ins, and we can remark that the subgraph consisted by nodes which are connected to the eliminated node is replaced by a complete graph after the elimination. That is, all nodes which are connected to the eliminated node are directly connected each other after the elimination.

In case of the band matrix method it is known that fill-ins appear only for entries inside of the bandwidth. For example, if the matrix

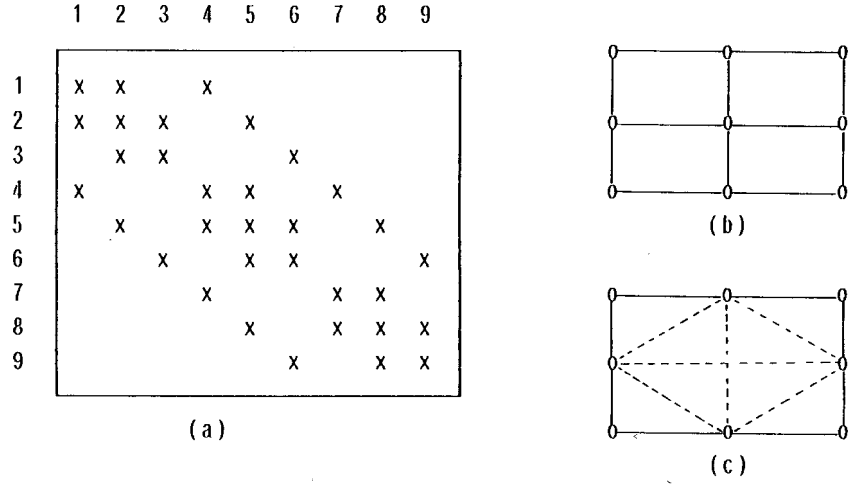

Fig. 2 Matrix and Its Graph

of Fig.2-a is eliminated according to the ordering of the matrix, all fill-ins appear within the halfbandwidth 4 including the main diagonal.

3. NESTED DISSECTION METHOD

Let R be a rectangular area, and subdivide the domain into N*N small rectangles by placing N nodes on each edge of the domain. Then, a grillage with (N+1)*(N+1) nodes are placed there.

Now, we consider on the elimination of nodes set in the area. A part of the grillage is shown as M(0) in Fig.3. The effect of the elimination of a node is restricted only within the subarea consisting of nodes which are connected to the eliminated one. The state after the first elimination is shown as M(1) in the figure, and bigger rectangle with 8 nodes show a complete graph. M(2) shows new state after the elimination of 2 nodes. At this elimination we can remark that no fill-ins appear for nodes which locate on the common edge of these two complete graphs, because fill-ins are already generated at the first elimination. The figures in Fig.3 clarify that fill-ins once generated are effectively used for the successive eliminations.

Nested dissection method uses the property of the elimination procedure mentioned above effectively. Assume a grillage with (N+1)*(N+1) nodes is placed in a rectangular area. The rectangular area is divided into four smaller rectangles, and call nodes on the bisecting lines as

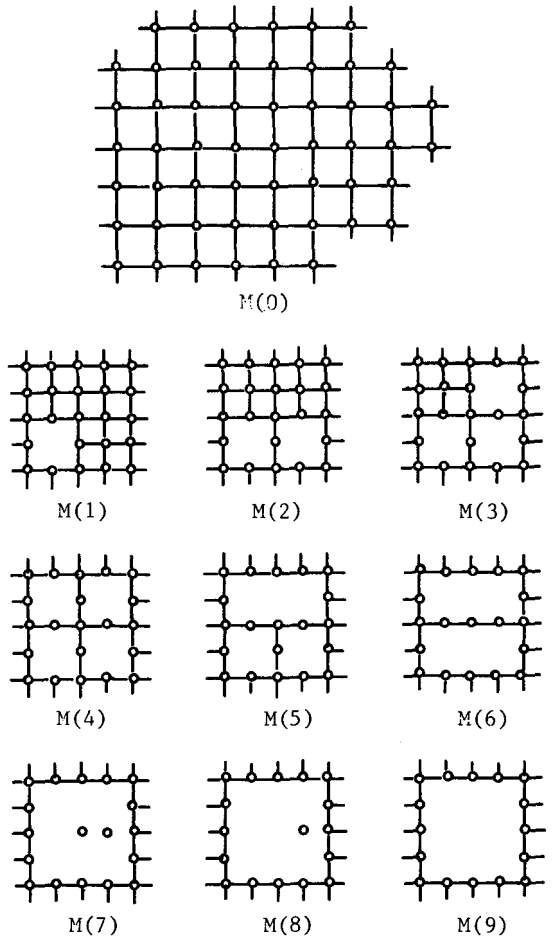

Fig.3 Nested Dissection Method

G(X). Each rectangle is divided into four again, and nodes newly selected are classified into G(X-1). This subdivision is repeated untill all nodes are included in some groups G(j). The final group is set in G(1). Then, "X" of G(X) indicates the number of groups. The elimination is started from nodes in the final group, and continued to nodes in successive group till all nodes are eliminated. The elimination ordering of nodes in the same group is arbitrary, but the elimination of another group must be done after the elimination of previous one. The different ordering of the elimination for nodes in the same group does not give any difference of tne number of fill-ins.

The elimination ordering of the nested dissection method can decrease total number of fill-ins, since

(1) the number of nodes connected to a node to be eliminated is

restricted to be small, and

(2) nonzero entries once generated during the elimination are modified as many times as possible not to create new fill-ins.

4. PROBLEMS AND RESULTS ON TRANSPUTER

For nodes on the grillage the elimination process can be easily parallelized. Since the elimination of a node modifies only the values of entries as shown in Fig.3, another node which is not influenced by the elimination can be eliminated at the same time.

This property of the nested dissection method suggests the applicability of it to the parallel computation. Obviously, the elimination of nodes included in the group, G(X), must be done in one processor, because they form a complete graph at the stage. That is, the elimination procedure of the nested dissection method can be parallelized except the treatment for nodes in G(X).

For the comparison of the efficiency of the nested dissection method, we choose following two solvers:

1. Gauss Elimination Method
2. Band Matrix Method

In following discussion GEM, BMM and NDM are used for Gauss Elimination Method, Band Matrix Method, and Nested Dissection Method, respectively.

These three solvers are tested using single processor and also two processors. The parallel coding for GEM and BMM is as following: Odd rows are solved by the root processor, and even rows are by the subprocessor. The parallel coding for NDM is due to the method mentioned above.

Now, we explain the problems used for the test. Two problems are prepared for our tests, in which Test 1 and 2 include 49 and 225 nodes, respectively. Fig's 4 and 5 show the groups of nodes of Test 1 and Test 2, respectively. In the figures, numbers from 1 to 4 indicate the nodes clasified into G(1), G(2), G(3) and G(4), respectively. Test 1 is almost one fourth of Test 2, which is enclosed by nodes classi-

1	2	1	3	1	2	1
2	2	2	3	2	2	2
1	2	1	3	1	2	1
3	3	3	3	3	3	3
1	2	1	3	1	2	1
2	2	2	3	2	2	2
1	2	1	3	1	2	1

Fig. 4 Groups of Nodes of Test 1

1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
2	2	2	3	2	2	2	4	2	2	2	3	2	2	2
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
3	3	3	3	3	3	3	4	3	3	3	3	3	3	3
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
2	2	2	3	2	2	2	4	2	2	2	3	2	2	2
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
2	2	2	3	2	2	2	4	2	2	2	3	2	2	2
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
3	3	3	3	3	3	3	4	3	3	3	3	3	3	3
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1
2	2	2	3	2	2	2	4	2	2	2	3	2	2	2
1	2	1	3	1	2	1	4	1	2	1	3	1	2	1

Fig. 5 Groups of Nodes of Test 2

Table 1 Results of Test 1 and Test 2

(msec)

	Test 1		Test 2	
	Single	Para	Single	Para
GEM	321.98	275.33	31690.05	24149.63
BMM	17.98	20.74	288.96	272.19
NDM	17.73	17.02	253.89	233.22

fied in G(4).

For NDM the elimination is started from nodes in G(1), and successively continued to nodes in G(2), G(3) and G(4). For BMM the elimiantion ordering is determined so that the half bandwidth is minimised. In these tests the half bandwidth is 7 for Test 1 and 15 for Test 2, and all data are stored in 2-dimensioanl array. Same elimination orderings are introduced in GEM, but the elimination is applied for all entries of the coefficient matrix A.

The results of Test 1 and 2 are summarized in Table 1. From the table we obtain following conclusions.

(1) Nested dissection method is the fastest solver among them for the single and also parallel computings.

(2) The efficiency of the parallel computaion for the nested dissection method appears already in small-size problem like Test 1. On the other hand, the band matrix method requires longer execution time in this case, and it can show the efficiency of the parallel computation for larger problem like Test 2. This difference is caused by the I/O time for the parallel procedure of these solvers.

The results of this table suggest that the nested dissection method becomes more effective for larger problems.

5.CONCLUDING REMARKS

The efficiency of the nested dissection method is surveyed using a parallel computer, and the comparison of CPU time showed that it is faster than the Gauss elimination method and the band matrix method. Moreover, the results suggest that it will be more effective for larger problems.

Test problems treated in this investigation are selected so that the nested dissection method can be easily applied. If more general problems are chosen for the tests, the use of the nested dissection method becomes difficult. In this case, the one-way dissection method should be introduced instead of the nested dissection method.[3]

ACKNOWLEDGEMENT

This work was financially supported by the scientific research fund of the Ministry of Education of Japan.

REFERENCES

[1] A. George," Nested dissection of a regular finite element mesh", SIAM J. Numer. Anal., Vol.10, No.2 (1973) pp.345-363

[2] T. Taniguchi & K. Numata,"Fundamental study on the fill-in minimization problem", Memoirs of School of Eng., Okayama Univ., Vol.15-1 (1980) pp.101-110

[3] A. George,"An automatic one-way dissection algorithm for irregular finite element problems", Lecture Notes in Mathematics 630 (ed.G.A. Watson) (1977) pp.76-89