Memoirs of the School of Engineering, Okayama University, Vol. 16, No. 1, November 1981

Bandwidth Minimization Algorithm for Finite Element Mesh

Takeo TANIGUCHI*

(Received September 18, 1981)

Synopsis

Renumbering algorithms commonly in use for the band solver are generally applicable for any kind of linear equations, and, therefore, we may say that they cann't effectively utilize the characteristics of the finite element mesh. In this paper we investigate the characteristics of the finite element mesh systems, and introduce them into Taniguchi-Shiraishi Algorithm which already introduced some properties of FEM mesh systems. And through several numerical experiments it is proved that this improved algorithm is one of the fastest one.

1. Introduction

By the application of Finite Element Method we oftenly encounter to solve a large sparse set of linear equations

$$A x = b \qquad (1)$$

, and as its solver Band Matrix Method is most commonly used. But, since the efficiency of the solver (i.e. the execution time and memory) wholly depends on how we can decrease the half bandwidth of A in eq.(1), a number of renumbering algorithms have been proposed in the last decade[1,2,3,4]. But, we may say that all of them are proposed for general purpose but not only for the finite element models.

On the other hand, the conditions required for the renumbering algorithm are 1). less execution time and 2). better result. Then, it is hopeful that the introduction of the characteristics of the finite

* Department of Civil Engineering

element mesh systems may improve above two conditions, even if the new algorithm can be applied only for the finite element mesh.

For the general purpose renumbering algorithms only the connectivity relationship between vertices in a graph obtained from A matrix is used, and, then, for the finite element mesh what kind of additional informations are allowed to use? Since FEM is applied only for the problem with boundaries and the vertices on the boundaries are placed by the analyst, himself, they are easily distinguished from the residuals. That is, the matter that the vertices in the graph may be divided into two groups is the first information. Second information is that the finite element mesh system is rather systematic and simple comparing to the graph obtained from general linear equations, because the mesh patterns used in FEM have the restrictions of numerical and discretization errors.

Among these two additional informations the first one is already introduced in the renumbering method by Taniguchi and Shiraishi[5], and they proposed a quite different starategy for minimizing the bandwidth. Then, in this paper the second information is introduced in their algorithm and its improvement is tried. The result of this improvement is that the execution time decreases to about one half of original one and that the obtained half bandwidth is as good as the original one gives.

2. Taniguchi-Shiraishi Algorithm

The half bandwidth, HBW, of A matrix in eq.(1) is expressed as

$$HBW = \max_{i=1}^{n} (j - i) \tag{2}$$

, where j is the column number of the last non-zero element in the i-th row. If we use a graph obtained from A matrix, then

$$HBW = \max_{v_j \in \text{adj. } v_i} | j - i | \tag{3}$$

, where v_i and v_j are vertices labeled "i" and "j", respectively, and the relation of $v_j \in$ adj. v_i indicates that v_j is a member of vertices adjacent to v_i.

In order to minimize HBW, row and column permutations are neccessary for eq.(2), and vertex-renumbering for eq.(3). That is, the minimization requires numerous repetitions of above procedures.

But, according to [6], the minimum value of HBW is decided by one of the characteristics of the graph, itself.

Here, we define a term called "Level Structure" for the explanation of the property which determines min. HBW [4]. A level structure of a graph G(X,E) is a partition

$$L = \{L_0, L_1, L_2, \ldots, L_\ell\} \tag{4}$$

of the node set X such that

$$\text{adj.}(L_i) \subseteq L_{i-1} \cup L_{i+1}, \qquad 0 < i < \ell \tag{5}$$

$$\text{adj.}(L_0) \subseteq L_1, \quad \text{adj.}(L_\ell) \subseteq L_{\ell-1} \tag{6}$$

The number ℓ is called the length of the level structure, and the width w(L) of the level structure is defined by

$$w = \max \{|L_i| \,|\, L_i \in L\} \tag{7}$$

Then,

$$\text{HBW} \propto w \tag{8}$$

That is, since the width of the level structure determines the half bandwidth, how to minimize the width is almost equivalent to the minimization of HBW.

Cuthill-McKee algorithm which is the most popular method is as following: At first, select vertices which satisfy a condition required by the user, and construct as many level structures from each selected vertex as the number of selected vertices, and choose one level structure which gives the smallest width among them.

On the other hand, well-known Gibbs-Poole-Stockmeyer algorithm is as following: At first, find out two vertices which locate at both ends of the longitudinal axis of the graph, and construct two level structures from them, and reconstruct only one level structure by using them as its width is minimized.

Let's consider above two methods on a finite element model of a two-dimensional continuum like a plate. Since the condition of the starting vertex in Cuthill-McKee algorithm intends to find out appropriate vertices on the boundary, we may say that C-M Algorithm,too, constructs level structures from end of graph as G-P-S algorithm does. But, successive construction of L_i from L_{i-1} or L_{i+1} oftenly enlarges the width of the obtained level structure. Therefore, G-P-S algorithm has the step to reconstruct level structure. Anyhow, both of them aim to find out the width of the graph by successive construction of level structure from the end of the longitudinal axis of the graph.

Since it is evident from eq.(8) that the half bandwidth is determined by the width, we may directly search the width if possible.

Assume the level structure of a finite element graph obtained as

$$L = \{L_0, L_1, L_2, \ldots, L_\ell\}.$$

Then, most of L_i in L include, at least, two boundary vertices which are vertices located on the boundary. Therefore, from the definition of the width, the width of the level structure includes the maximum number of vertices among which, at least, two are the boundary vertices.

Let g(0) be the boundary vertices, and let's construct the level structure from g(0) as following;

$$g = \{g(0), g(1), g(2), \ldots, g(r), g(r+1)=\emptyset\} \qquad (9)$$

From a vertex in g(r), namely u_1, we reconstruct new level structure

$$g' = \{g'(0)=u_1, g'(1), g'(2), \ldots, g'(r')\} \qquad (10)$$

, where g'(r') is the first set of vertices which includes more than two boundary vertices, namely u_2 and u_3. Then, the shortest path connecting u_2, u_1 and u_3 in this ordering may locates at the widest portion of the graph, that is, it may coincide the width of the level structure which governs the minimum half bandwidth.

Above idea for searching the width of a level structure is directly introduced in the renumbering method by Taniguchi and Shiraishi as following. As the input data we prepare not only the connectivity between vertices but also additional data, that is, the number of boundary vertices g(0) which are numerically ordered clockwisely from an arbitrarily selected vertex, labeled "1", and the imaginary connectivity for vertices in other boundaries. This algorithm is in Fig.1.

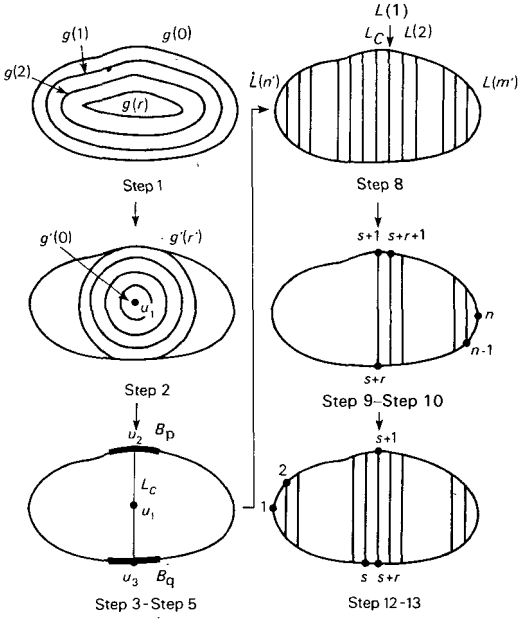

Fig.1 Taniguchi-Shiraishi Algorithm

ALGORITHM

Step 1. Construct a level structure from g(0) as shown in eq.(9).

Step 2. Reconstruct a level structure from u_1 which is selected among the vertices in g(r).

Step 3. Divide all boundary vertices in g'(r') into connected subgraphs, S_i, where i > 1, in general.

Step 4. Search two connected subgraphs, namely S_p and S_q, for which |b-b'| has the maximum value, where b and b' are the number of a vertex belonging to S_p and S_q, respectively.

Step 5. Select middle vertices, namely u_2 and u_3, from S_p and S_q, respectively.

Step 6. Search the shortest path connecting u_2 and u_3 through u_1. We call this shortest path Lc.

Step 7. Remove all imaginary connectivity for boundaries.

Step 8. For both sides of Lc, construct level structures from Lc;
{L'(n'), L'(n'-1), ... , L'(1), Lc, L(1), L(2), ..., L(m')}

Step 9. Renumbering for vertices in Lc.

Step 10 and 11. Renumbering for all vertices in one side of Lc.

Step 12 and 13. Renumbering for residual vertices.

This algorithm has following merits comparing to other algorithms: 1). This method doesn't require the process to find out the starting vertex which should locate on one end of the longitudinal axis of the graph. 2). The portion of the graph which decides the half bandwidth is, at first, searched in this algorithm, though others search it indirectly by the construction of level structure from a starting vertex.

3. Improvement of Taniguchi-Shiraishi Algorithm

3-1. On Searching Level Structure

As obvious from the algorithm given in the previous section, it mainly consists of the procedure of constructing level structure. Therefore, if the procedure to get level structure is improved, then it is expected that the execution-time is largely saved. For this improvement, we introduce the characteristics of the finite element mesh system mentioned in Section 1.

Typical finite element mesh system of 2-dimensional continuous media consists of 1) triangular or 2) square configuration element.

Assume that the successive i levels are already obtained from L_0 (see Fig.2 and 3), and we aim to search the next level, namely L_{i+1}. In T-S algorithm, all members for L_{i+1} are found out by using the

connectivity relations of each vertex in L_i. But, in the case of Fig. 2, all vertices in L_{i+1} may be found out by using about a half of the number of vertices in L_i which are denoted by o. Therefore, for getting the level structure we require only the connectivity of $\frac{1}{2}\Sigma|L_i|$ vertices though the original method requires $\Sigma|L_i|$ vertices. If the mesh system consists of only square element as shown in Fig.3, this new method requires only about one third of the original method.

Above two examples are the discussion for regular finite element systems, but the FE mesh system we encounter is generally irregular. Therefore, we have to consider the procedure of constructing the level structure for general triangular and also quadrilateral finite element mesh system, but we treat only the former case in this paper.

Assume that successive i levels of the level structure are already obtained. Therefore, we find all the member for L_{i+1} from L_i.

From the example of regular triangular mesh system, it is obvious that every second vertex in L_i are, at least, neccessary for searching all member of L_{i+1}. Hereafter, we call this procedure as ESVS (the abbrebiation of Every Second Vertex Searching Method). Since this procedure is invalid for irregular mesh system, we modify it so as to search all member of L_{i+1}.

Assume that $u \in L_i$ and $v \in L_{i+1}$, then $v \in$ adj.u from the definition of the level structure. If ESVS is applied for L_i, any vertex in L_i, namely u_α, is 1) used for searching L_{i+1} or 2) not used for it. In the former case, all vertices in L_{i+1} which are adjacent to u_α can be surely searched. Therefore, we may investigate only for the latter case.

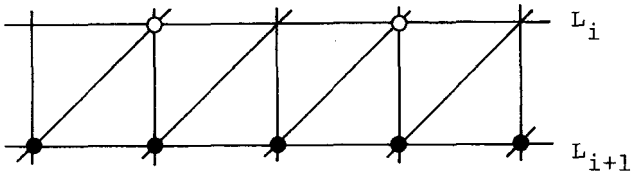

Fig.2 Triangular Finite Element Mesh

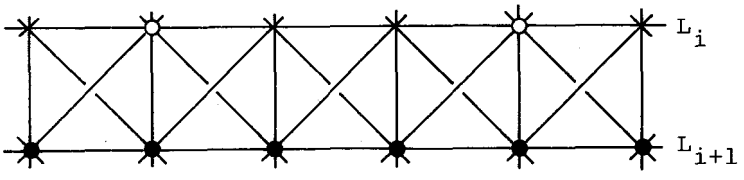

Fig.3 Quadrilateral Finite Element Mesh

Since the mesh system consists of only triangle elements, then at least two vertices, namely w's, which are adjacent to u_α and v_β must exist and they must locate in L_i and/or L_{i+1}. If u_α is the boundary vertex, then there must be only one w. Therefore, we have to consider following two cases. (See Fig.4)

1). $w \in L_i$

2). $w \in L_{i+1}$

Case 1) : If $w=u_{\alpha-1}$ or $u_{\alpha+1}$, then v_β is surely searched from w. If w is $u_{\alpha-2}$ or $u_{\alpha+2}$, then $u_{\alpha-1}$ has no connection with L_{i+1}. For this case it is obvious that these vertices should not be counted for searching every second vertex in Li.

Case 2) : The simplest case of this type is illustrated in Fig.4,too. $v_{\beta-1}$ and $v_{\beta+1}$ may be searched from $u_{\alpha-1}$ or $u_{\alpha+1}$, but v_β is not searched as far as u_α is not used. From this example we may conclude that if the number of vertices not yet ordered in L_{i+1} and adjacent to u_α in L_i is larger than two, we have to use the vertex, namely u_α , for searching L_{i+1}.

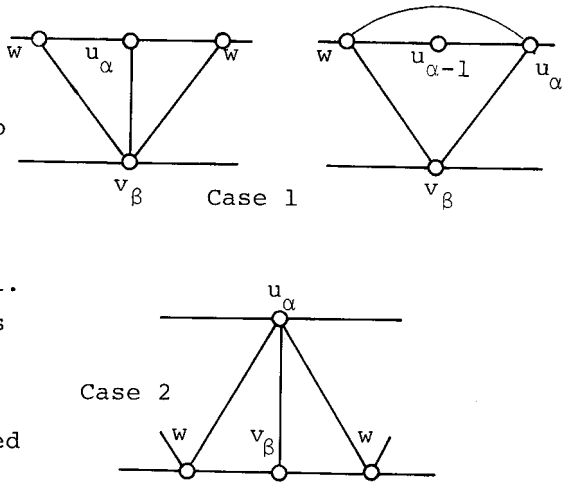

Fig.4 Two Cases

Let's call the number of residual vertices from u_α as R-deg.u_α. Then, above considerations are summarized as followings:

1. The vertex in L_i whose R-deg = 0 may not only be used for searching L_{i+1} but also not be counted for ESVS.
2. The vertex in L_i whose R-deg $\geq$ 2 must be selected for ESVS.

The treatment of a vertex in L_i whose R-deg = 1 is not yet discussed. But, as far as we take ESVS procedure, whether this type of vertices are selected for ESVS wholly depends on whether its number is odd or even. Are all vertices for L_{i+1} surely searched by above procedure? From above considerations;

1). v_β must be connected to only one vertex, namely u_α, in L_i,

2). therefore, u_α must have more than three connectivities between vertices in L_{i+1}, and

3). all vertices except v_β must be searched before v_β, and , therefore, R-deg of u_α = 1.

Fig.5 illustrates graphs satisfying above three conditions. Then, we conclude that some vertices for L_{i+1} may not be searched by above procedure based on ESVS.

In order to find out v_β subjecting above conditions, it is obvious that the strategy of ESVS must be modified.

Let's denote the number of vertices in L_i and L_{i+1} which is adjacent to u_α by $Deg.u_\alpha$. Then, the vertex v mentioned above is adjacent to u_α which satisfies $Deg.u_\alpha \geq 5$ and $R\text{-}deg.u_\alpha = 1$.

Above discussions about the midification of ESVS result in as followings:

1). ESVS may be used as the main strategy for setting level structure.

2). The vertices {u} of R-deg.u =0 should not be counted for ESVS.

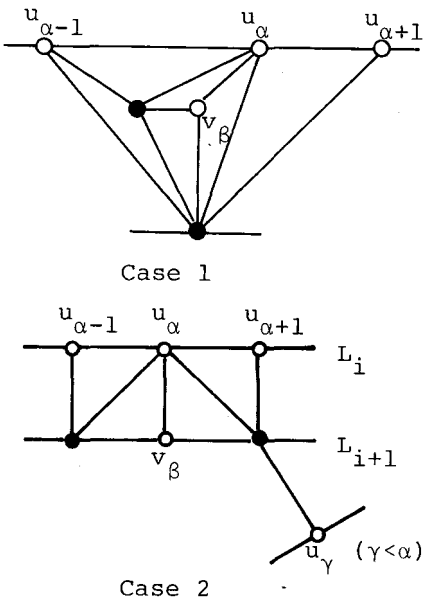

Fig.5 Exceptional Cases

3). The vertices {u} of $R\text{-}deg.u \geq 2$ must be additionally included in ESVS.

4). The vertices {u} satisfying $Deg.u \geq 5$ and $R\text{-}deg.u = 1$ must be additionally included in ESVS.

From these results we can propose new algorithm for setting level structure as following. Note that this algorithm is applicable only for the triangular finite element mesh system.

Here, we consider the stage of finding $L_{i+1} = \{v_1, v_2, \cdots, v_k, \cdots, v_\beta\}$ from $L_i = \{u_1, u_2, \cdots, u_j, \cdots, u_\alpha\}$. Then, this algorithm is repeated untill all vertices (or neccessary vertices) in a graph are searched. We set X=2 and Y=5 for triangular mesh system.

New Algorithm for Constructing Level Structure

Step 1. j=1. Find out $\{v \mid v \in adj.u_j\}$.

Step 2. c=0.

Step 3. j=j+1. If $j=\alpha$, find out $\{v \mid v \in adj.u_j\}$ and go to Step 10. Otherwise, go to Step 4.

Step 4. Calculate $Deg.u_j$ and $R\text{-}deg.u_j$.

Step 5. If $R\text{-}deg.u_j=0$, go to Step 3. Otherwise, go to Step 6.

Step 6. c=c+1. If c=X, find out $\{v \mid v \in adj.u_j\}$ and go to Step 2. Otherwise, go to Step 7.

Step 7. If R-deg.$u_j \geq X$, find out $\{v \mid v \in adj.u_j\}$ and go to Step 2. Otherwise, go to Step 8.

Step 8. If Deg.$u_j \geq Y$, go to Step 9. Otherwise, go to Step 3.

Step 9. Find out $\{v \mid v \in adj.u_j\}$ and go to Step 2.

Step 10. L_{i+1} is obtained.

3-2 Improvement of Residual Steps of the Original Algorithm

u_1 is arbitrarily selected in Step 2, but it should be selected among vertices with maximum degree, because the excentricity of the location of Lc should be excluded.

The procedure to get u_2 and u_3 in Step 4 and 5 is complicated, and it requires a lot of judgement if there exist a lot of subgraphs. We replace it by the comparison between middle vertices of every two subgraphs.

Now, we should discuss on the input-data for interior boundary vertices. It is obvious that the location of interior boundary prevents the construction of level structure in Step 1 and 2. Thus, imaginary connectivity relations are given for vertices in each interior boundary. If there exist σ interior boundaries and each contains ε vertices, then at least $\Sigma[\varepsilon_i(\varepsilon_i+1)/2]$ imaginary connections must be added to the actual connectivity relations. We replace it by following procedure. For each interior boundary, give an imaginary vertex which is connected to all vertices in the boundary, because this procedure requires only $\Sigma 2\varepsilon_i$ additional input data.

The steps from 8 to 13 are replaced by following procedure. After the setting of Lc all member in Lc are renumbered from "1", and all vertices in one side of Lc are successively renumbered as each level is obtained. After the last level of the side is obtained, the numbering is reversed, and successive renumbering is proceeded for the other side.

4. New Renumbering Algorithm

The input data are 1). the number of vertices N, 2). the number of outer boundary vertices NLE, 3) the interior boundary vertices, and 4).the connectivity relations between all vertices. Furthermore, we give imaginary vertex for each interior boundary and also give the imaginary connectivity between the imaginary vertex and all vertices on the boundary.

Algorithm

Step 1. Construct level structure from the outer boundary.

Step 2. Search a vertex in the last level which has the maximum degree among them, and denote it u_1.

Step 3. Construct level structure from u_1 till the boundary vertices are included for the first time.

Step 4. Divide all boundary vertices in the last level into connected subgraphs. If there exists only one subgraph, obtain one more level.

Step 5. Search the middle vertex for each subgraph, and we obtain a set of middle vertices $\{b_j\}$.

Step 6. Find out two vertices, namely b_x and b_y, which minimize $| |x-y|-NLE/2|$, and denote them u_2 and u_3, respectively.

Step 7. Search the shortest path, Lc, connecting u_2, u_1 and u_3, and give labels for them from "1".

Step 8. Search the first level set for one side of Lc, and give them the successive labels. Repeat the level setting and labeling procedure till all vertices on one side of the graph are labeled.

Step 9. Give the reverse renumbering to all vertices in the levels obtained above.

Step 10. Above two steps are repeated for the residuals of the graph, and the renumbering is completed.

Note that the new algorithm for constructing level structure given in Section 3-1 is used in Step 1, 3, 8, 9 and 10 of above algorithm.

For the evaluation of this algorithm, the author compared its results with those by Cuthill-McKee algorithm which is most commonly in use. This algorithm requires the starting vertex which satisfies following relation on the degree, D,

$$D \leq D_{min} + \frac{n}{m} D_{max} \qquad (11)$$

, where D_{min} and D_{max} are the minimum and the maximum degrees in the graph, respectively, and "n" and "m" are the two parameters which the user can determine. For our numerical experiments we take 1). n=0, and 2). n=1 and m=2, and they are called Method 1 and Method 2 , respectively. Among them, Method 1 is the fastest case for Cuthill -McKee algorithm, because the starting vertex must satisfy the minimum degree. Furthermore, the author modified Cuthill-McKee algorithm by adding the procedure of finding the starting vertex which is proposed

in Ref.[3]. We call this modified method as Method 3.

As the test examples three graphs presenting finite element mesh are selected, and the results are summarized in Table 1. And, the obtained renumbering results are shown in Fig's 6, 7 and 8.
From these numerical experiments following results are obtained:

1). The execution time required by the new algorithm is almost a linear function of the number of vertices, and it requires only as long execution time as Method 3 requires.

2). The half bandwidth obtained by the new algorithm is the minimum or near minimum value among them.

Since Method 3 is a modified method of Cuthill-McKee algorithm in order to avoid the repetition of setting level structures, it may be thought as one of the fastest algorithms. Therefore, this new algorithm proposed in this paper is also one of them.

		New Algorithm	Method 1	Method 2	Method 3
Ex.1	HBW	9	9	10	10
	T(sec)	0.011	0.350	0.020	0.008
Ex.2	HBW	12	13	18	16
	T(sec)	0.024	3.709	0.030	0.025
Ex.3	HBW	22	21	24	25
	T(sec)	0.050	44.832	0.180	0.060

Table 1. Comparison of Results

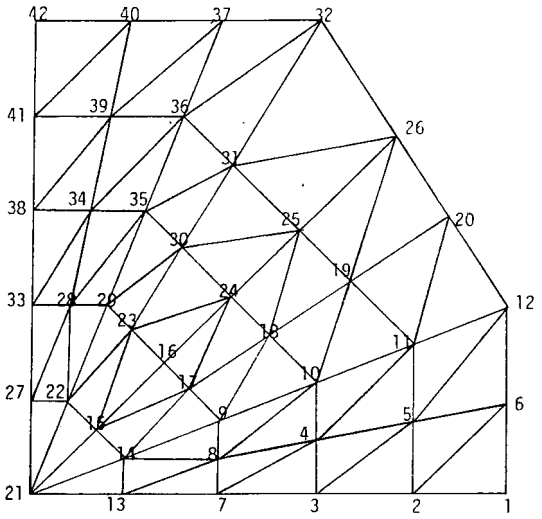

Fig.6 Example 1
(42 Vertices)

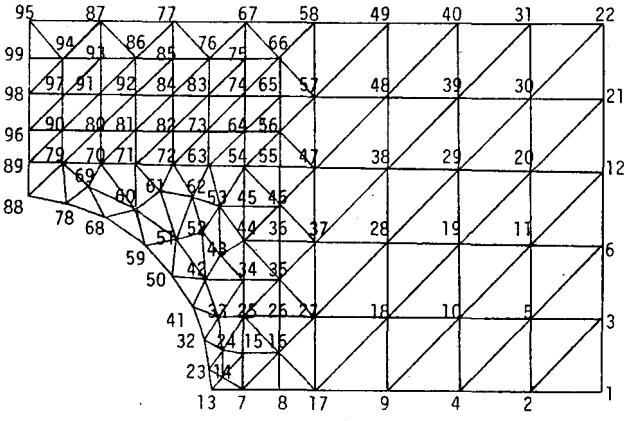

Fig.7 Example 2 (99 Vertices)

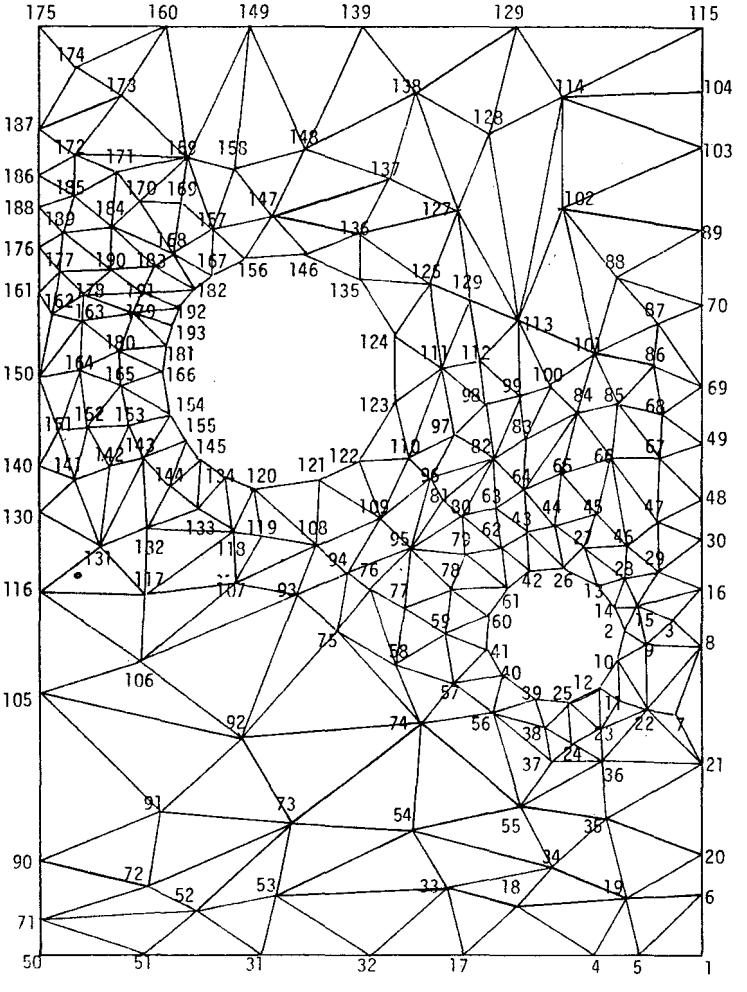

Fig.8 Example 3 (193 Vertices)

5. Concluding Remarks

From the numerical experiments of the new renumbering algorithm in Section 4 it becomes obvious that the execution time depends almost on how long execution time the procedure of setting level structure requires. Therefore, it is expected that for quadrilateral finite element mesh system we can design a new renumbering algorithm by modifying Taniguchi-Shiraishi algorithm which requires about one third of the execution time by original one. But, since the idea presented in this paper is invalid for the system like 5-point difference method, more saving of the execution time for it is impossible.

Acknowledgement

The author thanks Dr. J.K. Reid of A.E.R.E., Harwell, England, for the suggestion of the idea of how to obtain level structures. The assistance of Mr. T. Kimoto is also gratefully acknowledged.

References

[1]. E.Cuthill & J.McKee,"Reducing the Bandwidth of Sparse Symmetric Matrices", Proc. of ACM National Conference,(1969), 157-172

[2]. E.Cuthill,"Several Strategies for Reducing the Bandwidth of Matrices", Sparse Matrices and Their Applications(ed. by D.J.Rose &R.A.Willoughby),(1972), 157-166

[3]. N.E.Gibbs, W.G.Poole & P.K.Stockmeyer,"An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix", SIAM J. Numer. Anal., 13 (1976), 236-250

[4]. J.A.George,"Solution of Linear Systems of Equations: Direct Methods for Finite Element Problems", Lecture Notes in Mathemetics 572(ed.by V.A.Baker) (1976), 53-101

[5]. T.Taniguchi & N.Shiraishi,"New Renumbering Algorithm for Minimizing the Bandwidth of Sparse Matrices", Advances in Eng. Software, 2(4) (1980), 173-179

[6]. I.konishi, N.Shiraishi & T.Taniguchi,"Reducing the Bandwidth of Structural Stiffness Matrices", J. Struct. Mech. 4(2) (1976), 197-226