SMiRT-12 / K. Kussmaul (Editor)

B07/1

PARALLEL CAE SYSTEM FOR LARGE-SCALE 3-D FINITE ELEMENT ANALYSES

G. Yagawa, H. Kawai and S. Yoshimura

University of Tokyo, Tokyo, Japan

ABSTRACT

This paper describes a new pre- and post-processing system for the automation of large-scale 3D finite element analyses. In the pre-processing stage, a geometry model to be analyzed is defined by a user through an interactive operation with a 3D graphics editor. The analysis model is constructed by adding analysis conditions and a mesh refinement information to the geometry model. The mesh refinement information, i.e. a nodal density distribution over the whole analysis domain is initially defined by superposing several locally optimum nodal patterns stored in the nodal pattern database of the system. Nodes and tetrahedral elements are generated using some computational geometry techniques whose processing speed is almost proportional to the total number of nodes. In the post-processing stage, scalar and vector values are evaluated at arbitrary points in the analysis domain, and displayed as equi-contours, vector lines, iso-surfaces, particle plots and realtime animation by means of scientific visualization techniques. The present system is also capable of mesh optimization. A posteriori error distribution over the whole analysis domain is obtained based on the simple error estimator proposed by Zienkiewicz and Zhu. The nodal density distribution to be used for mesh generation is optimized referring the obtained error distribution. Finally nodes and tetrahedral elements are re-generated. The present remeshing method is one of the global hr-version mesh adaptation methods. To deal with large-scale 3D finite element analyses in a reasonable computational time and memory requirement, a distributed / parallel processing technique is applied to some part of the present system. Fundamental performances of the present system are clearly demonstrated through 3D thermal conduction analyses.

1. INTRODUCTION

A utilization of a massively parallel computer or multiple supercomputers connected among others through a high speed network enables us to perform large-scale 3D finite element analyses over millions of degrees of freedoms (Yagawa et al. 1991, 1992a). To deal with such large-scale analyses in a reasonable computational time and memory requirement, it is strongly necessary to develop a fully automated, quickly responding and user-friendly CAE system.

In the present study, a new CAE system operating on a network of workstations is developed based on the following technologies :

(1) Fully automatic mesh generation with a fine contol of nodal density distribution (Yagawa et al., 1992b, 1992c)

(2) Automatic posteriori error estimation (Zienkiewicz and Zhu, 1987, 1991) and global hr-version mesh adaptation

(3) Distributed / parallel post-processing

In the subsequent sections, each technique is explained in detail after describing the system configuration. Finally, fundamental performances of the developed system are clearly demonstrated through 3D thermal conduction analyses.

2. SYSTEM CONFIGURATION

A schematic configuration of the present CAE system is shown in Fig. 1. The system is composed of the following five modules, i.e. a geometry modeler, a mesh generator, a file I/O converter, a physical quantity evaluator, and a scientific visualizer.

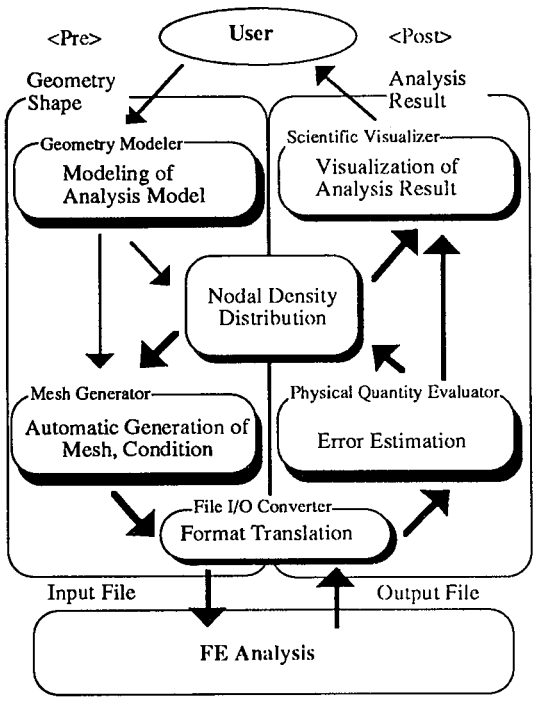


Fig.1 System configuration

The geometry modeler builds a geometry model through an interactive operation with a user. A mouse and a key board are input devices here (Bier, 1990, Cunningham, 1992). An analysis model is composed of the geometry model, analysis conditions and a mesh refinement information. The geometry model employed here is a combination of the following two geometry representations, i.e. constructive solid geometry (CSG) and boundary representation (B-rep) (Miller, 1989, Requicha and Rossignac, 1992). Analysis conditions are expressed in terms of polynomials of coordinates, and attached on any portions of vertices, edges, faces and volume of the geometry model. The mesh refinement information is expressed as a superposition of several locally-optimum nodal density distributions which are chosen from the nodal pattern database of the system. The superposition is carried out by a logical OR operation using a fuzzy knowledge processing technique (Zadeh, 1968, Yagawa et al., 1992b, 1992c).

The mesh generator fully automatically creates a finite element mesh for the given geometry data and the mesh refinement information. Referring the nodal density distribution defined over a whole analysis domain, nodes are first generated by the bucketing method whose processing speed is proportional to the total number of nodes (Asano, 1985, Yagawa et al., 1992c). Then tetrahedral elements are generated using the Delaunay triangulation (Bowyer, 1981, Watson, 1981). In the process, the system extracts an information of mesh surface, which is later utilized in the scientific visualizer module.

The file I/O converter is a kind of interface program between the present CAE system and any finite element program. Referring a finite element model, the converter makes an input file specialized for a finite element program, and reads analysis results from an output file of the program.

The physical quantity evaluator calculates fundamental variables, their partial derivatives *w.r.t.* coordinates and their integrals over element or domain. A posteriori error estimation is also performed in the module. The error estimation results in an optimized mesh refinement information, i.e. an optimized

nodal density distribution. Based on the distribution, a mesh is re-generated. Since the mesh is completely re-generated, being independent of the initially used mesh, it is named here a global hr-version mesh adaptation. In the current version of the system, the Z-Z error estimator proposed by Zienkiewicz and Zhu (Zienkiewicz and Zhu, 1987, 1991) is employed. It should be noted here that any other error estimators can be very easily implemented in the present CAE system because of the accompanied powerful post-processing techniques. The module can operate on a local area network (LAN) of workstations, which will be described later in detail.

The scientific visualizer creates images of analysis results on a computer display through an interactive operation with a user. For example, it visualizes contour plots, vector arrow plots, deformation plots, iso-value surface plots, and particle track (Lorensen and Cline, 1987, Gallagher and Nagtegaal, 1989, Bala, 1991, Dannelongue, 1991, Farrell et al., 1991, Gallagher, 1991, Koyamada and Nishino, 1991, Thalmann, 1991, Gallagher and Selker, 1992). Quantities to be visualized are calculated by the physical quantity evaluator.

The processes described above form a loop, which is repeated until an error distribution meets a certain criterion. A user can watch each process on the loop through the scientific visualizer, and stop it at any point.

3. AUTOMATED PRE-PROCESS

Here we describe the automatic mesh generation technique used in the present system. First, we introduce a data structure of an analysis model and a nodal density distribution. Then the algorithms for node generation, element generation and analysis condition attachment are described.

3.1 Data Structure of Analysis Model

The analysis model utilized in the present system is composed of a geometry model, analysis conditions and a mesh refinement information, i.e. a nodal density distribution. The geometry model has to have the two functions of IN / OUT detection and boundary evaluation, which are frequently called by the mesh generator as described later. The functions have to operate as fast as possible. In the current version of the system, the combination of the CSG and B-rep models is utilized as a geometry model (Miller, 1989, Requicha and Rossignac, 1992). The IN / OUT detection is carried out based on the CSG model, while the boundary evaluation is carried out based on the B-rep model.

Analysis conditions expressed in terms of polynomials of coordinates are attached on any portions of the geometry model, i.e. vertices, edges, faces and volume.

A nodal density distribution over a whole analysis domain is attached on any portions of the geometry model. A nodal pattern represents a locally-optimum distribution of nodal density as shown in Fig. 2. A distribution of nodal density over a whole analysis domain is an assembly of several nodal patterns. Since the ranges of the nodal patterns may overlap each other, a logical OR operation of the fuzzy knowledge processing technique is utilized to assemble the nodal patterns (Zadeh, 1968, Yagawa et al., 1992b, 1992c). Assuming that $\mu A(x_p, y_p)$ and $\mu B(x_p, y_p)$ are nodal density functions of two patterns A and B, respectively, a greater value between μ_A and μ_B is adopted as the final value of nodal density at the point $p(x_p, y_p)$. In practice, a 3D grid which is defined, being independent of the mesh used, is utilized to store the nodal density distribution. A nodal density value, i.e. a scalar value is assigned on each grid point. A nodal density value at any location within a hexahedral cell is calculated through a tri-linear interpolation among eight grid points of the cell. Calculation time is independent of the grid size and the complexity of the nodal density distribution. The grid size is chosen considering the complexity of the geometry model used.

According to our experience, a grid size ranging from 20 x 20 x 20 to 50 x 50 x 50 seems appropriate to represent usual analysis models. In this case, a memory requirement to store the nodal density distribution is not more than 1 Mbytes. The assembly process of multiple nodal patterns is performed by evaluating a nodal density value at each grid point.

The data structure described above is constructed in the geometry modeler module. The input data required to generate the analysis model is relatively small, and does not depend on the scale of the analysis model.

3.2 Algorithm of Mesh Generation

A mesh is generated in the mesh generator module. The process is composed of the following five subprocesses, i.e. node generation, element generation, element shape smoothing, re-numbering, and analysis condition attachment.

Node generation is one of time consuming processes in a whole automatic mesh generation process. Here we adopt the following procedure for node generation. Nodes are generated so as to meet the complicated distribution of nodal density over a whole analysis domain. The most disadvantageous feature of conventional node generation techniques is that node generation time is in general proportional to the order of $O(n^2)$, where n is the total number of nodes. Such a slow node generation speed may cause a serious problem in practical mesh generation for 3D solid geometries. In the present study, we adopt the bucketing method for fast node generation (Asano, 1985, Yagawa et al., 1992c). Fig. 3 shows the fundamental principle of the bucketing method, taking a 2D mesh generation as an example without any loss of generality. Let us assume that a distribution of nodal density over a whole analysis domain is already given as shown in Fig. 3(a). At first, we define a rectangular enveloping the analysis domain as shown in Fig. 3(b). In 3D cases, a box is utilized to envelop an analysis domain. Next, we divide the rectangular into a number of small sub-rectangulars, each of which is named "Bucket". Nodes are generated bucket by bucket. Taking one bucket, let us explain the procedure of the present node generation.

At first, a number of candidate nodes are prepared as shown in the l.h.s. of Fig. 3(c). The distance of two neighboring candidate nodes is set to be smaller than the minimum distance of nodes to be placed in the relevant bucket. In the present study, the ratio of these two distances is chosen to be roughly one sixth through trials and errors. Next candidate nodes are pick up one by one, starting from the left-bottom corner of the bucket, and are put into the bucket. A candidate node is adopted as a node to be generated in the bucket when the following two criteria are satisfied :

(a) The candidate node is inside the analysis domain. (IN / OUT check)
(b) The distance of the candidate node and the nearest node already generated in the bucket meets the nodal density at that point to some extent.

Among several algorithms for the IN / OUT check of the criteria (a), the following criterion is employed here because of simplicity in implementation :

"A straight line passing any node inside a closed domain crosses the domain boundary odd times."

This criterion is easily examined using the nodal coordinates and the geometric data of the analysis domain. It should be noted here that those nodes already generated in the neighboring buckets are also examined for

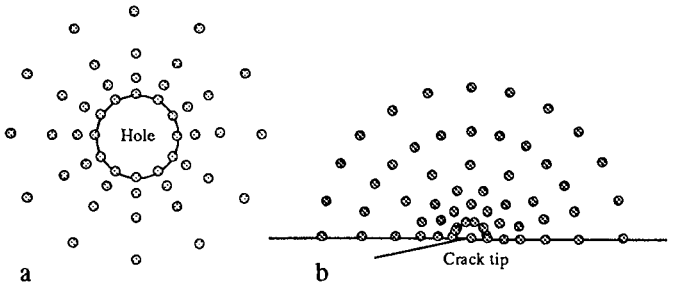
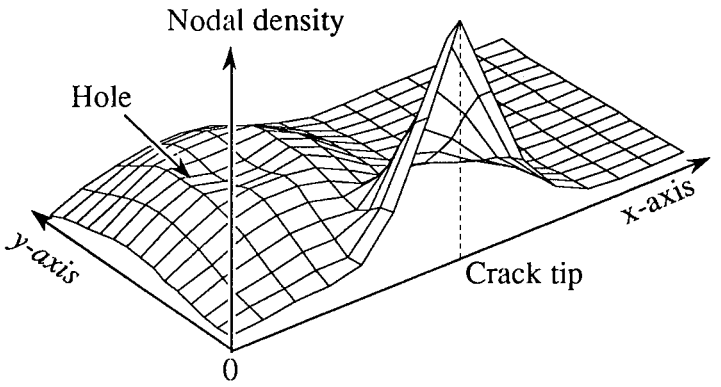
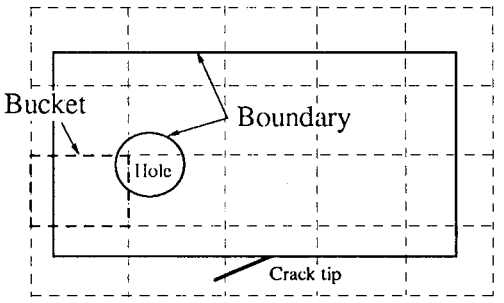


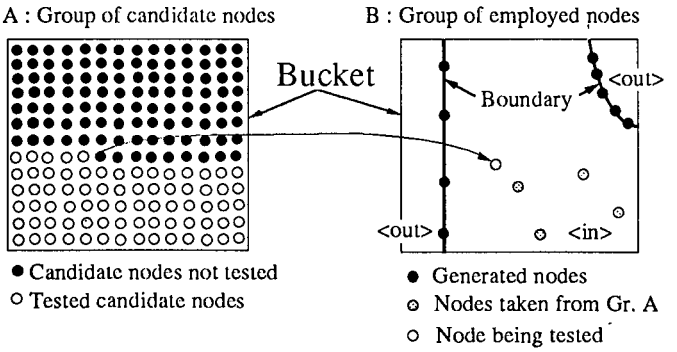
Fig.2 An example of nodal pattern
(a) For a hole
(b) For a crack-tip



(a) Example of nodal density distribution



(b) Example of bucket decomposition



(c) Node generation in one of buckets

Fig. 3 Node generation based on bucketing method(2D case)

the criterion (b) when a candidate node is possibly generated near the border of the relevant bucket. Owing to the present bucketing method, the number of examinations of whether nodes are generated or not can be reduced significantly, and then a node generation speed is remained to be proportional to O(n). It should be also noted that in three-dimensional solid geometries nodes are generated in the following order ; vertices, edges, faces and domain.

After node generation, tetrahedral elements are generated using the Delaunay triangulation (Bowyer, 1981, Watson, 1981). A smoothing operation of element shape and the re-numbering operation of nodal numbers are then performed. The processes mentioned above are performed fully automatically, and their processing speeds are proportional to the scale of the analysis, i.e. the total number of nodes.

Finally, analysis conditions are attached on each node or element. It should be noted here that each node and element knows what geometry portion it is generated from.

4. AUTOMATED ERROR ESTIMATION AND GLOBAL MESH ADAPTATION

Here we describe an automated posteriori error estimation, and an accompanied mesh refinement technique using a mesh refinement information and the computational geometries.

4.1 Global Mesh Adaptation

As described previously, a mesh is initially generated by choosing several nodal patterns and superposing them. After the finite element analysis using the initial mesh, an error distribution over a whole analysis domain is obtained from a posteriori error estimation. When a user specifies an allowable error limit, say, 5 %, one can calculate a mesh refinement factor by dividing the calculated error norm by the allowable error limit. Then the nodal density distribution is re-constructed by multiplying the mesh refinement factor to a value of nodal density at each grid point. The modified distribution of nodal density is used to generate a new mesh. This process is one of the hr-version mesh adaptation methods. Since the mesh is re-generated, irrespective of a prevously used mesh, the present mesh adaptation process is named the global hr-version mesh adaptation. Its algorithm is summarized in Fig. 4.

4.2. Z-Z Error Estimator

Obviously the present mesh adaptation process can employ any kind of error estimators. As one of examples, the Z-Z error estimator proposed by Zienkiewicz and Zhu (Zienkiewicz and Zhu, 1987, 1991) is employed in the present system. The error estimator which can be evaluated element by element involves two derivatives. One is directly calculated from analysis results, while the other should be calculated from an exact solution. Derivatives evaluated using a higher order interpolation function are utilized as the better approximation of the exact solution. The better approximation of derivatives can be calculated by projecting the derivatives calculated at integral points onto nodes. In the process, an averaging method of nodal values embedded in the scientific visualizer module is used.

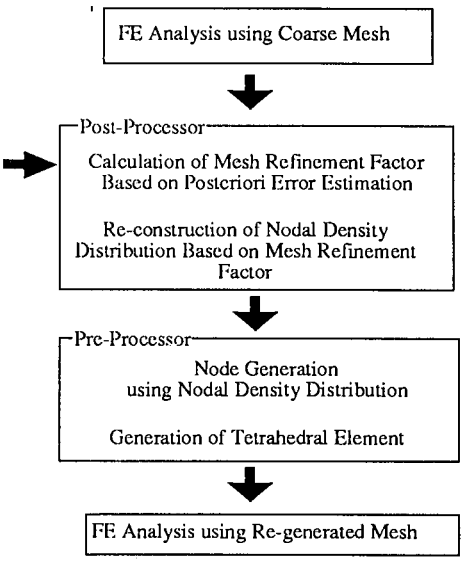


Fig. 4 Flow of global mesh adaptation process

5. DISTRIBUTED / PARALLEL PROCESSING

5.1. Finite Element Post-processor for Large-Scale 3D Analysis

Suppose that there is a huge amount of analysis results derived from a large-scale 3D finite element analysis with one million degrees of freedoms. If the mesh is displayed on a screen, one can not distinguish each element because the mesh subdivion is too fine. Such a situation has already been popular in the field of the finite difference analyses using more than one million grid points. In such large-scale analyses, scientific visualization techniques are inevitable. Well-known scientific visualization techniques for the finite element method are listed below.

(1) Surface visualization

It includes contour plot for scalar quantity, and arrow plot and deformation plot for vector quantity on mesh surface or a cross section of mesh along any plane.

(2) Volume visualization (Lorensen and Cline, 1987, Gallagher and Nagtegaal, 1989, Bala, 1991, Dannelongue, 1991, Farrell et al., 1991, Gallagher, 1991, Koyamada and Nishino, 1991, Thalmann, 1991, Gallagher and Selker, 1992)

It includes iso-value surface plot for scalar quantity, and particle track for vector quantity. The volume visualization technique is more important in 3D analyses. It is required to interactively visualize analysis results using a high speed graphics terminal or a graphics workstation. Otherwise, a user has to wait several minutes for every operations.

When a graphics workstation is available and an interactive visualization can be performed, the following two problems have to be solved. One is the reduction of a computational cost required for the extraction of the geometry data to be visualized. The other is the shortage of memory capacity of a graphics workstation. To overcome the problems, we propose here a utilization of multiple workstations connected among others through a local area network (LAN).

5.2 Architecture of Distributed / Parallel Post-processor

The most time consuming and memory consuming process among the whole pre- and post- processes is the extraction of geometry data to be visualized. In the present system, this process is involved in the physical quantity evaluator module. Then a distributed and parallel processing technique is applied to the module.

By nature, most calculations of the finite element analyses can be performed element by element. It is also true in a post-process. Thus, a domain decomposition approach is adopted. It should be noted here that it is not necessary to apply a parallel processing technique to the visualization module at an application program level because of the following three reasons. First, such an acceleration technique is already realized inside a graphics library or a graphics hardware by each workstation vender. Second, a distributed and parallel processing technique using a local area network (LAN) is not suitable for an interactive operation like a real time animation (Thalmann, D., 1991) and a direct manipulation of objects (Bier, 1990, Cunningham, 1992). Third, an amount of the geometry data to be visualized is not proportional to the scale of analysis results. If appropriate visualization techniques are employed, an amount of the geometry data is proportional to the scale of resolution of computer screen.

An architecture of the developed distributed / parallel module is shown in Fig. 5. One graphics workstation and several popular workstations are connected among others through a LAN. The physical quantity evaluator module is composed of a master process and several slave processes. The master process

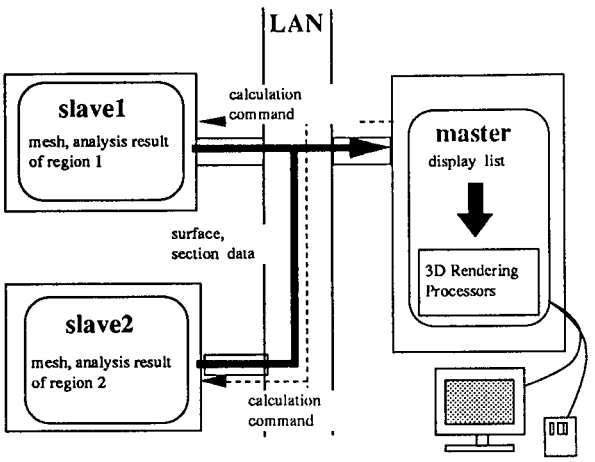


Fig. 5 A schematic view of distributed and parallel system implemented on LAN

is placed in the graphics workstation, while the slave processes are distributed among the workstations. A mesh and analysis results are decomposed to the same number of subdomains as that of the slave processes. The data of the subdomains are distributed among the slave processes, and dealt with independently. When any visualization command is issued, each slave process accesses all the elements of the assigned subdomain, and extracts the geometry data to be visualized. This calculation is performed in parallel. The geometry data of each subdomain is sent to the master process, and visualized on the screen of the graphics workstation.

As for a realtime animation of analysis results of a time dependent problem, the geometry data over all time steps are collected in the slave processes, and sent to the master process at once. The master process displays them in a real time manner.

5.3 Scientific Visualization

In the current version of the system, the present system can deal with some surface visualization techniques such as contour plot, vector arrow plot, and deformation plot on mesh surface and cross section, and some volume visualization techniques such as iso-value surface plot and particle track.

As for contour plot on mesh surface, the mesh surface data which are extracted by the mesh generator module beforehand are loaded to the master process. Scalar quantity data over a whole analysis domain are distributed among the slave processes. All the scalar values on nodes located on the mesh surface are sent to the master process. They are visualized referring only the geometry data on the master process.

As for iso-value surface plot, the slave process checks whether the range of scalar value of each element of the assigned subdomain includes the given scalar value. If it is true, a fragment of iso-value surface is extracted by the Marching Cube method (Lorensen and Cline, 1987, Gallagher and Nagtegaal, 1989, Gallagher, 1991). The fragments extracted are sent to the master process, and visualized as an iso-value surface.

The parallel efficiency of the present physical quantity evaluator module may not be so high because each subdomain is assigned to the slave process in a static manner. In other words, no dymamic allocation of a whole workload is carried out during the process. Even though, the present system can deal with a huge amount of analysis results of large-scale 3D finite element analyses owing to a utilization of a huge memory space distributed among multiple workstations.

6. IMPLEMENTATION

The developed system is named as "WH". The WH is built with C++ considering object-oriented programming. The WH is composed of a number of original class libraries, i.e. basic data structure and mathematical operation, the user interface, the mesh generator and the finite element program. The WH can operate on any Unix-based workstations with the X Window system (X11R3 or later version), and uses their

own specific graphics library and toolkit. In the current version, the present system operates on Sun XGL, HP starbase, Silicon Graphics GL and apollo DOMAIN GMR-3D.

7. EXAMPLES

Here several numerical examples are demonstrated. First, some analysis results of a heat conduction problem are visualized. A mesh and iso-value surfaces of temperature are shown in Figs. 6 and 7, respectively. The scale of the present analysis model is about 15,000 nodes and 75,000 four-noded tetrahedral elements. The iso-surfaces of temperature are extracted in three seconds using four workstations, i.e. three Sun SPARCstation ELCs and one Sun SPARCstation 2GS.

Secondly, a fully automatic mesh adaptive process is shown through a heat conduction analysis. A solid model is created and temperature boundary conditions are attached to the analysis model. An initial mesh is generated without any special nodal patterns, namely, a flat nodal density as shown in Fig. 8. A heat conduction analysis is performed using the initial mesh. Some analysis results are shown in Fig. 9. An error is estimated, and then the nodal density distribution is modified. An iso-surface of the mesh refinement factor (more than twice) is shown in Fig. 10, and a newly generated mesh is shown in Fig. 11.

8. CONCLUSIONS

The parallel CAE system for the automation of large-scale 3D finite element analyses is developed. In the system, solid modeling of geometry and analysis condition data, fully automatic mesh generation for large-scale FEM analysis, posteriori error estimation and remeshing, three dimensional scientific visualization are integrated, and some time consuming and memory consuming modules are implemented on a local area network of workstations.

The fundamental performances of the present system are clearly demonstrated through its application to three-dimensional thermal conduction analyses.

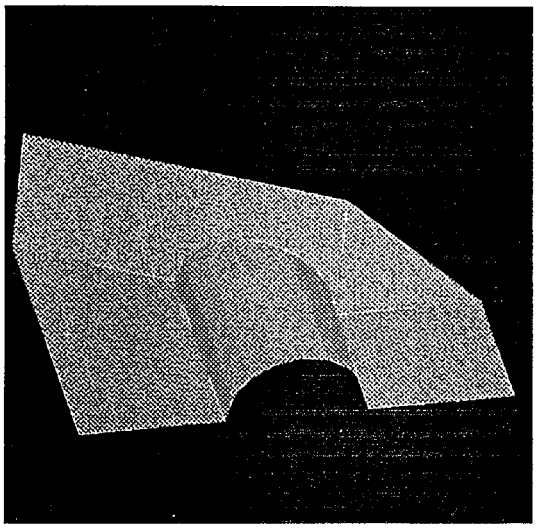


Fig. 6 Mesh subdivision(Example 1)

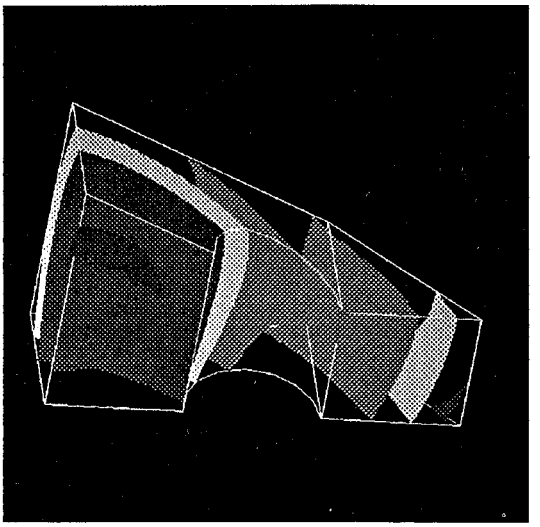


Fig.7 Example of iso-value surfaces (Temperature distribution)

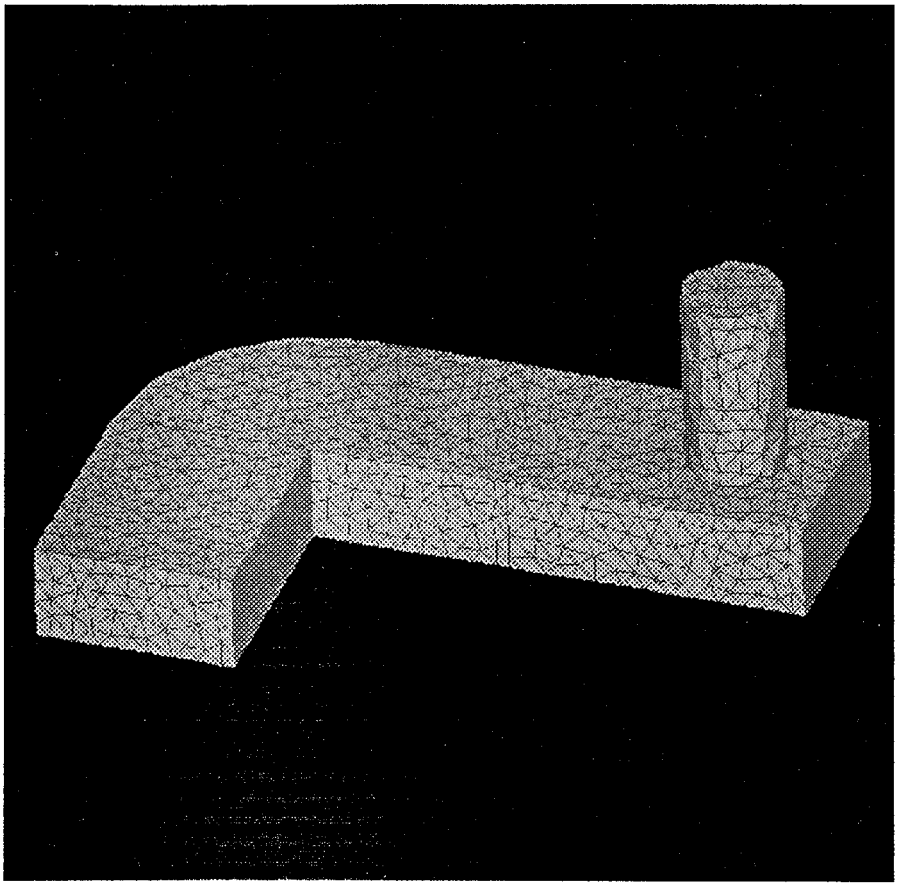


Fig. 8 Mesh subdivision(Example 2)

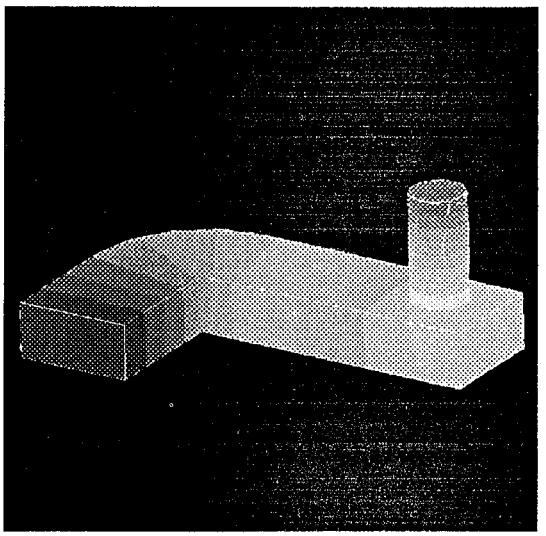


Fig. 9 Surface contour of temperature calculated using the mesh shown in Fig.8

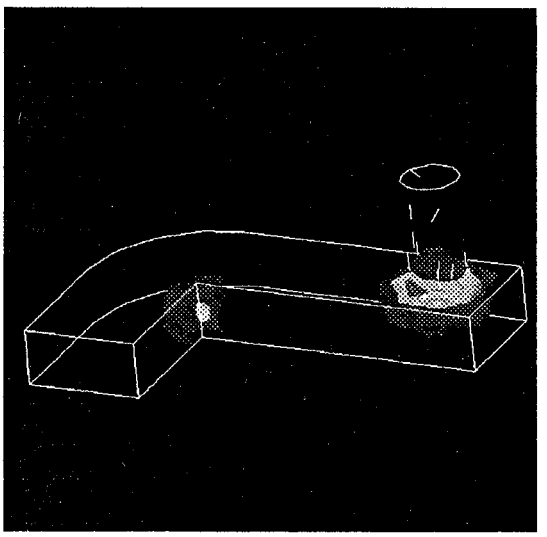


Fig. 10 Iso-surfaces of mesh refinement factor

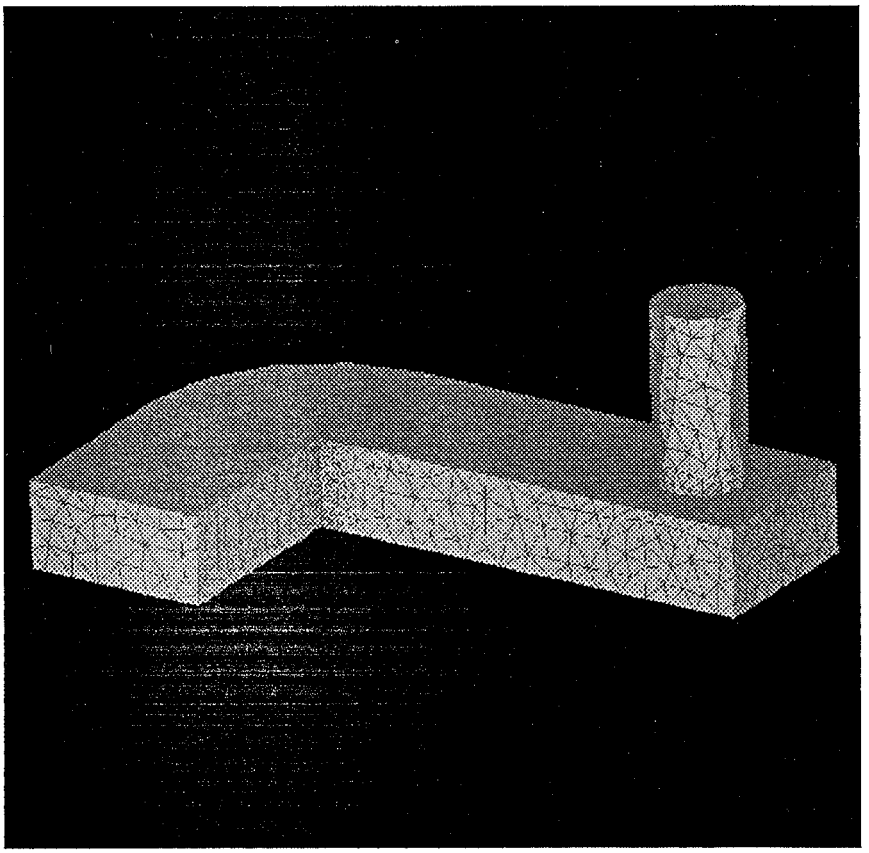


Fig. 11 Refined mesh

Reference

Asano, T. (1985). Practical Use of Bucketing Techniques in Computational Geometry, Computational Geometry, pp.153-195, North-Holland.

Bala, G. P. (1991). FEMvis : An Interactive Visualization Tool for Mechanical Analysis, IBM J. Res. Develop., 35, pp. 4-11.

Bier, E. A. (1990). Snap-dragging in Three Dimensions, ACM SIGGRAPH, Comp. Graph., 24, pp. 193-204.

Bowyer, A. (1981). Computing Dirichlet Tessellations, The Computer Journal, 24, pp. 162-166.

Cunningham, S., Craighill, N. K., Fong, M. W. and Brown, J. R. (1992). Computer Graphics Using Object-oriented Programming, John Wiley & Sons.

Dannelongue, H. H. (1991). Volumetric FE Results Interpretation, ASME PVP-209, Comp.Graph. Data Manage., pp. 3-9.

Farrell, E. J., Appino, P. A., Foty, D. P. and Linton, T. D. Jr. (1991). Visual Interpretation of Multidimensional Compu tations and Transister Design, IBM J. Res. Develop., 35, pp. 26-43.

Gallagher, R. S. and Nagtegaal, J. C. (1989). An Efficient 3-D Visualization Technique for Finite Element Models and Other Coarse Volumes, ACM SIGGRAPH Comp. Graph., 23, pp. 185-194.

Gallagher, R. S. (1991). Direct Generation of Isosurfaces from Finite Element Models, ASME PVP-209, Comp. Graph. Data Manage., pp. 21-26.

Gallagher, R. S. and Selker, P. J. (1992). Three-dimensional Volume Visualization in FE Analysis, Mech. Engng., 5, pp. 54-57.

Koyamada, K. and Nishio, T. (1991). Volume Visualization of 3D Finite Element Method Results, IBM J. Res. Develop., 35, pp. 12-25.

Lorensen, W. E. and Cline, H. E. (1987). Marching Cubes : A High Resolution 3D Surface Construction Algorithm, ACM SIGGRAPH Comp. Graph., 21, pp. 163-169.

Miller, J. R. (1989). Architectural Issues in Solid Modelers, IEEE Comp. Graph. Appl., 9, pp. 72-87.

Requicha, A. A. G. and Rossignac, J. R. (1992). Solid Modeling and beyond, IEEE Comp. Graph. Appl., 9, pp. 31-44.

Thalmann, D. (1990). Scientific Visualization and Graphics Simulation, John Wiley & Sons.

Watson, D. F. (1981). Computing the n-imensional Delaunay Tessellation with Application to Voronoi Polytopes, The Computer Journal, 24, pp. 167-172.

Yagawa, G., Soneda, N. and Yoshimura, S. (1991). A Large Scale Finite Element Analysis Using Domain Decomposition Method on a Parallel Computer, Comp. Struct., 38, pp. 615-625.

Yagawa, G., Yoshimura, S., Yoshioka, A. and Soneda, N. (1992a). Large Scale Elastic-Plastic Finite Element Analysis Using Domain Decomposition Method and Computer Network, Proc. 3rd Int. Conf. Computational Plasticity, Barcelona, pp. 2161-2183.

Yagawa, G., Yoshimura, S., Soneda, N. and Nakao, K. (1992b). Automatic Two- and Three-dimensional Mesh Generation Based on Fuzzy Knowledge Processing, Comp. Mech., 9, pp. 333-346.

Yagawa. G., Yoshimura, S., Nakao, K. and Tsuru, D. (1992c). A Large Scale Automatic Mesh Generation Method Using Fuzzy Knowledge Processing and Computational Geometry : Its Application to 2D Plane, 3D Shell and 3D Solid, Trans. Japan Soc. Mech. Engineers, 58A, pp. 243-251 (in Japanese).

Zadeh, L. A. (1968). Fuzzy Algorithms, Information and Control, 12, pp. 94-102.

Zienkiewicz, O. C. and Zhu, J. Z. (1987). A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis, Int. J. Numer. Meth. Engng., 24, pp. 337-357.

Zienkiewicz, O. C. and Zhu, J. Z. (1991). Adaptivity and Mesh Generation, Int. J. Numer. Meth. Engng., 32, pp. 783-810.