This is to certify that the
thesis entitled

Modular Finite Element Modeling By Distributed Internet
Engineering Design Agents

presented by

Umar Farooq

has been accepted towards fulfillment
of the requirements for the

M. S degree in Mechanical Engineering

Clark Radcliffe
Major Professor's Signature

Dec 11, 2002
Date

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE

MODULAR FINITE ELEMENT MODELING BY DISTRIBUTED INTERNET ENGINEERING DESIGN AGENTS

By

Umar Farooq

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

2002

ABSTRACT

MODULAR FINITE ELEMENT MODELING BY DISTRIBUTED INTERNET ENGINEERING DESIGN AGENTS

By

Umar Farooq

Finite Element Analysis of structural assemblies using distributed Internet Engineering Design Agents (i-EDA) is presented here. Finite Element modeling of distributed models in the past has been a complicated issue. Typically, the assembly of these structures has required global reformulation of the Finite Element equations for solution and this reformulation consumes time and prevents efficient development of large models. Structural models with incompatible Finite Element meshing are assembled here using of the Modular Modeling Method (MMM). The method presented here does not require reformulation and allows efficient distributed model assembly and solution.

The MMM is a power-based systematic technique that eliminates global reformulation of model equations. The method uses displacement and work constraints to assemble the components at connections. This method is implemented through a set of Internet Engineering Design Agents (*i*-EDA) to assemble the Finite Element Models from models provided by component subsystem agents. The structural analysis of Plate Assembly of two triangular plates is used as an example and 4 cases presented. Models of the plate component and an assembly of these two plates are published as software agents on the worldwide Internet.

TABLE OF CONTENTS

List of Tables iv

List of Figures v

Introduction 1

The Modular Modeling Method 4

Modular Assembly of Two Components 6

Assembly Examples for FEM Plate Structure Models 10

Case 1: Two, Single Element, Nodally Compatible, Plate Models 11

Case 2: Nodally Compatible Component Plates with 4 Elements 14

Case 3: Nodally Incompatible Plate Assembly Model. 16

Case 4: Assembly of Higher Resolution Finite Elements. 19

Internet Agents for Plate Models and their Assembly 23

Conclusions 27

Appendix 30

References 40

List of Tables

Table 1 Displacement Values At Each Node For Both Plates With 1 Element Each 13

Table 2 Nodal Displacements For Both Plates With 4 Elements In Each Plate 16

Table 3 Displacement Values Of Assembled Plate Of Incompatible Nodal Geometry Using Modular Modeling 18

Table 4 The Displacement Values Of Assembled Plate Of Incompatible Nodal Geometry Using Modular Modeling Method 21

List of Figures

Figure 1 The Steel Plate Example 10

Figure 2 Finite Element Meshes Of Plates and Assembly 12

Figure 3 Component Plate FEM Meshes With 4 Elements Each And The Assembled Model 14

Figure 4 Assembly of Models having Incompatible Nodal Geometry. Grayed Node Is Condensed Out 17

Figure 5 Comparisons Of The Displacement Values At Nodal Interface Boundary 19

Figure 6 Higher Resolution Components with Incompatible Nodal geometry 20

Figure 7 Assembled Model With Incompatible Nodal Geometry. The Gray Nodes Indicate Condensed Nodes 21

Figure 8 Comparisons Of All Cases At Nodal Interface Boundary 22

Figure 9 The *i*-EDA System Schematic Showing A User Client That Queries Model Agents Registered in the Agent Registry Using Valid Queries From the Query Ontology. 23

Figure 10 The Framework of an Agent in the *i*-EDA System 24

Figure 11 The Front Panels Showing Stiffness Matrices Of The Upper And Lower Component Plates 25

Figure 12 The Front Panel Of Plate Agent Assembler. 26

Introduction

Structural analysis problems frequently involve the assembly of two or more components to form the structural system to be analyzed. Structural design and analysis widely incorporates the Finite Element method. Finite Element method is a systematic modeling method that generates mathematical models in structural design [Zienkiewicz, 1977 [i]]. While the Finite element method (FEM) can yield accurate, detailed stress solutions for individual subsystem components, FEM is not efficient at assembling components into assemblies. This is because every change in a substructure finite element model requires global reformulation of equations to ensure compatible nodal geometry of components at their interfaces. Reformulation of finite element mesh equations is the single most time consuming activity in FEM analysis. It involves rewriting the set of equations for each component and hence prevents efficient development of assembly models. Additionally, the need to reformulate component nodal geometries prevents the reuse of FEM models in a distributed modeling system.

At each component connection, the finite element method typically requires compatible node geometry across the boundary connecting components. Mesh nodal compatibility is typically required to construct a finite element model of an assembly from the finite element models of its components. Methods that do not require nodal compatibility still require that at the common interface the nodes on all subdomains must have the same number and same type of degrees of freedom. These methods either need interface elements to connect subsystems [Aminpour et al, 1995[ii]] or require additional constraints, typically in form of Lagrange multipliers for coupling the components into system [Farhat and Geradin, 1992[iii], Farhat and Roux, 1991[iv]]. Both cases add to the degrees of freedom of the system that results in overall increase of the size of the problem. The Modular Modeling Method does not require equation reformulation or nodal compatibility and does not add additional degrees of freedom to the system by adding extra constraints to it.

Modular Modeling is a relatively new method for solving structural analysis applications of distributed models. Modular modeling is a systematic equation assembly scheme well suited to multi DOF systems. While this method can be used in several energy domains like Electrical, Hydraulics, Acoustics and Heat transfer, the method will be applied here to structures. This method is a power-based systematic modeling method that eliminates equation reformulation from large model design, development, and refinement across multiple energy domains [Byam and Radcliffe, 1999[v]]. The Modular Modeling Method is a systematic approach to generate mathematical models with fixed input-output structure.

The Modular Modeling Method couples the subsystems or components having independent finite element modeling. The components need not to be nodally compatible. The Modular Modeling Method uses constraint equations that identify the relationship of the components to derive a model of the assembly of those components. These assembly models then have the same fixed input-output structure as their components and can be used to derive more complex assemblies.

Agent software facilitates the engineering design in a globally distributed environment. Gosciak, [2001[vi]] developed a component based design agent prototype based on the i-EDA methodology [Radcliffe and Sticklen, 2001[vii]] that uses a strict communication protocol. The Design Agents are capable of representing the whole system by providing answers to system attribute queries. In the work presented here, individual components are modeled as design agents, their assembly is done according to Modular Modeling Method and then the agents and the assembly are published on the internet using the i-EDA system.

The Modular Modeling Method

Fixed input-output structure is important to the Modular Modeling Method. Fixed input-output structure requires standardization of input and output variables. In the structural models discussed here, all inputs are nodal forces and all outputs are nodal displacements. With standardized input and output variables, the mathematical model that describes each component can remain fixed, independent of the system model into which it is inserted. The assembled system generated by Modular Modeling has the same standardized input and output variables [Byam and Radcliffe, 1999[viii]].

Structural components defined by the FEM can be assembled using the Modular Modeling method. In this work, the components are constructed independently and their finite element models need not to have compatible nodal geometry. FEM models generated without applied boundary conditions have rigid body modes and singular stiffness matrices. Modular Modeling uses systems of constraint equations to connect these singular component stiffness matrices by requiring conservation of power at structural connections. This eliminates the need for both global reformulation and compatible component nodal geometry. Because assembly models are assembled without applied boundary conditions, they also have rigid body modes and singular stiffness matrices.

Displacement Constraints connect two components by requiring consistent displacements along their connecting boundaries. In this work, the components are termed as Master and Slave components because the interface boundary displacements of the slave component are written in terms of the master component. The idea of Master and Slave is not new. Cook et al [1989[ix]] used these terms within a subsystem. In the substructure or component the nodes at boundary were called master and internal nodes interior in that component were termed as the Slave nodes. Here, the whole component is assigned the respective name. The component having fewest nodes at connection is used as the master component. This allows nodal displacements of the slave to be written in terms of the master component and results in removal of unconstrained nodes of slave component at the connection. Work constraints require conservation of work along the boundary of physical connections. By definition, work is the product of force and displacement. The total work done across the connection boundary must sum to zero in this static analysis.

Assembly of components using these constraints condenses out the connection boundary nodes of slave component and retains connection boundary nodes of the master component. The assembled model has the same fixed input output structure as its constituent components. The model now can either be used a new subsystem to connect to other subsystems or some boundary conditions and loads can be applied to allow for the solution of the system displacement response.

Modular Assembly of Two Components

Static analysis of assembly of two components using the MMM is presented here. The analysis assembles two components, which may or may not have nodal compatibility at their connecting boundary. The stiffness matrices of both components are obtained independently and are written in the linear form,

$$[\mathbf{K}][\mathbf{U}] = [\mathbf{F}] \tag{1}$$

K is a component structure stiffness matrix, **U** is the vector of component nodal displacements and **F** is vector of component nodal loads. Because every nodal displacement has an associated nodal force, **K** is square. Additionally, **K** is often symmetric [Zienkiewicz and Taylor, 1989[x]] and, in the MMM assembly, if all component stiffness are symmetric, the assembly stiffness will also be symmetric. Here the component structures have rigid body modes associated with them, thus **K** is singular.

The assembly of the component equations begins with the unconstrained, uncoupled, system equations,

$$\begin{bmatrix} \mathbf{K}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_2 \end{bmatrix} \begin{bmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{bmatrix} \tag{2}$$

Displacement and work constraints will be used to assemble the subsystems. The displacement constraints are a set of equations that identify the physical relationships between displacement output values at each connected node of the components. The displacement values of connected nodes of one component are written in terms of displacement values of the other component. The system equations are first rearranged,

$$[\mathbf{U}] = \begin{bmatrix} \overline{\mathbf{U}} \\ \hat{\mathbf{U}} \end{bmatrix}$$
$$[\mathbf{F}] = \begin{bmatrix} \overline{\mathbf{F}} \\ \hat{\mathbf{F}} \end{bmatrix} \tag{3}$$

where $\overline{\mathbf{U}}$ and $\overline{\mathbf{F}}$ are component displacements and forces at unconnected nodes while $\hat{\mathbf{U}}$ and $\hat{\mathbf{F}}$ are nodal displacement and force variables included in the work constraints. Using the master "M" and slav*e* "S" component notation, the stiffness matrices of the components obtained are represented in the form of master and slave subsystems.

$$\left[\begin{array}{cc:cc} \mathbf{K}_{M_{11}} & \mathbf{K}_{M_{12}} & \multicolumn{2}{c}{\multirow{2}{*}{\mathbf{0}}} \\ \mathbf{K}_{M_{21}} & \mathbf{K}_{M_{22}} & & \\ \hdashline \multicolumn{2}{c:}{\multirow{2}{*}{\mathbf{0}}} & \mathbf{K}_{S_{11}} & \mathbf{K}_{S_{12}} \\ & & \mathbf{K}_{S_{21}} & \mathbf{K}_{S_{22}} \end{array}\right] \left[\begin{array}{c} \overline{\mathbf{U}}_M \\ \hat{\mathbf{U}}_M \\ \hdashline \overline{\mathbf{U}}_S \\ \hat{\mathbf{U}}_S \end{array}\right] = \left[\begin{array}{c} \overline{\mathbf{F}}_M \\ \hat{\mathbf{F}}_M \\ \hdashline \overline{\mathbf{F}}_S \\ \hat{\mathbf{F}}_S \end{array}\right] \tag{4}$$

The displacement constraints give the slave nodal displacement $\hat{\mathbf{U}}_S$ as a linear combination of the master nodal displacements $\hat{\mathbf{U}}_M$

$$[\hat{\mathbf{U}}_S] = [\mathbf{S}][\hat{\mathbf{U}}_M] \tag{5}$$

When these displacement constraints are applied, the number of columns of the system equations is reduced by the number of slave displacements along interface boundary, the constrained assembly equations result in a rectangular assembly matrix

$$\left[\begin{array}{cc:c} \mathbf{K}_{M_{11}} & \mathbf{K}_{M_{12}} & \mathbf{0} \\ \mathbf{K}_{M_{21}} & \mathbf{K}_{M_{22}} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{K}_{S_{12}}\mathbf{S} & \mathbf{K}_{S_{11}} \\ \mathbf{0} & \mathbf{K}_{S_{22}}\mathbf{S} & \mathbf{K}_{S_{21}} \end{array}\right] \left[\begin{array}{c} \overline{\mathbf{U}}_M \\ \hat{\mathbf{U}}_M \\ \hdashline \overline{\mathbf{U}}_S \end{array}\right] = \left[\begin{array}{c} \overline{\mathbf{F}}_M \\ \hat{\mathbf{F}}_M \\ \hdashline \overline{\mathbf{F}}_S \\ \hat{\mathbf{F}}_S \end{array}\right] \tag{6}$$

Work constraints are applied to remove $\hat{\mathbf{F}}_M$ and generate a square system matrix. The work constraints require work done across the assembled component boundaries to be zero when no external forces are present at the connected nodes. If some external forces are present at the connected nodes (for example reaction forces), then this relation sums to that external force vector.

$$\hat{\mathbf{F}}_M^{\mathbf{T}} \hat{\mathbf{U}}_M + \hat{\mathbf{F}}_S^{\mathbf{T}} \hat{\mathbf{U}}_S = \hat{\mathbf{F}}_{ext}^{\mathbf{T}} \hat{\mathbf{U}}_M \tag{7}$$

where the first product is the work done on the master component by the connection and the second term is the work done on the slave component. Applying displacement constraints on (7) yields

$$[\hat{\mathbf{F}}_M^{\mathbf{T}} + \hat{\mathbf{F}}_S^{\mathbf{T}} \mathbf{S}] \hat{\mathbf{U}}_M = \hat{\mathbf{F}}_{ext}^{\mathbf{T}} \hat{\mathbf{U}}_M \tag{8}$$

Because (8) must be valid for any displacement vector $\hat{\mathbf{U}}_M$,

$$\hat{\mathbf{F}}_M + \mathbf{S}^{\mathbf{T}} \hat{\mathbf{F}}_S = \hat{\mathbf{F}}_{ext} \tag{9}$$

The work constraints (9) applied to (6) results in a square, symmetric, assembly stiffness model,

$$\left[\begin{array}{c:c:c} \mathbf{K}_{M_{11}} & \mathbf{K}_{M_{12}} & \mathbf{0} \\ \hdashline \mathbf{K}_{M_{21}} & \left[\mathbf{K}_{M_{22}} + \mathbf{S}^{\mathbf{T}} \mathbf{K}_{S_{22}} \mathbf{S}\right] & \mathbf{S}^{\mathbf{T}} \mathbf{K}_{S_{21}} \\ \hdashline \mathbf{0} & \mathbf{K}_{S_{12}} \mathbf{S} & \mathbf{K}_{S_{11}} \end{array}\right] \left[\begin{array}{c} \bar{\mathbf{U}}_M \\ \hdashline \hat{\mathbf{U}}_M \\ \hdashline \bar{\mathbf{U}}_S \end{array}\right] = \left[\begin{array}{c} \bar{\mathbf{F}}_M \\ \hdashline \hat{\mathbf{F}}_{ext} \\ \hdashline \bar{\mathbf{F}}_S \end{array}\right] \tag{10}$$

This new mathematical model of the assembly has different characteristics from its component models in terms of nodes and geometrical shape. The matrix is symmetric as long as the constituent stiffness matrices of master and slave components are symmetric.

The assembly model completely describes the structural properties of the assembled model, has the same form as component models (1) and can be used to assemble more complex assemblies.

Assembly Examples for FEM Plate Structure Models

The mathematical modeling of the structural assembly of two components has been described using the Modular Modeling Method. This method will be applied below to a square steel plate (Figure. 1) assembled from two triangular component models and solved to verify the modular assembly method discussed above.

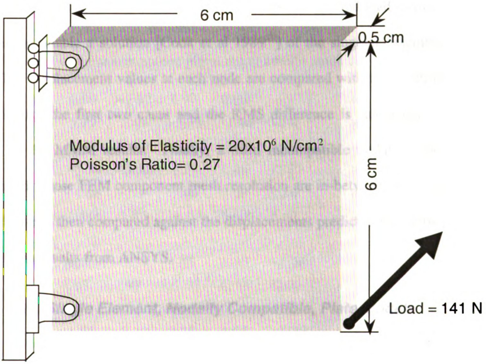

Figure 1: The Steel Plate Example

Four cases are presented for different finite element meshes of each triangular plate. The first two cases have compatible interface geometry with different grid size and are used to compare the MMM result with conventional FEM assembly. The third case generates an assembly with incompatible nodal geometry from the elements developed in cases one and two. This case has no conventional FEM model comparison, however, so its accuracy must be evaluated against similar FEM. models that should bound this case from both above and below. Finally, a case of higher mesh resolution is presented to demonstrate that, the higher the FEM component model resolution, the better is the accuracy of the resulting, assembled, MMM model.

The individual component stiffness matrices are computed using triangular finite elements by standard finite element procedures [Segerlind, 1984[xi]] and assembled with the Modular Modeling Method. Boundary conditions and applied load shown in Figure 1 are applied to enable a solution [Cook et al 1989[xii]] of the singular assembled stiffness matrix. The displacement values at each node are compared with the solution generated by ANSYS for the first two cases and the RMS difference is calculated to verify the accuracy of the MMM results. Finally, a third incompatible nodal geometry case is demonstrated whose FEM component mesh resolution are in-between the first two cases. This final case is then compared against the displacements predicted by the two; high and low resolution results from ANSYS.

Case 1: Two, Single Element, Nodally Compatible, Plate Models

Case one is a simple 2D problem (Fig. 2) that assembles two triangular plates each consisting of one triangular element with three nodes. Plates are assumed to have 2 DOF at every node and two nodes along each side. Because their meshes have the same number and placement of boundary nodes, the plates have compatible nodal geometry at the boundary. Plates are assigned master and slave names, the stiffness matrices of both plates derived, and the components are assembled using MMM. The boundary conditions and loads are then applied and the nodal displacements found at each node for comparison with commercial finite element software ANSYS results.

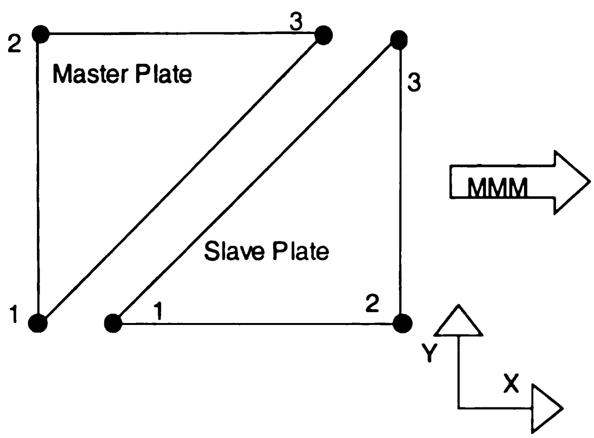

Figure 2 Finite Element Meshes Of Plates and Assembly

The FEM models of these component plates are relatively small and are given as.

$$\begin{bmatrix} 1.97 & 0.00 & -1.97 & 1.97 & 0.00 & -1.97 \\ 0.00 & 5.39 & 1.46 & -5.39 & -1.46 & 0.00 \\ -1.97 & 1.46 & 7.36 & -3.42 & -5.39 & 1.97 \\ 1.97 & -5.39 & -3.42 & 7.36 & 1.46 & -1.97 \\ 0.00 & -1.46 & -5.39 & 1.46 & 5.39 & 0.00 \\ -1.97 & 0.00 & 1.97 & -1.97 & 0.00 & 1.97 \end{bmatrix} x10^6 \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{2x} \\ u_{2y} \\ u_{3x} \\ u_{3y} \end{bmatrix}_{\mathbf{M}} = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{2x} \\ f_{2y} \\ f_{3x} \\ f_{3y} \end{bmatrix}_{\mathbf{M}} \tag{11}$$

$$\begin{bmatrix} 5.39 & 0.00 & -5.39 & 1.46 & 0.00 & -1.46 \\ 0.00 & 1.97 & 1.97 & -1.97 & -1.97 & 0.00 \\ -5.39 & 1.97 & 7.36 & -3.42 & 1.97 & 1.46 \\ 1.46 & -1.97 & -3.42 & 7.36 & 1.97 & -5.39 \\ 0.00 & -1.97 & -1.97 & 1.97 & 1.97 & 0.00 \\ -1.46 & 0.00 & 1.46 & -5.39 & 0.00 & 5.39 \end{bmatrix} x10^6 \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{2x} \\ u_{2y} \\ u_{3x} \\ u_{3y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{2x} \\ f_{2y} \\ f_{3x} \\ f_{3y} \end{bmatrix}_{\mathbf{S}} \tag{12}$$

The assembly of master (11) and slave (12) models uses the displacement constraints (5),

$$\begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \end{bmatrix}_{\mathbf{M}} \tag{13}$$

so that the constraint matrix $\mathbf{S} = \mathbf{I}$. The work constraints (9) are,

$$\begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \end{bmatrix}_M + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \end{bmatrix}_S = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \end{bmatrix} \tag{14}$$

Using (10), the assembly model in case 1 is,

$$\begin{bmatrix} 7.36 & -3.42 & -1.97 & 1.46 & -5.39 & 1.97 & 0.00 & 0.00 \\ -3.42 & 7.36 & 1.97 & -5.39 & 1.46 & -1.97 & 0.00 & 0.00 \\ -1.97 & 1.97 & 7.36 & 0.00 & 0.00 & -3.42 & -5.39 & 1.46 \\ 1.46 & -5.39 & 0.00 & 7.36 & -3.42 & 0.00 & 1.97 & -1.97 \\ -5.39 & 1.46 & 0.00 & -3.42 & 7.36 & 0.00 & -1.97 & 1.97 \\ 1.97 & -1.97 & -3.42 & 0.00 & 0.00 & 7.36 & 1.46 & -5.39 \\ 0.00 & 0.00 & -5.39 & 1.97 & -1.97 & 1.46 & 7.36 & -3.42 \\ 0.00 & 0.00 & 1.46 & -1.97 & 1.97 & -5.39 & -3.42 & 7.36 \end{bmatrix} x10^6 \begin{bmatrix} u_{2x} \\ u_{2y} \\ u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \\ u_{4x} \\ u_{4y} \end{bmatrix} = \begin{bmatrix} f_{2x} \\ f_{2y} \\ f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \\ f_{4x} \\ f_{4y} \end{bmatrix} \tag{15}$$

The assembly process (3) – (10) requires a reordering of the nodal variables and the above result reflects the new nodal displacement and force order. Now boundary conditions and Nodal forces are applied to further reduce the order of the problem (see Appendix). The result of application of boundary conditions and load is the set of nodal displacements shown in Table 1. Because standard element equations are used for the finite element model, these results agree exactly with the results computed by ANSYS for this simple model.

Table 1 Displacement Values At Each Node For Both Plates With 1 Element Each

Node Number	X-Displacement U_x (ANSYS) (cm)	X-Displacement U_y MMM (cm)	Y-Displacement U_x (ANSYS) (cm)	Y-Displacement U_y (MMM (cm)
1	0.0000E-04	0.0000E-04	0.0000E-04	0.0000E-04
2	0.0000E-04	0.0000E-04	0.1403E-04	0.1403E-04
3	-0.1137E-04	-0.1137E-04	0.4407E-04	0.4407E-04
4	0.3137E-04	0.3137E-04	0.6350E-04	0.6350E-04
RMS DIFFERENCE		**U_x=0.0000E-04**		**U_y=0.0000E-04**

Case 2: Nodally Compatible Component Plates with 4 Elements

This case involves two finite element models with 3 nodes on each side. Again this is a case of compatible nodal geometry. This case demonstrates that MMM produces the same results as a standard FEM package for multiple element models and will be used for later verification of the nodally incompatible model.

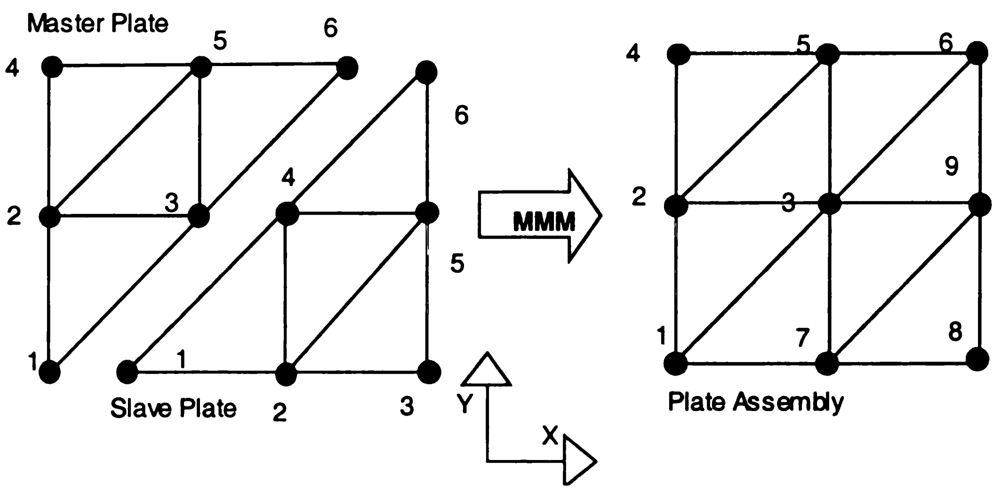

Figure 3 Component Plate FEM Meshes With 4 Elements Each And The Assembled Model

The master plate stiffness matrix,

$$
\mathbf{K}=\begin{bmatrix}
1.97 & 0.00 & -1.97 & 1.97 & 0.00 & -1.97 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 5.39 & 1.46 & -5.39 & -1.46 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
-1.97 & 1.46 & 14.72 & -3.42 & -10.78 & 3.42 & -1.97 & 1.97 & 0.00 & -3.42 & 0.00 & 0.00 \\
1.97 & -5.39 & -3.42 & 14.72 & 3.42 & -3.93 & 1.46 & -5.39 & 3.42 & 0.00 & 0.00 & 0.00 \\
0.00 & -1.46 & -10.78 & 3.42 & 14.72 & -3.42 & 0.00 & 0.00 & 3.93 & 3.42 & 0.00 & -1.97 \\
-1.97 & 0.00 & 3.42 & -3.93 & -3.42 & 14.72 & 0.00 & 0.00 & 3.42 & -10.78 & -1.46 & 0.00 \\
0.00 & 0.00 & -1.97 & 1.46 & 0.00 & 0.00 & 7.36 & -3.42 & -5.39 & 1.97 & 0.00 & 0.00 \\
0.00 & 0.00 & 1.97 & -5.39 & 0.00 & 0.00 & -3.42 & 7.36 & 1.46 & -1.97 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & -3.42 & -3.93 & 3.42 & -5.39 & 1.46 & 14.72 & -3.42 & -5.39 & 1.97 \\
0.00 & 0.00 & -3.42 & 0.00 & 3.42 & -10.78 & 1.97 & -1.97 & -3.42 & 14.72 & 1.46 & -1.97 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.46 & 0.00 & 0.00 & 5.39 & 1.46 & 5.39 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & -1.97 & 0.00 & 0.00 & 0.00 & 1.97 & -1.97 & 0.00 & 1.97
\end{bmatrix} x10^{6} \qquad (16)
$$

Nodal Displacements,

$$\mathbf{U} = \begin{bmatrix} u_{1x} & u_{1y} & u_{2x} & u_{2y} & u_{3x} & u_{3y} & u_{4x} & u_{4y} & u_{5x} & u_{5y} & u_{6x} & u_{6y} \end{bmatrix}_{\mathbf{M}}^{T}$$

and Nodal forces,

$$\mathbf{F} = \begin{bmatrix} f_{1x} & f_{1y} & f_{2x} & f_{2y} & f_{3x} & f_{3y} & f_{4x} & f_{4y} & f_{5x} & f_{5y} & f_{6x} & f_{6y} \end{bmatrix}_{\mathbf{M}}^{T}$$

The Slave plate stiffness matrix,

$$\mathbf{K} = \begin{bmatrix} 5.39 & 0.00 & -5.39 & 1.46 & 0.00 & 0.00 & 0.00 & -1.46 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.97 & 1.97 & -1.97 & 0.00 & 0.00 & -1.97 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ -5.39 & 1.97 & 14.72 & -3.42 & -5.39 & 1.46 & -3.97 & 3.42 & 0.00 & -3.42 & 0.00 & 0.00 \\ 1.46 & -1.97 & -3.42 & 14.72 & 1.97 & -1.97 & 3.42 & -10.78 & -3.42 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & -5.39 & 1.97 & 7.36 & -3.42 & 0.00 & 0.00 & -1.97 & 1.46 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.46 & -1.97 & -3.42 & 7.36 & 0.00 & 0.00 & 1.97 & -5.39 & 0.00 & 0.00 \\ 0.00 & -1.97 & -3.97 & 3.42 & 0.00 & 0.00 & 14.72 & -3.42 & -10.78 & 3.42 & 0.00 & -1.46 \\ -1.46 & 0.00 & 3.42 & -10.78 & 0.00 & 0.00 & -3.42 & 14.72 & 3.42 & -3.93 & -1.97 & 0.00 \\ 0.00 & 0.00 & 0.00 & -3.42 & -1.97 & 1.97 & -10.78 & 3.42 & 14.72 & -3.42 & -1.97 & 1.46 \\ 0.00 & 0.00 & -3.42 & 0.00 & 1.46 & -5.39 & 3.42 & -3.93 & -3.42 & 14.72 & 1.97 & -5.39 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.97 & -1.97 & 1.97 & 1.97 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & -1.46 & 0.00 & 1.46 & -5.39 & 0.00 & 5.39 \end{bmatrix} x10^{6} \qquad (17)$$

The corresponding Nodal Displacements and Forces are

$[u_{1x} \quad u_{1y} \quad u_{2x} \quad u_{2y} \quad u_{3x} \quad u_{3y} \quad u_{4x} \quad u_{4y} \quad u_{5x} \quad u_{5y} \quad u_{6x} \quad u_{6y}]^T$ and

$[f_{1x} \quad f_{1y} \quad f_{2x} \quad f_{2y} \quad f_{3x} \quad f_{3y} \quad f_{4x} \quad f_{4y} \quad f_{5x} \quad f_{5y} \quad f_{6x} \quad f_{6y}]^T$ respectively.

The displacement constraints (5) for this case are

$$\begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \\ u_{6x} \\ u_{6y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{4x} \\ u_{4y} \\ u_{6x} \\ u_{6y} \end{bmatrix}_{\mathbf{M}} \qquad (18)$$

Work Constraints using $\mathbf{S} = \mathbf{I}$ are

$$\begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \\ f_{6x} \\ f_{6y} \end{bmatrix}_M + \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{4x} \\ f_{4y} \\ f_{6x} \\ f_{6y} \end{bmatrix}_S = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \\ f_{6x} \\ f_{6y} \end{bmatrix} \qquad (19)$$

The assembly is done using the above constraints. By applying boundary conditions and loads, nodal displacements are found and compared with ANSYS results. The RMS difference calculated for this nodally compatible assembly shows no difference in results.

Table 2 Nodal Displacements For Both Plates With 4 Elements In Each Plate

Node Number	X-Displacement (U_x) (ANSYS) (cm)	X-Displacement (U_x) (Modular Modeling) (cm)	Y-Displacement U_y (ANSYS) (cm)	Y-Displacement U_y (Modular Modeling) (cm)
1	0.0000E-04	0.0000E-04	0.0000E-04	0.0000E-04
2	0.0067E-04	0.0067E-04	0.1493E-04	0.1493E-04
3	0.0009E-04	0.0009E-04	0.3641E-04	0.3641E-04
4	0.0000E-04	0.0000E-04	0.2411E-04	0.2411E-04
5	-0.1621E-04	-0.1621E-04	0.3796E-04	0.3796E-04
6	-0.2241E-04	-0.2241E-04	0.7092E-04	0.7092E-04
7	0.3003E-04	0.3003E-04	0.4024E-04	0.4024E-04
8	0.5739E-04	0.5739E-04	1.0127E-04	1.0127E-04
9	0.0319E-04	0.0319E-04	0.7784E-04	0.7784E-04
RMS DIFFERENCE	**U_x=0.0000E-04**		**U_y=0.0000E-04**	

Case 3: Nodally Incompatible Plate Assembly Model.

Here a plate problem with incompatible interface nodes is solved. The assembly involves a master plate having 2 nodes on each side and a slave plate with 3 nodes on each side as shown in figure (4). This case is combination of previous two cases. The finite element model already found in (12) is connected with finite element model already found in (16) to make an assembly of finite element models having incompatible nodal geometry.

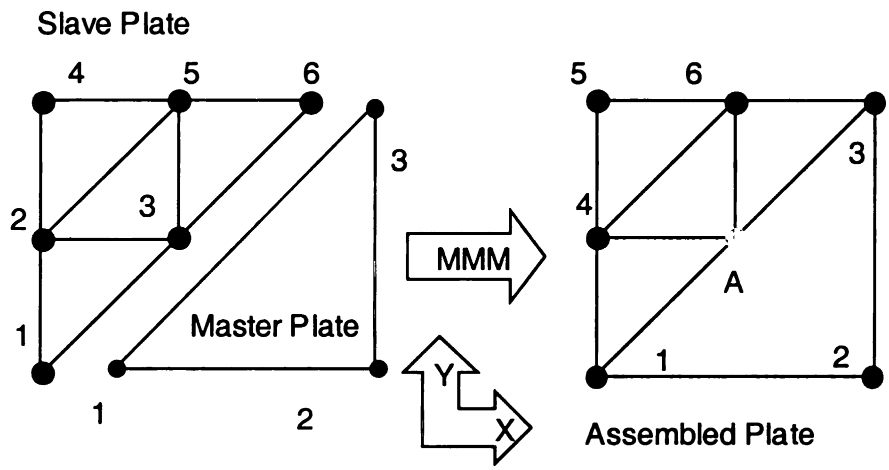

Figure 4 Assembly of Models having Incompatible Nodal Geometry. Grayed Node Is Condensed Out

The displacement constraints given for this case are

$$\begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \\ u_{6x} \\ u_{6y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \end{bmatrix}_{\mathbf{M}} \tag{20}$$

Note that the S matrix is not identity in this case as opposite to the previous two cases. The work constraints conserving work at the boundary in this case are

$$\begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \end{bmatrix}_M + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \\ f_{6x} \\ f_{6y} \end{bmatrix}_S = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \end{bmatrix} \tag{21}$$

The assembled model uses the constraint equations (20-21) for assembly. The application of nodal loads and boundary conditions results in nodal displacements shown in table 3.

Table 3 Displacement Values Of Assembled Plate Of Incompatible Nodal Geometry Using Modular Modeling

Node Number	X-Displacement U_x (cm)	Y-Displacement U_y (cm)
1	0.0000E-04	0.0000E-04
2	0.3236E-04	0.7543E-04
3	-0.1884E-04	0.5699E-04
4	-0.0648E-04	0.1281E-04
5	0.0000E-04	0.2237E-04
6	-0.1584E-04	0.3035E-04

The displacement values are in accordance with the expectations as this model lies in between the stiff model (case 1) and the relatively compliant model (case 2). The grayed node "A" is the simple linear interpolation of nodes 1 and 3 and its displacement values are defined by the displacement constraints in terms of nodes 1 and 3. The connection force values can be ascertained by substituting the displacement values calculated above (table 3) in the assembled model.

The comparison of displacement values for all 3 cases shown in graph 1 indicates the behavior of displacement values at the interface. Case 3 is less stiff than case 1 and more stiff than case 2 so the nodal displacement values obtained in this case are expected to lie between the first two cases. Figure 5 indicates the expected results. It also indicates that whether the nodal interface geometry is compatible or not, MMM assembles the model and gives the expected results. Thus it can be assumed that MMM correctly assembles the model

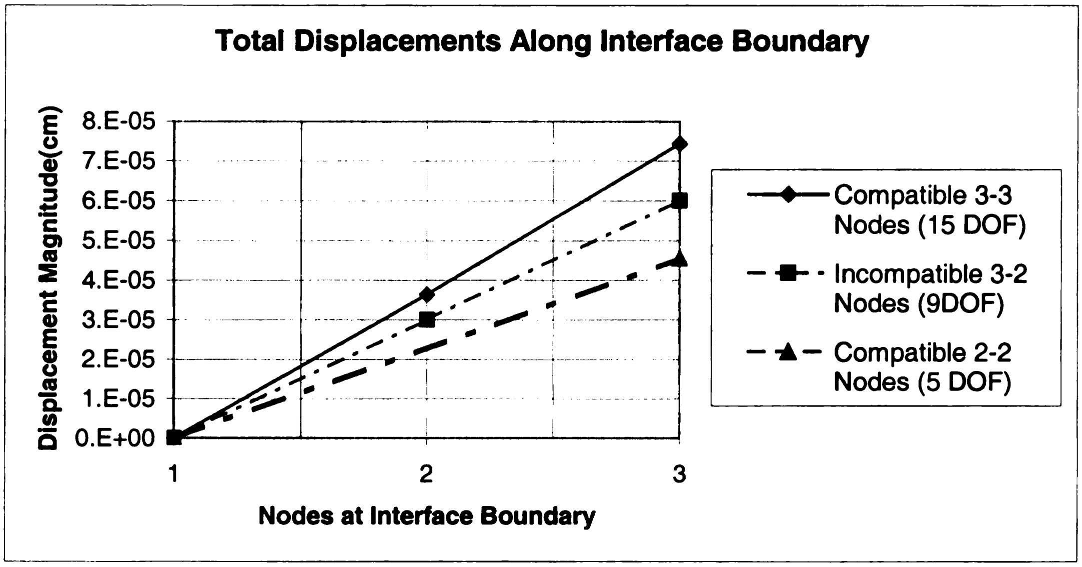

Figure 5 Comparisons Of The Displacement Values At Nodal Interface Boundary

Case 4: Assembly of Higher Resolution Finite Elements.

The assembling of higher resolution meshes is done in the same way as discussed above. Consider the same example plate with the master and slave plates having 3 and 4 nodes along their boundaries shown in figure (6). This is a problem with relatively large DOF and more incompatibility at the interface boundary. The displacement and work constraints are given respectively as

$$
\begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{3x} \\ u_{3y} \\ u_{6x} \\ u_{6y} \\ u_{10x} \\ u_{10y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 \\ 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{1x} \\ u_{1y} \\ u_{4x} \\ u_{4y} \\ u_{6x} \\ u_{6y} \end{bmatrix}_{\mathbf{M}} \tag{23}
$$

$$
\begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{4x} \\ f_{4y} \\ f_{6x} \\ f_{6y} \end{bmatrix}_{\mathbf{M}} + \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 \\ 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^{T} \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{3x} \\ f_{3y} \\ f_{6x} \\ f_{6y} \\ f_{10x} \\ f_{10y} \end{bmatrix}_{\mathbf{S}} = \begin{bmatrix} f_{1x} \\ f_{1y} \\ f_{4x} \\ f_{4y} \\ f_{6x} \\ f_{6y} \end{bmatrix} \tag{24}
$$

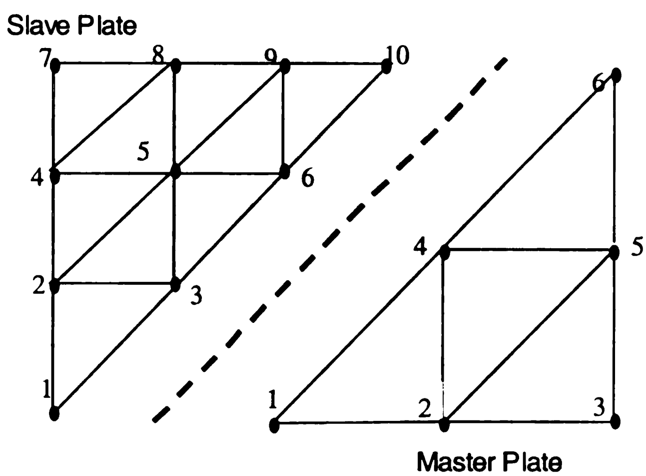

Figure 6 Higher Resolution Components with Incompatible Nodal geometry

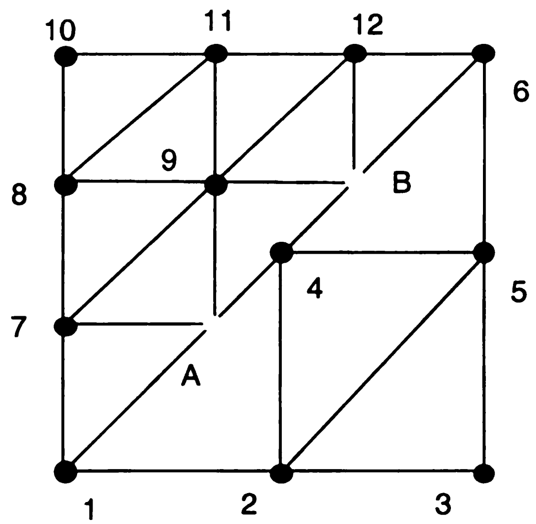

Figure 7 Assembled Model With Incompatible Nodal Geometry. The Gray Nodes Indicate Condensed Nodes

The displacement values obtained at each node are given in Table 4.

Table 4 The Displacement Values Of Assembled Plate Of Incompatible Nodal Geometry Using Modular Modeling Method

Node Number	X-Displacement U_x cm	Y-Displacement U_y (cm)
1	0.0000E-04	0.0000E-04
2	0.3153E-04	0.4606E-04
3	0.5899E-04	1.1037E-04
4	-0.0075E-04	0.4265E-04
5	0.0188E-04	0.8704E-04
6	-0.2874E-04	0.7985E-04
7	-0.0005E-04	0.1397E-04
8	-0.0669E-04	0.2114E-04
9	-0.0918E-04	0.3178E-04
10	0.0000E-04	0.2965E-04
11	-0.1688E-04	0.3380E-04
12	-0.2610E-04	0.5476E-04

The case predicts the displacement response of the system at meshes that have higher resolutions than the previously discussed cases. It is evident from the results that the displacement response actual solution as the number of DOF of system increase as shown in fig 8. Again, the displacement values of gray shaded nodes "A" and "B" are the combination of nodal values of 1, 4 and 6 as defined in (23).

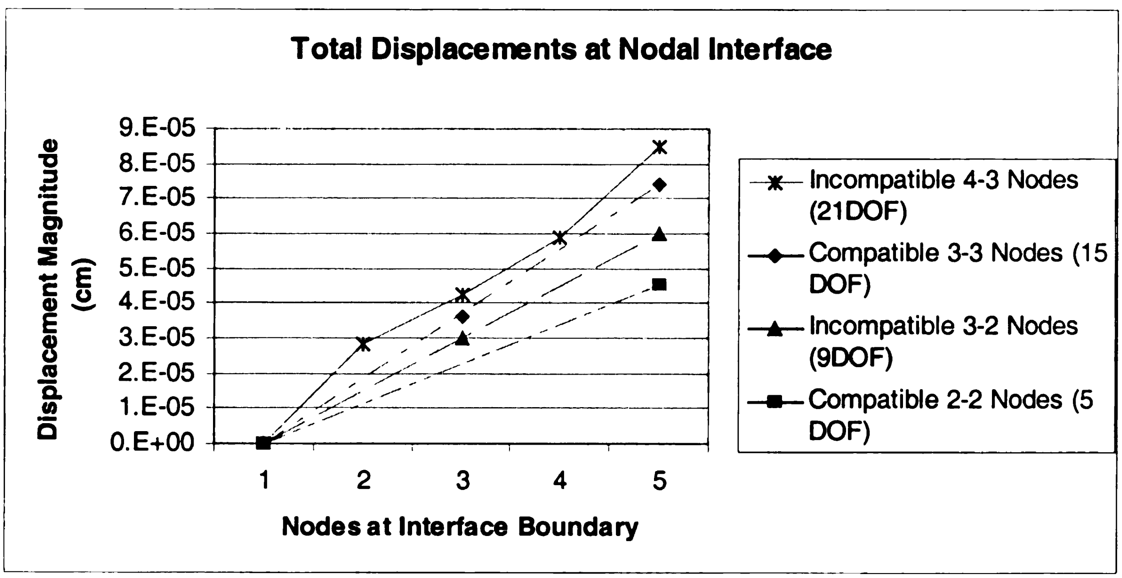

Figure 8 Comparisons Of All Cases At Nodal Interface Boundary

Internet Agents for Plate Models and their Assembly

A globally distributed market requires software to design, manufacture and market the assemblies of components and subsystems. An internet-based engineering design system would reduce design cycle time and increase efficiency. These internet-based agents should, not only be able to effectively and rapidly assemble these components but also protect proprietary information underlying these models. In earlier work by Gosciak (2001), a prototype system of Internet Engineering Design Agents (i-EDA) was organized (Radcliffe and Sticklen, 2001) to facilitate the exchange of engineering design performance data between corporate organizations while protecting proprietary design information. The current work incorporates the plate models and their assemblies into the i-EDA system (Fig 9).

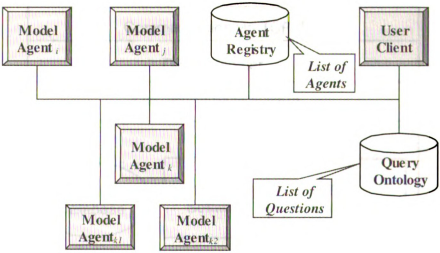

Figure 9 The *i*-EDA System Schematic Showing A User Client That Queries Model Agents Registered in the Agent Registry Using Valid Queries From the Query Ontology.

Users of the *i*-EDA system employ client software to submit queries to agents representing physical components and systems from which a new design is to be assembled. The agents are registered with the *i*-EDA Registry. The *i*-EDA system includes an Ontology, or ordered list, of valid queries. Typical queries return information on engineering, geometric, and or economic performance of the physical objects they represent. The model agents represent either components or assemblies of components. The framework of an *i*-EDA agent is shown in Fig. 10. The individual design agent includes a network communication protocol, a query handler, a knowledge based system and the resource set. Queries are received via the communication protocol and parsed by query handler into a suitable form. The knowledge base then utilizes the internal resources to assemble the response. For agents representing system, these resources may consult agents representing system components.

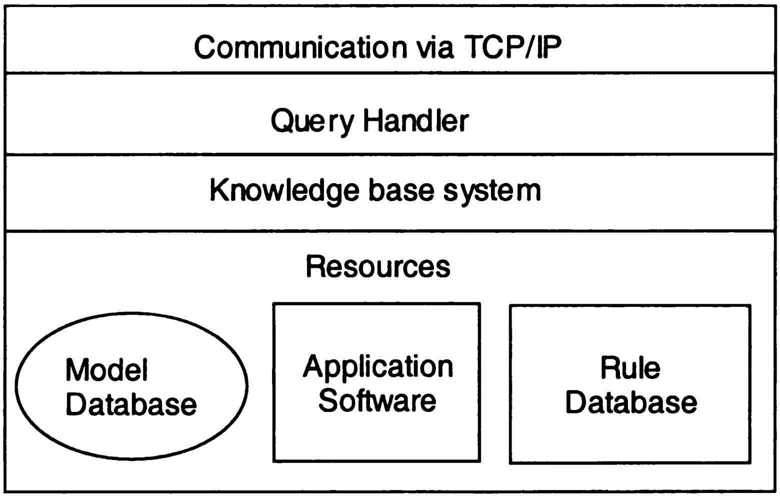

Figure 10 The Framework of an Agent in the *i*-EDA System

The distributed Design Agents for the plates, represented by two separate companies "Upper Plates Inc" and "Lower plates Inc" were added to the existing *i*-EDA system. These design agents are component agents that provide independent answers to the query for stiffness and return the plate stiffness matrix for each of these components. In addition queries for cost, color, weight, area, size and delivery time are answered. These plate models follow the same architecture as existing agents for truss and spans system assemblies formed from bar components. The attributes are requested as queries to the design agents. For answering the queries, separate software routines address each query attribute for any part number of the specific plate. The front panel, block diagram and framework of the plate agents are similar.

The stiffness query responses for the 1-element models of the upper and lower plates are shown in Fig (11). For a specific plate part number as shown, the agents build and display the FEM model of the plate. Here, the stiffness matrices displayed on the front panels of both plate agents are same as the stiffness matrices generated in (13-14).

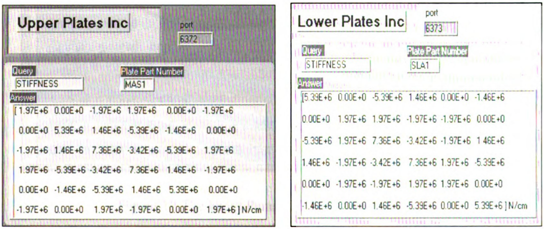

Figure 11 The Front Panels Showing Stiffness Matrices Of The Upper And Lower Component Plates

The plate agent assembler "Miller Smith Plates"(fig 12) is a system class of design agents that uses the information provided by the component plate agents to answer the queries. As an example, this design agent takes the FEM models provided by the other plate agents as input, assembles the combined Plate stiffness matrix using Modular Modeling Method for a plate part number and represents the stiffness of the assembled plate as a new attribute. Different part numbers for different plates that may or may not have compatible nodal geometry are visible on the front panel. Future class of agents can further use the information provided by this agent in the similar fashion. This is the advantage of i-EDA that generalized architecture allows for use of the resources by other design agents. The stiffness matrix obtained here is same as obtained in (15).

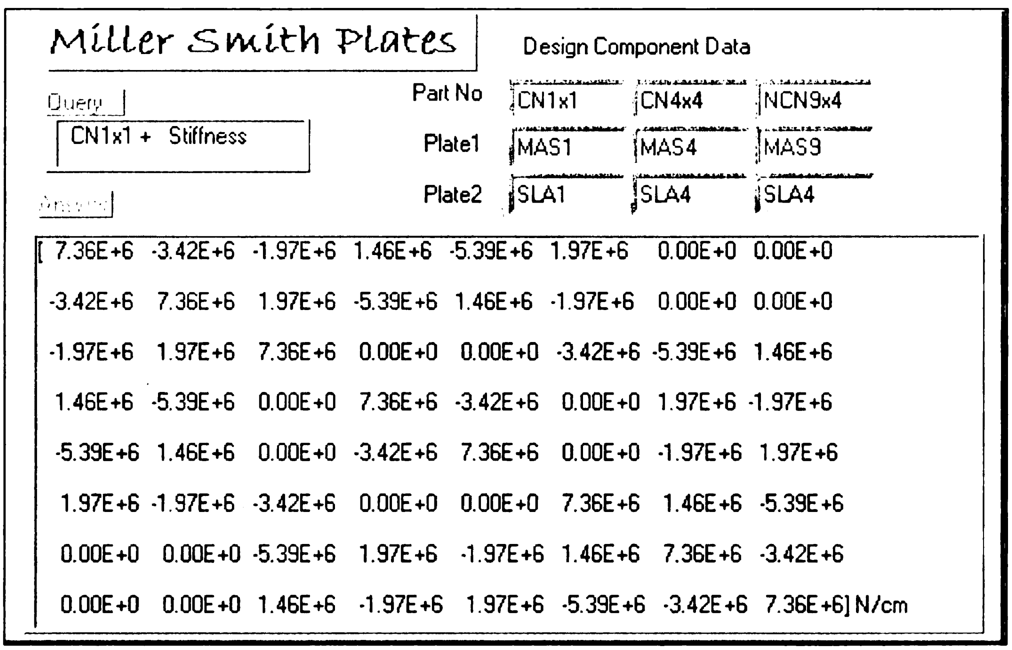

Figure 12 The Front Panel Of Plate Agent Assembler.

Conclusions

The new method (MMM) of solving structural problems is both efficient and time saving. It avoids the reformulation of system equations and allows easy assembling for problems that involve Incompatible Finite Element meshing without adding constraints to them. Also, it condenses the order of problem to a level that is far more convenient than some of the methods currently used.

Modular modeling of structures using independent, Finite Element formulations for components has been explained. This method does not require components to have compatible finite element grids, assembles the components without reformulating equations or requiring any interface element or additional constraints.

For assembling the two components the displacement and work constraints were applied after the partitioning of components into connected and unconnected nodes. These equations conserved the work done at connections and eliminated the linearly dependent equations of the system thus reducing the order of the system. The assembly model formed from its component models had different shape and nodal geometry but the same input output structure as its components. This model can now be used independently as a new finite element model of plate assembly. This model can also be used as a component for further assemblies.

Four test cases were presented. Each test case represents differing resolution models of identical physical plate geometry, boundary conditions and nodal loads. The first two cases are of compatible nodal interface geometry, first case having course mesh resolution resulting in a stiff model, and the second having relatively fine mesh resulting in a compliant problem. Solving these cases demonstrated that MMM produces the same results as any standard FEM used for the solution of such problems. The third case is an incompatible nodal interface geometry case that uses the FEM models obtained by the first two cases. This case demonstrated that the MMM produced the expected results that are less stiff than the first case and less compliant than the second case. The fourth case completed the examples by demonstrating that as resolution of models increases there is an improvement in the results, and that MMM accurately predicts the displacement response along the boundary of models.

These test cases show

- Modular Modeling Method derived for assembling components agrees exactly with the standard FEM package hence it is valid.
- Modular Assembly has the same fixed input output structure as its components. The assembly of components does not require reformulation of system equations and the order of the system equations is reduced.
- Modular Modeling Method accurately predicts the behavior of models having incompatible nodal interface geometry.
- Method uses simple constraints to assemble any two components. The methodology is generic for all cases and no special element or additional constraints are required to assemble the models with incompatible interface boundary.

Internet Design agents for component plate models and their assemblies have been built for global distribution of engineering design database. It is shown that the plate agent models generate the same FEM models as other commercial FEM packages. The plate assembly agent assembles the FEM models of different plates provided by plate agent models using the Modular Modeling Method. The plate assembly agents maintain the structural properties of assembly by maintaining the fixed input output structure thus enabling the assembly to be used as a component for further assemblies. The system of *i*-EDA allows publishing of these models over the global internet market.

Structural assemblies of models involving incompatible nodal geometry require special treatment either in the form of global reformulation or additional constraints. The Modular Modeling Method presented here avoids the global reformulation of equations or additional constraints and efficiently assembles the systems giving accurate results. Thus it avoids complicated designs and saving computation time. The modeling methods developed here form the analytical foundation for new *i*-EDA agents permitting distributed modeling of structural systems.

Appendix

The Finite Element Model Of Plate

The triangular plates used in analysis are equal in dimensions (6 x 6 x 0.5) and are assembled to form a square plate shown in (figure 1). Plates are made of steel and have a Young's Modulus of 30 x 10^6 N/cm^2. Each plate uses one or more Triangular Elements to form the stiffness matrix of the plate. The stiffness matrix obtained for each plate is according to standard FEM procedure [Segerlind, 1984] for triangular elements.

For case 1, after the model is assembled in (15), the rigid body modes are removed by the application of Boundary conditions and Nodal forces according to (figure 1). The applied forces can be resolved into 100 N in x and y directions. Also, since node 1 has zero displacement in both x and y directions and node 3 has zero displacement in x direction only, hence the equations associated with these nodal displacements are deleted and the model is reduced to the form

$$\begin{bmatrix} 7.36 & 1.45 & -1.97 & 0.00 & 0.00 \\ 1.45 & 7.36 & 0.00 & -1.97 & 1.97 \\ -1.97 & 0.00 & 7.36 & 1.45 & -5.39 \\ 0.00 & -1.97 & 1.45 & 7.36 & -3.42 \\ 0.00 & 1.97 & -5.39 & -3.42 & 7.36 \end{bmatrix} x10^6 \begin{bmatrix} u_{2_y} \\ u_{3_x} \\ u_{3_y} \\ u_{4_x} \\ u_{4_y} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 100 \\ 100 \end{bmatrix}$$

This square symmetric matrix is inverted and nodal displacements are ascertained.

Script CJR

This script executes the Modular Modeling examples for Structural analysis Problem The user provides the Plates' dimensions and their connectivity

One Element Example

```
 XM = [0 0 6]; XS = [0 6 6];          Input of dimensions For both plates
 YM = [0 6 6]; YS = [0 0 6];
 SM =[1 0 0 0 0 0;                    Connectivity Matrix for Master Plate
       0 1 0 0 0 0;
       0 0 0 0 1 0;
       0 0 0 0 0 1];
SS = SM;                              Connectivity Matrix for Slave Plate
```

For Both Plates With 4 Elements

```
XM = [0 0 3 0 3 6]; YM=[0 3 3 6 6 6]        4 Elements in Both Plates
XS = [0 3 6 3 6 6]; YS=[0 0 0 3 3 6]
SM = [1 0 0 0 0 0 0 0 0 0 0 0 ;
       0 1 0 0 0 0 0 0 0 0 0 0 ;
       0 0 0 0 1 0 0 0 0 0 0 0 ;
       0 0 0 0 0 1 0 0 0 0 0 0 ;
       0 0 0 0 0 0 0 0 0 0 1 0 ;
       0 0 0 0 0 0 0 0 0 0 0 1 ];

SS =  [1 0 0 0 0 0 0 0 0 0 0 0;
       0 1 0 0 0 0 0 0 0 0 0 0;
       0 0 0 0 0 0 1 0 0 0 0 0;
       0 0 0 0 0 0 0 1 0 0 0 0;
       0 0 0 0 0 0 0 0 0 0 1 0;
       0 0 0 0 0 0 0 0 0 0 0 1];
```

For Master Plate With 9 Elements And Slave Plate With 4 Elements

```
XM = [0,0,2,0,2,4,0,2,4,6]; YM = [0,2,2,4,4,4,6,6,6,6];
XS = [0,3,6,3,6,6]; YS = [0,0,0,3,3,6] ;
SM = [1 0 0 0  0   0  0 0 0 0  0  0  0 0 0 0 0 0 0 0 ;
       0 1 0 0  0   0  0 0 0 0  0  0  0 0 0 0 0 0 0 0 ;
       0 0 0 0 1/2 0  0 0 0 0 1/2 0  0 0 0 0 0 0 0 0 ;
       0 0 0 0  0  1/2 0 0 0 0 0 1/2 0 0 0 0 0 0 0 0 ;
       0 0 0 0  0   0  0 0 0 0  0  0  0 0 0 0 0 0 1 0 ;
       0 0 0 0  0   0  0 0 0 0  0  0  0 0 0 0 0 0 0 1 ];

SS = [1 0 0 0 0 0 0 0 0 0 0 0;
       0 1 0 0 0 0 0 0 0 0 0 0;
       0 0 0 0 0 0 1 0 0 0 0 0;
```

```
        0 0 0 0 0 0 0 0 1 0 0 0 0;
        0 0 0 0 0 0 0 0 0 0 0 1 0;
        0 0 0 0 0 0 0 0 0 0 0 0 1];
KM = Build (XM, YM,'M');            Finds Stiffness Matrices from function Build
KS  = Build (XS, YS,'S');

[K] = Assemble (KM,KS,SM,SS)    Stiffness of Assembled Components from Assemble
```

Boundary Conditions and Solution of equations.

Compatible Plates With 1One Element In Each Plate

```
Kmodular = K([[2:2],[5:8]],[[2:2],[5:8]]);        Application of Boundary Conditions
Fmodular = [0;0;0; 100; 100];                        Application of Nodal Forces.
Umodular = Kmodular\Fmodular                      Nodal Displacements
```

Compatible Plates With Each having Four elements :-

```
Kmodular = ktransform([[1:2],[4:6],[9:18]],[[1:2],[4:6],[9:18]]);
Fmodular = zeros (15,1);
Fmodular (12,1) = 100;
Fmodular (13,1) = 100;
Umodular = Kmodular\Fmodular;
```

Finally Master Plate with 9 elements and slave with 4 elements

```
 Kmodular = K ([[1:6], [8:12], [15:26]], [[1:6], [8:12], [15:26]]);
 Fmodular = zeros (23,1);
 Fmodular (20,1) = 100;
 Fmodular (21,1) = 100;
 Umodular = kmodular\fmodular
```

The Function ASSEMBLE

```
function [KC] = Assemble( KM, KS,SM,SS)
[rowa cola] = size(KM);
[rowb colb] = size(KS);

 TM = 1:cola;
 TS = 1:colb;

[SM,KM,TM,count] = Kparse(SM,KM,TM);
[SS,KS,TS,countS] = Kparse(SS,KS,TS);

katrans = KM ;
kbtrans = KS;
ka11 = katrans([1:count],[1:count]);
ka12 = katrans([1:count],[count+1:rowa]);
ka21 = katrans([count+1:rowa],[1:count]);
ka22 = katrans([count+1:rowa],[count+1:rowa]);

kb11 = kbtrans([1:countS],[1:countS]);
kb12 = kbtrans([1:countS],[countS+1:rowb]);
kb21 = kbtrans([countS+1:rowb],[1:countS]);
kb22 = kbtrans([countS+1:rowb],[countS+1:rowb]);

[rowka11 colka11] = size(ka11);
[rowkb11 colkb11] = size(kb11);

ktransform = [ ka11              ka12                    zeros(rowka11,colkb11);
               ka21     [ka22+(SM')*(kb22)*(SM)]         (SM')*(kb21);
        zeros (rowkb11, colka11)        kb12*SM           kb11];

 KC=ktransform;
```

**************************** ************************************

The Function BUILD

```
function [k]= Build(X,Y,plate)
%      This function computes stiffness matrices of Triangular Plates with Plates'
%      dimensions given as input.
%      k = Output in the form of stiffness matrix
%      X = Input of X co-ordinate of any Triangular Plate
%      Y = Y co-ordinate of the triangular plate.
%      Plate = Master or Slave Plate

[numelements, nodesperside] = CalcElements(X);
N = connect (nodesperside, plate);
NI = N(1,:); NJ = N(2,:); NK = N(3,:); elem = length(NI);
```

**************************** *************************************

The Function CalcElements

```
function [numelements, nodesperside] = CalcElements (X)
%       function CalcElements (X)
%       This function computes the number of elements and nodes
%       in each side of plate, given the nodal Positions/total nodes.

m = length(X);          Calculates length of input vector
dummy=3;    A dummy variable introduced to map the nodal elements and dimensions.
n=1;            A counter
m = m - dummy;.
while m > 0             For more than one elements procedure is straightforward.
        dummy = dummy+1;
        m = m-dummy;
        n = n+1;
end
nodesperside = n + 1; Gives nodes on each side of plate.
numelements  = n^2   Gives the total number of elements
```

************************* *************************************

The Function Kparse

```
function [S,K,T,count]=Kparse(S,K,T)
%       Kparse parses a stiffness matrix K
%       based on a supplied constraint matrix S
%       and modifies the mapping transform vector T
[N,M]=size(K);          find the number of DOF N
[M,P]=size(T)           Is T a column vector?
if P~=1; T=T'; M=P; end       If not, make it a column vector

Test entry data
if size(T) ~= N
        disp('size(T) not equal to DOF')
        return
end
if N~=M
        disp('stiffness matrix and T do not agree')
        return
end

Parse stiffness matrix
j=1 ;   Point to end
count=0 ;
for i=1:N
        n=N-i+j;        start at last column
        if S(:,n)==0    test for no involvement in constraints S
        K=[ K(:,n)  K(:,1:n-1)  K(:,n+1:N)];  If not move nth column left to top
        K=[ K(n,:); K(1:n-1,:); K(n+1:N,:)];  and move nth row up to top
```

```
        T=[T (n);T(1:n-1);T(n+1:N)];        Also reorder transformation map T
        S=[S (: ,n) S(:,1:n-1) S(:,n+1:N)]; also move nth column  of S left to top
        j=j+1 ;                             found a constraint
        count=count+1;
        d= N-count
        end
end
 S=S (:,count+1:M);
```

*********************** ************************************

The Function Stiffnessmat

```
function K = stiffnessmat(X,Y,NI,NJ,NK,elem)
%       Stiffness Matrix Computation Function for a 2D Plate
%       X = X Coordinate of the Plate
%       Y = Y Coordinate of the Plate
%       Nn= Node Numbering for each element, n=I,J,K
%       elem = Total number of elements in the Plate

Defining System Constants
EE = 20*10^6            Modulus of Elasticity in N/cm^2
u = 0.27;               Poisson's Ratio
t = 0.5;                Thickness in cm


Kall= zeros(2*length(X),2*length(Y));       Allocation of space for Variables

for n=1:elem                                This Loop Computes 'B' used ...
        Bi(n)=Y(NJ(n)) - Y(NK(n));          in Triangular elements
        Bj(n)=Y(NK(n)) - Y(NI(n));
        Bk(n)=Y(NI(n)) - Y(NJ(n));
        Ci(n)=X(NK(n)) - X(NJ(n));
        Cj(n)=X(NI(n)) - X(NK(n));
        Ck(n)=X(NJ(n)) - X(NI(n));
end

M = [  1 X(NI(1)) Y(NI(1));
       1 X(NJ(1)) Y(NJ(1));
       1 X(NK(1)) Y(NK(1))];

A = 0.5*det(M)                  Area of each ELEMENT

DD = (EE/(1-u*u))*[ 1     u     0;        Since areas are the same for all elements
                    U     1     0;        Compute Just ANY One of them
                    0     0     (1-u)/2];

ij = 1
```

```
for n=1:elem
        BB(:,ij:ij+5) = (0.5*[Bi(n)    0       Bj(n)   0       Bk(n)        0 ;
                              0        Ci(n)   0       Cj(n)   0       Ck(n);
                              Ci(n)    Bi(n)   Cj(n)   Bj(n)   Ck(n)   Bk(n)])/A;
K(:,ij:ij+5)  = t*A*((BB(:,ij:ij+5)')*DD)*BB(:,ij:ij+5);
ij = ij+6;
end
ijk=zeros(1,6);muj = 1;

for n = 1:elem
        ijk(1) = 2*NI(n) -1;
        ijk(2) = 2*NI(n) ;
        ijk(3) = 2*NJ(n) -1;
        ijk(4) = 2*NJ(n) ;
        ijk(5) = 2*NK(n) -1;
        ijk(6) = 2*NK(n);
        Ktemp  = K(:,muj:muj+5);
                for mj=1:6
                for mi = 1:6
                        Kall(ijk(mi),ijk(mj)) = Kall(ijk(mi),ijk(mj)) + Ktemp(mi,mj);
                end
                end
        muj = muj +6;
end
K=Kall;
*********************** *******************************************
```

The Function Connect

```
function [nodalcords] =  connect(nodesperside,plate)
%       function nodalcords =  connect(nodesperside,plate)
%       This function computes nodal coordinates " N "
%       if the input is in the form of nodes on any side,
%       for both Master and Slave Plates.

switch plate
case 'M'
        nodesperside = (nodesperside-1)*(nodesperside-1);
        maxrow = sqrt(nodesperside) + 1;
        maxcol = sqrt(nodesperside) + 1;
        row = maxrow;
        col = 1;
        count  = 0;
        rownum = 1;
        coltest = 1;
        structmatM = zeros(maxrow,maxcol);
        nodalcords = zeros(3,nodesperside);
while coltest <= maxcol
```

```
for traverse = 1 : rownum
count = count  + 1;
structmatM(row,col) =   count ;
col = col + 1;
end
rownum = rownum + 1;
        if row ~= 1
        row = row -1;
        end
col = 1;
coltest = coltest + 1;
end
structmatM;
nodecol = 1;
noderow = 1;
flag = 1;
row = maxrow;
col = 1;
nodetest = 1;
        while nodetest <= nodesperside
        if flag ==1
        nodalcords (noderow,nodecol) = structmatM(row,col);
        row = row - 1;
        col = col + 1;
        noderow = noderow + 1
        nodalcords(noderow,nodecol) = structmatM(row ,col);
        noderow = noderow + 1
        col  = col - 1;
        nodalcords(noderow,nodecol) = structmatM(row,col);
if structmatM(row+1,col+1)==0
flag = 1; col = 1;
else
flag = 2;
end
elseif   flag == 2
row = row +1;
nodalcords(noderow,nodecol) = structmatM(row,col);
col = col + 1;
noderow = noderow + 1;
nodalcords(noderow,nodecol) = structmatM(row,col);
noderow = noderow + 1;
row = row - 1;
nodalcords(noderow,nodecol) = structmatM(row,col);
row = row+1;
flag = 1;
end
```

```
        nodetest = nodetest + 1;
        nodecol = nodecol + 1;
        noderow = 1;
end

otherwise
        maxrow = nodesperside;
        maxcol = nodesperside;
        row = maxrow;
        col = 1;
        count  = 0;
        rownum = maxrow;
        coltest = 1;
        structmat = zeros(maxrow, maxcol);
        elem = (nodesperside-1)^2;
        nodalcords = zeros(3,elem);
        while coltest <= maxcol
                for traverse = 1:rownum
                        count = count  + 1;
                        structmat (row, col) = count ;
                        col = col + 1;
                end
                rownum = rownum - 1;
                if row ~= 1
                        row = row -1;
                end
                coltest = coltest + 1;
                col = coltest;
        end
        structmat;      row = maxrow;       col = 1;
        noderow = 1;  nodecol=1;    nodetest = 1;
        flag = 1;
        while nodetest <= elem
                if flag == 1
                        nodalcords(noderow,nodetest) = structmat(row,col);
                        col = col + 1;
                        noderow = noderow  + 1;
                        nodalcords(noderow,nodetest) = structmat(row,col);
                        row = row -1;
                        oderow = noderow + 1;
                        nodalcords(noderow,nodetest) = structmat(row,col);
                                if col == maxcol
                                        col = (maxrow - row) + 1;
                                        flag = 1;
                                else
                                        flag = 2;
```

```
                    end
            else
            row = row + 1;
            nodalcords(noderow,nodetest) = structmat(row,col);
            col = col + 1;
            row = row -1;
            noderow = noderow  + 1;
            nodalcords(noderow,nodetest) = structmat(row,col);
            col = col - 1;
            noderow = noderow + 1;
            nodalcords(noderow,nodetest) = structmat(row,col);
            row = row + 1;
            flag = 1;
      end

      noderow = 1;
      nodetest = nodetest + 1;
      end
nodalcords;
end
```
******************** **

References

[i] Zienkiewicz, O. C., 1977, *The Finite Element Method*, McGraw Hill Inc., New York

[ii] Aminpour, M A , Ransom, J.B and McCleary, S.L ,1995, " A Coupled Analysis Method For Structures With Independently Modelled Finite Element Subdomains" Vol 38 *International Journal For Numerical Methods In Engineering*

[iii] Farhat C and Geradin M, 1992 "Using a reduced number of Lagrange Multipliers For Assembling Parallel Incomplete Field Finite Element Approximations" Vol 97 pp 333-354 *Computational Methods in Applied Mechanical Engineering*

[iv] Farhat C and Roux, F.X, 1991 "A Method Of Tearing And Interconnecting And Its Parallel solution Algorithm" Vol 32 pp 1205- *International Journal For Numerical Methods In Engineering*

[v] Byam, P.B and Radcliffe, C.J., 1999 " Modular Modeling of Engineering Systems Using Fixed Input-Output Structure" IMECE, ASME, New York.

[vi] Gosciak,G. J, 2001 "Internet Engineering Design Agents", MS thesis, Michigan State University, East Lansing, Michigan.

[vii] Radcliffe, C.J. and Sticklen, J., 2001, "Method and System for Creating Designs Using Internet-based Agents", U.S. Patent 6,295,535, Sept. 25.

[viii] B.P. Byam and C.J. Radcliffe, 1999, "The Impact of Causality on Dynamic System Modeling", ASME No. G01128, ISBN: 0-7918-1634-6 ISSN: 1522-2799, DSC-Vol. 67, ASME Internat'l Mech. Engineering Congress and Exhibition, Nashville, Nov. 14-19.

[ix]. Cook, R. D, Malkus, D. S. and Plesha, M. C., 1989, *Concepts and Applications of Finite Element Analysis*, pp 259, 3rd Ed, John Wiley & Sons, New York.

---

[x] Zienkiewicz O.C, and Taylor, R.L, 1989, *The Finite Element Method,* 4th Ed, pp 479, McGraw-Hill, UK

[xi] Segerlind, L.J 1984, *Applied Finite Element Analysis*, 2nd Ed, John Wiley& Sons, New York

[xii] Cook, R. D, Malkus, D. S. and Plesha, M. C., 1989, *Concepts and Applications of Finite Element Analysis*, p 48 3rd Ed, John Wiley & Sons, New York.