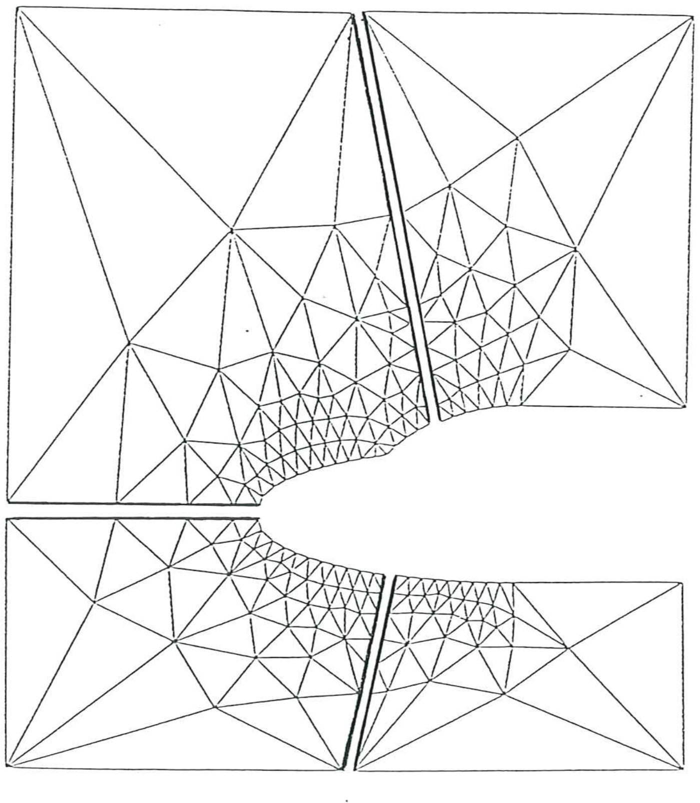
PARAMESH
Two Dimensional Unstructured Parallel Mesh Generation Program

J.C. Mestres

E. Oñate

G. Bugeda



Publication CIMNE Nº 61, March 1995

PARAMESH
Two Dimensional Unstructured Parallel Mesh Generation Program

J.C. Mestres

E. Oñate

G. Bugeda

Publication CIMNE Nº 61, March 1995

International Center for Numerical Methods in Engineering

Gran Capitán s/n, 08034 Barcelona, Spain

CONTENTS

PARAMESH. Two Dimensional Unstructured Parallel Mesh Generation Program

Index of Figures

Index of Tables

1. INTRODUCTION

The necessity to quickly generate unstructured big meshes is common use to many fields and is showing its importance in structures analysis and computational fluid mechanics.

The two main generation method families, the so-called advancing front algorithms -whereby nodes and connections are created together-, and the so-called generalized Voronoi algorithms -whereby an existing grid is improved by the introduction of nodes to be connected subsequently-, introduce just one node or just one element, according to the corresponding criterion, at each step of the process, thus, becoming essentially scalars.

On the other hand, as the introduction of a node or an element only requires checking the local neighborhood for compatibility, one may introduce many node localizations, what involves a large quantity of operations throughout the process.

Moreover, the goal of compatibility, what means not overlapping, implies the definition of some form of distance as enabling factor to achieve parallelism.

Definitely, all mentioned algorithms can take advantage of an inherent parallelism which is independent of the final size of the mesh [1].

The effective use of any parallel machine requires some general steps in order to get an optimal solution, this includes to break up the problem to be solved into pieces, to hand each processor a piece of the problem to, finally, assemble the results.

To take advantage of the advent of parallel machines, all aspects of simulation have to be ported to them and made to run efficiently to drastically reduce the time required to carry out numerical simulations.

PARAMESH is an integrated software packet for the partition of plane domains in a number of subdomains corresponding to the number of processors that were going to be used in a parallel computer (CRAY T3D or analogous), the mesh generation for each subdomain performed for everyone of the processors, and lastly, the assembly of the individual outcomes in four different output files:

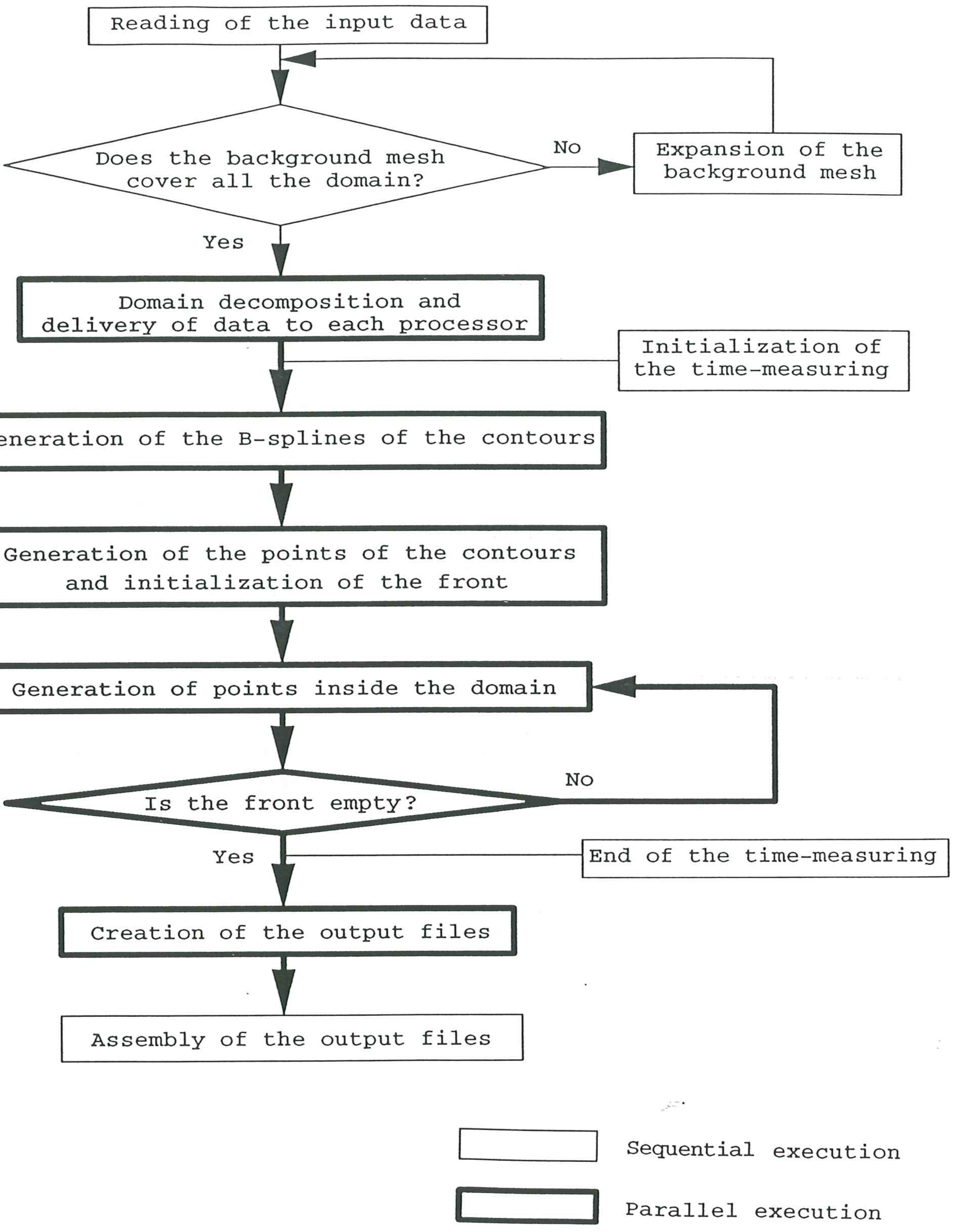
- An ASCII file (with extension .txt) that gives information of the number of generated elements and grids per processor, with the time spent by each processor to generate the mesh and the total number of generated elements and grids.

- A formatted file (with extension .fla) to be used by FLAVIA post-processor.

- A formatted file (with extension .fv) to be used by FEMVIEW program.

- A formatted file (with extension .cal) to be used, when different loads and constraints are imposed, by CALSEF program.

The outlines of PARAMESH are showed in the figure 1, which represents the flowchart of this program, pointing out the parts of the program that are executed by a single processor (serial execution) and the parts that are performed by all of them (parallel execution), as well as the beginning and the end of the section of the program whose CPU time is measured, which corresponds to the explicit mesh generation.



Reading of the input data
Does the background mesh cover all the domain?
No
Expansion of the background mesh
Yes
Domain decomposition and delivery of data to each processor
Initialization of the time-measuring
Generation of the B-splines of the contours
Generation of the points of the contours and initialization of the front
Generation of points inside the domain
Is the front empty?
No
Yes
End of the time-measuring
Creation of the output files
Assembly of the output files
Sequential execution
Parallel execution

Figure 1. PARAMESH flow chart. This diagram shows the processes performed in parallel and in sequential execution and the section of the program whose CPU time is measured.

2. TECHNICAL DESCRIPTION

PARAMESH uses the so-called B-splines curves to design the contours that define the global domain.

Upon each point in the definition of the B-spline it can be imposed several types of condition according to the degree of continuity that is to be imposed on the curve.

The maximum degree of continuity that may be attained is C2, that is, the slope and curvature are continuous at the point in question. This degree of continuity can be reduced to C1 or C0, but in these cases it is necessary to impose an additional condition for each degree of continuity that is reduced with respect to C2. These conditions are imposed upon the slope or the curvature which the curve is expected to have at that point. In the same way, the conditions can precede or follow the point in the sense that they are imposed on the curve in the interval preceding or following the point.

The conditions upon the slope require two pieces of information:
- the slope (in degrees)
- a multiplication factor f such that the derivatives of the curve in its parametric form x'(t), y'(t), are equal to f*sin(p) and f*cos(p), respectively.

The conditions upon the curvature only allow the imposition of null curvature preceding and/or following. For this, it is necessary to provide a factor h such what is actually imposed:

$$x''(t)-h*x'(t)=0$$
$$y''(t)-h*y'(t)=0$$

From this, it may be obtained the following, which is in fact what is used:

$$x'(t)*y''(t)-x''(t)*y'(t)=0$$

This ensures null curvature at the point. Normally, null values are used for h, although on occasions, interpolated curves more suitable for the geometry can be obtained by its variation.

Once the contours have been defined, PARAMESH splits the domain into the desired number of subdomains. There are different ways of doing that; in order to simplify the process, a simple way will be shown in the examples.

However, the chosen method allows to design what subdomains are to be constructed through the definition of the common borders and in that way, adaptive mesh generation and remeshing will be possible thanks to different spacing definitions.

Next, PARAMESH makes up the unstructured meshes, using the advancing front technique.

The process begins reading the name of the data file that contains all needed information for the mesh generator. At the end, PARAMESH will name all of the generated files with the name of the invoked data file filename and their corresponding extension.

This report will consider an academic domain composed of a squared area with a squared hole inside to compare the different times.

Examples will be run on the original domain -figure 2- and on the domains, shown by figures 3 and 4, created by geometric bisections from the first, and whose results will then be compared amongst them and with the sequential ones, too.

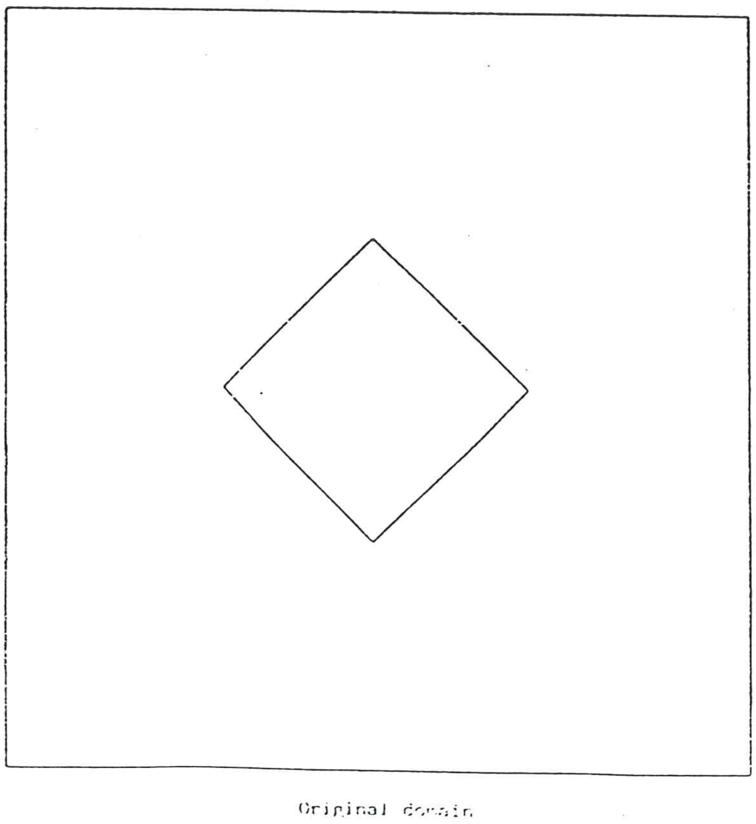


Figure 2. Original reference domain.

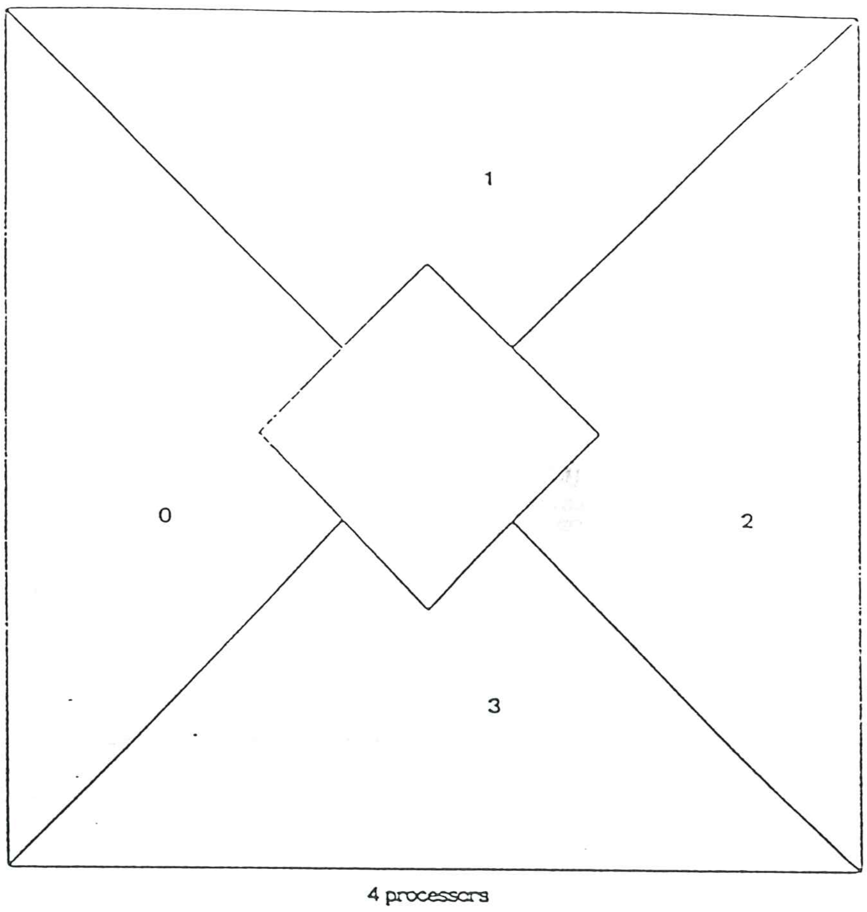


Figure 3. Original reference domain splitted in four subdomains.

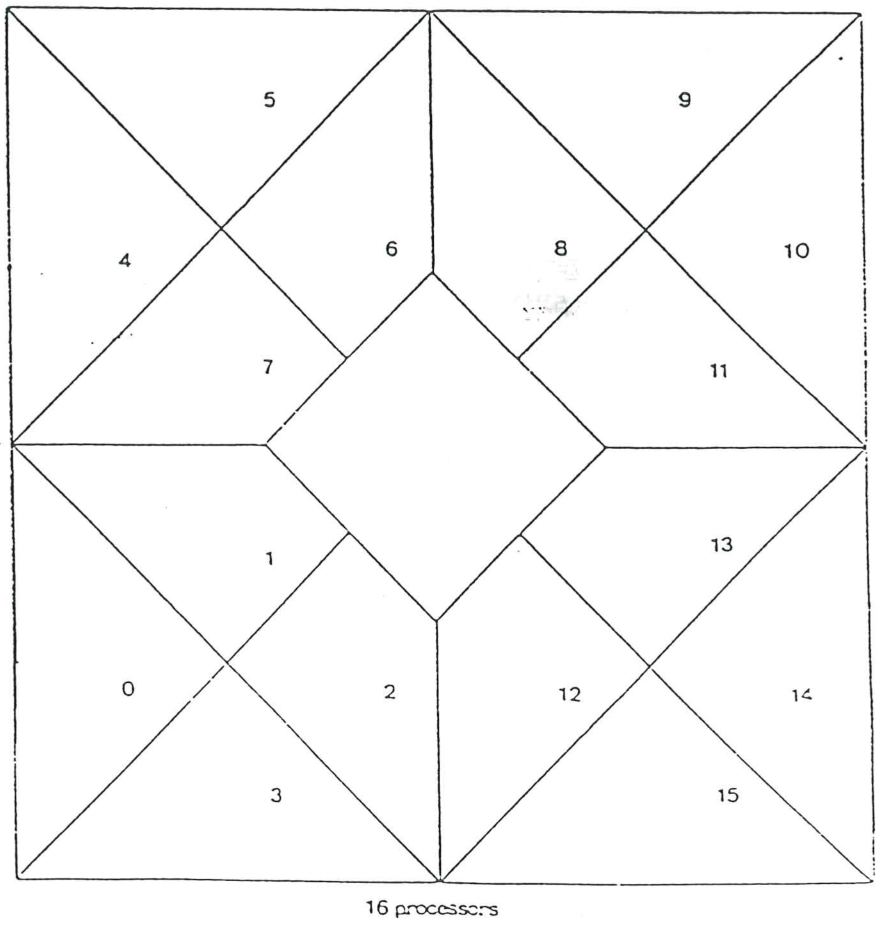


Figure 4. Original reference domain splitted in sixteen subdomains.

2.1. Structure of the data file

The data which contain all needed information, supplied by the data file read at the beginning of the process, are described in the following paragraphs.

There are various groups of records, some of which must be repeated several times.

There are various records throughout the file that are not interpreted and these are called free. Comments and mnemotechnical help may be written upon them to aid interpretation of the data file. In the shown examples of a data file, these lines are introduced by the string 'GROUP'.

In this way it is possible to construct a general template for the data file making it easier to fill in the information required by the program.

All read lines, apart from the first and two last groups, are read in free format to facilitate the construction of this file.

The complete form of the first case shown in the examples is listed next.

This file contains a lot of information relative to structural constraints and materials properties that are used to calculate stresses and strains of different kinds of structures, which are kept for compability with other various programs. (Group 9, for instance, informs about the possibility of structured layers circumscribing the different domains, and not even has to be included when unnecessary).

On the other hand, many of them can be supressed in these examples for the mesh generation. This fact, with the possibility of an easy simulation of the contour allows a very simple introduction of the data, which is shown after the explanation of all the parameters introduced.

GROUP 1: Number of processors and name of the model, used to name output files.

```
    4BIG1
    GROUP 2:     mxelem     mxpoin     mxside
                   2000       2000       2000
    GROUP 3:     nnode      neleb      npoib      ndeft
                   3          2          4          1
    GROUP 4:     ndofn      nmats      nprop
                   2          1          2
    GROUP 5:     ncfix      ncapa
                   1          0
```

GROUP 6: Number of referencing nodes and curves of the contour.

```
     12   2
    GROUP 7: Coordinates.
      1   0.      0.
      2   0.     70.
      3  70.     70.
      4  70.      0.
      5  35.     21.
      6  21.     35.
      7  35.     49.
      8  49.     35.
      9  28.     28.
     10  28.     42.
     11  42.     42.
     12  42.     28.
    GROUP 8: First curve (=outer square).
      4   2   1   1
    GROUP: Characteristics.
      1   5   2    0.
      4   5   2    0.
      3   5   2    0.
      2   5   2    0.
    GROUP: Second curve (=inner square).
      8   2   1   1
    GROUP: Characteristics.
      5   5   2   0.
     12   5   2   0.
      8   5   2   0.
     11   5   2   0.
      7   5   2   0.
     10   5   2   0.
      6   5   2   0.
      9   5   2   0.
```

```
GROUP  10:  Materials  properties.
 1.0E+06  0.3E+00
GROUP  11:  Constraints  conditions.
   0   2
   1   2   1   2      0.      0.      0.      0.
   1   1   1   2      0.      0.      0.      0.
GROUP  12:  Background  mesh  coordinates.
   1      -10.     -10.     27.      1.      1.      0.
   2      -10.     100.     19.      1.      1.      0.
   3      100.     100.     11.      1.      1.      0.
   4      100.     -10.      3.      1.      1.      0.
GROUP  13:  Background  mesh  connections.
   1   1   3   2
   1   1   4   3
GROUP  14:  Points  per  processor.
  7   7   7   7
GROUP  15:  Referenced  points.
  0   1   2 -10   0  10   9
  0   2   3  11   0  11  10
  0  -3   4  12   0  12  11
  0  -4   1  -9   0   9  12
```

2.1.1. Group 1. Number of processors and name of the model.

- free record

- NPRCS, TITLE
Format(I2,A6).

NPRCS: number of processors.

TITLE: name of the model, that will name the different output files: (TITLE.txt, TITLE.fla, TITLE.fv and TITLE.cal).

This group has a defined format as the program has to check that the name given to the data file has not more than six characters, as well as the number of processors has not more than two digits (the program also verifies it does not surpass 64, which is the maximum number allowed and that is a power of 2).

2.1.2. Group 2. Maximum number of elements, nodes and sides.

- free record

- MXELEM MXPOIN MXSIDE

MXELEM: maximum number of elements that can be generated for each processor.

MXPOIN: maximum number of points that can be generated for each processor.

MXSIDE: maximum number of sides that can be generated for each processor.

2.1.3. Group 3. Nodes per element and number of elements and nodes of the background mesh.

- free record

- NNODE NELEB NPOIB NDEFT

NNODE: number of nodes to use for each element and for the background mesh.

NELEB: number of elements in the background mesh.

NPOIB: number of points in the background mesh.

NDEFT: to indicate where the sizes of the elements in the background mesh are defined. These can be defined on the nodes (NDEFT=1) or upon the elements (NDEFT=2).

2.1.4. Group 4. Number of materials and properties.

- free record

- NDOFN NMATS NPROP NGENTYP NOFFS

NDOFN: number of degrees of freedom per node.

NMATS: number of different materials.

NPROP: number of properties characterising a material.

2.1.5. Group 5. Number of problems and layers.

- free record

- NCFIX NCAPA

NCFIX: number of sets of constraints.

NCAPA: if=1, layer elements are generated and layer information will be read.

2.1.6. Group 6. Number of referencing nodes and curves of the contour.

- free record

- NCOOT NCONT

NCOOT: number of points in the definition of the geometry.

NCONT: number of curves in the definition of the geometry.

2.1.7. Group 7. Coordinates of the referencing nodes of the contour.

- free record

The following record is repeated NCOOT times:

-JCOOT COORT(1,JCOOT) COORT(2,JCOOT)

JCOOT: number of the point to which the coordinates refer to.

COORT(1,JCOOT): x-coordinate.

COORT(2,JCOOT): y-coordinate.

2.1.8. Group 8. Number of referencing nodes and characteristics of the different curves of the contour.

This group is repeated NCONT times:

- free record

-NCOOR INDAC INDLA MATN1 MATN2

NCOOR: number of points that define this curve.

INDAC: indicates whether the curve is open (1) or closed (2).

INDLA: indicates whether the curve is a contour (1) or an interior (2) curve. In contour curves the elements are generated facing in the direction in which the points are provided, on the right-hand side.

MATN1: type of material of the elements placed to the right of the curve according to the sense of its direction.

MATN2: type of material of the elements placed to the left of the curve according to the sense of its direction. This datum is only necessary when INDLA=2.

- free record

The following record is repeated NCOOR times:

- JCOOR INDCO(1,JCOOR) INDCO(2,JCOOR) VALO1 VALO2 VALO3 VALO4

JCOOR: number of the point through which the curve passes.

INDCO(1,JCOOR): type of the imposed condition upon the continuity of the curve at this point (see following page).

INDCO(2,JCOOR): number of points of control in the interval between this point and the next one (minimum 2). These points are positioned within the interval in a way that they correspond to equal variations of the parameter t -arc's length- in the definition of the curve.

VALO1,VALO2: first condition on the curve for the cases that need it.

VALO3,VALO4: second condition on the curve for the cases that need it.

Definitions of types of points:

INDCO(1,JCOOR)	CONT. CONDITIONS	VALO1	VALO2	VALO3	VALO4
1	C2				
2	C1 Null curv. before	h	0		
3	C1 Null curv. after	h	0		
4	C1 Fixed slope	p	f		
5	C0 Null curv. before & Null curv. after	h1	0	h2	0
6	C0 Null curv. before & Fixed slope after	h1	0	p2	f2
7	C0 Fixed slope before & Null curv. after	p1	f1	h2	0
8	C0 Fixed slope before & Fixed slope after	p1	f1	p2	f2

In the case of open curves (INDAC=1), there are only two possible situations at the first and last points and these are the following:

INDCO(1,JCOOR)	CONT. CONDITIONS	VALO1	VALO2	VALO3	VALO4
1	- Null curvature	h	0		
2	- Fixed slope	p	f		

2.1.9. Group 9. Number and conditions of the layers.

- free record

- NCUCAP

NCUCAP: number of layer records.

The following two records are repeated NCUCAP times:

- LSEGNR(i) LEXT(1,i) LEXT(2,i) LCNORM(i) LCAPAS(i)

LSEGNR(1:NCUCAP): curve number with layer.

LSEGNR(1:NCUCAP): curve number with layer.

LEXT(1,1:NCUCAP): beginning point of the layer.

LEXT(2,1:NCUCAP): end point of the layer.

LCNORM(1:NCUCAP): type of the generation algorithm for the layer points.
=2: standard algorithm
=1: improved algorithm (e.g. for sharp edges).

LCAPAS(1:NCUCAP): number of layers.

- DISCAP(1:LCAPAS(i),1:NCUCAP)

DISCAP(1:LCAPAS(i),1:NCUCAP): distance of every single layer level measured in normal direction from the basic curve.

2.1.10. Group 10. Materials properties.

- free record

The following record is repeated NMATS times:

- JMATS PROPS(1,JMATS) PROPS(2,JMATS) PROPS(3,JMATS) PROPS(4,JMATS) PROPS(NPROP,JMATS)

JMATS: number of the material.

PROPS(1:NPROP,JMATS) : properties of the material.

2.1.11. Group 11. Constraints conditions.

This group is made up of the instructions for defining the contour's conditions. There are NCFIX sets of different constraints, but all of them must have the conditions defined upon the same points and upon the same degrees of freedom. The only values that can vary are the prescribed ones for the constraints. The instructions are the following:

- free record

- NPOIC NLADS

NPOIC: number of constraints upon specific points.

NLADS: number of constraints defined upon sections of the curve.

The following record is repeated NPOIC times (or if NPOIC=0, it is ignored):

- NNODC NLIBE (VALOR(ICFIX), ICFIX=1,NCFIX)

NNODC: number of the node restricted.

NLIBE: number of the degree of freedom restricted (1 or 2).

VALOR(ICFIX): value of the displacement imposed for the set of constraints ICFIX.

The following record is repeated NLADS times (or if NLADS=0, it is ignored):

- NCURV NLIBE NODO1 NODO2 (VAL1(ICFIX), VAL2(ICFIX), ICFIX=1,NCFIX)

NCURV: number of the curve where the constraints are defined.

NLIBE: number of the degree of freedom restricted (1 or 2).

NODO1: number of the point where the constraint begins (global numbering).

NODO2: number of the point where the constraint finishes (global numbering).

VAL1(ICFIX): value of the displacement imposed for the set of constraints ICFIX at NODO1.

VAL2(ICFIX): value of the displacement imposed for the set of constraints ICFIX at NODO2.

The value of the constraint at each point of the curve placed between NODO1 and NODO2 is linearly interpolated within the arc. The curve is always run in the sense in that which has been defined from NODO1 to NODO2. NODO1 and NODO2 are related to the global numbers of the points of definition of the curve, it is, to the number which coordinates are stored in COORT.

2.1.12. Group 12. Background mesh coordinates.

- free record

The following record is repeated NPOIB times (if NDEFT=2, DACON is ignored):

- IPOIB COORB(1,IPOIB) COORB(2,IPOIB) (DACON(I,IPOIB),I=1,4)

IPOIB: number of the point in the background mesh.

COORB(1,IPOIB): x-coordinate of point IPOIB in the background mesh.

COORB(2,IPOIB): y-coordinate of point IPOIB in the background mesh.

DACON(1,IPOIB): size of element defined at the point.

DACON(2,IPOIB): coefficient of stretching.

DACON(3,IPOIB): cosine of the angle to define the stretching.

DACON(4,IPOIB): sine of the angle to define the stretching.

It is important to point out that is not mandatory for the background mesh to cover all the domain. PARAMESH itself checks whether it does or not and in this case, it expands the background mesh in order to contain all the area to be meshed.

2.1.13. Group 13. Background mesh connections.

- free record

The following record is repeated NELEB times (if NDEFT=1, DACON is ignored):

- IELEB (LNODB(INODE,IELEB),INODE=1,NNODE) (DACON(I,IELEB), I=1,4)

IELEB: number of the element of the background mesh.

IELEB: number of the element of the background mesh that is being defined.

LNODB(INODE,IELEB): nodes of the background mesh that belong to this element. They must be given in an anti-clockwise sense, as many for the elements of 3 nodes as for those of 6.

DACON(1,IELEB): size of the element defined at the point.

DACON(2,IELEB): coefficient of stretching.

DACON(3,IELEB): cosine of the angle to define the stretching.

DACON(4,IELEB): sine of the angle to define the stretching.

2.1.14 Group 14. Points per processor.

- free record

- NPTSP

- NPTSP (NPRCS): number of points to define the geometry of the different subdomains.
Format (NPRCS(I4))

This group has a defined format as PARAMESH has to look for the coordinates of the corresponding points for each subdomain.

2.1.. Group 15. Referenced points.

- free record

The following record is repeated NPRCS times:

- NVRTS
Format ((NPTSP(IPRCS))(I4))

This group has a defined format as PARAMESH generates the contour points between two extremes of a segment of the initial advancing front in the established order.

- NVRTS: number of the vertices used to define the geometry.

The number of the vertices used to define the geometry -including virtual points quoted '0'-, of the different subdomains, according to the following criterion:

Every pair of consecutive points indicates the origin and end of a segment of the subdomain's contour to be meshed.

They are quoted in the advancing sense of the advancing-front technique, it is, clockwise sense for exterior borders, and anti-clockwise sense for interior ones.

However, for compatibility between common 'exterior' boundaries, shared by different subdomains, each one of these boundaries has to be meshed in opposite sense in one subdomain and in the other.

To quote this, the anti-clockwise sense will be pointed by the end for the advancing front (clockwise sense, origin of the meshing segment, therefore) as negative.

In the example, as can be seen in figure 5, the segments 2 to 10, 11 to 3, 12 to 4 and 1 to 9, accomplish that fact, so, for these segments, 10, 3, 4 and 9 must be quoted as negative.

When defining exterior parts of the boundaries, the origin and the end of these can represent a polyline, implying more than two points to be defined. In this case, a '0' ought to precede the couple of extremes. In this way, the program knows when has to seek the definition of the curve that modelizes the border and therefore it can simulate the contour by B-splines.

However, PARAMESH considers straight lines if none zero precedes a couple of numbers (what happens, for instance, in the interior boundaries of subdomains), and consequently, if the contour is composed by only straight lines -as the example-, it will not be necessary to search the definition of the exterior lines and the program will treat these points as extremes of segments.

It allows to eliminate the related records to obtain a data file more simple than the one previously seen, shown in the following page.

```
  Nprcs & Title
   4BIG1
  Maximum number of elements, points and sides
  2000  2000  2000
  Nodes/element,  background  mesh:  elements,  points  and  size
location
  3 2 4 1



  Nodes and curves
  12  0
  Coordinates
   1    0.    0.
   2    0.   70.
   3   70.   70.
   4   70.    0.
   5   35.   21.
   6   21.   35.
   7   35.   49.
   8   49.   35.
   9   28.   28.
  10   28.   42.
  11   42.   42.
  12   42.   28.


  Constraints
   0  0
  Background  mesh  coordinates
   1      -10.      -10.      27.       1.       1.       0.
   2      -10.      100.      19.       1.       1.       0.
   3      100.      100.      11.       1.       1.       0.
   4      100.      -10.       3.       1.       1.       0.
  Background  mesh  connections
     1   1   3   2
     1   1   4   3
  Points  per  processor
     7   7   7   7
  Referenced  points
    -1   2 -10   6   9
     2   3 -11   7  10
     3   4 -12   8  11
     4   1   9   5  12
```

This example corresponds to the domain decomposition of the figure 5, where the reference points and the nodes generation sense -arbitrary- on the interior curves are shown.

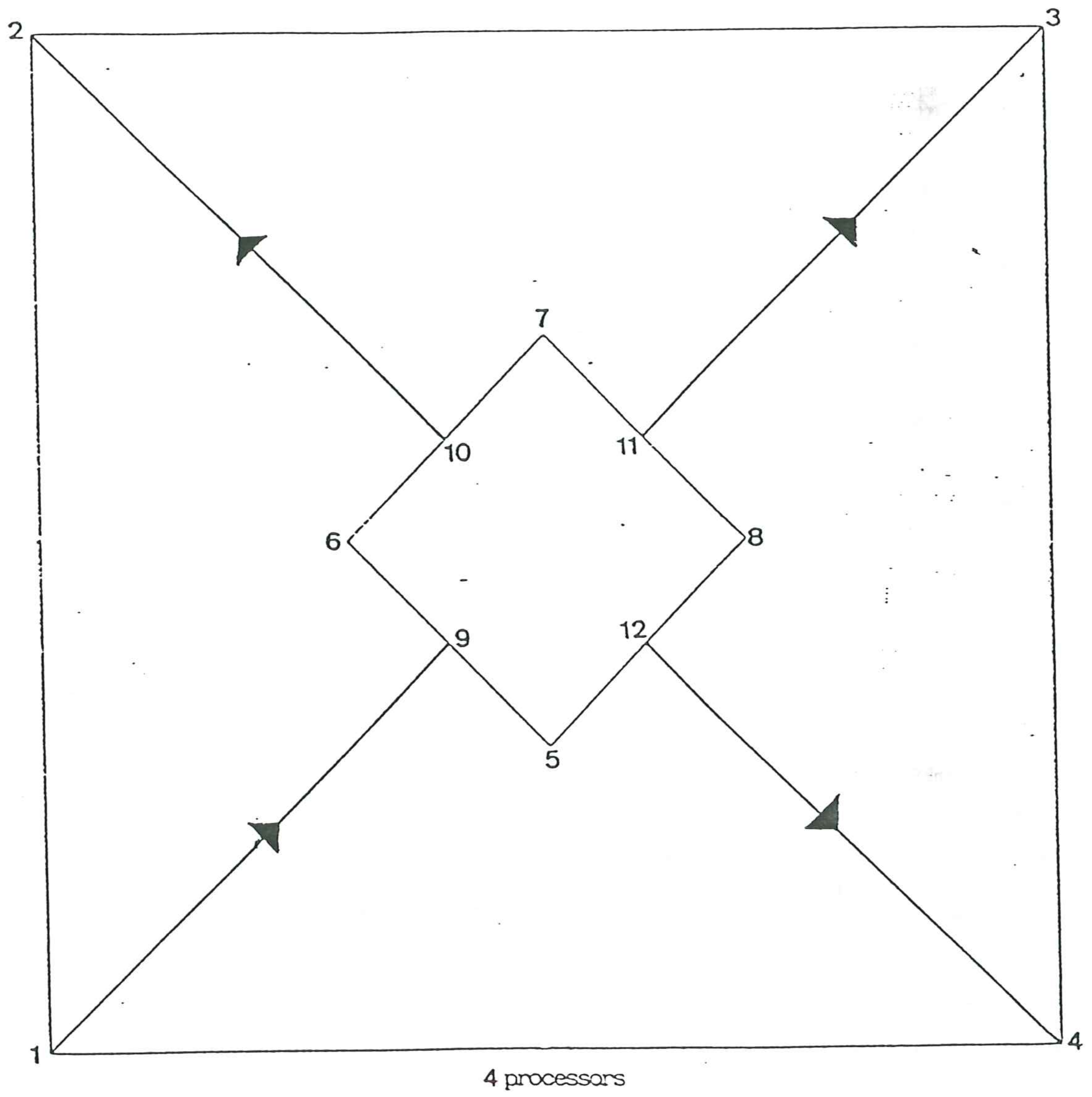


Figure 5. Domain decomposition and nodes generation sense.

3. RESULTS

With all the geometric conditions introduced (as well as structural conditions if the input file for the CALSEF program is desired) we are in disposition to run PARAMESH.

The input data file presented before corresponds to the first model to be studied.

This model, called BIG1 -shown in figure 6- will give an idea of the goodness of this initial domain-decomposition. It is advisable to do a first rough estimation of the elements share and, consequently, remake the subdomains when balance among different processors is not good enough.

After these first results, PARAMESH will be run on the same general domain with new spacing definitions, what will give models BIG2 and BIG5, obtained dividing all first spacings by 2 and 5, respectively.

Later on, the first spacings -the ones related to the model BIG1- will be divided by 10, obtaining a new reference model, which will be called SMALL1. Dividing its spacings again by 2 and 5, we will obtain models SMALL2 and SMALL5, obtained dividing all the spacings by 2 and 5, respectively.

All the results will be obtained using a parallel machine CRAY T3D.

As it was said, this method can serve to define adaptive meshes, depending on the elements distribution around the domain to get a bigger concentration of subdomains where a bigger element concentration is to be obtained.

A similar number of elements generated by each processor can be achieved by doing so.

3.1. Models BIG1, BIG2 and BIG5.

For the different number of processors, this test shows a good balance.

This approximation, although the time spent by processor number 3 when using four processors is clearly bigger than the times spent by the other processors, the overall performance can be supposed acceptable to our objectives.

However, the time required with sixteen processors is greater than with four processors for the models BIG1 and BIG2. This amazing result can be explained as a bigger number of iterations in the calculus of the generation of the contours and the generation of nodes on them (what should be of relatively insignificant importance faced to the interior nodes and elements generation -what in fact absorbs the practical totality of the CPU time in most cases) has a decisive weight here, as the number of elements is very small.

In successive figures, after the numerical results for the three cases -sequential program, run with four processors and run with sixteen processors-, the resulting meshes are shown for all the models, as well as the mesh for every single processor when four processors are used and the assembly in groups of four when sixteen processors are used when it refers to model BIG1.

It is important to emphasize that in spite of being generated independently by each processor, the nodes on the coincident contours are the same, as PARAMESH generates them in the same sense independently of the sense of the advancing front, and in this way, it can avoid a great transfer of information among processors, what means a great saving of CPU time.

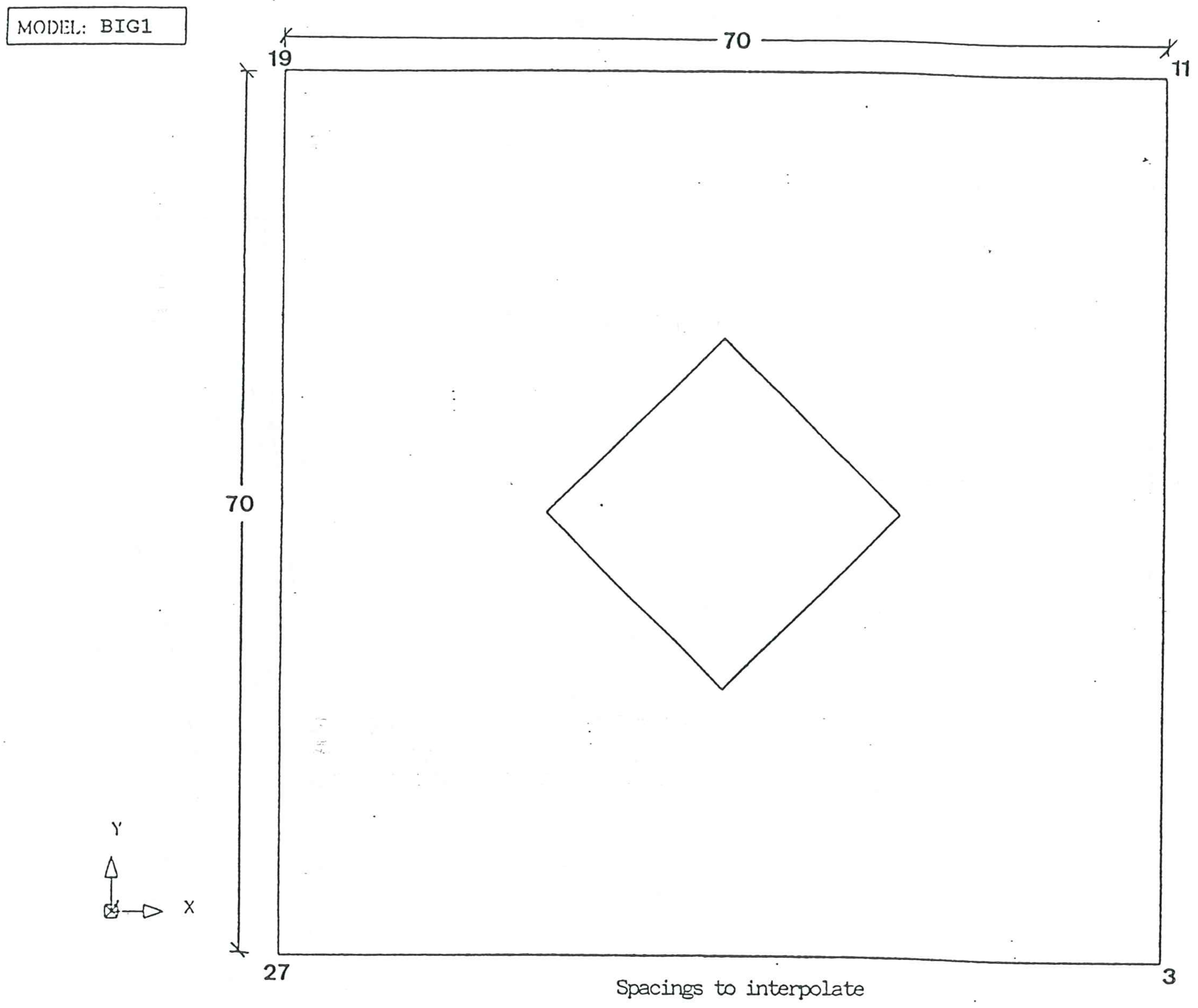
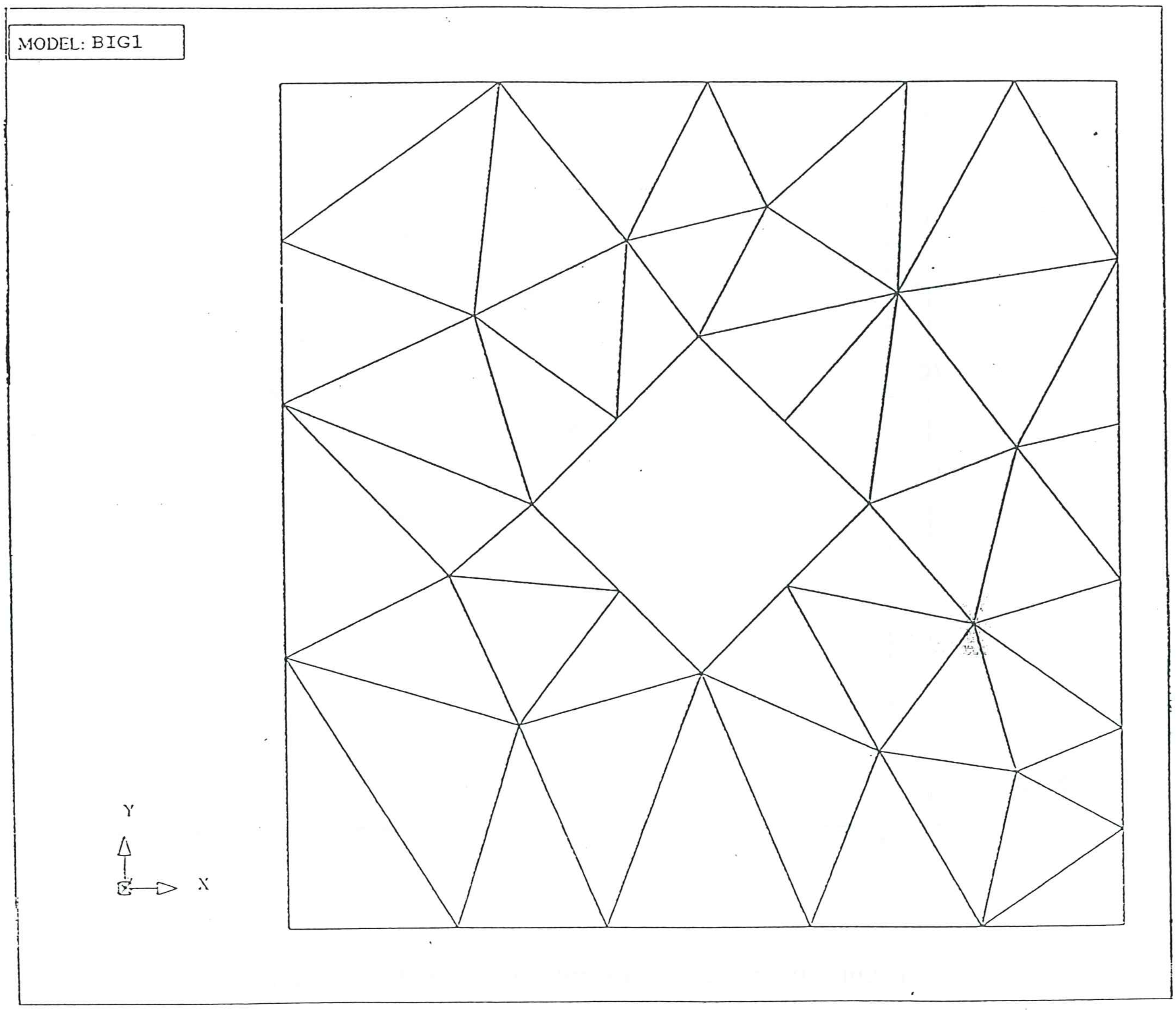


Figure 6. Spacings to interpolate (Model BIG1)

```
MODEL BIG1 (Sequential)

Number of generated elements=        48
Number of generated nodes=           38

                The time spent to generate the mesh has been         82 msec
```



Sequential program (48 elements, 38 nodes)

Figure 7. Sequential program (Model BIG1)

```
 MODEL BIG1    04 PROCESSORS

 ::::::::::::
 processor 00
 ::::::::::::
 Number of generated elements=    14
 Number of generated nodes=       14
                        Time spent by processor  0 has been     12 msec
 ::::::::::::
 processor 01
 ::::::::::::
 Number of generated elements=    16
 Number of generated nodes=       16
                        Time spent by processor  1 has been     20 msec
 ::::::::::::
 processor 02
 ::::::::::::
 Number of generated elements=    20
 Number of generated nodes=       19
                        Time spent by processor  2 has been     24 msec
 ::::::::::::
 processor 03
 ::::::::::::
 Number of generated elements=    17
 Number of generated nodes=       17
                        Time spent by processor  3 has been     15 msec
 Total number of generated elements=    67
 Total number of generated nodes=       50

 Maximum time spent to generate the mesh by a single processor=        24 msec
 Total CPU time spent to generate the mesh=                            71 msec
```

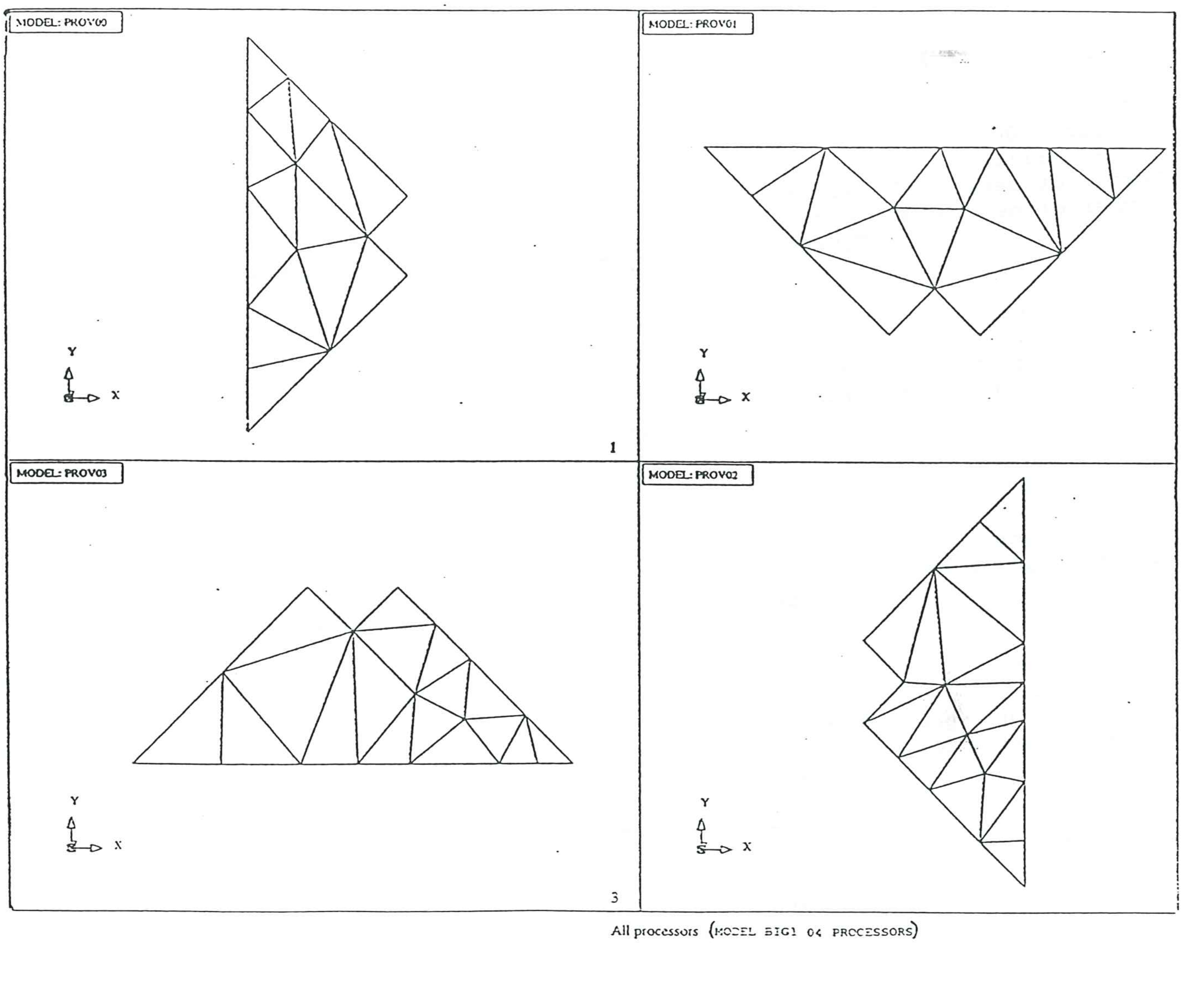


Figure 8. All Processors (Model BIG1, 4 Processors)

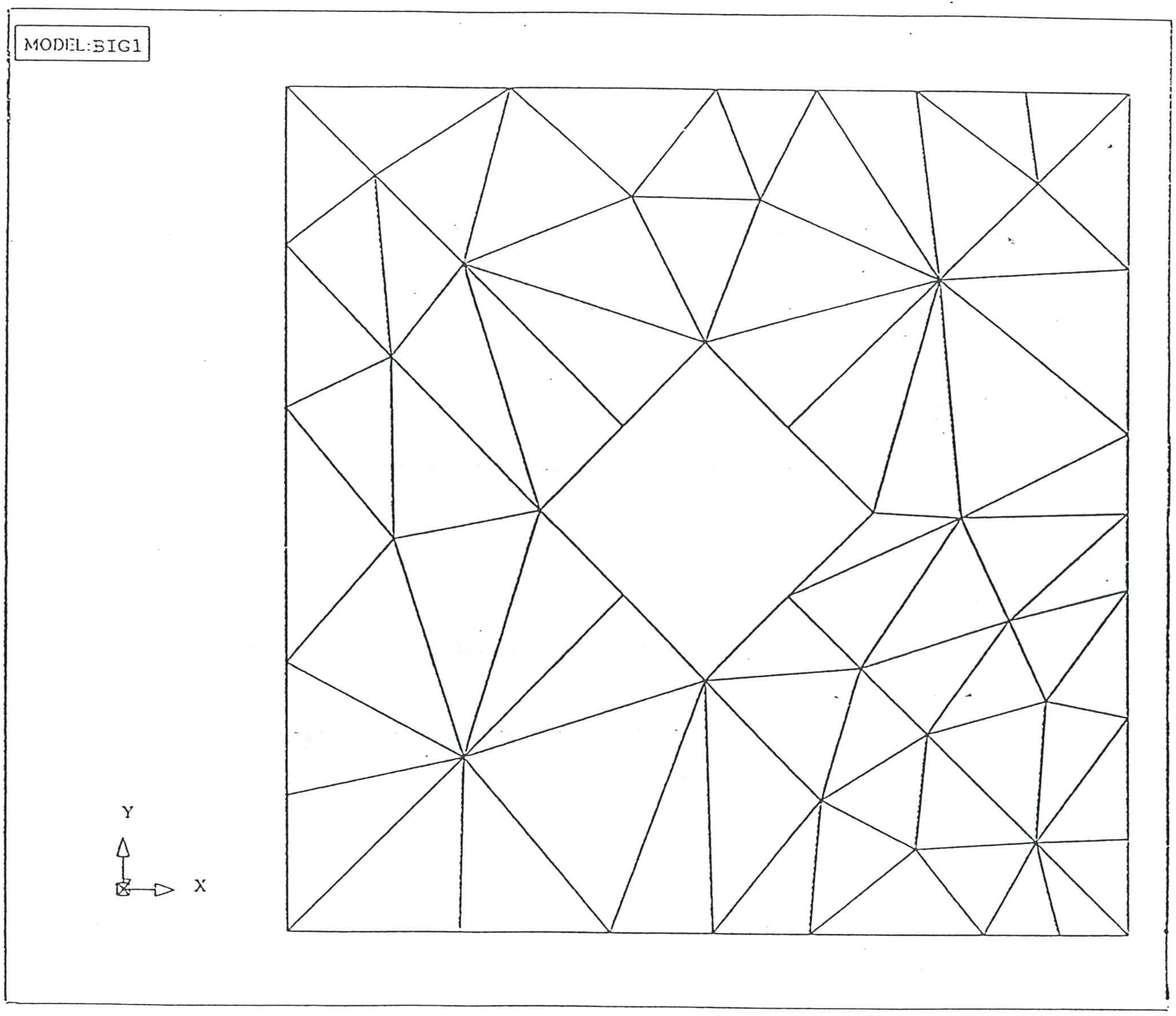


Figure 9. 4 Processors (Model BIG1)

MODEL BIG1   16 PROCESSORS

```
::::::::::::
processor 00
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  0 has been      5 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  1 has been      5 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  2 has been      5 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  3 has been      5 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  4 has been      4 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  5 has been      4 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  6 has been      4 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                       Time spent by processor  7 has been      5 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                    Time spent by processor  8 has been      5 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=      5
Number of generated nodes=         7
                    Time spent by processor  9 has been      6 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=      5
Number of generated nodes=         7
                    Time spent by processor 10 has been      6 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=      6
Number of generated nodes=         7
                    Time spent by processor 11 has been      5 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=      4
Number of generated nodes=         6
                    Time spent by processor 12 has been      5 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=      6
Number of generated nodes=         7
                    Time spent by processor 13 has been      5 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements=      7
Number of generated nodes=         8
                    Time spent by processor 14 has been      6 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=      8
Number of generated nodes=         9
                    Time spent by processor 15 has been      6 msec

Total number of generated elements=     77
Total number of generated nodes=        53

Maximum time spent to generate the mesh by a single processor=        6 msec
Total CPU time spent to generate the mesh=                           81 msec
```

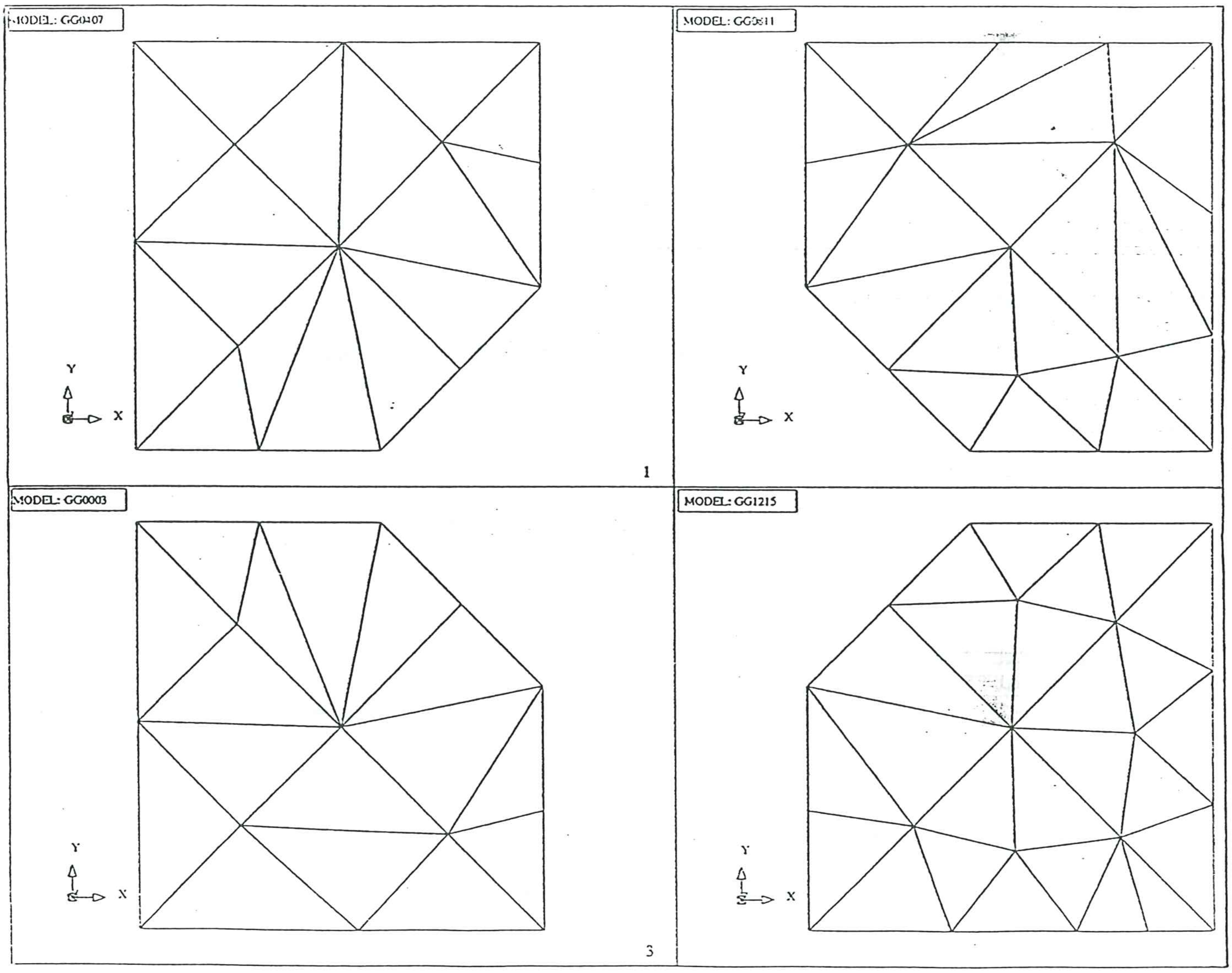


Figure 10. Groups of 4 Processors (Model BIG1, 16 Processors)

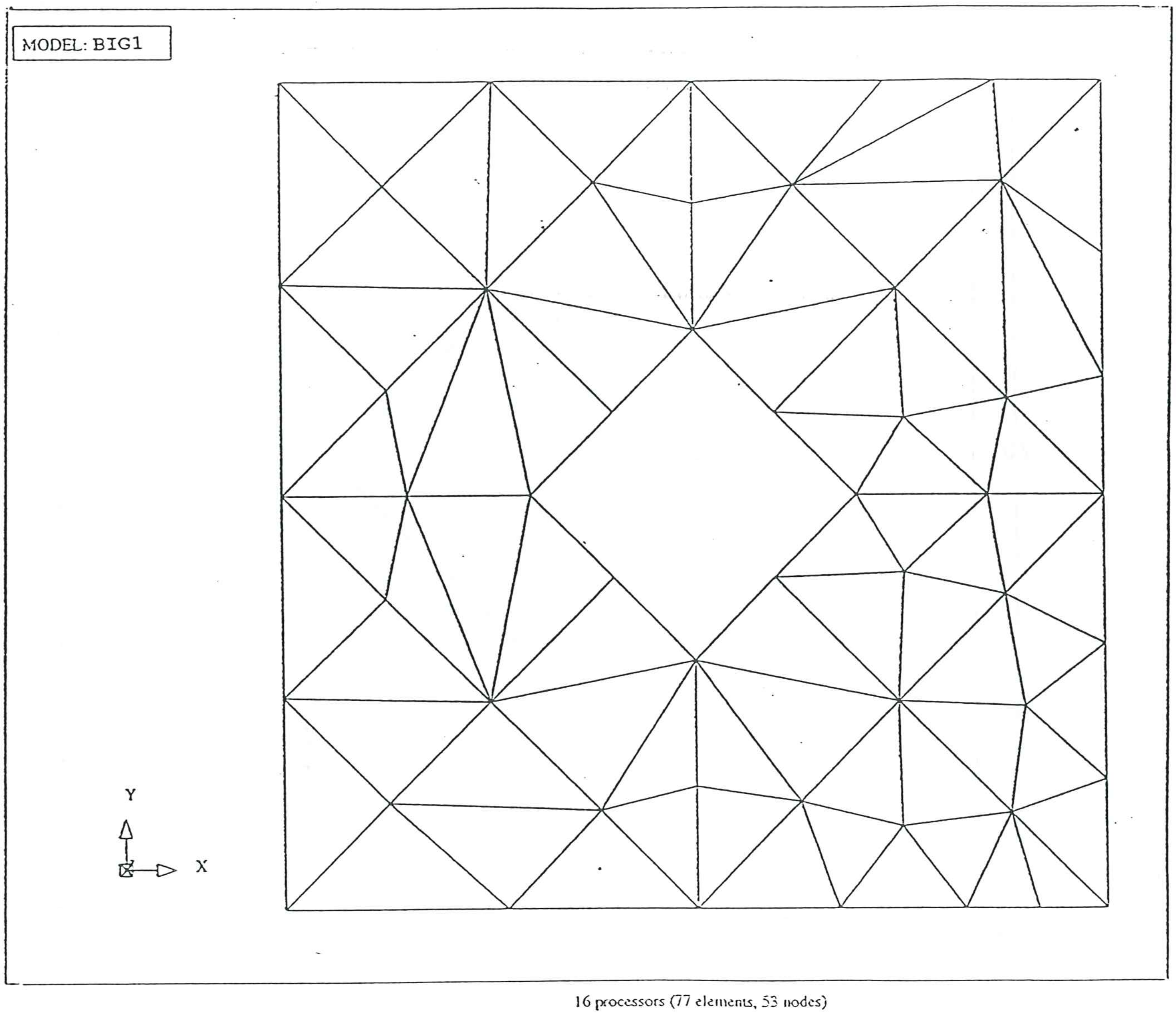


Figure 11. 16 Processors (Model BIG1)

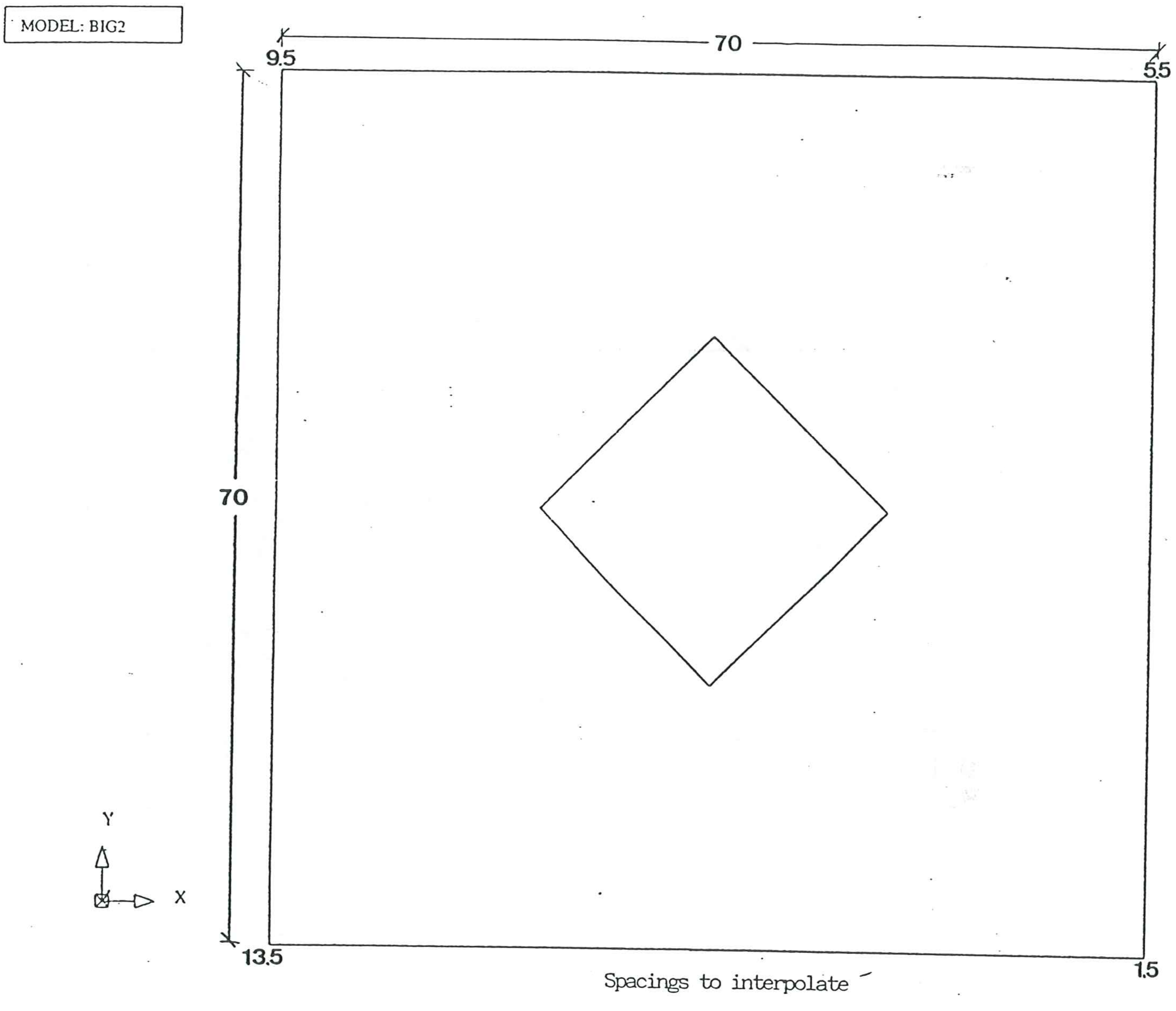
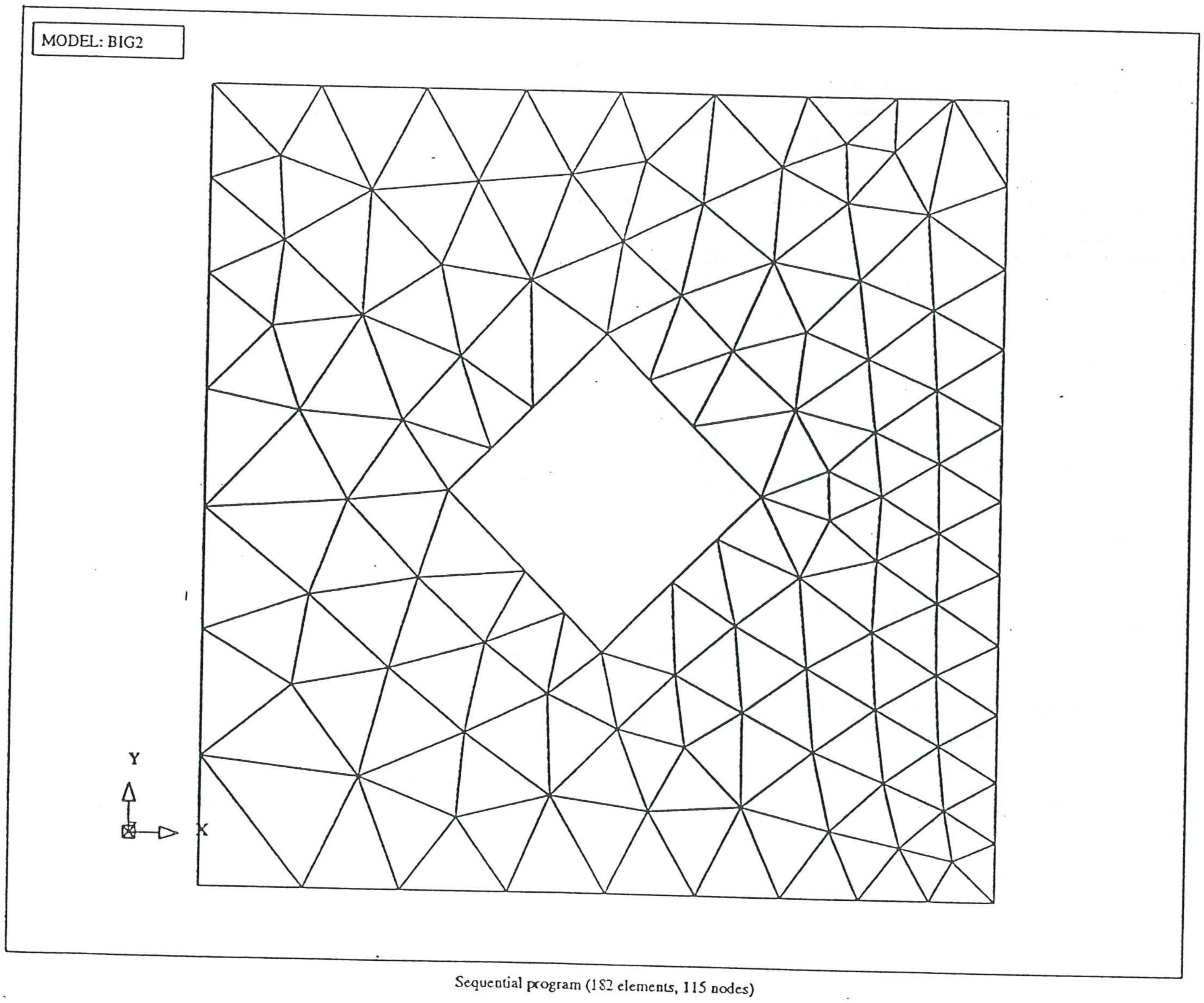


Figure 12. Spacings to interpolate (Model BIG2)

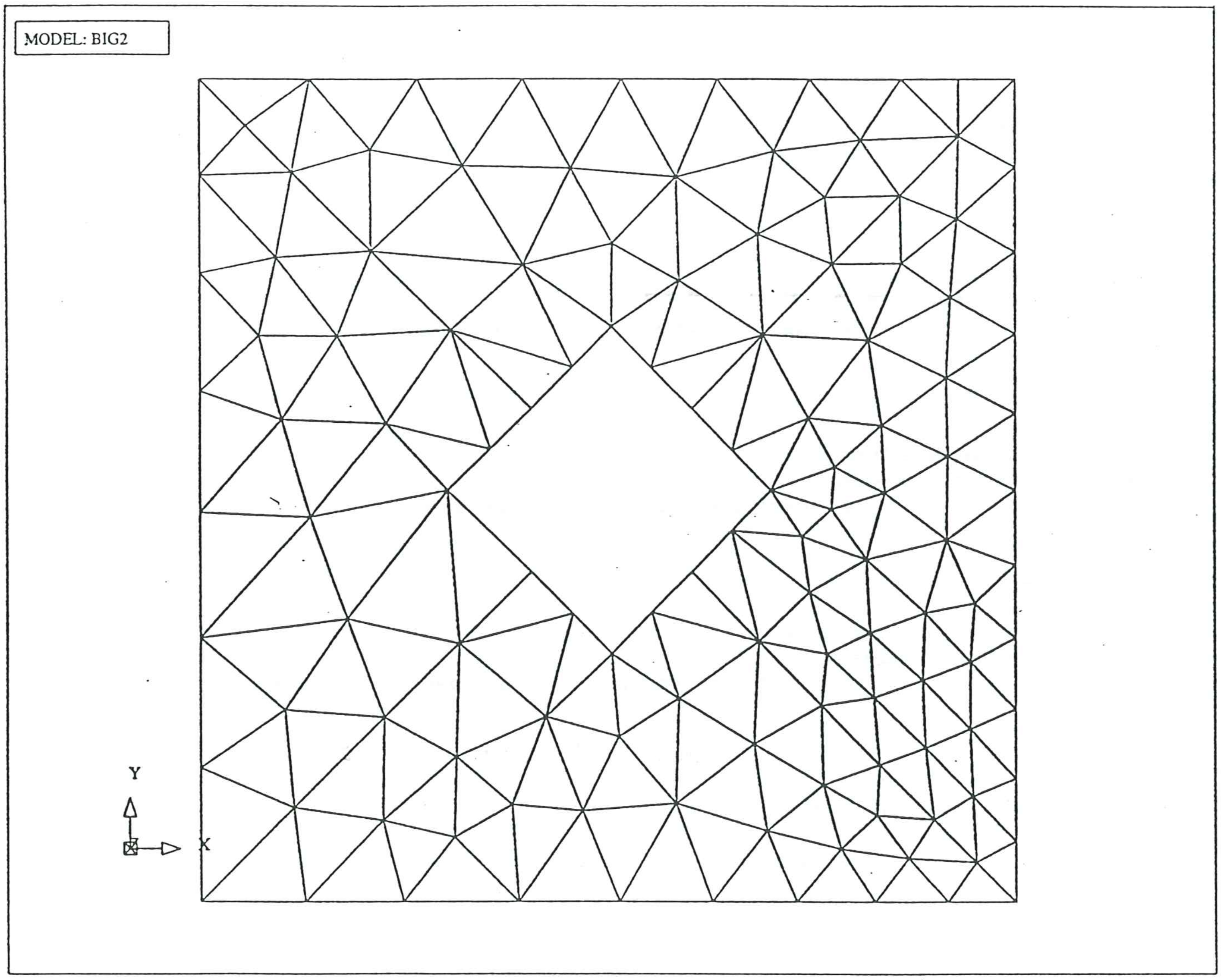
```
MODEL BIG2 (Sequential)

Number of generated elements=        182
Number of generated nodes=           115

          The time spent to generate the mesh has been        456 msec
```



Sequential program (182 elements, 115 nodes)

Figure 13. Sequential program (Model BIG2)

```
MODEL BIG2    04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=      33
Number of generated nodes=         27
                     Time spent by processor  0 has been       24 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=      43
Number of generated nodes=         34
                     Time spent by processor  1 has been       30 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=      74
Number of generated nodes=         51
                     Time spent by processor  2 has been       47 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=      51
Number of generated nodes=         38
                     Time spent by processor  3 has been       32 msec

Total number of generated elements=     201
Total number of generated nodes=        126

Maximum time spent to generate the mesh by a single processor=       32 msec
Total CPU time spent to generate the mesh=                          133 msec
```

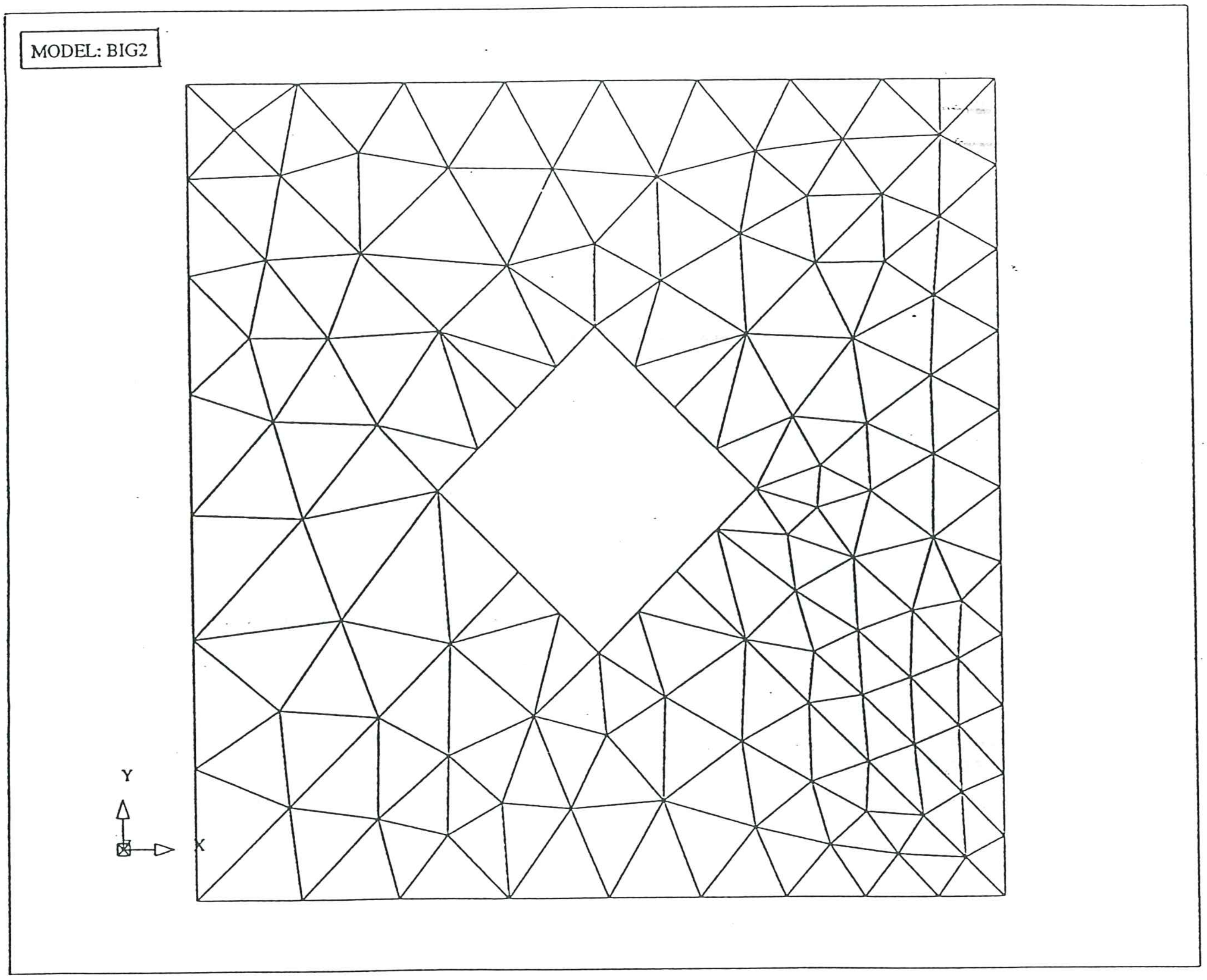


All 4 processors (201 elements, 126 nodes)

Figure 14. 4 Processors (Model BIG2)

```
MODEL BIG2   16 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  0 has been       10 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=      7
Number of generated nodes=         9
                    Time spent by processor  1 has been        8 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  2 has been       11 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  3 has been       10 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  4 has been       11 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  5 has been       10 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=     12
Number of generated nodes=        12
                    Time spent by processor  6 has been       11 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=     10
Number of generated nodes=        11
                    Time spent by processor  7 has been       10 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=      12
Number of generated nodes=         12
                        Time spent by processor  8 has been       11 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=      18
Number of generated nodes=         16
                        Time spent by processor  9 has been       13 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=      18
Number of generated nodes=         17
                        Time spent by processor 10 has been       14 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=      13
Number of generated nodes=         13
                        Time spent by processor 11 has been       11 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=      13
Number of generated nodes=         13
                        Time spent by processor 12 has been       11 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=      13
Number of generated nodes=         13
                        Time spent by processor 13 has been       11 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements=      24
Number of generated nodes=         20
                        Time spent by processor 14 has been       17 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=      22
Number of generated nodes=         19
                        Time spent by processor 15 has been       16 msec

Total number of generated elements=     222
Total number of generated nodes=        138

Maximum time spent to generate the mesh by a single processor=       17 msec
Total CPU time spent to generate the mesh=                          185 msec
```



All 16 processors (222 elements. 138 nodes)

Figure 15. 16 Processors (Model BIG2)

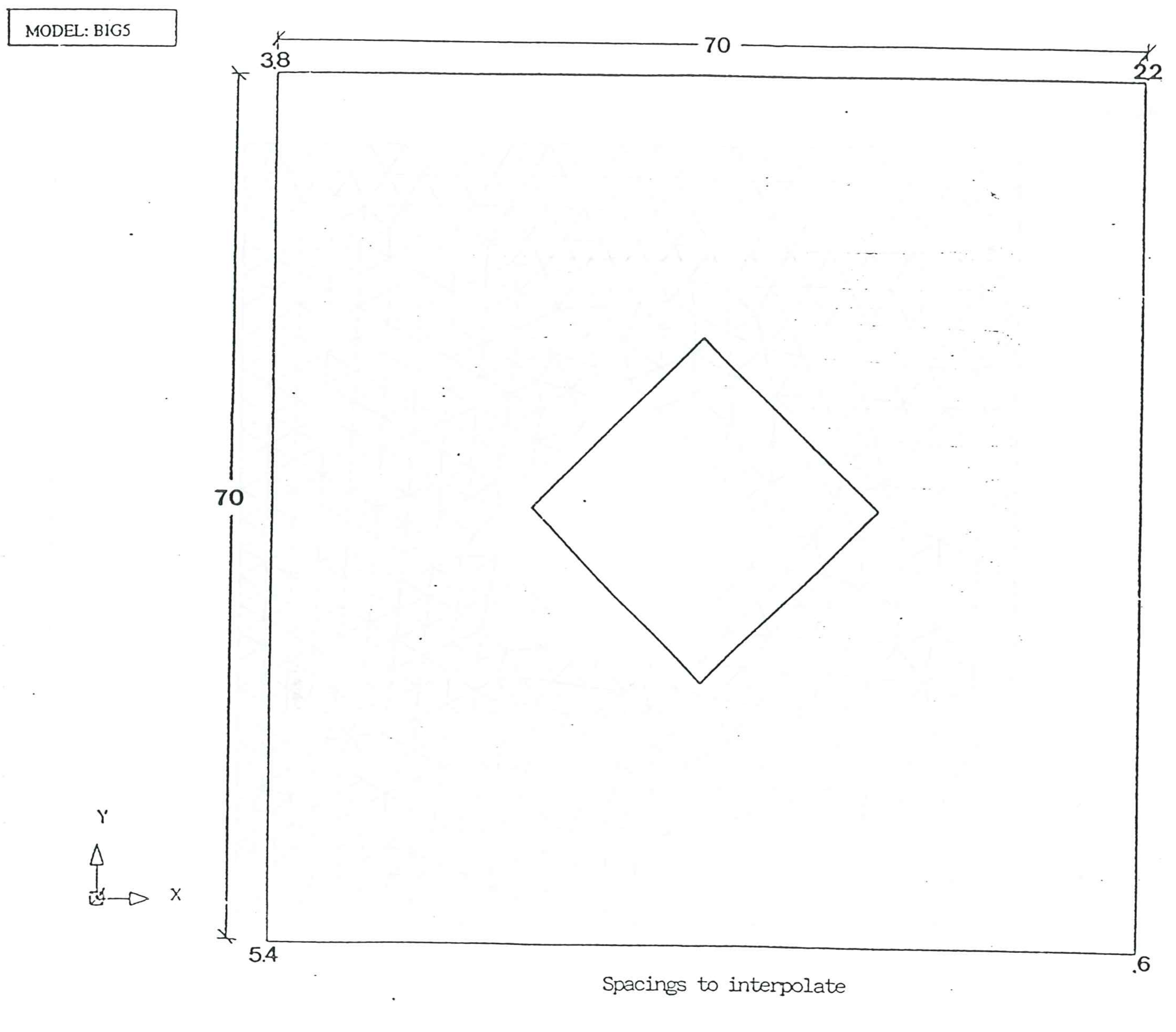


Figure 16. Spacings to interpolate (Model BIG5)

```
MODEL BIG5 (Sequential)

Number of generated elements=      1028
Number of generated nodes=          570

          The time spent to generate the mesh has been       5015 msec
```

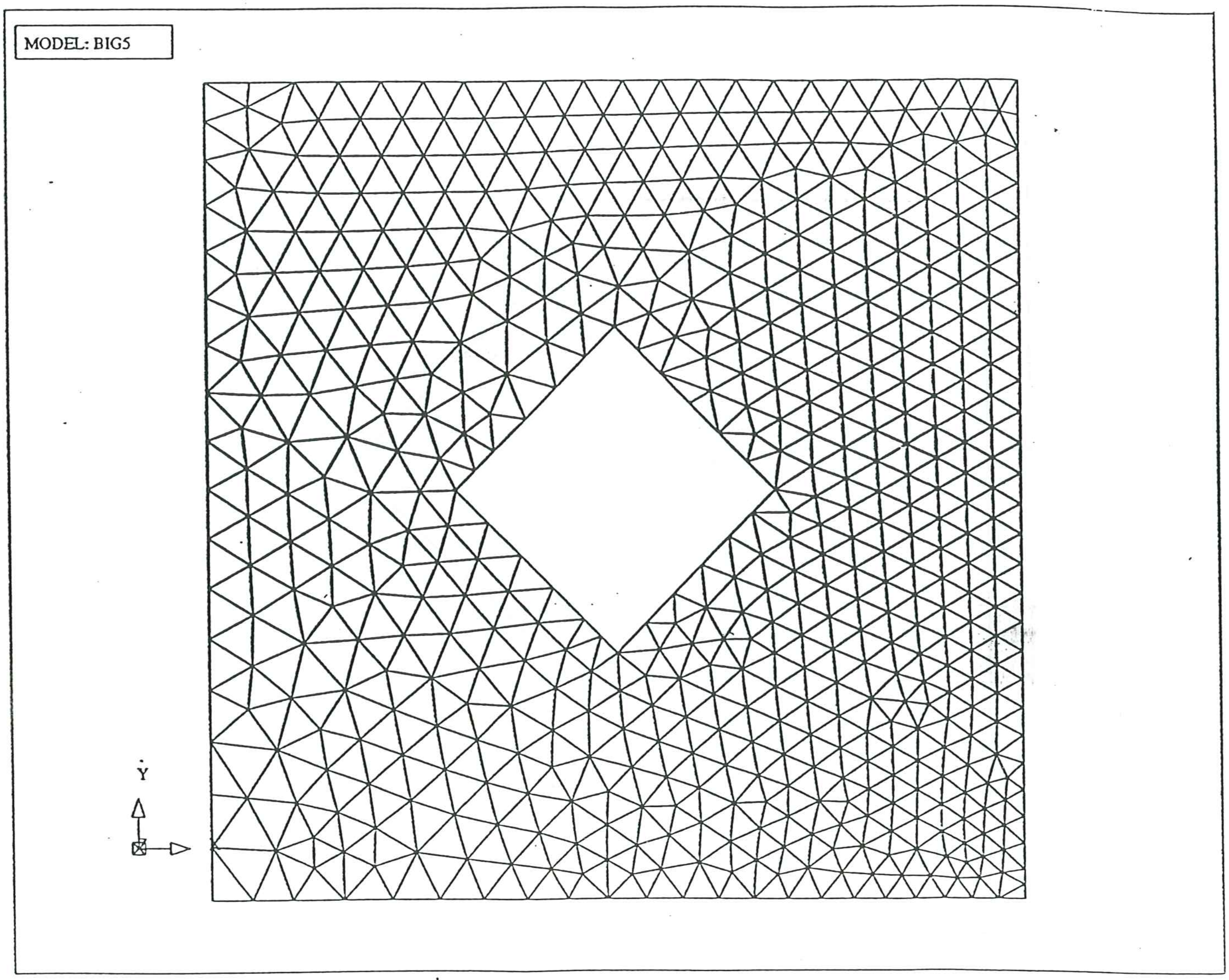
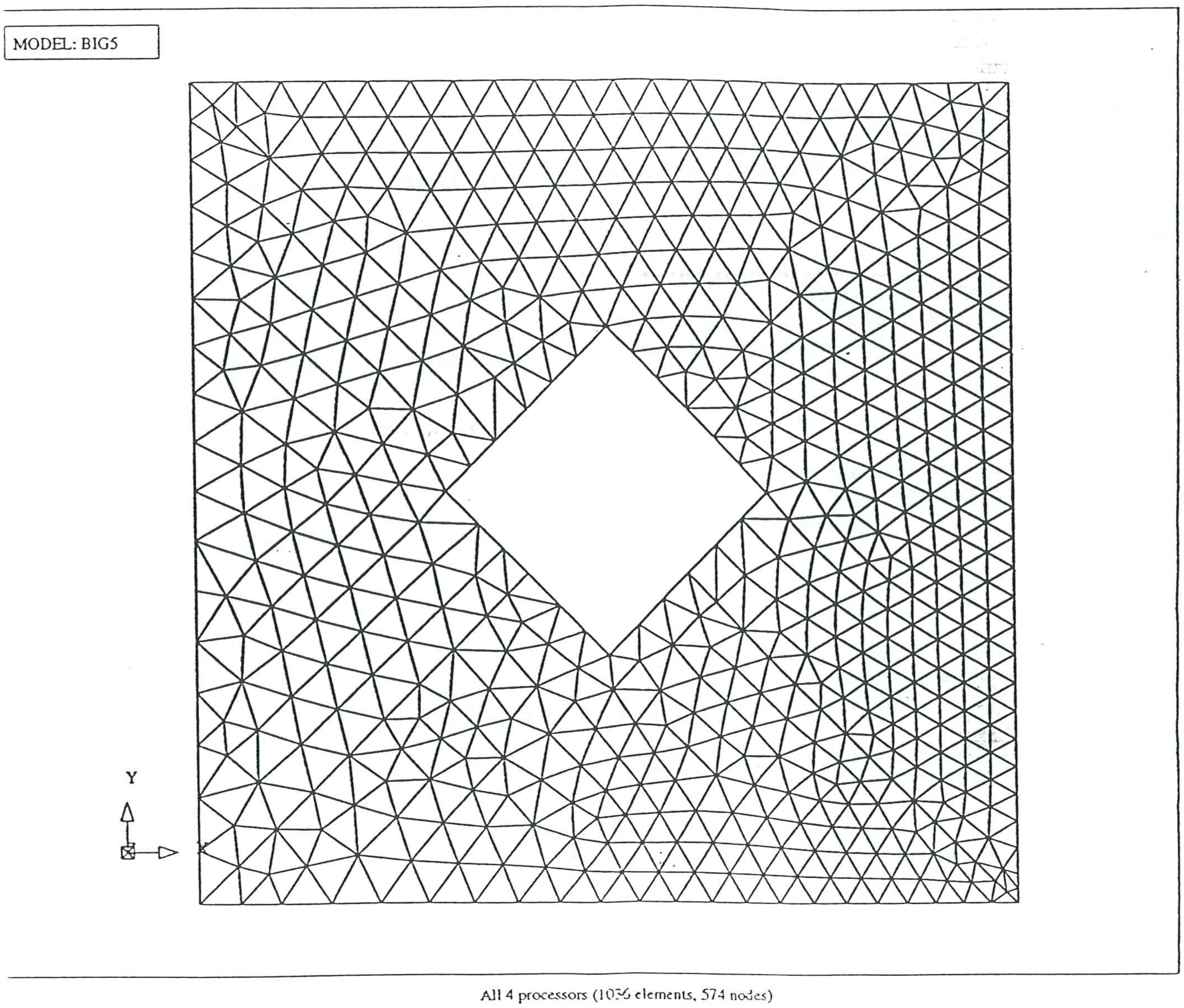


Figure 17. Sequential program (Model BIG5)

```
MODEL BIG5    04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=     180
Number of generated nodes=        113
                       Time spent by processor  0 has been      138 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=     232
Number of generated nodes=        142
                       Time spent by processor  1 has been      169 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=     355
Number of generated nodes=        209
                       Time spent by processor  2 has been      293 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=     269
Number of generated nodes=        162
                       Time spent by processor  3 has been      189 msec

Total number of generated elements=  1036
Total number of generated nodes=      574

Maximum time spent to generate the mesh by a single processor=   293 msec
Total CPU time spent to generate the mesh=                        789 msec
```

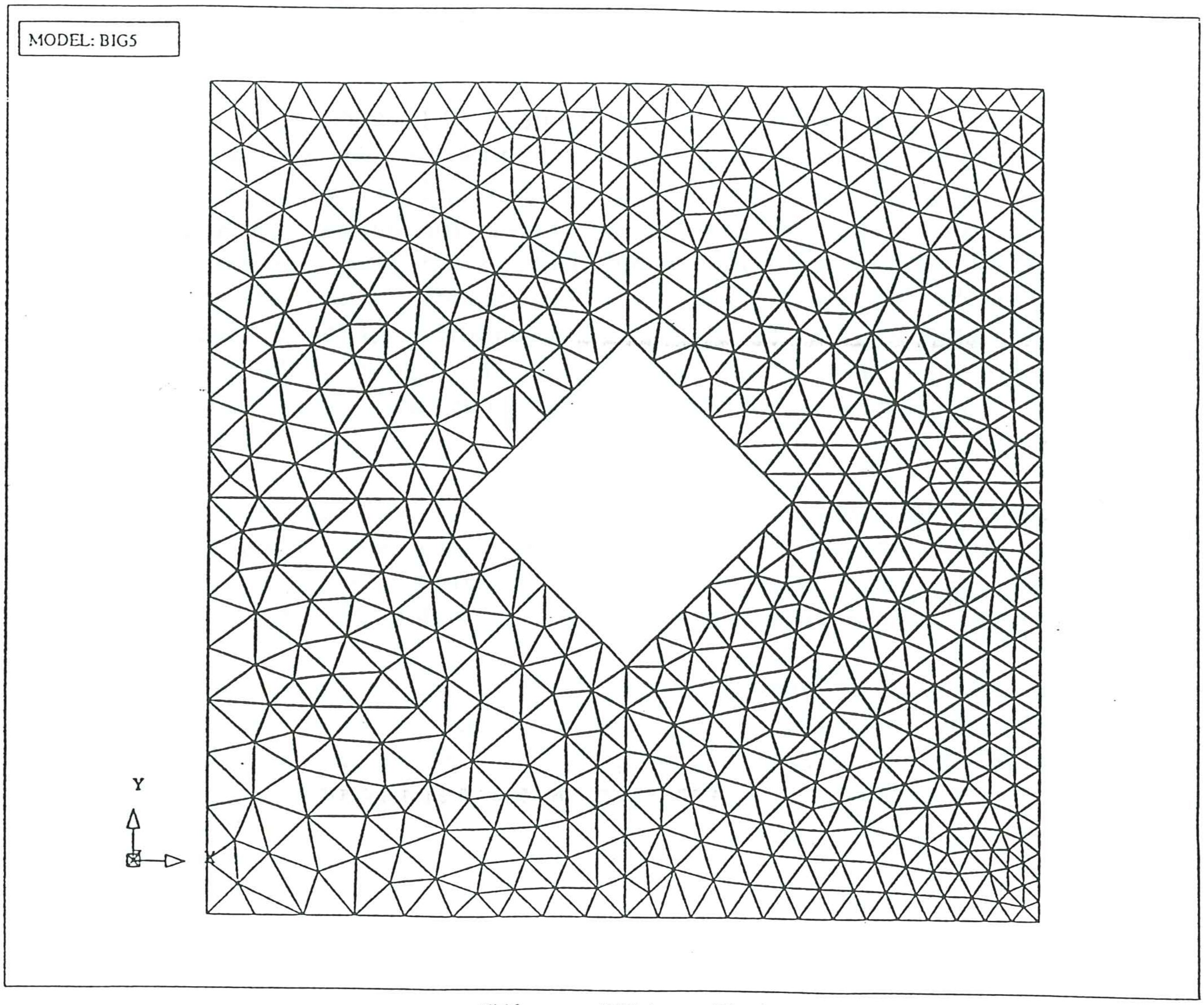


All 4 processors (1036 elements, 574 nodes)

Figure 18. 4 Processors (Model BIG5)

```
MODEL BIG5   16 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=     48
Number of generated nodes=        35
                     Time spent by processor  0 has been      29 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=     45
Number of generated nodes=        33
                     Time spent by processor  1 has been      29 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=     50
Number of generated nodes=        36
                     Time spent by processor  2 has been      33 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=     54
Number of generated nodes=        39
                     Time spent by processor  3 has been      33 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=     59
Number of generated nodes=        42
                     Time spent by processor  4 has been      37 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=     62
Number of generated nodes=        44
                     Time spent by processor  5 has been      40 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=     58
Number of generated nodes=        40
                     Time spent by processor  6 has been      34 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=     48
Number of generated nodes=        35
                     Time spent by processor  7 has been      33 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=     60
Number of generated nodes=        42
                       Time spent by processor  8 has been        39 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=     77
Number of generated nodes=        53
                       Time spent by processor  9 has been        48 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=     90
Number of generated nodes=        61
                       Time spent by processor 10 has been        60 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=     81
Number of generated nodes=        54
                       Time spent by processor 11 has been        48 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=     64
Number of generated nodes=        44
                       Time spent by processor 12 has been        39 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=     75
Number of generated nodes=        51
                       Time spent by processor 13 has been        47 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements=    122
Number of generated nodes=        79
                       Time spent by processor 14 has been        80 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=     95
Number of generated nodes=        64
                       Time spent by processor 15 has been        59 msec

Total number of generated elements=   1088
Total number of generated nodes=       600

Maximum time spent to generate the mesh by a single processor=        80 msec
Total CPU time spent to generate the mesh=                           688 msec
```



All 16 processors (1088 elements, 600 nodes)

Figure 19. 16 Processors (Model BIG5)

3.2. Models SMALL1, SMALL2 and SMALL5

The results for these models are an extension of those obtained for models BIG1, BIG2 and BIG5, but now the tendency of the form of the function that relates the CPU time spent with the number of elements or with the number of nodes created will be precisely defined as the weight of the generation of the B-splines or the nodes on the contour becomes insignificant in front of the CPU time spent to generate the whole mesh.

As well as it has been done with models BIG1, BIG2 and BIG5, in successive figures, after the numerical results for the three cases -sequential program, run with four processors and run with sixteen processors-, the resulting meshes are shown for models SMALL1 and SMALL2, but not with model SMALL5, as the huge number of elements of the meshes obtained with this model makes advisable to show the graphic of a part of it as representation of the achieved solution. For this reason, it is drawed only the figure of the mesh generated by processor number 0 with four and with sixteen processors.

In an analogous manner as it has been done with model BIG1, there are shown the meshes for every single processor when four processors are used and the assembly in groups of four when sixteen processors are used, when it comes to model SMALL1.

Furthermore, it will be redundant to get more results, numeric or graphical by continuing splitting our domain, and even more, for a next partition, a number rounding 400.000 of elements will be obtained, what exceeds in 2D the goals of the majority of projects.

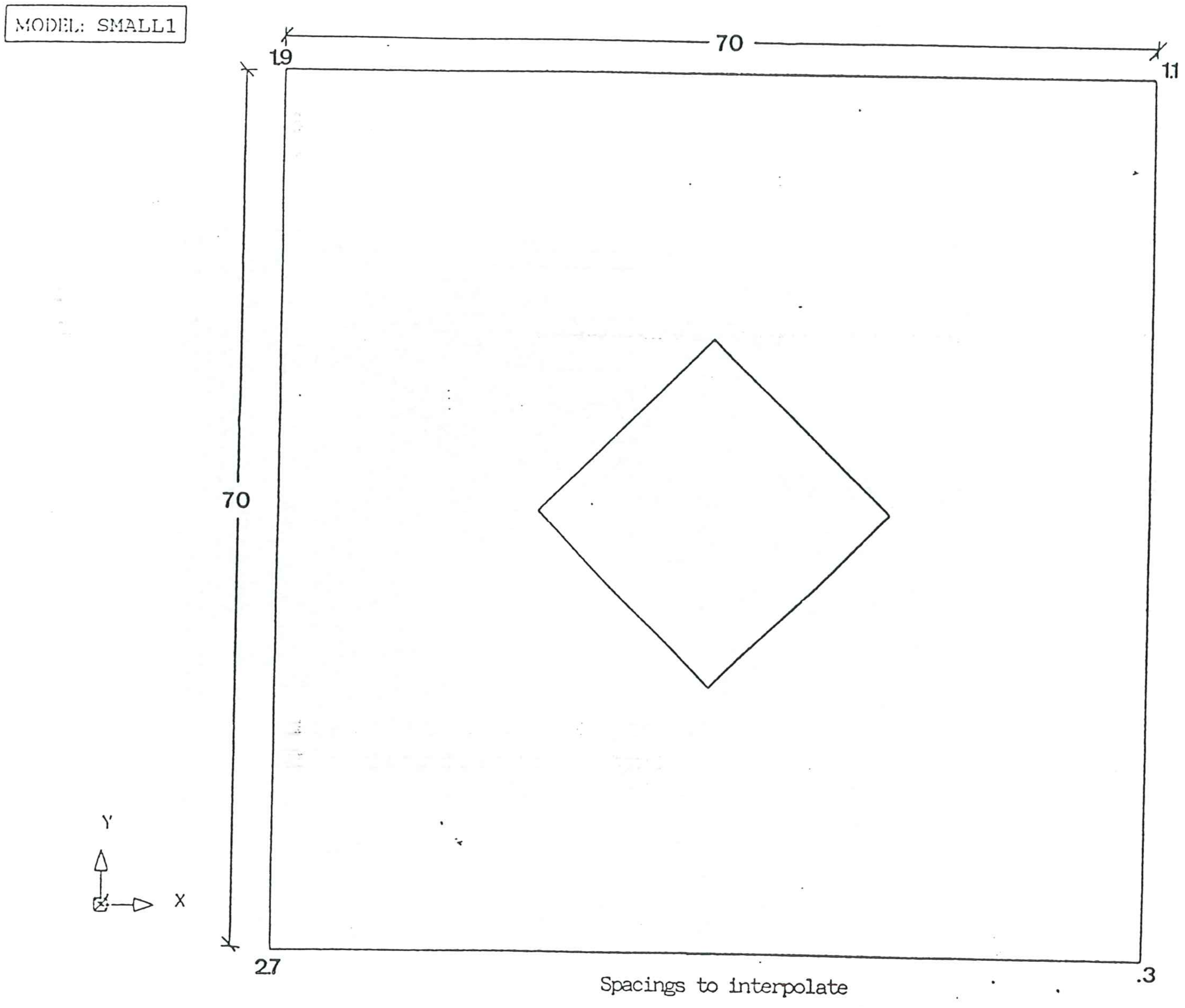
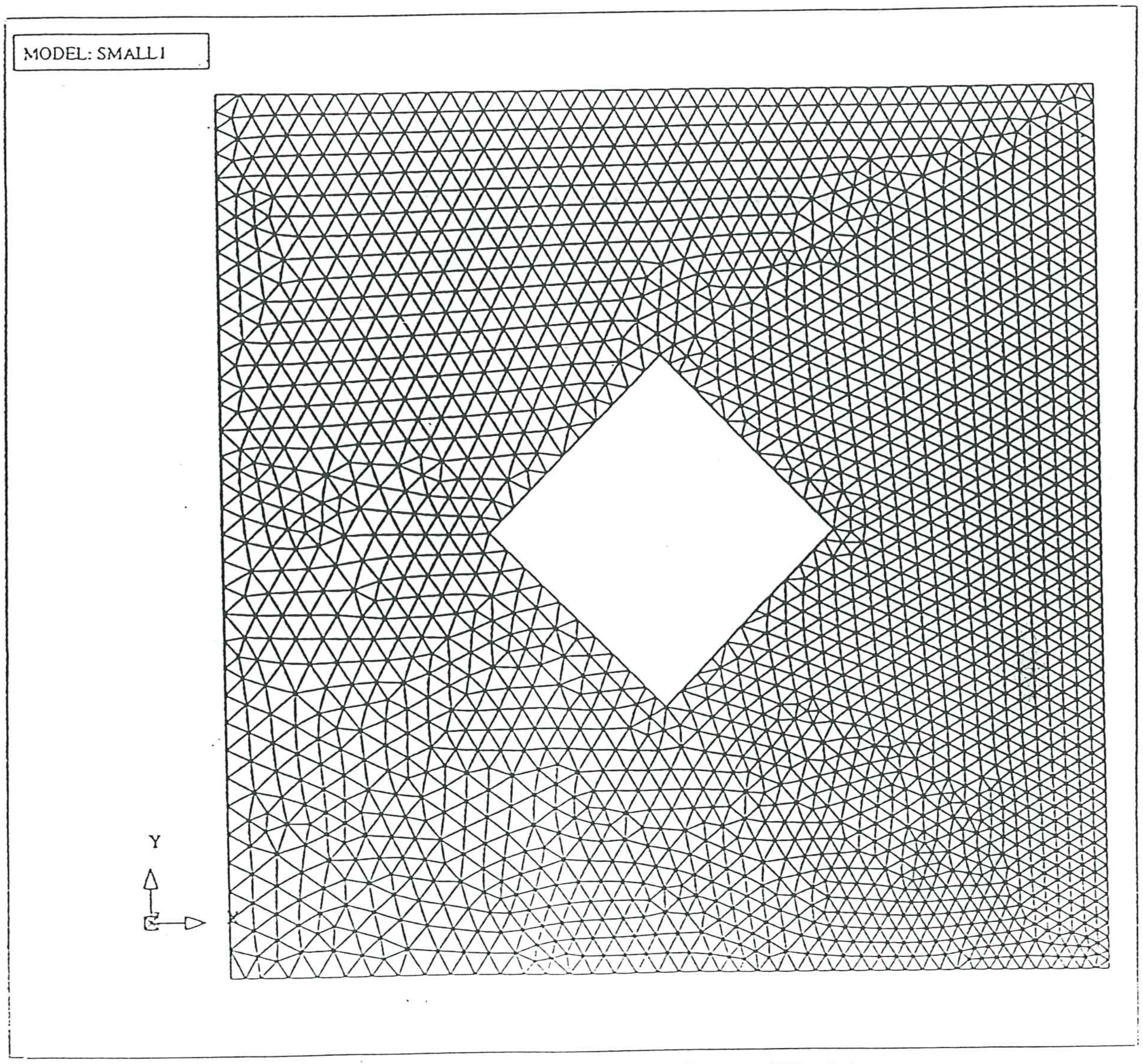


Figure 20. Spacings to interpolate (Model SMALL1)

```
MODEL SMALL1 (Sequential)

Number of generated elements=  4097
Number of generated nodes=     2159

           The. time spent to generate the mesh has been     33566 msec
```



Sequential program (4097 elements, 2159 nodes)

Figure 21. Sequential program (Model SMALL1)

```
MODEL SMALL1 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=    703
Number of generated nodes=       395
                     Time spent by processor  0 has been      775 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=    911
Number of generated nodes=       506
                     Time spent by processor  1 has been     1256 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=   1404
Number of generated nodes=       763
                     Time spent by processor  2 has been     1865 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=   1008
Number of generated nodes=       557
                     Time spent by processor  3 has been     1474 msec

Total number of generated elements=  4026
Total number of generated nodes=     2124

Maximum time spent to generate the mesh by a single processor=  1865 msec
Total CPU time spent to generate the mesh=                      5370 msec
```

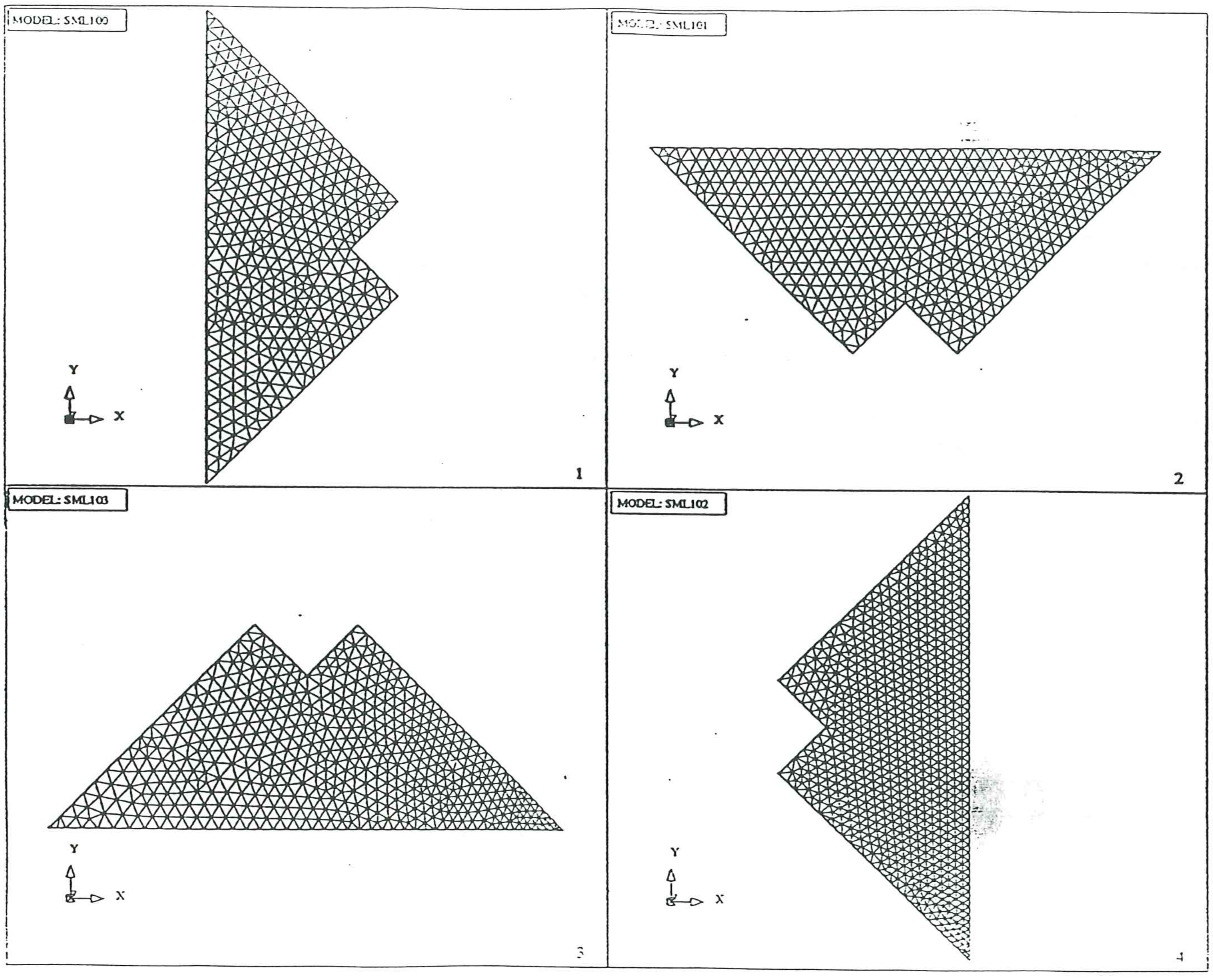
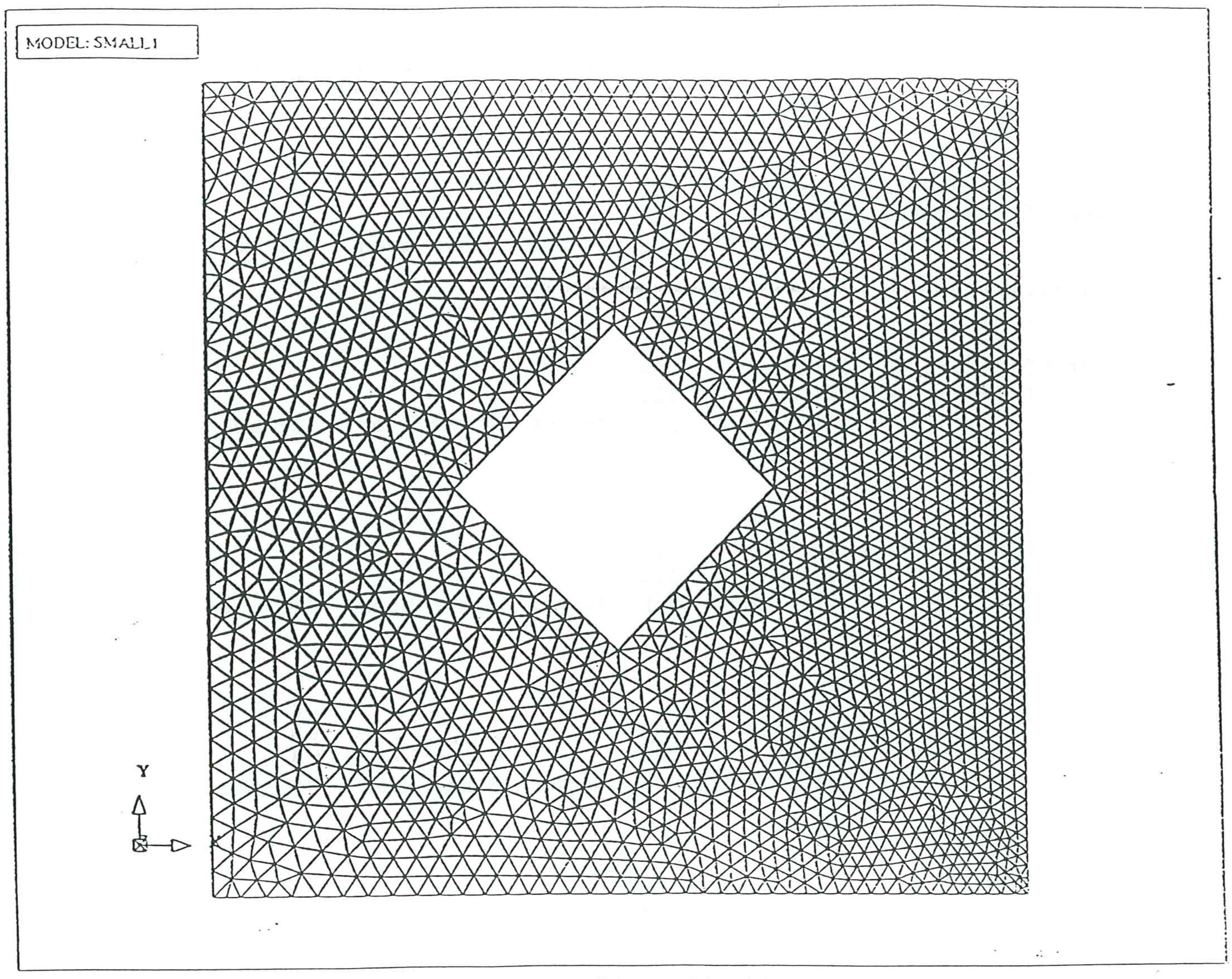


Figure 22. All Processors (Model SMALL1, 4 Processors)

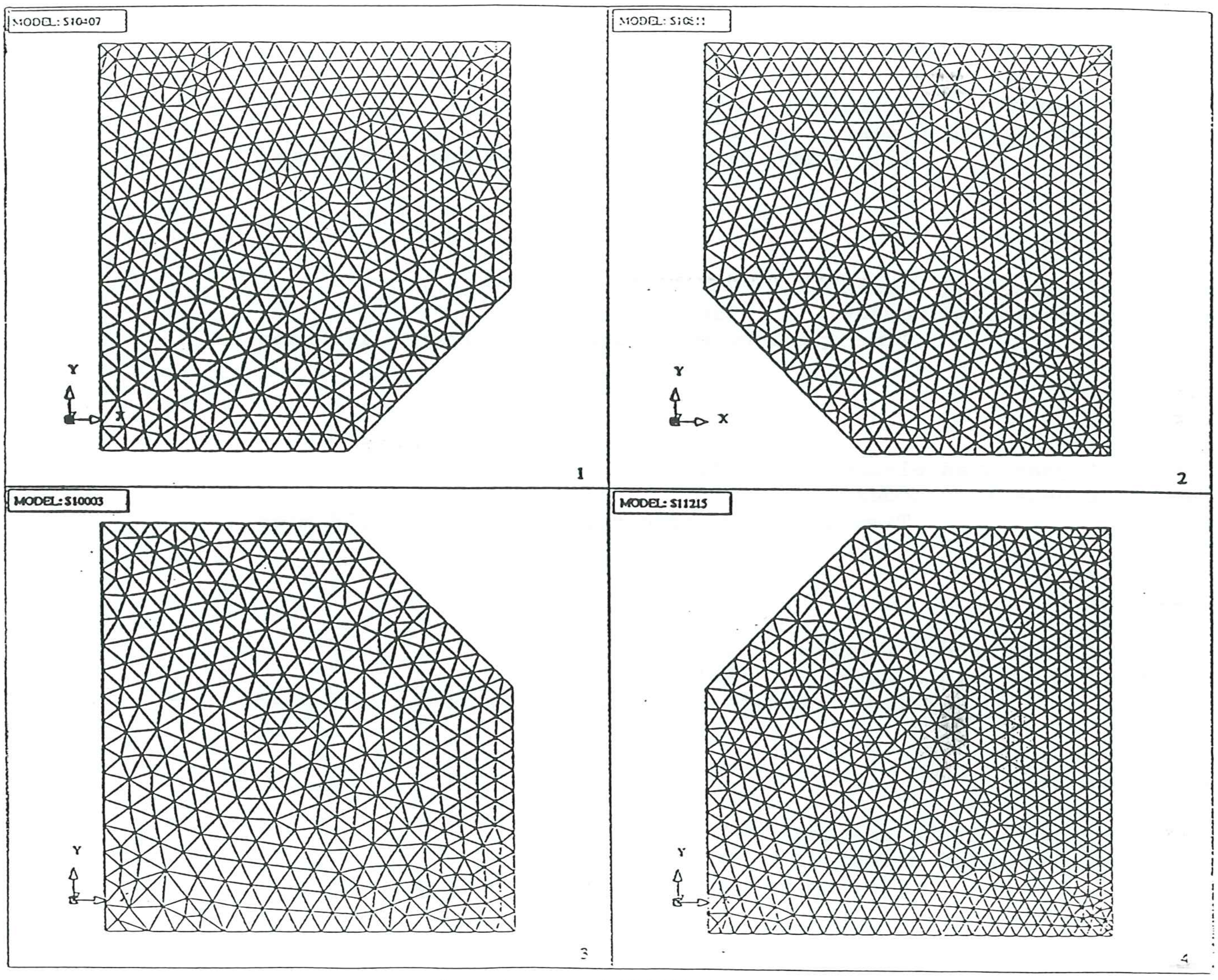


4 processors (4026 elements, 2124 nodes)

Figure 23. 4 Processors (Model SMALL1)

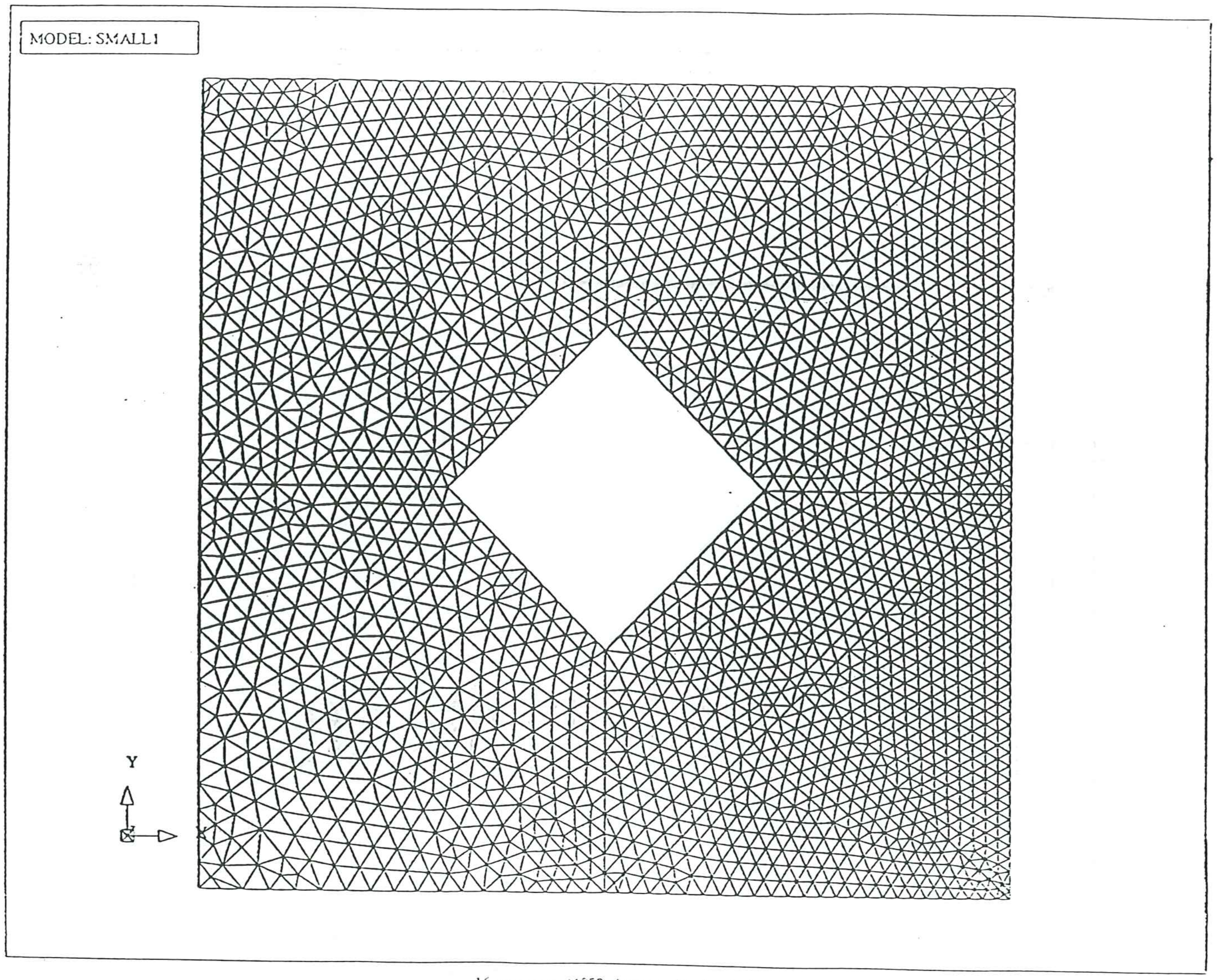
```
MODEL SMALL1 16 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=    174
Number of generated nodes=       \108
                       Time spent by processor  0 has been    131 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=    170
Number of generated nodes=       104
                       Time spent by processor  1 has been    131 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=    188
Number of generated nodes=       115
                       Time spent by processor  2 has been    145 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=    196
Number of generated nodes=       121
                       Time spent by processor  3 has been    158 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=    213
Number of generated nodes=       130
                       Time spent by processor  4 has been    159 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=    242
Number of generated nodes=       146
                       Time spent by processor  5 has been    191 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=    208
Number of generated nodes=       125
                       Time spent by processor  6 has been    158 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=    188
Number of generated nodes=       114
                       Time spent by processor  7 has been    151 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=    218
Number of generated nodes=       131
                    Time spent by processor  8 has been   192 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=    291
Number of generated nodes=       173
                    Time spent by processor  9 has been   238 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=    351
Number of generated nodes=       206
                    Time spent by processor 10 has been   302 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=    251
Number of generated nodes=       150
                    Time spent by processor 11 has been   195 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=    244
Number of generated nodes=       145
                    Time spent by processor 12 has been   214 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=    277
Number of generated nodes=       164
                    Time spent by processor 13 has been   210 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements=    470
Number of generated nodes=       270
                    Time spent by processor 14 has been   440 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=    372
Number of generated nodes=       218
                    Time spent by processor 15 has been   308 msec

Total number of generated elements=  4053
Total number of generated nodes=     2138

Maximum time spent to generate the mesh by a single processor=  440 msec
Total CPU time spent to generate the mesh=                     3323 msec
```



Groups of 4 processors (MODEL SMALL1 16 PROCESSORS)

Figure 24. Groups of 4 Processors (Model SMALL1, 16 Processors)



16 processors (4053 elements, 2135 nodes)

Figure 25. 16 Processors (Model SMALL1)

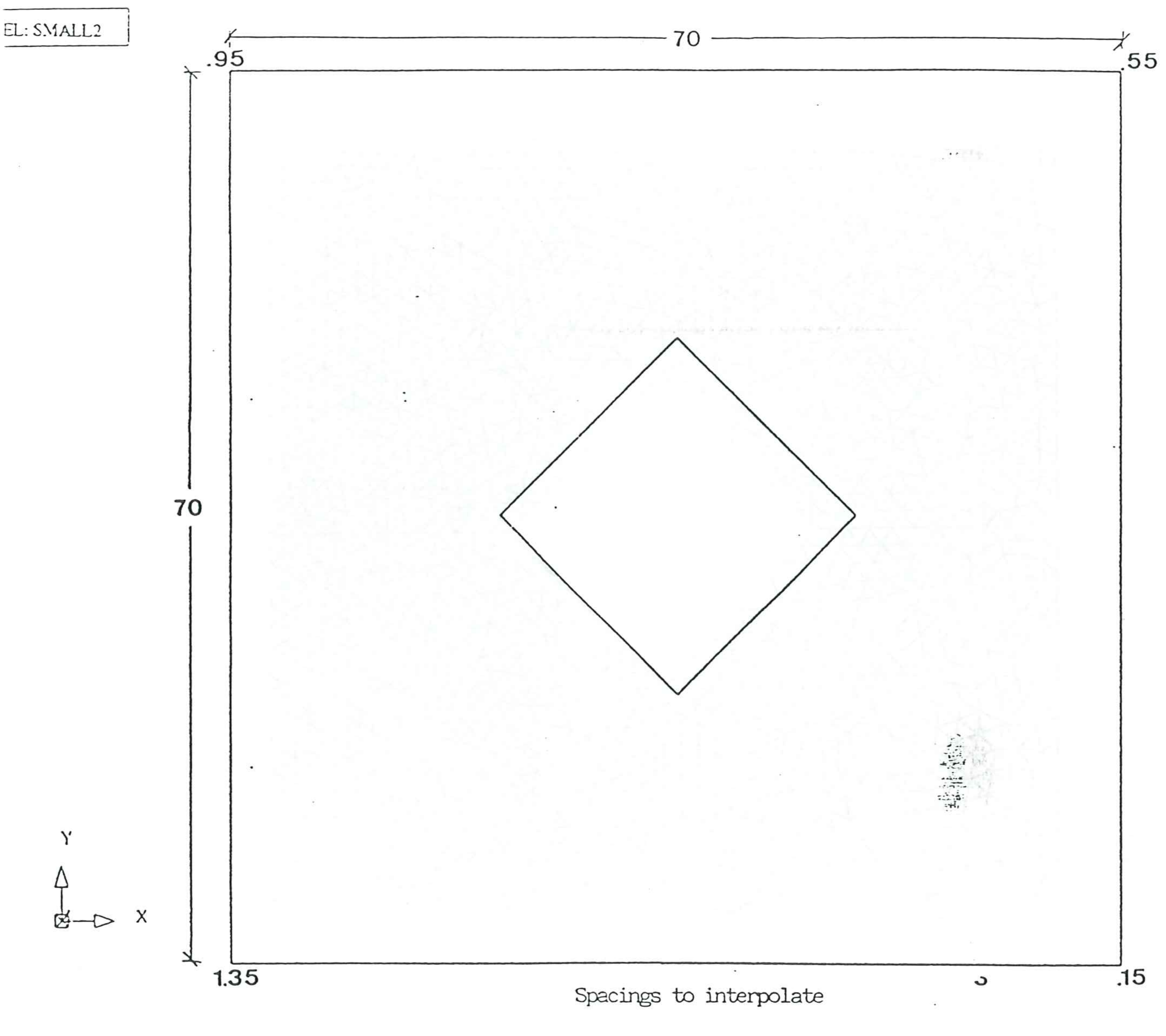
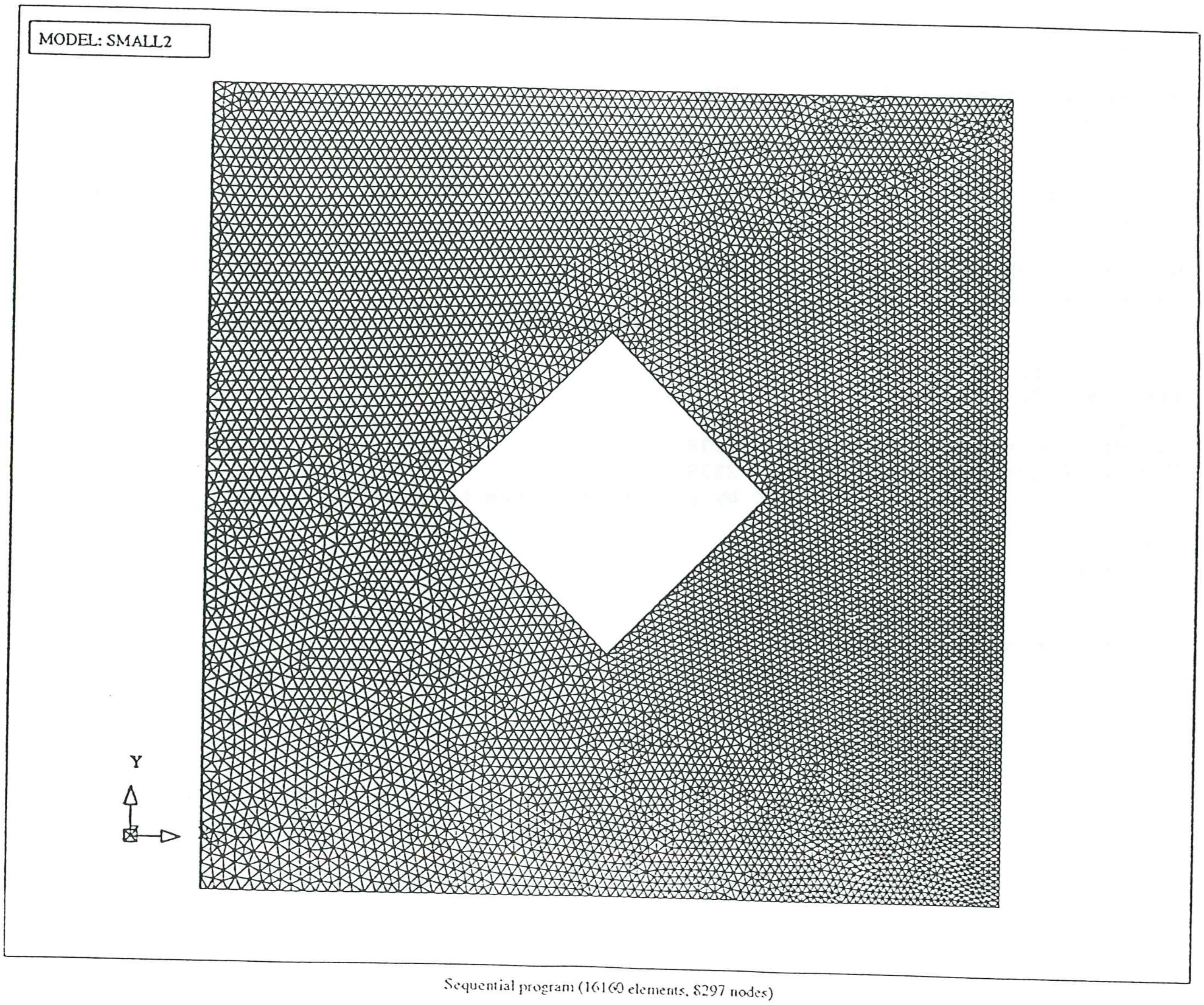


Figure 26. Spacings to interpolate (Model SMALL2)

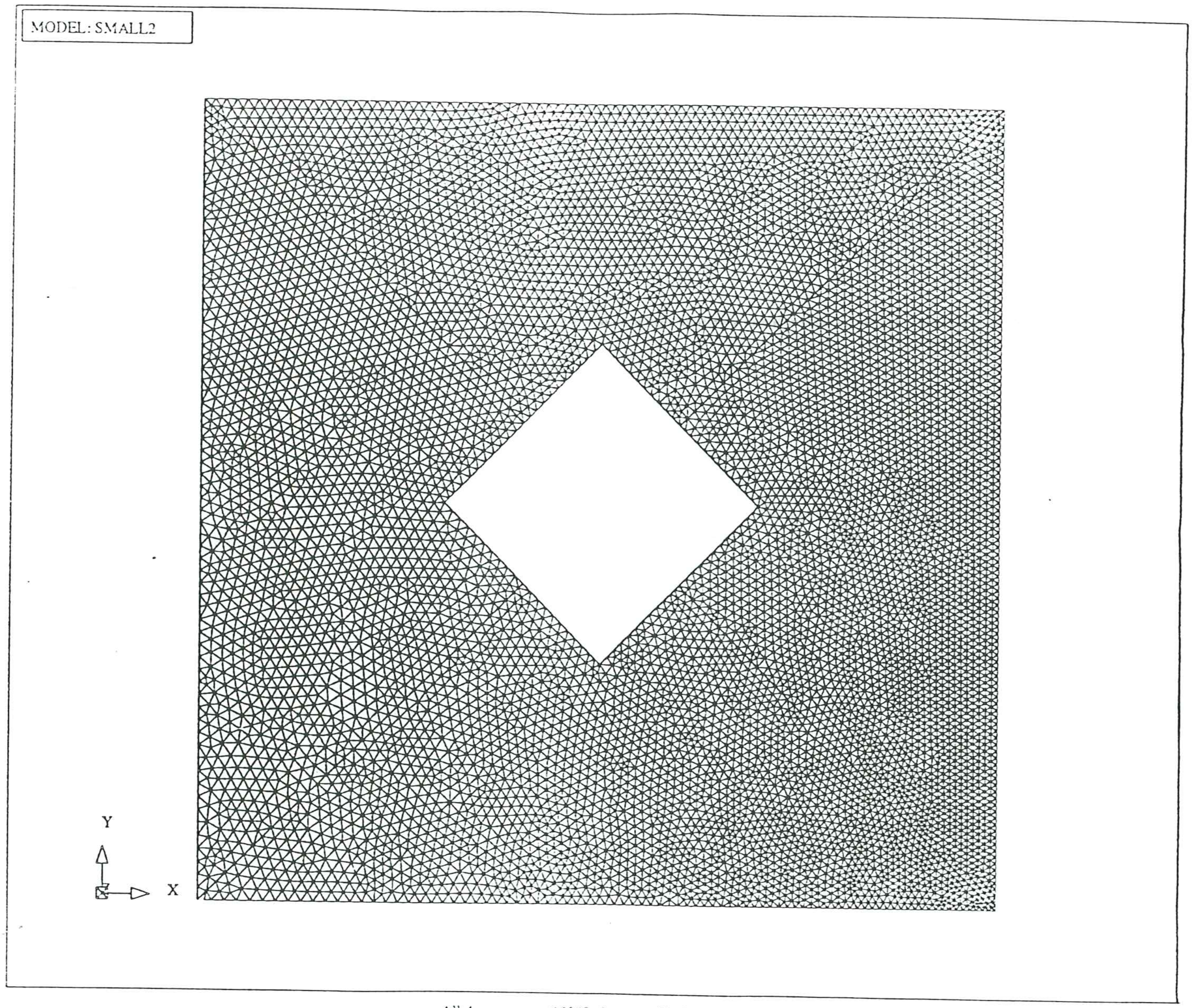
```
MODEL SMALL2 (Sequential)

Number of generated elements=     16160
Number of generated nodes=         8297

          The time spent to generate the mesh has been     236735 msec
```



Sequential program (16160 elements, 8297 nodes)

Figure 27. Sequential program (Model SMALL2)

```
MODEL SMALL2 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=   2780
Number of generated nodes=      1475
                     Time spent by processor  0 has been     4957 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=   3747
Number of generated nodes=      1972
                     Time spent by processor  1 has been     8048 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=   5438
Number of generated nodes=      2839
                     Time spent by processor  2 has been    12946 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=   4134
Number of generated nodes=      2170
                     Time spent by processor  3 has been     7448 msec

Total number of generated elements= 16099
Total number of generated nodes=     8268

Maximum time spent to generate the mesh by a single processor= 12946 msec
Total CPU time spent to generate the mesh=                      33399 msec
```



All 4 processors (16099 elements, 8268 nodes)

Figure 28. 4 Processors (Model SMALL2)

```
MODEL SMALL2 16 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=    665
Number of generated nodes=       373
                       Time spent by processor  0 has been      645 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=    612
Number of generated nodes=       342
                       Time spent by processor  1 has been      610 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=    715
Number of generated nodes=       397
                       Time spent by processor  2 has been      640 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=    772
Number of generated nodes=       430
                       Time spent by processor  3 has been      708 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=    816
Number of generated nodes=       453
                       Time spent by processor  4 has been      894 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=    924
Number of generated nodes=       510
                       Time spent by processor  5 has been     1060 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=    797
Number of generated nodes=       439
                       Time spent by processor  6 has been      821 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=    679
Number of generated nodes=       377
                       Time spent by processor  7 has been      630 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=    840
Number of generated nodes=       462
                    Time spent by processor  8 has been      860 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=   1147
Number of generated nodes=       627
                    Time spent by processor  9 has been     1280 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=   1378
Number of generated nodes=       748
                    Time spent by processor 10 has been     1651 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=   1054
Number of generated nodes=       574
                    Time spent by processor 11 has been     1073 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=    930
Number of generated nodes=       509
                    Time spent by processor 12 has been     1014 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=   1132
Number of generated nodes=       615
                    Time spent by processor 13 has been     1163 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements=   1848
Number of generated nodes=       992
                    Time spent by processor 14 has been     2448 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=   1565
Number of generated nodes=       844
                    Time spent by processor 15 has been     1849 msec

Total number of generated elements=  15874
Total number of generated nodes=      8156

Maximum time spent to generate the mesh by a single processor=   2448 msec
Total CPU time spent to generate the mesh=                      17346 msec
```

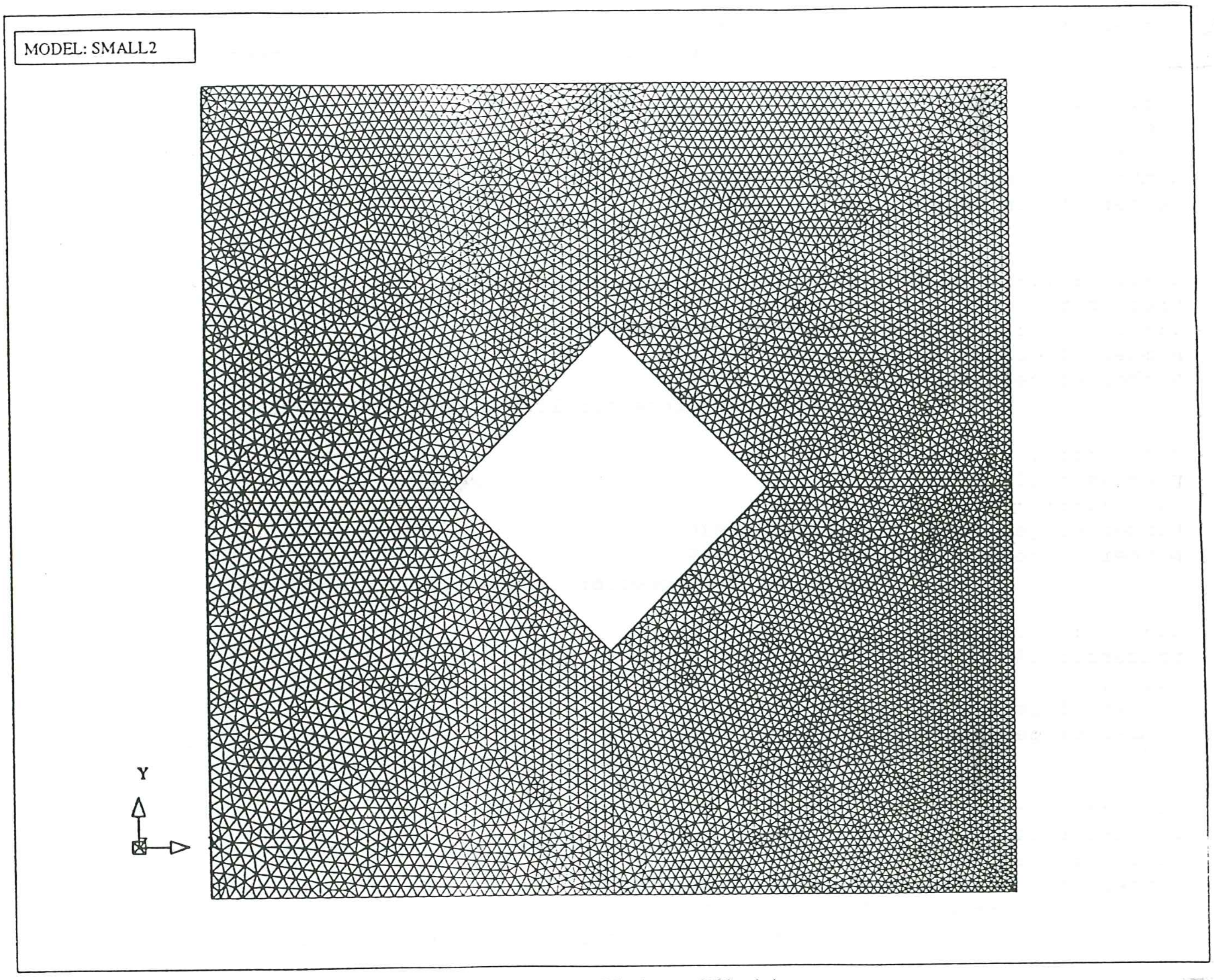


Figure 29. 16 Processors (Model SMALL2)

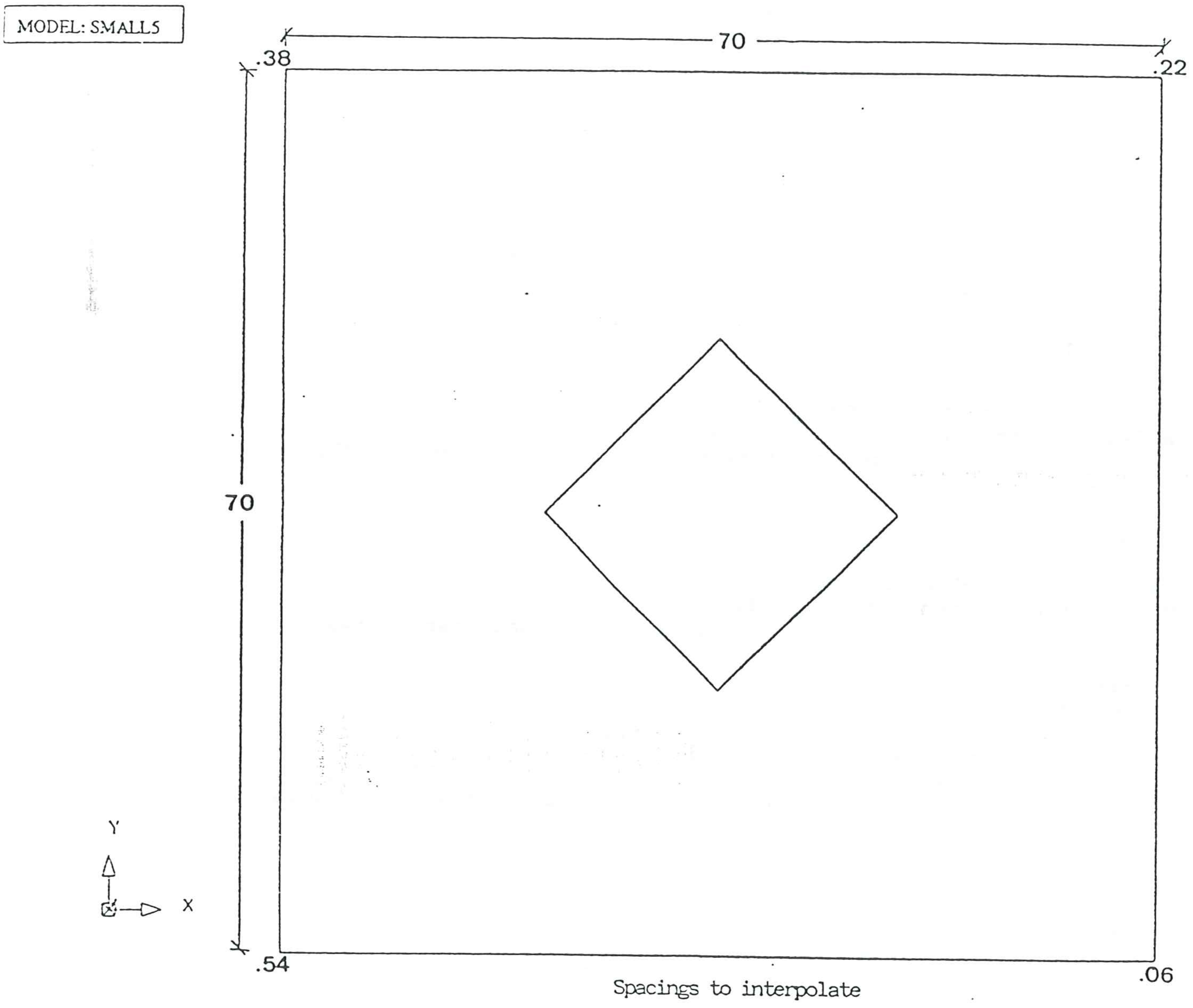
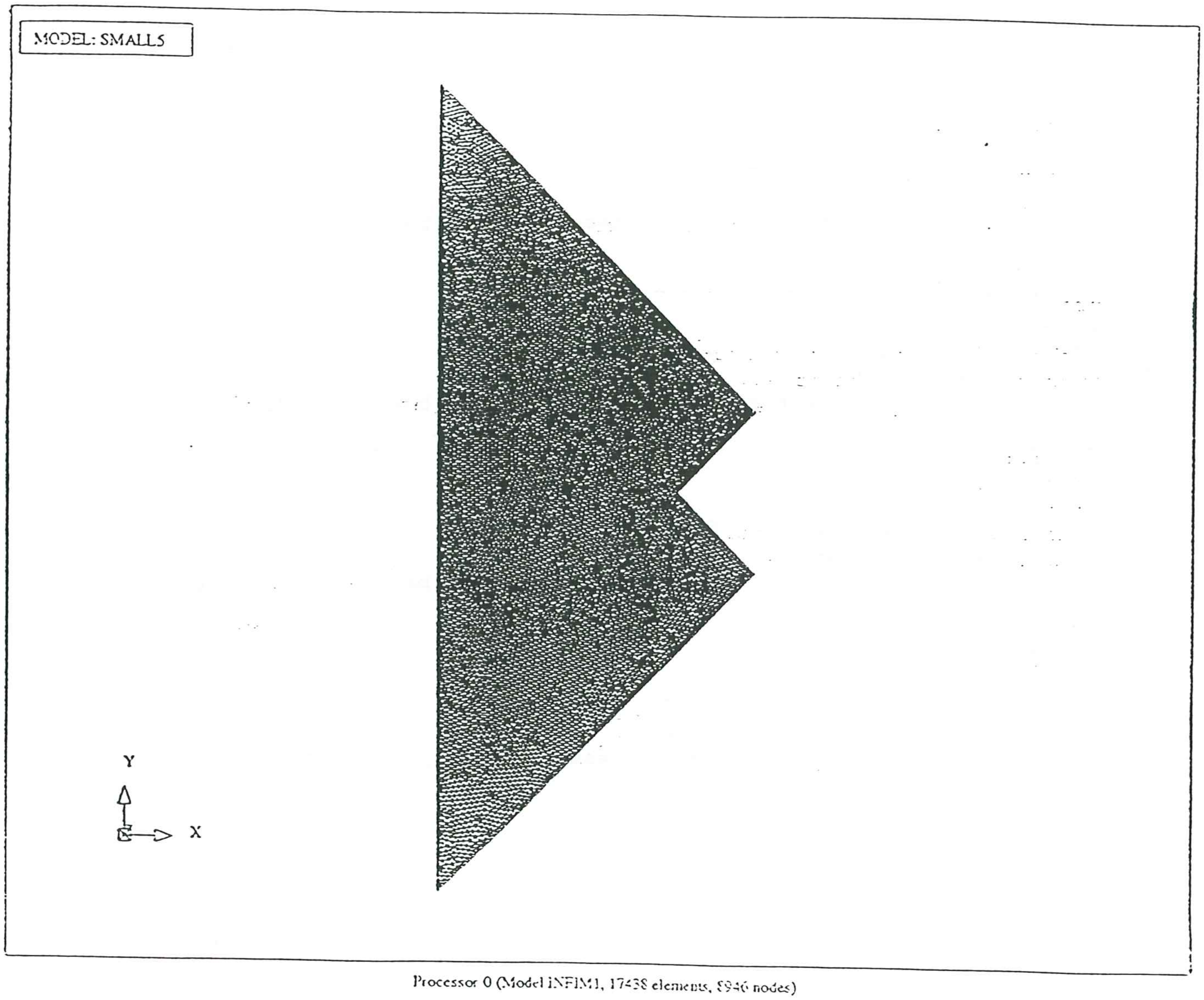


Figure 30. Spacings to interpolate (Model SMALL5)

```
MODEL SMALL5 (Sequential)

Number of generated elements=   101961
Number of generated nodes=       51521

          The time spent to generate the mesh has been   3530231 msec
```

```
MODEL SMALL5 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=  17438
Number of generated nodes=      8946
                      Time spent by processor  0 has been   75235 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=  23707
Number of generated nodes=     12119
                      Time spent by processor  1 has been  106430 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=  32473
Number of generated nodes=     16331
                      Time spent by processor  2 has been  172490 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=  25549
Number of generated nodes=     13054
                      Time spent by processor  3 has been  104609 msec


Total number of generated elements=     99657
Total number of generated nodes=        49893

Maximum time spent to generate the mesh by a single processor=  172490 msec
Total CPU time spent to generate the mesh=                      458764 msec
```

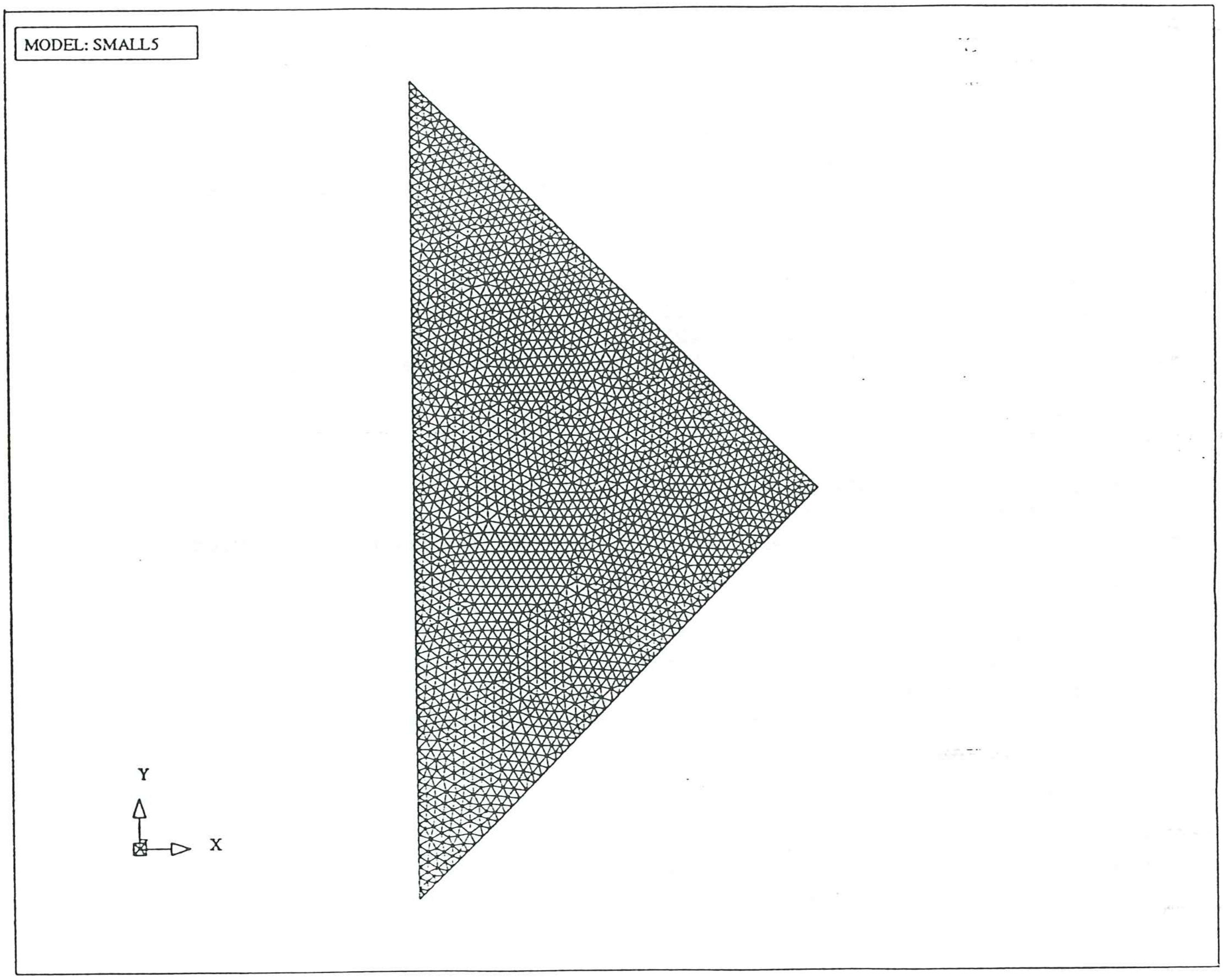


Processor 0 (Model INFIM1, 17438 elements, 8946 nodes)

Figure 31. Processor number 0. (Model SMALL5, 4 Processors)

```
MODEL SMALL5 16 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=  4262
Number of generated nodes=     2239
                     Time spent by processor  0 has been     7467 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=  3937
Number of generated nodes=     2065
                     Time spent by processor  1 has been     7933 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=  4753
Number of generated nodes=     2481
                     Time spent by processor  2 has been    11283 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=  4925
Number of generated nodes=     2578
                     Time spent by processor  3 has been     8578 msec

::::::::::::
processor 04
::::::::::::
Number of generated elements=  5197
Number of generated nodes=     2718
                     Time spent by processor  4 has been    10278 msec

::::::::::::
processor 05
::::::::::::
Number of generated elements=  6036
Number of generated nodes=     3144
                     Time spent by processor  5 has been    17946 msec

::::::::::::
processor 06
::::::::::::
Number of generated elements=  5053
Number of generated nodes=     2634
                     Time spent by processor  6 has been    11901 msec

::::::::::::
processor 07
::::::::::::
Number of generated elements=  4302
Number of generated nodes=     2252
                     Time spent by processor  7 has been     7630 msec
```

```
::::::::::::
processor 08
::::::::::::
Number of generated elements=  5471
Number of generated nodes=     2845
                     Time spent by processor  8 has been    12874 msec

::::::::::::
processor 09
::::::::::::
Number of generated elements=  7550
Number of generated nodes=     3915
                     Time spent by processor  9 has been    23873 msec

::::::::::::
processor 10
::::::::::::
Number of generated elements=  8922
Number of generated nodes=     4616
                     Time spent by processor 10 has been    21745 msec

::::::::::::
processor 11
::::::::::::
Number of generated elements=  6601
Number of generated nodes=     3421
                     Time spent by processor 11 has been    11160 msec

::::::::::::
processor 12
::::::::::::
Number of generated elements=  6018
Number of generated nodes=     3124
                     Time spent by processor 12 has been    13467 msec

::::::::::::
processor 13
::::::::::::
Number of generated elements=  7302
Number of generated nodes=     3780
                     Time spent by processor 13 has been    13717 msec

::::::::::::
processor 14
::::::::::::
Number of generated elements= 11875
Number of generated nodes=     6118
                     Time spent by processor 14 has been    35026 msec

::::::::::::
processor 15
::::::::::::
Number of generated elements=  9992
Number of generated nodes=     5161
                     Time spent by processor 15 has been    20965 msec

Total number of generated elements= 102916
Total number of generated nodes=     53253

Maximum time spent to generate the mesh by a single processor=   35026 msec
Total CPU time spent to generate the mesh=                      235843 msec
```



Processor 0 (Model SMALL5, 4262 elements, 2239 nodes)

Figure 32. Processor number 0. (Model SMALL5, 16 Processors)

4. CONCLUSIONS

Considering the tendency of the increasing time in function of the increase of the number of elements (what depends on the number of iterations required as well as the average number of tries required to find the host element for each point to be interpolated [2]) generated by PARAMESH in each processor, it is clear the enhancement represented by parallelization.

To illustrate the different comments to the numerical results, various figures and tables are added.

The number of elements and the number of nodes for models BIG1, BIG2 and BIG5 created under all procedures are shown in table 1.

	BIG1		BIG2		BIG5	
	Elements	Nodes	Elements	Nodes	Elements	Nodes
Seqtial.	48	38	182	115	1028	570
4 Procs.	67	50	201	126	1036	574
16 Procs.	77	53	222	138	1088	600

Table 1. Number of elements and number of nodes for models BIG1, BIG2 and BIG5.

The number of elements and the number of nodes for models SMALL1, SMALL2 and SMALL5 created under all procedures are shown in table 2.

	SMALL1		SMALL2		SMALL5	
	Elements	Nodes	Elements	Nodes	Elements	Nodes
Seqtial.	4097	2159	16160	8297	101961	51521
4 Procs.	4026	2124	16099	8268	99657	49893
16 Procs.	4053	2138	15874	8156	102916	53253

Table 2. Number of elements and number of nodes for models SMALL1, SMALL2 and SMALL5.

The most important parameters of the study to relate with these figures are the required times to generate the meshes. For this reason, the next graphics will show the variation of the CPU and elapsed times as function of the variation of the number of nodes (or elements, it could be, as the relationship between these two parameters seems to be a nearly linear one).

The different obtained times are shown in table 3, for models BIG1, BIG2 and BIG5 and in table 4 for models SMALL1, SMALL2 and SMALL5.

The first value -the CPU time- expresses the addition of the CPU time spent by all the processors in the respective procedure, it is, the whole amount of CPU spent.

The second value -the elapsed time- expresses the waiting time since the process has begun until it finishes; as the different processors run synchronously, it will represent for the parallel procedures the time spent by the processor that takes longer.

All times are expressed in milliseconds.

	BIG1		BIG2		BIG5	
	CPU	Elapsed	CPU	Elapsed	CPU	Elapsed
Seqtial.	82		456		5015	
4 Procs.	71	24	133	47	789	293
16 Procs.	81	6	185	17	688	80

Table 3. CPU and elapsed time -msec- for models BIG1, BIG2 and BIG5.

	SMALL1		SMALL2		SMALL5	
	CPU	Elapsed	CPU	Elapsed	CPU	Elapsed
Seqtial.	33566		236735		3530231	
4 Procs.	5370	1865	33399	12946	458764	172490
16 Procs.	3323	440	17346	2448	235843	35026

Table 4. CPU and elapsed time -msec- for models SMALL1, SMALL2 and SMALL5.

To get the searched relationship between the number of nodes and the spent time, let's be assumed a relation of the type $y = kx^{\alpha}$, being y the CPU time and x the number of nodes.

We will consider $x = 10^{\eta}$. Therefore, $\eta = \mathrm{Log}_{10}x$.

Operating on both sides of the expression, we will get

$$\mathrm{Log}_{10}y = \mathrm{Log}_{10}k + \alpha\eta$$

and making $K = \mathrm{Log}_{10}k$,

$$\mathrm{Log}_{10}y = K + \alpha\eta,$$

and considering $y = 10^{\xi}$, or $\xi = \mathrm{Log}_{10}y$,

we arrive to the relation

$$\xi = K + \alpha\eta$$

Under this form it is possible to represent the desired relationship in a double logarithmic scale as a straight line.

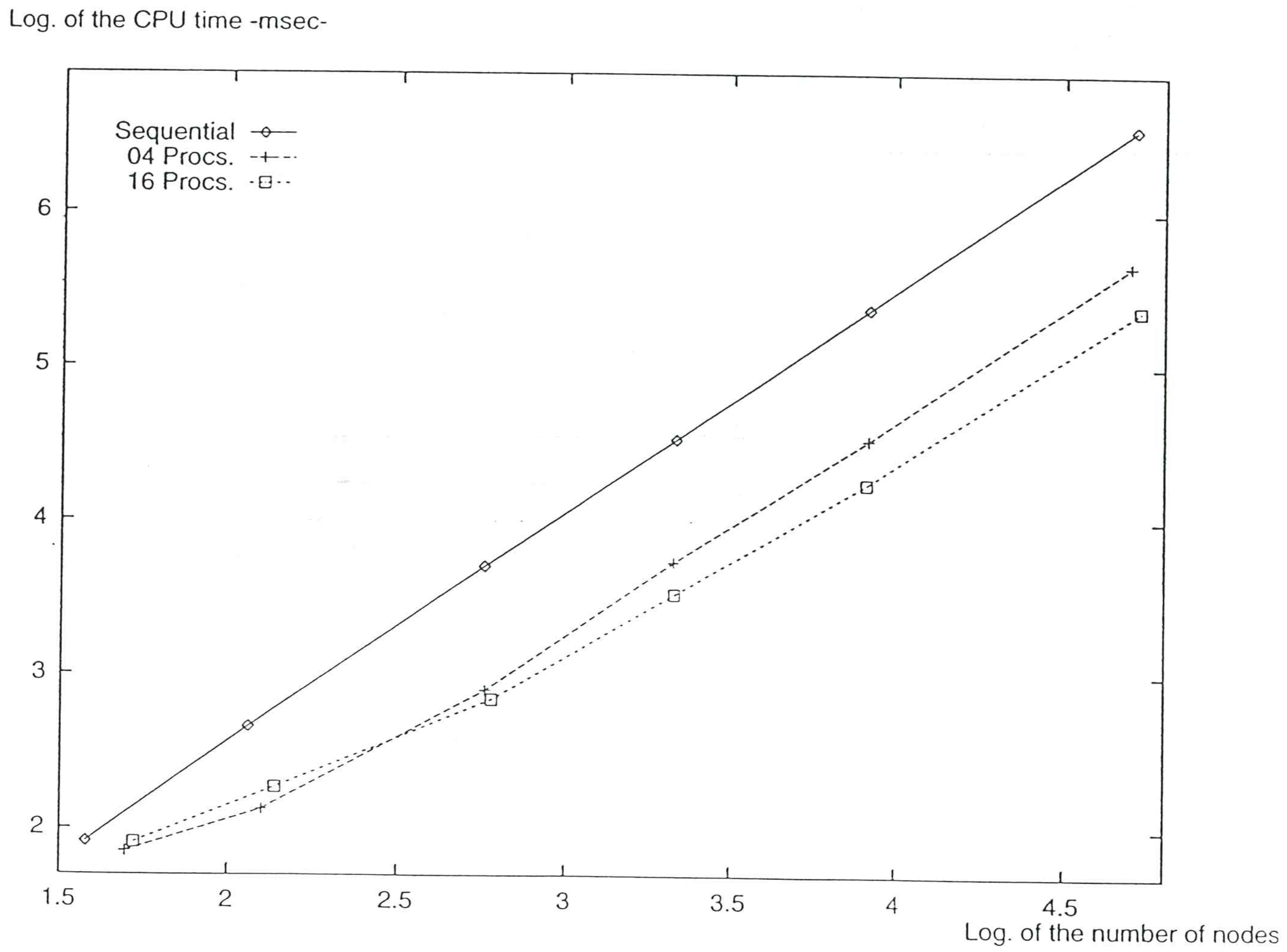


Figure 33. Relationship between the logarithm of the CPU time -msec- and the logarithm of the number of nodes: $\xi = K + \alpha\eta$

With these results, it is possible to estimate the exponent α. From the values obtained in the sequential procedures, it acquires an approximate value of 1.480 (the quotient

$$\frac{\text{Log } 3530231\text{-Log } 82}{\text{Log } 51521 \text{ - Log } 38} = \frac{6.5477\text{-}1.9138}{4.712 \text{ - } 1.58}).$$

As it can be seen in the figure 33, this value is very well represented along all the curve, as there are not problems of balancing among processors. For the parallel procedures, the slope of the straight line becomes well represented for numbers of nodes sufficiently big, because the influence of linear factors (generation of the contours and the points of them) is more negligible in front of the rest when the number of interior nodes is bigger in front of the contour ones.

With these considerations in mind, the exponent for the different parallel procedures has been calculated. For the procedure with four processors, we get an approximate value of 1.426 (calculating the slope from the model SMALL5 -458764 milliseconds and 49893 nodes- to the model BIG5 -789 milliseconds and 574 nodes) and for the procedure with sixteen processors, we get an approximate value of 1.326 (calculating the slope from the model SMALL5 -235843 milliseconds and 53253 nodes- to the model SMALL1 -3323 milliseconds and 2138 nodes).

When it comes to the elapsed times instead of the total CPU times, the considerations about the importance of a good balance become decisive to get the smallest possible time, and as there is no any addition of individual results from other processors, it would be even more desirable to get straight lines in the double logarithmic scale graphic.

Taking as reference the same models considered in the estimation of the slope for the CPU times, we obtain now a slope for the four processors procedure of 1.428 -approximately the same as the obtained for the ensemble of the processors, what implies a very good sharing.

The slope for the sixteen processors procedure of 1.361, (obviously, the value for the sequential procedure will be the same as the CPU times and elapsed times are the same) that has a bigger difference respect the slope obtained for the ensemble of the processors, what could be explained by a greater gap between the fastest and slowest processors.

It is easily observed, indeed, comparing the number of elements generated by the two extremes in both cases.

In model SMALL5, for instance, the number of elements generated by processor number 2 is 32473, less than twice the number of elements generated by processor number 0, which is 17438.

On the other hand, for the same reference model, the number of elements generated by processor number 14 is 11875, what represents more than three times the number of elements generated by processor number 1, which is 3937.

Figure 34 shows on the Y-axis the logarithm of the elapsed time spent in milliseconds for the three different procedures: sequential, with four processors and with sixteen processors.

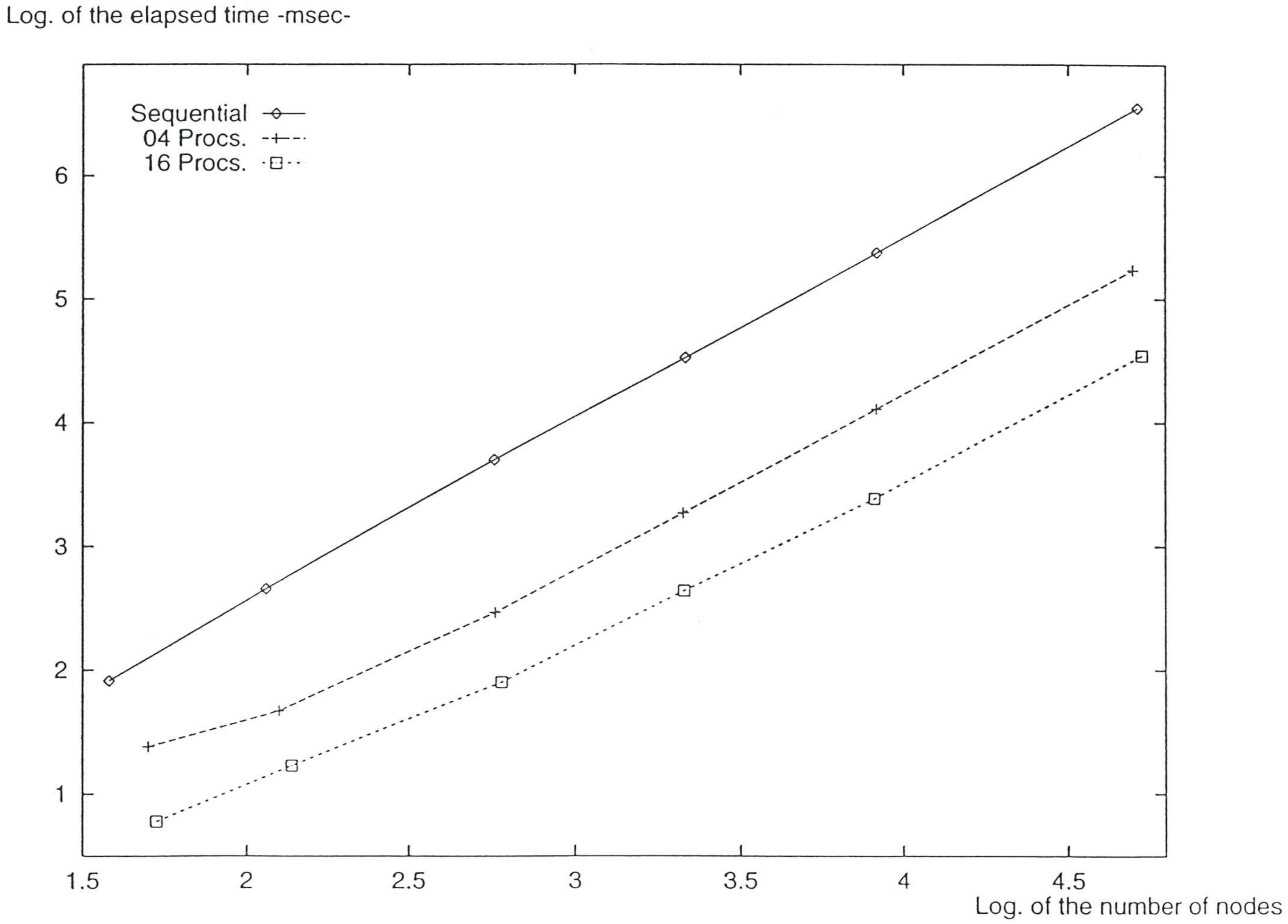


Figure 34. Relationship between the logarithm of the elapsed time -msec- and the logarithm of the number of nodes: $\xi = K + \alpha\eta$

As a synthesis of these results, to present the enhancements of parallelization, two final tables with their corresponding graphics are included.

Table 5 gives the quotients of the CPU times between the sequential and four processors procedures, between the four and sixteen processors procedures and between the sequential and sixteen processors procedures concerning to all models.

To have a comparative value for the number of elements or nodes in every model, as these numbers did not coincide exactly, it has been chosen a common value for each model. Hence, for model BIG5, the arithmetic mean is 1050.67, obtained from $\frac{1028+1036+1088}{3}$,consequently, the number 1000 is taken as the common number of nodes for the different procedures (moreover, the most precise number of generated elements for the corresponding spacings is the obtained with the sequential procedure, as there are not imposed nodes inside the whole domain, what would justify a bigger approach to that number).

From this value, all the rest can be calculated easily. When the spacing in all the nodes increases twice, the number of elements increases four times (the square of 2). Analogously, a decrease of 2.5 times the spacing in all the nodes will represent a decrease of 6.25 times (the square of 2.5).

From this reference, the different number of elements considered will be: 40, 160, 1000, 4000, 16000 and 100000 for models BIG1, BIG2, BIG5, SMALL1, SMALL2 and SMALL, respectively.

	BIG1	BIG2	BIG5	SMALL1	SMALL2	SMALL5
Seqtial./4 Procs.	1.155	3.429	6.356	6.251	7.088	7.695
4 Procs./16 Procs.	0.877	0.719	1.146	1.616	1.925	1.945
Seqtial./16 Procs.	1.012	2.465	7.289	10.101	13.648	14.969

Table 5. Ratios between the CPU times for the different procedures in all models.

It is interesting to see the relationship between the obtained values for the different procedures, as it can be an indicator of the goodness of the achieved balance among processors.

In fact, for perfectly balanced work-share among processors, the improvement for four processors in front of the sequential procedure would be the same as the improvement for sixteen processors in front of the four processors procedure.

In this case, the time spent by sequential procedure would be

$$y = kx^{\propto}$$

For the four processors procedure, the time spent would be

$$y = \sum_{i=1}^{4} k(x_i)^{\alpha} = k\frac{x^{\propto}}{4^{\propto - 1}}$$

For the sixteen processors procedure, the time spent would be

$$y = \sum_{i=1}^{16} k(x_i)^{\alpha} = k\frac{x^{\propto}}{16^{\propto - 1}}$$

And the quotient between the two first exposed times and between the two last would be exactly the same: $\frac{1}{4^{\propto - 1}}$.

In our examples, the ratio between the two parallel procedures goes from 1.616 to 1.945 for the three SMALL models, and the ratio between the sequential and with four processors procedures is not too far, as it goes from 6.251 to 7.695, what seems a rather acceptable balance and will coincide with previous observations.

It can be appreciated a regular relationship between the two parallel procedures. To consider the previously exposed relationship $\frac{1}{4^{\propto - 1}}$, for an average value $\propto$ of 1.376, a ratio of 1.684 would be achieved, what can be a representative value for the biggest values (the most reliable) of the shown ratios in the table.

All these values have their graphical representation in figure 35, that represents the relationship between the CPU times for the different procedures in all models and the logarithm of the number of generated elements -that has been substituted by the representative number for the respective models- divided by 10.

Most of the previous observations apply for the elapsed times. However, it can not be expected the same degree of proportionality and only for a perfectly balanced domain decomposition the quotient between the obtained times for every couple of procedures will have the same simple expression. In fact, in this case, the time spent for the four processors procedure will be $k\frac{x^{\propto}}{4^{\propto}}$ and $k\frac{x^{\propto}}{16^{\propto}}$ for the sixteen processors procedure, with a new common ratio between the consecutive procedures of $\frac{1}{4^{\propto}}$.

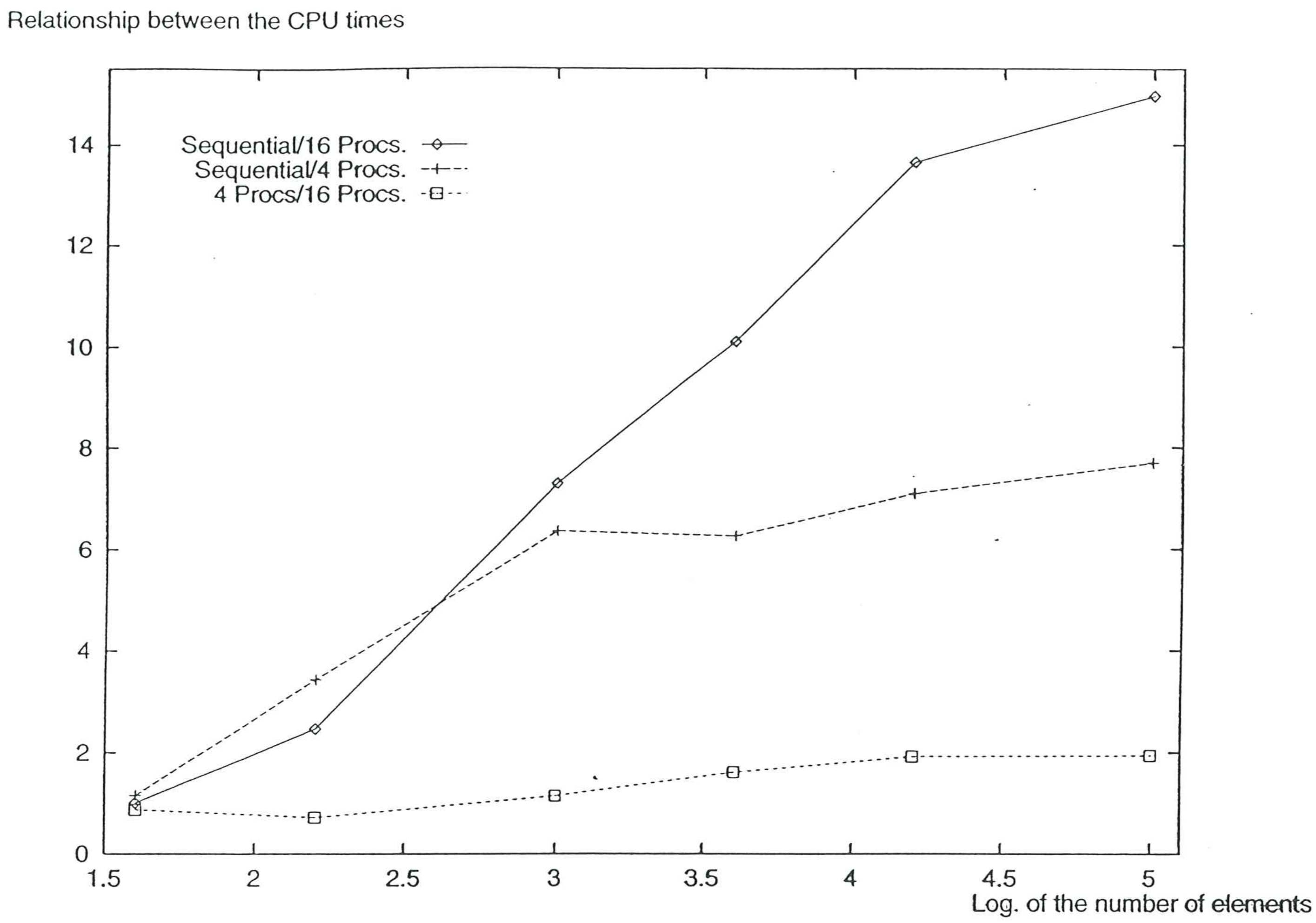


Figure 35. Relationship between the CPU times.

Again, the elapsed times are also presented. The quotients between the different procedures are shown in Table 6.

	BIG1	BIG2	BIG5	SMALL1	SMALL2	SMALL5
Seqtial./4 Procs.	3.417	9.702	17.116	17.898	18.286	20.466
4 Procs./16 Procs.	4.	2.765	3.662	4.238	5.288	4.925
Seqtial./16 Procs.	13.667	26.823	62.725	76.286	96.705	100.789

Table 6. Ratios between the elapsed times for the different procedures in all models.

All these values are equally represented in figure 36. In spite of having such a more simple mathematical significance than the total CPU time, the elapsed time is, to practical effects, the most important value to be considered. It represents, definitely, the difference in waiting time between different procedures, so the effective speed-up that can be achieved.

So, with the synchronism of the parallel processors, the time needed to run model SMALL5 sequentially is more than 20 times the time needed to run it with 4 processors and more than 100 times the time needed to run it with 16 processors.

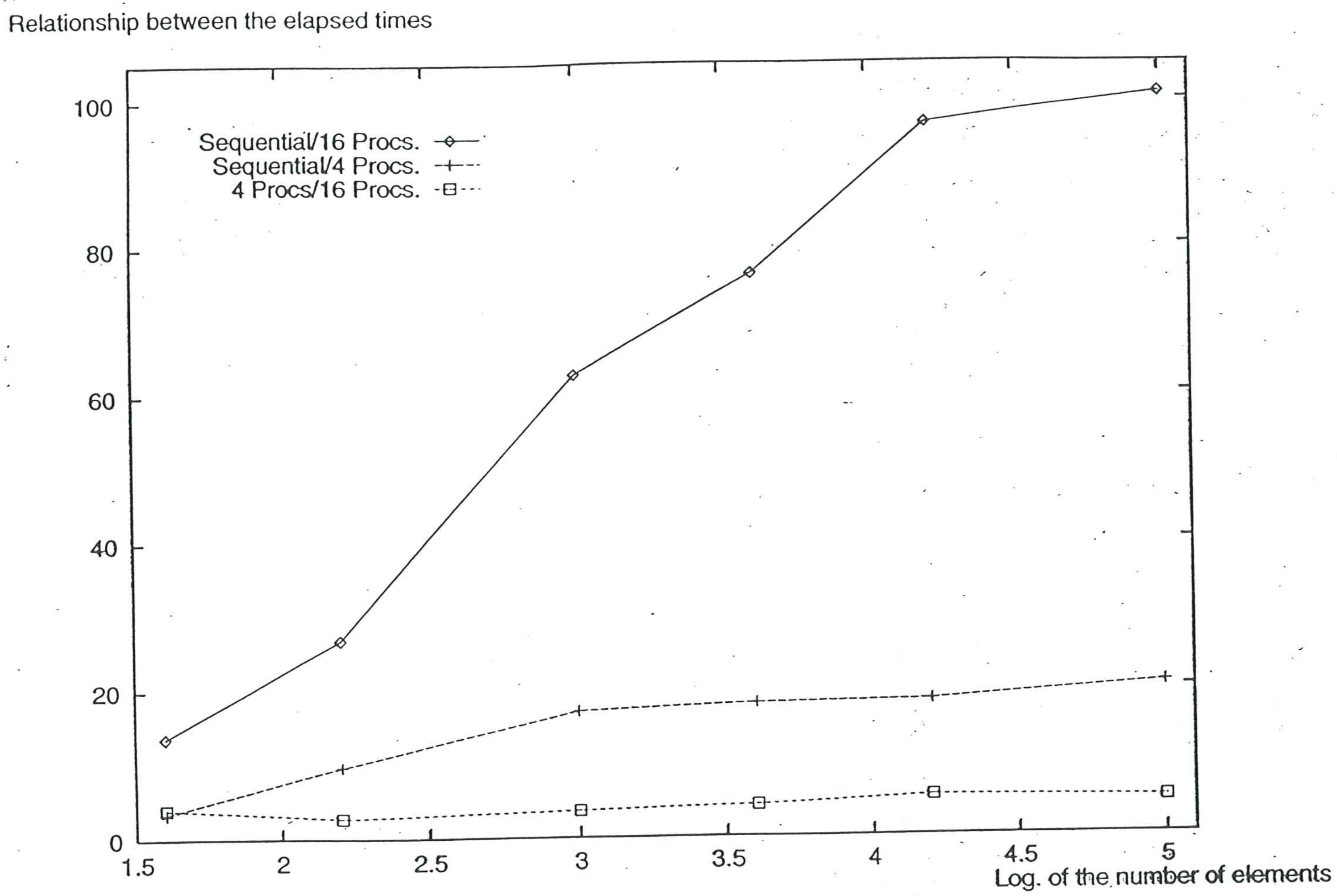


Figure 36. Relationship between the elapsed times.

These figures can be a proof of the conclusions of this study: the improvements presented for the different procedures agree with the goals of parallelization, important speed-ups and minimization of the interprocessor transfers of information.

5. EXAMPLE 1. AN ADAPTIVE MESH

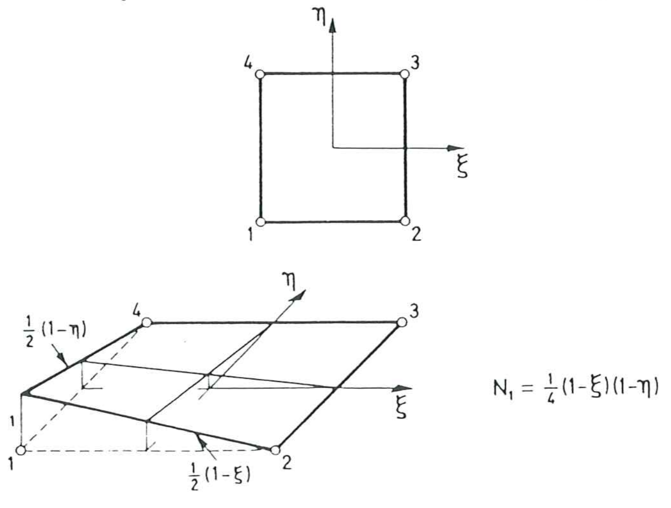
To illustrate an example of a simple mesh adaptation, the same domain that has been developed till now will be considered but, in order to have a clearer difference amongst the number of elements generated by the different processors for the considered decomposition, the spacings in the four corners will be now: 27, 1, 9 and 3, instead of 27, 11, 19 and 3, as it is shown in the figure 38, on the next page.

With these new spacings, we will get 18, 31, 46 and 23 elements for the processors number 0, 1, 2 and 3, respectively, obtaining the mesh showed in the figure 39, for the so-called model TEST, and the corresponding times, also showed in these pages.

A first approximation will consider the shape functions for a quadrilateral element of four nodes.

With a proper change of variables any quadrilateral domain can be converted into a normalized square, being the coordinates of its four corners: (-1,-1), (1,-1), (1,1) and (-1,1), what corresponds to the natural coordinates.

For this case, the shape functions are shown in figure 37 [3].



$$\boxed{N_i = \frac{1}{4}(1+\xi\xi_i)(1+\eta\eta_i)}$$

Nodo	ξ_i	η_i
1	-1	-1
2	1	-1
3	1	1
4	-1	1

Figure 37. Natural coordinates and shape functions for a quadrilateral element of four nodes.

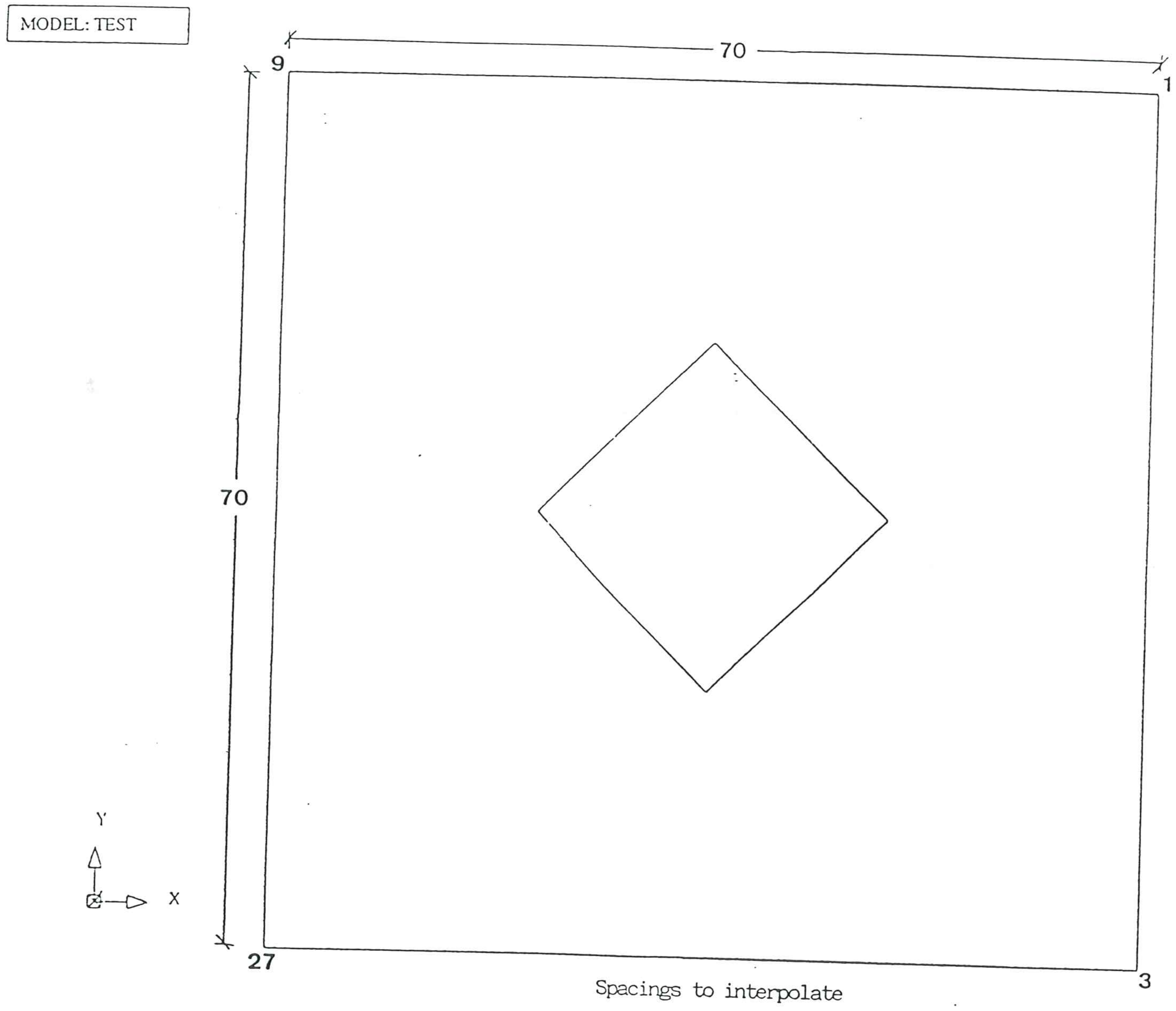
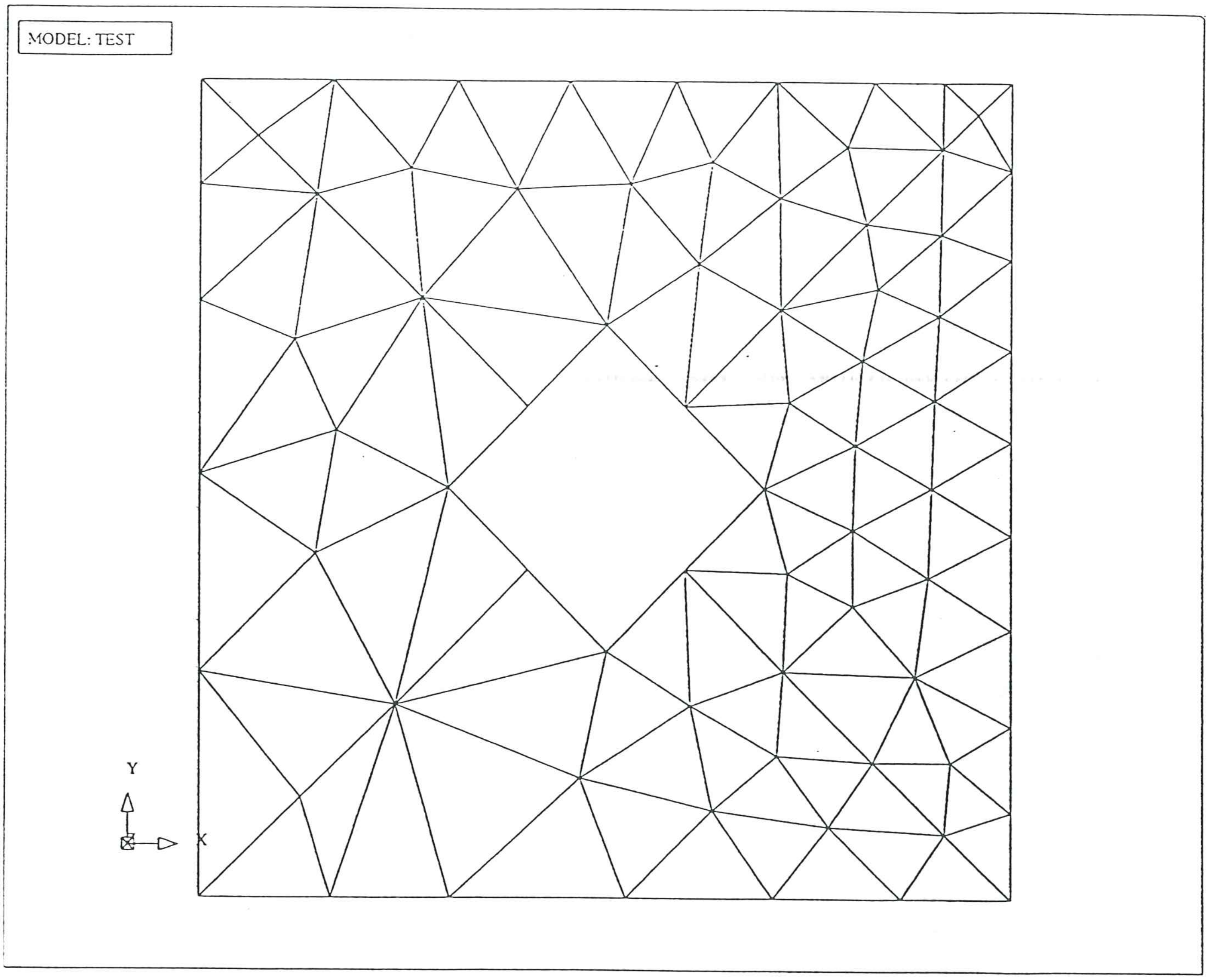


Figure 38. Spacings to interpolate (Model TEST)

```
MODEL TEST    04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=      18
Number of generated nodes=         17
                     Time spent by processor  0 has been       14 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=      31
Number of generated nodes=         26
                     Time spent by processor  1 has been       24 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=      46
Number of generated nodes=         34
                     Time spent by processor  2 has been       33 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=      23
Number of generated nodes=         20
                     Time spent by processor  3 has been       17 msec

Total number of generated elements=    118
Total number of generated nodes=        77

Maximum time spent to generate the mesh by a single processor=       33 msec
Total CPU time spent to generate the mesh=                           88 msec
```

```
MODEL TEST00 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=      15
Number of generated nodes=         14
                     Time spent by processor  0 has been       12 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=      21
Number of generated nodes=         18
                     Time spent by processor  1 has been       15 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=      45
Number of generated nodes=         32
                     Time spent by processor  2 has been       33 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=      34
Number of generated nodes=         26
                     Time spent by processor  3 has been       25 msec

Total number of generated elements=    115
Total number of generated nodes=        77

Maximum time spent to generate the mesh by a single processor=       33 msec
Total CPU time spent to generate the mesh=                           86 msec
```



Maximum processor's time: 33 msec, total time: 88 msec

Figure 39. First mesh. (Model TEST)

By their own definition, the shape functions take the value 1 at their corresponding nodes, and 0 in the rest. However, to take in consideration the different spacings for the different nodes, it will be necessary to define a weighting factor for each function to represent these differences.

The same weighting factor for all the functions will imply equal subdomains for each processor, it is, a quarter of the normalized square.

For our case, it will represent a quarter of the real domain for every processor. With this decomposition, we will obtain 15, 21, 45 and 34 elements, getting the auxiliary mesh showed in figure 39.

This coarse mesh will serve to achieve an estimation of the different proportions of the domain to be handed to each processor.

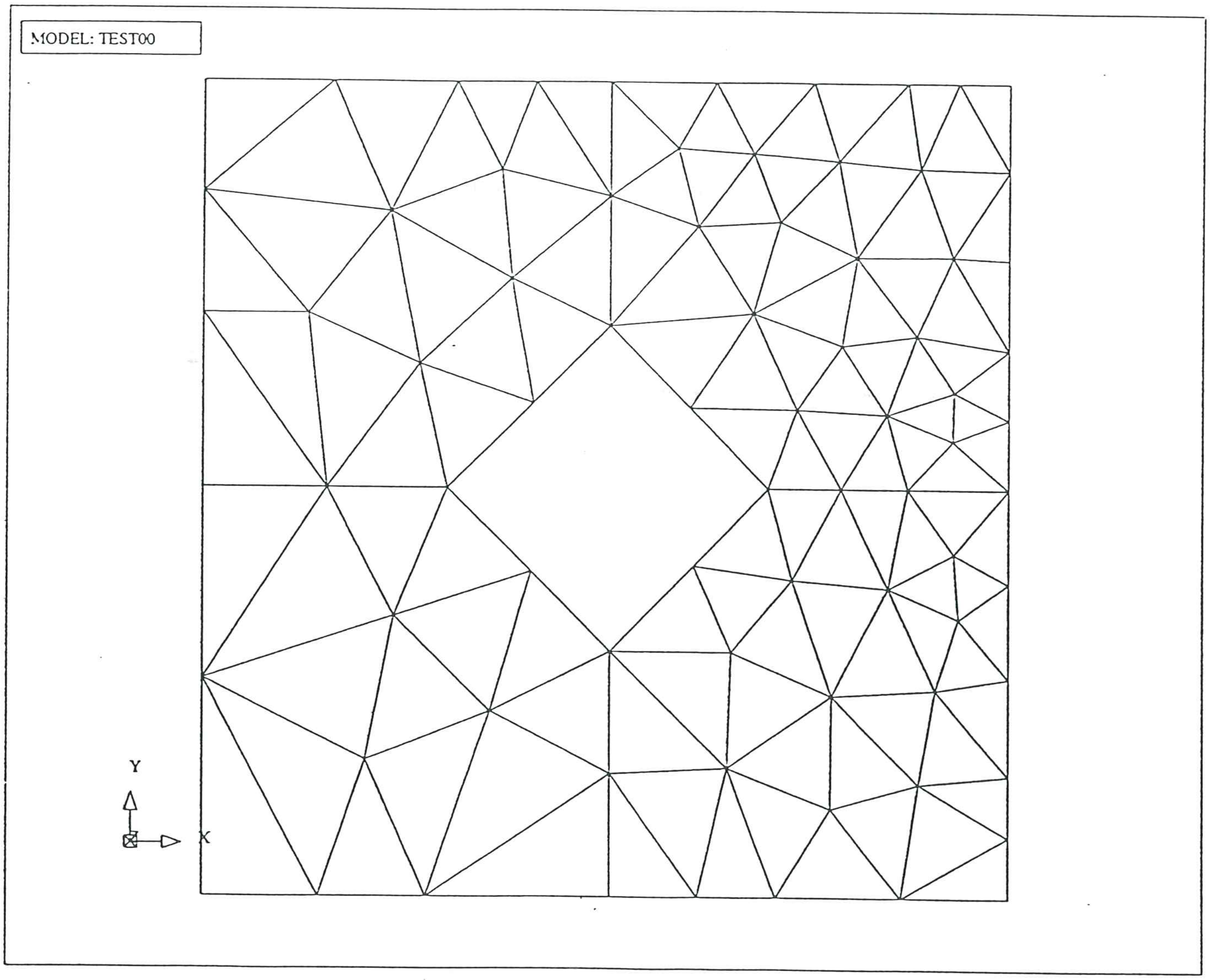
A way to do it could be to consider between every couple of adjacent nodes, a relative weighting factor equal to the number of elements achieved by the neighbor.

In our case, for instance, it will represent for the couple of the processors number 0 and 1, weighting factors of 21 and 15, respectively; for the couple of the processors number 1 and 2, weighting factors of 45 and 21, respectively, and so on.

For every processor we could consider an influence area equal to the zone where its shape function is bigger than the one of the neighbor.

In our case, for the couple of the processors number 0 and 1, the line what will divide their respective areas, it is, the geometric space where the shape functions are the same is the line $\eta=1/6$, what corresponds in real coordinates for our usual reference system, $y=5.83$.

For the couple of the processors number 1 and 2, the line what will divide their respective areas, it is, the geometric space where the shape functions are the same is the line $\xi=24/66$, what corresponds in real coordinates for our usual reference system, $x=12.72$.



Maximum processor's time: 33 msec, total time: 86 msec

Figure 40. Auxiliary mesh. (Model TEST00)

Operating in the same way for the other two couples of processors, we will get the line y=4.873 for the couple of processors number 2 and 3, and the line x=13.571 for the couple of processors number 3 and 0.

To get balanced subdomains to be meshed, we could join the points of intersection of every line with the segment of the contour comprised between the two referenced processors with the opposite, resulting two lines that intersect in a point, of natural coordinates (0.374, 0.148) and real ones (13.090, 5.174) -figure 41-, what could be approximately considered a point equally influenced by the four shape functions.

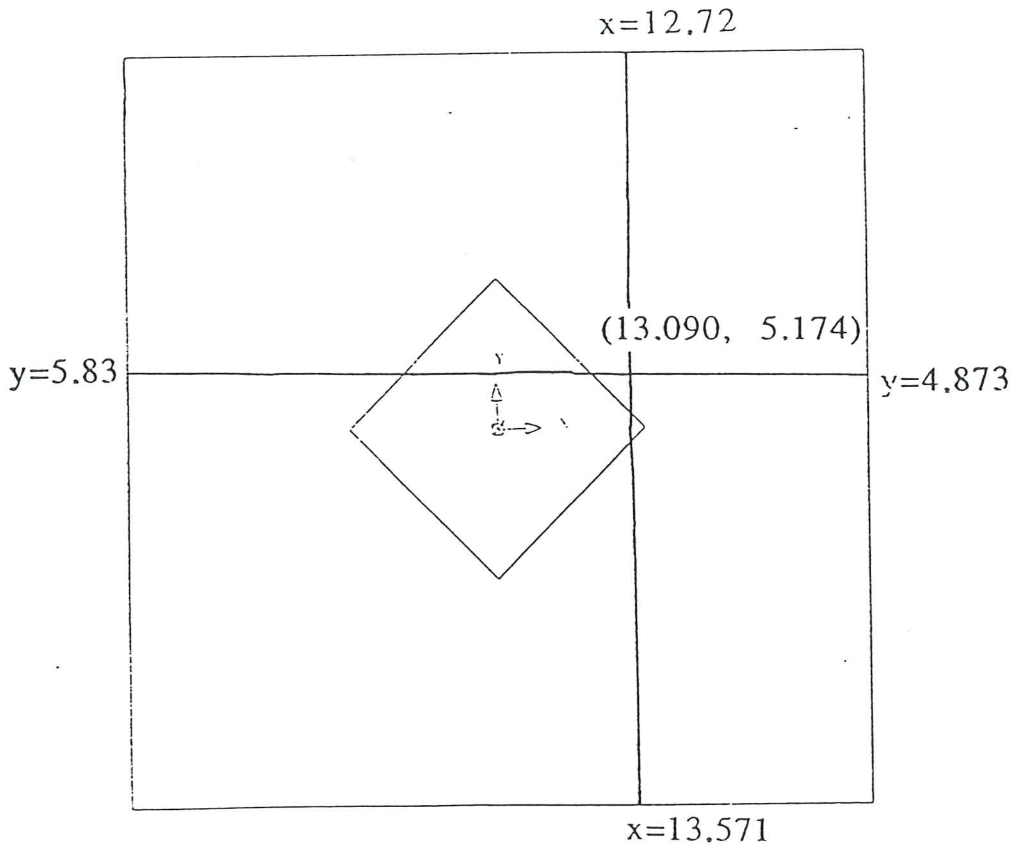


Figure 41. Intersection point.

In such a way; a division of four balanced subdomains -figure 42- has been obtained, and recursively operating there would have been obtained sixteen, or eight, for example, if eight representative nodes had been taken in consideration.

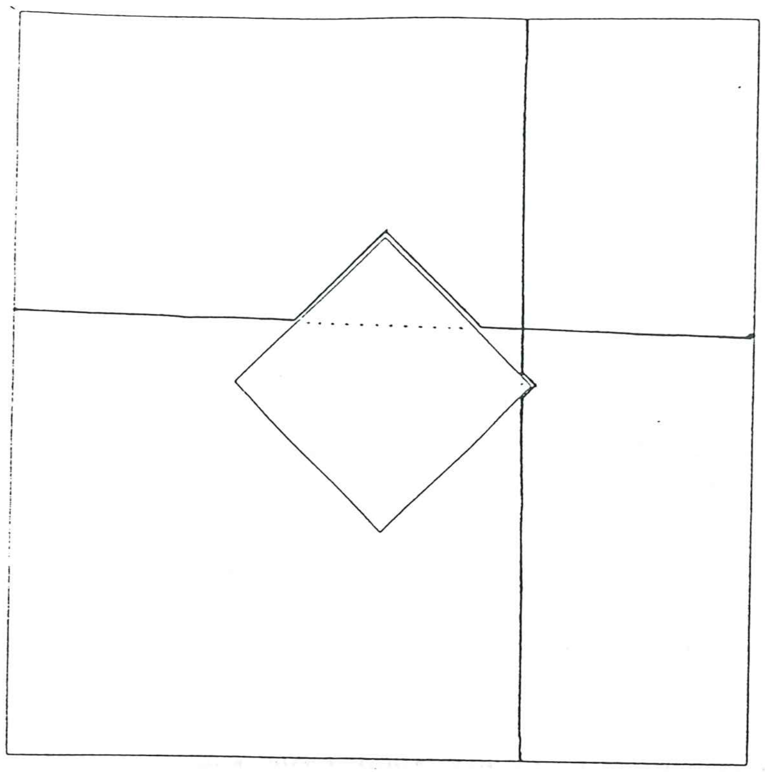


Figure 42. Balanced subdomains.

To simplify the study and to avoid polygonals of very short sides, due to the proximity of the found intersection point with one of the inside corners of the domain, the subdomains to be meshed are represented by the figure 43. There will be a little increase on the area to be meshed by processors number 0 (whose area included a part beyond the hole) and number 2 and a reduction for processors number 1 (with an augmentation, however, on a more densified zone) and number 3.

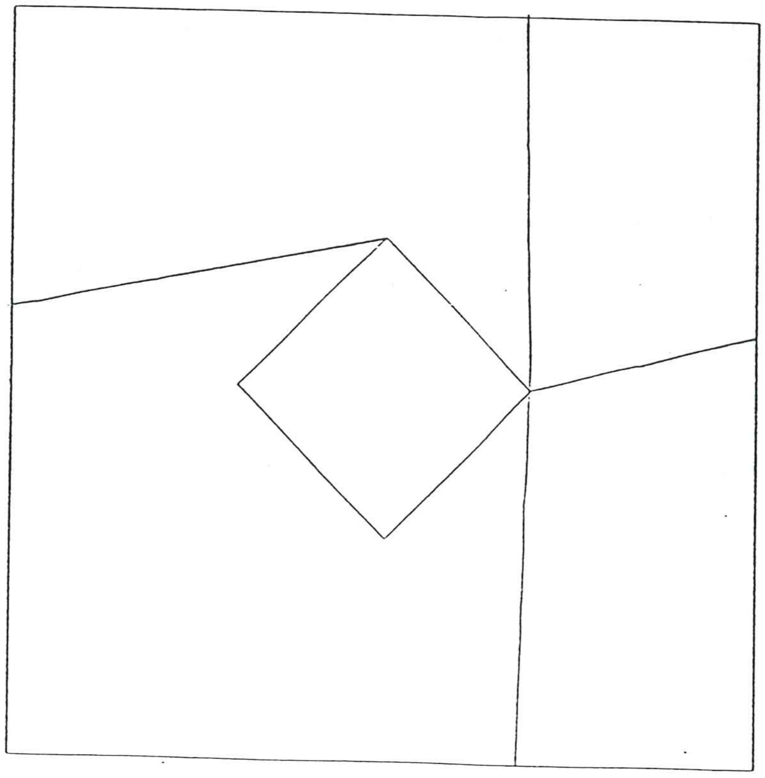


Figure 43. Study subdomains.

With the defined subdomains, the meshes relative to the different processors are shown in figure 44, whose assembly gives the resulting mesh shown in figure 45.

Obviously, it is a far better balanced mesh than the two previously considered. For the first domain decomposition, the range of times varies between 14 and 33 milliseconds for a total of 88 milliseconds, whilst with the auxiliary mesh, the range was broader, from 12 to 33 milliseconds, for a smaller total of 86 milliseconds.

With the adapted mesh, a better sharing of work is observed, with only a difference of 3 milliseconds between the quickest processor (processor number 3) and the slowest ones (processors number 0 and number 1), to achieve a total CPU time of 79 milliseconds, approximately a reduction of a 10% about the other two considered meshes total CPU times.

It is a result of a closer number of elements and nodes generated by every processor.

In the adapted case, the number of generated elements goes from 26 to 31, whereas in the first domain decomposition, it goes from 18 to 46, and in the auxiliary mesh, it goes from 15 to 45.

In a similar way, in the adapted case, the number of generated nodes goes from 21 to 26, whereas in the first domain decomposition, it goes from 17 to 34, and in the auxiliary mesh, it goes from 14 to 32.

Relative to the elapsed times the improvement is still bigger. Whilst in the two first meshes, this value was 33 milliseconds, for the adaptde mesh it was only 21 milliseconds, what represents a reduction of more than a 36%.

This simple example has showed the importance of a good domain decomposition and, even with the approximations, an easy way to get it has been showed.

```
MODEL TESTAD 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=     31
Number of generated nodes=        26
                     Time spent by processor  0 has been      21 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=     31
Number of generated nodes=        25
                     Time spent by processor  1 has been      21 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=     28
Number of generated nodes=        22
                     Time spent by processor  2 has been      19 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=     26
Number of generated nodes=        21
                     Time spent by processor  3 has been      18 msec

Total number of generated elements=    116
Total number of generated nodes=        77

Maximum time spent to generate the mesh by a single processor=      21 msec
Total CPU time spent to generate the mesh=                          79 msec
```

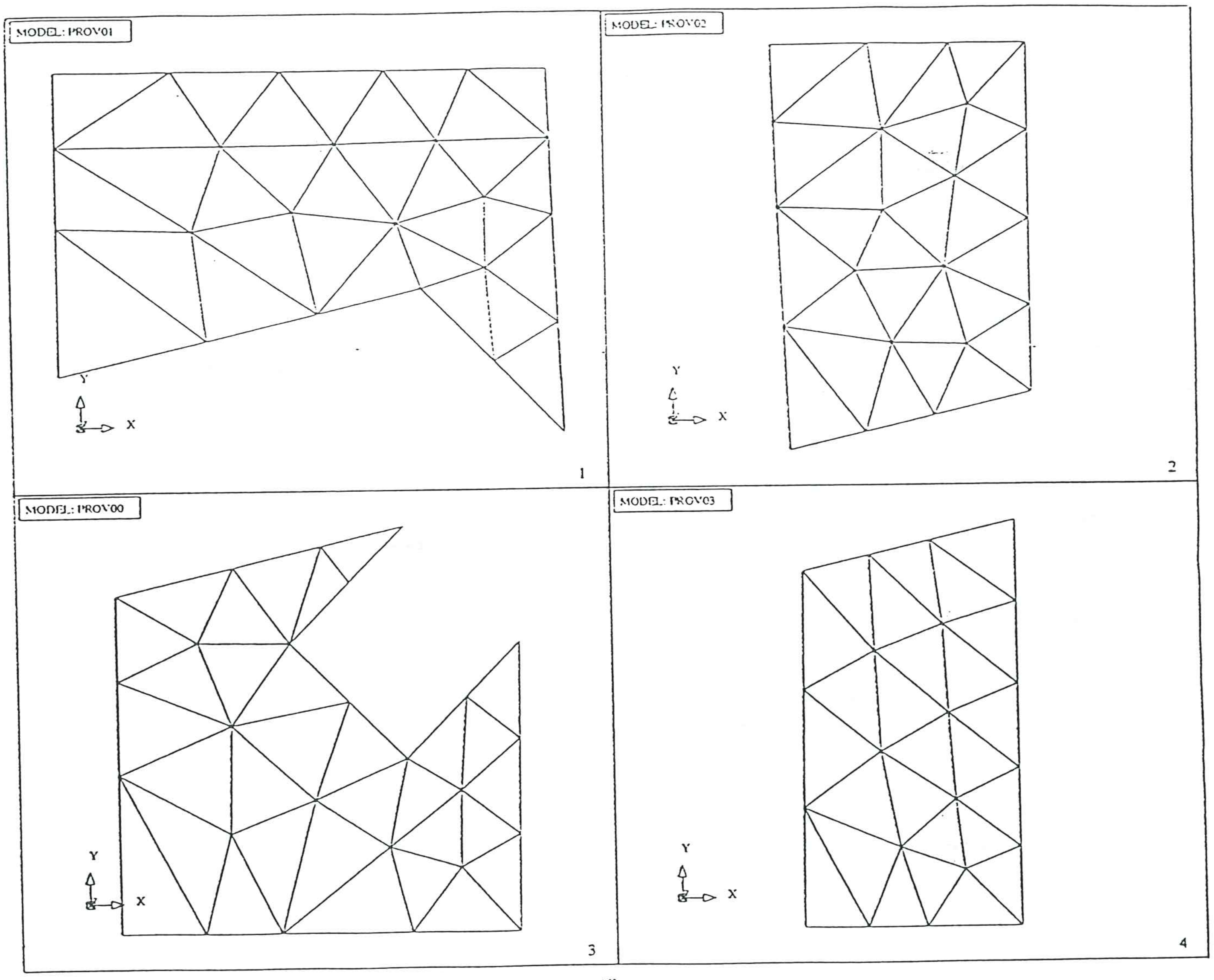
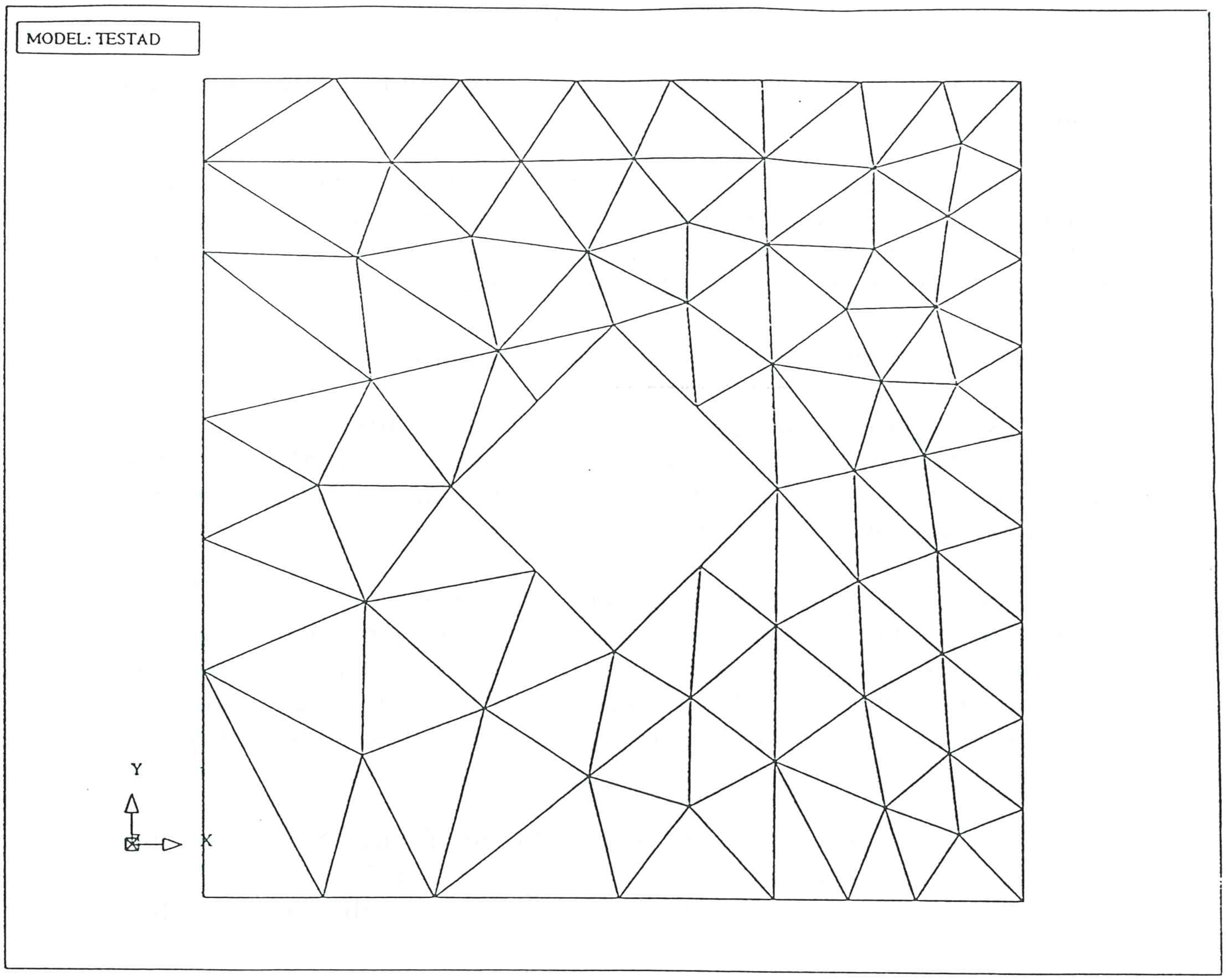


Figure 44. All processors for the adapted mesh. (Model TESTAD)



Maximum processor's time 21 msec, total time: 79 msec

Figure 45. Adapted mesh. (Model TESTAD)

6. EXAMPLE 2. A REAL MESH

To finish this report the final mesh of a real example is presented. The example has been taken from a double ellipse representing the front of an aircraft and it has been taken from a well-known workshop on hypersonic flows [4].

Two procedures have been run, the sequential case and with four processors.

In the first case, 315 elements for 183 nodes have been obtained, whereas with four processors there are 305 elements for 179 nodes, although two new nodes on the exterior contour have been added to better define the subdomains.

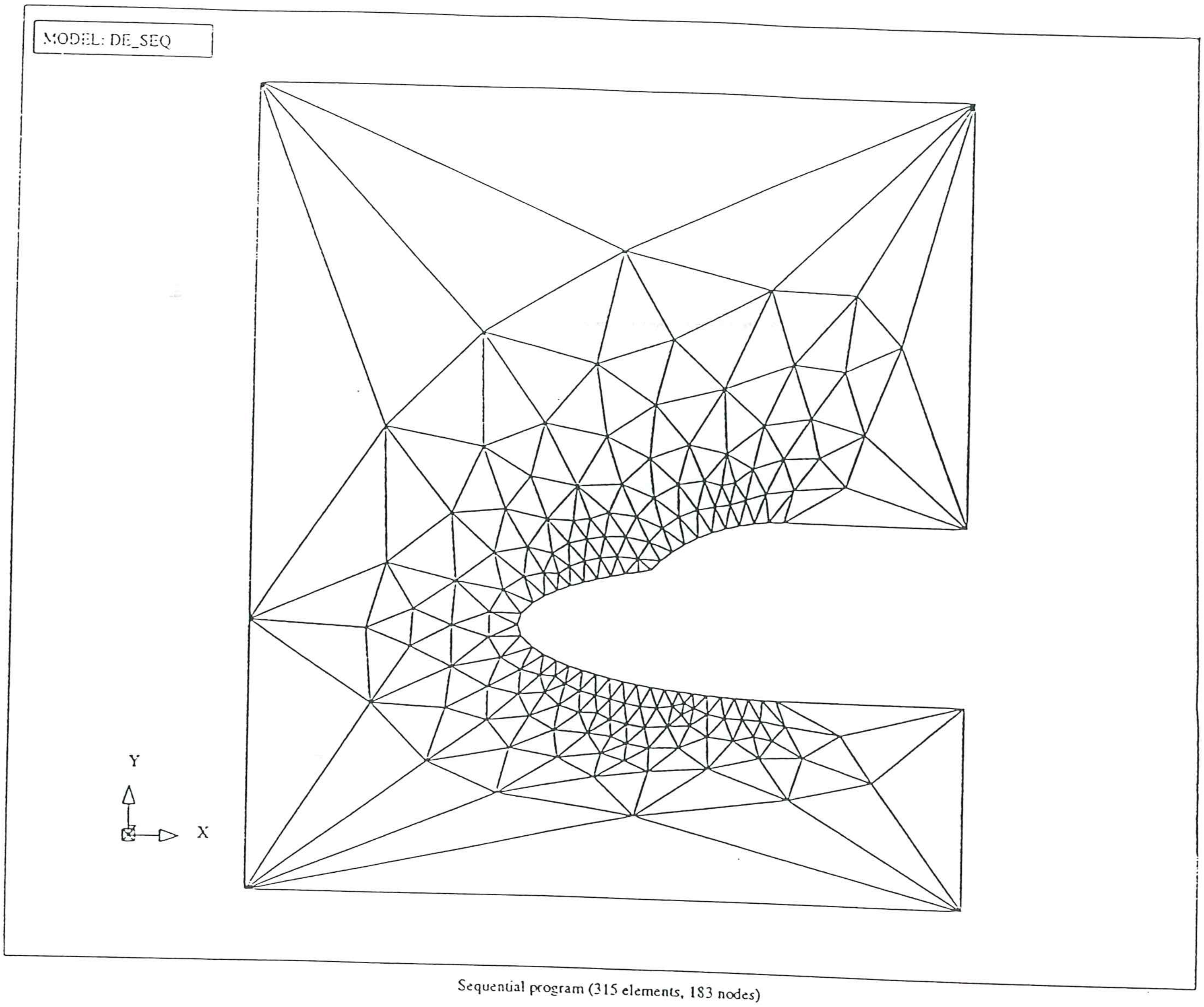
Both overall meshes are shown in figures 46 and 47.

It is important to point out the great difference in time observed between both procedures, as the sequential one takes more than a second versus little more than a fifth from the procedure run with four processors.

This great difference between both procedures, nearly five times, bigger than what should be expected in the domain seen in the previous chapters, is mainly due to the generation of the contour of the aircraft, much more complicated than the generation in the square domain, what penalizes the procedure that lasts more.

Successive figures -from figure 48 to figure 52- show the meshes corresponding to each processor and, finally, in figure 53, it can be seen the final smoothed mesh, once all the individual meshes have been joined for the smoothing.

```
MODEL DE_SEQ (Sequential)

Number of generated elements=     315
Number of generated nodes=        183

The time spent to generate the mesh has been        1073 msec
```



Sequential program (315 elements, 183 nodes)

Figure 46. Sequential program (Model DOUBLE ELLIPSE)

```
MODEL DEL_4p 04 PROCESSORS

::::::::::::
processor 00
::::::::::::
Number of generated elements=      79
Number of generated nodes=         52
                      Time spent by processor  0 has been        99 msec

::::::::::::
processor 01
::::::::::::
Number of generated elements=     112
Number of generated nodes=         71
                      Time spent by processor  1 has been        57 msec

::::::::::::
processor 02
::::::::::::
Number of generated elements=      54
Number of generated nodes=         36
                      Time spent by processor  2 has been        25 msec

::::::::::::
processor 03
::::::::::::
Number of generated elements=      60
Number of generated nodes=         40
                      Time spent by processor  3 has been        38 msec

Total number of generated elements=      305
Total number of generated nodes=         179

Maximum time spent to generate the mesh by a single processor=      99 msec
Total CPU time spent to generate the mesh=                         219 msec
```

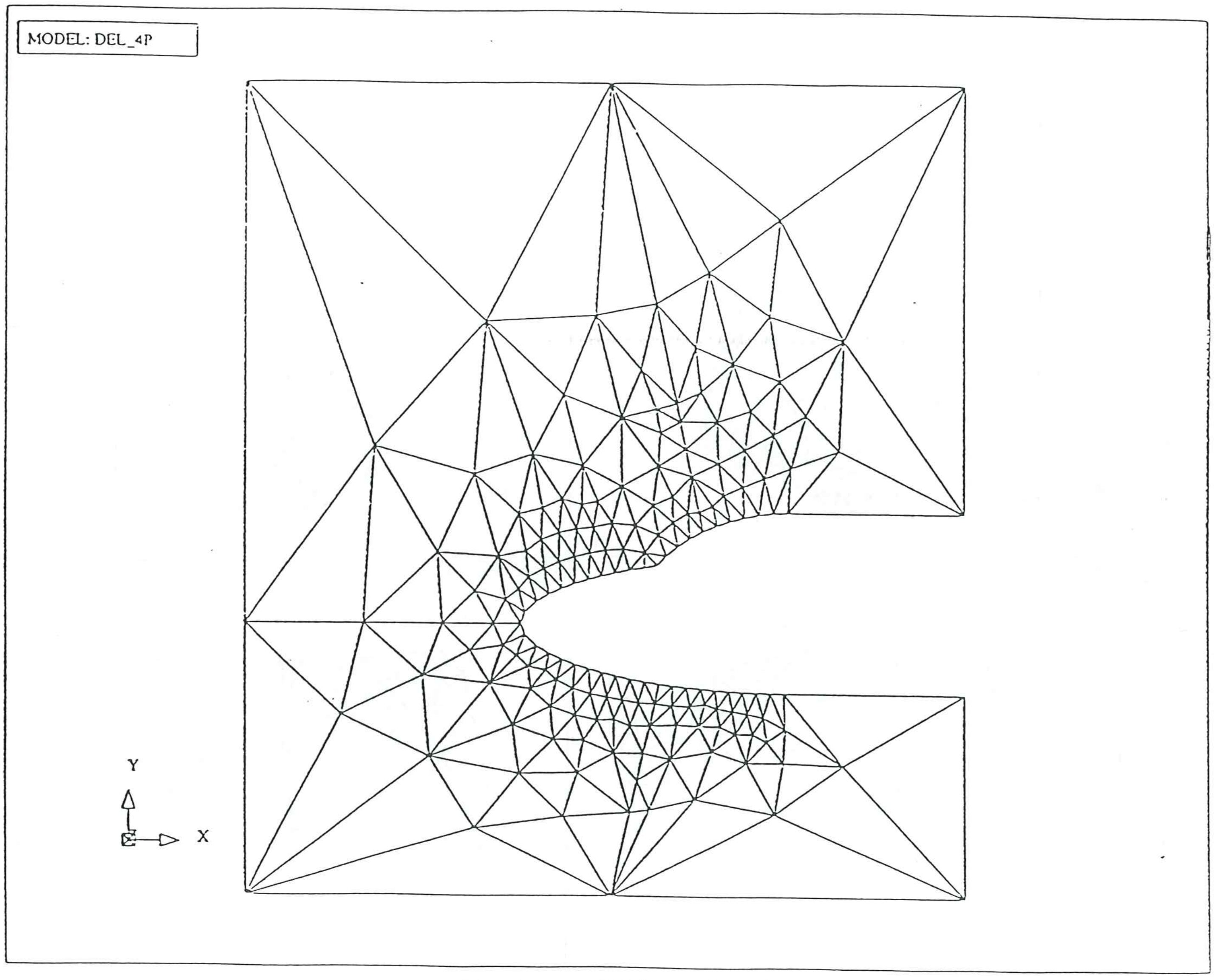


Figure 47. Whole mesh (Model DOUBLE ELLIPSE, 4 Processors)

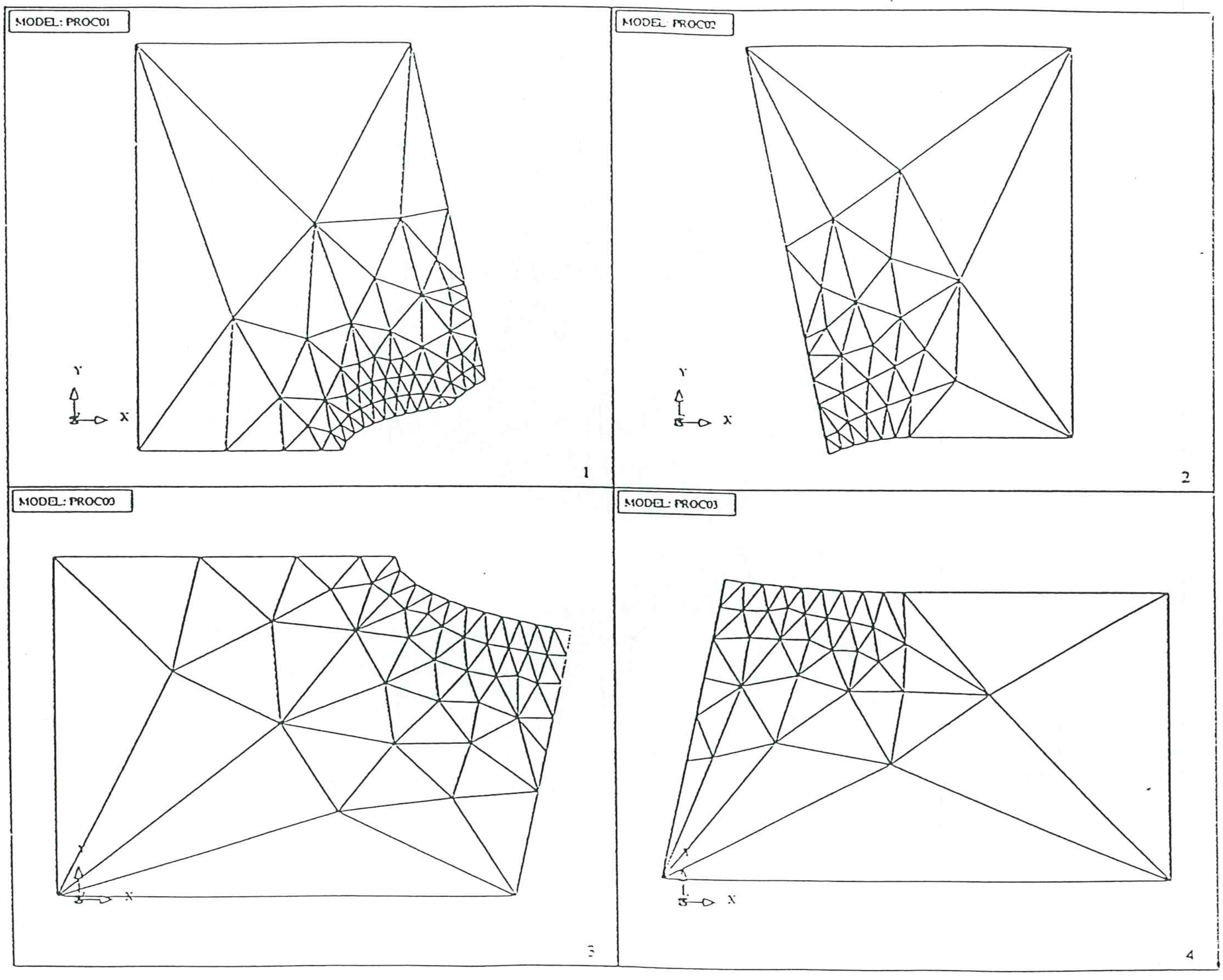
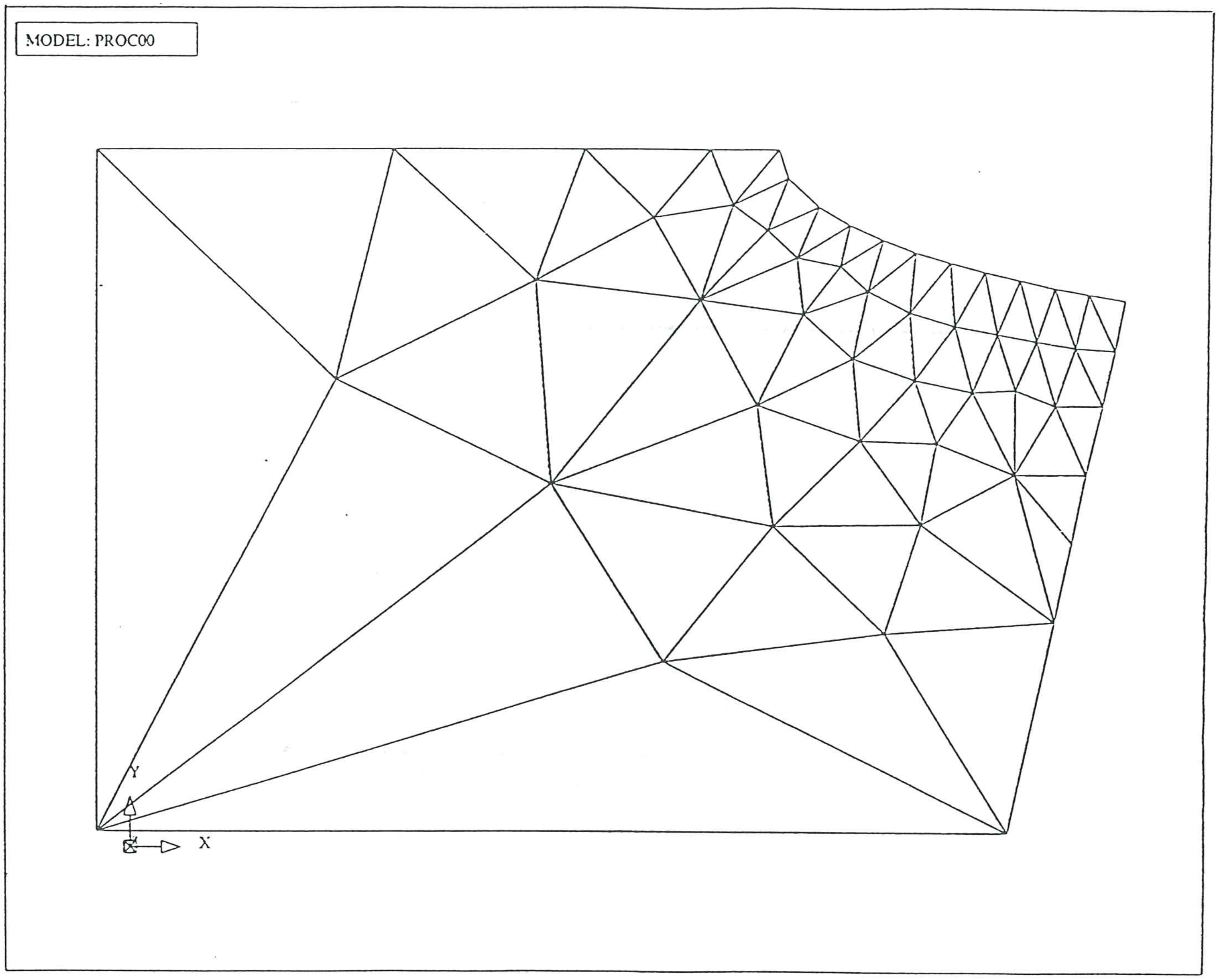
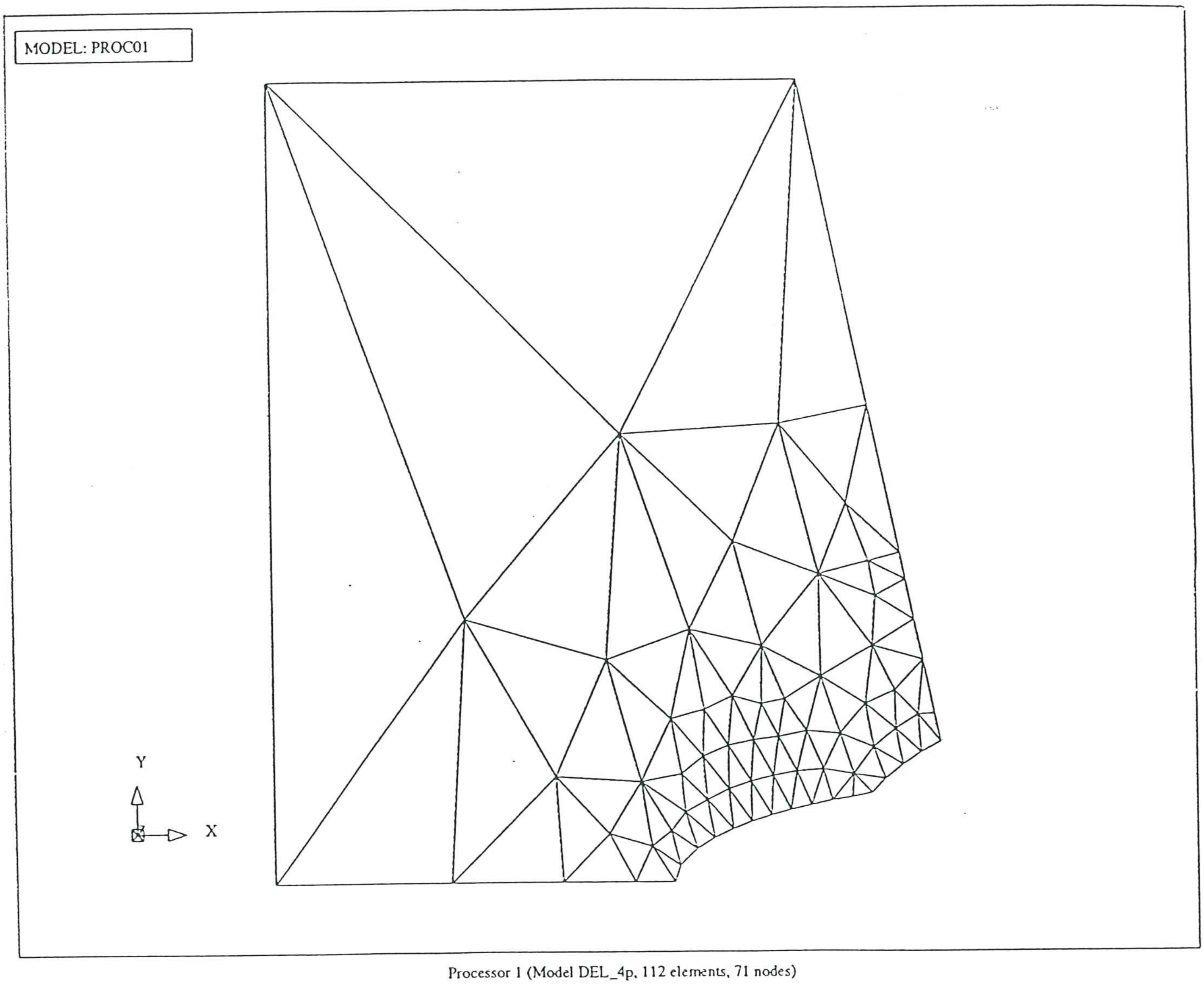


Figure 48. All Processors (Model DOUBLE ELLIPSE, 4 Processors)



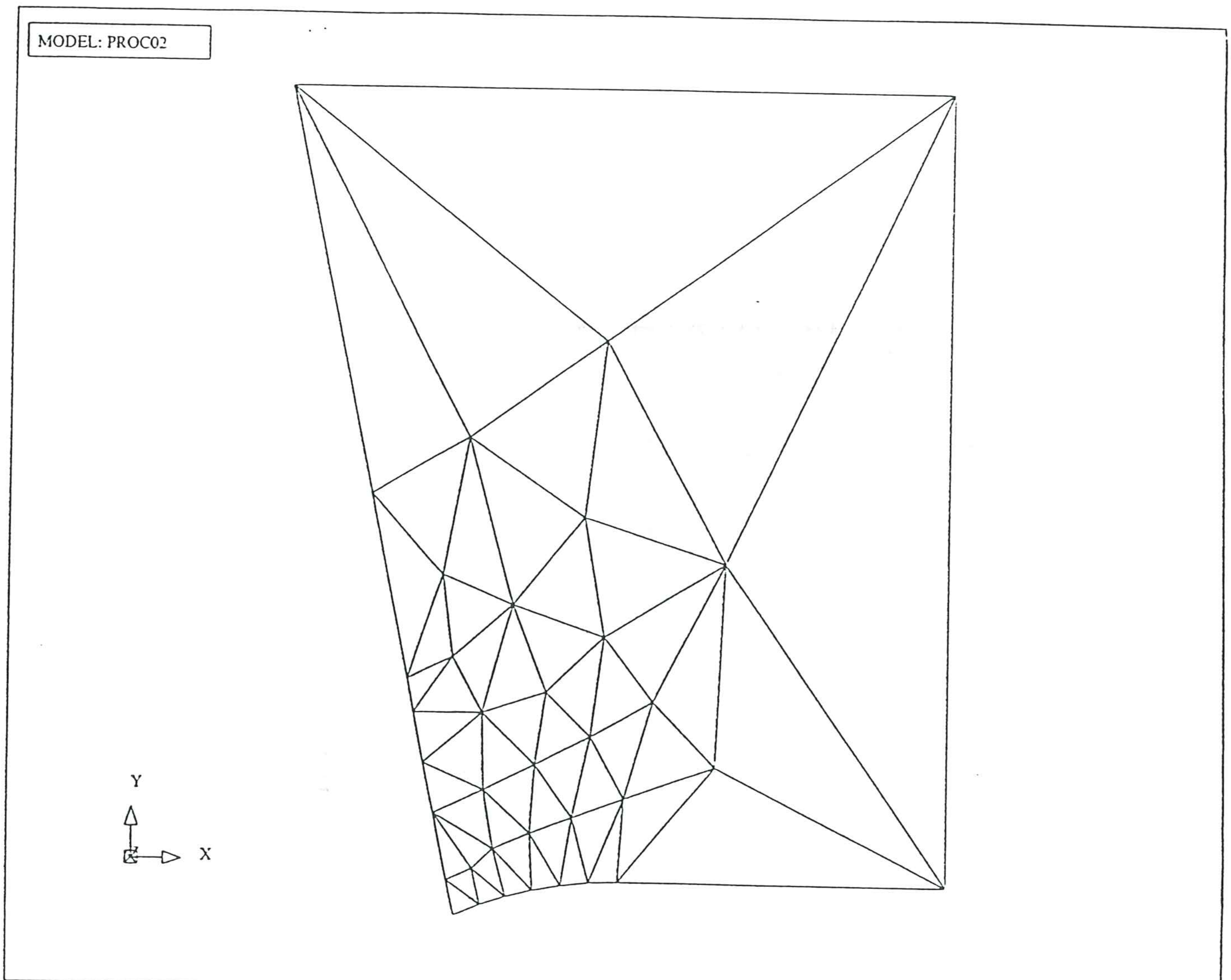
Processor 0 (Model DEL_4p, 79 elements, 52 nodes)

Figure 49. Processor number 0 (Model DOUBLE ELLIPSE, 4 Processors)



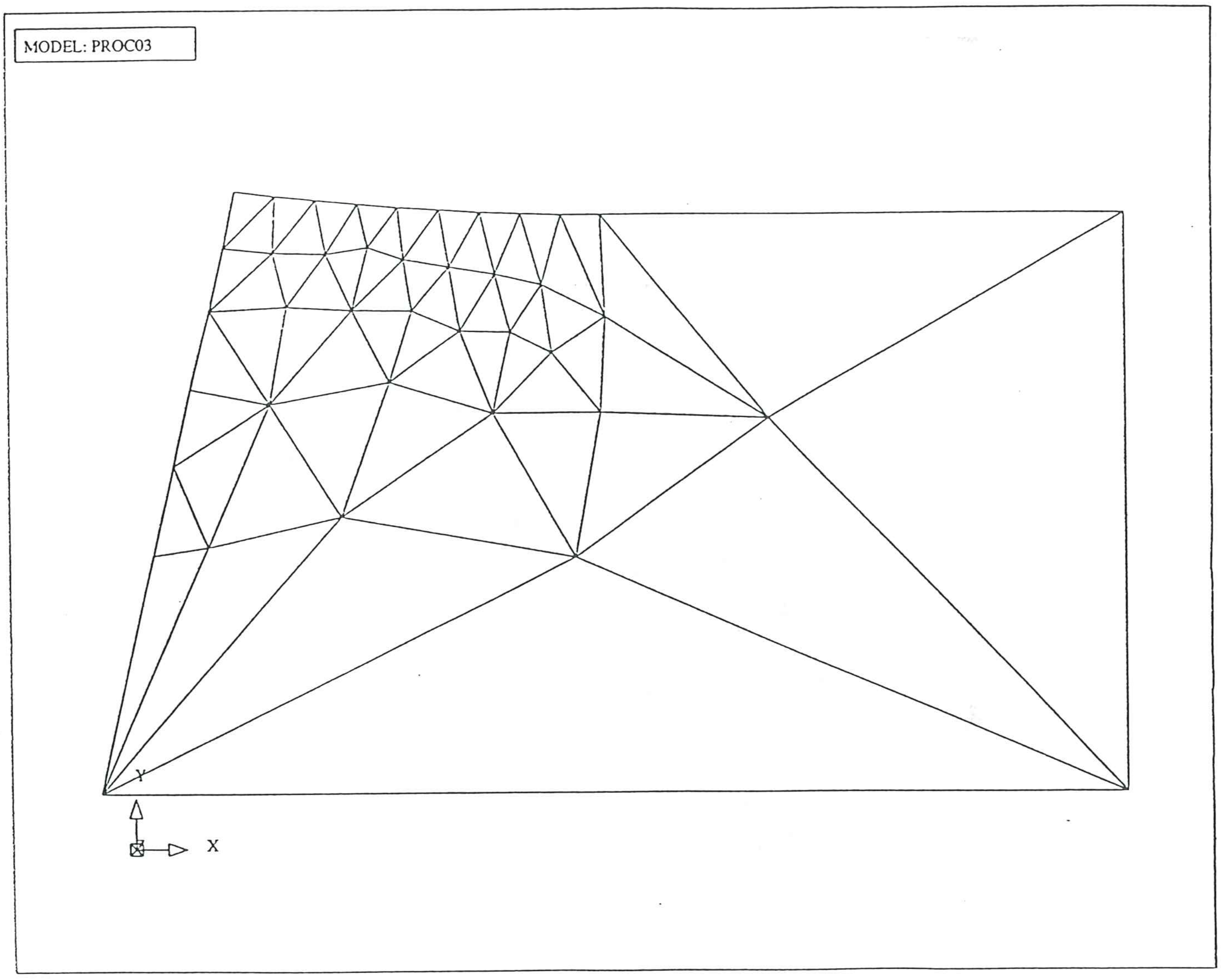
Processor 1 (Model DEL_4p, 112 elements, 71 nodes)

Figure 50. Processor number 1 (Model DOUBLE ELLIPSE, 4 Processors)



Processor 2 (Model DEL_4p, 54 elements, 36 nodes)

Figure 51. Processor number 2 (Model DOUBLE ELLIPSE, 4 Processors)



Processor 3 (Model DEL_4p, 60 elements, 40 nodes)

Figure 52. Processor number 3 (Model DOUBLE ELLIPSE, 4 Processors)

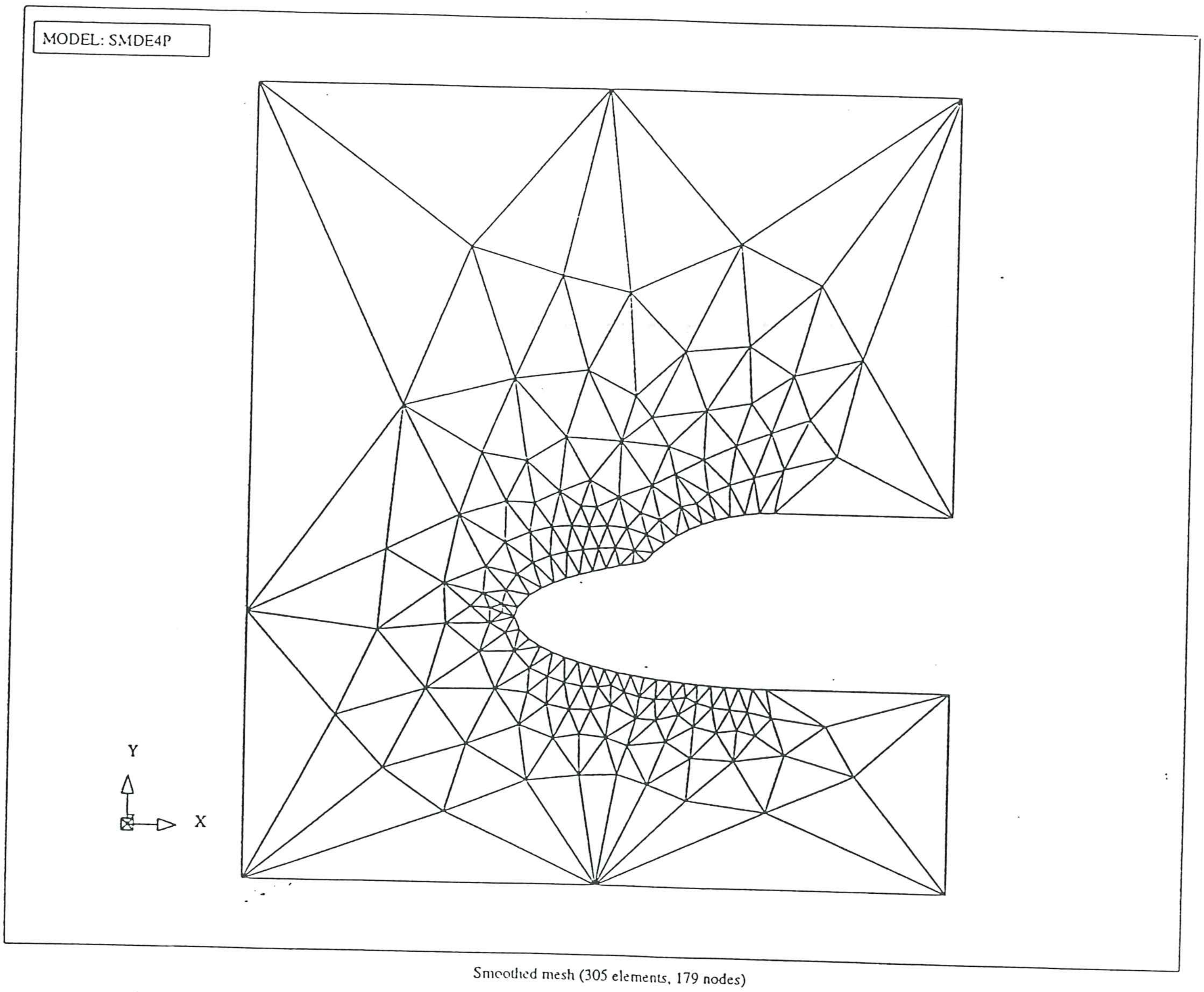


Figure 53. Smoothed mesh (Model DOUBLE ELLIPSE, 4 Processors)

ACKNOWLEDGEMENTS

The authors would like to thank CRAY Research, Inc. for its generous support, and in particular for allowing the access to its CRAY T3D computer at the École Polytechnique Fédérale de Lausanne, as well as the Fundació Catalana per a la Recerca, who partially funded this work by providing an scholarship to one of the authors.

REFERENCES

[1] Löhner R., Camberos J. and Merriam M., *"Parallel Unstructured Grid Generation"*, Computer Methods in Applied Mechanics and Engineering 95, pp. 343-57, 1992.

[2] Löhner R., *"Robust, Vectorized Interpolation Algorithms for Unstructured Grids"*, GMU/CSI, the George Mason University Publications, Fairfax, 1994.

[3] Oñate, E., *"Cálculo de Estructuras por el Método de Elementos Finitos"*, 1st. Edition, CIMNE, International Center for Numerical Methods in Engineering, Barcelona, pp. 196-7, 1992.

[4] Radespiel R., Herrmann U., Longo J.M.A., *"Flow over a double ellipsoid"*, Workshop on Hypersonic Flows for Reentry Problems, January 22-25, 1990, Antibes (France), INRIA, Institut National de Recherche en Informatique et en Automatique, Paris, Tome 4, p. III, 1990.