SAND88–1326
Unlimited Distribution
Printed June 1988
Second Printing May 1991
Third Printing January 1992
Fourth Printing August 1992
Fifth Printing August 1993

Distribution
UC-32

FASTQ Users Manual Version 1.2

Ted D. Blacker
Applied Mechanics Division II
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

The FASTQ code is an interactive two–dimensional finite element mesh generation program. It is designed to provide a powerful and efficient tool to both reduce the time required of an analyst to generate a mesh, and to improve the capacity to generate good meshes in arbitrary geometries. It is based on a mapping technique and employs a set of *higher–order* primitives which have been developed for automatic meshing of commonly encountered shapes (i.e. the triangle, semi–circle, etc.) and conditions (i.e. mesh transitioning from coarse to fine mesh size.) FASTQ has been designed to allow user flexibility and control. The user interface is built on a layered command level structure. Multiple utilities are provided for input, manipulation, and display of the geometric information, as well as for direct control, adjustment, and display of the generated mesh. Enhanced boundary flagging has been incorporated and multiple element types and output formats are supported.

Contents

Figures

1. Introduction

The **FASTQ** code is an interactive two–dimensional finite element mesh generation program. It provides a powerful and efficient tool to both reduce the analyst's time generating a mesh, and improve the general capacity to generate *good* mesh in general geometries.

FASTQ's roots emerge from several earlier mesh generation codes, including **QMESH** [1,2] , **RENUM** [1], and **QNUM** which were written by Ron E. Jones. Although the basic mapping mesh generation concepts have been stolen[1] from these codes, the routines used have been rewritten, upgraded to FORTRAN77, and enhanced to provide greater capacity and maintainability before being incorporated. **FASTQ** also employs a set of *higher–order* primitives which have been developed for automatic meshing of commonly encountered shapes[2] and conditions.[3]

FASTQ has been designed to allow user flexibility and control. The user interface is built on a layered command level structure. Multiple utilities are provided for input, manipulation, and display of the geometric information, as well as for direct control, adjustment, and display of the generated mesh. Boundary flagging of nodes as well as element sides has been incorporated as explained in [3]. Although the basis of the code has been developed for four node quadrilateral mesh generation, eight and nine node quadrilateral elements as well as two and three node bar elements are also supported. Output can be written in either the **GENESIS**[3], **ABAQUS**[4], or **NASTRAN**[5] formats.

Although knowledge of a few basic commands is sufficient to begin using **FASTQ** on a production basis, much of the power of the code is contained in the *higher–order* meshing primitives, various interactive options, and useful utilities. Thus, the user is encouraged to review this manual carefully so as to gain full access to the utility of **FASTQ**.

1.1 Features

This section introduces the major features of **FASTQ**. Each of these features will be discussed in detail in later sections of the manual.

[1] "Lesser artists borrow, great artists steal." Igor Stravinsky

[2] Such shapes include the triangle, the semi–circle, the circle, and the pentagon.

[3] Mesh transitioning from coarse to fine mesh size is such a condition.

1.1.1 User Interface

The user interface in **FASTQ** facilitates ease of use and familiarity. Simply typing FASTQ will initiate the code if installed correctly on your computer. Several optional run-line parameters are supported as explained in Section 4.4. The interface is based on a layered command structure which allows the user to tell the program what to do rather than forcing the program to ask the user for each instruction as explained in Section 2.1. This design allows the user to quickly access only those utilities needed while bypassing any unnecessary steps. This layered structure begins with a list of main options from which the user indicates his/her choice. If the option or utility chosen contains other options, the user again indicates which suboption is appropriate. This continues until the option chosen represents some procedure to execute rather than a list of further choices. This layered structure is at most three levels deep and the current level is always apparent from the prompt displayed on the screen.

1.1.2 Help Facility

A help facility is accessible in **FASTQ** at any level of the command structure. If the user wishes to review his/her available options, simply typing HELP will display a list of options with a brief description of their purposes. If an incorrect option is chosen, the user will be presented with this help listing to allow for an appropriate choice. The help option is fully explained in Section 2.2.

1.1.3 Geometric Data Input and Validation

The geometry to be meshed can be input in a number of ways. Describing the geometry involves the definition of entities such as points, lines, regions, etc. as explained in Section 3.1. These entities can be defined by:

1. **KEYIN.** Input at the keyboard using the *KEYIN* option.
2. **FASTQ File.** Reading a preconstructed free formatted ASCII file, referred to as the FASTQ file, using the main *READ* option. Since the FASTQ file is free formatted ASCII, any standard text editor may be used to view, correct, or generate the FASTQ file.
3. **DIGITIZE.** Digitizing the geometry directly from a drawing or connecting existing definitions (points to form lines, etc.) using a digitizing pad and the *DIGITIZE* option.

4. **All Of The Above.** Any imaginative combination of these techniques useful to the particular problem.

Once the geometry has been input into **FASTQ**, it can be validated and manipulated as explained in Section 3.3. Individual entities can be listed on the screen using the *LIST* option, adjusted or corrected using the *KEYIN* option, or deleted using the *DELETE* or *FLUSH* options. The geometry can also be graphically displayed using the main *GRAPHICS* option as explained in Section 3.4. A number of identifiers such as point numbering, line intervals, and boundary flags can be optionally displayed to aid in quick visual verification.

1.1.4 Mesh Generation and Validation

The actual generation of the mesh proceeds once the geometry has been defined. The user has varying degrees of control over the actual generation process ranging from completely automatic generation using the defined processing parameters, to control over individual mesh generation steps including a plot of the resulting mesh at each step. The mesh generation process is described in detail in Section 4.3.

FASTQ's design is based on four node quadrilateral mesh generation with other element types supported as discussed in Section 1.1.6. Quadrilateral elements are generated from geometric regions which are a closed set of lines without interior holes. Elements are generated using a mapping technique traditionally successful for rectangular shaped regions (rectangle primitive) with opposing sides of equal interval assignment. A set of *higher-order* primitives have been developed which enable the meshing of other primitive shapes including the circle, semi–circle, triangle, and quarter–circle as well as easy transitioning for rectangles with unequal opposing subdivision assignments. Special mesh adjustment techniques from the **QMESH** code[1,2] including smoothing, element deletion, and element restructuring are also provided. These techniques may be useful when the initial mesh is not of acceptable quality.

Once generated, the mesh can be interactively displayed and verified as explained in Section 4.6. This allows for the display of node numbers, element numbers, and boundary flags, as well as plotting by region number, element number, and material number.

1.1.5 Boundary, Loading, and Block ID Flags

Identifier flags can be attached to the mesh for use in defining the appropriate boundary conditions, loading conditions, and material properties. Actual loads and

boundary conditions are not input directly in **FASTQ**, but must be indicated in the input to the analysis code by association of the identifier flag and the condition. The same holds true for material properties. To adequately accomplish this task, three types of flags are available:

1. **Node Flag.** The node flag is attached to the individual nodes of the mesh (see Sections 3.1.9 and 3.1.10.) This flag can be used for point or distributed loadings, and for boundary restriction or imposed displacement.

2. **Element Side Flag.** The element side flag is attached to a side of individual elements in the mesh (see Section 3.1.11.) This flag is useful in defining pressure and internal mesh slide line definitions depending on the code used for the analysis.

3. Block ID Flag. The block ID flag is attached to all individual elements within a region or barset. This flag can be used for body loads and material identification. (see Sections 3.1.5 and 3.1.6.)

1.1.6 Supported Element Types and Connectivity

FASTQ was originally designed to support only the four node quadrilateral element, but has since been expanded to include two and three node bar elements, and eight and nine node fully isoparametric quadrilateral elements.

Two and three node bar elements are generated using groups of lines, or BARSETs, as explained in Section 4.3.1. Element ordering is critical in some applications, and can be controlled using an auxiliary point to indicate outside and inside of the elements to be generated. Mesh optimization is available to reduce the bandwidth and wavefront of the mesh.

Four node quadrilateral elements are generated from closed regions as explained in Section 4.3.2. The element connectivity is produced in counterclockwise order, and mesh optimization is available to provide a reduced bandwidth and wavefront mesh ordering.

Eight and nine node isoparametric elements are generated from the original four node quadrilateral mesh by the additional generation of midside and center nodes as explained in Section 4.4. Again, the element connectivity is generated in counterclockwise order, and mesh optimization is available.

1.1.7 Output File Format

The mesh can be output in three formats:

1. The **GENESIS**[3] format which is compatible with SNLA Department 1520's in house analysis codes.
2. The **ABAQUS**[4] input format requirements.
3. The **NASTRAN**[5] input format requirements.

All of these formats are discussed in Section 4.7. The **GENESIS** database has an advantage over the other two. It can be read back into **FASTQ** to verify a previously generated mesh as explained in Section 4.1.

1.1.8 Batch Mode Operation

FASTQ can also be used in batch mode. When run in batch mode, the program reads the input data from the specified FASTQ input file, generates the mesh automatically, and outputs that mesh into a GENESIS mesh file. Implementing **FASTQ** in batch mode is machine dependent, but has been provided for with command procedures available on the VAX/VMS computer system and on the CRAY/CTSS computer system. These procedures are covered in detail along with an explanation of several optional parameters in Section 2.3 of this manual.

1.1.9 Graphics Linkage and Support

The **FASTQ** code is device independent using 1520's Dual Virtual Device Interface (DVDI)[6] to load the appropriate terminal and hard copy drivers at run time. The terminal drivers offer plotting at the terminal and the hardcopy file devices allow the user to print a hardcopy version of the plot. The devices provided in DVDI include, but are not limited to:

- Lear Siegler 220/230 (Envision)
- Tektronics 4014, 4105, 4107, 4109, 4115, 4125, 4218, 4129, 4207, 4208
- Raster Tech One–25 (DMA mode or RS232 mode)
- QMS Laser Printer File

- HP 7580 Plotter File
- Imagen Laser Printer File

In general, no special capabilities of individual graphics terminals have been incorporated into the coding. The exception is that since the Lear Siegler terminal is used extensively in Department 1520, screen clearing and screen switching facilities have been taken advantage of to enhance user interaction. All graphics coding uses the **PLT** package[7] developed by Colin Selleck.

1.2 FUTURE ENHANCEMENTS

FASTQ is an evolving code and can be expected to change periodically to accomodate needs of the Finite Element Method analysts. Research is currently being conducted into further automating quadrilateral mesh generation by development of Artificial Intelligence (AI) Knowledge System techniques to automatically decompose any arbitrary geometry into meshable primitive regions[9]. These capabilities will be added to the code as they are developed. Any changes to **FASTQ** will be reflected in a news letter which is sent to users periodically. It is suggested that such news letters be appended to this manual by the user so that an up-to-date reference can be maintained.

1.3 MANUAL SUMMARY

The **FASTQ** user interface is discussed in detail in Chapter 2 along with optional run-line parameters. Chapter 3 covers geometric data definitions, input methods, manipulation, verification, and plotting. Chapter 4 explains mesh generation and display. Appendix A outlines the FASTQ file format and Appendix B gives an overview of the mouse buttons. Appendix C gives examples of actual FASTQ files along with the plotted geometric data and the corresponding mesh generated from the file. Appendix D gives an example interactive session with **FASTQ**.

2. USER INTERFACE

The **FASTQ** user interface is simple in structure to allow the beginning user or intermittent user easy access to the full facilities of the program, but direct enough that the experienced or well trained user should not be encumbered by unnecessary prompts and/or responses. A layered command structure is used for the interface to enable this flexibility. Some power is lost with this command structure in the sense that only a limited number of responses are allowable at any level. However, the simplicity of design and use enabled with this scheme were judged to be overriding benefits as described in this chapter. If working in the VAX/VMS environment, the user may spawn a subprocess from any level of the program. This allows the user to temporarily halt execution, do any number of other tasks in the spawned process, and then return by logging out of the spawned process. **FASTQ** will be at the same level with the database intact when the spawned process is terminated.

Easy access and use of the utilities provided depends on a basic understanding of the interface and of the overall design of the program. The interface and overall design of **FASTQ**, as well as parameters for batch use and specific machine dependent run-line parameters are described in this chapter.

2.1 LAYERED COMMAND STRUCTURE

FASTQ interacts with the user by means of a layered command structure. This structure employs levels of acceptable responses from the user. Within each level the set of correct responses varies to accomodate the utilities to be performed. A correct response either initiates an action[4] or indicates a movement from one level to another. This movement must be systematic in that only hierarchical moves are allowed. The user may go up to the higher level from which the current level was chosen, or down to a new lower level with its own allowable options.

The program structure places general tasks at higher levels than more specific tasks. For instance, the task of geometric data input is higher than toggling on the node numbering for mesh plotting. This structure dictates that the user respond first to choose the general task and then choose again to indicate the specifics of that task. As an example, if the user wished to type in new nodal boundary flags for a given line, the main option *KEYIN* would first be chosen to indicate that input was

[4]Actions include the plotting of geometric data, the writing of a mesh output file, etc.

to be received from the keyboard. Next from the *KEYIN* level, the *BOUNDARY* option would be chosen to indicate that boundary flag(s) are to be entered. And finally, the *LINE* option of the *BOUNDARY* level would be chosen to indicate that nodal boundary flags were to be entered along specific line(s). The prompt would then indicate the proper format for entering the data, and the user could begin input of the line boundary flag information.

To aid in using and understanding the structure of **FASTQ**, a flow chart has been prepared which diagrammatically displays all the options and suboptions available. This is shown in Figure 1. The beginning or intermittent user should find reference to the flowchart helpful during program execution. The user always enters the program at the top level of the command structure and encounters the prompt:

ENTER OPTION:

The available options at the top level are shown at the left of Figure 1. Some of these options, such as *WRITE* and *READ*, perform functions directly. That is, when chosen, they execute a procedure rather than present a new set of suboptions. Others, such as *LIST*, *MESH*, and *KEYIN*, contain suboptions, and the layering in some instances extends to a third level. As each option or suboption is chosen, the prompt is modified to show the user where he is. If, for example, the user responds to the initial prompt with:

ENTER OPTION: M

the *MESH* option will be chosen, and the prompt becomes:

ENTER MESH OPTION:

Again, the available *MESH* options are shown in Figure 1. If next, the user responds with:

ENTER MESH OPTION: G

the *MESH GRAPHICS* option will be designated, and the prompt becomes:

ENTER MESH GRAPHICS OPTION:

The available *MESH GRAPHICS* options are also shown in Figure 1. Ascending from a lower back to a higher level is accomplished by hitting the return key. In this case, two return keys will return the user to the prompt:

ENTER OPTION:

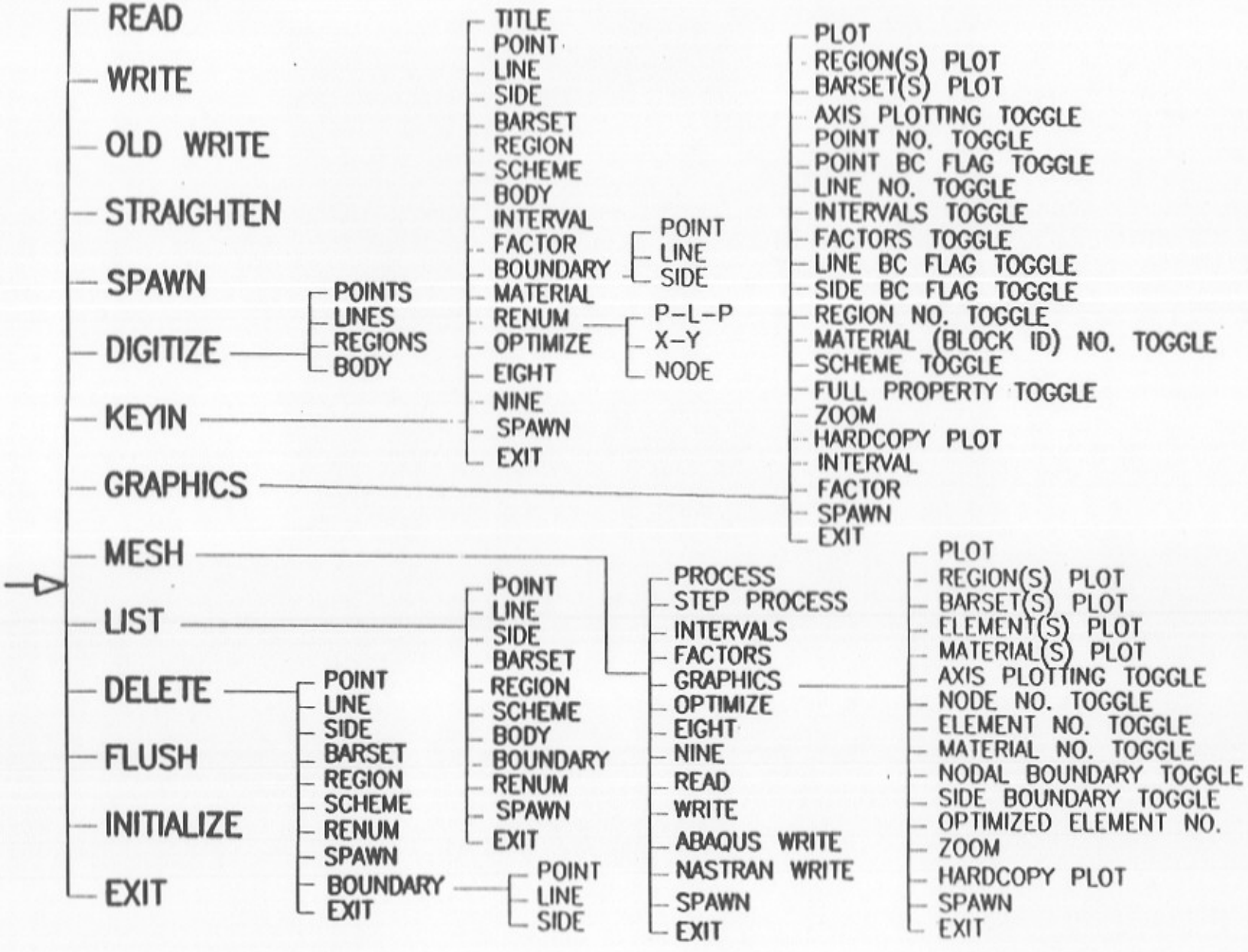


Figure 1. FASTQ FLOWCHART

The user chooses an option by spelling enough of the option to distinguish it from other choices being offered, and then hitting return. This usually requires one letter of the option, and at most requires three letters. When the user does not specify enough of the option to avoid a conflict[5] the program defaults to a prioritized option. For instance, in the *MESH* option, the *EIGHT* and *EXIT* suboption begin with the letter *E*, but **FASTQ** defaults to the *EXIT* suboption if only an *E* is specified by the user. The appropriate response at each level for any option can be shown with the online help facility as described in Section 2.2. The location of the asterisk in this help message also shows which of the options will win in a conflict situation.

FASTQ allows any series of commands to be included in the response at any level. This alleviates multiple prompt/response cycles for the user familiar with the program structure. For instance, if the user has read in the geometry and wishes to plot the geometry with line intervals and region schemes displayed, he may respond to the main prompt with:

ENTER OPTION: G,I,SC,P

The G chooses the *GRAPHICS* option, the I toggles intervals on, the SC toggles schemes on, and the P plots the geometry. The user remains in the level of the last command. In this case the current level is shown by the prompt:

ENTER GRAPHICS OPTION:

If the geometry appears acceptable and the user next wishes to mesh the problem and display the results, he may respond with:

ENTER GRAPHICS OPTION: ,M,P,G,P

The initial comma, or blank field, returns the user to the main option, the M indicates the *MESH* option, the P indicates automatic processing, the G indicates that the *MESH GRAPHICS* option should be chosen, and the P plots the resulting mesh. Care should be taken with this capability, as a bad choice in a string of commands could produce quite unexpected behavior. The destructive choices such as *EXIT* or *FLUSH* are protected with confirmation inquiries where possible to help the user avoid drastic mistakes.

2.2 INTERNAL HELP FACILITY

If the user is unfamiliar with the options available at any level, typing HELP or HE will produce a brief description of the options available. Also, if the user types

[5] If several options at a given level begin with the same letter, the response is ambiguous, and a conflict will result.

an incorrect option, the help message will be displayed. The message contains brief descriptions of each option. It also indicates how much of the option must be typed for recognition, and which option will be chosen if the response is ambiguous. This is done by the insertion of an asterisk (*) after the last letter of the option needed to distinguish it from other options. For example, the *LIST* option help message is:

THE FOLLOWING LIST OPTIONS ARE AVAILABLE:

*P*OINT*	*= LISTS POINT DATA*
*L*INE*	*= LISTS LINE DATA*
*SI*DE*	*= LISTS SIDE DATA*
*BA*R SETS*	*= LISTS BAR SET DATA*
*R*EGION*	*= LISTS REGION DATA*
*SC*HEME*	*= LISTS SCHEME DATA*
*BOD*Y*	*= LISTS REGIONS IN THE BODY*
*B*OUNDARY*	*= LISTS BOUNDARY FLAGS*
*REN*UM*	*= LISTS RENUM CARDS*
*EI*GHT*	*= LISTS 8 NODE GENERATION TOGGLE*
*N*INE*	*= LISTS 9 NODE GENERATION TOGGLE*
*S*PAWN*	*= SPAWNS A SUBPROCESS*
*E*XIT*	*= EXITS FASTQ*
	(CARRIAGE RETURN TO EXIT)

In this example, a response of B would produce a listing of the boundary flags, a BA would enable listing of barsets, and a response of BOD is required for a listing of all regions and barsets in a body. If more extensive information about a specific option is desired the user must refer to the appropriate section of this manual.

2.3 RUN-LINE PARAMETERS / BATCH MODE OPERATION

FASTQ uses run-line parameters to indicate specific machine dependent modes of operation[6] and specific device linkages where needed. Since these parameters are machine dependent, all currently supported operating environments and modes will be discussed. The use of several site independent routines which have been made available as part of the Department 1520 **SUPES** software package[8] has enabled the generation of a truly site and run mode independent version of code. The run-line parameters are used by command files which set the appropriate process parameters for the site dependent features.

[6] Batch mode operation versus interactive mode.

2.3.1 VAX/VMS Interactive and Terminal Batch Mode

Running interactively or in terminal batch mode on the VAX is initiated by simply typing FASTQ and optional run-time parameters in the following format:

```
$ FASTQ [FASTQ_file.ext] [terminal device] [hardcopy device] [output_file.ext]
```

If none of the optional parameters are supplied, the user will be prompted for the terminal and hardcopy device, and **FASTQ** will start with no initial data. If an input file is indicated it should be a FASTQ file containing the geometric definitions for the problem as explained in Appendix A. The default extension is .FSQ if no extension is specified. If no data file is to be included, and the user wishes to input the device information as part of the run-line command, a place holding set of double quotes ("") should be included for the input file specification.

If an output file is specified, **FASTQ** will run in terminal batch mode. In this mode, **FASTQ** runs as if in batch mode, reading the input file, meshing the geometry, and outputting the mesh in the **GENESIS**[3] format to the file specified. The printed output is sent to the screen, but the user has no option of interacting with the program. Since no plotting is performed in this mode, the device driver specifications are meaningless. A set of double quotes ("") may be used as place holders for each of the devices.

2.3.2 VAX/VMS Batch Mode

Batch mode submits the job to the batch queue. When invoked, the program will read the FASTQ file, process the mesh, and write a GENESIS output file. Printed program output is written to the log file. Batch mode is invoked by submitting a batch job using the submit command or other utilities such as the 1520 BATCH utility. Run-line parameters to indicate the FASTQ file name and the GENESIS output file name are optional and can be invoked in the following formats:

```
$ BATCH FASTQ [FASTQ_file.ext] "" "" [GENESIS_file.ext]
```

```
$ SUBMIT/NOTIFY FASTQ [FASTQ_file.ext] "" "" [GENESIS_file.ext]
```

The double quotes ("") are used as place holders for the device drivers. No plotting is provide in batch mode, and thus the device drivers are meaningless.

2.3.3 CRAY/CTSS - Batch and Interactive Mode

Running interactively on CTSS is somewhat different than other environments. When the procedure is first run, it will link an executable version and leave the files fastqx, fastqbin, stkfnt, romfnt, and ssrfnt in the user's area for use when the procedure is rerun. Terminal batch or interactive mode is based on the parameters supplied. True batch mode must be activated using the PROD utility. The format is as follows:

```
fastq [dev= ] [i= ] [o= ] [m= ]
```

Each of these parameters are defined as follows:

dev Any valid graphics device supported by the Sandia Virtual Device Interface. The default is ls5. When this keyword is omitted, and an executable of **FASTQ** does not reside in your area, the ls5 device driver is used. If an executable resides in your area when this keyword is included, the old executable will be deleted, and a new version will be linked with the new device. This parameter has no meaning in terminal batch or truly batch mode.

i The name of the FASTQ file containing the input data. The default is tape1. When this keyword is omitted and the user is running from the terminal, **FASTQ** runs in truly interactive mode. If this keyword is present and the user is running from the terminal then **FASTQ** runs in terminal batch mode. When submitting the job with PROD for a true batch run, the program attempts to read tape1 if no other file is specified.

o The name of the file for the printed output from **FASTQ**. In interactive mode this parameter has no meaning. In terminal batch mode, the default is the screen. In true batch mode the default is tape6. This file will not be overwritten if it already exists.

m The name of the binary GENESIS database file. In interactive mode this parameter has no meaning. In terminal or true batch mode, the default is tape9. This file will not be overwritten if it already exists.

Note that any interaction with the CTSS environment must be done with lower case letters. However, once **FASTQ** execution has begun, no such restriction is enforced. Upper case letters can be used interchangeably with lower case at will. The use of upper case letters is advantageous since titles can thus be represented in upper case letters and FASTQ files from other systems need not be converted to lower case to be used on CTSS.

It is also important to understand that the CTSS operating system is not amenable to interactive use, and thus long waits should be expected between each prompt/response cycle. No typing ahead is permitted on CTSS and doing so will result in unpredictable execution results.[7] The user should always wait until the input prompt appears before attempting a response. This problem can be alleviated somewhat by using the multiple response capability of **FASTQ**. If a series of responses can be planned for, the entire sequence can be input as one response as explained in Section 2.1. In this manner multiple prompt/response cycles can be reduced to one, and several wait cycles avoided.

[7] A program bomb is one such possibility.

3. PROBLEM DEFINITION

The problem to be meshed is defined in **FASTQ** by using a number of *wire frame*[8] type entities and problem description parameters.[9] These entities are defined and related in a manner appropriate to describe the boundaries of regions to be meshed with quadrilateral elements, or the outlines to be meshed with bar elements. The fineness and gradation of the mesh is also controlled by these problem definitions.

This task is a critical step toward the generation of a good mesh and is potentially very time consuming. Thus a number of utilities have been provided to aid in this process. This chapter first defines all of the necessary problem parameters and geometric entities, and then describes the three methods[10] of entering these definitions into the database. Geometric data manipulation is then covered as well as the graphical verification process.

3.1 GEOMETRY AND PROBLEM DEFINITIONS

The items, or entities, used to define the problem and the geometry to be meshed are explained in this section. The necessary information to define each item is described. However, discussion of the actual format used for input into the FASTQ file is reserved for Appendix A of this manual where the FASTQ data file is described in detail. Likewise, the responses for item definition during an interactive input session are reserved for Section 3.2.2.

3.1.1 Title

The title allows an identifying string to be attached to the file and subsequently to the generated mesh. This string can be up to 72 characters long.

3.1.2 Points

Points are generally used to indicate actual locations within the geometry. Each point is identified by a point number and an x, y or R, θ coordinate pair of values.

[8]The wire frame definitions in 2-D represent a complete, unambiguous geometric definition. The same in not true for 3-D geometries.

[9]The boundary flags and problem title are some such parameters.

[10]The three methods are the FASTQ file, *KEYIN*, and *DIGITIZE*.

All lines employ points as either the line terminators, or line parameter specifiers. When used as parameter specifiers, the point may actually not refer to a geometric location but instead merely store some needed value as its x or y coordinate as is the case with the CIRR line described in Section 3.1.3.

It must be understood that points and nodes are not the same entity. A node is associated with the generated finite element mesh coordinates, whereas a point deals only with the defining geometry. Because a node is always generated at the end points of a line when the mesh is generated, nodes and points may share the same coordinates. Also, boundary flags can be attached to the nodes generated at points through the point boundary flag described in Section 3.1.9.

3.1.3 Lines

Lines are used to define the boundary of geometric regions. Lines may be straight, portions of circular arcs, portions of logarithmic spirals, or portions of parabolic arcs. A line definition includes each of the following items:

Line Number Identifies the line.

Line Type The type of line to be generated. Available line types are as follows:

- **STR** A straight line.
- **CIRC** A circular arc (or logarithmic spiral) about a center point.
- **CIRM** A circular arc with the third point on the arc.
- **CIRR** A circular arc with the radius given.
- **PARA** The tip of a parabolic arc.
- **CORN** A corner formed by two line segments. The first segment is from the first point to the third point, and the second segment is from the third point to the second point.

First Point Indicates a beginning point number for the line.

Second Point Indicates an ending point number for the line.

Third Point A third point needed to complete the line definition of types other than straight lines. This point is discussed in detail below.

Intervals The number of intervals or subdivisions of this line for mesh generation. These intervals are evenly spaced along the line unless a factor other than 1.0 is specified.

Factor The ratio of succeeding interval lengths along the line starting from the first point and progressing toward the second point. For instance, if a factor of 2.0 is specified, each interval along the line will be 2 times longer than the preceding interval. The default is 1.0.

The meaning of the third point for the six different line types available is as follows:

STR The third point has no meaning for a straight line definition.

CIRC The third point is the center point for the circular arc or logarithmic spiral. If the third point number has been entered as a positive integer, an arc is defined counterclockwise about this point from the first point to the second point. If the third point number has been entered as a negative integer, the arc is reversed to a clockwise one from the first point to the second point about the third point. Note that switching first and second points has the same effect as changing sign on the third point number. The actual arc is calculated as a logarithmic spiral that forms the degenerate case of a perfect circular arc if the distance from the first point to the third point and from the second point to the third point is equal. This representation allows for arcs in which the radius can change as a function of the angle traversed.

CIRM The third point, as the line type definition indicates, is another point on a circular arc connecting the two end points. A circular arc can always be drawn through three points unless the three points are colinear. In this case the line is generated as a straight line.

CIRR The third point is used to specify the radius of the arc to be generated. The first coordinate (x value) of the third point will be used as the radius of the arc to be generated. The second coordinate (y value) of the point is irrelevant. If the third point number is given as a positive integer, the center point is assumed to lie on the left of a straight line drawn from the beginning point to the ending point. If the third point number is a negative

integer, the center is assumed to be on the right of such a line. In either case, the arc that is generated and the center of the arc are always on opposite sides of this line. Thus only arcs less than 180 degrees can be generated with this definition.

PARA The third point is the peak of the parabola with the beginning and ending points equal arc lengths away on either side of the third point. In other words, to generate a parabolic arc the three points must form an isosceles triangle. This definition limits the user to generation of the tip of parabolic shapes only.

CORN The third point must be the corner point of the line as explained in the definition above. The only advantage of this line over two straight lines is that the number of intervals for the two lines combined can be controlled with one interval setting.

3.1.4 Sides

Sides are groups or sets of lines. The lines contained in a side may or may not be continuous portions of the geometry depending on the use of the side. Sides can be used for three purposes:

1. **Forced Rectangle Primitive.** If the user wishes to override the default initial mesh generation for rectangular primitives as explained in Section 4.3.2, he/she may force certain lines to form logical sides for mapping. These lines must form a continuous segment of the boundary. Forcing logical sides is accomplished by placing all the lines to be included on the first logical side of the region as one geometric side definition. Then, if the regions specified scheme has no initial primitive indicators (see Section 4.3.2,) the mesh generator will attempt to locate the first two corner nodes at the end points of this geometric side. Once these corners have been forced, the choice of nodes for the other two logical corners for mapping are committed based on the remaining number of nodes in the perimeter. The forcing of corner choices is almost always unnecessary and in some cases severely limits the capacity of the meshing algorithms to generate a good mesh.

2. **Reduce Input Complexity.** If the region to be defined contains a large number of lines, lines may be grouped together to reduce the size of the region card. These lines must also form a continuous segment of the boundary. If this is the reason for using sides, including the appropriate primitive region scheme for such a region allows the mesh generator to position corner nodes

where **FASTQ** determines they are necessary for the best resulting mesh. This includes locations within the center of a geometric side (see Section 4.3.2.)

3. **Define Sets of Lines.** A side may be used to group lines into a set for ease in specifying common intervals, common factors, or common boundary flags (see Section 3.1.10, 3.1.11,and Appendix A.) The lines included in a side for this application need not be a continuous portion of a boundary.

A side definition includes each of the following items:

Side Number	Identifies the side.
Line Number 1	The first line of the side.
	⋮
Line Number n	The n'th line of the side.

3.1.5 Regions

Regions are the basis for all quadrilateral mesh generation performed by **FASTQ**. A region is simply a list of sequential lines and/or sides which enclose an area to be meshed. Dividing or decomposing an arbitrary geometry into regions appropriate for meshing is largely an art. Primitive regions, or regions which can be nicely meshed by **FASTQ** are discussed in Section 4.3.2. A specific meshing instruction or *scheme* is associated with each region as explained in the scheme definition below. A block ID number is also associated with each region. The block ID allows for specific material properties and/or body forces to be associated with each region of the geometry. This region block ID can be shared by multiple regions, but it cannot match any barset block ID since the element types are different.

In the FASTQ file, all side numbers in the region list are given as a positive integer, while line numbers are specified as their respective line number negated. A region definition includes each of the following items:

Region Number	Identifies the region.
Block ID Number	The block ID to be given to all elements in the region.
Side/Line Number 1	The first side or line of the region.
	⋮
Side/Line Number n	The n'th side or line of the region.

3.1.6 Barsets

Barsets are the basis for all two and three node element generation performed by **FASTQ**. A barset is simply a list of sequential lines along which bar elements are to be generated. A block id number is associated with each barset to allow for different material properties, inertial properties, or body forces to be associated with each barset within the geometry. The barset block id can be shared with other barsets but cannot match any region block id since the elements types are different.

An inside point may be included in the definition to force a specific element connectivity or *inside* and *outside* aspect to each element. This is particularly needed when applying pressure loads to an axisymmetric shell element. If an inside point is specified, the element is numbered such that the first point is to the right of a vector drawn from the inside point to the midpoint of the element. The second point is conversely to the left of that vector. If no inside point is specified, connectivity progresses from the beginning point to the ending point of each line in the barset.

A barset definition includes each of the following items:

Barset Number	Identifies the barset.
Block ID Number	The block ID to be given to all elements in the barset.
Inside Point	A point on the inside of all lines in the barset.
Line Number 1	The first line of the barset.
	⋮
Line Number n	The n'th line of the barset.

3.1.7 Schemes

Schemes are used to specify a specific meshing algorithm as well as various mesh adjustment procedures for quadrilateral mesh generation within regions. A scheme can be applied to an individual region or can be input as a default for all regions not otherwise specified. The details of all possible scheme symbols and their meanings are discussed in Section 4.3.2 of this manual. A scheme definition includes each of the following items:

Applicable Region	The region number for which this scheme applies.[11]
Scheme	A string of symbols used to define the scheme.

[11] A 0 for the applicable region number inputs a new default scheme.

3.1.8 Body

The body defines the regions and barsets to be included in the mesh. This definition is handy if several possible geometric configurations are to be used in the problem. All the necessary data may be included in the file and then only the data required for a specific problem as specified in the body card will be used to generate the mesh. For instance, if the effect of an added flange on a given geometry is being investigated, the total geometry can be included in the FASTQ file. Then, only the body card must be changed for the generation of a mesh with or without the flange regions. There is no limit to the number of regions and barsets allowed in one body. All regions in a body definition are indicated by specifying the region number as a positive integer. Barsets are included by negating the barset number. A body definition includes each of the following items:

Region/Barset Number 1 The first region or barset to be included in the body.

⋮

Region/Barset Number n The n'th region or barset to be included in the body.

3.1.9 Point Boundary Flags

Point Boundary Flags are used to attach boundary conditions to finite element nodes that are generated at point locations. Since all line end points will align with a generated mesh node, any line end point may be flagged. The flag can be used to reference the node for restraint or loading of some type in the analysis. Multiple flags can be attached to the same point number, and thus the same node, if desired. A Point Boundary Flag definition includes the following items:

Point Boundary Flag The integer flag number.

Point Number 1 The first point to be flagged with this flag.

⋮

Point Number n The n'th point to be flagged with this flag.

3.1.10 Line Boundary Flags

Line Boundary Flags are used to attach boundary conditions to finite element nodes that are generated along specific lines in the geometry. The flag can be used to reference nodes for restraint or loading in the analysis. Multiple flags can be attached to the same line number if desired. The definition may contain lines and geometric sides, although the geometric sides are used as merely a convenient way of entering multiple lines. If a side is used in the definition it must be negated. A Line Boundary Flag definition includes the following items:

Line Boundary Flag The integer flag number.

Line/Side Number 1 The first line or side to be flagged with this flag.

⋮

Line/Side Number n The n'th line or side to be flagged with this flag.

3.1.11 Side Boundary Flags

Side Boundary Flags are used to attach boundary conditions to finite element sides which are generated along specific lines in the geometry. The flag can be used to reference element sides for attaching loadings[12] or for attaching boundary conditions[13] in the analysis. Multiple flags can be attached to the same line number if desired. It should be noted that the Side Boundary Flag is named to indicate flagging of element sides and not geometric sides as defined in Section 3.1.4. The definition may contain lines and geometric sides, although the geometric sides are used as merely a convenient way of entering multiple lines. If a side is used in the definition it must be negated. A Side Boundary Flag definition includes each of the following items:

Side Boundary Flag The integer flag number.

Line/Side Number 1 The first line or side to be flagged with this flag.

⋮

Line/Side Number n The n'th line or side to be flagged with this flag.

Remember, a Line Boundary Flag attaches flags to nodes, and a Side Boundary Flag attaches flags to element sides.

[12]Pressure loadings may be attached as element side loads.

[13]Slide line definitions are easily handled as element side boundary conditions.

3.2 GEOMETRIC DATA INPUT

There are three available methods of inputting a problem's geometry and meshing parameters into the **FASTQ** database.

1. **FASTQ File.** Data may be input into **FASTQ** by reading a previously generated FASTQ file. This file may have been written from a previous **FASTQ** session, or it may be created from a text editor. Merging of several files is possible.
2. **Keyboard Input.** Data can be input from the keyboard using the *KEYIN* option.
3. **Drawing Digitization.** Data can be digitized from a scaled drawing of the geometry using a digitizing pad and cursor in the *DIGITIZE* option.

Since each method has advantages and disadvantages, it is often expedient to combine the three input techniques when defining the problem geometry.

3.2.1 FASTQ File Input and File Merging

Geometric data is input from a file during a read operation in both interactive and batch mode operations as explained in Section 2.3. The FASTQ file is a free formatted ASCII file which can be created with a write operation in **FASTQ** or with any text editor. The fact that a user's favorite text editor can create the FASTQ file allows considerable flexibility. The *SPAWN* option explained at the beginning of Chapter 2 is often useful in combination with text editing. The *SPAWN* option allows the user to exit **FASTQ** temporarily, use the text editor to make modifications in the FASTQ file, reenter **FASTQ** at the same level, and then input the new information into **FASTQ** with a *READ*. When running interactively, the *READ* option will ask the user for a file name. The user has three chances to respond with an existing file. Three failures at opening the file specified will return the user to the main level of **FASTQ**. Thus, if this option is chosen inadvertently, three returns will undo the mistake with no loss of existing data.

Acceptable data delimiters in the FASTQ file include a space, a comma or an equal sign. The 1520 Free Field Reader routines are employed as fully explained in [8]. Some important advantages of using these routines include the free formatting capability and multiple data delimiters as explained above, but also the continuation card feature. If the amount of data being used to define a geometric entity, for instance the lines in a region, exceed the space available on a line, an asterisk (*) at

the end of the line indicates that the next line should be treated as a continuation of the current line. Comments may be included in the file with the use of a dollar sign ($). All data after the dollar sign on a line are ignored. Because **FASTQ** ignores comments in a file, the comments cannot be replaced when saving the file with a **FASTQ** *WRITE* operation.

As each line or card is read from the file it is interpreted and stored. Any data errors are reflected to the screen (or the output device in batch mode) as they are located. For actual mesh generation, data compatibility must be checked to see that all lines have existing and valid point numbers, all regions have existing and valid line and/or side numbers, etc. However, data compatibility is not checked when the file is read since it is assumed that the user may wish to only define portions of the data with any given file. This allows the user to write out and read in the geometry defined up to some point in the problem without being required to complete all definitions before doing so. This also allows the user to sequence the data file in any order. All point definitions need not come before all line definitions, etc. The only exception is that interval and factor cards (see Appendix A) must appear after the definitions of the lines or sides being used since intervals and factors cannot be assigned to nonexistent items. When the file is written with a **FASTQ** *WRITE*, the interval and factor cards are not replaced, as the information is stored on the specific line cards.

Reading of the file terminates when an EXIT card is found. If an EXIT card is not found in the deck, **FASTQ** accepts the data input but signals to the user that it terminated input only when there were no more lines in the file to read. Once the file has been read, a summary of the number of new definitions input from the file is printed.

FASTQ has the ability to merge two existing FASTQ geometries into one database. This is accomplished by first defining one geometry in **FASTQ** using any of the input options. Once the first set of data is in place, a read option can initiate a merge of other data stored in a FASTQ file. If the database is not empty when a read option is chosen, which is the case in this situation, the user is asked if data merging is desired. Responding affirmatively will initiate the merging.

A merge may require a renumbering of some of the identification numbers of points, lines, etc. If the item identification number being added to the database during a merge coincides with an existing item, the item being added will be given a new unused identification number. The user will be informed of all renumbering by a message to the screen indicating the change. Any references to the old numbering in the data being merged will be replaced with the new numbering in the final combined database. Aside from this particular instance, **FASTQ** performs no renumbering of

the geometric data. After a merge has been completed, a summary of the number of new items added as well as the total number of items currently in the database is printed.

The format of each item or entity in the FASTQ file is documented in Appendix A of this manual.

3.2.2 Keyboard Input

All geometric data can be defined interactively by typing in appropriate values at the keyboard. This input method is largely contained within the *KEYIN* option of the main menu in **FASTQ**. As in user file preparation, keyboard input requires that all point coordinates, the connectivity between all lines, sides, and regions, and the specific numbering scheme to be used are defined explicitly by the user. Although somewhat tedious when compared to digitization, the user can with the *KEYIN* option insure that point coordinates are as exact as he wishes, and that item numbering will be handled as he prescribes.

Since each geometric entity requires varying amounts and types of input, a prompt showing the amount and sequence of the needed information is supplied during keyboard input. For instance, if the *SIDE* suboption of the *KEYIN* option is chosen, the following prompt is given:

ENTER SIDE DATA IN THE FOLLOWING FORMAT:
[SIDE NO.,LINE 1,LINE 2,...,LINE N]
HIT RETURN TO END INPUT
>

The response to the prompt is parsed using the 1520 Free Field Reader, and thus acceptable delimiters include the space, the comma, or the equal sign. A typical response to the above prompt could be:

> 1,4,2,8,204

which specifies that Side 1 consists of Lines 4, 2, 8, and 204. **FASTQ** then waits for the next side to be specified. An extra return following the last item to be defined ends that sequence of input. Thus all items of the same type may be entered sequentially without returning to the menu between entries.

All of the data fields of a given item need not be specified. For instance, when entering a point, leaving a blank field[14] for the x value or leaving off the y value will

[14] A blank field may be two consecutive commas (,,).

default the respective value to zero. Likewise, for lines, only the beginning, ending, and center point (if requested) must be explicitly specified. Intervals and factors are defaulted to 0 and 1.0 respectively. If invalid or insufficient information is given, **FASTQ** does not attempt to input the bad data, informs the user of the problem, and waits for further input.

If corrections to existing data are desired, redefinition of the data item using *KEYIN* will replace the old definition with the updated version. Also many entity properties can be added or changed in *KEYIN* without changing the entire definition. These properties include:

Line Intervals Requires the line number (or negative side number) and the new number of intervals.

Line Factors Requires the line number (or negative side number) and the new factor.

Region Block ID Requires the region number and its new block ID.

Boundary Flags Requires the boundary flag and the new entity numbers to be added to its association list.

Because line intervals and line factors are often changed during the process of interactive mesh generation and geometric data validation, these values may also be entered in the *GRAPHICS* or the *MESH* option of the main menu of **FASTQ**. Intervals and factors may be assigned to all lines within a side by specifying a negative side number. This allows the user to maintain consistency for the set of lines in the side. It is useful for problems involving many layered regions where it is desired that the boundaries between layers always have the same number of intervals and consistent factors.

As with file reading, data compatibility is not checked during *KEYIN* input but only when mesh generation is initiated. Thus the order in which the data is input is not crucial. Regions can be input before lines, etc. The only exceptions to this is that intervals, factors, and material numbers cannot be assigned to nonexisting entities.

3.2.3 Digitizer Input

Often the quickest and most direct method of defining the geometry to be meshed is the use of a digitizing pad and a scaled drawing of the object to be modeled.

The *DIGITIZE* option of the main **FASTQ** menu is used for this type of input. Digitization has a number of attractive benefits.

1. **Fewer Measurements.** On any scaled drawing of the geometry, the coordinates of only two points must be explicitly defined. From this initialization all coordinates of subsequently entered points are calculated by **FASTQ** using the mouse position relative to those two points. This greatly reduces the amount of information the user must calculate or measure prior to inputting the geometry.

2. **Automatic Numbering.** No item numbering or item connectivity is required from the user. The digitizing routine keeps track of all numbering considerations internally. This eliminates the need to define which points are used to form which lines, etc.

3. **Combinined Input Utility.** This digitization process can be used in conjunction with other input methods to speed the total problem definition when accuracy is of concern, or even when a scaled drawing is not available. For instance, once the points have been input, the mouse could be used to rapidly generate the lines and subsequently define regions.

The digitization routines do not handle some types of data input. This includes the title, line intervals, region block IDs, schemes, and side definitions. These must be entered at the keyboard or by editing the FASTQ file. Also digitizable line types are limited to straight lines and arcs with the center given. However, since historically these two lines types have been essentially the only types used, this should not be a great hindrance.

The digitizer accuracy may or may not be a problem depending on the requirements of the user and the size of the drawing to be digitized. In general, the larger the drawing, the more accurately it can be digitized. Straight lines will always remain straight, but may be slightly askew. The errors are generally not significant, but may be a problem in axisymmetric geometries where the center of the object must not deviate from the centerline. The *STRAIGHTEN* option described in Section 3.3.4 can be used to correct this. Circular arcs can rarely be input as true circles due to the slight inaccuracies of the digitizing tablet, but exact arcs are not needed for mesh generation to proceed and the inaccuracies introduced are usually insignificant enough that they do not effect analysis results.

Initialization of the Digitizing Tablet

When entering the *DIGITIZE* option, **FASTQ** requests the coordinates of the two points to be used to initialize the drawing coordinates to the tablet coordinates

with the prompts:

USER COORDINATE SYSTEM MUST NOW BE SET UP.
DO NOT MOVE THE DRAWING UNTIL FINISHED
TYPE IN THE COORDINATES OF THE LOWER LEFT POINT:

>

and:

TYPE IN THE COORDINATES OF THE UPPER RIGHT POINT:
>

A x, y or R, θ coordinate pair must be entered for each prompt. **FASTQ** defaults any coordinates not specified to be zero. The lower left point should be the lower left most portion of the drawing to be plotted on the terminal screen. It need not be the lower left most portion of the geometry. The same is true of the upper right point, in that it only designates the plotting limits, and not the limits of the geometry.

FASTQ next initializes the drawing by requesting that the two points entered be digitized with the mouse, as the prompt indicates:

NOW DIGITIZE THOSE TWO POINTS:
PUSH BUTTON "1" ON MOUSE FOR LOWER LEFT
PUSH BUTTON "2" ON MOUSE FOR UPPER RIGHT
PUSH BUTTON "E" ON MOUSE TO END

Placing the mouse cursor over the lower left location and pushing the *1* button on the mouse will initialize the lower left location. Likewise a *2* button choice initializes the upper right location. To end initialization, the *E* button must be pushed. If the mouse input has been recognized by the program, **FASTQ** will notify the user with one of the following messages:

POINT 1 INPUT
POINT 2 INPUT
INITIALIZATION COMPLETE

If the cursor happens to slip, and a bad digitization has possibly occurred when entering the lower left or upper right points, the user simply pushes the appropriate button again at the desired location. Digitizing order is not important, and the latest

digitized location for each point will be used when the *E* button ends initialization. The user will then be presented a brief overview of the button definitions and will be queried as to readiness to begin. This overview of the buttons remains on the alpha screen of the Lear Seigler terminal although not visible because the alpha screen is initially turned off for digitization. The overview can be shown during digitization by simply turning the alpha screen back on again.[15]

The screen display limits are next set to the initialization limits, and any data currently in the database within the display limits will be drawn on the screen. Initialization limits the range of data that will be plotted on the screen, but does not limit the range of data that may be digitized. Data points outside the initialization points may be digitized as long as the mouse crosshairs are over a sensitive area of the tablet.

Once a drawing has been thus initialized to the digitizing tablet, the user may go in and out of the digitization option at will without reinitialization as long as the drawing is not moved on the board. If the drawing must be moved to capture more of the geometry, the user must choose the *INITIALIZATION* option of **FASTQ**. This option erases the old initialization of the drawing, and the user will be prompted for new initialization points upon subsequently entering the *DIGITIZE* option.

Digitizer Mouse Button Definitions

The digitization routine in **FASTQ** is currently based on a DIGIPAD digitizing tablet with a 12 button mouse. The available buttons include the numbers *0* through *9* and the letters *A* through *E*. Various sequences have been defined using these buttons to enable geometric data input. The following section describes the mouse options, and is designed as a complete reference. However, familiarity with the digitizing tool is best gained through actual experience with the hardware. A brief description of each button definition is included in Appendix B as a quick reference guide.

Each mouse button sequence can be classified as a *base button* with an optional *prefix* button. For instance the *1* button could be classified as the *enter point* base button, *2* as the *enter straight line* base button, etc. There are only two prefix buttons, *0* and *C*. The *0* prefix button indicates that you wish to reference an existing point rather than create a new one at that location. This button is essential for joining regions to existing data, for closing a region back to an existing point, and for using the digitizer to connect points previously entered. The *C* prefix button has an opposite effect in that the closest point is always found, but a new point is entered

[15]The Lear Seigler alpha screen is toggled on and off by the *Function 1* key sequence.

using the closest points coordinates.[16] This is necessary for the generation of slide lines in the mesh when two separate mesh surfaces are needed at the same geometric location.

The following is a complete explanation of each defined mouse button sequence.

1 **Enter Point.** *1* enters the current x,y location as a new point, and assigns the next sequential point number to that point. The point location is displayed on the screen with an *X*. This point number is then stored as the *last point* for further reference.

01 **Get Closest Point.** *01* searches the database for the closest point to the current x,y location of the mouse, and enters that point as the *last point* for further reference. No new point is entered into the data base or displayed. This allows reference to existing points for line and arc generation and for region closures.

C1 **New Point At Closest Point.** *C1* finds the closest point just as *01* does, but it adds a new point to the data with the same coordinates as the closest point. This new point is stored as the *last point*. This option is used for defining slide line endpoints.

2 **Straight Line To New Point.** *2* enters the current x, y location as a new point, with a new number, just as *1* does, but also enters a straight line into the database drawn from the *last point* to this *new point*. This line is also automatically numbered and displayed on the screen. The *new point* is then stored as the *last point* for further reference. If no *last point* has been stored, no line is input.

02 **Straight Line To Closest Point.** *02* searches the database as *01* does to find the closest point. This point is used as the *new point* for straight line generation as in *2*. If the closest point found is the *last point*, no line is generated. This option allows for closing the region, or for connecting existing points into lines.

C2 **Straight Line To New Point Over Closest Point.** *C2* enters a new point with the closest point's coordinates as *C1* does, and then enters a new line as in *2*. Again this is used for slide line definitions.

3 **Ccw Arc To New Point.** *3* enters the current x, y location as the *new point* and designates a counterclockwise circular arc to be drawn from

[16] This new point is essentially on top of the closest point

the *last point* to this *new point* about a *center point* to be entered next. Upon entering *3*, a *center point* must be entered for the arc definition to be completed. The center may be entered with either a *1* to designate a new point or a *01* to refer to the closest existing point. This *center point* will be input into the data base without regard to it being a true center or not. **FASTQ** uses a spiral representation to smooth the arc between the *new point* and the *last point* about the center given. After the arc has been drawn, the *new point* (not the *center point*) is stored as the *last point* for further reference. Thus, the line input sequence can continue without re-entering the arc endpoint with a *01*.

03 **Ccw Arc To Closest Point.** *03* finds the closest point to the current location, and uses it as the arc endpoint, or *new point*. No additional points are added to the database, and arc generation continues as in *3*. If the closest point found is the *last point*, no arc is generated.

C3 **Ccw Arc To New Point Over Closest Point.** *C3* enters a new point with the closest point's coordinates, as in *C1*, and draws a counterclockwise arc as in *3*. Again this is used for slideline generation.

4 **Cw Arc To New Point.** *4* designates a clockwise arc as opposed to the counterclockwise arc in *3*.

04 **Cw Arc To Closest Point.** *04* designates a clockwise arc as opposed to the counterclockwise arc in *03*.

C4 **Cw Arc To New Point Over Closest Point.** *C4* designates a clockwise arc as opposed to the counterclockwise arc in *C3*.

5 **Show Current Location.** *5* draws a plus sign + on the screen to show the current location of the mouse relative to the data drawn on the screen.

6 **Dissect Closest Line Perpendicularly.** *6* perpendicularly intersects the closest line to the current mouse location and breaks the line into two lines at this intersection. The common end point of the two new lines is located on the original line. This common end point is then stored as the *last point*. All line attributes of the original line are inherited by the two new lines including line factor, line type, and boundary flags. If intervals had been specified on the original line, these are divided between the new lines based on relative lengths. Also any regions or sides which used the original line in their definition will be redefined to include the two new lines.

A **Enter Enclosing Region.** *A* defines a region as the set of connecting lines from the database which most tightly enclose the current x,y location of the mouse. If such a definition of lines does not exist in the database, the terminal *beeps* and no region is input. If such a line connectivity can be made, the region will be sequentially numbered and entered into the data base. Also, a diamond will be displayed on the screen. This diamond will be drawn half way along the diagonal between the maximum x,y location, and the minimum x,y location. Depending on the skewness of the region, this may or may not fall within the boundary of the region.

D **Delete Closest Point And Definitions.** *D* searches the database for the point closest to the current x, y location. Once found, this point, and all lines, sides, and regions which are defined using this point are deleted. The items deleted are erased from the screen.

E **Exit.** *E* exits the digitization routines and returns the user to the main menu.

Digitizing Effectiveness

Although digitization is a powerful geometric input method, several techniques can be employed to reduce inadvertent errors and subsequent frustrations. These techniques have proven useful to those familiar with digitization as well as to beginning users.

1. **Outline First.** Before filling in any interior region divisions it is often beneficial to outline the geometry being input. This reduces the number of times the user must remember whether the line is to end at an existing point or at a new point. Be careful to close the outline with one of the *0* prefix button sequences. Interior divisions are then generated with the *0* prefix button sequences to tie to the existing points or with the *6* key to break an existing line.

2. **Input Long Lines and Break Lines Often.** Use of the line break key *6* is especially helpful when inputting and dividing a geometry into meshable regions. Inputting the longest line possible reduces the chance for digitization inaccuracies and assures that straight edges will always remain straight since the break key places the dividing point on the original line. If it is necessary to eliminate all digitization inaccuracies such as on an axisymmetric center line, straighten the long line first using the STRAIGHTEN option (see Section 3.3.4.) Then return to digitization and break this long line into the needed segments.

This eliminates the need to correct each segment and thus reduces the chance of error as well as the required time.

3. **Input Regions As Soon As Possible.** Whenever all the lines needed to enclose a region have been defined, try to input the region using the *A* button. If errors exist and the region does not close, it is always easier to make the corrections for that region's lines only. If regions are only entered after all the lines have been input, many cascading errors may exist which require substantial deletions and redefinitions.

4. **Correct Region Closure Problems Early.** One of the most frequent problems encountered is that all the lines to close the region appear to be on the screen while digitizing, but the region does not close properly using the *A* button. The routine used to determine the enclosing region is designed for polygonal regions, and thus treats all arcs as if they were straight lines. It uses the list of points to find the nearest straight line connection above the mouse location as the start for the closure and then proceeds clockwise. Thus, the mouse location may be critical if the line above the mouse is an arc. Often, however, the error lies in the definition of the lines enclosing the region. When connecting lines, if an existing point was not referenced and instead a new point was generated at almost the same location, the lines may appear to close on the screen when in fact, they do not. The easiest way to find the offending set of points is to exit digitization, enter the *GRAPHICS* main option and draw the current geometric data with point numbers turned on. Any overlapping points will overwrite each other's numbers, and are thus easily identifiable. It is always advisable to correct the closure problem before going on to other region definitions, as subsequent definitions may be tied to these points.

5. **Input Slide Lines By Halves.** When inputting slide lines, first input all items using one side of the slide line including all necessary regions. Once that definition is complete, the other half may be successfully entered. Care must be taken in this area since slide lines generally lie in the same physical location. The algorithms to find the closest point and closest set of existing lines to enclose a region always search from most recent data backwards into the database. Thus if two entities exist at the same location, the search algorithm will choose the most recent as being applicable. Inputting by halves insures that only the points and lines most recently input are used.

6. **Digitize Larger Drawings in Sections.** When the drawing being used is larger than the tablet, the user may digitize the entire object in a piecewise fashion. The *INITIALIZE* main option of **FASTQ** allows the user to move the drawing to a new position, and reinitialize it using two new points. Thus

all the data from one section can be digitized, the drawing then repositioned and reinitiallized, and the procedure repeated. This technique is often more desirable than reducing the original drawing to fit on the tablet since the larger the drawing the more accurate the digitization possible.

7. **Interrupt Digitizing As Often As Needed.** Many times during digitization of complex drawings it is helpful to use the geometric verification utilities of graphics, listings, etc. as explained in Section 3.4. Since digitization can be entered and exited as often and whenever needed without causing any real interruption in the process, the user should feel free to do so as often as desired. Incremental error correction is always less complicated and less frustrating.

8. **Input Points By Hand When Accuracy is Essential.** Whenever accuracy of the digitizer is of concern, the points may be input using a text editor and the **FASTQ** file, or using the *KEYIN* option. Once the points have been defined, the digitizer can be used to connect these points into lines, and subsequently into regions very rapidly. In this manner, one of the major concerns when using the digitizer, that of accuracy, can be eliminated.

9. **Digitize Complex Geometries By Parts.** Geometries where natural parts of the geometry are separable may be digitized piecewise. Each piece can be written to a file when complete, and later recombined using the file merging capability explained in Section 3.2.1. This technique can often reduce the complexity of the data being input and connected using the digitizer, and thus reduce the chance for confusion and error.

3.3 GEOMETRIC DATA MANIPULATION

Once the geometric data has been input, several utilities may be employed to interactively display and modify the geometric database. If the user desires, modification of the data can also be accomplished by *SPAWN*ing from **FASTQ**, editing the **FASTQ** file to make the needed changes, and then returning to **FASTQ** and reading in the new file.

3.3.1 Clearing Geometric Data

The *FLUSH* main option of **FASTQ** may be chosen to delete all existing data. Since this is a particularly dangerous option, it is protected by a confirming inquiry to insure that the user understands the results of a *FLUSH*.

3.3.2 Deletions

Individual items or groups of items may be deleted from the data as desired using the *DELETE* option of **FASTQ**. This option contains suboptions which deal with the deletion of all types of geometric data. The appropriate category for deletion must be chosen from amongst these options. When a suboption is chosen, for instance the *POINTS* suboption, the prompt requests a list of items to be deleted:

DELETE POINTS <I1> THROUGH <I2>:
HIT A RETURN TO END
>

The response can be a limiting list as suggested, or if only one item is to be deleted, the item number alone is sufficient to enable deletion.

3.3.3 Redefinitions

There are several ways to redefine geometric data items. Of course, the item can always be deleted and then reinput, but this is usually unnecessary. The most direct way of changing data is to simply reinput the item from the keyboard using the *KEYIN* main option of **FASTQ**. When this is done, the old definition is overwritten with the new information. Often the entire definition does not need to be reentered to change specifics of the entity. Line intervals, line factors, and region block ID numbers can be changed in the *KEYIN* option without effecting the rest of the definition as can be seen in the flow chart in Figure 1. Points or lines to be added to an existing boundary flag, and regions or barsets to be added to the current body definition can also be input without effecting previously included items by simply choosing the appropriate suboption in *KEYIN*.

3.3.4 Straightening Lines

With the use of digitization as an input method, it is at times necessary to correct digitization errors in the data input. The *STRAIGHTEN* option will force a line to run truly horizontal or truly vertical. This is done by changing either the x or y coordinate of the end points to be the same user input value. This is particularly helpful for axisymmetric centerlines. When the *STRAIGHTEN* main option of **FASTQ** is chosen the user is prompted for a limiting line list (I1 to I2) and a direction. This direction must be given as either the X (horizontal) or Y (vertical) direction. The program then cycles through each of these lines, displays the current coordinate values of the end points of the line, and asks the user for a corrected value.

Note that to straighten a line in the X direction, the Y coordinate values of the end points must be the same and vice-versus.

3.3.5 Data Listing

A full data listing facility is included as the *LIST* main option of **FASTQ**. As in the *DELETE* and *KEYIN* options, suboptions are available for any geometric entity. The listing utility allows the user to view the information for all items in the class (e.g. all lines) or for any user defined subset. As an example, when the suboption *REGIONS* is chosen, the following prompt is presented:

LIST ALL REGIONS?

A Y response will result in a full listing whereas an N response brings up the prompt:

LIST REGIONS <I1> THROUGH <I2>

>

A limiting list may be input, or a single integer will suffice for a listing of one region. If no items of the chosen data type are currently defined, that information will be indicated when the suboption is chosen. The list presented is displayed in pages, or screens, to prevent scrolling past needed information. After each screen has been viewed, the user progresses to the next screen with a carriage return as indicated in the prompt.

3.4 GRAPHICAL DISPLAY OF GEOMETRIC DATA

One of the best validations of the geometric data is the main *GRAPHICS* option. Through this option the data can be quickly scanned for overall correctness with options to toggle the display of most of the parameters affecting the meshing process such as line intervals, boundary flags, etc. for quick visual verification. No plotting of side data has been implemented since sides are combinations of lines and sides contain no specific properties. Often during this verification process, the user can detect line intervals or line factors which must be changed. Therefore, the *KEYIN* options for inputting intervals and factors have also been provided in *GRAPHICS* to allow quick changing and replotting; thus avoiding any option/suboption menu traversal.

3.4.1 Plotting

Plotting in the *GRAPHICS* option is accomplished with the *PLOT* command. When the *PLOT* command is issued only those entities currently flagged as *active* are plotted. Each time the *GRAPHICS* option is entered all points, lines, and regions are flagged as being *active* and the display limits are set to the extremes of the point data. This *active* set can be adjusted by region (*RPLOT*) as explained in Section 3.4.2. The data information (e.g. point numbers, line intervals, etc.) currently toggled ON is plotted along with the geometric shapes as explained in Section 3.4.4 and 3.4.5. If the data is extensive, and the user does not wish to wait for a complete plot, the plotting may be aborted with a Control C and the user will be returned to the *GRAPHICS* prompt.

3.4.2 Plotting By Region

Limiting the data plotting to a specific region or set of regions is often useful when only a limited portion of the geometry is in need of verification, or if the total amount of data for the problem is excessive. The *RPLOT* option of *GRAPHICS* is used to limit plotting to a desired set of region(s). The prompt will request a limiting list of regions (I1 through I2) for plotting. If individual or nonconsecutive regions are desired the user may simply enter one region number (I1) and hit return. Input of additional region(s) to be plotted continues until a solitary return is entered. The regions thus chosen are then plotted. These regions with their associated lines and points remain as the *active* regions. Subsequent *PLOT* and *ZOOM* commands will only operate on this region set and their associated lines and points. This *active* set is reset to all regions when the *GRAPHICS* option is exited and reentered. Redefinition to a new set of active regions can be accomplished by simply repeating the *RPLOT* option.

The *RPLOT* option is particularly useful when the boundaries between regions are not shared as is often the case in slide line definitions. These unshared boundary definitions may overlay each other, and finding correct point and line information associated with a particular region requires the plotting and zooming of these regions independently.

3.4.3 Plotting By Barset

Plotting can also be limited to a specific barset through use of the *BPLOT* option. The implementation is identical to the *RPLOT* option described in Section 3.4.2.

3.4.4 Zoom

The *ZOOM* option is useful in viewing specific portions of the data. There are three possible *ZOOM* implementations:

1. **Cursor Picked Zoom.** When using the cursor to control the zoom window, specify the zoom with the command Z,C. This enables zoom from the cursor locations. The user positions the cursor at a lower left location, strikes any key, and repeats the operation for the upper right location.

2. **Keyboard Specified Zoom.** When specifying limits from the keyboard, type in the command Z, [XMIN], [XMAX], [YMIN], [YMAX] where the values in brackets are numbers. If any of the fields are left blank, the current zoom limits for those values are substituted.

3. **Default Zoom Limits.** When the user wishes to return to the default zoom limits which encompass the *active* regions and barsets, he/she simply types in the command Z with no additional parameters. This will restore the plot to the default zoom limits.

3.4.5 Entity and Property Labeling

Several toggles have been provided as an aid in verification of the mesh defining entities. For instance, to check for appropriate interval and scheme assignments to a region, the user can turn on *INTERVALS* and *SCHEMES*, and turn off every other item. When a toggle is triggered, it switches values (either ON to OFF or OFF to ON) and displays its new setting. Each toggle retains its setting until specifically changed. It is not effected by enter/exit cycles into *GRAPHICS*. The user may review the current setting of all *GRAPHICS* toggles using the *GRAPHICS STATUS* option. The information is plotted on the screen in different colors to help distinguish one property from the other. The available toggles, their default settings, and the color used to plot them on the LS5 terminal are as follows:

TOGGLE	DEFAULT	LS5 COLOR
POINT - Point numbers	ON	Yellow
PBOUNDARY - Point boundary flags	OFF	Pink
LINE - Line numbers	ON	White
INTERVALS - Line Intervals	OFF	Pink
FACTORS - Line Factors	OFF	Red
LBOUNDARY - Line boundary flags	OFF	Blue
SBOUNDARY - Side boundary flags	OFF	Yellow
REGION - Region numbers	ON	Blue
MATERIAL - Region block ID (Material)	OFF	Red
SCHEME - Region scheme	OFF	White

When more than one piece of information is to be displayed at the same location, the information is separated by the forward slash /, and it is ordered in the same order as the list of toggles above. For instance, if line numbers, intervals, and line boundaries are to be listed the line number would be shown first, the number of intervals assigned next, and the line boundary last; all separated by forward slashes.

When more than one boundary flag of the same type has been attached to a point or line, only the first boundary flag in the data is displayed graphically. For a full listing of all flags attached to a point or line the *LIST* option (see Section 3.3.5) should be used.

The *INTERVAL* and *FACTOR* options in *GRAPHICS* refer to the display toggles rather the the input of new intervals and factors for lines as is the case in the *KEYIN* or *MESH* options. This has necessitated that the input of interval and factor changes to the **FASTQ** data in the *GRAPHICS* option be done using options with names different than in *KEYIN* or *MESH*. To input intervals in the *GRAPHICS* option, the user must choose *II*NTERVAL* and to input factors in the *GRAPHICS* option, the user must specify the *IF*ACTOR* option.

3.4.6 Full Property Display

All properties associated with an entity can be turned ON with the *FULL* toggle. The default for this toggle is OFF. When the *FULL* toggle is ON properties are displayed for each entity whose numbering toggle is currently ON. As explained in Section 3.4.5, the color and order of the plotted data are the key to understanding the information which will be displayed. Turning the *FULL* option ON and OFF does not effect the settings of other property toggles although it may override them. For instance, if *LINE* numbers are ON and line *FACTOR*s are OFF when the *FULL* option is turned ON, line factors will be plotted, as they are one of the properties

associated with lines. However, the line factor toggle remains OFF, and when full is turned OFF again, no line factors will be plotted.

3.4.7 Axis Plotting

The *AXIS* toggle has been provided to allow the display of an axis system on the plot. The default for this toggle is OFF. The axis is expanded slightly so as not to overlay any line or point data falling along the extremes of the current *ZOOM* limits.

3.4.8 Hardcopy Plots

Hardcopy plots of the data are available by specifying the *HARD* option when operating in the VAX/VMS environment. This option dumps the graphic information to a file for printing on the specified hardcopy device. The file name reflects the input file specified as a runline parameter if operating in the VAX/VMS environment. If no input file was specified as a runline parameter, the file name of FASTQ is used. The extension to the file reflects the three letter code of the device used for hardcopy printing. For instance, if the input file name was specified on the runline as T15RM5.FSQ and the hardcopy device was QMS, then the hardcopy file would be T15RM5.QMS. All hardcopy plots for one session are placed in the same hardcopy file. If no hard copy device has been initialized (see Section 2.3.1,) **FASTQ** indicates such and continues.

4. MESH GENERATION AND PROCESSING

This chapter deals with the mesh generation and validation process. This includes reviewing previously generated meshes, control during the meshing process, actual mesh generation including various schemes for meshing a region and how they effect the mesh, mesh optimization, graphical mesh verification options, and the mesh output interfaces. All of the options discussed in this chapter are accessible under the main *MESH* option.

4.1 DIRECT MESH INPUT

A previously generated mesh which has been output in the GENESIS[3] database format can be read directly into **FASTQ** using the *MESH READ* option. Thus a mesh can be reviewed and verified as desired without a regeneration of the data. However, most analysts have found that the mesh generation process itself is fast enough that the cost of regenerating the mesh is insignificant to the cost of storing and manipulating the sometimes large mesh output files. Instead of using the actual mesh file for data archival, it is standard practice to only save the **FASTQ** input file and to regenerate the mesh from that file as it is needed.

Once the *MESH READ* option has been specified, **FASTQ** allows the user three chances to specify the appropriate file. If after the third try, the file has not been successfully opened, the program returns to the *MESH* option for further input. The mesh currently defined is not deleted until a new file is opened successfully. Thus, if the *MESH READ* option is chosen by mistake, the user can type nonexistent file names three times, and he will return to the *MESH* option without loss of the current mesh.

4.2 MESH GENERATION CONTROL

There are two ways to control the generation of a mesh from a geometric definition. The first method is automatic processing where the user has no real control of the resulting mesh within a region after processing has begun. This is the procedure used by **FASTQ** in batch mode. The second method allows direct step-by-step user control of the regions to be meshed and of the schemes used to generate mesh within those regions. After generation has been completed using either control method, it can be displayed, verified, and written to a file. Exiting the *MESH* option always clears any generated mesh from the database.

4.2.1 Automatic Processing

Automatic processing is triggered by choosing the *MESH PROCESS* option. When this option is chosen, any existing mesh is first deleted from the database. Then the parameters defined in the geometric database are used to generate the mesh. Messages as to the success and progress of the mesh generation process are written to the screen if running **FASTQ** in interactive mode, or are written to the output file or device if running in batch mode. Any errors in the data definitions and connectivity are flagged and reported as they occur. If the error can be corrected by **FASTQ**, a message of what action was taken is output, and mesh generation continues. This is generally what occurs when the total number of intervals specified around a region are odd. If **FASTQ** cannot correct an error, and the error is limited to a specific region or barset, this region or barset is skipped and all remaining regions are meshed.

4.2.2 STEP Processing

Direct user control of the mesh generation process is enabled by choosing the *MESH STEP* option. When this option is chosen, any existing mesh is deleted, and all regions and barsets within the body are checked to insure that definitions are allowable and complete. The user is then prompted for which barsets and/or regions he wishes to process. The prompt may be answered with either a range of region or barset numbers or an integer specifying a single region or barset. The user is given the opportunity of including other regions or barsets for step processing once the specified entities have been meshed.

Since barset meshing (2-node elements) requires no scheme, the mesh is generated for the barsets without any further interaction with the user. Quadrilateral meshing of regions is much more complicated, and thus each step of the region's meshing scheme can be directly controlled and the results of each step can be plotted by the user during step processing.

Each character in the region's scheme indicates a particular operation as described below and in Section 4.3.2. Each step taken to interactively process the region may contain one or more of these valid scheme characters or step control characters. The first step always uses information from the scheme[17] about the initial meshing primitive to be used for the initial mesh generated in the region (see Section 4.3.1.) The user is initially given the option of using the scheme currently assigned to the region as the first step. If the initial scheme is chosen as the first step, this step is processed and the user is prompted for further steps. Otherwise,

[17]The scheme contains this information either explicitly or by default.

the user respecifies the desired initial scheme. Subsequent steps are then taken until the user is satisfied with the resulting mesh or chooses to abandon processing of the region. A response of H will result in the display of a help message listing all possible scheme and step control characters and a brief description of each. A response of P will plot the current mesh on the terminal. If the region is accepted at the end of step processing, the complete sequence of scheme characters is stored as that regions new scheme[18] for subsequent processing of the region. The available stepping control characters are:

P Plots the mesh in its current state.

H Displays the help message.

E Ends processing and saves the mesh.

Q Quits processing without saving the mesh.

O (Originate) Erases the current accumulated scheme and restarts processing of the region.

4.3 MESH GENERATION

The mesh generation in **FASTQ** is tied directly to the geometric definitions used in forming regions and/or barsets to be meshed. The element types currently supported include the 2-node and 3-node bar, the 4-node quadrilateral, the 8-node isoparametric quadrilateral, and the 9-node isoparametric quadrilateral as discussed in Section 1.1.6. Mesh generation for barsets and regions is discussed in this section.

4.3.1 Barset Mesh Generation

Bar element (two or three node) generation is based on geometric lines and is significantly less complicated than quadrilateral processing. Generation of 2-node bar elements is accomplished by dividing the given lines at appropriate spacings specified by the number of intervals and the factor for each of the lines. These nodes are then connected to form elements. If the quadrilateral mesh generation flag is set for eight or nine node quadrilateral element generation, all bar elements will also be given midside nodes resulting in 3-node bar elements. Again, as mentioned in Section 3.1.6, a specific element connectivity can be forced by using an inside point in the definition of the barset. If an inside point is specified, the element is numbered

[18] All stepping control characters except P are excluded from the saved scheme.

such that the first element node is to the right of a vector drawn from the inside point to the midpoint of the element. The last element node is conversely to the left of that vector. If no inside point is specified, connectivity progresses from the beginning point to the ending point of each line in the barset.

4.3.2 Quadrilateral Mesh Generation

The quadrilateral mesh generation algorithm in **FASTQ** is based on the mapping of a region into a unit square[10,11]. The real power of **FASTQ** is in its ability to not only mesh the basic rectangular region with all quadrilateral elements, but also to mesh other primitive shapes algorithmically. In order to accomplish this task the basic algorithm has been extended to mesh the nonrectangular primitive shapes of circles, quartercircles, semicircles, triangles, and penatgons as well as to mesh transition regions from fine to coarse element sizes with little or no user interaction being required. Several mesh modification options are also available to correct some meshing problems encountered when working with more arbitrarily shaped regions.

The quadrilateral mesh generated within a region is controlled by the scheme attached to each region. This scheme indicates, either explicitly or by default, an initial meshing primitive, subsequent adjustment of the initial mesh, and if step processing, control of the interactive meshing process. A scheme can either be associated with a particular region using the scheme card (see Section 3.1.7,) can be defaulted for all nonspecified regions, or can be defaulted by no user specification. The default scheme in **FASTQ** when otherwise left unspecified by the user is *M* indicating a general rectangle primitive mesh.

The generation of eight or nine node elements is controlled by the element type flag. This flag can be set in the *MESH* or *KEYIN* option of **FASTQ** with the appropriate *EIGHT* or *NINE* option toggle. It can also be set with a card in the FASTQ file consisting of the word *EIGHT* or *NINE* indicating which type of element is desired. The default is generation of four node quadrilateral elements. Only one type of quadrilateral element may be generated for a body, but quadrilateral elements may be combined with bar elements in the same body.

Initial Mesh Generation

When processing regions, **FASTQ** first checks all the data associated with the region for connectivity and validity. The results of this check are output to the user to enable correction of any existing problems with the data. **FASTQ** then generates an initial mesh for each region. The initial mesh is generated based on variations of the basic rectangular mapping algorithm[10,11] depending on which meshing primitive

scheme has been indicated, and the location of nodes around the perimeter of the region.

FASTQ generates the perimeter nodes of the region based on the line definitions and their interval and factor assignments. All initial meshing primitives require an even number of nodes[19] around the perimeter. If an odd number has been specified, the program attempts to correct the error by increasing the number of intervals on the line in the boundary which has the largest interval spacing and has not yet been used for meshing of a previous region. Once a line has been used for meshing of a region, its intervals remain fixed to insure continuity across boundaries. Since **FASTQ** is only attempting to correct the error locally, a line interval change in one region may force cascading interval changes in other unprocessed regions which contain the same line.

Once the perimeter has been generated successfully, the program generates all interior nodes, elements, and connectivity using mapping algorithms based on the primitive indicated in the scheme for the region. **FASTQ** currently contains six meshing primitive scheme indicators. **FASTQ** searches the entire scheme for the primitive scheme characters and thus they need not be the first letter of the scheme. The generation of the mesh is always dependent on the existence of natural corners[20] and sides[21]. These natural sides are in no way tied to the geometric sides explained in Section 3.1.4, in that the user need not explicitly define a side. In fact, in only one primitive, the forced rectangle primitive, is the geometric side definition of the user followed in initial mesh generation. The primitive schemes are as follows:

M	General Rectangle Primitive (Start of a Circle Primitive).
T	Triangle Primitive.
B	Transition Primitive.
C	Semicircle Primitive.
U	Pentagon Primitive.
otherwise	Forced Rectangle Primitive.

[19]The total number of nodes around the perimeter is always equal to the total number on intervals on the lines in the perimeter.

[20]Natural corners have interior angles of about 90 degrees.

[21]Natural sides are continuous portions of the boundary that have no interior angles deviating far from 180 degrees.

General Rectangle Primitive - M An *M* in the scheme indicates that the region will be meshed as a rectangle, and that **FASTQ** will algorithmically choose four *good* corners for this rectangle from the perimeter nodes around the region. This choice is based on interior angles of perimeter nodes and boundary interval assignments. For this scheme to produce a good initial mesh, the region must have 4 natural corners and the user must specify equal intervals on opposing natural sides of the region. No formal side definitions are followed in the choice of the best corners. Figure 2 shows some example rectangular primitive geometries with line interval assignments and scheme plotted, and the resulting mesh. Circular regions, regions with one natural side and no natural corners, can be meshed using the general rectangular primitive *M* with one or more necklaces with smoothing *NS* subsequently added as explained in Section 4.3.2. Figure 3 shows some example circular primitive geometries assignments and scheme plotted, and the resulting mesh.

It is possible that incorrect interval assignments will force mapping corners for the region to lie along fairly straight sides. This problem is usually apparent in a plot of the resulting mesh, but under certain situations the mapping produces completely flat elements along a straight side, and the plot may still appear normal. Thus, whenever a mapping corner has a large interior angle, the user is given a warning to indicate that there may be problems in the final mesh. This warning will always appear when processing circular regions as they contain no natural corners, and in this instance the warning can be ignored.

Triangle Primitive - T A *T* in the scheme indicates that the region should be initially meshed as a triangle. The definition of a triangle is general in that regions containing 3 natural corners can often be meshed successfully with this algorithm. For instance, quarter circular shapes are handled nicely by this primitive. Again, the algorithmic choice of good corners is based on interior angles of perimeter nodes and boundary interval assignments. This algorithm requires that there be at least 6 intervals around the perimeter of the region. For this scheme to produce a good initial mesh the natural side with the most intervals must have at least 2 less intervals than the sum of the intervals on the remaining two sides. The triangle primitive algorithm divides the initial region into three rectangular subregions based on region shape and interval assignment. Each of these subregions are then meshed and combined into one region for subsequent processing. This always results in one of the interior nodes being connected to only three elements instead of the standard four elements. The location of this node can be adjusted with interval assignments. As all three natural side interval assignments become equal, the node becomes equally far from each boundary. An often beneficial next step is the application of the length weighted Laplacian smoother *6S* or centroid area pull smoother *4S* as explained in Section 4.3.2. Figure 4 shows some example triangular primitive geometries with line interval assignments

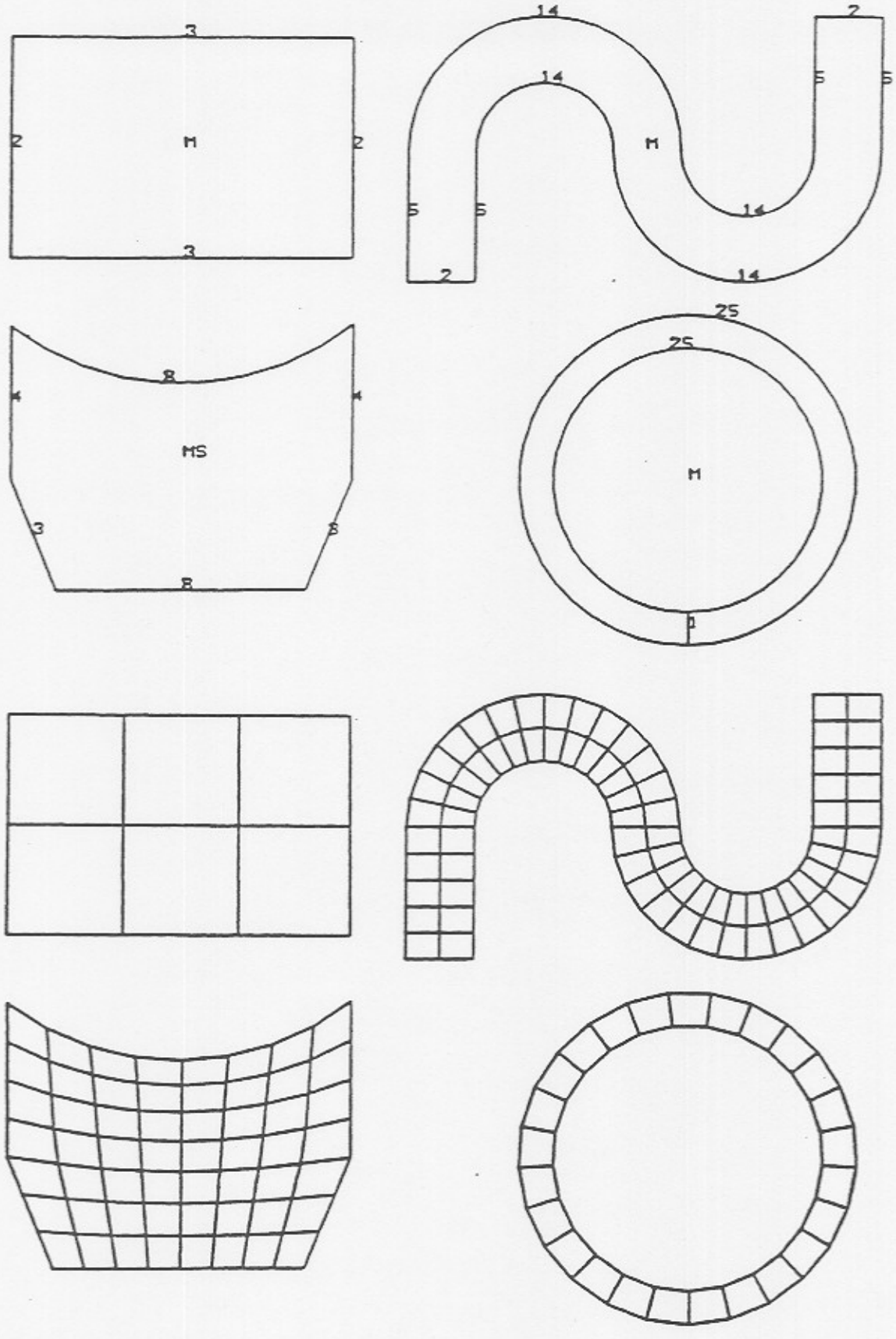


Figure 2. RECTANGULAR PRIMITIVE GEOMETRIES AND MESH

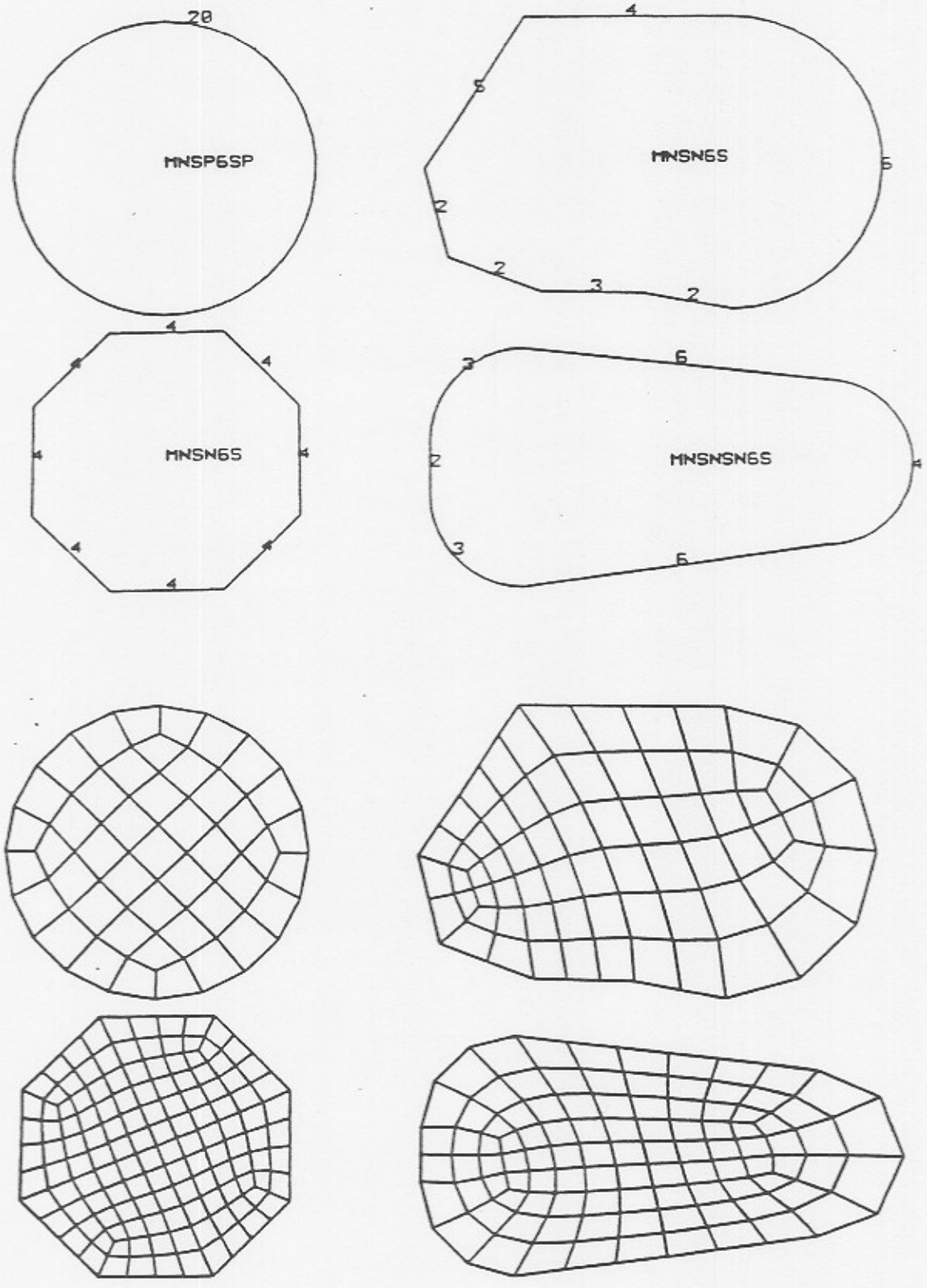


Figure 3. CIRCULAR PRIMITIVE GEOMETRIES AND MESH

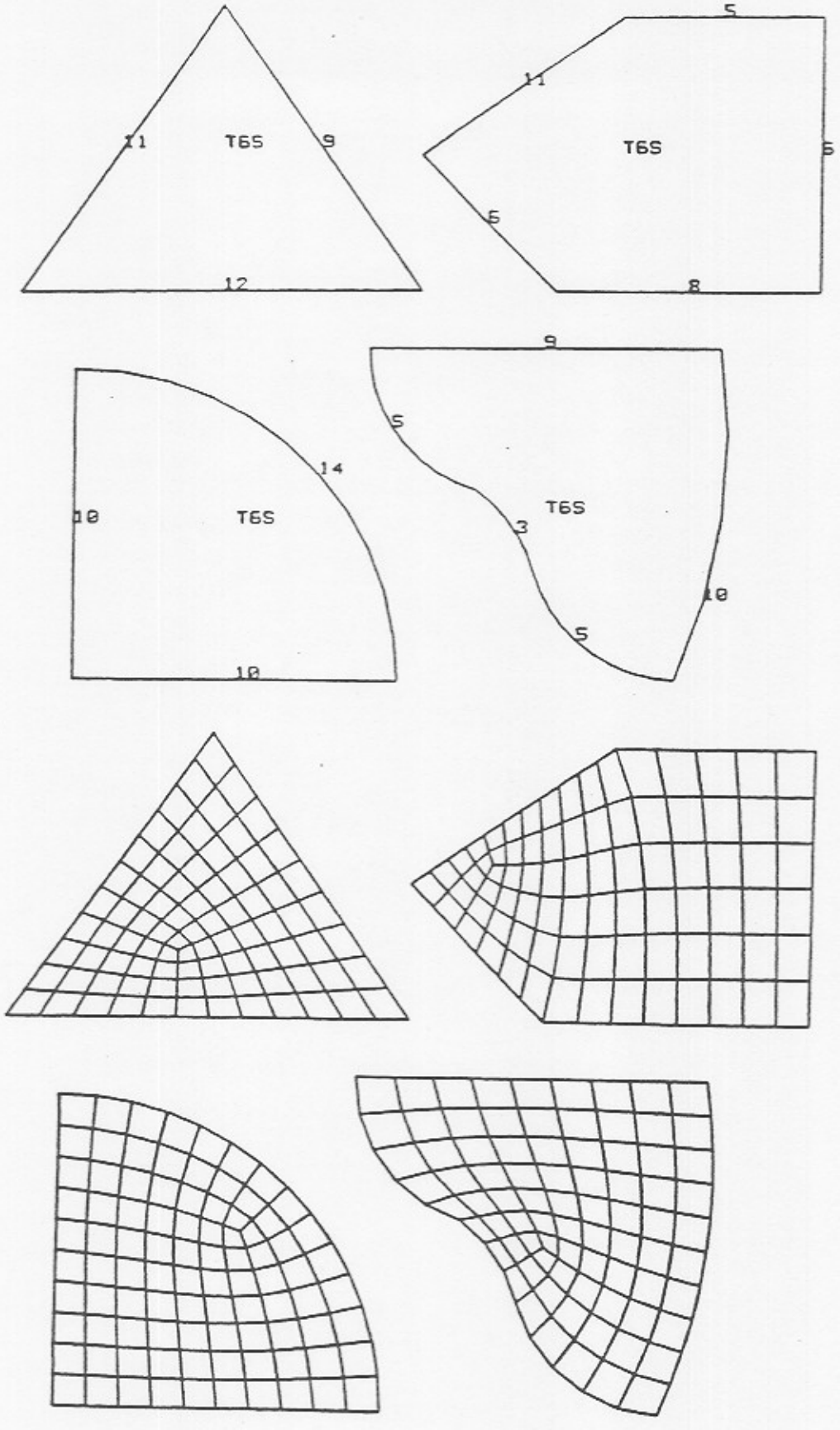


Figure 4. TRIANGULAR PRIMITIVE GEOMETRIES AND MESH

and scheme plotted, and the resulting mesh.

Transition Primitive - B A *B* in the scheme indicates that the region will be meshed as a transition between differing numbers of elements on opposing sides of the region. This primitive has been designed for use with rectangular shaped regions, but has proven to be useful for other shapes as well. The algorithm requires that there be at least 8 intervals around the perimeter of the region. No relationship is imposed between intervals of lines on opposing sides of the transition region. The algorithm uses the perimeter nodes to choose four best corners, and designates a base side that contains the most unmatched intervals, that is the opposing natural side contains a different number of intervals than the base side. For this scheme to produce a good initial mesh, there must be at least 4 intervals in the perimeter which are not along the natural base. The algorithm divides the initial region into two logical triangles, and subsequently divides each of the two triangles into three subregions. This results in a total of six rectangular subregions. This always produces two interior nodes attached to only 3 elements in the mesh. Again the location of these two nodes can be controlled with boundary interval assignment. The length weighted Laplacian smoother *6S* or the centroid area pull smoother *4S* may be helpful in improving the element quality after initial generation (see Section 4.3.2.) Figure 5 shows several transition region geometries with line interval assignments and scheme plotted, and the resulting mesh.

Semicircle Primitive - C A *C* in the scheme indicates that the region will be meshed as a semicircular region. This primitive has been designed for use with regions with two natural corners. It uses the same strategy as the transition region except that only two natural corners are chosen instead of four. The formation of two triangles again requires that there be at least 8 intervals around the perimeter of the region. No relationship is imposed between intervals of opposing sides of the region. However, for this scheme to produce a good initial mesh, there must be at least 4 intervals on each of the natural sides. As in the transition region, this algorithm always produces two interior nodes attached to only 3 elements in the mesh and the location of these two nodes can be controlled with boundary interval assignments. The length weighted Laplacian smoother *6S* or the centroid area pull smoother *4S* may be helpful in improving the element quality after initial generation (see Section 4.3.2.) Figure 6 shows several semicircular primitive geometries with line interval assignments and scheme plotted, and the resulting mesh.

Pentagon Primitive - U A *U* in the scheme indicates that the region will be meshed as a pentagon primitive. This primitive has been designed for regions containing 5 natural sides and thus 5 natural corners. This primitive is useful for meshing

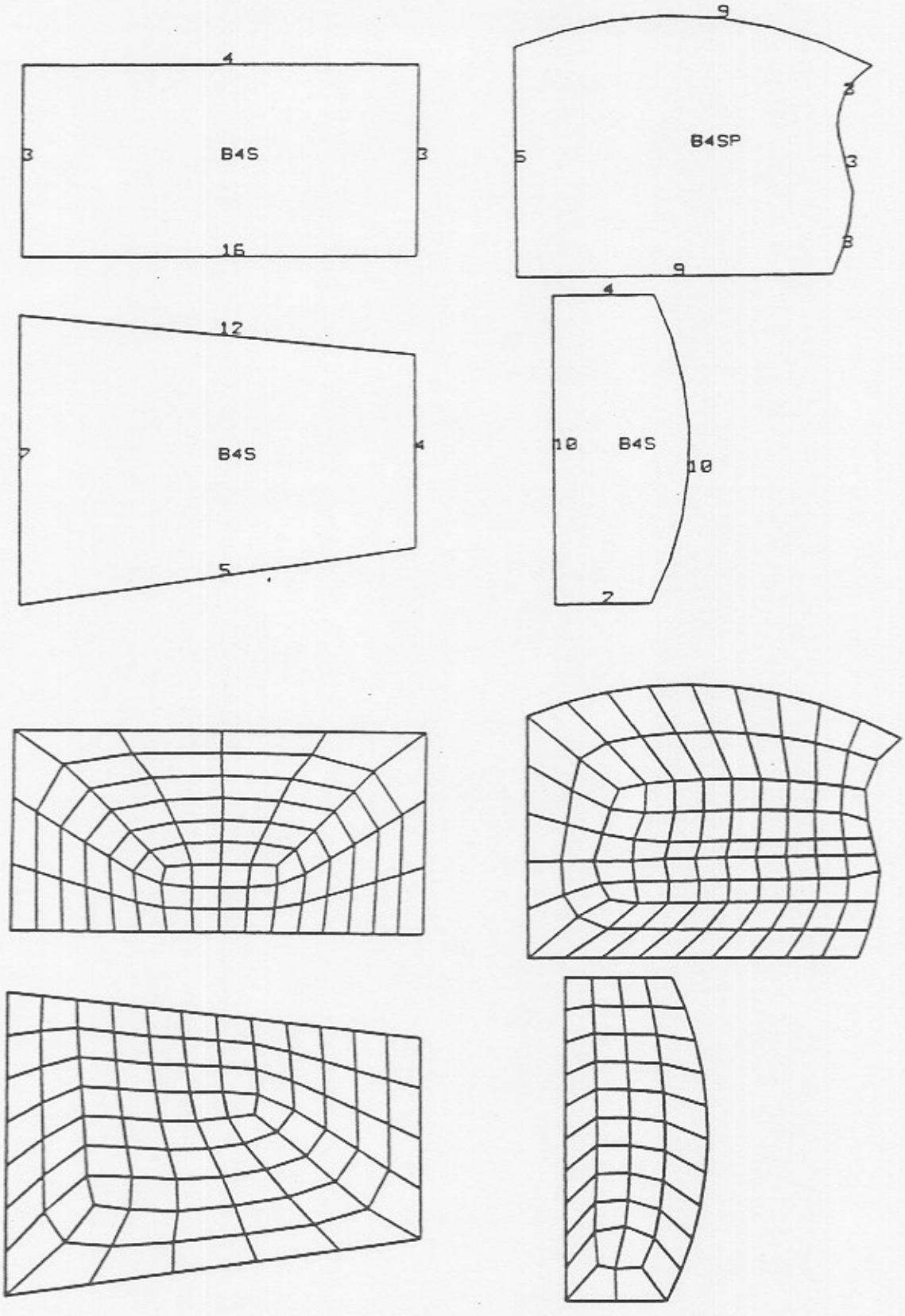


Figure 5. TRANSITION PRIMITIVE GEOMETRIES AND MESH

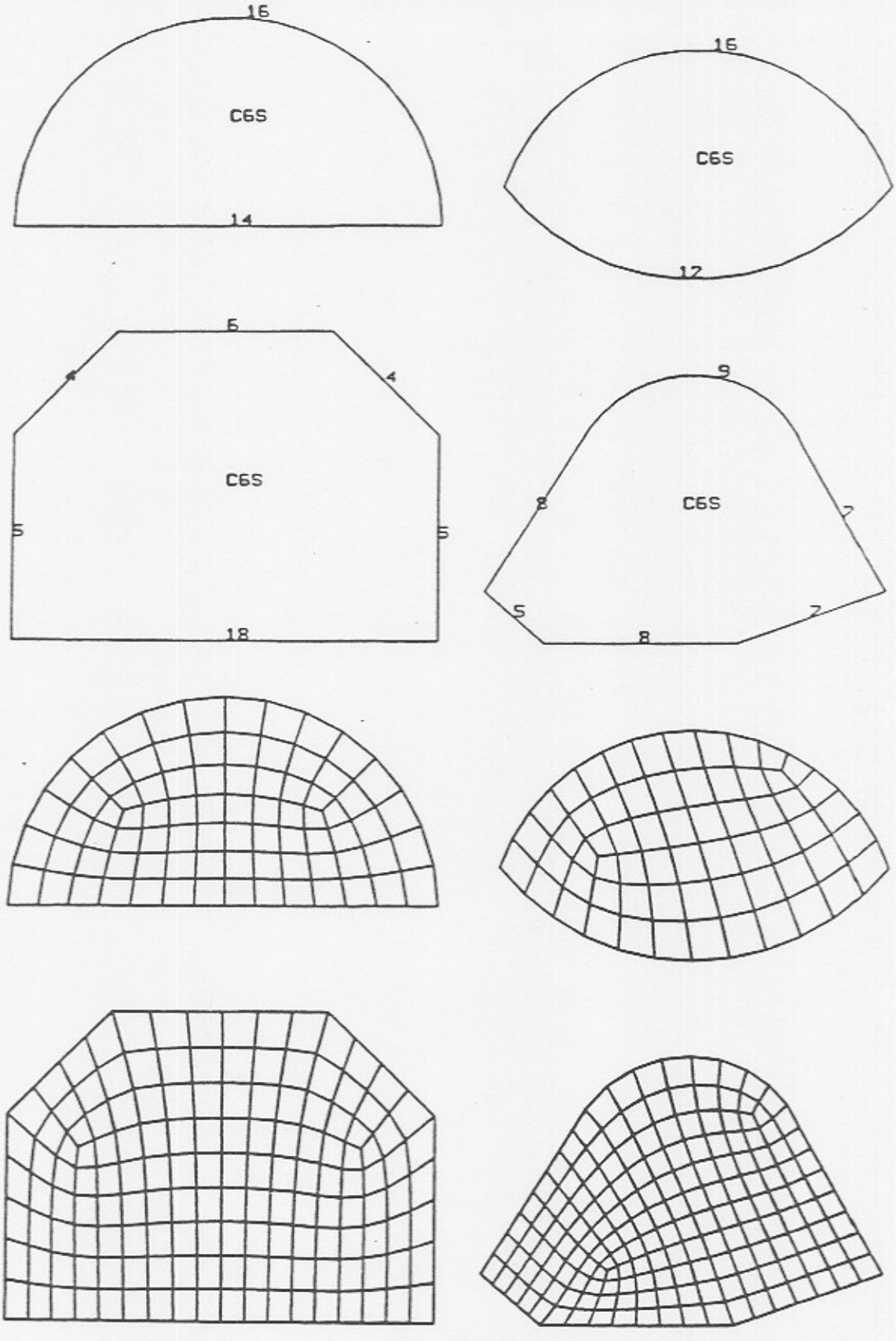


Figure 6. SEMICIRCULAR PRIMITIVE GEOMETRIES AND MESH

around holes in square geometries and a number of other shapes as shown in Figure 7. The algorithm subdivides the initial region into five subregions, and then combines these regions into the completed initial mesh. This requires that there be at least 10 intervals around the perimeter of the region to produce a mesh. Two other requirements must be met for this scheme to produce a good initial mesh. First, each of the natural sides must have at least 2 intervals. Second, the sum of the intervals on any two consecutive sides must be less than the sum of the intervals on the remaining three sides. This algorithm always produces one interior node which is attached to five elements rather than the standard four elements. The location of this node can be controlled by interval assignment. As interval assignments for all five sides become equal, the node connected to five elements becomes equally far from each boundary. Again, the length weighted Laplacian smoother *6S* may be helpful in improving the element quality after initial mesh generation (see Section 4.3.2.) Figure 7 shows several pentagon primitive geometries with line interval assignments and scheme plotted, and the resulting mesh. This primitive was developed by Michael B. Stephenson at Brigham Young University for inclusion in **FASTQ**.

Forced Rectangle Primitive The lack of any other initial meshing scheme characters (*B*, *C*, *M*, *U*, or *T*) indicates that the region will be meshed as a forced rectangle. This is the most restraining of the initial meshing primitives (*M* is much more general.) The forced rectangle requires that the first geometric side be explicitly defined by the user, and that this geometric side correspond to the logical side for mapping. This forces the choice of all corners, the first two from the side defined, and the last two based on the requirement that intervals on sides 1 and 2 are respectively equal to intervals on opposing sides 3 and 4. If no initial side is specified, the first line in the region list of lines is taken as forming the first logical side.

Mesh Modification

The initial mesh generated for a region can sometimes be improved by subsequent processing. The results of this processing is region dependent. The geometry of the region and the connectivity of the initial mesh generated for that region often are predominant factors in the success of the modification attempts. Generally a poorly generated initial mesh (e.g. two sides of an element fall along the same natural side, nodes initially placed outside the initial boundaries, etc.) is difficult to correct with subsequent processing. The exception to this is the circle primitive (see Section ??.) Problems with element size variations and angle distortions, on the other hand, can often be resolved with further processing.

The modification scheme options can be divided into three groups. The first group is the smoothing operations. The smoothers adjust interior node locations

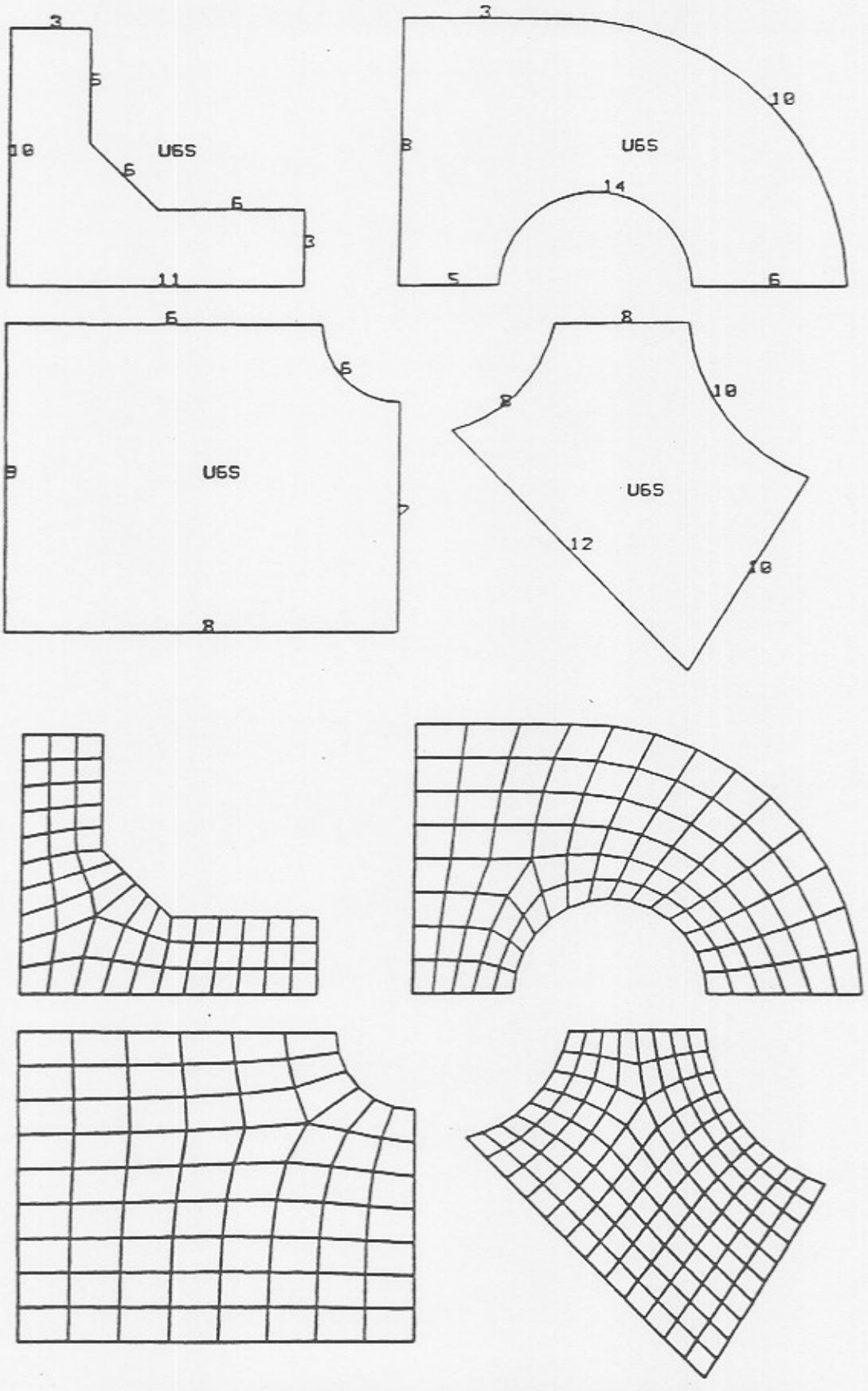


Figure 7. PENTAGON PRIMITIVE GEOMETRIES AND MESH

but do not change the initial element structure. The second group is the structural change operations. These operations change the connectivity and/or the number of elements in the region. The third group of options are parameter modifiers. These parameters adjust the effect of smoothing and structural change operations, and allow for repetitive sequences of operations.

Smoothing Operations **FASTQ** implements 6 different smoothing algorithms [2]. Inclusion of the letter *S* in the scheme initiates smoothing using the algorithm currently in effect. Specifying numerals in the region scheme sets the corresponding smoother active for use when the next *S* in the scheme is reached. If an *S* is encountered in the scheme and no changes have occurred in the mesh or in the choice of active smoother since the last smoothing operation, the *S* is ignored. The smoothing process is iterative, and the user can control the number of allowed iterations through changing parameters described in Section 4.3.2. A complete description of each smoother is included in[2] and thus only a cursory description will be attempted here. The default smoother for rectangle primitive initial meshes before any structure adjustment operations is the Equipotential smoother *1*. The default for all other meshes is the Area Pull And Laplacian smoother *2*. The smoothers available are as follows:

1 Equipotential.

2 Area Pull and Laplacian.

3 Centroid Inverse Area Push And Laplacian.

4 Centroid Area Pull.

5 Laplacian.

6 Length Weighted Laplacian.

Equipotential Smoother - 1. A *1* in the scheme sets the active smoother to be the equipotential smoother if the mesh is regular in that the initial meshing primitive was rectangular and no structural adjustment operations have occurred. Otherwise the Area Pull and Laplacian smoother is used. This smoother attempts to move nodes to position a set of regular *vertical* lines and *horizontal* lines as two intersecting sets of equipotentials. It converges quickly, is effective on concave and convex regions, and can sometimes pull nodes which have fallen outside the region back into the interior.

Area Pull and Laplacian Smoother - 2. A *2* in the scheme sets the active smoother to be the area pull and Laplacian smoother. It is the default smoother

for nonregular meshes. This smoother moves nodes to let small elements grow at the expense of larger ones. It tends to pull nodes back into the interior of the region although not as well as the Equipotential smoother.

Centroid Inverse Area Push And Laplacian - 3. A *3* in the scheme sets the active smoother to be the Centroid Inverse Area Push And Laplacian smoother. This smoother pushes small elements out to equalize adjacent element areas. It requires that no nodes be initially outside the boundary, and is fairly slow to converge. It does have a strong tendency to produce nearly square or rectangular elements.

Centroid Area Pull Smoother - 4. A *4* in the scheme sets the active smoother to be the centroid area pull smoother. This smoother attempts to force equal areas by pulling a node based on the direction of the centroid of all connected elements weighted by their respective areas. It works as well as the Laplacian smoother and has a tendency to pull nodes which fall outside the region back into the interior.

Laplacian Smoother - 5. A *5* in the scheme sets the active smoother to be the Laplacian smoother. This smoother is the most common algorithm used in other codes, and repositions nodes based on the average of the locations of all the nearest neighbors. It performs well if the region being smoothed is nicely convex. Otherwise, it often pushes nodes outside the boundary.

Length Weighted Laplacian - 6. A *6* in the scheme sets the active smoother to be the Length Weighted Laplacian. This smoother moves a node along a vector sum of weighted vectors from the node to nearest neighbor nodes. The weighting is based on the relative distance to neighbor nodes. It tends to move nodes so that attached element sides are of more equal length. It does not perform well on concave sides, but does well when large node movements within the region are required.

Structural Change Operations The second group of mesh improvement options is the structural change operations. These operations change the connectivity and number of elements within the region. There are three structural change operators:

R Restructures connectivity within adjacent elements.

D Deletes all elements with angles smaller than a threshold.

W Deletes the element with the worst interior angle under a threshold.

N Adds an additional row of elements around the boundary.

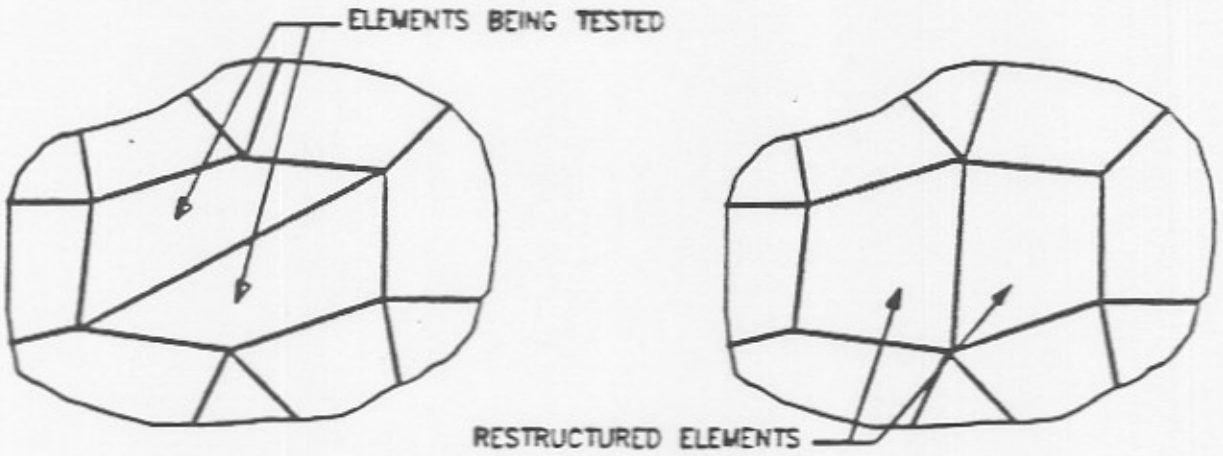


Figure 8. MESH RESTRUCTURING

These operations are described fully in[2] and thus only a cursory description will be attempted here. Parameters described in Section 4.3.2 can be used to modify the effect of these operators.

Restructure - R. An *R* in the scheme causes **FASTQ** to search the mesh for adjacent elements where removing the shared line and replacing it with another results in two better elements. A case where this is beneficial is shown in Figure 8. The quality of the elements is judged on a condition number which is sensitive to nonrectangular elements (nonsquare corners.) This is an iterative and relatively expensive process since it requires that each adjacent pair of elements have the respective merit of all three possible connections evaluated. If a better choice can be made based on this evaluation, the connectivity of the two elements is adjusted to implement the change. Restructuring does not change the number of elements or nodes in a region.

Delete - D. A *D* in the scheme causes the deletion of all elements whose smallest interior angle is less than a threshold amount. The deletion is accomplished by collapsing the two nodes on the shortest diagonal into one as shown in Figure 9. The value of the threshold can be adjusted with the *V* parameter as explained in Section 4.3.2. Since boundary nodes must remain in place to assure connectivity to adjacent regions, only those elements which will not effect the boundary nodes may be deleted.

Worst Element Deletion - W. A *W* allows the deletion of only one element. If any elements contain interior angles less than the threshold amount, the element with the smallest interior angle is deleted.

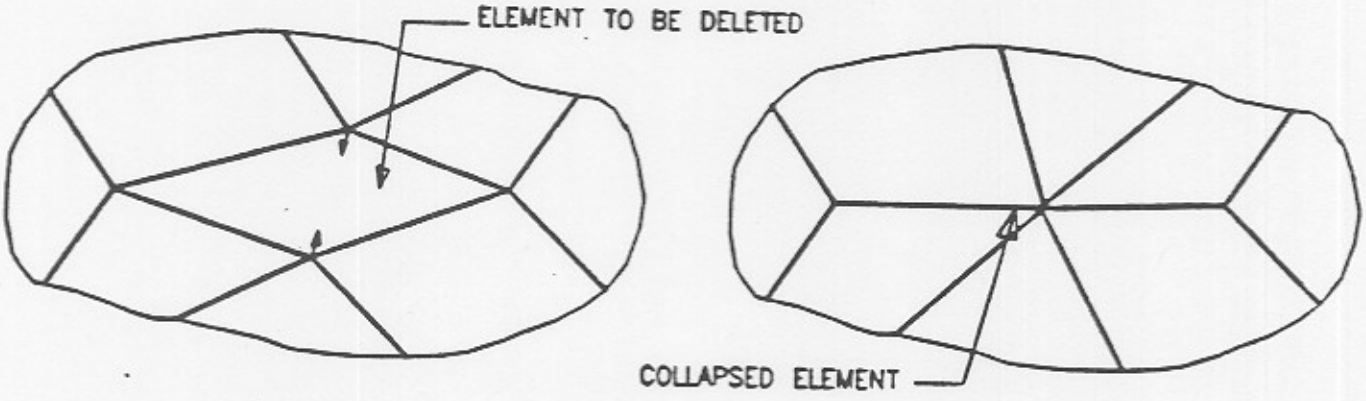


Figure 9. ELEMENT DELETION

Necklacing - N. An *N* causes a shrinking of the original elements, and the generation of an extra row of elements around the boundary of the region as shown in Figure 10. This should usually be followed by a smoothing operation. Multiple necklaces may be installed on a region.

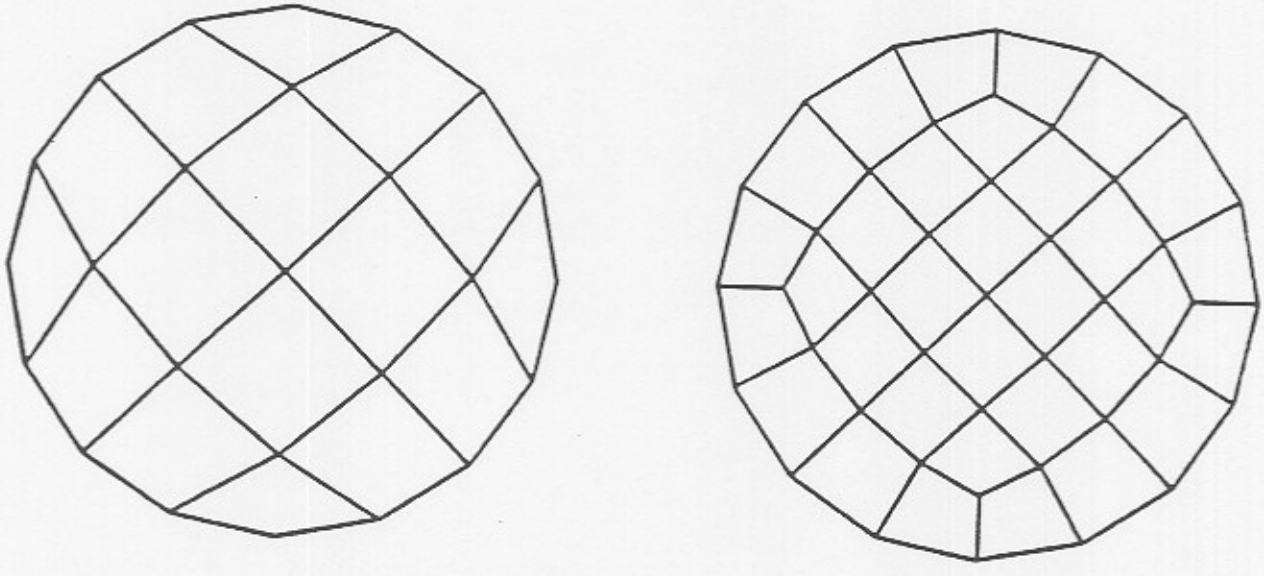


Figure 10. REGION NECKLACING

Scheme Parameter Controls Several parameters controlling the effect of smoothers and of the structural change operations can be changed by the user. These parameters include:

A or +A	Increases smoothing parameter by 0.1 to force more equal element areas. The default is 0.7. The maximum is 1.0.
-A	Decreases smoothing parameter by 0.1.
F or +F	Increases smoothing relaxation parameter by 0.25. The default is 1.0. This increases node movement per iteration.
-F	Decreases the smoothing relaxation parameter by 0.25.
I or +I	Increases the maximum allowable number of iterations for smoothing by 50%. The default is 5 times the number of elements.
-I	Decreases the maximum allowable iterations for smoothing by 33%.
J or +J	Increases the node movement tolerance for convergence of smoothing by a factor of the cube root of 2. The default is 3% of the smallest element side length.
-J	Decreases the node movement tolerance for convergence of smoothing by a factor of the cube root of 2.

A sequence of operations can be repeated with the use of the left *(* and right *)* parenthesis. This causes the scheme sequence contained within the parenthesis to be repeated until the repetition causes no change in the mesh. For poor initial meshes, a loop of *(DRS)* is sometimes useful. Nested loops are also supported.

4.4 MESH RENUMBERING

The mesh for each region is stored individually until all region processing has been completed. After processing has been completed, all regions are combined for final node and element numbering. **FASTQ** does not duplicate the generation of any nodes, and thus it does not do any purging of duplicate nodes. The renumbering process is one of replacing the internal numbering scheme of nodes and elements with a sequential numbering. Nodes are renumbered based on generation order unless optimization (see Section 4.5) is in effect. Elements are renumbered with a sequential numbering system within each block ID of elements to match the GENESIS[3] output.

This final mesh is then stored in the database for verification, graphical display, and output to the analysis code.

During the renumbering process extra nodes are generated for eight and nine node quadrilateral elements if the appropriate flag has been set (see Section 4.3.2.) Midside nodes are placed on the geometric boundary for element sides which have been generated along the boundary. All other midside nodes are generated at midpoints on the element sides. The center node for nine node quadrilaterals is placed halfway between the two closest opposing midside nodes.

4.5 MESH OPTIMIZATION

FASTQ's mesh optimizer performs two functions. First, it optimizes bandwidth or node renumbering, and second, it optimizes wavefront or element reordering.[22] Both functions are enabled when a RENUM card is found in the file, or when the *OPTIMIZE* option is chosen in the *MESH* or *KEYIN* menus. Since the reordering of elements is based on the renumbered nodes, allowing for two distinct optimizers would have been an unnecessary complication.

An important distinction exists between optimum node renumbering and element reordering. Node numbers are actually reassigned during the bandwidth optimization. The old node numbers are not maintained and all boundary flags are transferred from old to new node numbers. On the other hand, elements are not renumbered. The elements are mapped into an ordering array which designates the sequence in which the elements should be processed by a wavefront solver. Thus all boundary flag information which refers to element numbers remains associated with the unchanged element number, and not the element order. This data management scheme has been driven by the GENESIS[3] database format which forces all element numbers to be consecutive within a block.

The Cuthill-McKee algorithm[2] is employed for node renumbering and subsequent element reordering. With this algorithm, a seed list of nodes is generated as explained in the control options below. Any series of these controls can be added to the data for processing. For instance, an *X-Y* control can be followed by a *P-L-P* control and three *NODE* controls for the same object. The seed list is extended until all renumbering controls have been exhausted. After controls have been exhausted, this initial list is sequentially numbered and then replaced by all the nearest neighbors to the list. This process is repeated until all nodes in the mesh have been renumbered. All nodes in the mesh will be found even if the geometry contains unconnected regions which is often the case for problems involving slide lines. Element reordering

[22] There is a difference between renumbering and reordering. Read on.

uses the new node numbers to sort attached elements for wavefront optimization.

Renumbering control can be specified on data cards in the FASTQ file (see Appendix A,) or may be entered interactively using the *KEYIN RENUM* option.

4.5.1 Default Optimization Control

The default control is indicated when a card with only the keyword *RENUM* is located in the input FASTQ file, or when the optimize toggle has been set interactively as explained above. This allows optimization to be accomplished, but offers no direction as to where to begin the process. The default control simply chooses the first node in the database as the seed list to begin optimization.

4.5.2 P-L-P Renumbering Control

The *P-L-P* renumbering control allows the user to indicate that nodes along a series of points and lines be added to the seed list for the Cuthill-McKee processor. This option enables a maximum list of 11 alternating point and line numbers from which nodes are selected as the seed list to start the renumbering. The point and line numbers must be continuous around a portion of the boundary of the object.

4.5.3 X-Y Renumbering Control

The *X-Y* renumbering control allows the user to being optimization in a certain physical location. The node closest to this location is entered into the seed list for the processor.

4.5.4 NODE Renumbering Control

The *NODE* renumbering control allows the user to indicate the series of actual nodes based on their *NUIDs (Node Unique Identifiers)* to form the seed list for renumbering. The *NUID* is an internal numbering scheme given to nodes during mesh generation. The *NUID* of nodes at points is the point number. The *NUID* of nodes along a line is:

$$NUID = 10^9 + (LineNumber \times 10^5) + SequenceNumber$$

The sequence number goes from 2 to the number of intervals along the line. The *NUID* of nodes interior to a region is:

$$NUID = (RegionNumber \times 10^5) + SequenceNumber$$

The sequence number goes from 1 to the number of interior nodes in the region. Using the *NODE* card, the user can specify any node in the mesh as a node to be added in the initial seed list.

4.6 GRAPHICAL DISPLAY OF MESH DATA

One of the best methods of validation of the mesh generated by **FASTQ** is the *MESH GRAPHICS* option. Through this option the mesh can be quickly scanned for correctness and goodness with options to toggle the display of all information contained in the mesh database.

4.6.1 Plotting

Plotting in the *MESH GRAPHICS* option is accomplished with the *PLOT* command. When the *PLOT* command is issued only those entities currently flagged as *active* are plotted. Each time the *MESH GRAPHICS* option is entered all elements and nodes are flagged as being *active* and the display limits are set to the extremes of the nodal data. This *active* set can be adjusted by region, barset, block ID number, or element number as described in Section 4.6.2. The mesh information (e.g. element numbers, nodal boundary flags, etc.) which is currently toggled ON is plotted with the mesh. If the mesh is extensive and the user does not wish to wait for a complete plot, the plotting may be aborted with a Control C and the user is returned to the *MESH GRAPHICS* prompt.

If mesh optimization has been enabled, then the plotting of the elements to the screen follows the optimized element sequence. That is, the outline of the element which is to be processed first in the optimized wavefront mapping is plotted first, etc. As a result, the element plotting sequence dynamically displays the development of the optimized wavefront.

4.6.2 Plotting By Attributes

Plotting by attributes limits the plotting to elements and nodes which have chosen characteristics. When these options are chosen, the *active* elements are replaced with the new list. This is useful when only a limited portion of the mesh needs verification or if the mesh is too large to verify at one time. The following attribute plotting is available:

RPLOT Plots elements and nodes by region(s).

BPLOT Plots elements and nodes by barset(s).

MPLOT Plots elements and nodes by block ID(s) (material number).

EPLOT Plots elements and their nodes by element number.

FASTQ will request a beginning (I1) through an ending (I2) attribute for plotting. If the mesh with individual or nonconsecutive attributes is desired the user may simply enter one number (I1) and hit return. Input of additional attribute numbers continues until a solitary return signals completion as in other **FASTQ** input. The elements, nodes, and activated parameters[23] (see Section 4.6.4) in the portions of the mesh chosen are then plotted. These elements and nodes remain *active* and subsequent *PLOT* and *ZOOM* commands operate on this set. The *active* set is cleared whenever one of the attribute plotting options are chosen, and thus the options may not be mixed. This *active* set is reset to the complete mesh when the *MESH GRAPHICS* option is exited and reentered.

Plotting by attributes is particularly useful when the boundaries between regions are not shared as is often the case in slide line definitions. Often these unshared boundary definitions overlay each other, and finding correct node numbers and boundary flags associated with a particular region or element require the plotting and zooming of these elements and nodes independently.

4.6.3 Zoom

The *ZOOM* option is useful in viewing specific portions of the mesh. There are three possible *ZOOM* implementations:

1. **Cursor Picked Zoom.** When using the cursor to control the zoom window, specify the zoom with the command Z,C. This enables zoom from the cursor locations. The user positions the cursor at a lower left location, strikes any key, and repeats the operation for the upper right location.

2. **Keyboard Specified Zoom.** When specifying limits from the keyboard type in the command Z, [XMIN], [XMAX], [YMIN], [YMAX] where the values in brackets are numbers. If any of the fields are left blank, the current zoom limits for those values are substituted.

[23] The activated parameters are those whose toggles are currently ON.

3. **Default Zoom Limits.** When the user wishes to return to the default zoom limits which encompass the *active* portions of the mesh, he/she simply types in the command Z with no additional parameters. This will restore the plot to the default zoom limits.

4.6.4 Mesh Numbering and Parameter Display

Several toggles are available to enable/disable display of the mesh numbering and associated parameters. Each time a toggle is chosen, it switches (toggles) to the opposite setting and indicates its new status (either ON or OFF.) Each toggle retains its status until specifically changed. It is not effected by enter/exit cycles into *MESH GRAPHICS*. The user may review the current setting of the all toggles using the *MESH GRAPHICS STATUS* option. The information is plotted to the screen in different colors to help distinguish one item from the other. The available toggles, their default settings, and the color used to plot them on the LS5 terminal are as follows:

TOGGLE	DEFAULT	COLOR
NNUMBERING - Node numbers	OFF	Yellow
ENUMBERING - Element numbers	OFF	Blue
OORDER - Optimized element ordering	OFF	Blue
MNUMBERING - Block ID (material) numbers	OFF	Red
NBOUNDARY - Nodal boundary flags	OFF	Pink
SBOUNDARY - Element side boundary flags	OFF	White

When more than one piece of information is to be displayed at the same location, the information is separated by the forward slash /, and it is ordered in the same order as the list of toggles above. For instance, if node numbers and nodal boundary flags are to be listed, the node number would be shown first followed by the list of nodal boundary flags attached to the node; all separated by forward slashes. If more than one boundary flag is attached to nodes or element sides they are all listed. The data is plotted on the mesh at the appropriate location. Node numbers are positioned beside nodes and element numbers at the center of the element. Block ID numbers are placed at the center of the element separated from the element number by a forward slash / if element numbering is also enabled.

The element numbering and element optimized order numbering toggles are mutually exclusive. Only one of the two may be plotted at a time. Thus, turning ON one automatically turns OFF the other.

4.6.5 Axis Plotting

The *AXIS* toggle has been provided to allow the display of an axis system on the plot. The default for this toggle is OFF. The axis is expanded slightly so as not to overlay any mesh falling along the extremes of the current *ZOOM* limits.

4.6.6 Hardcopy Plots

Hardcopy plots of the data are available by specifying the *HARD* option when operating in the VAX/VMS environment. This option dumps the graphic information to a file for printing on the specified hardcopy device. The file name reflects the input file specified as a runline parameter if operating in the VAX/VMS environment. If no input file was specified as a runline parameter, the file name of **FASTQ** is used. The extension to the file reflects the three letter code of the device used for hardcopy printing. For instance, if the input file name was specified on the runline as T15RM5.FSQ and the hardcopy device was QMS, then the hardcopy file would be T15RM5.QMS. All hardcopy plots for one session are placed in the same hardcopy file. If no hard copy device has been initialized (see Section 2.3.1,) **FASTQ** indicates such and continues.

4.7 OUTPUT OF MESH DATA

The generated mesh may be output in three different formats.

1. **GENESIS.** The default output when **FASTQ** is run in batch mode is the GENESIS[3] database. This output is available as the *MESH WRITE* option during interactive use.

2. **ABAQUS.** The *MESH ABAQUS* option outputs the mesh in an **ABAQUS**[4] formatted file. This option is only available in interactive mode.

3. **NASTRAN.** The *MESH NASTRAN* option outputs the mesh in a **NASTRAN**[5] formatted file. This option is only available in interactive mode.

Since **FASTQ** has been designed to facilitate the data structure supported in **GENESIS**, some of the boundary flagging and optimization techniques do not directly match the **ABAQUS** and **NASTRAN** formats. Thus, outputs to **ABAQUS** and **NASTRAN** are adjusted to use the boundary flagging and optimization incorporated in **FASTQ** optimally.

4.7.1 GENESIS Output

The **GENESIS** output format is fully supported as described in[3]. Elements are written out in sequential blocks according to block ID numbers. If optimization has been performed, the optimized element ordering is written in a mapping array as described in Section 4.5. Node and element side boundary flags are stored by flag number and element connectivity again as described in[3].

4.7.2 ABAQUS Output

The **ABAQUS**[4] output format allows the output of all supported element types and boundary conditions in **FASTQ**. The full **ABAQUS** input file is not supported in that loads and material definitions must be added directly to the file. However, the block ID attached to elements is translated to a material ID in **ABAQUS**, and thus the additional information can be added easily.

The element blocks are output as element sets (ELSET.) The name assigned to the element set is the concatenation of the letters MAT and the block ID number. The **ABAQUS** element types for each of the element sets is defaulted based on element connectivity as follows:

2 node bar:	C1D2
3 node bar:	C1D3
4 node quad:	CPE4
8 node quad:	CPE8
9 node quad:	INTER9

The user is given the option to allow the default type, or to change the element type for each element block before it is written to the file. If optimization has been enabled, the optimized element order (see Section 4.5) is output as the element number. The nodal boundary flags are output as nodal sets (*NSET*.) The name assigned to each nodal boundary flag set is the concatenation of the letters *NB* and the nodal boundary flag number. The element side boundary flags are also output as nodal sets. The name assigned to all nodes within each side boundary flag set is the concatenation of the letters *SB* and the side boundary flag number. The **ABAQUS** input format does not support a true side boundary condition as does the **GENESIS** format, and thus only the nodes of the side boundary can be output.

4.7.3 NASTRAN Output

The **NASTRAN**[5] format allows the output of all two node bar elements and four or eight node quadrilateral element types. The full **NASTRAN** data deck is

not supported in that loads and material definitions must be added directly to the file. However, the block ID attached to elements is output as the property card ID in **NASTRAN**, and thus the additional information can be added easily.

The **NASTRAN** element type for each of the element sets is defaulted based on element connectivity as follows:

2 node bar:	CBAR
3 node bar:	CBAR (middle point is ignored)
4 node quad:	CQUAD4
8 node quad:	CQUAD8
9 node quad:	CQUAD8 (center point is ignored)

The user is given the option to allow the default type, or to change the element type for each element block before it is written to the file. The nodal boundary flags are output as single point constraint (*SPC*) cards. The boundary flag number is used as the identification number of the single point constraint set. The user is asked for the degrees of freedom which are to be restrained for each nodal boundary flag set as they are output. Element side boundary flags are not supported in the **NASTRAN** output.

References

[1] R. E. Jones, *Users Manual for QMESH, A Self-Organizing Mesh Generation Program,* SLA-74-0239, Sandia National Laboratories, 1974.

[2] R. E. Jones, *QMESH: A Self-Organizing Mesh Generation Program,* SLA-73-1088, Sandia National Laboratories, 1974.

[3] L. M. Taylor, D. P. Flanagan, W. C. Mills-Curran, *The GENESIS Finite Element Mesh File Format,* SAND86-0910, Sandia National Laboratories, 1986.

[4] Hibbitt, Karlsson and Sorensen, Inc., *ABAQUS User's Manual Version 4.5,* Hibbitt, Karlsson and Sorensen, Inc., 1984.

[5] MacNeal-Schwendler Corporation, *MSC/NASTRAN User's Manual Version 65,* ISSN 0741-8019, The MacNeal-Schwendler Corporation, 1985.

[6] D. P. Flanagan, *Dual Device, Shared Library, and Switchable Device Systems for the SVDI,* SAND88-1341, Sandia National Laboratories, 1988.

[7] C. B. Selleck, M. P. Sears, *PLT - A Multipurpose Plotting Package,* SAND83-0627, Sandia National Laboratories, 1983.

[8] D. P. Flanagan, W. C. Mills-Curran, L. M. Taylor, *SUPES A Software Utilities Package for the Engineering Sciences,* SAND86-0911, Sandia National Laboratories, 1986.

[9] T. D. Blacker, J. L. Mitchiner, L. R. Phillips, Y. T. Lin, *Knowledge System Approach to Automated Two-Dimensional Quadrilateral Mesh Generation,* Proceedings of 1988 ASME International Computers in Engineering Conference, July, 1988.

[10] W. A. Cook, W. R. Oakes, *Mapping Methods for Generating Three-Dimensional Meshes,* Comp. Mech. Eng., 1, pp. 67-72, 1983.

[11] W. J. Gordon, C. A. Hall, *Construction of Curvilinear Co-Ordinate Systems and Applications to Mesh Generation,* Int J. for Numerical Methods in Engineering, Vol. 7, pp. 461-477, 1973.

A. FASTQ File Format

FASTQ files are read in free format using the 1520 Free Field Reader[8]. This format supports continuation lines if the information exceeds the line length. The continuation is indicated by placing an asterisk (*) at the end of the line to be continued, signifying the next line is to be considered part of the current line. Data delimiters include the space, the comma (,), the semicolon (;), and the equal sign (=). Blank fields or place holders are indicated by two consecutive delimiters other than the space. Comments can be added to any line with a dollar sign ($) preceding the comment. Any information on a line beyond the dollar sign is ignored by **FASTQ**.

No specific cards or ordering of cards is required in the FASTQ file. However, any cards after the EXIT card will be ignored. Also, the FACTOR and INTERVAL cards can only have meaning if placed after the lines and sides to which the factors or intervals are to be assigned.

A.1 Title Card

The title card contains 1 field:

1 The string TITLE to indicate that the next card in the deck is to be the title.

Example:

```
TITLE
MC3831 HOUSING - Z GRAVITY LOADING - T. Blacker 2/4/86
```

A.2 Point Card

The point card has 4 fields:

1 The string POINT to identify the card

2 The point number

3 The x coordinate

4 The y coordinate

Examples:

```
POINT 5 6.128E+00 9.632E-03
POINT,8,.5,2.3
```

A.3 Line Card

The line card has 8 fields:

1 The string LINE to identify the card
2 The line number
3 The line type (i.e. STR,CIRC,CIRR,CIRM,CORN,PARA)
4 Point 1 of the line
5 Point 2 of the line
6 Point 3 of the line (enter a zero or a blank field for straight lines)
7 Number of intervals (optional)
8 Factor to increment interval size (optional)

Examples:

```
LINE 12 CIRC 14 12 13 2 1.0000
LINE,123,CIRM,23,48,1,3
LINE 113 STR 14 15 0 9 .5
LINE,3,STR,4,1,,0,.2
```

A.4 Side Card

The side card may have as many fields as needed to complete the definition:

1 The string SIDE to identify the card
2 The side number
3 Line 1 for the side
4 Line 2 for the side
⋮
N+2 Line n for the side

Examples:

```
SIDE 1 12 10
SIDE,1,3,4,18,42
```

A.5 Barset Card

The barset card may have as many fields as needed to complete the definition:

1 The string BARSET to identify the card

2 The barset number

3 The block ID number

4 The inside point (optional - enter a zero or blank field if not being used.)

5 Line 1 for the barset

6 Line 2 for the barset

⋮

N+4 Line n for the barset

Examples:

```
BARSET 6 11 35 3 27 1
BARSET,5,1,,1,3,8,13,9
```

A.6 Region Card

The region card may have as many fields as needed to complete the definition:

1 The string REGION to identify the card

2 The region number

3 The block ID number

4 Side (or line) 1 for the region

5 Side (or line) 2 for the region

⋮

N+3 Side (or line) n for the region

Note: Any lines used instead of sides to define the region must be negated.

Examples:

```
REGION 6 1 -5 -53 7 -51
REGION,2, 1 , 2 , -66 , -41 , 5
```

A.7 Scheme Card

The scheme card has 3 fields:

1 The string SCHEME to identify the card

2 The region number this scheme applies to (a zero indicates use as the default scheme)

3 The scheme as defined in Section 4.3.2.

Example:

```
SCHEME 0 MS
SCHEME,3,NS(DRS)P
```

A.8 Interval Card

The interval card may have as many fields as needed to complete the definition:

1 The string INT[24] to identify the card

2 The number of intervals to be assigned to the given lines or sides

3 Line (or side) 1 to which this number of intervals is to be assigned

4 Line (or side) 2 to which this number of intervals is to be assigned

⋮

N+2 Line (or side) n to which this number of intervals is to be assigned

Note: Any sides to be included in the interval card must be negated. The intervals are actually stored as a line attribute, and thus each side is expanded and the interval placed on the lines in each side rather than remaining as an attribute of the side. This requires that the effected line and side definitions must precede the interval card.

Example:

[24] The string INTERVAL may be spelled out further if desired.

```
INTERVALS 12 3 13 -8 7
INT,8,-1
```

A.9 Factor Card

The factor card may have as many fields as needed to complete the definition:

1 The string FAC[25] to identify the card

2 The factor to be applied to the given lines or sides

3 Line (or side) 1 to which this factor is to be applied

4 Line (or side) 2 to which this factor is to be applied

⋮

N+2 Line (or side) *n* to which this factor is to be applied

Note: Any sides to be included in the factor card must be negated. The factors are actually stored as a line attribute, and thus each side is expanded and the factor placed on the lines in each side rather than remaining as an attribute of the side. This requires that the effected line and side definitions must precede the factor card.

Example:

```
FACTOR 1.2 1 3 -2 8
FAC,.7,-2
```

A.10 Body Card

The body card may have as many fields as needed to complete the definition:

1 The string BODY to identify the card

2 Region (or barset) 1 for the body

3 Region (or barset) 2 for the body

⋮

N+1 Region (or barset) *n* for the body

[25]The string FACTOR may be spelled out further if desired.

Note: Any barsets to be included in the body must be negated.

Example:

```
BODY 1 2 5 7 9 14
BODY,3,4,-108,150,-1033
```

A.11 Point Boundary Flag Card

The point boundary flag card may have as many fields as needed to complete the definition:

1 The string POINBC to identify the card

2 The flag number

3 Point 1 to which the flag applies

4 Point 2 to which the flag applies

⋮

N+3 Point n to which the flag applies

Note: Point Boundary Flag Cards with the same flag number are combined in the FASTQ definition of the flag.

Examples:

```
POINBC 105 68
POINBC,1,2,5,10,6
```

A.12 Line Boundary Flag Card

The line boundary flag card may have as many fields as needed to complete the definition:

1 The string LINEBC to identify the card

2 The flag number

3 Line 1 to which the flag applies

4 Line 2 to which the flag applies

⋮

N+3 Line n to which the flag applies

Note: Line Boundary Flag Cards with the same flag number are combined in the FASTQ definition of the flag.

Examples:

```
LINEBC 105 68
LINEBC,1,2,5,10,6
```

A.13 Side Boundary Flag Card

The side boundary flag card may have as many fields as needed to complete the definition:

1 The string SIDEBC to identify the card

2 The flag number

3 Line 1 to which the flag applies

4 Line 2 to which the flag applies

⋮

N+3 Line n to which the flag applies

Note: Side Boundary Flag Cards with the same flag number are combined in the FASTQ definition of the flag.

Examples:

```
SIDEBC 105 68
SIDEBC,1,2,5,10,6
```

A.14 Renumbering Optimization Card

The renumbering optimization card may have as many fields as needed to complete the definition. However, only one card (80 characters) can be used.

1 The string RENUM to identify the card

2 The optional renumbering optimization control (P-L-P, NODE, X-Y)

If a renumbering optimization control has been specified the remaining fields have meaning based on the control option chosen. For the P-L-P option they are defined as:

3 First point whose node is to be added to the seed list

4 First line whose nodes are to be added to the seed list

5 Second point whose node is to be added to the seed list

6 Second line whose nodes are to be added to the seed list

⋮

N Last line or point whose node(s) are to be added to the seed list

For the X-Y option they are defined as:

3 X location

4 Y location (the closest node to this position is added to the seed list)

For the NODE option they are defined as:

3 First NUID (Node Unique ID) of a node to be added to the seed list

4 Second NUID (Node Unique ID) of a node to be added to the seed list

⋮

N The last NUID of a node to be added to the seed list

Examples:

```
RENUM
RENUM,P-L-P 1 20 2 14 3 8 3
RENUM NODE 1 330002 330005 3
RENUM X-Y 3.45 2.03
```

A.15 Exit Card

The exit card has 1 field:

1 The string EXIT to signal an exit from reading of the data

Note: No data will be read past this card.

B. DIGITIZER MOUSE BUTTON REFERENCE

1	Enter Point
01	Get Closest Point
C1	New Point At Closest Point
2	Straight Line To New Point
02	Straight Line To Closest Point
C2	Straight Line To New Point Over Closest Point
3	Ccw Arc To New Point
03	Ccw Arc To Closest Point
C3	Ccw Arc To New Point Over Closest Point
4	Cw Arc To New Point
04	Cw Arc To Closest Point
C4	Cw Arc To New Point Over Closest Point
5	Show Current Location
6	Dissect Closest Line Perpendicularly
A	Enter Enclosing Region
D	Delete Closest Point And Definitions
E	Exit

C. EXAMPLE MESH PROBLEMS

The example problems demonstrate some of the meshing capabilities of FASTQ. Each example contains a brief description of the problem, a listing of the FASTQ file used to generate the example, and a plot showing:

1. The initial geometry of the problem.
2. The decomposition of the geometry into meshable regions with line interval assignments and region schemes displayed.
3. The resulting mesh for the problem.

C.1 Example A

Example A shows the use of three of the initial meshing primitives. The geometry consists of two materials as shown in Figure 11. The decomposition consists of two semicircular regions (upper portion), one triangular region (the lower left portion,) and several general rectangular regions. The mesh was generated with a uniform size over the entire geometry. The FASTQ file listing for example A is:

```
TITLE
  Example A
POINT      1      8.0000000000E+01      0.0000000000E+00
POINT      2      8.0000000000E+01      2.0000000000E+01
POINT      3      7.0000000000E+01      2.0000000000E+01
POINT      4      6.0000000000E+01      3.0000000000E+01
POINT      5      7.0000000000E+01      3.0000000000E+01
POINT      6      7.0000000000E+01      5.0000000000E+01
POINT      7      5.0000000000E+01      5.0000000000E+01
POINT      8      5.0000000000E+01      3.7000000000E+01
POINT      9      4.0000000000E+01      3.7000000000E+01
POINT     10      4.0000000000E+01      6.0000000000E+01
POINT     11      1.0000000000E+01      6.0000000000E+01
POINT     12      1.0000000000E+01      3.3299999237E+01
POINT     13      4.3299999237E+01      0.0000000000E+00
```

```
POINT    14      9.0000000000E+01      2.0000000000E+01
POINT    15      9.0000000000E+01      7.0000000000E+01
POINT    16      6.0000000000E+01      7.0000000000E+01
POINT    17      6.0000000000E+01      8.7699996948E+01
POINT    18      6.0000000000E+01      1.3330000305E+02
POINT    19      4.0000000000E+01      1.3330000305E+02
POINT    20      4.0000000000E+01      6.7699996948E+01
POINT    21      4.3299999237E+01      3.3299999237E+01
POINT    22      4.0000000000E+01      1.0000000000E+02
POINT    23      6.0000000000E+01      1.1000000000E+02
POINT    24      7.0000000000E+01      3.7000000000E+01
POINT    25      7.0000000000E+01      0.0000000000E+00
POINT    26      6.0000000000E+01      5.0000000000E+01
POINT    27      5.0000000000E+01      6.8849998474E+01
POINT    28      9.0000000000E+01      5.0000000000E+01
LINE      1   STR     1     2     0     5  1.0000
LINE      2   STR     2     3     0     3  1.0000
LINE      3   STR     3     6     0     7  1.0000
LINE      4   STR     6     7     0     5  1.0000
LINE      5   STR     7     8     0     3  1.0000
LINE      6   STR     8     9     0     3  1.0000
LINE      7   STR     9    10     0     7  1.0000
LINE      8   STR    10    11     0     7  1.0000
LINE      9   STR    11    12     0     7  1.0000
LINE     10  CIRC    12    13    21    13  1.0000
LINE     12   STR     2    14     0     2  1.0000
LINE     14   STR    15    16     0     7  1.0000
LINE     15   STR    16    17     0     4  1.0000
LINE     16  CIRC    17    18    23    18  1.0000
LINE     17   STR    18    19     0     5  1.0000
LINE     18  CIRC    19    20    22    27  1.0000
LINE     19   STR    20    10     0     2  1.0000
LINE     22   STR     8    24     0     5  1.0000
LINE     23   STR     3    24     0     5  1.0000
LINE     24   STR     6    24     0     3  1.0000
LINE     25   STR     9    12     0     7  1.0000
LINE     27   STR     3    25     0     5  1.0000
LINE     28   STR    13    25     0     6  1.0000
LINE     29   STR     1    25     0     3  1.0000
LINE     30   STR    16    20     0    10  1.0000
LINE     34   STR    17    18     0    12  1.0000
```

```
LINE     36   STR     7    27     0     6  1.0000
LINE     37   STR    16    27     0     3  1.0000
LINE     38   STR    20    27     0     3  1.0000
LINE     39   STR     6    28     0     5  1.0000
LINE     40   STR    14    28     0     8  1.0000
LINE     41   STR    15    28     0     6  1.0000
REGION  115     2   -22   -24    -4    -5
REGION  116     2    -7    -8    -9   -25
REGION  117     2   -23   -22    -6   -25   -10   -28   -27
REGION  118     2   -27   -29    -1    -2
REGION  120     6   -16   -34
REGION  121     6   -38   -37   -15   -34   -17   -18
REGION  122     6   -36   -38   -19    -7    -6    -5
REGION  123     6    -2   -12   -40   -39   -24   -23
REGION  124     6   -39   -41   -14   -37   -36    -4
SCHEME  117 T6S
SCHEME  120 C6S
SCHEME  121 C6S
SCHEME    0 M
BODY    115   116   117   118   122   123   124   120   121
EXIT
```

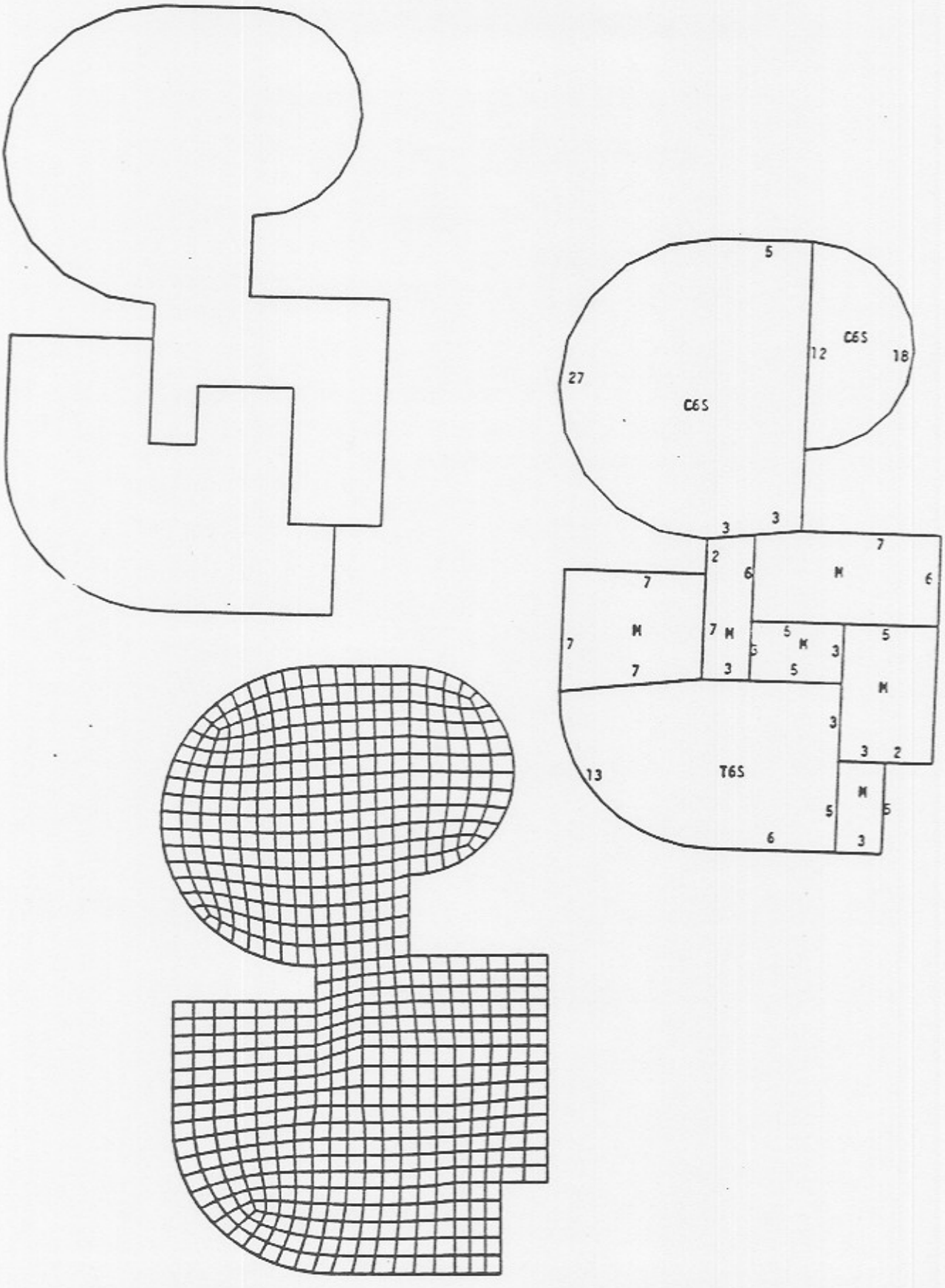


Figure 11. EXAMPLE A - GEOMETRY, DECOMPOSTION, AND MESH

C.2 Example B

Example B shows the use of the transition and the circular initial meshing primitives. The geometry consists of three parts as shown in Figure 12. The decomposition consists of a circular region at the center, two transition regions on either side of the circular region, and several general rectangular regions. The mesh was generated with a fine mesh around the center circular region and a uniform size over the remaining geometry. The FASTQ file listing for example B is:

```
TITLE
 Example B
POINT 1 14.25 0.0
POINT 2 14.25 1.31
POINT 3 13.25 0.0
POINT 4 13.25 0.31
POINT 5 11.34 6.34
POINT 6 11.34 1.31
POINT 7 10.65 6.34
POINT 8 10.65 0.31
POINT 9 10.65 1.88
POINT 10 10.65 4.19
POINT 11 10.35 4.19
POINT 12 10.35 2.26
POINT 13 8.56 1.88
POINT 14 8.56 2.26
POINT 15 10.25 3.255
POINT 16 10.15 2.315
POINT 17 8.845 2.315
POINT 18 9.132 4.195
POINT 19 8.845 4.195
POINT 20 10.15 4.295
POINT 21 9.132 4.575
POINT 22 10.54 3.548
POINT 23 10.54 2.962
POINT 24 10.35 3.783
POINT 25 10.41 3.658
POINT 26 10.35 2.703
POINT 27 10.41 2.828
POINT 28 10.15 3.783
POINT 29 10.09 3.658
```

```
POINT 30 9.96 3.548
POINT 31 9.96 2.962
POINT 32 10.09 2.828
POINT 33 10.15 2.703
POINT 34 10.35 4.295
POINT 35 10.35 4.575
POINT 36 8.845 4.575
POINT 37 10.5 6.759
POINT 38 11.13 6.759
POINT 39 11.13 7.509
POINT 40 8.845 7.509
POINT 41 10.47914 3.768088
POINT 42 10.47914 2.717912
POINT 43 10.02086 3.768088
POINT 44 10.02086 2.717912
POINT 45 10.5171580049809l5d0 7.5089998245239l6d0
POINT 46 14.25d0 0.309984335210950l5d0
POINT 47 11.3400001525878l9d0 1.879989205682840l7d0
POINT 48 11.3400001525878l9d0 2.570000360721345l5d0
POINT 49 10.6499996185302l3d0 1.310010772545643l6d0
POINT 50 10.3499944453632l2d0 1.879999995231628l4d0
POINT 51 11.3400001525878l9d0 4.190011015260621l0
LINE 1 STR 3 1 0 4 1.0
LINE 3 STR 4 3 0 1 1.0
LINE 4 STR 2 6 0 12 1.0
LINE 5 STR 4 8 0 11 1.0
LINE 8 STR 24 11 0 2 1.0
LINE 9 STR 10 11 0 1 1.0
LINE 10 STR 5 7 0 3 1.0
LINE 11 STR 7 10 0 9 1.0
LINE 12 STR 12 14 0 7 1.0
LINE 13 STR 14 13 0 2 1.0
LINE 15 STR 33 16 0 2 1.0
LINE 16 STR 17 16 0 5 1.0
LINE 17 STR 18 19 0 1 1.0
LINE 18 STR 19 17 0 8 1.0
LINE 19 STR 21 18 0 2 1.0
LINE 20 CIRC 23 22 15 8 1.0
LINE 21 CIRC 22 30 15 3 1.0
LINE 22 STR 22 25 0 1 1.0
LINE 23 CIRC 24 25 41 1 1.0
```

```
LINE 24 CIRC 31 23 15 3 1.0
LINE 25 CIRC 30 31 15 8 1.0
LINE 26 CIRC 27 26 42 1 1.0
LINE 27 STR 27 23 0 1 1.0
LINE 28 STR 20 28 0 2 1.0
LINE 29 CIRC 29 28 43 1 1.0
LINE 30 STR 29 30 0 1 1.0
LINE 31 STR 31 32 0 1 1.0
LINE 32 CIRC 33 32 44 1 1.0
LINE 33 STR 12 26 0 2 1.0
LINE 34 STR 20 34 0 1 1.0
LINE 35 STR 34 35 0 2 1.0
LINE 36 STR 21 36 0 1 1.0
LINE 37 STR 35 37 0 9 1.0
LINE 38 STR 37 38 0 3 1.0
LINE 39 STR 38 39 0 3 1.0
LINE 41 STR 40 36 0 12 1.0
LINE 42 STR 18 20 0 4 1.0
LINE 43 STR 21 35 0 5 1.0
LINE 44 STR 37 45 0 3 1.0
LINE 45 STR 39 45 0 3 1.0
LINE 46 STR 40 45 0 6 1.0
LINE 47 STR 4 46 0 4 1.0
LINE 48 STR 1 46 0 1 1.0
LINE 49 STR 2 46 0 4 1.0
LINE 52 STR 9 47 0 3 1.0
LINE 58 STR 6 49 0 3 1.0
LINE 59 STR 9 49 0 2 1.0
LINE 60 STR 8 49 0 4 1.0
LINE 61 STR 6 47 0 2 1.0
LINE 63 STR 12 50 0 2 1.0
LINE 64 STR 13 50 0 7 1.0
LINE 65 STR 9 50 0 1 1.0
LINE 67 STR 10 51 0 3 1.0
LINE 68 STR 47 51 0 10 1.0
LINE 69 STR 5 51 0 9 1.0
REGION 785 4 -25 -24 -20 -21
REGION 787 6 -17 -18 -16 -15 -32 -31 -25 -30 -29 -28 -42
REGION 788 6 -19 -42 -34 -35 -43
REGION 789 6 -38 -39 -45 -44
REGION 790 6 -44 -46 -41 -36 -43 -37
```

```
REGION 792 2 -3 -1 -48 -47
REGION 793 2 -58 -60 -5 -47 -49 -4
REGION 794 2 -59 -58 -61 -52
REGION 795 2 -12 -13 -64 -63
REGION 796 2 -9 -8 -23 -22 -20 -27 -26 -33 -63 -65 -52 -68 -67
REGION 797 2 -67 -69 -10 -11
SCHEME 785 NSNSN6S
SCHEME 787 B6S
SCHEME 796 B6S
SCHEME   0 M
BODY 785 787 788 789 790 792 793 794 795 796 797
EXIT
```

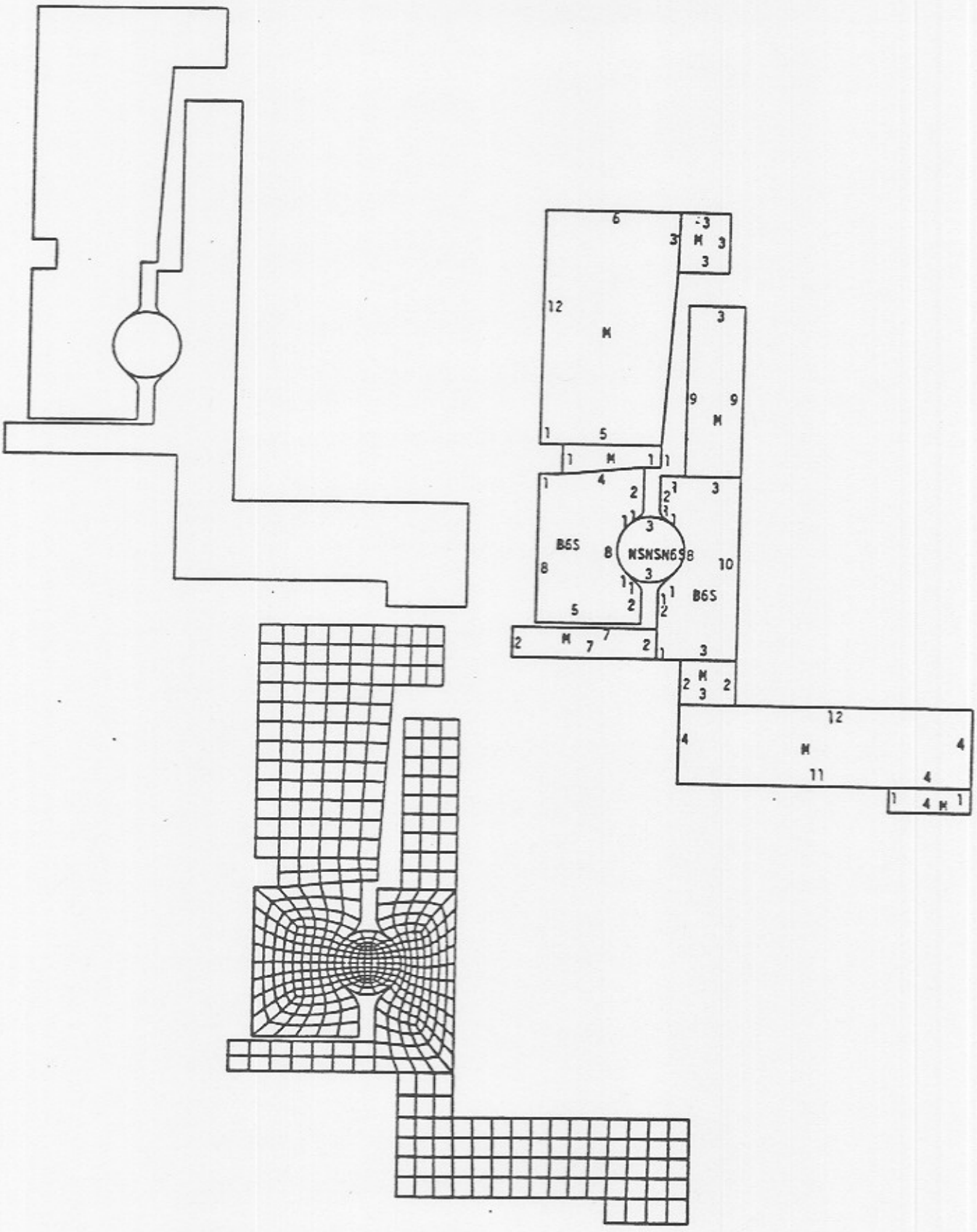


Figure 12. EXAMPLE B - GEOMETRY, DECOMPOSTION, AND MESH

C.3 Example C

Example C shows the use of the pentagon and circular initial meshing primitives. The geometry consists of a material with two holes, one of which has been filled with another material as shown in Figure 13. The decomposition consists of all pentagon regions for the base material, and a circular region for the material added. The mesh was generated with a fine mesh in the added material, and a uniform size over the remaining geometry. The FASTQ file listing for example C is:

```
TITLE
  Example C
POINT     1     -5.0000000000E-01       0.0000000000E+00
POINT     2      1.7999999523E+00       0.0000000000E+00
POINT     3      2.0000000000E+00       2.0000000000E+00
POINT     4     -1.0000000000E+00       2.5000000000E+00
POINT     6      8.5000002384E-01       2.5000000000E-01
POINT     7      8.5000002384E-01       8.5000002384E-01
POINT     8      2.0000000000E+00       1.1250000000E+01
POINT     9     -2.0000000298E-01       1.7999999523E+00
POINT    10     -2.0000000298E-01       1.5000000000E+00
POINT    11     -9.7352996469E-02       2.2409157753E+00
POINT    12     -1.3197770715E-01       2.0921864510E+00
POINT    13      8.5000002384E-01       0.0000000000E+00
POINT    14      1.4470223188E+00       7.9029798508E-01
POINT    15      1.8747524023E+00       7.4752479792E-01
POINT    16      9.8335999250E-01       2.0560386181E+00
POINT    17      9.1611999273E-01       1.4463456869E+00
POINT    18     -4.9417418242E-01       1.7411651611E+00
POINT    19     -8.3461540937E-01       1.6730768681E+00
POINT    20      4.0507856011E-01       1.2525479794E+00
POINT    21      2.2193882614E-02       1.5984314680E+00
POINT    22      9.8191998899E-02       1.7671138048E+00
POINT    23      9.4124209881E-01       1.6741379499E+00
POINT    24     -2.0000000298E-01       0.0000000000E+00
POINT    25      2.5000000000E-01       8.5000002384E-01
POINT    26     -2.0000000298E-01       8.5000002384E-01
LINE     10   CIRC     4     11      8      5  1.0000
LINE     11    STR    11     12      0      2  1.0000
LINE     13    STR     2     13      0      9  1.0000
LINE     14    STR    13      6      0      2  1.0000
```

```
LINE      16  CIRC     6    14     7    11  1.0000
LINE      17   STR     2    15     0     6  1.0000
LINE      18   STR     3    15     0    10  1.0000
LINE      19   STR    14    15     0     4  1.0000
LINE      20  CIRC    16     3     8     8  1.0000
LINE      21  CIRC    11    16     8     9  1.0000
LINE      23  CIRC    14    17     7    13  1.0000
LINE      26  CIRC    12    18     9     7  1.0000
LINE      27   STR     4    19     0     5  1.0000
LINE      28   STR     1    19     0    14  1.0000
LINE      29   STR    18    19     0     3  1.0000
LINE      31  CIRC    17    20     7     6  1.0000
LINE      34   STR    20    21     0     3  1.0000
LINE      35  CIRC    22    12     9     5  1.0000
LINE      36  CIRC    21    22     9     2  1.0000
LINE      37   STR    17    23     0     2  1.0000
LINE      38   STR    16    23     0     3  1.0000
LINE      39   STR    22    23     0     7  1.0000
LINE      40  CIRC    10    21     9     3  1.0000
LINE      41  CIRC    18    10     9     5  1.0000
LINE      42   STR     1    24     0     3  1.0000
LINE      43   STR    13    24     0     9  1.0000
LINE      45  CIRC    25     6     7    12  1.0000
LINE      46  CIRC    20    25     7     6  1.0000
LINE      47   STR    10    26     0     6  1.0000
LINE      48   STR    24    26     0     7  1.0000
LINE      49   STR    25    26     0     4  1.0000
REGION     4     2   -16   -14   -13   -17   -19
REGION     5     2   -38   -37   -23   -19   -18   -20
REGION     6     2   -26   -11   -10   -27   -29
REGION     7     2   -39   -38   -21   -11   -35
REGION     8     2   -36   -34   -31   -37   -39
REGION     9     2   -41   -29   -28   -42   -48   -47
REGION    10     2   -46   -34   -40   -47   -49
REGION    11     2   -49   -48   -43   -14   -45
REGION    12     4   -16   -23   -31   -46   -45
SCHEME    12 NSNSN6S
SCHEME     0 U6S
BODY       4     5     6     7     8     9    10    11    12
EXIT
```

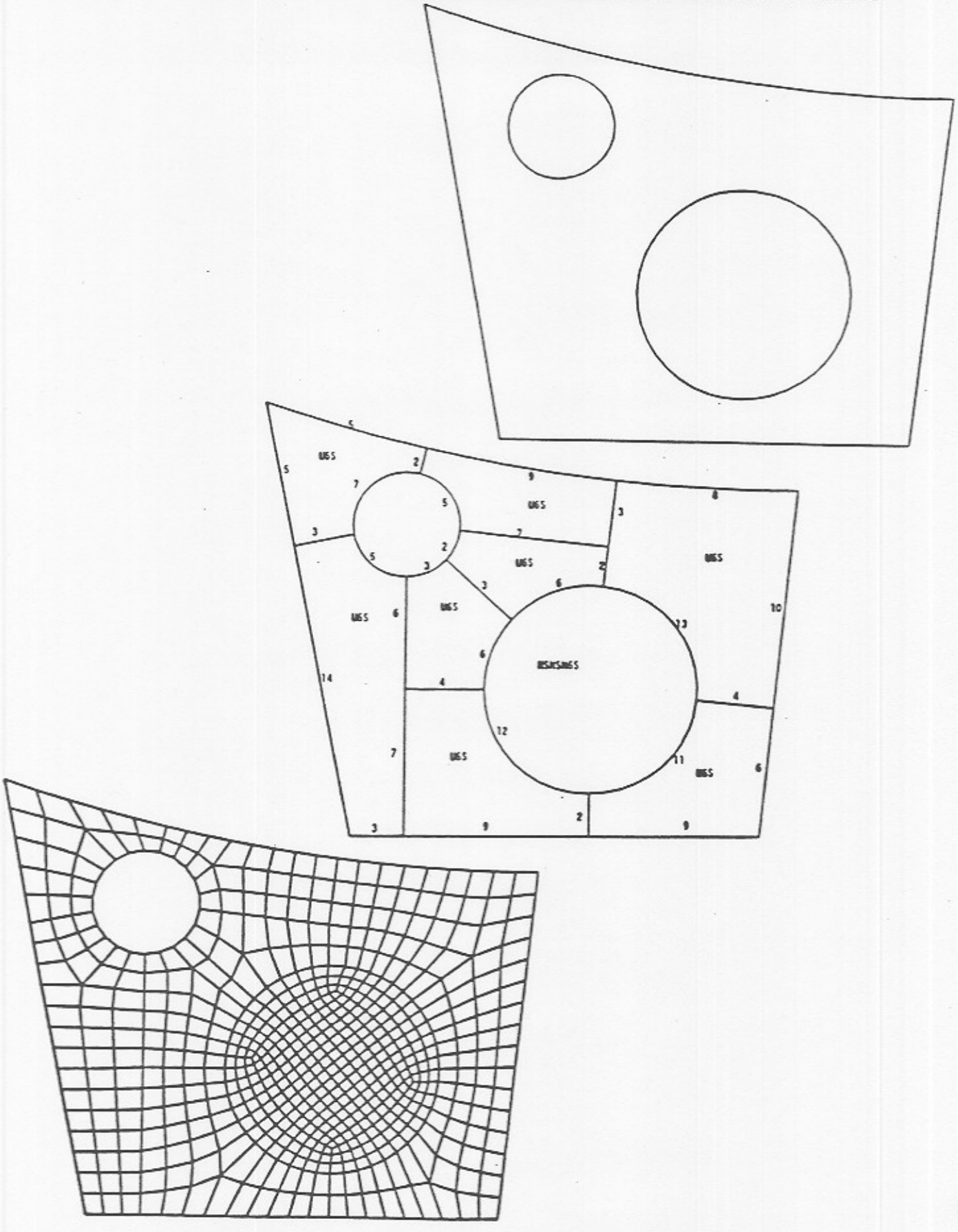
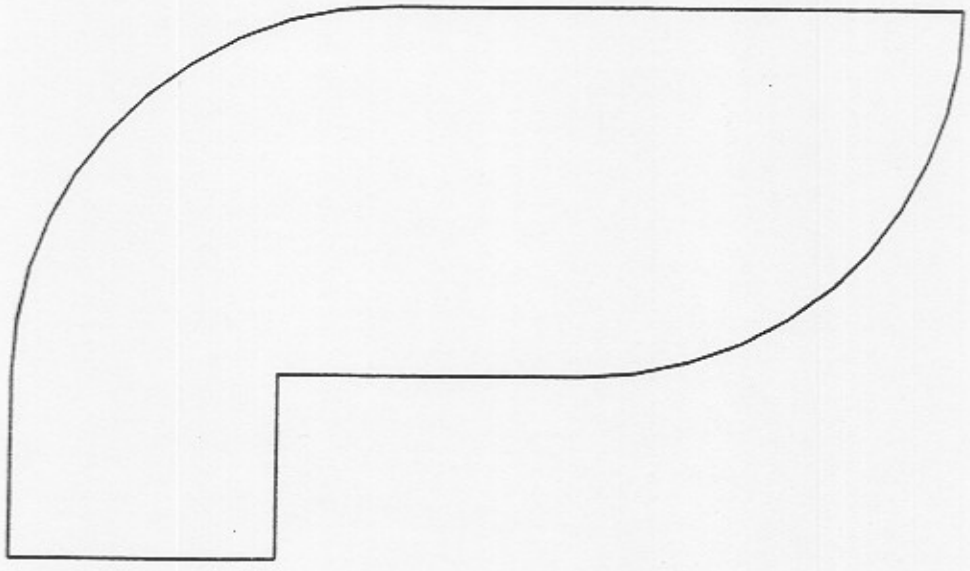


Figure 13. EXAMPLE C - GEOMETRY, DECOMPOSTION, AND MESH

4. AN EXAMPLE INTERACTIVE SESSION WITH FASTQ

The example interactive session is used to show the interactive definition, correction, plotting, and meshing capabilities of **FASTQ** for the relatively simple geometry shown below:



It is assumed that the user is running in the VAX/VMS environment, and that he/she will define the geometry using *KEYIN*. **FASTQ** prompts and output are printed in *SLANTED TEXT*, and the user response is given in SANS SERIF TEXT.

The user first enters **FASTQ** with the command:

```
$ FASTQ "" LS5 QMS
```

which designates that there is no initial file (the double quotes ""), that the plotting device is LS5 (Lear-Siegler terminal), and that the hardcopy device is QMS (the QMS Laser Printer.) **FASTQ** responds with the prompt:

WELCOME TO FASTQ
DATE: 05/04/88
TIME: 11:41:29
VERSION: FASTQ 1.2
ENTER OPTION:

The user, who knows the geometry must first be defined responds with:

ENTER OPTION: K

indicating the *KEYIN* option. He next enters the point data by responding to the **FASTQ** prompt:

ENTER KEYIN OPTION:

with:

ENTER KEYIN OPTION: P

to begin entering points. The prompt/response cycle for entering the points is as follows:

ENTER POINT DATA IN THE FOLLOWING FORMAT:
[POINT NO., X, Y]
HIT RETURN TO END INPUT
> 1
> 2,7
> 3,7,5
> 4,10,5
> 5,15,5
> 6,25,15
> 7,15,15
> 8,10,15
> 9,,5
> *<CR>*

FASTQ returns to the *KEYIN* prompt and the user next inputs the straight lines with the following prompt/response cycle:

ENTER KEYIN OPTION: LI ST
ENTER STRAIGHT LINE DATA IN THE FOLLOWING FORMAT:
[LINE NO., POINT 1, POINT 2, NO. INTERVALS, FACTOR]
HIT RETURN TO END INPUT
> 1,9,1
> 3,8,6
> 5,3,5
> 6,3,2
> 7,2,1
> 8,9,3
> *<CR>*

The user has seen the need to decompose the object into 2 regions with line 8 which separates the rectangular protrusion from the rest of the geometry. Note that the user decided not to input intervals with the line definitions, but only after the geometry was completely defined and validated. **FASTQ** again returns to the *KEYIN* prompt and the user next inputs the arcs. He cannot remember the key word for arcs, so he lets **FASTQ** prompt him in the following manner:

ENTER KEYIN OPTION: LI

THE FOLLOWING LINE TYPES ARE AVAILABLE:
*S*TRAIGHT = STRAIGHT LINE*
*CI*RCULAR = CIRCULAR CCW ARC ABOUT A CENTER*
*3*CIRCULAR= CIRCULAR ARC WITH 3RD ARC POINT*
*R*CIRCULAR= CIRCULAR ARC WITH RADIUS*
*CO*RNER = 2 LINE SEGMENTS JOINED*
*P*ARABOLA = PARABOLIC SHAPED LINE*
WHICH LINE TYPE WOULD YOU LIKE TO ENTER: CI
NOTE: IF A CW ARC IS DESIRED, ENTER
THE CENTER POINT AS NEGATIVE
ENTER CIRCULAR ARC LINE DATA IN THE FOLLOWING FORMAT:
[LINE NO., POINT 1, POINT 2, CENTER, NO. INTERVALS, FACTOR]
HIT RETURN TO END INPUT
> 2,8,9,4
> 4,6,5,7
> *<CR>*

Again, the user has chosen to wait on assignment of intervals. He next defines the region in the following interaction:

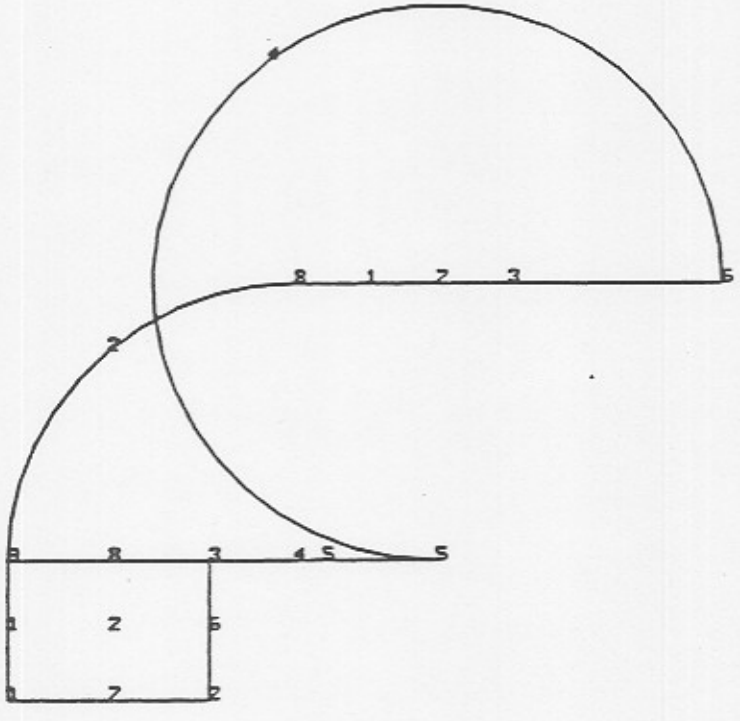
ENTER KEYIN OPTION: R

ARE YOU USING SIDES IN DEFINING REGIONS? N
ENTER REGION DATA IN THE FOLLOWING FORMAT:
[REGION NO., MATERIAL NO., LINE 1, LINE 2, ..., LINE N]
HIT RETURN TO END INPUT
> 1,1,8,5,4,3,2
> 2,1,1,8,6,7
> *<CR>*

The user has completed the basic geometry definitions, and now wishes to verify the data graphically. He is still in the *KEYIN* option, and thus the interaction is:

ENTER KEYIN OPTION: ,G P

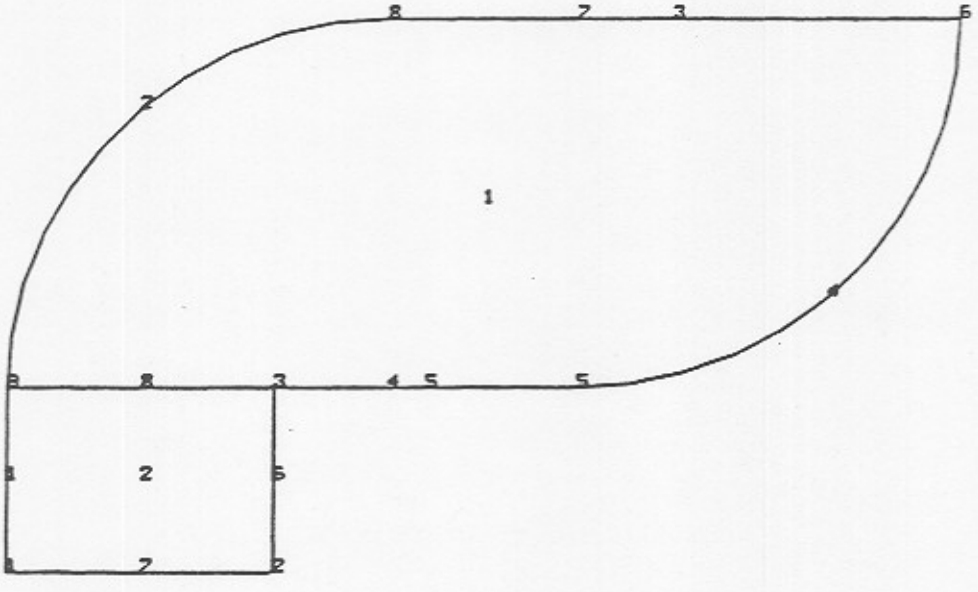
This response contains an initial blank field (the comma) which gets the user back to the main level, then chooses the *GRAPHICS* option (the *G*,) and asks for plotting (the *P*.) The following plot is shown on the screen.



It is evident that line 3 has been defined as a counterclockwise line instead of a clockwise line, and thus the user returns to the *KEYIN* option to fix the error and replot the geometry in the following interaction:

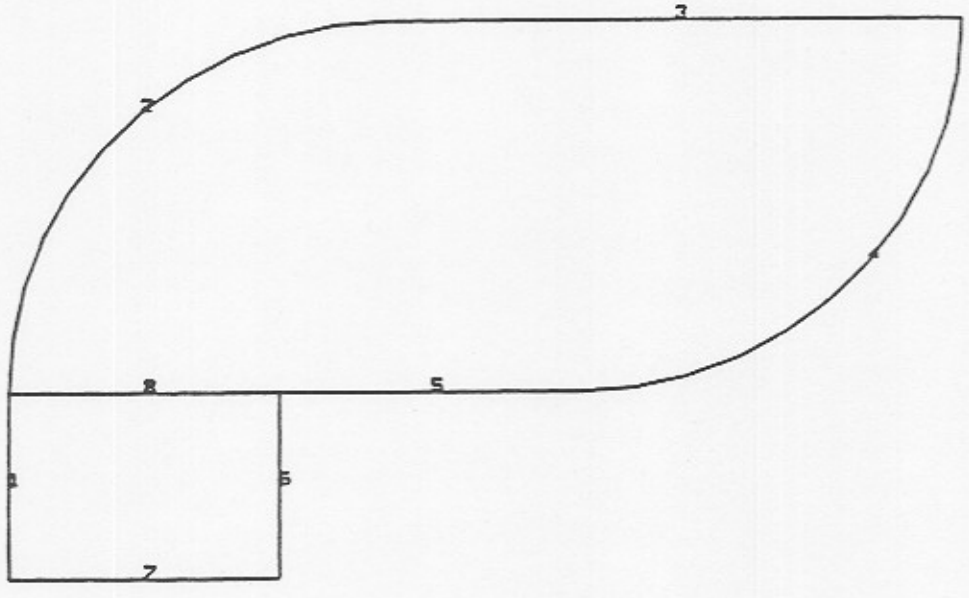
ENTER GRAPHICS OPTION: ,K
ENTER KEYIN OPTION: LI CIRC
NOTE: IF A CW ARC IS DESIRED, ENTER
THE CENTER POINT AS NEGATIVE
ENTER CIRCULAR ARC LINE DATA IN THE FOLLOWING FORMAT:
[LINE NO., POINT 1, POINT 2, CENTER, NO. INTERVALS, FACTOR]
HIT RETURN TO END INPUT
> 4,6,5,-7
> *<CR>*

ENTER KEYIN OPTION: ,G P



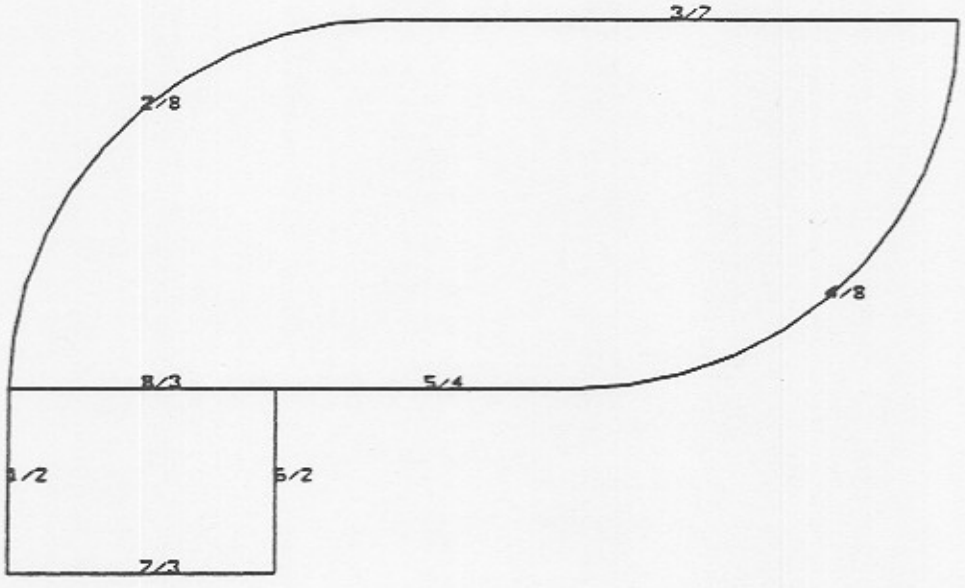
Now, the geometry looks correct, and the user wishes to input intervals and verify the interval assignment graphically. The user first turns region and point numbers off, and plots the geometry with only line numbers left on for clarity. He then inputs the interval and verifies the input graphically in the following manner:

ENTER GRAPHICS OPTION: PO RE P
POINT LABELS - OFF
REGION LABELS - OFF



ENTER GRAPHICS OPTION: II
ENTER LINE INTERVALS IN THE FOLLOWING FORMAT:
[LINE NO. (OR NEG SIDE NO.), INTERVALS]
HIT RETURN TO END INPUT
> 1,2
> 6,2
> 8,3
> 7,3
> 5,4
> 3,7
> 4,8
> 2,8
> *<CR>*

ENTER GRAPHICS OPTION: I P
INTERVAL LABELS - ON



Satisfied with the definitions, the user turns to meshing the geometry. Region 2 is definitely a rectangle primitive. However, the shape of region 1 is not one he quickly recognizes as a primitive, but the user wishes to experiment a little with various meshing options for this region. If unsuccessful, he can always use the digitizer to subdivide region 1 as necessary. Since the meshing of region 1 will require interactive processing, the *STEP* option is chosen. The meshing begins with the interaction:

ENTER GRAPHICS OPTION: ,M ST

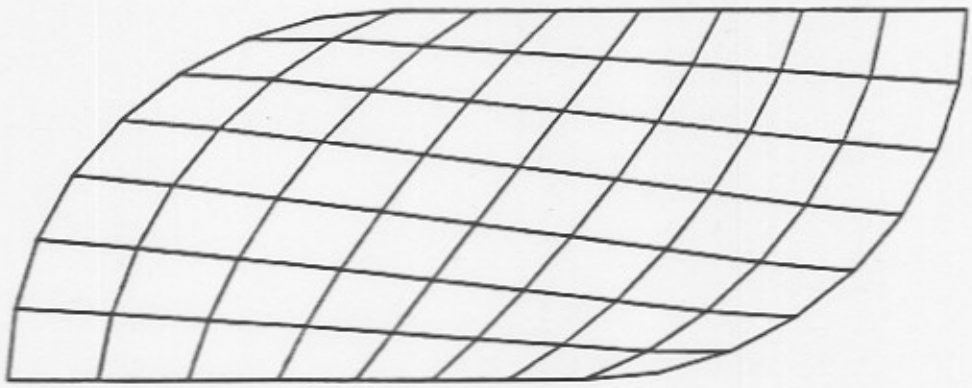
INITIAL CHECK BEGUN FOR REGION: 1
*** WARNING: CORNER(S) OF THE REGION HAVE ***
*** LARGE ANGLES (> 150 DEGREES.) ***
*** POORLY FORMED MESH MAY RESULT ***
INITIAL CHECK BEGUN FOR REGION: 2
STEP PROCESS REGIONS I1 THROUGH I2:
> 1

NOW PROCESSING REGION: 1

INITIAL MESH DEFINED USING THIS SCHEME:
M

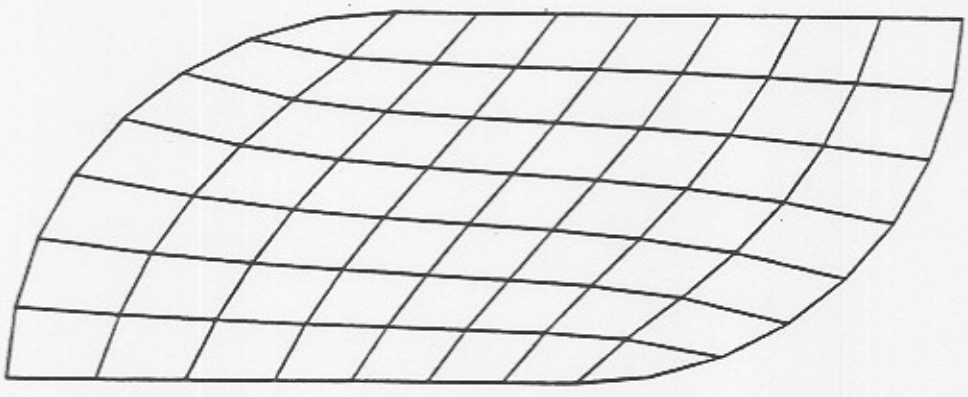
USE CURRENT SCHEME TO BEGIN PROCESSING? Y
*** WARNING: CORNER(S) OF THE REGION HAVE ***
*** LARGE ANGLES (> 150 DEGREES.) ***
*** POORLY FORMED MESH MAY RESULT ***
GENERAL RECTANGLE PRIMITIVE PROCESSING USED

FURTHER PROCESSING STEPS: P

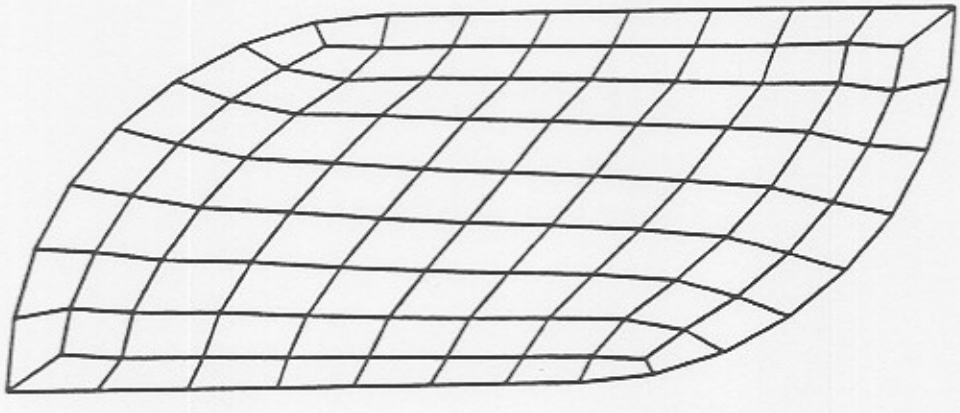


It is evident from the warnings and the plot that this mesh is not acceptable as it contains some rather flat elements. Some adjustment attempts are made, and plotted in the following interaction:

FURTHER PROCESSING STEPS: SP



FURTHER PROCESSING STEPS: NSP

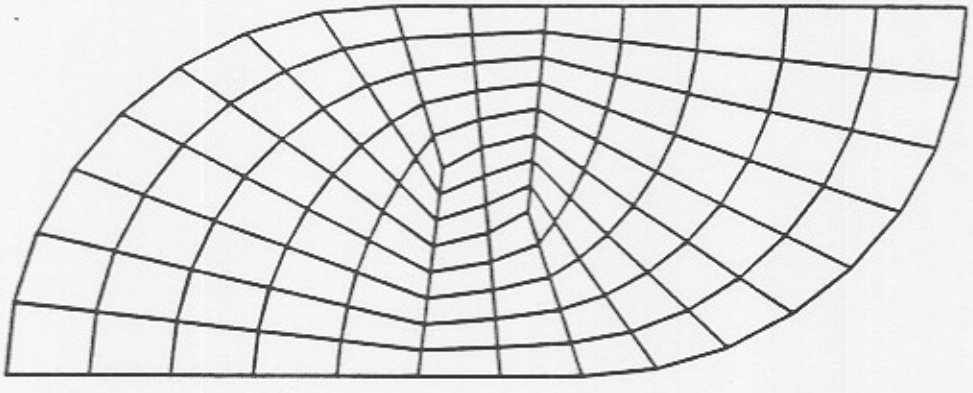


Although this mesh looks better, the user decides to see what would happen if he treated the region as a semicircle, since it really only contains two good corners. He abandons the processing done thus far, and retries the mesh using the semicircle primitive in the following manner:

FURTHER PROCESSING STEPS: O
PROCESSING RETURNED TO ORIGINAL

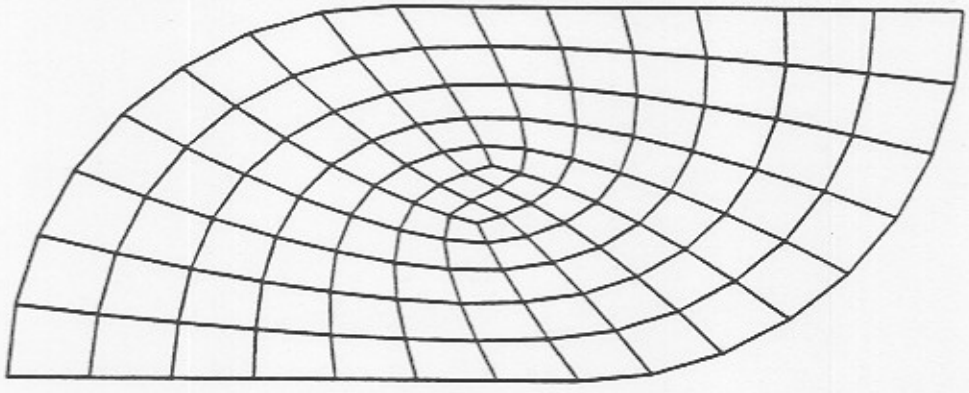
INITIAL MESH DEFINED USING THE SCHEME:
M

USE CURRENT SCHEME TO BEGIN PROCESSING? N
ENTER PROCESSING SCHEME: CP
SEMICIRCLE PRIMITIVE REGION PROCESSING USED



The boundary elements are now much more regularly shaped, but the interior elements suffer somewhat. The user attempts to help the mesh with:

FURTHER PROCESSING STEPS: 6SP
SMOOTHING TYPE SET TO "LENGTH WEIGHTED LAPLACIAN"
LENGTH WEIGHTED LAPLACIAN SMOOTHING COMPLETED



This mesh is deemed acceptable, and the user ends processing of this region. He could go on with region 2, but he is confident that there will be no problem with this region, so he decides to save region 1's scheme with an *EXIT* from *STEP PROCESS*ing in the following manner:

FURTHER PROCESSING STEPS: E
REGION 1 SUCCESSFULLY COMPLETED AND SAVED

PROCESS ADDITIONAL REGIONS? N

NUMBERING OF GENERATED OUTPUT BEGUN

****
MESH PROCESSING COMPLETED
NODES: 122; ELEMENTS: 106; MATERIALS: 1
****

The user then automatically processes both regions and displays the completed mesh in the following sequence:

ENTER MESH OPTION: P,G,P

MESH PROCESSING BEGUN

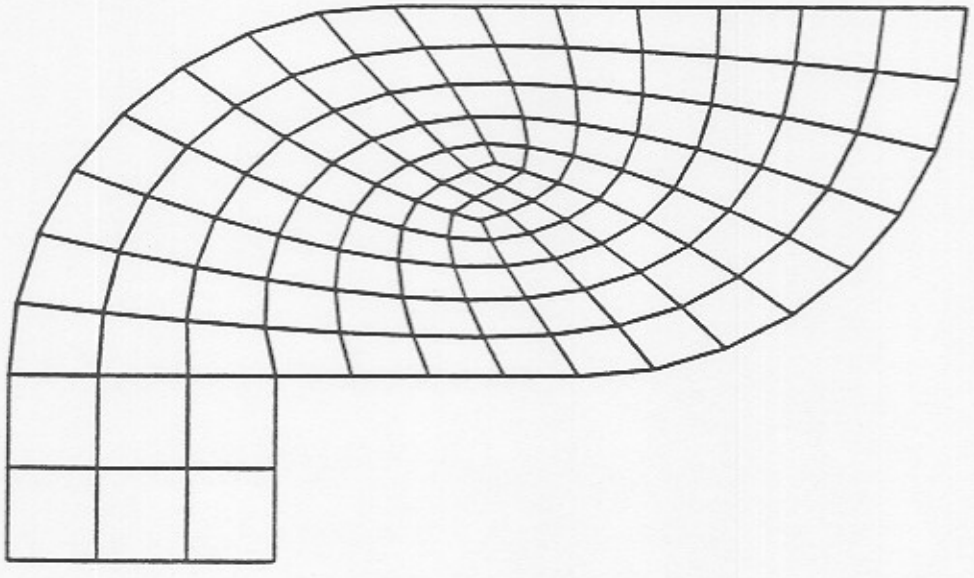
INITIAL CHECK BEGUN FOR REGION 1
INITIAL CHECK BEGUN FOR REGION 2

NOW PROCESSING REGION 1
SEMICIRCLE PRIMITIVE REGION PROCESSING USED
SMOOTHING TYPE SET TO "LENGTH WEIGHTED LAPLACIAN"
LENGTH WEIGHTED LAPLACIAN SMOOTHING COMPLETED
REGION 1 COMPLETED

NOW PROCESSING REGION 2
GENERAL RECTANGLE PRIMITIVE PROCESSING USED
REGION 2 SUCCESSFULLY COMPLETED AND SAVED

NUMBERING OF GENERATED OUTPUT BEGUN

****
MESH PROCESSING COMPLETED
NODES: 130; ELEMENTS: 112; MATERIALS: 1
****



The mesh looks approprite, so the user writes the mesh to a **GENESIS**[3] file in the following sequence:

ENTER MESH GRAPHICS OPTION: ,W,TEST.GEN

GENESIS OUTPUT FILE SUCCESSFULLY WRITTEN

The user has finished the mesh and attempts to exit without remembering that the problem geometry has not yet been written to a FASTQ file. **FASTQ** catches the user's error and the session is completed with the following interaction:

ENTER MESH OPTION: E
NO OUTPUT FILE WRITTEN

EXIT ANYWAY? N

ENTER MESH OPTION: ,W,TEST.FSQ,E
FASTQ DATA FILE SUCCESSFULLY WRITTEN

CPU SECONDS USED: 3.340
_____

Distribution:

Brigham Young University (3)
Department of Civil Engineering
CB 214
Provo, Utah 84602
Michael B. Stephenson

Cornell University
School of Civil and Environmental Engineering
Hollister Hall
Ithaca, NY 14853
C. H. Conley

Geological Survey
Branch of Pacific Marine Geology
345 Middlefield Road, MS 999
Menlo Park, California 94025
Kevin E. Gartner

KTECH Corp.
901 Penn. NE
Albuquerque, NM 87110
Steve Sauer

Lockheed Missiles & Space Co. (2)
B-157 PO Box 3504
Sunnyvalle CA 94088
Haluk Ozdemir Org. 81-12
Eric Matheson Org. 59-22

Los Alamos National Laboratories
CTR-4 MSF644
PO Box 1663
Los Alamos, New Mexico 87544
Jeff Hill

Monsanto Research Corporation
PO Box 32
Miamisburg, Ohio 45342
Tien S. Chou

Morton Thiokol, Inc.
Huntsville Division
Engineering Department
PO Box 7501
Huntsville, Alabama 35807-7501
A. L. Perroni

ONWI
BATTELLE Project Management
505 King Avenue
Columbus, Ohio 43201
E. Gregory McNulty, Ph.D., P.E.

Re/Spec Inc. (2)
3815 Eubank
PO Box 14984
Albuquerque, NM 87191
Malcolm Panthaki
Mark Blanford

Rice University
Department of Geology and Geophysics
PO Box 1892
Houston, Texas 77251
Dr. Jean-Claude DeBremaecker

United States Department of the Interior
Bureau of Mines
5629 Minneahaha Ave South
Minneappolis, Minnesota 55417
Steven Crum

University of Texas
Dept. of ASE/EM
Austin, Texas 78712
Eric Becker