MEEM 5408 Design Automation: Theory and Implementation
ASSIGNMENT 4 - FALL 2004
“Program Integration Using Files”

Write, compile and document a C++ program that executes several iterations of a Finite Element Analysis with varying geometry. In this assignment the Finite Element Analysis will be performed using ABAQUS. Your program must create an ABAQUS input file, execute the analysis, read the output file, and repeat for several iterations.

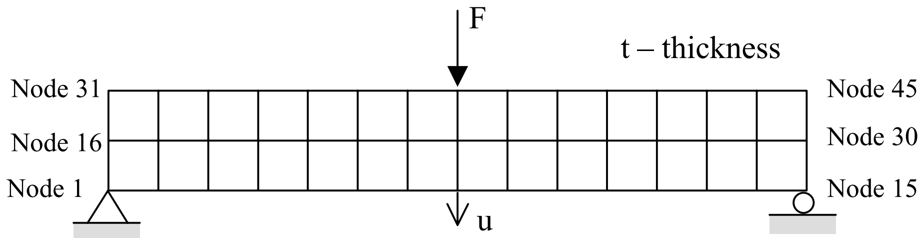

Figure 1. 2D beam

Figure 1 shows a possible geometry and mesh you could use, but you can also make up your own. It must have geometry, force and restraint boundary conditions and a point of interest. To create and understand the ABAQUS input file format, you can:

I. Study the “ABAQUS/Standard” manual.
II. Try creating a sample file using the ABAQUS user interface.
III. Study the input file for the problem in Figure 1, which is attached and described below.

To access the ABAQUS/Standard manual:

1. At a unix prompt, type `abaqus -cae`.
2. From the user interface menu select Help Getting Started. From here, switch to the “Getting Started with ABAQUS/Standard: Keywords Version” manual.
3. Close the ABAQUS program to allow others to use it. (We only have five floating licences available.)

To use the ABAQUS user interface:

1. At a unix prompt, type `abaqus -cae`.
2. From the user interface menu select Help Getting Started to access the ABAQUS documentation. Try doing the first tutorial.
3. Once a job has been created in the “Job” module, select “Write Input” in the “Job Manager.” Study the input file that is created. “Submit” the input file to make sure it works.

The problem in Figure 1 is a 2-D plane stress problem. The input file for this problem is attached below in the Appendix. Here is the description of it:

- Under "*Heading" is a comment describing the problem being solved.
- Beside and under "** Job name:" are settings for the particular job.
- Under "** PARTS" is a listing of parts that have been defined.
- Under "** ASSEMBLY" is a listing of part instances that make up the problem. Each part instance has it's own set of nodes and elements.
- Under "*Node" is the node information for the given instance. It is in the format: <Node No.>, <X coord>, <Y coord>.
- Under and beside "*Element" are the type of ABAQUS element (CPS4R) and the element connectivities given in a clock-wise fashion: <Element No.>, <Node 1>, <Node 2>, <Node 3>, <Node 4>.
- Under "** Region" are defined the element sets. In this example all elements are in the same set.
- Under "** Section" the material is assigned to the elements of each element set. In this case, the material named "STEEL" is assigned, and the thickness 1.2345 is assigned.
- Below each "*Nset" is defined a set of nodes. The sets are used for specifying which nodes the restraints and loads should be applied to. In this case, the first set, comprising node 38, is for applying the load. The second set, comprising node 1 is for the pinned restraint and the third set, comprising node 15, is for the roller restraint.
- Under "** Materials" is defined the material properties for each material. In this case, for the material named "STEEL," E = 2e+11 and the $\nu = 0.27$.
- Under "** STEP" is the information for each time step. In this case there is only one step.
- Under "** BOUNDARY CONDITIONS" are the boundary conditions for each step. In this case, the first node set is restricted in the 1 and 2 directions (X and Y directions) and the second node set is only restricted in the 2 direction (Y direction).
- Under "** LOADS" are the loads for each step. In this case, only one node set has a force of 0 in the 1 direction and –1000 in the 2 direction.
- Under "** OUTPUT REQUESTS" are the data that are required to be calculated. In this case, under "FIELD OUTPUT", it is requested to print the node displacements (U) to the output file: "*Node Print U". Note that this has been manually edited from "*Node Output U", which only sends the data to the database.

Your program must re-create input files with different geometry (thickness) in each iteration. To perform the analysis using the input file below (Assignment4.inp), the program must automatically perform the unix command:

```
abaqus job=Assignment4 input=Assignment4
```

To make this system call use the `system()` function declared in stdlib.h. Note that the results will not be immediately ready when the `system()` function returns. To see if the results are ready, you need to check if the lock-file (Assignment4.lck) is still there. To wait for the results, the following function may be helpful.

```
#include <time.h>

void Wait(float sec)
{
  timespec_t t1, t2;
  clock_gettime(CLOCK_REALTIME, &t1);

  do clock_gettime(CLOCK_REALTIME, &t2);
  while (t2.tv_sec - t1.tv_sec < sec);

}
```

Once the output file is ready, the lock-file will be removed and the results will be ready in the database file (Assignment4.odb) and in text format (Assignment4.dat). You will need to read in the results of interest from this file. In particular, you will need to read in the displacement of the node for the point of interest.[*] These steps will need to be repeated for each iteration, varying the geometric parameter. Your program will need to print out the displacement of the point as a function of the geometric parameter.

The program and documentation must be developed individually. You may discuss how to do something with others, but you may not look at source code or reports from others, and they may not look at yours.

Write up a report with the same sections as in the first assignment. The source code must be fully documented, including the same header information specified in the first assignment. **The source code will be evaluated for style, neatness, correctness and efficiency.**

Email the source code to me at bettig@mtu.edu. The report must be handed-in in class. The report and source code are due Friday, October 29.

---

[*] Note that results sent to the database file (with "*Node Output" or "*Element Output") can be viewed in the ABAQUS Visualization module. It is not required in this course, but normally you will need to verify that your FEA results are valid.

Appendix: Assignment4.inp

```
*Heading
 Assignment 4
** Job name: Job-1 Model name: Model-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-1
*End Part
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-1-1, part=Part-1
*Node
      1,           0.,           0.
      2,          10.,           0.
      3,          20.,           0.
      4,          30.,           0.
      5,          40.,           0.
      6,          50.,           0.
      7,          60.,           0.
      8,          70.,           0.
      9,          80.,           0.
     10,          90.,           0.
     11,         100.,           0.
     12,         110.,           0.
     13,         120.,           0.
     14,         130.,           0.
     15,         140.,           0.
     16,           0.,          10.
     17,          10.,          10.
     18,          20.,          10.
     19,          30.,          10.
     20,          40.,          10.
     21,          50.,          10.
     22,          60.,          10.
     23,          70.,          10.
     24,          80.,          10.
     25,          90.,          10.
     26,         100.,          10.
     27,         110.,          10.
     28,         120.,          10.
     29,         130.,          10.
     30,         140.,          10.
     31,           0.,          20.
     32,          10.,          20.
     33,          20.,          20.
     34,          30.,          20.
     35,          40.,          20.
     36,          50.,          20.
     37,          60.,          20.
     38,          70.,          20.
```

```
     39,          80.,          20.
     40,          90.,          20.
     41,         100.,          20.
     42,         110.,          20.
     43,         120.,          20.
     44,         130.,          20.
     45,         140.,          20.
*Element, type=CPS4R
 1,  1,  2, 17, 16
 2,  2,  3, 18, 17
 3,  3,  4, 19, 18
 4,  4,  5, 20, 19
 5,  5,  6, 21, 20
 6,  6,  7, 22, 21
 7,  7,  8, 23, 22
 8,  8,  9, 24, 23
 9,  9, 10, 25, 24
10, 10, 11, 26, 25
11, 11, 12, 27, 26
12, 12, 13, 28, 27
13, 13, 14, 29, 28
14, 14, 15, 30, 29
15, 16, 17, 32, 31
16, 17, 18, 33, 32
17, 18, 19, 34, 33
18, 19, 20, 35, 34
19, 20, 21, 36, 35
20, 21, 22, 37, 36
21, 22, 23, 38, 37
22, 23, 24, 39, 38
23, 24, 25, 40, 39
24, 25, 26, 41, 40
25, 26, 27, 42, 41
26, 27, 28, 43, 42
27, 28, 29, 44, 43
28, 29, 30, 45, 44
** Region: (Section-2:Picked)
*Elset, elset=_PickedSet2, internal, generate
  1,  28,   1
** Section: Section-2
*Solid Section, elset=_PickedSet2, material=Steel
1.2345,
*End Instance
*Nset, nset=_PickedSet4, internal, instance=Part-1-1
 38,
*Nset, nset=_PickedSet5, internal, instance=Part-1-1
 1,
*Nset, nset=_PickedSet6, internal, instance=Part-1-1
 15,
*End Assembly
**
** MATERIALS
**
*Material, name=Steel
*Elastic
 2e+11, 0.27
** ----------------------------------------------------------------
```

```
**
** STEP: Step-1
**
*Step, name=Step-1
The static analysis of the shape.
*Static
1., 1., 1e-05, 1.
**
** BOUNDARY CONDITIONS
**
** Name: BC-1 Type: Displacement/Rotation
*Boundary
_PickedSet5, 1, 1
_PickedSet5, 2, 2
** Name: BC-2 Type: Displacement/Rotation
*Boundary
_PickedSet6, 2, 2
**
** LOADS
**
** Name: Load-1   Type: Concentrated force
*Cload
_PickedSet4, 1, 0.
_PickedSet4, 2, -1000.
**
** OUTPUT REQUESTS
**
*Restart, write, frequency=1
**
** FIELD OUTPUT: F-Output-2
**
*Output, field
*Node Print
U
**
** HISTORY OUTPUT: H-Output-1
**
*Output, history, variable=PRESELECT
*El Print, freq=999999
*Node Print, freq=999999
*End Step
```