Finite Element Programming

Assignment 1: Applying a mesh database
(10% of the course grade)

Due: February 14, 2019

The purpose of this assignment is to develop a simple mesh manipulation application building on the PUMI mesh infrastructure. Using the techniques described in class, you will write procedures to execute meshing operations using the PUMI unstructured mesh infrastructure.

For each of the operations you are asked to provide:

1. A description of the algorithm (e.g., which set of adjacencies are used to get the requested adjacency or how the mesh was generated and adjacencies constructed, etc.)
2. A copy of the code that is fully commented.
3. The other specific items requested in the statement of the specific operation.

The operations to execute include:

1. Calculate the volume of the mesh regions for predefined mesh tet-mesh-1. Provide a list of element numbers and volumes for each element.
2. Determine the number of mesh edges that each vertex bounds. Provide that information for each vertex.
3. Create a uniform 2-D quadrilateral mesh of 4 elements by 6 elements on a square of edge length 8.0 with its lower left vertex placed at 0.0. Provide a list that gives the coordinates for each vertex. Provide a list of the vertex numbers for each element. Provide an image of the mesh.
4. For the mesh, mixed-mesh-1 determine and provide the list of mesh faces classified on model face 81.
5. For the mesh mixed-mesh-1 provide a list that indicates the type of region (tet, hex, etc.) for each of the mesh regions.
6. For the mesh tet-mesh-1, convert the linear elements into quadratic elements by adding mid-edge nodes. Provide a list giving the coordinates of the mid-edge nodes for all the mesh edges.

The needed documentation and files include the:

1. PUMI interface function definitions in PUMI User's guide at https://scorec.rpi.edu/~seol/PUMI.pdf and introduction document at https://scorec.rpi.edu/~seol/PUMI-intro.pdf.
2. Instructions to create mesh images are in the Appendix of the PUMI User's Guide.
3. Build instructions and files for tex-mesh-1, mixed-mesh-1 in the shared CCI directory: `~/barn-shared/fep/a1`

Questions are welcome and further clarifications will be provided through the Github issues page: https://github.com/SCOREC/fep/issues.

Finite Element Programming

Assignment 2: Exercising meshes in Parallel (5% of the course grade)

Due: February 28, 2019

The purpose of this assignment is to further exercise the ability to deal with meshes, now on a parallel computer using distributed memory (MPI based) parallelism.

Using the techniques described in class, you will write a set of procedures to execute a set of parallel meshing operations using the PUMI unstructured mesh infrastructure.

For each of the operations you are asked to provide:
1. A description of the algorithm used.
2. A copy of the code that is fully commented.
3. The other specific items requested in the statement of the specific operation.

The operations to execute include (more detailed instructions to follow):
1. Load the partitioned mesh parallelMesh.smb, and determine the number of regions using each mesh vertex accounting for the full mesh (including edges on other parts). For each part, provide a list for only the vertices owned by that part and indicate the local vertex number (remember not all vertices will be owned by that part) and the number of mesh regions that vertex bounds (on all parts).
2. Load the partitioned mesh parallelMesh.smb, and determine the number of edges using each mesh vertex accounting for the full mesh (including edges on other parts). For each part, provide a list for only the vertices owned by that part and indicate the local vertex number (remember not all vertices will be owned by that part) and the number of mesh faces that vertex bounds (on all parts).
3. Load the partitioned mesh parallelMesh.smb and determine the list of mesh regions on part 1 that have one or more mesh faces on the partition model face between parts 1 and 2. Apply mesh migration to migrate those mesh regions from part 1 to part 2. Report the number of mesh faces on the partition model face between parts 1 and 2 before and after the migration process.

The needed documentation and files include the:
1. PUMI interface function definitions in PUMI User's guide at https://scorec.rpi.edu/~seol/PUMI.pdf and introduction document at https://scorec.rpi.edu/~seol/PUMI-intro.pdf.
2. Build instructions and files for parallelMesh.smb in the shared CCI directory: `~/barn-shared/fep/a3`

Questions are welcome and further clarifications will be provided through the Github issues page: https://github.com/SCOREC/fep/issues.

Finite Element Programming

Assignment 3: Reordering nodes and elements (5% of the course grade)

Due: March 11, 2019

This assignment builds on the first assignment and represents a reordering algorithm that you can use in your finite element analysis program.

There are three example meshes to which you are to apply the reordering procedure. Two of the examples are with "linear elements" meaning there are nodes at the mesh vertices only. The third example has "quadratic elements" meaning there are nodes at each mesh vertex and the mid-point of each mesh edge. The meshes and starter code have been added to your shared directory at:

```
~/barn-shared/fep/a2
```

You must turn in your algorithm design, your documented code, two figures (per mesh) from Paraview showing the node and element labeling before and after reordering. Please refer to the Appendix of the PUMI User's Guide (http://scorec.rpi.edu/pumi/PUMI.pdf) for details on generating pictures of a Numbering. Finally, questions are welcome and further clarifications will be provided through the Github issues page: https://github.com/SCOREC/fep/issues.

Finite Element Programming

Assignment 4 Option 1:
Complete Finite Element Code
(30% of the course grade)

Due: April 4, 2019

The goal of this project is to develop a finite element analysis code. You have to select one of two options for doing this. Option 1 involves writing a complete code for solving linear elliptic problems. The FE code should build directly on the mesh database and reordering used/developed in the earlier assignments. To demonstrate the code you will solve specific 2-D linear elastic problems over domains of isotropic materials (if you want to solve some other PDEs, that is possible with preapproval of the instructor). For this assignment you can do everything in serial. However, if you want to give it a go, feel free to develop parallel analysis code.

Additional information with more specific instructions will follow over the next few weeks. You will want to start developing the pieces of the program as we cover the specific technical topics.

The program you develop will have the following features:

1. A general design to allow expansion to include additional capabilities.
2. Operate off a geometry-based problem definition.
3. A two-dimensional element library including three- and four-sided elements with linear and quadratic shape functions. Appropriate numerical integration orders should be used. (Can be 3D if you would like.)
4. Global equation solver and associated assembly routines that account for the fact that the global system is sparse and symmetric. You can either write these from scratch, or use a library. If you are attempting to do a parallel version, I recommend you use the PETSc library with SUPERLU being a good option. If you do use a solver library, your documentation must describe the specific method used by the selected solver.
5. Recovery of secondary variables.
6. Provide output in terms of numerical results and/or graphical output.

The project write-up should contain the following:

7. A technical description of the finite elements and numerical methods implemented. This includes
8. Documentation of the program design including all major components. If objected-oriented methods are used, this will be your classes, etc. If you use "structured programming" this would be your abstract data types, etc.
9. Pseudo code for the entire code.
10. A carefully designed set of test problems that clearly demonstrate the correct implementation of all program functions. (This is a very important part of the program development process. You should include unit tests (e.g., correct element stiffness matrices, etc.) simple problems showing each capability and more complex examples. This part tends to be the one students do not do well.)
11. Listing of the program code. (Be sure the code is well commented.)

Finite Element Programming

Assignment 4 Option 2:
Add New Element to a Finite Element Framework
(30% of the course grade)

Due: March 28, 2019

The goal of this project is to develop a finite element analysis code. Option 2 involves adding a new finite element to an existing finite element framework for something more complex than a linear elliptic problem. Important items to consider before selecting this option:

- If you select this option for Assignment 4, you have to be sure your course project is something substantially different, or in addition to, adding a new element and clearly demonstrating it properly works.
- You have to submit a proposal on what you propose for this assignment and have it approved in advance. This includes indicating what you expect to specifically demonstrate and document in your write-up. See the Option 1 description for input on some of what you should consider to demonstrate and document.
- Depending on which finite element framework you decide to work with, there will be little or no support or help available. You will be responsible for getting things put together, etc. (Just getting some of these frameworks running on a system can be a big job.)

Option 2 is suggested only for those that either already working with, or plan to be working with, a specific finite element (or finite volume) framework.

Some of the possible frameworks are: MFEM, Albany, FEniCS, PHASTA, Deal.II and OpenFoam.