Workflow on Running PHASTA

Fanlong Meng

December 23, 2014

1 Introduction

PHASTA is an fluid solver, which is short for a Parallel Hierarchic Adaptive Stabilized Transient Analysis. PHASTA can model both compressible and incompressible, laminar and turbulent, steady and unsteady flows in 3D, using unstructured grids. This documentation is a short description on how to run PHASTA as well as its pre and post-processor on SCOREC workstations and the CCI's IBM BlueGene/Q system.

2 Accounts

In order to use PHASTA to solve fluid mechanics problems, you need to have access to either the SCOREC workstations or the CCI system.

You can contact Cameron Smith at 'smithc11 at rpi dot edu' to get access to the SCOREC computer.

You can refer to the following FAQ webpage on how to get CCI account and access to CCI.

`https://secure.cci.rpi.edu/wiki/index.php/Frequently_Asked_Questions`

3 Getting Started at SCOREC/CCI

For users new to SCOREC and IBM BlueGene/Q or new to linux system, you can find detailed instructions about how to connect to the systems, how to move files into or out of the systems and some basics about the file system in Appendix A.1 to Appendix A.3.

New users need to get an environment set up on either SCOREC or IBM BlueGene/Q. There are several pieces of third-party software required to compile PHASTA and its preprocessor, as well as other programs that you may find useful in development. For various reasons these are not available by default, and need to be activated through the module system. You can download the core tools and compile the stack by following the instructions in Appendix A.4. After setting up the environment and compiling the stack, you can build the PHASTA pre-processor *chef*, download and build PHASTA and Model and mesh converters. Detailed instructions can be found in Appendix B.

4 Test Case Setup

In this document, a test case is demonstrated to show you how to setup a fluid problem and how to solve it with PHASTA.

Figure 1 depicts an incompressible flow problem that will be solved with DES(detached eddy simulation). The governing equations are incompressible Navier-stokes(NS) equations and a turbulent viscosity transport equation.

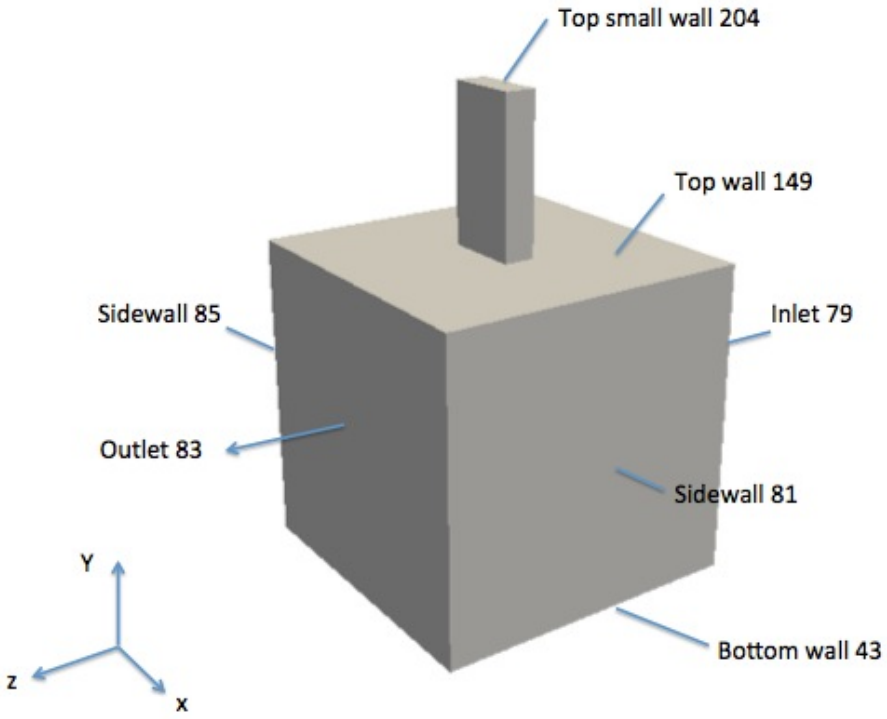

Figure 1: Test case problem

The boundary conditions for this problem are as follows,

- at inlet surface 79, the inflow velocity is $v = v_z = 1$, the inflow turbulent viscosity is $5.4e-05$;
- at outlet surface 83, the outflow traction is 0, the outflow pressure is 0 and the derivative of turbulent viscosity equals 0;
- at the small top wall 204, the turbulent viscosity is 0, and a jet velocity that is sinusoidal in time is prescribed in y direction;
- the top wall 149 and the four sided walls of the top region(not labeled in

Figure 1.) satisfy no-slip, no-penetration boundary condition, therefore the turbulent viscosity on the walls is also equal to 0;

- at the side wall 81 and 85 in the big region, $v_x = 0$, derivative of turbulent viscosity is 0 and the tractions on the walls are equal to 0;

- at the bottom wall 43, $v_y = 0$, derivative of turbulent viscosity is 0 and the tractions on the wall are equal to 0.

The initial conditions for the test problem are prescribed for the entire region: the initial velocity is $v = v_z = 1e-08$, the initial turbulent viscosity is $5.4e-05$, the initial pressure is 0.
Except for solving the incompressible NS equations, PHASTA is also able to solve Euler equations and compressible NS equations. Meanwhile, it can employ RANS, LES as well as DES for turbulent modeling. The level-set method is available for solving the problem with multiphase flow. Solving strategies can be modified by editing 'solver.inp'.

5 Mesh Generation

Before running PHASTA pre-processor *chef*, you need to generate a model file, a mesh file as well as an attribute(.spj) file describing the problem boundary conditions and initial conditions.

While setting up the problem, you need to make sure that you have the correct file format at different stage in the simulation. More detailed step by step instruction can be found in Appendix C.

First, you need to create a Parasolid file(.xmt_txt) using CAD software. Then, you can generate the Simmetrix mesh file using Simmetrix SimModeler GUI. A detailed mesh generation tutorial can also be found in here:
`https://www.scorec.rpi.edu/researchwiki/SimModeler`.

The mesh file for the test case problem is shown in Figure 2. It is obvious that the mesh size is relatively small near the section with small section area since the variation of velocity in that section is larger then the others. Also a 3D boundary layer is used in mesh generation on the highlighted surface.

With a Simmetrix model and SCOREC mesh, you can generate the .spj file that specifies the boundary and initial conditions. The ascii text file for boundary conditions is the only supported way to define boundary/initial conditions for this version of *chef*, and can only be generated manually. More details about how to create the boundary/initial conditions in spj file can be found in Appendix D.

Other than generating a simple mesh like this case, SimModeler is able to generate a much more complex mesh. For example, Figure 3 show a parallel unstructured mesh generation of 13 million element mesh on a complex geometric model from mesh controls defined in GUI. Figure 4 shows 16 parts of the 128 part partitioning of the 13 million element mesh of upright model.

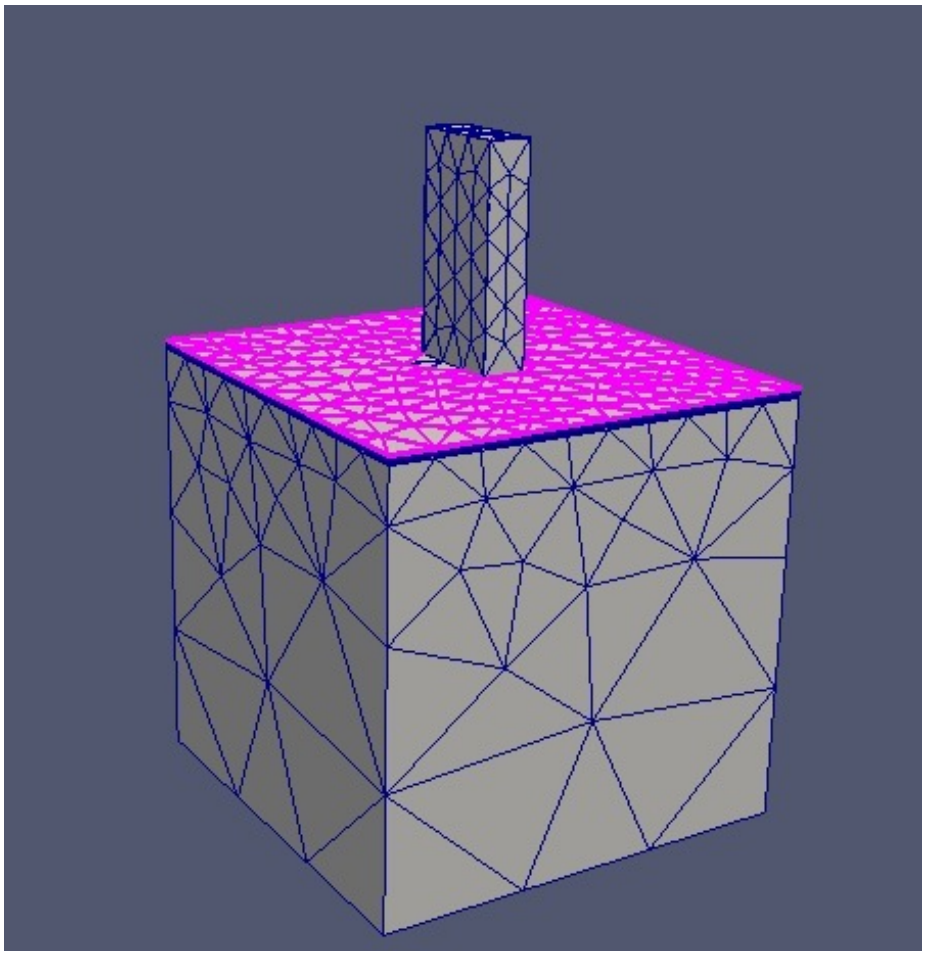

Figure 2: Mesh of test case

6 Run Preprocessor and PHASTA

After generating the required mesh files, you can run preprocessor *chef* to partition the mesh and generate the PHASTA input files. Then you can run PHASTA.

6.1 Run Preprocessor

You can run preprocessor *chef* using the following commands. For SCOREC computer:

```
cd crossflow
source /users/myusername/core-sim/env-scorec.sh
cd 4-1chef
```

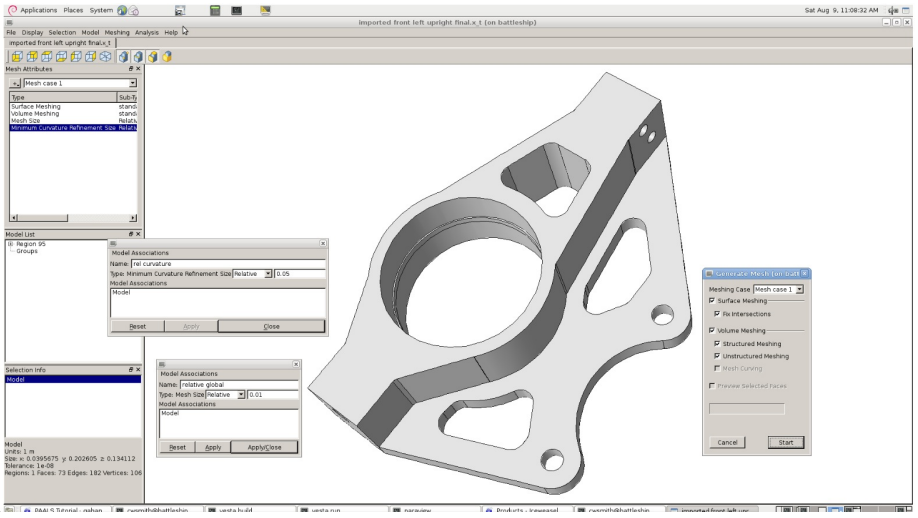

Figure 3: Suspension upright from the RPI Formula Hybrid race car in SimModeler GUI

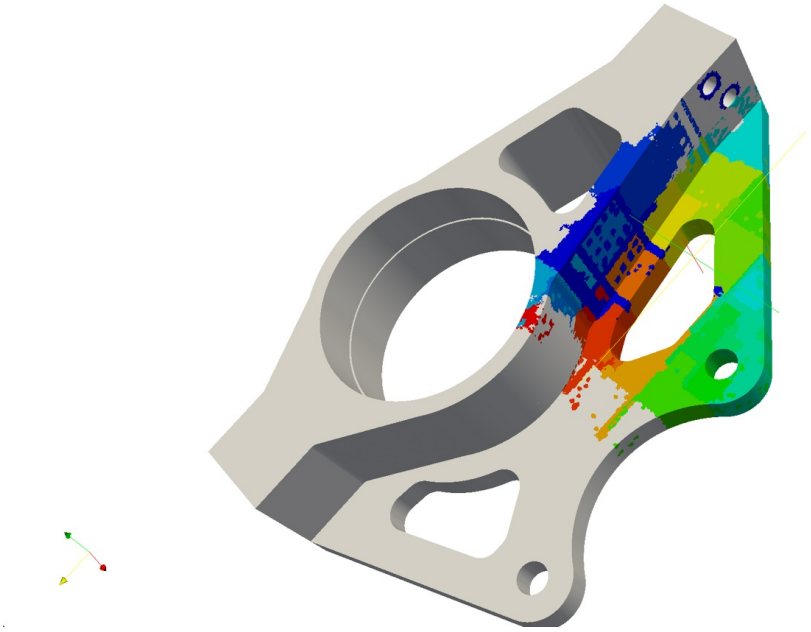

Figure 4: Parts 0-15 of the 128 part partitioning of 13 million element mesh of upright model

`/users/myusername/coreSim/build/test/chef`

For IBM BlueGene/Q:

```
cd crossflow
module clear
source /gpfs/u/barn/myprojectname/myusername/core-sim/env-amos.sh
cd 4-1chef
srun -t 5 -n 1 -o log /gpfs/u/barn/myprojectname/myusername/core-sim/build/test/chef
```

This should create a 4-procs_case directory containing the PHASTA input files in the posix format.

6.2 Run PHASTA

First you need to run program converterO2N to generate PHASTA input files in the syncoIO format, then you can run PHASTA. For SCOREC computer:

```
cd 4posixToSync
mpirun -np 4 /path/to/phasta/mrPhasta/build/bin/converterO2N
cd 4solve
mpirun -np 4 /path/to/phasta/mrPhasta/build/bin/phastaIC.exe
```

For IBM BlueGene/Q:

```
cd 4posixToSync
srun -t 20 -n 4 -o log /gpfs/u/barn/myprojectname/myusername/phasta/ &
mrPhasta/build/bin/converterO2N
cd 4solve
srun -t 20 -n 4 -o log /gpfs/u/barn/myprojectname/myusername/phasta/ &
mrPhasta/build/bin/phastaIC.exe
```

Running coverterO2N should create a 4-procs_case-SyncIO-1 directory containing phasta input files in the 'syncoIO' format. Running PHASTA should create a 4-procs_case directory containing restart-dat.[4—8—12—16—20].1 files in the posix format.

7 Output

One way to confirm that your solution is plausible is to compare the log file generated after running PHASTA with the reference file phasta_4.log_ref.

The following is a sample output from phSolver for one time step, due to the limited page size, the first four columns of output are displayed in the top table and the last four columns are displayed in the bottom one:

Timestep	CPU Time	Residual*	Normalized residual
10	3.113E+02	1.676E-05	(-10)

Max(DeltaU/U)	Max(Deltap/P)	NOT RELEVANT	CG - GMRES iterations
2.205E-02	3.361E-01	1215 -3 12	[35-10]

*Where Residual is only the residual from the momentum and continuity equations.

The difference between residuals of your calculation and the reference should be smaller than 10^{-6}. The small discrepancy is due to the partitioning generated by *chef* is slightly different from the reference. Partitioning is indeed non deterministic so this should be fine.

8 Post-processing

To visualize the results, you need to convert them from 'syncIO' format to 'posix' format. Before running Paraview, you first need to convert the restart files to 'posix' format which is a readable format in Paraview. You can either use Paraview in SCOREC workstation or download it onto your own desktop through: `www.paraview.org`

create 'IO.N2O.input' with the following content in folder /path/to/4solve:

```
N-geombc-fields-double: 3;
N-geombc-fields-integer: 20;
N-restart-fields-double: 1;
N-restart-fields-integer: 1;
N-steps: 20;
N-parts: 4;
N-files: 1;
restart, solution, double, block, 3;
restart, ybar, double, block, 3;
restart, errors, double, block, 3;
restart, dwal, double, block, 3;
```

where, N-steps is the total number of time-steps to load; N-parts is the total number of partitions/parts to load(the number of cores used for the calculation by PHASTA in this case 4)

then run converterN2O:

```
mpirun -np 4 /path/to/phasta/mrPhasta/build/bin/converterN2O \&> log
cat log
```

This should create a 4-procs_case-1PPP directory containing PHASTA output files in the 'posix' format.

Take geombc files in 'posix' format from the PHASTA input files to the new directory 4-procs_case-1PPP:

```
cd 4-procs\_case-1PPP
cp ../../../4-1chef/4-procs-case/geombc.dat.* ./
```

Now you have a full set of geombc and restart files in 'posix' format. Copy the phasta.pht file from the crossflow directory to the /4-procs_case-1PPP directory. Open the phasta.pht file with Paraview to visualize the posix formatted PHASTA files.

After running PHASTA, the output files can be visualized by Paraview. Figure 5 shows a surface cut at half plane in x-normal direction of turbulent viscosity. Figure 6 shows a iso-surface plot of the pressure filed.

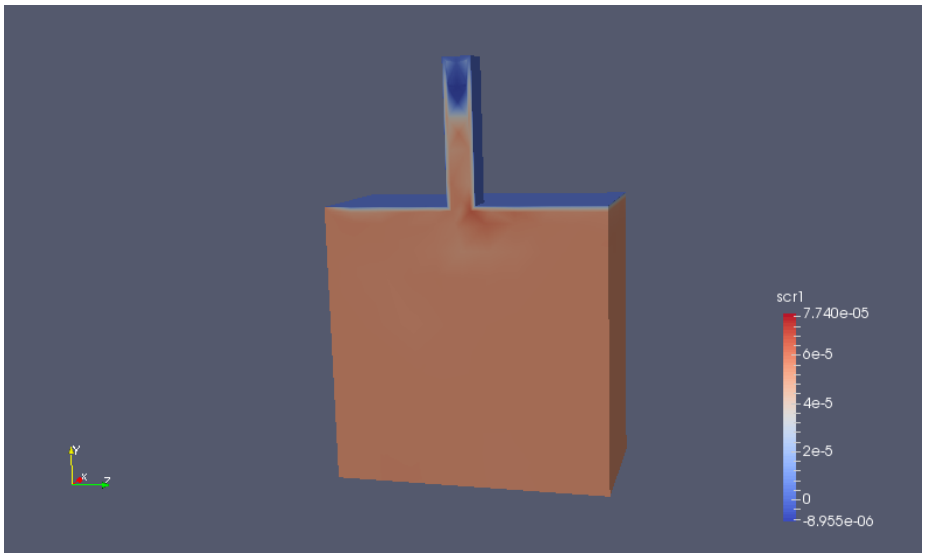

Figure 5: A cut surface of turbulent viscosity

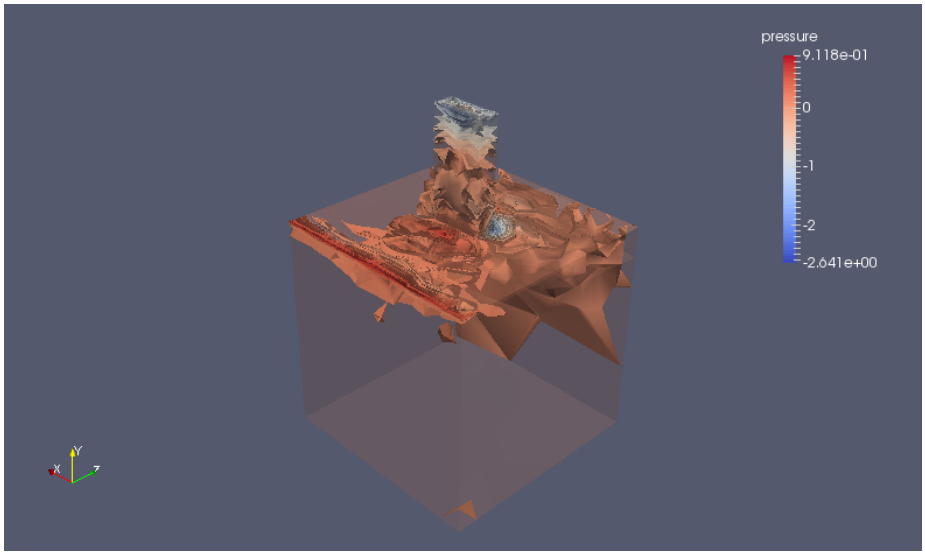

Figure 6: A isosurface of pressure field

A Getting Started at SCOREC/CCI

A.1 Connecting to the systems

A.1.1 SCOREC

After getting an account on SCOREC systems you should have a username and password. To access SCOREC systems, you must first go through the gateway machine at SCOREC called jumpgate. Running this from a terminal on your home computer will access jumpgate.

```
ssh myusername@jumpgate.scorec.rpi.edu
```

Enter your password at the prompt and you should end up logged into jumpgate. From here you can log in to any other SCOREC systems. A list of computers and their specs can be found here:
`https://www.scorec.rpi.edu/wiki/Main_Page`

Since you're already inside SCOREC systems, you don't need to repeat your username or scorec.rpi.edu.

```
ssh othello
```

A.1.2 CCI

The instructions of connecting to IBM BlueGene/Q can be found in here:
`https://secure.cci.rpi.edu/wiki/index.php/Landing_Pad`

A.2 File Systems

A.2.1 SCOREC

There are several directories at SCOREC where you can put your files. The two most important ones are your home directory and your fast tmp directory. Your home directory is a safe place for important but not too large files. After logging in to a SCOREC machine you are in your home directory.

```
/users/myusername/
```

The /lore directory is a much bigger space suitable for keeping a copy of the SCOREC code for compiling and testing, as well as input and output data for your programs. However, any important code changes should be committed to the source code repository and important files should be moved to your home directory if possible. Here is the command to move to your /lore directory.

```
cd /lore/myusername/
```

These directories are conveniently available on all SCOREC machines, so they look the same no matter where you log in and changes made on one machine show up in another.

A.2.2 CCI

CCI file system information can be found in:
`https://secure.cci.rpi.edu/wiki/index.php/File_System`

A.3 Moving Files

To move files to and from the SCOREC/CCI systems, a program similar to ssh called scp is useful. For example, to copy a file called mydata.txt from your home computer to your SCOREC/CCI home directory, run these on your home computer respectively.

```
scp mydata.txt myusername@jumpgate.scorec.rpi.edu:mydata.txt

scp mydata.txt myusername@lp01.ccni.rpi.edu:mydata.txt
```

Similarly, to copy the same file from your SCOREC/CCI home directory onto your home computer, run these on your home computer respectively.

```
scp myusername@jumpgate.scorec.rpi.edu:mydata.txt ./

scp myusername@lp01.ccni.rpi.edu:mydata.txt ./
```

As you can see, scp is a lot like cp except the source and destination files can be on different machines.

A.4 Compiling the Stack

There are several pieces of third-party software required to compile SCOREC/CCI code, as well as other programs that you may find useful in development. For various reasons these are not available by default, and need to be activated through the module system.

You can download the core tools 'core-sim' at:

```
cd
cp -r /users/mengf5/coresim.tar.gz ./
tar xzf coresim.tar.gz
```

for SCOREC workstations, and

```
cd /gpfs/u/barn/myprojectname/myusername/
cp -r /users/mengf5/coresimBGQ.tar.gz ./
tar xzf coresimBGQ.tar.gz
```

for IBM BlueGene/Q.

Then you have a new directory called 'core-sim' containing the code. Move into it.

```
cd core-sim
```

We have prepared a basic set of modules necessary to compile the code. This script should be executed once right after logging in to a computer. For SCOREC you should run:

```
source env-scorec.sh
```

The script for loading software dependencies on SCOREC systems env-scorec.sh won't work on IBM BlueGene/Q. For IBM BlueGene/Q you should run:

```
source env-amos.sh
```

Now to compile the code for the first time. In the core directory, make a build directory to contain the compiled files and move into it

```
mkdir build
cd build
```

Next you will need to configure some options related to how the code is compiled. There is already an example script setting default options: Compilation options are set by running CMake via the following script:

```
source ../config.sh
```

where CMake is an extensible, open-source system that manages the build process, more info about CMake can be found in:
`http://www.cmake.org/overview/`

Now compiling all the code is as simple as running the following command.

```
make -j 4
```

The -j 4 option tells the compilation system to compile 4 files at a time, which makes things faster on machines that have multiple cores. Again, if all goes well you will not see any compilation errors.

B Build PHASTA pre-processor/Model and Mesh Converters/Build PHASTA Flow Solver

B.1 Build PHASTA pre-processor and Model and Mesh Converters

After compiling the stack, build the PHASTA pre-processor *chef*:

```
make chef
```

then build the mesh format converter:

```
make convert
```

Meanwhile, you need to build the cadtosim to convert the model file from Parasolid format to Simmetrix format:

```
cd core-sim/test/cadToSim/
./build.sh
```

B.2 Build PHASTA Flow Solver

More reference about PHASTA can be found at:
`https://www.scorec.rpi.edu/wiki/PHASTA_Documentation`

You can download the PHASTA code for SCOREC following:

```
cd
cp -r /lore/cwsmith/develop/phasta.tar.gz ./
tar xzf phasta.tar.gz
```

for IBM BlueGene/Q:

```
cd /gpfs/u/barn/myprojectname/myusername/
cp -r /users/mengf5/phastaBGQ.tar.gz ./
tar xzf phastaBGQ.tar.gz
```

You need to clear the modules loaded for core-sim before building the PHASTA flow solver, then you can build the PHASTA flow solver following:

```
cd phasta
source setEnv.sh
cd mrPhasta
mkdir build
cd build
../../doConfigureGccMpich
make
```

C Mesh Generation

The model file in Parasolid format with suffix .xmt_txt can be used on SCOREC workstations. However, Since that the Parasolid modeling kernel is not available on the IBM BlueGene/Q, you have to use a model file in Simmetrix format with suffix .smd that is available. The mesh file will be used in *chef* is required to be in SCOREC MDS format with suffix .smb.

First, you need to generate a Parasolid file(.xmt_txt) using CAD software. To use this file on CCI, you can either convert it to smd file using following command:

```
cadtosim <parasolid model>
```

which will create a new .smd model in your directory called translatedmodel.smd.

Then, you can generate the Simmetrix mesh file using Simmetrix SimModeler GUI.

After successfully generating the Simmetrix mesh (.sms file), there is a converter from the Simmetrix mesh format to the SCOREC mesh format.convert the Simmetrix mesh that you create into a Scorec mesh:

```
convert <simmetrix model> <simmetrix mesh> <scorec mesh>
```

You can download the test case for SCOREC at:

```
cd
cp -r /users/mengf5/crossflow.tar.gz ./
tar xzf crossflow.tar.gz
```

for IBM BlueGene/Q at:

```
cd /gpfs/u/scratch/myprojectname/myusername/
cp -r /users/mengf5/crossflowBGQ.tar.gz ./
tar xzf crossflowBGQ.tar.gz
```

In the crossflow folder, the you can find the mesh, model file in correct format and .spj file under /crossflow.

D Boundary Conditions

The boundary and initial conditions need to be specified in .spj file manually. In this Section, full names and abbreviations for boundary conditions as well as some comments are documented: In .spj file, Essential boundary conditions are :

```
density D rho
temperature T
pressure P
comp1 C1
comp3 C3
scalar_1 S1 sc1
scalar_2 S2 sc1
scalar_3 S3 sc1
scalar_4 S4 sc1
```

,natural boundary conditions are:

```
mass flux MF
natural pressure NP
traction vector TV
heat flux HF
* turbulence wall TW
scalar_1 flux F1
scalar_2 flux F2
scalar_3 flux F3
scalar_4 flux F4
* surf ID SID
```

Initial conditions are:

```
initial pressure
initial velocity
initial temperature
initial scalar_1
initial scalar_2
initial scalar_3
initial scalar_4
```

A .spj file is composed of two kinds of lines, one is # comments begin with a pound sign, and it is followed by a line describing:

```
condition_name: model_id model_dim c0 c1 c2 c3.
```

The `model_id` and `model_dim` specify a geometric model entity. Typically `model_dim` = 2 for boundary conditions on model faces, or `model_dim` = 3 for initial conditions on model regions.
All conditions have one component except for the following:

```
comp1 4
comp3 4
initial velocity 3
traction vector 3
```

Both of `comp1` and `comp3` are essential boundary conditions affecting velocity.
`comp3` is a strict constraint, $velocity = m * (x, y, z)$
`comp1` is a planar constraint, $dot(velocity, (x, y, z)) = m$;

Both are listed in the spj file as: `m x y z`
Inital velocity and traction vector are both listed as: `x y z`.
The scalars `1` through `4` are not always present.
Note, `Surf ID` is an attribute which allows us to tag some faces. This is use to tell PHASTA to compute the force that applies on some tagged faces. To apply a varying BC (both in time and space), in the test case problem the sinusoidal velocity on surface 204, we first set up a constant zero velocity profile in the .spj file (search for face 204 in geom.spj file). Then, with the additional `Surf ID` associated with this face, PHASTA can retrieve the vertices located on this face through the `Surf ID` attribute and prescribe a new velocity profile, which acts similarly to a new inlet. This prescribed velocity profile is implemented in phSolver/incompressible/BCprofie.f.