Rocman/GEN2.5 Users Guide

Xiangmin Jiao

December 10, 2003

Disclaimation

As of Dec. 10, 2003, this users guide is superseded by the GENx Users Guide maintained by Mark Brandyberry, and hence this guide will no longer be maintained. The content of this document may be already obsolete.

1 Checking out and updating GEN2.5

You need a CVS account for CSAR's code repository in order to check out GEN2.5. Please contact Mark Brandyberry (mdbrandy@uiuc.edu) to create an account. Suppose you username is auser. You need to set the CVSROOT environment variable to ":pserver:auser@galileo.csar.uiuc.edu:/cvsroot".

To check out the codes, please use the command "cvs co -p genx/Codes"(The "-p" option tells cvs to prune empty directories and is optional). This creates a directory "genx/Codes". To update a working version of GEN2.5 that you checked out earlier, use the command "cvs update -d -p" in the genx/Codes directory (the "-d" option tells cvs to update new directories). The datasets and documents of GEN2.5 is located at genx/Data and genx/Docs, respectively. Typically, you do not want to check out genx/Data as a whole because it can contain gigabytes of data.

2 Building GEN2.5

You need to use gmake in genx/Codes to build GEN2.5. You can use the following options to customize the build:

- With/without debugging, controlled by DEBUG=1 or DEBUG=. Default is without debugging.
- With/without Charm++, controlled by CHARM=1 or CHARM=. Default is without charm.
- Static or dynamic linking, controlled by LIBSUF=a or LIBSUF=so. Default is dynamic linking. Static linking is only meant for debugging purposes.
- With/without Electric Fence, controlled by EFENCE=1 or EFENCE=. Default is without Electric Fence. E-fence is currently supported only with static linking, and cannot be used with Charm on Linux.
- Machine dependent options. On most platforms, it is set automatically. On IBM SP, please set OBJECT_MODE=32 or 64 to build 32- or 64- bit objects, respectively.

For example, the plain "gmake" on Linux builds GEN2.5 without debugging and without Charm++. "gmake DEBUG=1" builds GEN2.5 with debugging. Note that if you want to use the parallel version of Rocflu, which use Charm++, you must build with the options "CHARM=1". On a computer with multiple processors, you can build GEN2.5 in parallel using the option "-j n". For example, "gmake -j 4" will compile four files simultaneously.

If the build was successful, you should get an executable named genx.x (or two executables genx.x_flo and genx.x_flu if linked statically), one shared library (.so) per module in the lib subdirectory, and the utilities for Rocflo, Rocflu, and Rocface in the utilities subdirectory. You can test genx.x in the bare-bone mode (see Section 4) in the genx/Codes directory, without any input datasets of physics modules.

External libraries GEN2.5 uses two external libraries: Charm and HDF. Charm is needed only when built with CHARM=1. The makefiles of GEN2.5 try to locate these libraries from your home directory and system directories automatically. If Charm or HDF were not installed (or could not be found) on your system, you need to install the correct versions in your home directory under ~/charm and ~/HDF, respectively.

Please note that on Linux, genx needs to use SMP version of Charm (for example, net-linux-smp or mpi-linux-smp). On IBM SP, genx uses mpi-sp or mpi-sp-cc64 versions for 32- or 64-bit objects, respectively, so it is recommended to build both Charm versions using commands "./build FEM mpi-sp -O"and "./build FEM mpi-sp cc64 -O" under the Charm directory. See http://charm.cs.uiuc.edu for more detail.

3 Command-line options

The behavior of GEN2.5 is controlled by a combination of command-line options and control files. GEN2.5 recognizes the following command-line options:

- -v n: This controls the verbose level of Roccom. If n is a positive number, then Roccom will print out traces of COM_call_functions up to depth $(n+1)/2$. If n is an odd number, Roccom will print only the name of the function invoked by COM_call_function; if n is even, Roccom will also print the data types and values of the arguments passed to the function.

4 Control files

4.1 Main control file

The GENXControl.txt is the main control file that specifies the computational and IO modules to be used and controls time stepping. A sample GENXControl.txt is available in the genx/Codes directory. The first line of the file looks like

```
FullyCoupled Rocflo Rocfrac RocburnAPN Rocpanda
```

which indicates the modes of the manager, fluids, solids, and combustion codes, and the IO module, respectively. The manager's mode can be one of the following:

1. BareBone: Loads no computational modules and hence requires no input data. This is useful for debugging the driver and checking system environment.
2. FluidAlone: Loads only fluid and combustion modules (i.e., no solids).
3. SolidAlone: Loads only solid modules (i.e., no fluids and combustion).
4. FullyCoupled: Loads fluids, solids, and combustion modules.

Fluid mode is one of the following: Rocflo, RocfloDummy, Rocflu, and RocfluDummy. For the former two, Rocflo will be loaded; for the latter two, Rocflu will be loaded. Solid mode is either Rocfrac, RocfracDummy, Rocsolid, or RocsolidDummy. Combustion mode is either RocburnAPN, RocburnPY, or RocburnZN.

The fifth string of the first line specifies the IO module, which can be Rochdf, Rocphdf, and Rocpanda; If omitted, then Rochdf will be used by default, for compatibility with older versions. If Rocpanda is used, which is recommended for production runs on large number of processors, the RocpandaControl.txt file must present (see below). Another possible choice is Rocphdf, which uses pthread-version of Rochdf for background I/O of HDF files. It is recommended for debug runs on turing or interactive runs on production platforms, when the number of output files is not an issue or it is difficult to obtain additional processors for Panda servers.

4.2 Rocman

Rocman's control file is Rocman/RocmanControl.txt. A sample control file looks like as follows, which is also available at Codes/Rocman/RocmanControl.txt.

```
1 # Order of interpolation
1, 8.501e6 # Traction mode (1=pressure, 2=tractions), ambient pressure
1703.0 # Solid density for fluid-alone mode
1 2 100 1.e-6 # Data transfer parameters: verbose level, order of quadrature rules, max iterations, tolerance
    for iterative solver
```

The control file has four lines. The first line is either 0 or 1, indicating the order of interpolation (constant or linear; the default is 0). The second line starts with 1 or 2, indicating the mode of load transfer (no-sheer or with-sheer; the default is 1), optionally followed by the ambient pressure (the default is 0). The third line is the solid density (the default is 1703), which is used only in the FluidAlone mode. Optionally, the fourth line is the data transfer parameters, including the verbose level (default 1), order of quadrature rules (default 2), the maximum number of iterations (default 100), and the tolerance for the iterative solver (default 1.e-6). If the file is missing, the default values will be used.

4.3 Rocpanda

Its control file RocpandaControl.txt contains four lines:

```
C <#client processes>
S <#server processes>
M 1
D . d
```

The first two lines controls the number of client (computational) and server (Panda) processes, respectively. The third line indicates whether the servers should be scattered (M 1) or next to each other (M 0). The last line indicates the output directory, which typically should be "D . d" (current directory) for GEN2.5.

A sample RocpandaControl.txt is available in the genx/Codes/Rocpanda directory.

4.4 Computational-module control files

Each computational module has its own control file in the directory named after its module name. For example, the control file for Rocfrac is in the RocfracDummy or Rocfrac subdirectory (depending on whether it is running in dummy mode or not). The three Rocburn modules uses different control files, whose samples are available at their corresponding subdirectories in genx/Codes/Rocburn.

For Rocface, the input files (the overlay mesh) for the Rocflo and Rocfrac coupling should be in the directory Rocface/RocfloRocfrac. Likewise for the Rocflu and Rocsolid coupling, the files should be in Rocface/RocfluRocsolid. Note that for single-processor runs, Rocface does not read in the overlay mesh but write them out into the aforementioned subdirectories for restart. So it is important that the Rocface subdirectories exist.

5 Output conventions

In GEN2.5, all HDF I/O are done by Rochdf or Rocpanda invoked by Rocman. For a module X, Rocman writes HDF files into the directory X/HDFout. If the directory cannot be accessed, Rochdf/Rocpanda will try to write to the current directory instead. If the above attempt fails, then Rochdf will print out warning messages without aborting the execution, but you will not get output for the corresponding module and will not be able to restart. The filenames are encoded as

```
<prefix>_<timelevel>_<procrank>.hdf
```

where time level has format $xx.yyyyyy$, which correspond to $0.yyyyyy \times 10^{xx}$ nanoseconds. For Rochdf, the procrank correspond to the MPI ranks of the computational processes; for Rocpanda, they correspond to those of the Panda server processes. Note that the Rocburn HDF files uses the meshes stored in Rocflo/Rocflu HDF files. Therefore, if you move Rocburn HDF files elsewhere for visualization, you need to make sure to copy fluid HDF files as well and also to preserve the directory structure.

Timing data go into files <dir>/GENXTimingData????.txt, where <dir> is specified in GENXControl.txt. If the directory cannot be accessed, then GEN2.5 will write timing data to the current directory.

6 Restart

When solution files are written, the driver also writes an ASCII file GENX.restart.info, which contains a list of iteration indices and times when solutions were written out. To restart an execution, set the initial time to nonzero in GENXControl.txt. Then GENX will read the last line of GENX.restart.info and corresponding solution files to restart the execution.