LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-558859

Adding a MOAB Geometry Interface to SHARP Structural Mechanics

R.M. Ferencz and N.E. Hodge
Methods Development Group

May 29, 2012

Memo of Completion
DOE-NE NEAMS Milestone M3MS-12LL0603031

Auspices

Lawrence Livermore National Laboratory is operated by Lawrence Livermore National Security, LLC, for the U.S. Department of Energy, National Nuclear Security Administration under Contract DE-AC52-07NA27344.

Disclaimer

Adding a MOAB Geometry Interface to SHARP Structural Mechanics

A Memo of Completion for NEAMS FY12 Milestone M3MS-12LL0603031

R.M. Ferencz and N.E. Hodge
Methods Development Group

Abstract

We briefly summarize the development of, and test experience with, an initial data interface between the structural mechanics code Diablo and the SHARP reactor simulation system data hub MOAB. That interface has been exercised both to write MOAB databases from Diablo, and then also to use such a database to read in part of a simulation definition for a subsequent Diablo execution. All enhancements are integrated into the central Diablo source repository.

Introduction

The SHARP software system for advanced simulation of nuclear reactors and power plant systems is sponsored by DOE's Nuclear Energy Advanced Modeling & Simulation (NEAMS) program. Led by Argonne National Laboratory (ANL), SHARP has been architected as a federation of single-physics simulation tools to permit flexibility in programming languages and leveraging of past and on-going investments. Solution of multi-physics problems will be coordinated by, and data passed through, a central 'hub'. SHARP's hub implementation is utilizing MOAB: a Mesh-Oriented datABase [1]. This same data hub approach is also intended to enable multi-resolution simulations, e.g., lower-dimension plant-scale simulations can be informed by high-fidelity 3D models of particular critical components.

The structural mechanics module of SHARP will leverage the Diablo code from LLNL. The code has coupled thermo-mechanical capabilities for nonlinear solid and structural mechanics, making it a good complement with SHARP's thermal-hydraulic (e.g., Nek5000) and reactor transport (e.g., Proteus) modules. As a first step in the re-engagement of the LLNL team with the SHARP project, we have an initial deliverable to establish a data interface between Diablo and MOAB. In the PICS:NE project management system, this deliverable is designated M3MS-12LL0603031. This brief report serves as a record of completion of that Level 3 milestone.

Approach

We decided it was best to first become familiar with MOAB outside the context of interfacing with Diablo. While MOAB has its own C++ interface protocol, it also provides an ITAPS-compliant C interface that is the recommended pathway for

Fortran codes. ITAPS [2], or *Interoperable Technologies for Advanced Petascale Simulations*, was a multi-year effort sponsored by the DOE's Scientific Discovery through Advanced Computing (SciDAC) program. The ITAPS project defined an Application Programming Interface (API) called iMesh for libraries to store and retrieve unstructured mesh representations of computational domains. It also considered APIs for classes of supporting numerical operations such as mesh partitioning and mesh quality improvement. Reference implementations, or adaptation of existing libraries to the associated ITAPS interfaces, were also produced.

We began with MOAB version 4.1, originally released in August 2011. We found it straightforward to work through a progression of builds, first with the default GNU compilers, then the Intel and PathScale compilers typically used for Diablo. The simple demo codes provided with the MOAB distribution, and from the ITAPS website, were the basis for experimentation. In particular the `FindConnect.F90` program was extended to add extra nodes ("vertices"), then 1D, 2D and 3D example element connectivities to the exterior of the existing mesh in the file `125hex.vtk`. The results of these operations could be verified by writing out the revised mesh to another database and viewing it with the Visit [3] visualization tool. Both the example's source "`VTK`" format and MOAB's native "`.h5m`" database were exercised in this manner. (The MOAB native database was read into Visit using the latter's `ITAPS_MOAB` reader option.) The progression just described was later repeated when MOAB version 4.5.0 was released in March 2012.

With some confidence in our understanding of the iMesh interface and the basics of MOAB, we proceeded to interface Diablo. Our strategy was to first enable Diablo to *write* its mesh definition, read from its standard text input files, to a native MOAB database. Visualization of the resulting mesh database in Visit permitted rapid debugging of the interface calls. With that stage complete, we could then begin to selective add *read* functionality so that for some of its inputs, Diablo would read mesh definition data from an existing MOAB database. To ensure this was indeed the mode of operation, a revised Diablo input would be created for test problems wherein most of the nodal coordinate information would be deleted. Thus the simulation could only proceed correctly if indeed the proper mesh definition was being read from the MOAB database.

Example Result

Multiple tests cases were run to confirm the initial interface was operating correctly. The largest test case exercised was the ABTR core model created for an FY08 demonstration of fast reactor duct assembly bowing due to thermal distortions. Figure 1 shows the all-hexahedral mesh as rendered by Visit after reading it from a MOAB database produced by Diablo. Due to the binary data format of the MOAB database, the mesh file in that format required only about half the disk storage of the standard text input. Note that in order for Visit to read all our MOAB databases,

it was necessary to invoke the HDF5 option `HDF5_DISABLE_VERSION_CHECK` to avoid complaints that Visit was performing HDF5 reading using a slightly different version of the library than what was linked with Diablo and MOAB for writing.

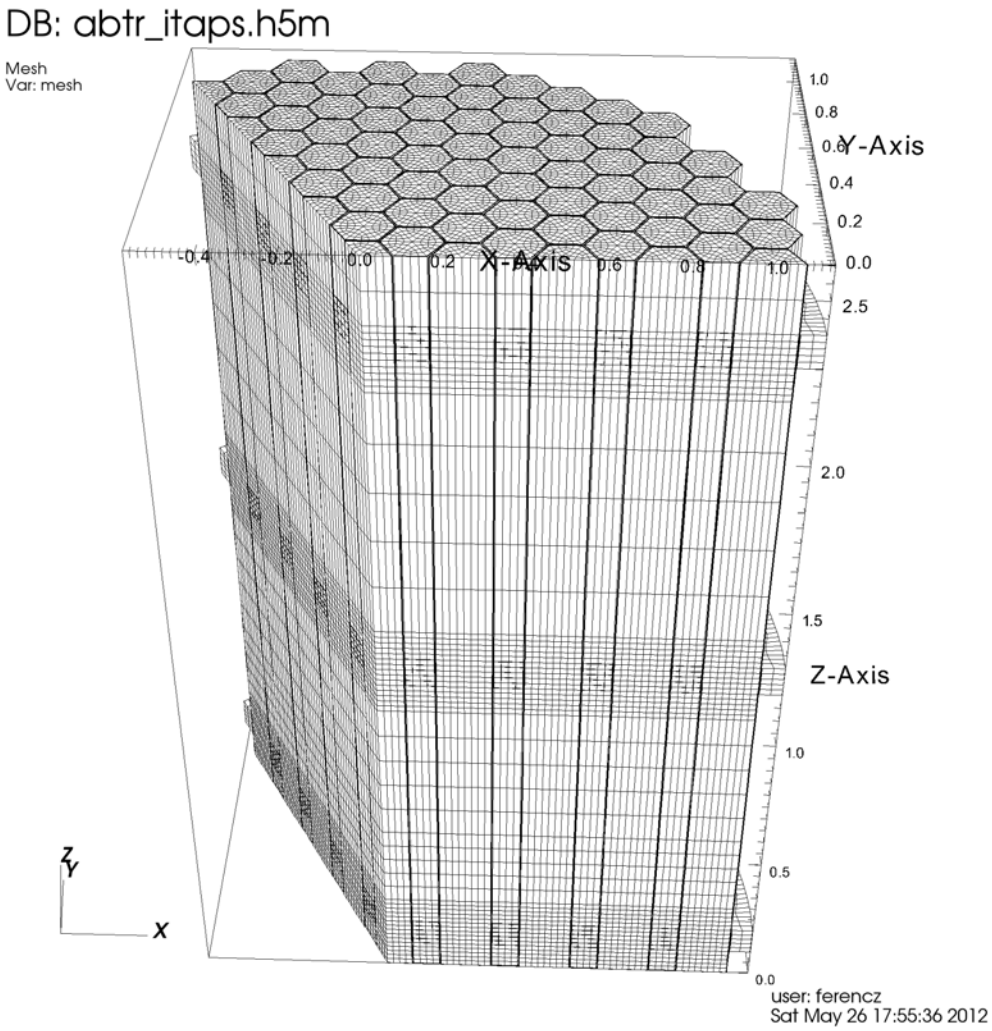

Figure 1. A structural finite element mesh for one-third of the ABTR conceptual design core geometry as rendered in Visit after reading a MOAB database written by the Diablo code.

Having created the MOAB mesh database for the ABTR core bowing model, we then ran the thermal part of the analysis using that database to supply mesh information. The thermal boundary conditions, still defined in the original input, were those provided by ANL [4] as part of the original demonstration. We chose to run thermal-only because it is such a quick computation, and it already provides the needed check that the geometry has been correctly imported through the MOAB interface.

Figure 2 illustrates a typical result, with the highest temperatures in the central fuel assemblies and the inner surface of the shield ring. These Diablo simulations were verified up to 512-way parallel executions.

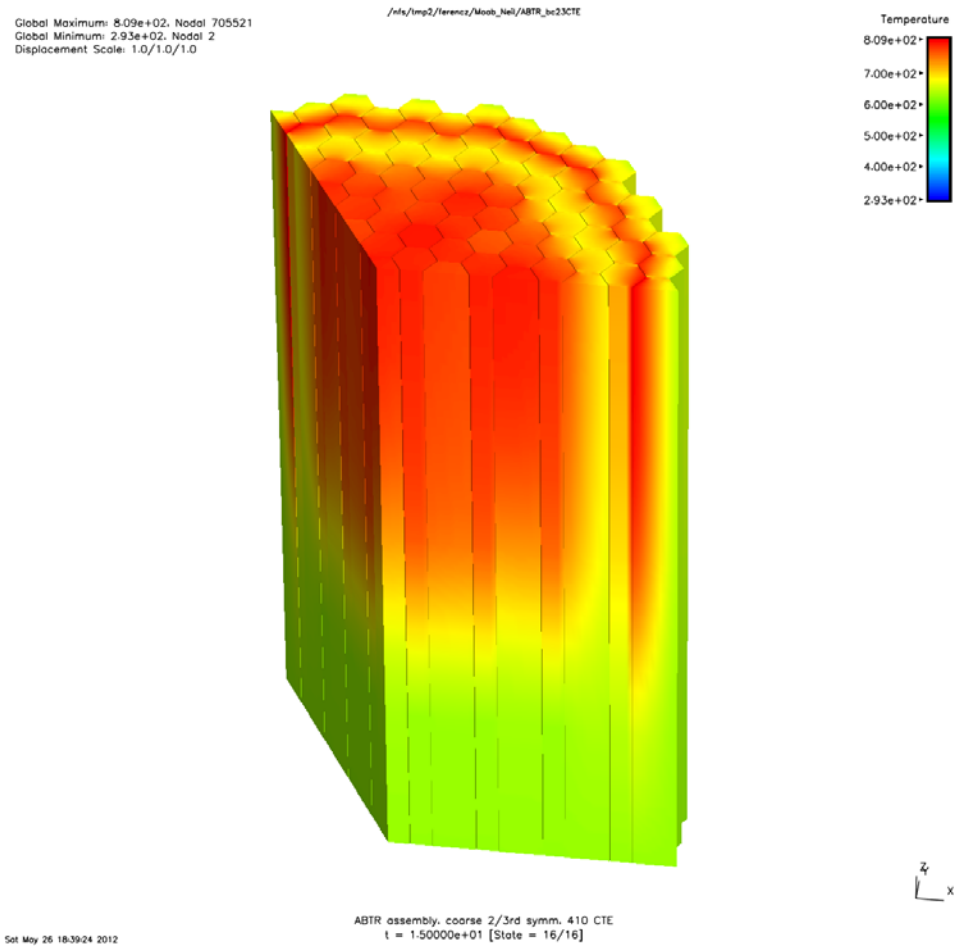

Figure 2. Temperature distribution (K) in the ABTR conceptual design core geometry, at power-to-flow ratio 1.0. Thermal-only response computed with Diablo using duct face tempera-ture boundary condition data provided by Dr. Won-Sik Yang, then of Argonne National Laboratory.

Lessons Learned

An interesting fault condition was noted from one Diablo test problem. That simple test uses a 16 x 16 x 16 uniform hex grid for the so-call Boussinesq problem of a concentrated load on an elastic half-space. Diablo is intended to permit arbitrary node numberings to permit analysts flexibility in making simple alterations or perhaps merging two meshes together. Thus this test problem had been

purposefully altered to not have the nodes defined in 1–*N* order, although in their totality the nodes did represent a complete set {1,2,…,*N*}. The resulting MOAB database written by Diablo was readily viewed in Visit. However, upon launching a simulation using that geometry, the analysis immediately terminated, complaining of non-positive element volumes. This experience led to the realization of implicit 1–N ordering. To test that hypothesis, a revised Diablo input was created where the nodes had been sorted to a monotonic 1,2,…,*N* order. Generating a Moab database from that input then supported the successful simulation execution of Diablo. In the future, we will consider extending the Diablo write function for MOAB vertices to output a sorted, monotonic order, regardless of the user's input. Where 'gaps' in the order exist, and full generality is desired, we may need to simply define the missing nodes with an arbitrary coordinate location even if no element connectivities reference those nodes.

Conclusions and Future Efforts

An initial data interface between the Diablo structural mechanics code and the MOAB database library has been established in fulfillment of the L3 milestone. We cannot claim expertise with MOAB yet, but we are confident that we have a strong basis for technical discussions with our SHARP colleagues and further development. Our work to date has included some initial efforts at using 'tags' to designate subsets of data. We need to acquire an understanding of the usage of tags being developed within the SHARP team to coordinate the interchange of data between physics modules. This process will begin to unfold as we work on our next deliverable: a bi-physics demonstration for the thermal-structural response of a pipe to a thermal mixing flow downstream of a Tee junction. We envision using tags to identify the thermal boundary conditions (locations and values) needed from the CFD module. Our understanding is that SHARP will eventually explore the use of the iFields capability under development in MOAB to streamline the designation and proper identification of such exchange quantities.

References

[1] T. Tautges *et al*., "MOAB: A Mesh-Oriented datABase", Argonne National Laboratory, no date. Web 28 May 2012 http://trac.mcs.anl.gov/projects/ITAPS/wiki/MOAB.

[2] L. Diachin *et al*., "ITAPS: Interoperable Technologies for Advanced Petascale Simulations ", Rensselaer Polytechnic Institute, 13 Dec 2010. Web 28 May 2012 http://www.itaps.org/.

[3] Visit, Lawrence Livermore National Laboratory, UCRL-WEB-229972, 14 May 2012. Web 28 May 2012 https://wci.llnl.gov/codes/visit/.

[4] W-S. Yang, Private Communication "Temperature fields for structural analysis", June 2007.