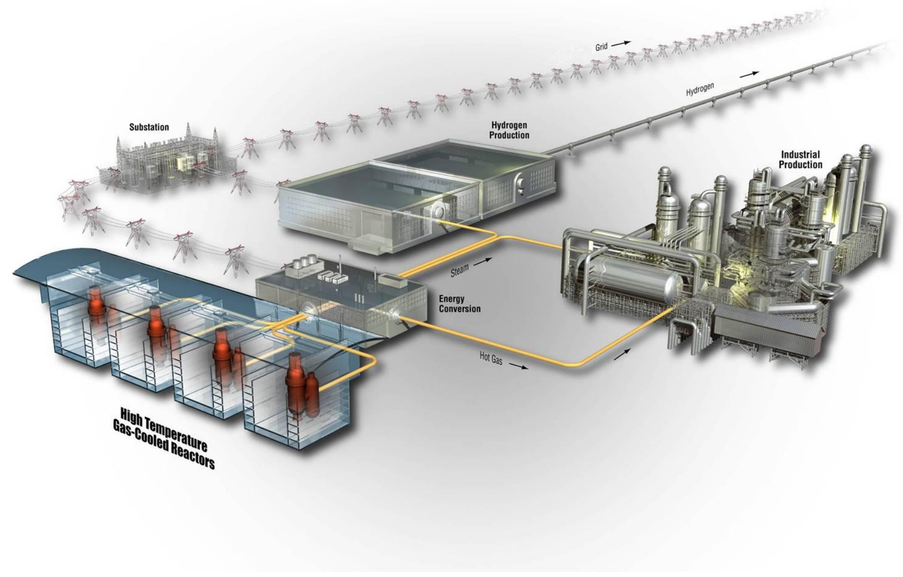
INL/EXT-12-27171

SHARP/PRONGHORN Interoperability: Mesh Generation

Avery Bingham
Javier Ortensi
Rajeev Jain
Iulian Grindeanu
Tim Tautges

September 2012



Idaho National Laboratory

INL/EXT-12-27171

SHARP/PRONGHORN Interoperability: Mesh Generation

INL: Avery Bingham and Javier Ortensi
ANL: Rajeev Jain, Iulian Grindeanu, and Tim Tautges

September 2012

Idaho National Laboratory
VHTR Program
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

VHTR Program

SHARP/PRONGHORN Interoperability: Mesh Generation

INL/EXT-12-27171

September 2012

Approved by:

Jason D. Hales FOR RCM
Richard Martineau
Dept. Manager, Fuel Modeling and Simulation

JDH 9/26/12
26 ~~Sep~~ 2012
Date

[illegible]
Hans Gougar
VHTR TDO Deputy Technical Director

9/26/12
Date

[illegible]
David Jensen
VHTR TDO QA

9/26/12
Date

[illegible]
Diane Croson
VHTR TDO Deputy Director

9/26/12
Date

SUMMARY

Progress toward collaboration between the SHARP and MOOSE computational frameworks has been demonstrated through sharing of mesh generation and ensuring mesh compatibility of both tools with MeshKit. MeshKit was used to build a three-dimensional, full-core very high temperature reactor (VHTR) geometry with 120-degree symmetry, which was used to solve a neutron diffusion critical eigenvalue problem in PRONGHORN. PRONGHORN is an application of MOOSE that is capable of solving coupled neutron diffusion, heat conduction, and homogenized flow problems. The results were compared to a solution found on a 120-degree, reflected, three-dimensional VHTR mesh geometry generated by PRONGHORN. The ability to exchange compatible mesh geometries between the two codes is instrumental for future collaboration and interoperability. The results were found to be in good agreement between the two meshes, thus demonstrating the compatibility of the SHARP and MOOSE frameworks. This outcome makes future collaboration possible.

CONTENTS

FIGURES

TABLES

ACRONYMS

ANL	Argonne National Laboratory
INL	Idaho National Laboratory
JFNK	Jacobian Free Newton Krylov
MOAB	mesh-oriented database
MOOSE	Multi-Physics Object-Oriented Simulation Environment
PMR	prismatic modular reactor
SHARP	Simulation-based High-efficiency Advanced Reactor Prototyping
VHTR	very high temperature reactor

SHARP/PRONGHORN Interoperability: Mesh Generation

1. INTRODUCTION

The computational analysis of complex physical systems (such as nuclear reactors) requires the ability to solve the behavior of several different physical phenomena (such as neutron transport, heat transfer, fluid flow, and thermal expansion) and to compute interactions among these physical phenomena. The goal of developing a computational framework capable of complex multiphysics analysis has been pursued independently at Argonne National Laboratory (ANL) and Idaho National Laboratory (INL). Essentially, there are two approaches to achieve this goal: (1) build a framework that can be used as an interface for existing codes, or (2) develop an application environment framework in which new physics could be added and coupled in a highly integrated fashion.

Historically, physics codes have been written to solve for a single physical phenomena such as neutron transport. Many of these codes have years, if not decades, of development and have been validated against experiments or verified against other codes. The Simulation-based High-efficiency Advanced Reactor Prototyping (SHARP) computational framework has managed to leverage existing codes as modules and provides the interface for communicating data among a number of separate physics solvers. Coupling independent codes is not new; however, the SHARP project has greatly reduced the difficulty associated with the coupling of separate physics solvers for reactor simulation.

In the Multiphyiscs Object-Oriented Simulation Environment (MOOSE) framework a different approach is taken. Selection of a mathematical model or set of governing equations is a task left to the scientist and can be adapted specifically to individual needs. Translating a mathematical formulation into an efficient computer program can be challenging. However, MOOSE provides a simple and flexible interface that allows for rapid formulation of mathematical models into code that is inherently parallel and portable from desktop machines to high-performance computers. MOOSE functions as a software library, which interfaces user-controlled physics applications with numerical libraries such as PETSc, while treatment of the mesh is handled by the libMesh library.

In MOOSE, a system of partial differential equations is solved using the Jacobian Free Newton Krylov (JFNK) method. Separate physics processes are encapsulated into kernels with each kernel responsible for evaluating its portion of the nonlinear residual and the preconditioning matrix or approximate Jacobian. Each kernel is implemented separately in order to provide flexibility to test different combinations of physics. By structuring the code this way, models of physical phenomena can be rapidly turned into code. This approach allows the scientist to test new models quickly while focusing on the physics being modeled.

One application of MOOSE called PRONGHORN is a tightly coupled multiphysics application code, which is capable of fully implicit calculations for analysis of nuclear reactors. Even though current development is focused on the analysis of very high temperature reactors (VHTR) (both pebble bed and prismatic core concepts), the scope of the code can be expanded to other reactor designs.

Code-to-code solution comparisons are simplified when code systems use the same or similar meshes. In this way, results from an ANL tool such as SHARP can be easily compared to or integrated with tools like INL's PRONGHORN. MOOSE does have some limited mesh generation capabilities, but it relies on separate applications such as Cubit for complex mesh generation. There are various tools available that can generate the geometry and associated mesh. MeshKit is one such tool that was developed as a module under the SHARP project at ANL. MeshKit is now offered as an open-source toolkit for mesh generation. The first step for interoperability between SHARP and PRONGHORN is to demonstrate the ability to operate on the same mesh. This report is primarily concerned with compatible mesh preparation between the two computational frameworks on similar representative geometries. Differences in the mathematical models and numerical solvers will not be dealt with extensively at this time.

2. OBJECTIVE

The primary purpose of this work is to show the interoperability of the meshing tools in MeshKit with the MOOSE computational framework in use at INL and, specifically, with the PRONGHORN application. The task has three benefits: (1) utilization of work already performed within the Department of Energy system, (2) simplification of the geometric modeling and the preparation analysis meshes for reactor applications at INL, and (3) allowance of future use of SHARP in comparison with INL tools on the same computational mesh.

Building a VHTR geometry mesh with MeshKit and solving the full-core neutron diffusion problem in PRONGHORN will satisfy the first steps toward code interoperability. Verification of the results will be carried by comparing the results with the new MeshKit mesh to previous results obtained with a mesh developed at INL.

Finally, installation of MeshKit on the INL High-Performance Computing cluster makes it available for future use with MOOSE-based applications.

3. SHARP DESCRIPTION

SHARP is a reactor simulation system developed by ANL as part of the U.S. Department of Energy Nuclear Energy Advanced Modeling and Simulation Program. It serves is a suite of physics simulation software modules and computational framework components that enables the user to evaluate performance and safety characteristics of nuclear reactors and their components. Essentially, SHARP models the physical processes that occur in a nuclear reactor core. The physics modeled include neutron transport with the PROTEUS suite of codes, thermal fluids with codes such as Nek5000 and Star CCM+, and other assorted physics such as fuel and structure behavior.

SHARP is a weakly coupled framework that builds on existing computer codes. SHARP allows users to attach new simulation modules to these older legacy codes, which avoids significant rewriting of code. Solutions found by the various physics codes must share data through a coordinated interface. This interface in SHARP is provided by the Mesh-Oriented Database (MOAB). MOAB is a library for representing unstructured and structured mesh and field data on a mesh. This interface allows for multi-resolution solutions, meaning low-dimension, plant-scale solvers can be informed by solutions from high-fidelity models. In addition, it allows for different physics to be modeled on separate mesh but enables them to share solution data. MOAB implements the Interoperable Technologies for Advanced Petascale Simulations iMesh interface. iMesh is a common interface to mesh data implemented by several different packages, including MOAB. Various tools (such as smoothing, adaptive mesh refinement, and parallel mesh communication) are implemented on top of iMesh. MOAB supports common parallel mesh operations such as parallel import and export (to/from a single HDF5-based file), parallel ghost exchange, communication of field data, and general sending and receiving of mesh and metadata between processors.

MeshKit is a module of SHARP in development at ANL, which is designed to facilitate generation of nuclear reactor geometries and the associated mesh.

4. MESHKIT DESCRIPTION

Nuclear reactor geometry can often be described as having a two-level hierarchy of lattices; the first level of the hierarchy corresponds to fuel assemblies, formed as a lattice of cylindrical pins or compacts, while in the second level, assemblies of various types are arranged in a lattice to form the reactor core. These pins typically contain uranium-based fuel, absorbing material for controlling the nuclear chain reaction, or instrumentation. The surrounding materials can function as coolant, a neutron energy moderator, or simply supporting structure. These materials are typically arranged in either a rectangular lattice for water-cooled reactors, or a hexagonal lattice for sodium and gas-cooled reactors. Assemblies vary by degree of uranium enrichment in the fuel material, type of control rod material, or other parameters.

Generation of geometry and mesh models for reactor cores can be a difficult process. Advantage can be taken of the structure inherent in this two-level hierarchy to automate much of the generation process, which can be augmented by user interaction at key points. MeshKit seeks to be the ideal geometry and mesh generation tool by attempting to balance both lattice-guided automation and allow opportunities for user interaction at key points in the process. Based on a small set of parameters, geometry and mesh are constructed for a variety of reactor core types arranged as both square and hexagonal lattices. MeshKit also has general mesh manipulation and generation functions such as copy, move, rotate, and extrude. In addition, new quad mesh and embedded boundary Cartesian mesh algorithms have been developed and can interface with several public-domain tetrahedral meshing algorithms (e.g., Gmsh and netgen).

5. PRONGHORN DESCRIPTION

PRONGHORN is a multiphysics reactor analysis application of MOOSE. PRONGHORN was initially developed at INL to model the pebble bed gas-cooled reactor using a two-group neutron diffusion model and a porous media flow model. Recent development in PRONGHORN has extended the neutron diffusion model to an arbitrary number of groups and extended the thermal fluid model to better capture the physics in prismatic VHTR. Just like all MOOSE application, problems are solved using the JFNK method. PRONGHORN can be run in serial or on massively parallel computers with one-, two-, three-, or axisymmetric RZ-geometry.

Current capabilities of PRONGHORN include solving steady-state and transient–coupled, homogenized, fluid flow-heat transfer problems and standard multigroup diffusion problems (fixed-source, criticality, and time-dependent). For the purpose of demonstrating the compatibility of meshes generated from MeshKit, the physics solved in this report will be limited to the k-eigenvalue calculation using the neutron diffusion approximation.

5.1 Numerical Method: Jacobian-free Newton-Krylov

The JFNK method is used to solve nonlinear systems of equations. JFNK is a combination of the Newton and Krylov methods. This method enables an algorithm to retain the quadratic convergence rate without forming a complicated Jacobian matrix typically required by the Newton method. In general, a nonlinear partial differential equation (PDE) can be written in the form:

$$F(U) = 0$$

where F is the nonlinear residual function and U a vector of independent variables, respectively. The first-order Taylor expansion about the previous iterate, U^k, can be expressed as

$$F(U) \approx F(U^k) + \left.\frac{\partial F}{\partial U}\right|_{U_k} (U - U^k) = 0,$$

which can be arranged into a linear system

$$J\partial U^k = -F(U^k)$$

where

$$J = \frac{\partial F}{\partial U}.$$

The Jacobian matrix, J, is difficult or impossible to compute analytically for complex multiphysics problems. The linear system above is solved using a Krylov method (typically GMRES) and the solution is updated as

$$U^{k+1} = U^k + d\partial U$$

where d is a scalar damping parameter $0 < d \le 1$, which is found using an optimization algorithm (line search or trust region) to minimize the residual. JFNK takes an advantage of the fact that a Krylov method merely requires a matrix-vector product, not the matrix itself, and that the following finite difference form can approximate the matrix-vector product as

$$Jv \approx \frac{F(U+hv)-F(U)}{h},$$

where h is a perturbation parameters. To solve the linear system efficiently, we must apply preconditioning. The preconditioning matrix M is often created by choice of linearization. The right-preconditioned system can be written as

$$JM^{-1}M\partial U^k = -F(U^k).$$

The matrix-vector product for the (right) preconditioned system becomes

$$JM^{-1}v \approx \frac{F(U+hM^{-1}v)-F(U)}{h}.$$

One can approximate the matrix-vector product with the following two steps:

1. $y = M^{-1}v$
2. $Jy \approx \dfrac{F(U+hy)-F(U)}{h}$,

Therefore, preconditioned Krylov iterations require the additional operation of Step 1. We choose the preconditioning matrix to be a simpler form of the original Jacobian matrix J (typically the main diagonal of the Jacobian matrix). Therefore, the explicit form of the Jacobian matrix is not required to solve the preconditioned system.

5.2 Mathematical Model for Core Neutronics

The governing equation for the k-eigenvalue calculation using the diffusion approximation for energy group g out of G total groups is the following:

$$-\nabla \cdot D_g \nabla \phi_g + \Sigma_{r,g}\phi_g - \sum_{\substack{g' \\ g' \neq g}}^{G} \Sigma_s^{g' \to g}\phi_{g'} = \frac{1}{k_{eff}} \sum_{g'=1}^{G} \chi_{g'g} \nu_{g'} \Sigma_{f,g'} \phi_{g'} \qquad g = 1, 2, \ldots, G$$

where each quantity is dependent on $\vec{r}$ and are defined as follows:

D_g	= the neutron diffusion coefficient $\left[cm\right]$
ϕ_g	= the neutron flux $\left[\frac{neutrons}{cm^2 s}\right]$
$\Sigma_{r,g}$	= the macroscopic removal cross section $\left[cm^{-1}\right]$
k_{eff}	= the critical eigenvalue or multiplication factor
$\chi_{g'g}$	= fission neutron yield into neutron energy group g caused by a neutron in group g'
$\nu_{g'}$	= number of neutrons per fission caused by a neutron in group g'
Σ_{fg}	= the macroscopic fission cross section $\left[cm^{-1}\right]$.

In PRONGHORN, the critical eigenvalue is first updated with a few iterations analogous to the power method to ensure the flux solution is within the Newton method convergence radius. We complete these "power method" steps by using the JFNK method to solve for the flux just as one would normally use inner iterations to update the source, and the eigenvalue is only updated at each nonlinear or Newton step. Once a few "power method" iterations are completed, the nonlinear eigenvalue solver takes advantage of the JFNK method by allowing the flux solution and the critical eigenvalue to be updated during the linear residual evaluations of the Krylov solver accelerating the eigenvalue problem.

5.3 Finite Element Discretization of Physics

In this section, we discuss spatial discretization of the neutron diffusion equation. Most often a linear finite element discretization would be applied. To derive the weak formulation, the governing equation is multiplied by a test function Ψ and integrated over the volume. The weak form for group g of the neutron diffusion k-eigenvalue problem can be written as

$$F(\phi_g) = \int_V \nabla\Psi \cdot D_g \nabla\phi_g \, dV - \int_\Gamma \Psi D_g \nabla\phi_g \cdot \vec{n} \, ds + \int_V \Psi \Sigma_{r,g} \phi_g \, dV$$
$$- \int_V \Psi \sum_{\substack{g' \\ g' \neq g}}^{G} \Sigma_s^{g' \to g} \phi_{g'} \, dV - \frac{1}{k_{eff}} \int_V \Psi \sum_{g'=1}^{G} \chi_{g'g} \nu_{g'} \Sigma_{f,g'} \phi_{g'} \, dV .$$

The final step finite element in discretization is to approximate ϕ_g as the following:

$$\phi_g \approx \phi_{g,h} = \sum_{i=1}^{N} \phi_{g,i} b_i$$

where $\phi_{g,i}$ is the value of the flux at the i^{th} local node and b_i is the basis function that is zero at all nodes but one. There are many choices for the test and basis functions. Here the Galerkin Method is used, which means that our test function is taken from the same space as our basis function. The basis functions are typically first or second order Lagrange polynomials.

5.4 Compatible Mesh Generation

MeshKit was used to generate the 120-degree, symmetric, full-core mesh of a VHTR. Generating this mesh directly with Cubit would have been quite time consuming and currently is not included in PRONGHORN's mesh generation capability. Construction of the one-third core mesh was done using PRONHORN's built-in capability. Both core models are of a simplified version of the Modular High Temperature Gas-cooled Reactor (MHTGR) reactor design. The model includes 660 homogenized prismatic fuel blocks that are arrayed in 10 layers that form an annular ring in a hexagonal lattice. The blocks have a hexagonal pitch of 36 cm. Figure 1 shows the radial layout of the fuel.

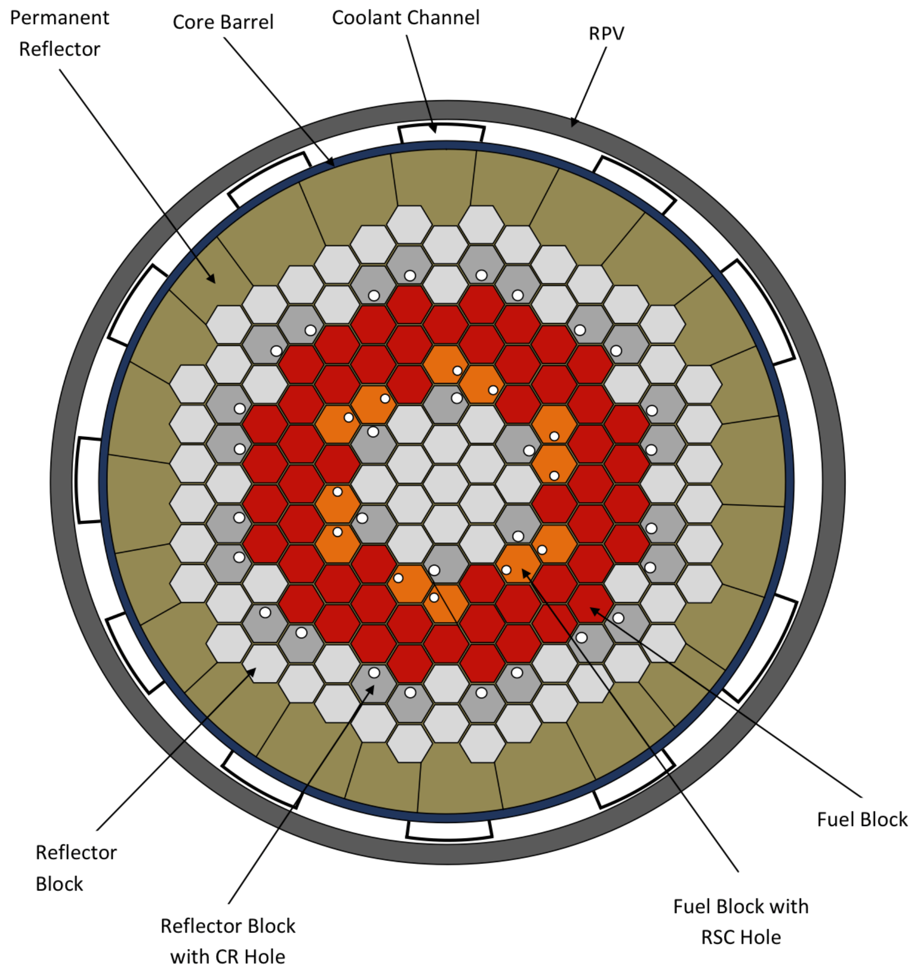


Figure 1. Core radial layout, fuel shown in red and orange.

The permanent reflector is approximated with the additional rings of hexagonal graphite blocks. Reflector blocks are modeled above and below the active core region, which is composed of 14 layers. The height of each layer is included in Table 1. The fuel is located in layers 3 through 12. The base mesh generated by MeshKit and PRONGHORN are presented in Figure 2.

Table 1. Axial core layers.

Absolute	Relative	Δz	
1303.74	1110.18	39.65	Layer 14
1264.09	1070.53	79.30	Layer 13
1184.80	991.24	79.30	Layer 12
1105.50	911.94	79.30	Layer 11
1026.20	832.64	79.30	Layer 10
946.90	753.34	79.30	Layer 9
867.60	674.04	79.30	Layer 8
788.30	594.74	79.30	Layer 7
709.00	515.44	79.30	Layer 6
629.70	436.14	79.30	Layer 5
550.41	356.85	79.30	Layer 4
471.11	277.55	79.30	Layer 3
391.81	198.25	99.20	Layer 2
292.61	99.05	99.05	Layer 1

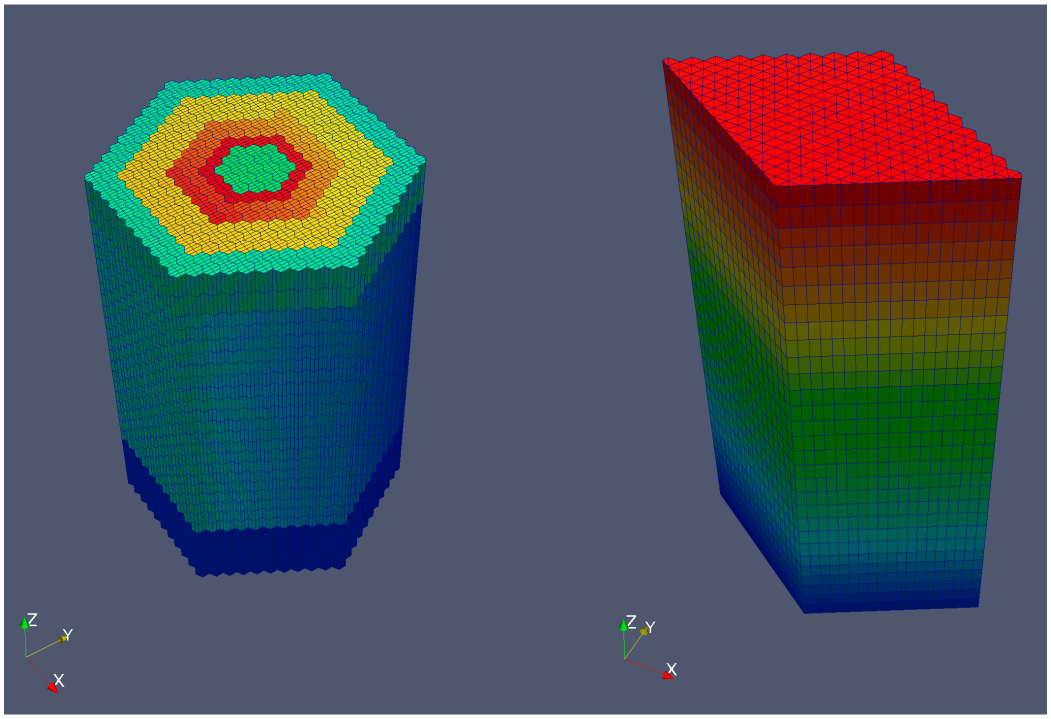


Figure 2. Base mesh generated by MeshKit (left) and PRONGHORN (right).

6. RESULTS

A two-group diffusion approximation using condensed cross-sections from the 26-group MHTGR benchmark problem was used to test the meshes. The meshes were spatially refined to achieve a spatially converged solution of approximately 10 pcm (per cent mili). The full convergence results are presented in Table 2.

Table 2. Comparison of results from Idaho National Laboratory and Argonne National Laboratory-generated mesh run with PRONGHORN.

Uniform Refinement	One-Third Core Mesh (INL)		Whole Core Mesh (ANL)		pcm Difference
	Keff	dofs	Keff	dofs	(ANL-INL)/ANL*1e5
Base mesh	0.97084	1.42E+05	0.97188	1.06E+06	107.0
1	0.96948	1.09E+06	0.97001	8.50E+06	54.6
2	0.96914	8.34E+06	0.96942	6.80E+07	28.9
3	0.96906	6.40E+07	0.96922	5.44E+08	16.5

The results demonstrate mesh compatibility between the MOOSE and SHARP frameworks. PRONGHORN was able to converge to the same solution for meshes generated by either MeshKit or PRONGHORN. The thermal and fast flux solutions for the full core model are shown in Figure 3. The thermal and fast flux solutions for the one-third core are shown in Figure 4.

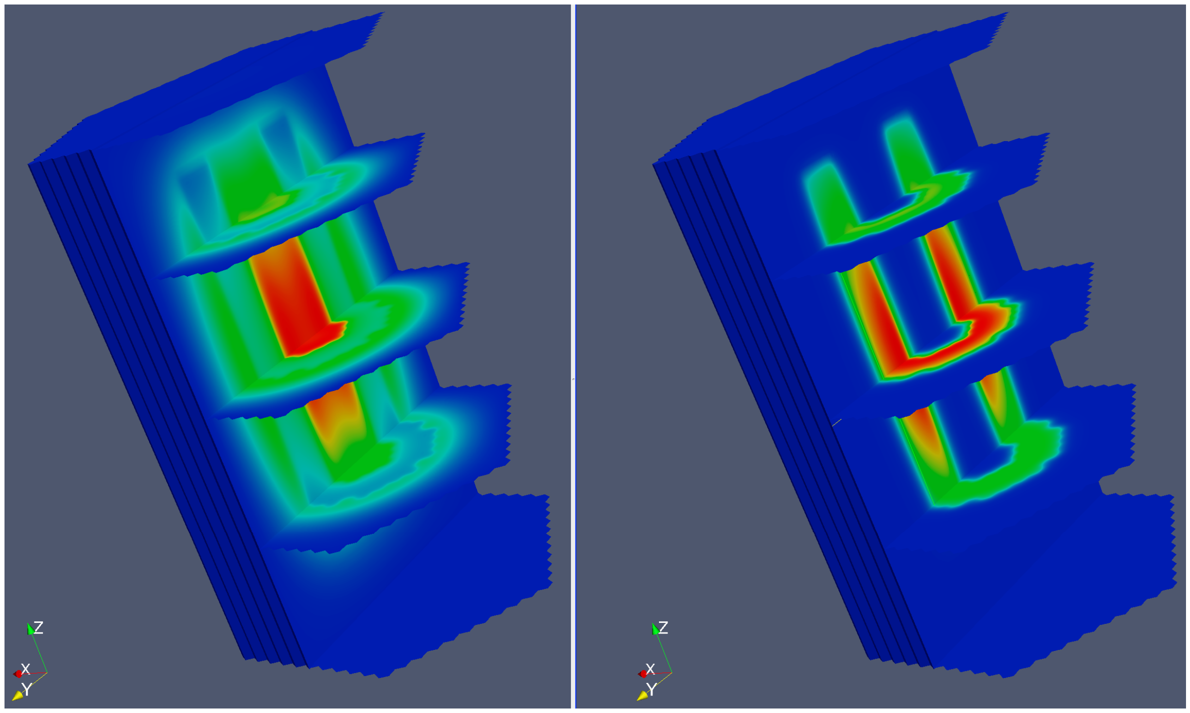


Figure 3. Whole core thermal (left) and fast (right) flux solutions.

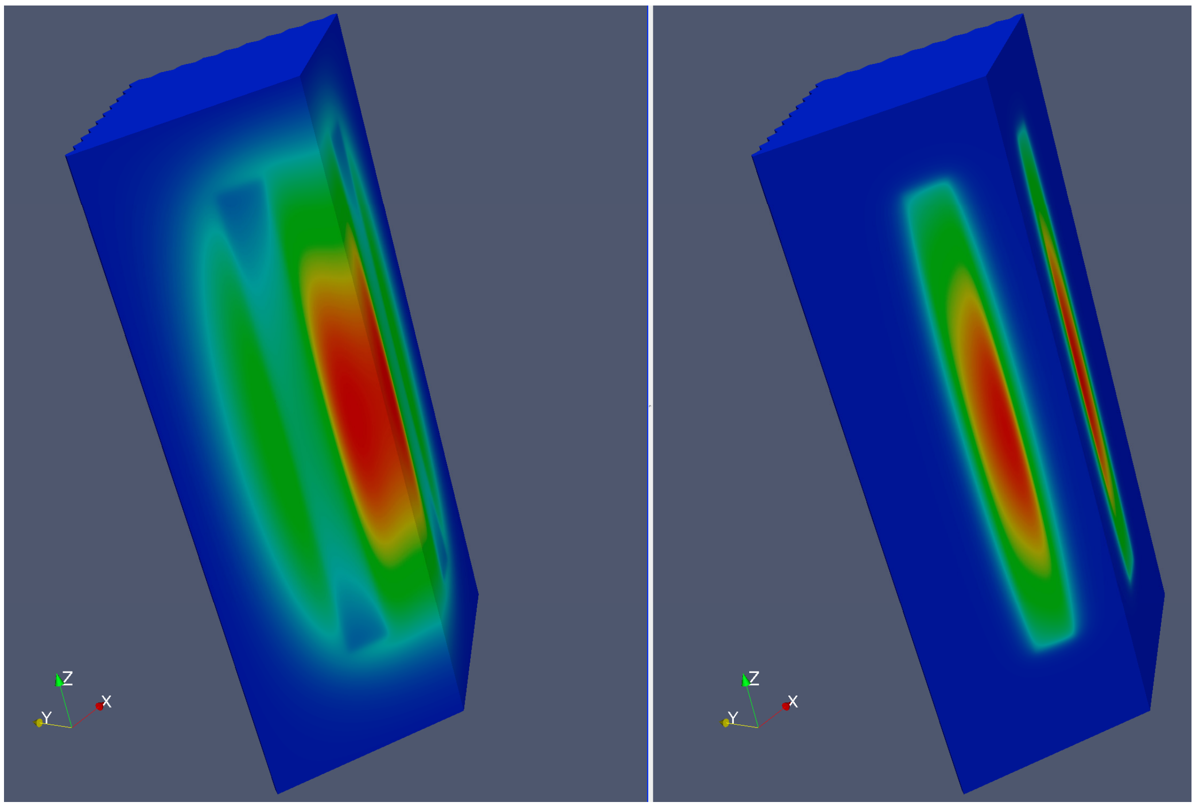


Figure 4. One third core thermal (right) and fast (left) flux solutions.

7. CONCLUSIONS

Although this work is very preliminary, we have demonstrated that interoperability between MOOSE and SHARP is possible. The PRONGHORN application is able to solve a complex problem on a mesh generated from MeshKit, requiring very little additional work. Some minimal programming was required to map block IDs in the mesh to material IDs used by PRONGHORN to assign cross sections. Further collaboration between INL and ANL should be pursued to take advantage of the capabilities already included in SHARP.

8. FUTURE WORK

Because SHARP can use high-fidelity solvers such as UNIC and Star-CCM+, we recommend using SHARP to generate reference solutions for the simplified coupled neutronics and thermal fluid model in PRONGHORN. SHARP could then be used to produce initial conditions for reactor transient calculations run in PRONGHORN. Additionally, there is the possibility of using the cross-section generation from MC2-3, a module of SHARP, to conduct transient runs in PRONGHORN at different depletion states.

It might be advisable to incorporate MOOSE with the SHARP frameworks. This action would maximize the advantage of both approaches to computational analysis of multiphysics systems. Applications in MOOSE could interact with existing codes in loosely coupled manner using the SHARP framework. Additionally, the scientist would be able to use MOOSE to develop new capabilities or to interact with existing applications in the MOOSE environment.

The current released version of MeshKit is built on the CGM library, which is only compatible with Cubit up to 12.2. We recommend future development of MeshKit to be compatible with the newer Cubit version (currently 13.2). Additionally, we recommend training of INL personnel in use of MeshKit and other SHARP applications.

9. ACKNOWLEDGEMENTS

The calculations in this report were performed with PRONGHORN, a parallel neutronics solver developed by INL using a mesh generated by MeshKit that was developed at ANL under the Nuclear Energy Advanced Modeling and Simulation Program. This research used resources of INL, which was supported by the U.S. Department of Energy, Assistant Secretary for the Office of Nuclear Energy, under DOE Idaho Operations Office Contract DE-AC07-05ID14517.

10. REFERENCES

Balay, S., K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, 2004, “PETSc users manual,” *Tech. Rep.* ANL-95/11, Revision 2.1.5, Argonne National Laboratory.

Gaston, D., C. Newman, G. Hansen, and D. Lebrun-Grandie, 2009, “MOOSE: A parallel computational framework for coupled systems of nonlinear equations,” *Nucl. Eng. Design*, Vol. 239, pp. 1768–1778.

Jain, R. and T. J. Tautges, 2012, “RGG: Reactor Geometry (and Mesh) Generator,” *Proceedings of ICAPP ’12*, Chicago, USA, June 24–28, 2012.

Kirk, B. S., J. W. Peterson, R. H. Stogner, and G. F. Carey, 2006, “libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simu- lations,” *Eng Comput-Germany*, Vol. 22, pp. 237–254.

Ortensi, J. et al., 2012, “Prismatic Coupled Neutronics Thermal Fluids Transient Benchmark of the MHTGR-350 MW Core Design,” released April 1, 2012.

Park, H., D. A. Knoll, D. R. Gaston, and R. C. Martineau, 2010, “Tightly coupled multiphysics algorithms for pebble bed reactors,” *Nuclear Science and Engineering*, Vol. 166, No. 2, pp. 118–133.

Tautges, T. J. and R. Jain, 2010, “Creating Geometry and Mesh Models for Nuclear Reactor Core Geometries Using a Lattice Hierarchy-Based Approach,” *Proceedings of the 19th International Meshing Roundtable*.