Integrated Fusion Neutronics Workflow for MCNP, OpenMC, and Shift[1]

J. W. Bae, K. Borowiec, A. Sircar, V. Badalassi

Oak Ridge National Laboratory
Oak Ridge, TN
baej@ornl.gov

INTRODUCTION

A fusion reaction creates high-energy neutrons: in particular, the deuterium–tritium fusion reaction creates a neutron with 14.06 MeV energy. The neutrons generated from this reaction then go on to interact with materials in the reactor, resulting in a myriad of unique effects that can be measured in detailed radiation transport calculations. Important metrics relevant for fusion reactors are as follows: (1) heat deposition, (2) neutron damage, (3) material activation, and (4) tritium breeding ratio.

Because of the complex geometry of fusion reactors and the wide range of neutron energies and materials present, homogenization and discretization of energy and space can present significant challenges in modeling a fusion system. Monte Carlo (MC) particle transport codes model energy and spatial dimensions in a continuous manner, which is a benefit that outweighs the increase in computational demand compared to the use of discretized methods (i.e., deterministic transport solvers).

Numerous MC particle transport codes are used for fusion neutronics, but this work focuses on three in particular: Monte Carlo N-Particle Transport Code (MCNP) [1], Shift [2], and OpenMC [3]. This paper introduces an automation framework to integrate the three codes by developing modules that streamline input generation and output postprocessing. The automation is to simplify multi-code verification efforts and to provide different MC code options for users with varying access levels (MCNP and Shift are export controlled). This framework is demonstrated on a tokamak design with an immersion blanket, similar to the affordable, robust, compact (ARC) type design [4], and assumes that the model geometries are in a computer-aided design (CAD) format, not in constructive solid geometry (CSG).

BACKGROUND

This work is a part of the Fusion Energy Reactor Models Integrator (FERMI) project, which aims to develop a multiphysics framework for fusion reactors. The primary motivation for this framework is to enable easier comparisons among varying software and methodologies. The ability to automatically generate standardized inputs and outputs for use in multiple software makes this comparison possible and increases reproducibility for verification and validation studies. Historically, conversion between software formats was performed manually, which significantly slows down the process of comparison and introduces room for error.

Another important motivation is to ensure that the full range of fusion neutronics analysis capabilities are covered. All three codes can perform cell-averaged flux spectra and reaction rate calculations, which are used for tritium breeding ratio and structural material activation calculations. However, certain capabilities, such as integrated deterministic variance reduction and unstructured mesh tallies, are not available in some MC codes. Depending on the usage, the aforementioned capabilities are essential. For example, in multiphysics coupling, it would be difficult to create a Cartesian or cylindrical mesh for the heat deposition values to appropriately model the fusion reactor. Therefore, the heat deposition values must be meshed on an unstructured mesh for more accurate representation. Integrated deterministic variance reduction is also important for improving the statistics of tallies in deeply shielded regions of the model, such as the magnets. Variance reduction is also necessary for obtaining a converged multigroup flux in mesh tallies, which is then used in follow-on activation and shutdown dose rate calculations.

METHOD

There are five major problem-specific inputs for a MC simulation model: (1) geometry, (2) material composition, (3) source definition, (4) MC parameters, and (5) tallies. With this in mind, a Python framework was developed that represents each of the components in a generic Python object, which then converts and compiles a full MC code input. This framework is similar to OpenMC's Python Application Programming Interface (API) but is available for all three codes. This framework also allows flexible definitions of inputs. For example, in Shift, material composition is defined by atomic density (units of atom/barn-cm), and in MCNP it is defined by a combination of mass (or atom) density combined with a mass (or atomic) fraction. For this framework, the user can define the material composition in any of the formats, and this composition will then be exported to the format for each MC code. In this case, it is Extensible Markup Language (XML) for OpenMC, and it is plain text (with specific syntax) for both MCNP and Shift. The Python framework performs similar conversion for the source definition, MC parameters, and tallies. The definitions are provided by the user as a Python dictionary with specific keys. The dictionary is then converted to each MC code's format and exported.

Geometry conversion between MC codes is usually a laborious process because of the syntax for CSG geometry

[1] Notice: This manuscript has been authored by UT-Battelle LLC under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan)

definition unique to each MC code—so, beginning with a CAD model as the geometry input file removes this necessity, which is the input format assumed for this framework of analysis. To perform particle transport on a CAD model, three methods are available. First, the CAD model can be converted to a CSG model using software such as McCad [5]. As an example, this methodology is often employed for models of ITER [6] equipment, as this process requires the integration of these equipment models into existing MCNP models of the machine. However, this method is time- and labor-intensive because it often requires extensive cleanup of the CAD model and introduces the potential for errors. The second method is to use a meshing software, such as Cubit [7], to mesh the geometry for subsequent particle transport. Currently, MCNP6 supports the capability of performing transport on these meshed geometries. The third and final method is to use the Direct-Accelerated Geometry Monte Carlo (DAGMC) [8] toolkit to track particles on a tessellated CAD geometry. In this framework, MCNP uses an unstructured mesh geometry file (abaqus file),[2] and OpenMC and Shift use a DAGMC geometry. A detailed flowchart of the framework is shown in Fig. 1.

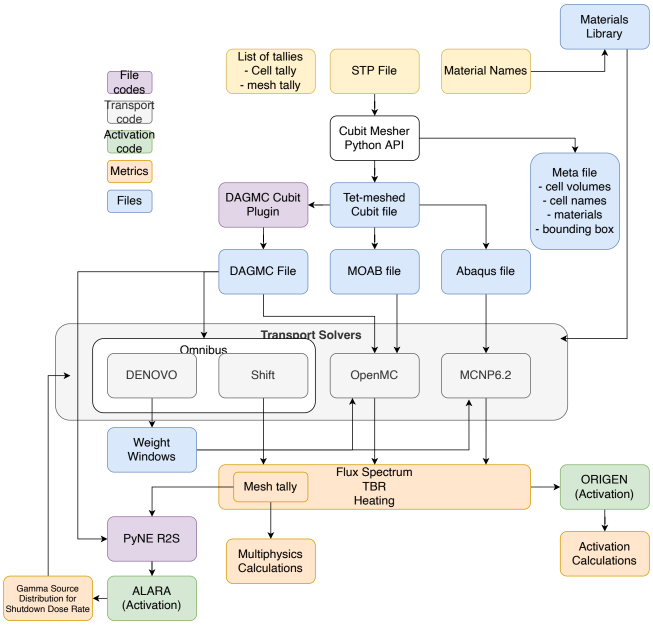

Fig. 1: Flowchart of the integrated MC code framework.

The automated input preparation is split into two steps. In step one, the CAD geometry file is loaded into Cubit to clean (e.g., imprinting and merging, removing overlaps) and mesh the geometry. Cubit then exports the meshed model to an abaqus file. Additionally, the model generates a Comma-Separated Values (CSV) file, in which the user fills out the material names for each volume in the CAD file. The material names correspond to a data library containing material compositions for common fusion reactor materials, fission reactor materials, and the compositions from the Pacific Northwest National Laboratory (PNNL) compendium [9]. During this step, a meta-description JavaScript Object Notation (JSON) file containing data such as the volume of each cell, material for each cell, and the bounding box for the entire geometry is exported.

In step two, the volume–material composition map, user-defined tally dictionary, and MC parameters are combined to create a full input file for MCNP, OpenMC, and Shift. The DAGMC files are generated for OpenMC and Shift with automated zero importance regions (regions that denote the ‘outside’ of the problem geometry), volume generation, and material grouping. Input file components for each MC code are stored in their respective subdirectories.

Following these two steps, the user can then run the input files and use the output parser module to obtain the tally results. The output parser module reads each code’s output file formats—plain text for MCNP, Hierarchical Data Format 5 (HDF5) for OpenMC and Shift—and stores the values in a standardized format and units (e.g., flux in $\frac{particle}{cm^2 \cdot sourceparticle}$, reaction rates in $\frac{reactions}{sourceparticle}$).

RESULTS

This automated workflow is tested on an ARC-class tokamak reactor design, shown in Fig. 2.

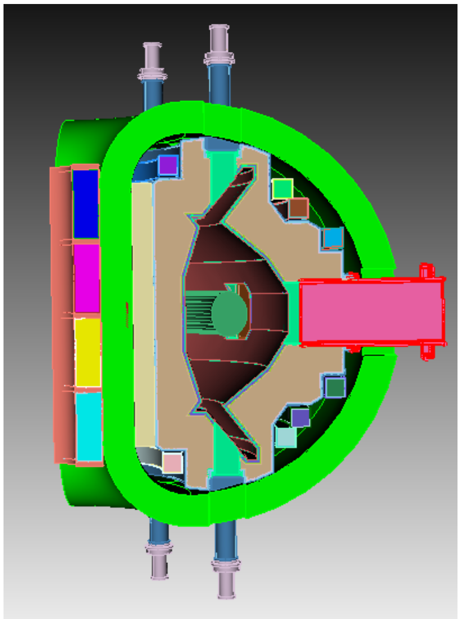

Fig. 2: Geometry of the immersion blanket tokamak 90-degree model, visualized in Cubit.

The user provides the CAD file, a CSV file that associates each volume with a material composition, boundary conditions (e.g., reflective), and a set of tallies. Then the module generates full inputs for MCNP, OpenMC, and Shift. In this model, reflective boundaries are at x=0 and y=0. Two metrics are reported to show the validity of the workflow. First, the 709-group [10] neutron flux spectrum ratios on the coolant channel (Fig. 3) and the blanket (Fig. 4) layer, and second, the tritium breeding ratio in the coolant channel and blanket.

[2] Although MCNP can be used with DAGMC, this requires a custom build of MCNP, which requires source code access to MCNP. On the other hand, the unstructured mesh (UM) capability is available by default in MCNP6.

All three codes were run with 500 million histories and with the ENDF-B/VII.1 [11] data library. No weight windows or source biasing was used. The flux spectra above 1 keV show good agreement between all three codes, within 9%; at lower energies results are not shown because there are not enough particles for a reasonable comparison. The tritium breeding ratio (Table I) for all three codes match within 1.5% near a total of ~1.128. It is possible that the flux and tritium breeding ratio discrepancy is due to data library differences or volume normalization, but further investigation will be required to confirm this hypothesis.

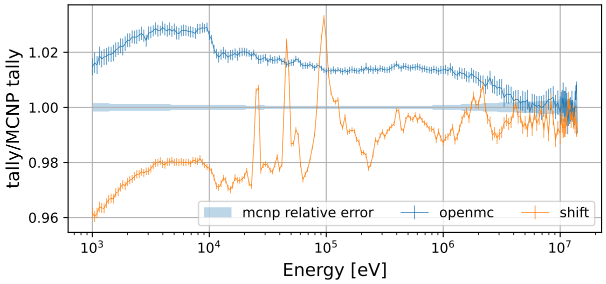

Fig. 3: 709-group neutron flux spectrum tally ratios in the coolant channel. The OpenMC and Shift values are divided by the MCNP results.

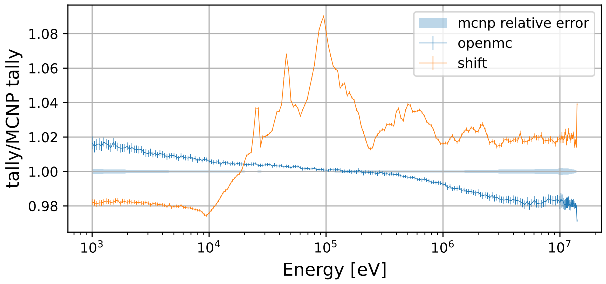

Fig. 4: 709-group neutron flux spectrum tally ratios in the salt blanket. The OpenMC and Shift values are divided by the MCNP results.

TABLE I: Tritium breeding ratio calculated for each code in the coolant channel and the blanket region. The MC mean values are reported with the standard deviation of the tally.

	Tritium breeding ratio [reactions/src. prt.]	
Code	Coolant channel	Blanket
MCNP	0.29104 ± 0.0002	0.83703 ± 0.0001
OpenMC	0.29639 ± 0.0001	0.83989 ± 0.0002
Shift	0.29581 ± 0.0001	0.84817 ± 0.0001
Max rel. diff.	1.6%	1.3%

CONCLUSION

An automated framework that integrates three MC particle transport codes—MCNP, OpenMC, and Shift—has been developed to allow comparison between software and tools as well as to improve reproducibility of MC calculations across platforms. This new development has been demonstrated on a 3D model of an ARC-class tokamak with an immersion blanket, and the results from the three codes match within 1.5%. This framework is expected to improve collaborations among organizations by providing a standardized method of producing MC code inputs and reading their results.

ACKNOWLEDGMENTS

This work was funded under contract ARPA-E GAMOW award DE-AR0001369 with the US Department of Energy.

This work was also performed in collaboration with the Commonwealth Fusion Systems, which provided the ARC-class tokamak geometry.

REFERENCES

1. C. WERNER, “MCNP6.2 Release Notes,” Tech. Rep. LA-UR-18-20808, Los Alamos National Laboratory, Albuquerque, NM (2018).
2. T. M. PANDYA, S. R. JOHNSON, T. M. EVANS, G. G. DAVIDSON, S. P. HAMILTON, and A. T. GODFREY, “Implementation, Capabilities, and Benchmarking of Shift, a Massively Parallel Monte Carlo Radiation Transport Code,” *Journal of Computational Physics*, **308**, 239–272 (Mar. 2016).
3. P. K. ROMANO, N. E. HORELIK, B. R. HERMAN, A. G. NELSON, B. FORGET, and K. SMITH, “OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development,” *Annals of Nuclear Energy*, **82**, 90–97 (Aug. 2015).
4. B. N. SORBOM, J. BALL, T. R. PALMER, F. J. MANGIAROTTI, J. M. SIERCHIO, P. BONOLI, C. KASTEN, D. A. SUTHERLAND, H. S. BARNARD, C. B. HAAKONSEN, J. GOH, C. SUNG, and D. G. WHYTE, “ARC: A Compact, High-Field, Fusion Nuclear Science Facility and Demonstration Power Plant with Demountable Magnets,” *Fusion Engineering and Design*, **100**, 378–405 (Nov. 2015).
5. G. D and T.-T. H, *Current Status of the CAD Interface Program McCad for MC Particle Transport Calculations*, American Nuclear Society, LaGrange Park (United States of America) (2009).
6. J. YANG, S. C. WILSON, S. W. MOSHER, and G. RADULESCU, “Integration of the Full Tokamak Reference Model with the Complex Model for ITER Neutronic Analysis,” *Fusion Science and Technology*, **74**, *4*, 277–287 (2018).
7. T. D. BLACKER, W. J. BOHNHOFF, and T. L. EDWARDS, “CUBIT Mesh Generation Environment. Volume 1: Users Manual,” Tech. Rep. SAND-94-1100, Sandia National Labs., Albuquerque, NM (United States) (May 1994).
8. P. P. H. WILSON, T. J. TAUTGES, J. A. KRAFTCHECK, B. M. SMITH, and D. L. HENDERSON, “Acceleration techniques for the direct use of CAD-based geometry in fusion neutronics analysis,” *Fusion Engineering and Design*, **85**, *10*, 1759–1765 (Dec. 2010).

9. R. S. DETWILER, R. J. MCCONN, T. F. GRIMES, S. A. UPTON, and E. J. ENGEL, “Compendium of Material Composition Data for Radiation Transport Modeling,” Tech. Rep. PNNL-15870-Revision-2, Pacific Northwest National Lab. (PNNL), Richland, WA (United States) (Apr. 2021).
10. “FISPACT-II Wiki,” Tech. rep., UK Atomic Energy Authority (2018).
11. M. B. CHADWICK, M. HERMAN, and ET AL., “ENDF/B-VII.1 Nuclear Data for Science and Technology: Cross Sections, Covariances, Fission Product Yields and Decay Data,” *Nuclear Data Sheets*, **112**, *12*, 2887–2996 (Dec. 2011).