Progress and Status of the Openmc Monte Carlo Code

International Conference on Mathematics and Computational Methods Applied to Nuclear Science & Engineering (M&C 2013)
Sun Valley, Idaho, USA, May 5-9, 2013, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2013)

PROGRESS AND STATUS OF THE OPENMC MONTE CARLO CODE

Paul K. Romano, Bryan R. Herman, Nicholas E. Horelik, Benoit Forget, and Kord Smith
Massachusetts Institute of Technology
Department of Nuclear Science and Engineering
77 Massachusetts Avenue, Cambridge, MA 02139
paul.romano@alum.mit.edu; bherman@mit.edu; nhorelik@mit.edu; bforget@mit.edu; kord@mit.edu

Andrew R. Siegel
Argonne National Laboratory
Theory and Computing Sciences and Nuclear Engineering Division
siegela@mcs.anl.gov

ABSTRACT

The present work describes the latest advances and progress in the development of the OpenMC Monte Carlo code, an open-source code originating from the Massachusetts Institute of Technology. First, an overview of the development workflow of OpenMC is given. Various enhancements to the code such as real-time XML input validation, state points, plotting, OpenMP threading, and coarse mesh finite difference acceleration are described.

Key Words: Monte Carlo, transport, open source, OpenMC

1. INTRODUCTION

With the continual improvement of high-performance computing, many in the reactor physics community have begun to envision a future in which Monte Carlo methods could be used for the simulation of large light-water reactors. While possessing a number of inherent advantages over deterministic methods, there are many challenges to overcome before Monte Carlo methods could realistically be used in commercial reactor analysis [1]. A great deal of research and development efforts are now focused on addressing some of these outstanding issues.

The Computational Reactor Physics Group at MIT has for the last few years focused on analyzing and developing solutions to some of the algorithmic problems that have been an impediment to successful large-scale parallel Monte Carlo simulations [2–4]. In addition, a collaboration with Argonne National Laboratory under the Center for Exascale Simulation of Advanced Reactors has helped to advance the theoretical understanding of domain and data decomposition algorithms [5, 6]. All of these efforts were made possible by the development of a modern Monte Carlo particle transport code, OpenMC [7].

The basic geometry and physics algorithms in OpenMC have been described in detail previously [8]. In the present work, we seek to shed light on some of the more recent developments within the OpenMC community with a particular focus on the software development workflow that has enabled high productivity development and encouraged participation by individuals and organizations outside of the immediate reactor physics community at MIT.

2. DEVELOPMENT

Having grown from a single developer at the beginning of 2011 to numerous developers across multiple institutions, the future success of OpenMC will strongly depend on a productive workflow. Traditionally, the development of software within the nuclear industry has been carried out within single institutions. To the authors' knowledge, there have not been many multi-organization collaborative software development efforts. In many cases, there is good reason for this — maintaining effective management of a project that many organizations are contributing to can be fraught with difficulty. On top of that, the open-source model introduces unique complexities into the software development process; who decides what ultimately goes into the "official" version? How is work delegated among developers scattered across the globe? In his famous essay "The Cathedral and the Bazaar", Eric Raymond noted the challenges of open-source projects, remarking that [9] open-source communities sometimes "resemble a great babbling bazaar of differing agendas and approaches ... out of which a coherent and stable system could seemingly emerge only by a succession of miracles." Despite these challenges, open-source projects can and often do thrive when an enthusiastic development community coalesces. Indeed, even at this early stage, we have received unsolicited contributions and bug-fixes from user/developers outside of MIT.

The decision to release OpenMC as open-source software and encourage participation from outsiders is predicated on the assumption that doing so will mutually benefit both the originating community (MIT) and those who adopt the code for further development. It is our hope and belief that by embracing open collaboration, researchers at different institutions can more effectively work towards a common goal rather than replicate already existing work.

2.1. Collaboration Workflow

Internally, we have adopted an integration-manager workflow as described by Chacon [10]. This workflow works particularly well with the GitHub hosting service that is used by OpenMC. On GitHub, each developer can easily fork a project creating their own public copy of the OpenMC repository. They are then free to make whatever changes and modifications they wish; note that they are not required to have any special access on the original repository. If a developer wants their changes to be merged into the official project, they can issue a *pull request* — at this point, the person designated as the integration manager then reviews the request and, if the changes are acceptable, merges it in.

One of the early concerns raised by a number of developers was the ability to keep some work private even though the project as a whole is open-source. The motivation for privacy could be unpublished research that is not ready to be shared, patentable ideas, commercial ideas, etc. A developer who wishes to have a private copy of OpenMC can do so either by having a local copy on their machine with a remote reference set up to the main OpenMC repository so that they can still pull updates. Another option is to maintain a private copy on GitHub. The general integration-manager workflow is shown in the diagram in Figure 1. The dashed lines represent *pulls* whereas solid lines indicate *pushes*. At MIT, work is carried out on branches (indicated by yellow boxes) of the main repository, public forks, and private copies depending on the nature of the development.

2.2. Real-time XML Validation

OpenMC uses an XML syntax for all user input and configuration files. The use of XML is of great benefit to both developers and users. Developers can make changes to the user input format easily, adding or modifying

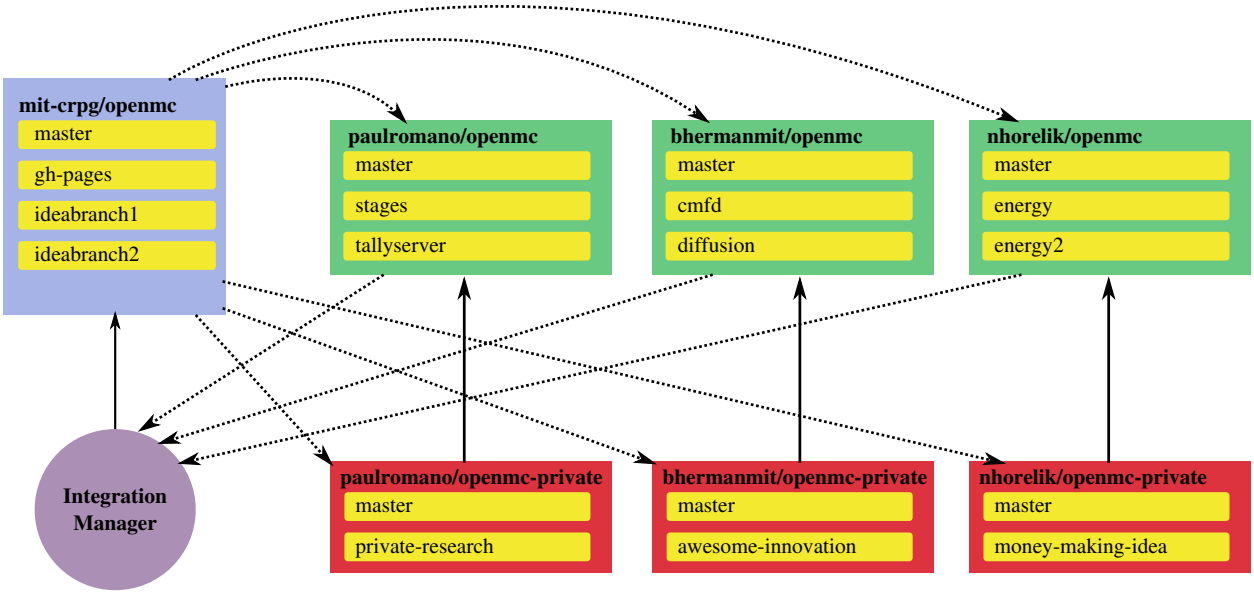

Figure 1. Integration manager workflow pattern used in OpenMC development.

options, and the xml-fortran parser within OpenMC gracefully handles the changes with little effort from the developer. Users are free to form their input files as they like, as long as the overall structure of the files is well-formed and the content conforms to the specification of the file. Until recently, it was only possible to check that user input files were well-formed prior to running, i.e. content was defined, properly delimited with a start and end tag, and properly nested. A set of schemata based on the RELAX NG schema language [11] has now made it possible to check not only that an input file is well-formed but also that it has the correct tags, attributes, and datatypes.

At present, there are two ways that input files can be checked for conformance against the schemata. The first method is post-validation where once an input file has been written, it is checked against a corresponding schema using a tool such as `jing` [12]. From a command-line, the user could enter `jing -c ~/openmc/src/templates/geometry.rnc geometry.xml` and any errors in the input file would be reported back. The first argument is the corresponding schema and the second argument is the XML input file. A more elegant method to check conformance is real-time validation with an editor such as GNU Emacs [13]. When using GNU Emacs to write an input file, the input is continually checked against a corresponding schema (based on the root element in the document). If any errors are found, they are highlighted in red giving the user immediate visual feedback. Figure 2 shows an example of an input file being validated against a schema in GNU Emacs.

If a schema-aware editor such as GNU Emacs is not available (or not desirable) to the user, the post-validation method can be effectively used for checking input files. In a production environment, it is easy to write a script that first calls `jing` to check for conformance against the schemata whenever OpenMC is run.

2.3. Other Enhancements

With the geometry and physics routines within OpenMC reaching a state of relative maturity and stability, more attention has been given to improving input/output options and implementing requested features. The

```
<?xml version="1.0"?>
<geometry>

  <!--
  ================================================
  Description: Simple thermal system from INDC-USA-107
  Case:        Problem 2 (1/4" pin, 2" lattice pitch)
  Written By:  Paul Romano
  Date:        10/30/2011
  ================================================
  -->

  <surface id="1" type="z-cylinder" coeffs="0. 0. 0.635" />
  <surface id="2" type="x-plane" coeffs="-2.54" boundary="reflectiv" />
  <surface id="3." type="x-plane" coeffs="2.54" boundary="reflective" />
  <surface id="4" type="y-plane" coeffs="-2.54" boundary="reflective" />
  <surface id="5" type="y-plane" coeffs="2.54" boundary="reflective" />

  <cell id="1" material="1" surfaces="-1" />
  <cell id="2" material="2" surfaces="1 2 -3 4 -5" >

</geometry>
```

Figure 2. Example of an XML input file in GNU Emacs being validated against a corresponding RELAX NG schema in real-time. Errors in the input are automatically highlighted in red.

following sections describe some of the more important developments made recently in OpenMC.

2.3.1. State Point Capability

The output format in OpenMC for tallies and other results originally consisted of plain text ASCII files. While these files may be convenient if only a few tally quantities are desired, it is generally more difficult to analyze results if thousands or millions of tally quantities are desired. For many problems being looked at with OpenMC, such as the Monte Carlo performance benchmark [14], the ability to view, analyze, and post-process results containing millions of tally quantities is essential. As such, the capability to write results to binary files in the form of *state points* has been introduced in OpenMC.

A state point file contains all the information needed either to determine confidence intervals for tally quantities or to restart the run completely. This information includes the random number seed used, meta-data describing the tallies, and the sum and sum of squares for each tally bin at whatever batch the state point was written. Note in particular that the mean and variance are not stored directly in the state point file — they must be calculated as part of the post-processing process. A restart capability would not be possible if the mean and variance were stored instead of the sum and sum of squares. Since it could be cumbersome for a user to ascertain their desired results from a binary file, a set of Python classes and scripts have been written to ease the burden of post-processing. The development of a graphical user interface is also currently underway that allows a quick means of viewing results for any combination of tally filters and scores*.

The generality of the state point model is particularly beneficial for research purposes. By default, a state point file is only written at the end of a simulation; however, the user can request that a state point file be written at any specified batch or at a fixed batch interval. Since the post-processing is the same whether the state point was written in the middle of a run or at the end, it is easy to obtain and analyze tally results at any

*See [8] for a more complete discussion of filters and scores.

batch. For example, a short script can be written that checks whether tally results satisfy a given criterion. One such criterion would be that 95% of all tally bins have 95% confidence intervals with a half-width of less than 1% of their respective means (*à la* Kord Smith challenge). As a final note, state point files can be written either in a raw binary format or in an HDF5 format. While the HDF5 format should be preferred and ensures portability across different architectures, the former is made available to ensure that users can take advantage of state point capabilities even on systems where HDF5 is unavailable.

2.3.2. Batching and Uniform Fission Source

One commonly encountered problem in Monte Carlo criticality calculations is systematic underprediction of confidence intervals. As discussed in [15], there are a few requisite conditions that can result in this phenomenon: use of the method of successive generations, use of a single fission generation for computing "realizations" of the random variable, and use of statistical formulas for uncorrelated variables. A number of solutions have been proposed for overcoming underprediction of confidence intervals including MacMillan's correction [16], Wielandt's method [17] and batching [15]. We have chosen to adopt the batching approach for its simplicity and relative ease of implementation. The ability to use the accumulated tally results from multiple successive fission generations as a single realization for statistical purposes has now been implemented in OpenMC.

In conjunction with batching capabilities, the uniform fission source (UFS) method proposed and implemented in the MC21 code by Sutton [15] has also been implemented in OpenMC. When solving for reaction rate distributions across an entire core, the UFS method is very effective in flattening the distribution of relative uncertainties.

As a simple demonstration of the effectiveness of the UFS method, we ran two simulations of a model of the OPR reactor [18], a full-core PWR model with enrichment zoning, first without the UFS method and then with the UFS method. In each case, 100 inactive batches and 1000 active batches were used with 1,000,000 particles per batch. No batching was applied in either case as the purpose was to look at the differences due to the UFS method. A tally for the fission reaction rate in each fuel pin was set up. In the UFS case, a $17 \times 17 \times 17$ mesh covering the whole core was used to redistribute source sites. Figures 3 and 4 show the distribution of relative errors of the reaction rate in each pin for the cases without and with the UFS method, respectively. The relative error here is defined to be the half-width of the 95% confidence interval divided by the sample mean. One can see that the UFS method has effectively flattened the distribution of relative errors. In addition, the maximum relative error has also been reduced from 2.9 percent to 2.0 percent. Figure 5 also shows a histogram of the relative error distributions for the two cases.

2.3.3. Plotting capabilities

A plotting capability has been built into OpenMC to produce raster images of slices through geometry. Plots can be produced by writing a *plots.xml* input file which specifies, at the very least, the origin and range of the plot and the basis vectors (currently limited to standard basis vectors). Beyond the required plotting information, numerous options are available that allow the plot to be colored either by material or cell, to assign specific colors to certain materials/cells, and to "mask" certain materials/cells. With the plotting specification, OpenMC calls a `find_cell` routine that determines what cell corresponds to each pixel and writes out a simple PPM format bitmap image[†] The PPM images can be natively viewed on many Linux

[†]The `find_cell` routine is also used during an actual simulation and therefore the image is representative of the geometry used during simulation.

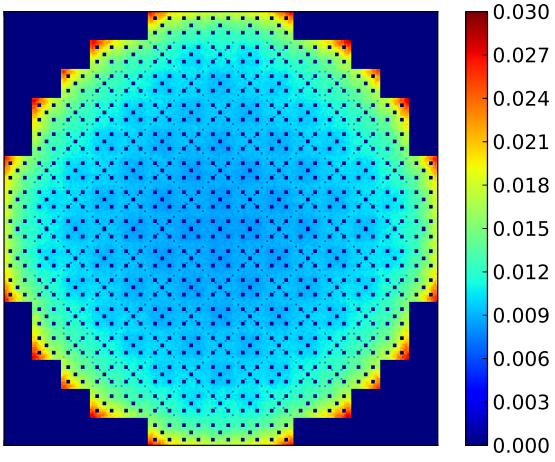

Figure 3. Relative error of the fission reaction rate in each pin of OPR without the UFS method.

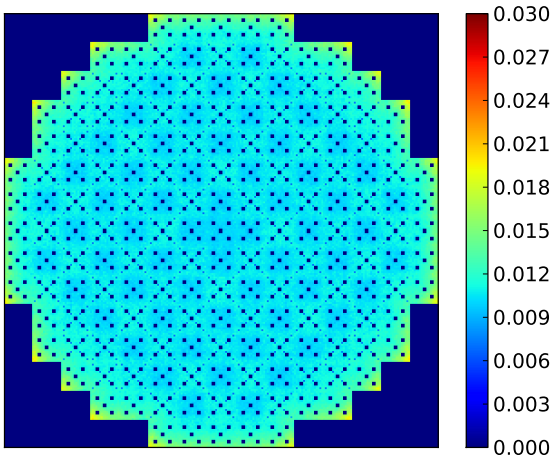

Figure 4. Relative error of the fission reaction rate in each pin of OPR with the UFS method.

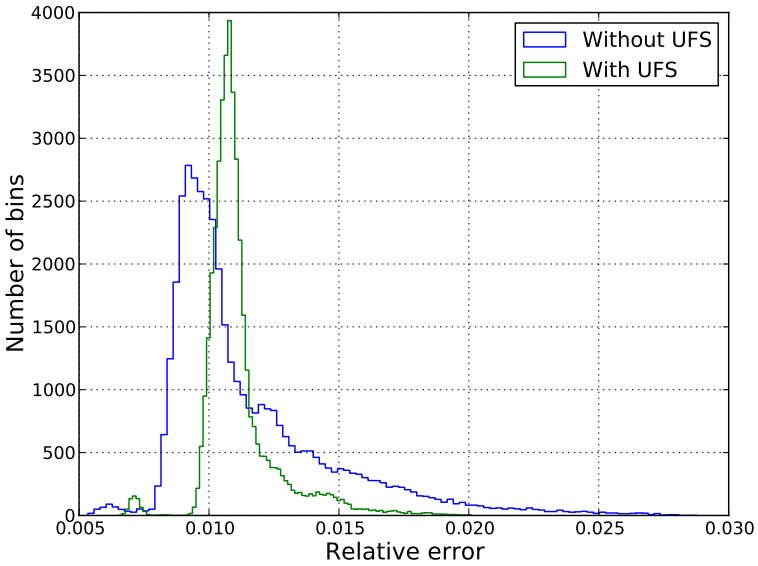

Figure 5. Relative error distribution for fission reaction rate tallies with and without the UFS method.

distributions or can also be converted to a compressed format such as PNG with an image converting utility. As an example of the current plotting capabilities, Figure 6 shows a plot of the OPR reactor model. In the future, more advanced plotting capabilities such as 3D point cloud plots based on ray-tracing may be added to OpenMC. Near-term development of a graphical user interface to enable interactive plotting is also underway.

2.3.4. Fixed Source Mode

One of the most commonly requested features for OpenMC by the nascent user community was the ability to perform fixed source calculations. As such, it is now possible to perform fixed source calculations within OpenMC. Efforts are also underway to add the ability to filter tallies by whether particles are uncollided,

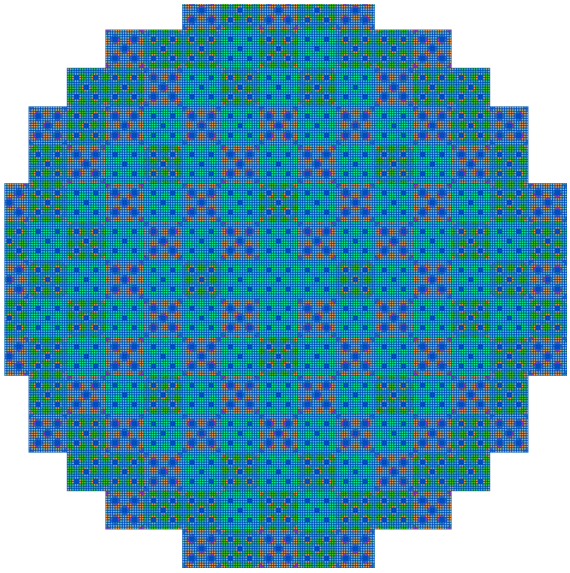

Figure 6. Plot of the OPR reactor model produced from OpenMC's native plotting capability.

have undergone one collision, have undergone two collisions, etc. Combined with the fixed source capability, this would provide an easy means of studying scattering distributions in particular nuclei and could be useful for validation and verification. As another example of the usefulness of fixed source calculations, we refer the reader to a paper by Herman et al. [19] wherein a fixed source calculation in MC21 is used to generate highly-accurate diffusion coefficients for use in a coarse mesh finite difference solver. The same methodology has been implemented in OpenMC.

2.3.5. Threading with OpenMP

The lion's share of research and development using OpenMC has focused on the development of parallel algorithms for use on leadership-class supercomputers (see e.g. [3, 4, 8]). While these efforts have greatly improved parallel scalability and overcome several algorithmic challenges in large-scale eigenvalue calculations, they have been focused primarily on distributed-memory implementations based on MPI. However, given the continued growth in concurrency via increasing numbers of cores per processor, it would behoove the OpenMC developers to implement some form of shared-memory parallelism. As such, a recent collaboration between MIT and Argonne National Laboratory under the Center for Exascale Simulation of Advanced Reactors (CESAR) has resulted in the ability to run multi-threaded simulations in OpenMC using the OpenMP API.

By using a directives-based API and a "coarse-grained" threading implementation where batches of particles are split up evenly across threads, only modest modifications to the code were necessary. The key aspects of the threading implementation are: 1) threading the main particle loop using the *omp parallel for* construct; 2) marking key global mutable data structures as *threadprivate*; and 3) marking all tally increments as *atomic* operations during the tracking of a particle. Significant speedups have been obtained on platforms ranging from dual-core laptops to clusters with nodes having four 12-core AMD Opteron processors. The CESAR studies have also helped to identify root causes for sub-optimal scaling on multi-core platforms [20].

2.3.6. User input improvements

A variety of improvements in user input have been adopted in OpenMC mostly as a result of feedback from the user community at MIT. We mention here a few related to material definitions. In many benchmark problems, such as those from the International Handbook of Evaluated Criticality Safety Benchmark Experiments [21], materials are defined by the concentration in atom/barn-cm of a combination of elements and isotopes. OpenMC has always had the capability to specify the units for the density of a material. Two new options further simplify material definitions. The first is the ability to use `units="sum"` indicating that the total density of the material is simply the sum of its constituents. The other option is to specify the concentration of an element, e.g. `<element name="Zr" ao="3.4056e-02" />`, and OpenMC automatically converts the element into its naturally-occurring isotopes by their relative abundances.

2.3.7. Rotation/translation

A number of ongoing projects at MIT may entail using OpenMC for analysis of the Advanced Test Reactor (ATR) at Idaho National Laboratory as well as the MIT Research Reactor (MITR). Both ATR and MITR have complex geometries that are not amenable to using typical rectangular lattices. For both of these models, having the ability to rotate and translate different components in the geometry is crucial. Rotation and translation of universes is now possible within OpenMC. As an example, Figure 7 shows a model of the ATR that uses rotation and translation.

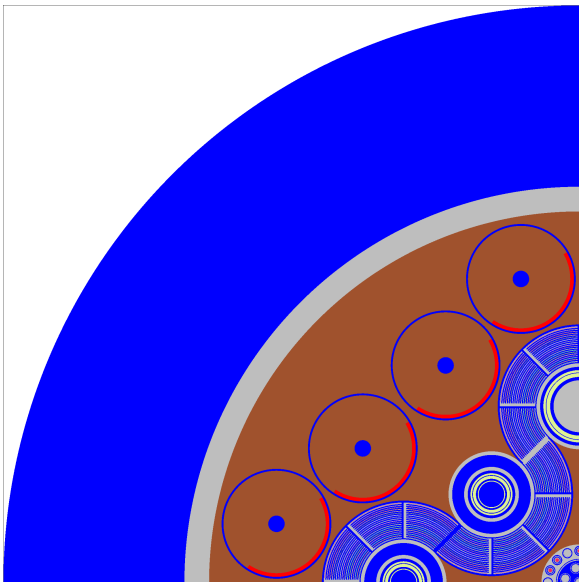

Figure 7. Geometry plot of a quarter-core OpenMC model of the Advanced Test Reactor.

2.3.8. Tally enhancements

One area of the OpenMC codebase that has undergone many changes, and may continue to evolve, is the tally system. Until recently, one could only obtain reaction rates for an entire material. However, many, if not all, application areas often require reaction rates on a per-nuclide basis. In addition to basic scoring functions (flux, fission rate, neutron production rate, etc.), users can now also specify a list of nuclides for which the scoring functions should be tallies. Work is now underway to add depletion capability in OpenMC which will rely on the ability to obtain nuclide reaction rates.

2.4. CMFD

The Coarse Mesh Finite Difference (CMFD) method has been widely applied in deterministic nodal diffusion calculations to reduce the number of fission source iterations. Recently, CMFD has been applied to reactor calculations using multigroup Monte Carlo [18] to accelerate convergence of the fission source. The CMFD acceleration method has been integrated into OpenMC. CMFD acceleration works by solving the multigroup neutron diffusion equation after each Monte Carlo batch to obtain a better estimate of the global fission source. This process can be described in three steps:

1. Compute diffusion parameters by preserving OpenMC neutron balance on a coarse mesh.
2. Solve the multigroup neutron diffusion equation to obtain fission source distribution.
3. Force Monte Carlo to use fission source distribution from Step 2 by modifying source weights.

It must be stressed that lack of proper neutron balance can lead to sub-optimal convergence. The acceleration process will only be effective if the balance of neutrons in each energy group and coarse mesh cell is mapped consistently from transport theory to low-order diffusion theory. This in turn requires a tally system, such as that in OpenMC, that can calculate integrated reaction rates and net currents on every coarse mesh surface. Homogenized macroscopic cross sections, needed in diffusion theory, are calculated to preserve reaction rates by dividing integrated reaction rate tallies by integrated flux tallies from OpenMC,

$$\overline{\overline{\Sigma}}^{g}_{\alpha_{l,m,n}} = \frac{\langle \Sigma_\alpha \phi \rangle^{g}_{l,m,n}}{\langle \phi \rangle^{g}_{l,m,n}}, \tag{1}$$

where $\overline{\overline{\Sigma}}^{g}_{\alpha_{l,m,n}}$ is a macroscopic cross section in coarse mesh cell (l, m, n) and energy group g calculated by preserving the arbitrary reaction rate α from OpenMC tallies. The numerator, $\langle \Sigma_\alpha \phi \rangle^{g}_{l,m,n}$, represents an integrated reaction rate tally, while the denominator, $\langle \phi \rangle^{g}_{l,m,n}$, is an integrated flux tally. To retain total neutron balance, the leakage rate out of a mesh cell must also be preserved. This is accomplished by adding a free equivalence parameter in the discretized Fick's Law that relates net current to scalar flux [22],

$$\overline{J}^{u,g}_{l+1/2,m,n} = -\tilde{D}^{u,g}_{l+1/2,m,n}\left(\overline{\overline{\phi}}^{g}_{l+1,m,n} - \overline{\overline{\phi}}^{g}_{l+1,m,n}\right) \boxed{+\hat{D}^{u,g}_{l+1/2,m,n}\left(\overline{\overline{\phi}}^{g}_{l+1,m,n} + \overline{\overline{\phi}}^{g}_{l+1,m,n}\right)}. \tag{2}$$

The unboxed portion of Eq. (2) represents conventional discretized Fick's Law. The surface area-averaged net current, $\overline{J}^{u,g}_{l+1/2,m,n}$, can be obtained from OpenMC's surface current tallies. Diffusion coefficients from neighboring cells, obtained from flux weighting the macroscopic transport cross section, are combined along with the cell volume into the coupling parameter, $\tilde{D}^{u,g}_{l+1/2,m,n}$. The cell-averaged flux, represented as $\overline{\overline{\phi}}^{g}_{l+1,m,n}$ can also be obtained from OpenMC tallies. Therefore, the only unknown is the equivalence parameter, $\hat{D}^{u,g}_{l+1/2,m,n}$.

In step 2, the multigroup diffusion equation is solved with this equivalence parameter and macroscopic cross sections using standard eigenvalue problem solution techniques. Eq. (2) is used for the current representation in the discretized diffusion equations. The final step involves modifying source weights before the next OpenMC batch to force Monte Carlo to use the diffusion estimate of the fission source distribution. These weights are modified by the ratio of the expected number of neutrons present in a coarse mesh cell from the

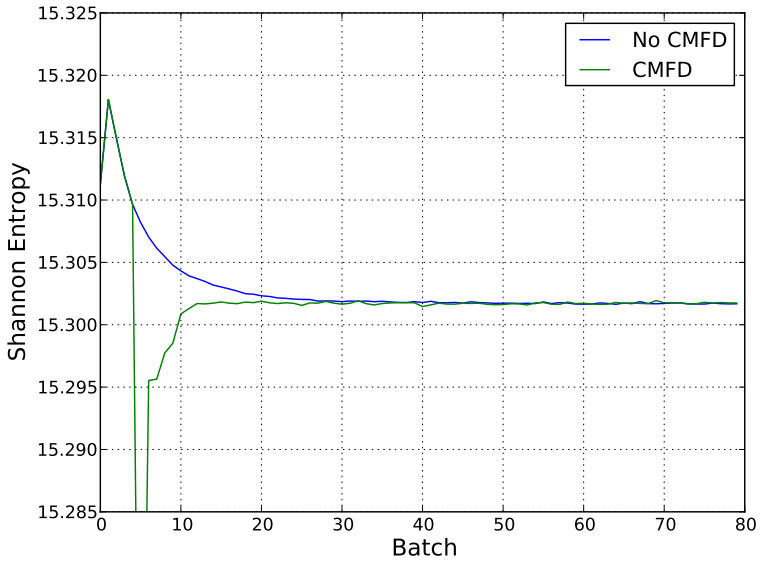

Figure 8. Comparison of source convergence with and without CMFD acceleration.

diffusion result to the actual number of source sites that were generated from the previous OpenMC batch. The following equation shows this modification:

$$w'_j = w_j \times \frac{N_h p^g_{l,m,n}}{\sum_i w_i} \tag{3}$$

where w_j and w'_j are the un-modified and modified weights of the jth neutron in coarse mesh cell (l, m, n) and energy group g, respectively. The expected number of neutrons from diffusion is calculated by multiplying the number of histories per batch, N_h with the normalized fission source distribution $p^g_{l,m,n}$. This is then compared to the total weight of source sites in the corresponding cell and energy group.

Source convergence was studied for the 2D OPR reactor with and without CMFD turned on during inactive batches. Figure 8 shows the effect of CMFD on source convergence as measured by the Shannon entropy of the fission source distribution. For a standard OpenMC run of the OPR core, the Shannon entropy convergences at approximately batch 40, at which point actives batches could begin to start accumulating statistical results. On the other hand, with CMFD active, the source distribution converges in only 15 batches. In this run, CMFD was turned on at batch 5 resulting in a sudden decrease in the CMFD Shannon entropy as shown in Figure 8. For this 2D OPR model, the use of CMFD saves about 25 inactive batches. It should be noted that the computational time to perform the linear solve of the CMFD system is negligible compared to the time to transport neutrons using Monte Carlo. The benefit of using CMFD will increase as the dominance ratio of the system increases, e.g. in 3D full-core reactor models.

3. CONCLUSIONS

In a relatively short period of time, OpenMC has become an important tool for reactor physics research at MIT. Ultimately, we view OpenMC as primarily being a tool for advanced R&D on Monte Carlo methods — if it happens to be useful for production analysis, then so be it, but that is not the explicit intent or goal of the project. By releasing OpenMC to the wider community under an open source license, it is the authors' hope

that others will be able to leverage the work put into OpenMC thus far. OpenMC is available under the permissive MIT open source license at `https://github.com/mit-crpg/openmc`.

The present work summarizes some of the latest developments within OpenMC. The integration-manager workflow presented allows anyone to contribute to OpenMC by forking the main repository and issuing pull requests. A number of developments were discussed which will help users be more productive, such as real-time XML validation, state points, material definition options, plotting capabilities, and CMFD acceleration. Lastly, improvements in geometry and tally capabilities were mentioned.

ACKNOWLEDGMENTS

This research was performed under appointment of the first and second authors to the Rickover Fellowship Program in Nuclear Engineering sponsored by Naval Reactor Division of the U.S. Department of Energy. This work was also supported in part by the Office of Advanced Scientific Computing Research, Office of Science, US Department of Energy, under Contract DE-AC02-06CH11357 and by the Consortium for Advanced Simulation of Light Water Reactors, an Energy Innovation Hub for Modeling and Simulation of Nuclear Reactors under US Department of Energy Contract No. DE-AC05-00OR22725. The first author would like to thank Anthony Scopatz for helpful discussion on the topic of workflow patterns.

REFERENCES

[1] W. R. MARTIN, "Challenges and Prospects for Whole-Core Monte Carlo Analysis," *Nucl. Eng. Technol.*, **44**, 151 (2012).

[2] P. ROMANO, B. FORGET, and F. BROWN, "Towards Scalable Parallelism in Monte Carlo Transport Codes Using Remote Memory Access," *Prog. Nucl. Sci. Technol.*, **2**, 670 (2011).

[3] P. K. ROMANO and B. FORGET, "Parallel Fission Bank Algorithms in Monte Carlo Criticality Calculations," *Nucl. Sci. Eng.*, **170**, 125 (2012).

[4] P. K. ROMANO and B. FORGET, "Reducing Parallel Communication in Monte Carlo Simulations via Batch Statistics," *Trans. Am. Nucl. Soc.*, **107** (2012), Accepted.

[5] A. SIEGEL, K. SMITH, P. FISCHER, and V. MAHADEVAN, "Analysis of communication costs for domain decomposed Monte Carlo methods in nuclear reactor analysis," *J. Comput. Phys.*, **231**, 3119 (2012).

[6] A. R. SIEGEL, K. SMITH, P. K. ROMANO, B. FORGET, and K. FELKER, "The effect of load imbalances on the performance of Monte Carlo codes in LWR analysis," *J. Comput. Phys.* (2012).

[7] MIT Computational Reactor Physics Group, "OpenMC Monte Carlo Particle Transport Code," `https://github.com/mit-crpg/openmc`, 2012.

[8] P. K. ROMANO and B. FORGET, "The OpenMC Monte Carlo Particle Transport Code," *Ann. Nucl. Energy*, **51**, 274 (2013).

[9] E. S. RAYMOND, *The Cathedral & the Bazaar*, O'Reilly Media, Cambridge, Massachusetts (1999).

[10] S. CHACON, *Pro Git*, Apress, Berkeley, California (2009).

[11] ISO/IEC JTC1/SC34, "Information technology – Document Schema Definition Language (DSDL) – Part 2: Regular-grammar-based validation – RELAX NG," ISO/IEC 19757-2:2008, International Organization for Standardization (2008).

[12] Thai Open Source Software Center Ltd., "Jing - A RELAX NG validator in Java," `http://www.thaiopensource.com/relaxng/jing.html`, 2012.

[13] Free Software Foundation, Inc., "GNU Emacs - GNU Project," `http://www.gnu.org/software/emacs/`, 2012.

[14] J. E. HOOGENBOOM, W. R. MARTIN, and B. PETROVIC, "The Monte Carlo Performance Benchmark Test - Aims, Specifications and First Results," *Proc. International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Rio de Janeiro, Brazil, 2011.

[15] D. J. KELLY, T. M. SUTTON, and S. C. WILSON, "MC21 Analysis of the Nuclear Energy Agency Monte Carlo Performance Benchmark Problem," *Proc. PHYSOR – Advances in Reactor Physics – Linking Research, Industry, and Education*, Knoxville, Tennessee, 2012.

[16] D. B. MACMILLAN, "Monte Carlo Confidence Limits for Iterated-Source Calculations," *Nucl. Sci. Eng.*, **50**, 73 (1973).

[17] B. C. KIEDROWSKI and F. B. BROWN, "Using Wielandt's Method to Eliminate Confidence Interval Underprediction Bias in MCNP5 Criticality Calculations," *Trans. Am. Nucl. Soc.*, **99**, 338 (2008).

[18] M. J. LEE, H. G. JOO, D. LEE, and K. SMITH, "Monte Carlo Reactor Calculation with Substantially Reduced Number of Cycles," *Proc. PHYSOR – Advances in Reactor Physics – Linking Research, Industry, and Education*, Knoxville, Tennessee, 2012.

[19] B. R. HERMAN, B. FORGET, K. S. SMITH, and B. N. AVILES, "Improved Diffusion Coefficients Generated from Monte Carlo Codes," *Proc. International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Sun Valley, Idaho, 2013.

[20] A. R. SIEGEL, K. SMITH, P. K. ROMANO, B. FORGET, and K. FELKER, "Multi-core performance studies of neutral particle Monte Carlo methods," *Int. J. High Perform. Comput. Appl.* (2012), Submitted.

[21] NEA Nuclear Science Committee, "International Handbook of Evaluated Criticality Safety Benchmark Experiments," NEA/NSC/DOC(95)03, OECD Nuclear Energy Agency (2009).

[22] K. S. SMITH and J. D. RHODES, "FULL-CORE, 2-D, LWR CORE CALCULATIONS WITH CASMO-4E," *Proc. PHYSOR 2002*, Seoul, Korea, 2002.