



Monte Carlo multiphysics simulation on adaptive unstructured mesh geometry

A.J. Novak^{a,*}, H. Brooks^b, P. Shriwise^c, A. Davis^b

^a University of Illinois, Urbana-Champaign, United States of America

^b United Kingdom Atomic Energy Authority, United Kingdom

^c Argonne National Laboratory, United States of America

ARTICLE INFO

Keywords:

Cardinal
AURORA
MOOSE
OpenMC
NekRS
Monte Carlo
CFD
CAD

ABSTRACT

Monte Carlo simulation based on Constructive Solid Geometry (CSG) brings unique challenges for multiphysics simulation, including establishing field transfers with mesh-based physics codes, the combination of stochastic and deterministic solvers, and high computational expense. In this work, an adaptive, on-the-fly mesh-based Monte Carlo geometry algorithm is implemented in Cardinal to reduce the barrier-to-entry for high-fidelity multiphysics by (i) eliminating ambiguity in defining CSG cells for temperature and density feedback, (ii) enabling simple mesh convergence studies, and (iii) more closely integrating Computer Aided Design (CAD) workflows with Monte Carlo methods. During Picard iterations, an OpenMC mesh geometry is adaptively refined or coarsened by contouring temperature and/or density fields from a thermal-fluid solver. This algorithm is applied to a full-core Molten Salt Fast Reactor (MSFR) geometry with NekRS Large Eddy Simulation (LES) coupled to OpenMC neutron transport. A performance study indicates a net speedup of 2.3× in the OpenMC solver when using an adaptive geometry for cell sizes chosen intermediate to the as-built CAD geometry versus 1:1 element tracking, which points to future algorithmic research in accelerated Monte Carlo mesh tracking.

1. Introduction

Recent growth in computing now permits large-scale multiphysics modeling using Monte Carlo methods, considered state-of-the-art for particle transport. The enormous diversity in the advanced reactor landscape requires a flexible and general-purpose computational framework, to which the Monte Carlo method is well-suited for benchmarking lower-fidelity tools, guiding multigroup cross section library development, and aspects of reactor design (a growing use case in the microreactor sector). However, many challenges arise when coupling a Monte Carlo transport solver to a typical mesh-based Partial Differential Equation (PDE) solver, due to the different geometry representations, the combination of stochastic and matrix-type solvers, and the high computational cost.

This paper focuses on methods for multiphysics feedback to unstructured mesh Monte Carlo simulations which are adaptive and flexible, with the goal of reducing the barrier-to-entry for high-fidelity multiphysics simulation. This paper will involve comparisons of Constructive Solid Geometry (CSG) and mesh-based geometries, and as such Section 1.1 first provides necessary background on the different geometry representations used in Monte Carlo codes. Section 1.2 then summarizes the challenges for multiphysics data transfers with Monte Carlo

solvers to provide motivation for the present work. An outline of the subsequent content is provided at the end of Section 1.2.

1.1. Geometry representations in Monte Carlo

Computational geometry in Monte Carlo radiation transport can take a number of forms, depending on the software in question. The most common geometry representations include (i) CSG, (ii) surface meshes, and (iii) volumetric meshes. Fig. 1 provides a conceptual depiction of these different geometries. Most codes support intermixing of these various geometry representations together into a single model using the notion of universes (Leppanen, 2022; Bennett et al., 2019; Ferraro et al., 2018) or layering (Biondo et al., 2022).

The most common geometry format is CSG, also referred to as “combinatorial geometry”. Regions of space are formed using boolean combinations of the “half spaces” of various first- and second-order surfaces (e.g. planes, cylinders, spheres), as well as fourth-order tori. For example, the CSG pincell shown in Fig. 1 can be constructed using the half spaces of six planes and a cylinder. CSG is well-suited to many problems due to its ability to exactly represent common fission reactor

* Corresponding author.

E-mail address: ajnovak2@illinois.edu (A.J. Novak).

<https://doi.org/10.1016/j.nucengdes.2024.113589>

Received 18 April 2024; Received in revised form 31 July 2024; Accepted 9 September 2024

Available online 26 September 2024

0029-5493/© 2024 Elsevier B.V. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

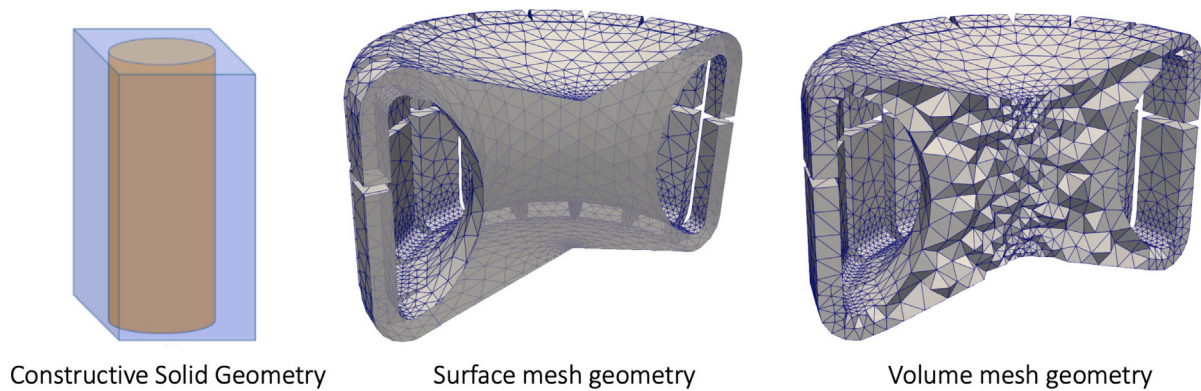


Fig. 1. Common geometry representations in Monte Carlo codes. The CSG example shows a pin cell, while the mesh examples show a fast-spectrum Molten Salt Reactor (MSR).

geometries and repeated geometry units in lattice structures. CSG is also appealing for its fast and robust ray tracing capabilities.

However, a far more common representation for many engineering workflows is Computer-Aided Design (CAD) geometry, usually the starting point for unstructured mesh generation. While several software libraries provide conversion capabilities from CAD into CSG (e.g., Große et al., 2013; Song et al., 2014; Wang et al., 2020), the process is imperfect and sometimes requires additional manual intervention (Häußler et al., 2016). Therefore, other forms of Monte Carlo geometry are instead based on unstructured meshes, of either (i) volumetric or (ii) surface type. These mesh representations are often colloquially referred to as “CAD transport”, though the geometry is represented using unstructured meshes instead of the splines in CAD software. Many CAD geometry engines do support ray tracing on boundary representations, but at high computational cost due to the root-finding needed for these high order surfaces (Tautges et al., 2009). Therefore, direct tracking on CAD has not yet seen widespread use for radiation transport.

Mesh-based geometries are appealing due to their ability to represent arbitrary geometry. Examples of geometries that cannot readily be built using CSG include the curved vessel walls of fast spectrum MSRs (Rouch et al., 2014), additively-manufactured fuels (Trammell et al., 2019), and many components in fusion devices (Häußler et al., 2016). Most multiphysics and engineering workflows are also already based on CAD software, which may offer synergies with mesh-based Monte Carlo methods.

Volumetric mesh geometry tracks particles within individual mesh elements. Each cell typically corresponds to a single mesh element and cell adjacency information is readily available from the mesh topology. Volumetric mesh geometry is supported in several Monte Carlo codes. Serpent’s volume meshes are based on the OpenFOAM polyhedral mesh format (Ferraro et al., 2018). These polyhedra are internally subdivided into tetrahedral volume meshes to simplify tracking and eliminate some numerical precision issues related to non-coplanar faces. MCNP6’s volume mesh geometry is based on the Abaqus format, and supports first and second-order tetrahedral, pentahedral, and hexahedral meshes (Martz, 2017). Using the CADMesh interface, GEANT4 supports tessellated surface meshes in addition to in-line conversion of surface meshes to Delaunay tetrahedralized volumes for more efficient tracking (Poole et al., 2012). A volumetric unstructured mesh mode is currently under development in OpenMC based on libMesh (Kirk et al., 2006; Shriwise et al., 2022a) with support for tetrahedral meshes. The JMCT code also supports volume unstructured mesh transport in tandem with domain decomposition (Wang et al., 2023).

Contrary to volume mesh tracking, surface mesh geometry tracks particles within a volume bounded by a triangulated surface. For radiation transport, such representations are appealing because, in many cases, the material cross sections, Σ , are uniform across multiple volume mesh elements and ray tracing through a volume mesh incurs

unnecessary re-sampling of flight paths (when using surface tracking) upon entering each new element. For the same level of detail, surface meshes capture arbitrary geometry with fewer elements than volumetric meshes, also resulting in lower memory requirements (Turner, 2017). In contrast, volume meshes can leverage the element adjacency and fast distance-to-surface computation in tetrahedra to sometimes yield performance gains when compared to large surface meshes (Poole et al., 2012).

Surface mesh geometry is supported in a large number of Monte Carlo codes. Serpent’s surface mesh geometry is based on native support for the Stereolithography (STL) triangular surface mesh format (Leppanen, 2022). Other Monte Carlo codes leverage the Direct Accelerated Geometry Monte Carlo (DAGMC) library (Wilson et al., 2010), which provides implementations of the necessary ray tracing kernels on MOAB (Tautges et al., 2004) triangular surface meshes which can then be called within the tracking routines of an existing Monte Carlo code. DAGMC has been integrated into MCNP (Wilson et al., 2010), OpenMC (Shriwise et al., 2020), Shift, and many other Monte Carlo codes.

Despite some of the advantages of mesh-based geometries, there are a number of key limitations. First and foremost, mesh-based geometries require greater attention to model preparation. Many CAD packages do not generate mesh geometry to the high standards needed for particle transport. For particle transport, each position in space should belong to a single cell, but the presence of small gaps and body overlaps in meshes generated from CAD software can result in lost particles or other numerical issues. Fixing these deficiencies can be a highly manual process.

Most unstructured mesh geometries used for Monte Carlo transport use first-order elements, which are incapable of perfectly capturing curved surfaces. This discretization error may distort material volumes (and mass), which for criticality applications may require careful attention to mesh generation and/or density corrections to preserve mass (Spencer and Alwin, 2019). Unstructured mesh geometries also necessitate higher memory usage to store the geometry representation and acceleration data structures than corresponding CSG representations (Leppanen, 2022; Yilmaz et al., 2024) (if they can be built for the geometry in question). In nearly all cases CSG matches or outperforms mesh representations in terms of runtime (Turner, 2017; Wilson et al., 2010; Leppanen, 2022; Häußler et al., 2016; Shriwise et al., 2022b; Yilmaz et al., 2024), though exceptions to this behavior have been observed due to the efficacy of acceleration structures, the problem physics (e.g. mean free path relative to cell size), and possibly code-specific implementation details.

1.2. Multiphysics feedback to Monte Carlo transport

Multiphysics feedback to a Monte Carlo code is typically in the form of geometry (e.g., thermal expansion) and material temperatures,

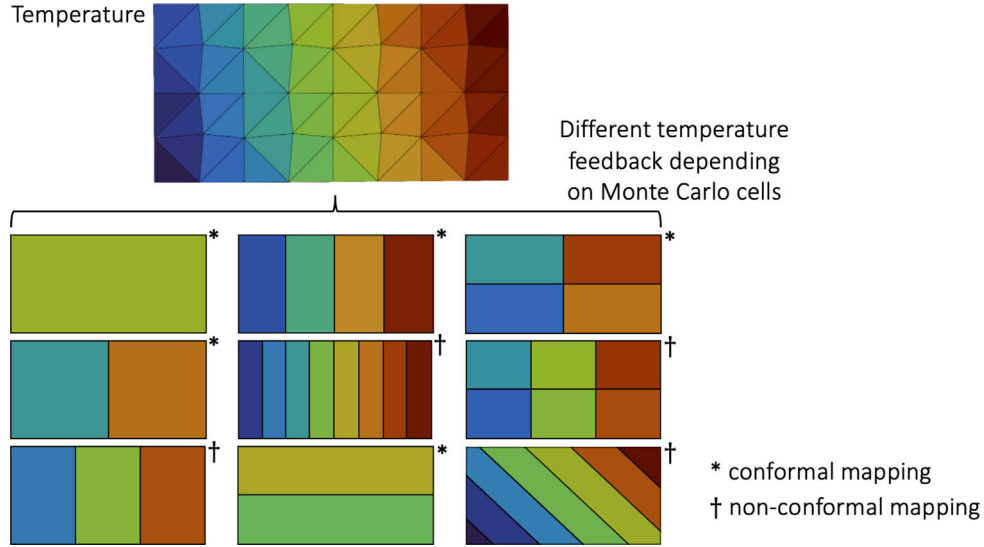


Fig. 2. Different choices for Monte Carlo CSG cells determines the “granularity” of multiphysics feedback. Examples of conformal and non-conformal mappings are denoted with asterisks or daggers.

densities, and nuclide concentrations. These data transfers are typically performed from a coupled mesh-based PDE solver, such as a Computational Fluid Dynamics (CFD) code, to the Monte Carlo application.

Understanding these data transfers requires knowledge of how Monte Carlo methods move particles through space. Rather than spatially and temporally discretizing PDEs with Degrees of Freedom (DOFs) at nodes, Monte Carlo methods explicitly simulate individual stochastic particle tracks through “cells”. For a total cross section $\Sigma_t(x)$, the distance-to-collision d is sampled as

$$\xi = \int_0^d \Sigma_t(x) \exp \left[- \int_{x_0}^x \Sigma_t(x') dx' \right] dx, \quad (1)$$

where $0 \leq \xi \leq 1$ is a uniformly-distributed random number. For arbitrary $\Sigma_t(x)$, Eq. (1) lacks a closed-form solution for d . However, if Σ_t is constant, Eq. (1) becomes

$$d = - \frac{\ln(\xi)}{\Sigma_t} \quad (2)$$

Three different options are generally used to evaluate Eq. (1):

- **Uniform-property cells:** Subdivide the domain into uniform-property cells such that Eq. (2) is valid.
- **Numerical solution:** Numerically solve Eq. (1), such as with numerical integration (“substepping”) or Newton iteration (Brown et al., 2004).
- **Rejection sampling:** Replace $\Sigma_t(x)$ in Eq. (1) by the “majorant” cross section Σ_{maj} , or the maximum Σ_t . Then, the distance-to-collision is sampled with $d = -\ln(\xi)/\Sigma_{maj}$ anywhere in the domain, with rejection sampling used to “accept” an individual collision based on the ratio $\Sigma_t(x)/\Sigma_{maj}$ (Leppanen, 2013).

These options have significant implications for multiphysics feedback because continuous distributions for temperature and density imply continuous variation in Σ_t . The rejection sampling approach is perhaps the most generalized technique for integrating feedback. For example, the Serpent 2 code can read temperature/density distributions from an OpenFOAM unstructured mesh, allowing feedback to be decoupled from the geometry. At a potential collision site $\vec{x}$, the local temperature/density is fetched from the mesh representation at $\vec{x}$.

However, most Monte Carlo codes use uniform-property cells, either because they (i) do not support rejection sampling and/or (ii) they lack a mechanism to read Σ_t from a continuously-varying distribution. OpenMC is one such code that does not yet fully support rejection

sampling. As OpenMC is the software used in this paper, all remaining discussion is based on the presumption of surface tracking using Eq. (2). Key differences from a rejection sampling approach will be noted where relevant.

For Monte Carlo codes using surface tracking in tandem with Eq. (2), multiphysics coupling requires pre-dividing the domain into “extra” cells ready to receive spatially-varying temperature and density fields. This concept is shown in Fig. 2 for applying a mesh-based temperature distribution (top) to CSG cells (bottom). Depending on how the CSG cells are constructed (nine possible variants are shown), particles will transport through different temperature/density fields. The specification of the CSG cell subdivisions is somewhat arbitrary. Ideally, each analyst would perform a CSG cell “refinement” study to understand the necessary resolution and make an informed choice for their system of interest. However, this process is tedious and often established by trial-and-error, and increases the barrier to adoption of Monte Carlo multiphysics simulation.

Another complexity with CSG is the additional effort needed to establish the mechanism by which the cells map to a general unstructured mesh. The simplest approach is to enforce a 1:1 correspondence between cells and mesh elements, though this often results in excessively high cell counts (Gurecky and Schneider, 2016) (degrading the tracking rate) and can be tedious to construct. Using Fig. 2 as an example, a 1:1 correspondence would require constructing the CSG cells to exactly match the element boundaries in the unstructured mesh used for temperature. More common is a conformal mapping, where N mesh elements are fully contained within a single cell (Romano et al., 2021), and the properties of the cell are taken as a volume-average over those N elements. This option may still require significant manual intervention during the meshing process to ensure that element surfaces do not intersect cell edges. For CFD methods in particular, developing a quality mesh can occupy 80% of the analyst’s time (Boggs et al., 2005); imposing additional constraints to facilitate data transfers to a Monte Carlo is undesirable.

A slight variation, a non-conformal mapping instead allows the N elements to span the cell boundaries (Novak et al., 2022a) (such as with a centroid cell-lookup). An example of a non-conformal mapping is shown in the lower right image of Fig. 2, which is shown in enlarged detail in Fig. 3.

Another very significant downside with CSG is its inability to easily represent arbitrary geometries. Building a CSG model of arbitrary curves requires numerous stages of manual decomposition and approximation.

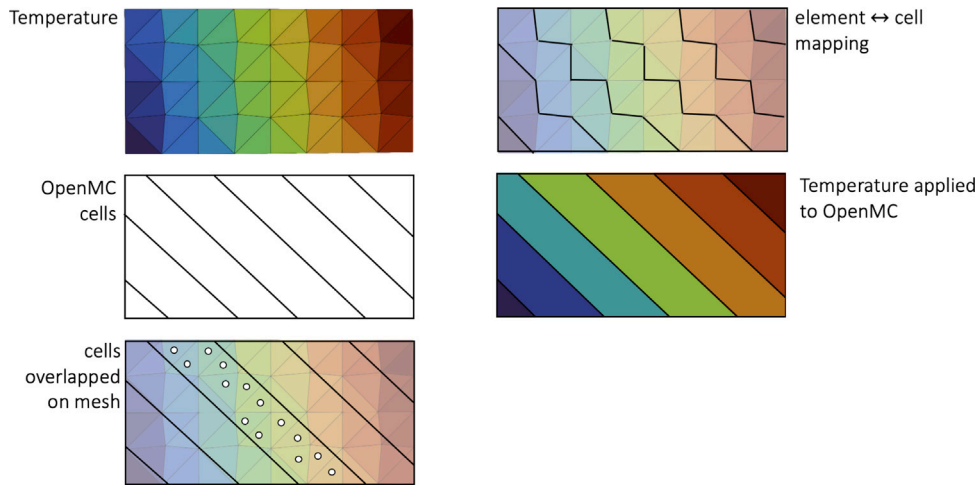


Fig. 3. Example of a non-conformal mapping based on element centroids (white dots in lower left image).

Recognizing these deficiencies of multiphysics coupling to CSG, we now turn to mesh-based Monte Carlo methods. Mesh-based geometries are a natural complement to multiphysics coupling to mesh-based codes. A CAD model may already be generated due to other code requirements, which could reduce the upfront effort to prepare models. Equally important is the wide availability of software libraries which support fast and robust spatial data transfers between different unstructured meshes, such as MOOSE (Permann et al., 2020), preCICE (Chourdakis et al., 2022), and SALOME (Bergeaud and Lefebvre, 2010). Such libraries can eliminate the need to develop custom scripts for transferring data between CSG and mesh-based solvers (a major hurdle in past multiphysics simulations based on CSG).

As described in Section 1.1, surface mesh representations offer the same general-geometry benefits of volume mesh representations, but with lower memory requirements. However, surface mesh geometries are confronted with some of the same issues encountered with CSG — the model should be pre-divided into smaller regions in order to represent temperature/density feedback with sufficient resolution. However, this process is just as tedious and complex as pre-dividing CSG cells.

This paper describes an approach for multiphysics feedback on adaptive unstructured surface mesh geometry which eliminates the need to pre-divide regions to receive temperature/density feedback. The algorithm is intended to reduce the barrier-to-entry for high-fidelity multiphysics by leveraging widely-used CAD model preparation workflows and an easily tunable refinement process. Section 2 describes the software tools leveraged in this work, which include OpenMC Monte Carlo transport (Romano et al., 2015), DAGMC (Tautges et al., 2009) surface mesh ray tracing, and two different approaches for integrating OpenMC into the Multiphysics Object-Oriented Simulation Environment (MOOSE) framework. Section 3 then discusses the adaptive unstructured mesh multiphysics algorithm, its software implementation, and a series of verification tests. Finally, Section 4 demonstrates the method for a multiphysics simulation of the Molten Salt Fast Reactor (MSFR) using OpenMC neutron transport coupled to NekRS spectral element CFD using Large Eddy Simulation (LES). Details of computational performance are given in Section 4.3, which indicate promising future avenues of study. Conclusions are given in Section 5.

2. Computational tools

The present research is conducted using open-source software which combine high-fidelity multiphysics solvers with a finite element/volume ecosystem for nuclear engineering modeling and simulation. This section briefly describes these tools.

2.1. MOOSE framework

The MOOSE framework is an open-source project for solution of coupled systems of nonlinear equations using the finite element and finite volume methods (Permann et al., 2020). Key strengths of MOOSE are its MultiApp and Transfer systems, which allow construction of tree-like code coupling hierarchies of arbitrary depth with flexible mesh-to-mesh solution transfers. An example of a MOOSE coupling between four different physics solves, with mesh-to-mesh transfers between different and non-overlapping meshes, is shown schematically in Fig. 4. The microstructure images (MultiApp C) are meant to convey the flexibility of MOOSE's multiscale approach, which can embed lower-length scale simulations within coarser-scale simulations.

Multiscale and multiphysics analyses are both straightforward and efficient to construct, leading to broad adoption of MOOSE as a framework-of-choice for both building new simulation capabilities (Williamson et al., 2021; Lee et al., 2021; Spencer et al., 2021; Hansel et al., 2021; Novak et al., 2021) as well as a platform for coupling existing codes (Brooks and Davis, 2023; Novak et al., 2022a). Another salient feature is a Quality Assurance (QA) software development program (Slaughter et al., 2021), which facilitates use for safety-relevant nuclear applications.

2.2. Cardinal

In recent years, MOOSE has added the capability for coupled solves with “external” applications that are based on entirely different solution methodologies and software stacks. An external code can be “wrapped” into MOOSE by overriding a few key interface functions that initialize, run, and postprocess results for the external code, while exposing the time stepping, synchronization, and data transfer systems in MOOSE.

Argonne National Laboratory (Argonne National Laboratory (ANL)), the University of Illinois at Urbana-Champaign (UIUC) and the United Kingdom Atomic Energy Authority (UKAEA) have been developing MOOSE wrappings of the OpenMC Monte Carlo code for high-fidelity multiphysics feedback in neutron-photon transport. AURORA (Brooks and Davis, 2023) (<https://github.com/aurora-multiphysics/aurora>) is the OpenMC-MOOSE coupling developed by the UKAEA, while Cardinal (Novak et al., 2022a) (<https://github.com/neams-th-coe/cardinal>) is the OpenMC-MOOSE coupling developed by UIUC/ANL. These two codes originally employed very different geometry models due to their initial focus on applications in fusion (for AURORA) and fission (for Cardinal). However, there are considerable overlaps between the simulation needs for nuclear fission and fusion. Both domains

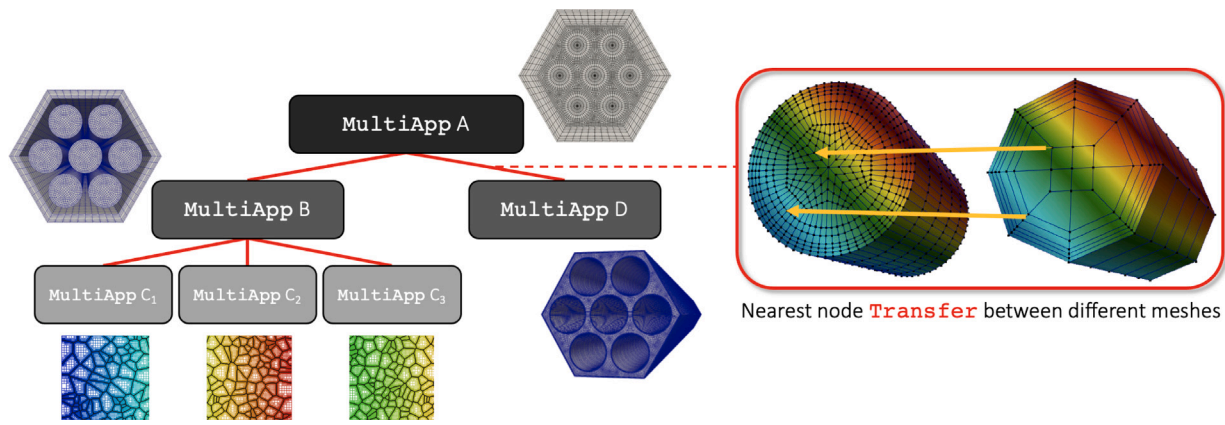


Fig. 4. MOOSE provides flexible infrastructure for multiscale and multiphysics analysis.

are concerned with interactions between particle transport, thermal-hydraulics, structural mechanics, material performance, and more. Ambitious development timelines are also proposed in each, with many first-of-a-kind advanced fission reactor operations targeting the 2026–2030 year time frame, and the UKAEA Spherical Tokamak for Energy Production (STEP) project targeting a prototype fusion plant in 2040. Given the extensive effort to stand up the “mechanics” of code coupling (data transfers, parallel communication, etc.) and the extensive software Verification and Validation (V&V) efforts required for licensing, there is a strong motivation to leverage development efforts across multiple institutions in the pursuit of a high-fidelity multiphysics simulation framework.

The software outcome of this joint effort is a porting of AURORA’s CAD surface mesh multiphysics algorithms into the Cardinal code base to enable new high-fidelity simulation couplings to NekRS. More details on the software-related impetus for this porting can be found elsewhere (Novak et al., 2023a). To avoid confusion between the capabilities of AURORA and Cardinal and to avoid oversaturation of this algorithmic discussion with library names, this paper will solely discuss the capabilities in Cardinal after this porting effort was completed. It should be recognized that the original conceptualization of the mesh skinning algorithm (described shortly) was developed by the UKAEA and first implemented in AURORA (Brooks et al., 2022; Brooks and Davis, 2023).

After this software porting, Cardinal now supports multiphysics coupling with two forms of OpenMC geometry, using either (i) DAGMC surface meshes or (ii) CSG. Irrespective of the geometry choice, the overall multiphysics algorithm is the same. First, the OpenMC heating distribution is tallied and mapped to a finite element basis on an unstructured MOOSE mesh. Then, MOOSE’s mesh-to-mesh Transfer system sends that tally to a coupled code, where it may be used as a volumetric heat source. For the inbound transfer, temperature T and density ρ are volume-averaged over all elements e which map to an OpenMC cell c , and the cell temperature and density are updated in OpenMC. Depending on the energy range in question and user settings, updates to temperature either use an on-the-fly Doppler broadening method or stochastic interpolation between cross section libraries. Convergence is obtained using Picard iteration. Now, the differences in Cardinal’s two geometry representations are described.

Prior multiphysics studies using Cardinal leveraged CSG (Novak et al., 2022a, 2023b; Merzari et al., 2021; Eltawila et al., 2024; Hegazy et al., 2024; Jiang et al., 2023; Chen et al., 2023). For these CSG models, Cardinal loops over the elements in a coupled mesh (for example, a heat conduction mesh), and maps each element to an OpenMC cell according to a centroid lookup. The CSG mapping is non-conformal, and does not require strict preservations of boundaries or volumes. Cardinal’s non-conformal CSG model was depicted in Fig. 3, and is shown again in Fig. 5. The inset in the left panel of Fig. 5 shows

an actual boundary between two CSG cells as a dashed white line. Elements straddling cell boundaries are mapped to a single CSG cell based on their centroid.

An alternate geometry representation (originally developed in AURORA) is a recent addition to Cardinal and the focus of this paper. This geometry is based on DAGMC surface meshes and is shown in Fig. 6 in a simplified form. After each Picard iteration, elements in a coupled unstructured mesh are “binned”/“contoured” according to temperature and density. This grouping is performed according to a user-specified “bin width” (e.g. 25 K). Then, the elements that occupy each bin are grouped and “skinned” to generate the closed DAGMC surface mesh bounding that region. OpenMC cells are then redefined using the new surface mesh regions. The skinning process invokes functionality in the MOAB mesh library to create triangulated surfaces from the boundary faces of collections of tetrahedral elements. In other words, the OpenMC geometry is adaptive, and changes in response to the coupled physics fields — eliminating a major hurdle in CSG-based multiphysics with surface tracking. For a small bin width (e.g. 0.1 K), this re-skinning results in a 1:1 correspondence between OpenMC cells and mesh elements and is equivalent to tracking particles on a volumetric unstructured mesh. Therefore, Cardinal now also supports a form of volumetric unstructured mesh coupling with OpenMC (but which begins from a surface mesh, rather than a volumetric mesh).

Cardinal also contains a NekRS–MOOSE coupling independent of the OpenMC–MOOSE coupling. NekRS (Fischer et al., 2022) is a spectral element CFD code with capabilities for Reynolds Averaged Navier Stokes (RANS), LES, and Direct Numerical Simulation (DNS) with both CPU and GPU backends. Cardinal supports several modes for coupling NekRS to MOOSE, including (i) boundary conjugate heat transfer, (ii) volumetric heat transfer, and (iii) systems-level coupling to CFD (Novak et al., 2022a, 2023b; Merzari et al., 2021; Huxford et al., 2023; Novak et al., 2022b). As the emphasis of this work is on CAD-based neutron transport multiphysics, we refer the reader to the literature for further details on how data transfers occur between NekRS and MOOSE (Novak et al., 2022a).

3. Adaptive unstructured mesh multiphysics in cardinal

In this section, a more detailed discussion of the DAGMC surface mesh multiphysics algorithm is described, including implementation details and verification tests.

3.1. Implementation details

The adaptive surface mesh geometry in Cardinal regenerates cell boundaries in response to temperature and density feedback, while also respecting material regions. To describe the full complexity of the

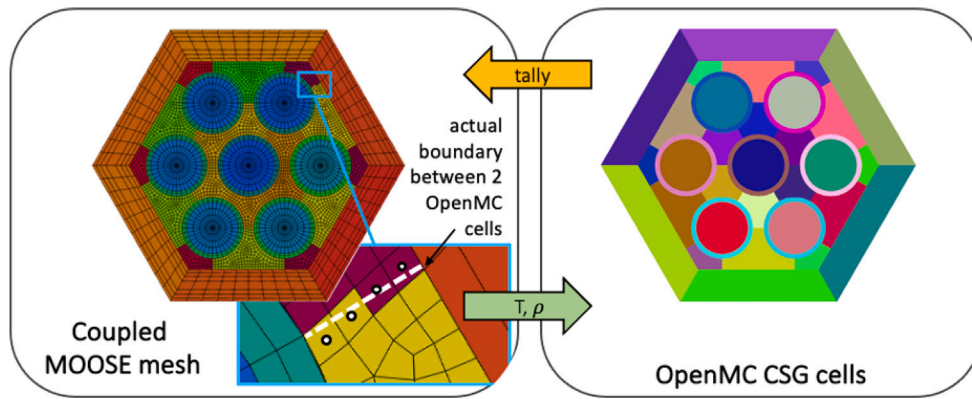


Fig. 5. CSG multiphysics coupling in Cardinal. The colors in the left image indicate the OpenMC cell ID to which each element maps.

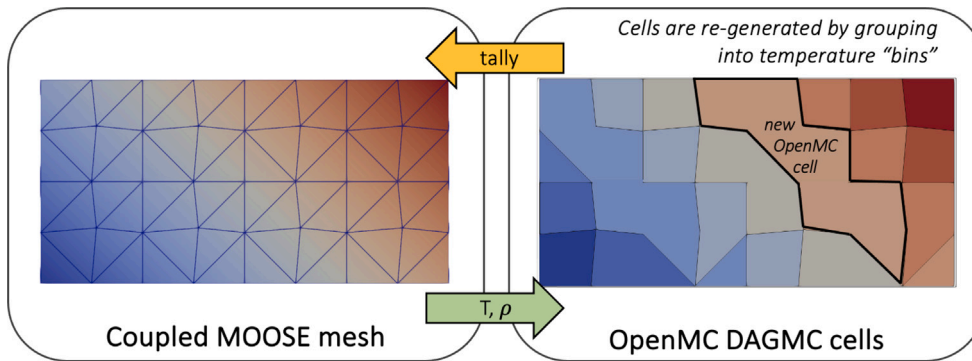


Fig. 6. DAGMC surface mesh multiphysics coupling in Cardinal.

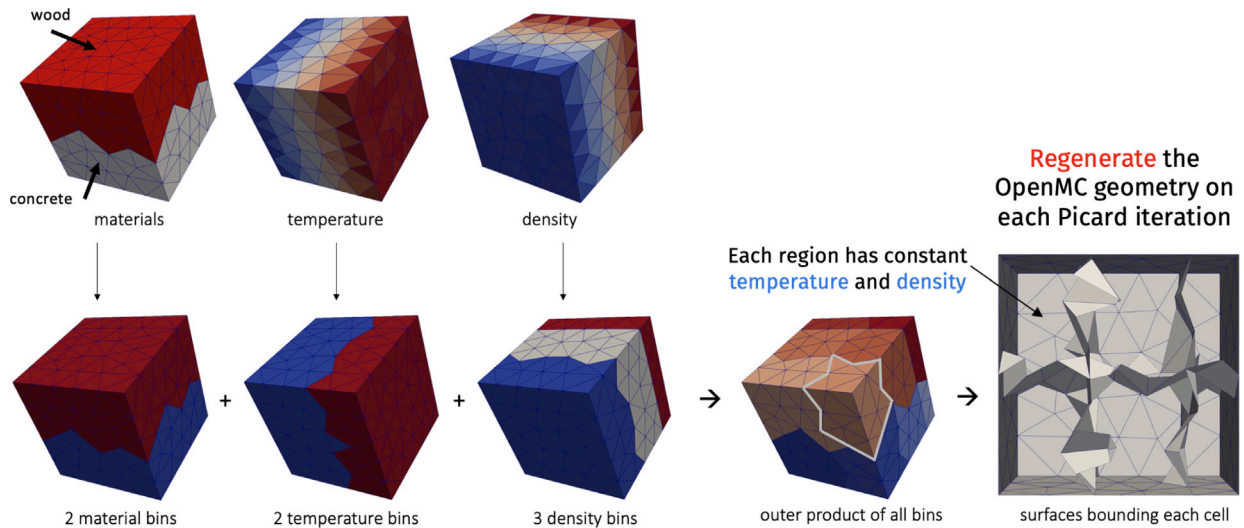


Fig. 7. Detailed depiction of DAGMC adaptive mesh geometry in Cardinal.

algorithm, Fig. 7 shows a hypothetical problem comprising a cube composed of two materials (e.g., wood and concrete) subject to gradients in temperature and density. These problem specifications are shown in the top row of Fig. 7 and are selected arbitrarily just for illustration.

Within Cardinal, the user specifies the binning/contouring interval separately for temperature and density. Supposing temperature to be in the range of 500 K to 600 K, choosing a temperature contouring interval of 50 K will mark elements into two groups: (i) Bin 0, with all elements for which $500 \leq T \leq 550$; and (ii) Bin 1, with all elements for which $550 < T \leq 600$. Supposing density to be in the range of 1.1 kg m^{-3} to 1.4 kg m^{-3} , selecting a contouring interval of 0.1 kg m^{-3}

will mark elements into three groups: (i) Bin 0, with all elements for which $1.1 \leq \rho < 1.2 \text{ kg m}^{-3}$; (ii) Bin 1, with all elements for which $1.2 \leq \rho < 1.3 \text{ kg m}^{-3}$; and (iii) Bin 2: all elements for which $1.3 \leq \rho \leq 1.4 \text{ kg m}^{-3}$. These bin “tags” are shown using colors in the bottom row of Fig. 7.

After the bin indices for material, temperature, and density are identified, each mesh element will then be grouped into a unique DAGMC cell for every unique combination of tags. In this example, there are $2 \cdot 2 \cdot 3 = 12$ new cells formed, one for each unique combination of material tag (2), temperature tag (2), and density tag (3) which exist in the problem. One bin is shown outlined in white in the bottom row,

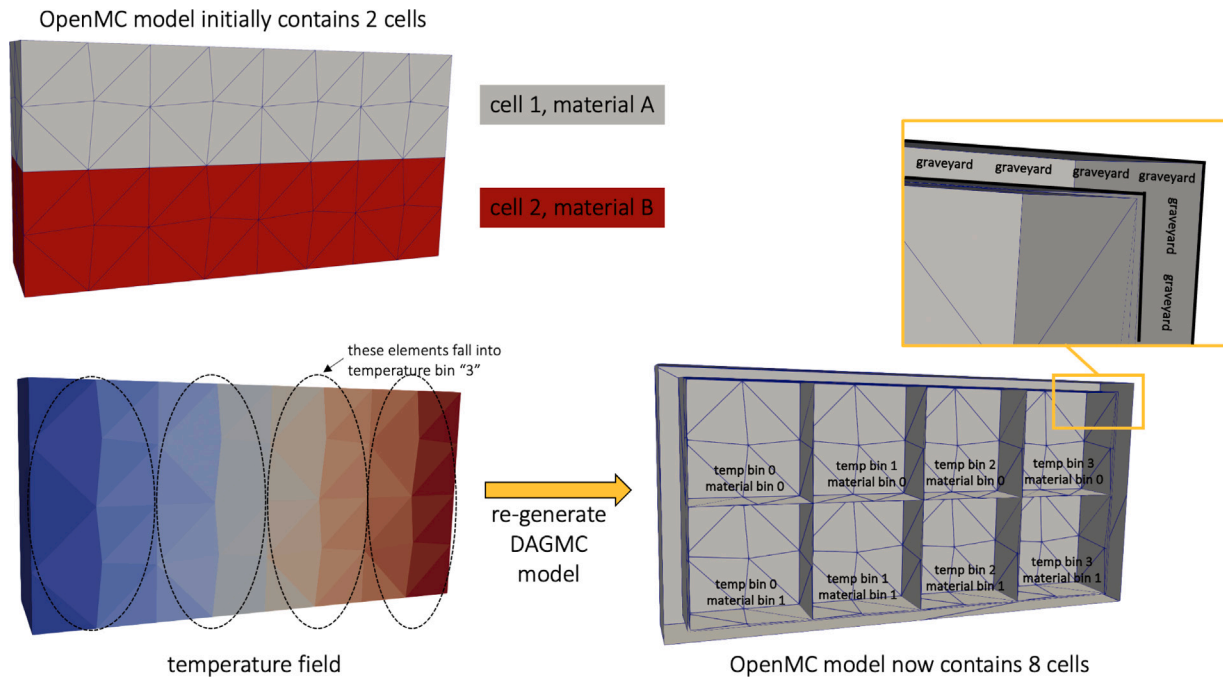


Fig. 8. Verification case used for assessing surface skinning algorithm.

and a cutaway slice through the skinned geometry is shown in the bottom right.

The implementation is designed for flexibility and usability. The unstructured mesh on which OpenMC reads temperature and density for skinning must be a tetrahedral volume mesh (because DAGMC only supports ray tracing between triangular surface meshes). However, MOOSE's mesh-to-mesh data transfers allow this tetrahedral volume mesh to be coupled to any other MOOSE application (which may use an entirely different mesh). This aspect will be leveraged in Section 4, where OpenMC is coupled to a NekRS hexahedral mesh. Furthermore, the user has two parameters with which to instantly vary the resolution of multiphysics feedback — the contouring intervals for (i) temperature and (ii) density. Refinement studies are straightforward to perform, and do not require modification of the original model. This simple refinement process is not possible for CSG models, as the addition of new surfaces to divide space must be created by the modeler manually.

The necessary input files include (i) the usual OpenMC XML input files; (ii) the usual DAGMC surface mesh geometry, in .h5m format; and (iii) the matching volumetric version of this surface mesh geometry, to use for the skinning (in any mesh format supported by MOOSE's FileMesh object). For more information on the user interface for the on-the-fly skinning, a tutorial for a tokamak and the MSFR example in Section 4 is available on the Cardinal website (Novak, 2024).

As one point of contrast, Serpent accomplishes multiphysics feedback by combining mesh-based data with rejection sampling (a.k.a delta tracking), which separates the notion of geometry and state (Leppanen, 2013). However, rejection sampling is restricted to collision estimators (Leppanen, 2010). Conversely, MOAB cell tallies support tracklength estimators, including on the re-skinned geometries.

3.2. Verification tests

A series of verification tests were incorporated in Cardinal's regression test suite to ensure equivalence between the skinned geometries and CSG representations. Here, one of these verification tests (Fig. 8) is described in order to convey the overall testing approach.

The domain consists of a slab containing two materials, each with a different enrichment of U-235 in uranium. These two materials are indicated as "material A" and "material B". A functional form for a temperature gradient is imposed (lower left), which is then skinned into four bins. This particular temperature gradient is selected such that the resulting contours result in skinned surfaces which can exactly be represented using CSG. After skinning, the model will then comprise eight cells (four temperature bins and two material bins). The region denoted as "graveyard" is related to the application of vacuum Boundary Conditions (BCs).

The skinning operation causes the initially two-cell model to be refined into eight cells, with each cell having a unique combination of temperature and material. Offline, an identical eight-cube CSG version of this geometry was constructed for comparing tallies (k , heating deposition, flux) with the skinned geometry. Identical results are obtained using CSG, which verifies the implementation. More complex test cases, which can be found in the Cardinal repository, use combinations of temperature and/or density skinning, mesh tallies, and relaxation to provide additional confidence in the implementation.

4. Fast spectrum molten salt reactor

In this section, the adaptive unstructured mesh geometry in Cardinal is applied to couple OpenMC CAD neutron transport to NekRS LES for a fast-spectrum MSR. Most fast-spectrum MSR design concepts use curved reactor vessel walls to enhance fluid mixing, which usually require mesh-based radiation transport. The current objective of our study is to present a large-scale, proof-of-concept demonstration of the adaptive mesh algorithm on a semi-realistic reactor geometry. Because several model simplifications are taken, the physical predictions should not be construed as the actual simulation of a real system — but only as a challenging deployment of the adaptive mesh algorithm. Section 4.1 describes the computational models employed, Section 4.2 presents the physics predictions and adaptive mesh application, and Section 4.3 provides a preliminary exploration of the relationship between computational runtime and contouring strategy.

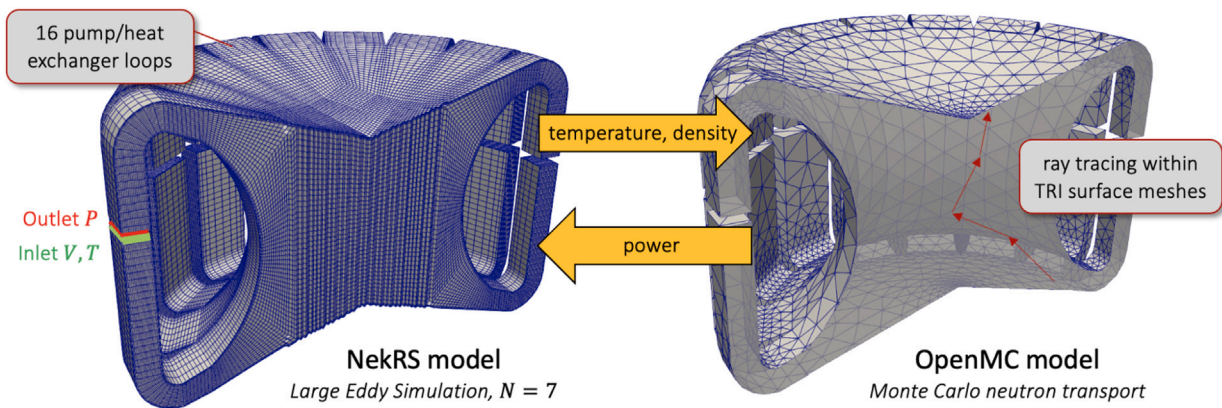


Fig. 9. Summary of the computational meshes used for NekRS and OpenMC, shown on a volume slice through the domain.

4.1. Computational model

The computational model is based on an open-source MSFR design (Rouch et al., 2014). This concept is a 16-leg design using a molten salt fuel with a nominal power of 3 GW, a bulk temperature rise from 898K to 998K, and a core flowrate of 18932 kg s^{-1} . The fuel salt is a lithium fluoride, actinide fluoride mixture. The full reactor is modeled by coupling NekRS LES with OpenMC neutron transport.

The OpenMC model uses DAGMC for ray tracing within triangular surface meshes in k -eigenvalue mode. An LES turbulence model is employed in the NekRS case to (i) demonstrate high-fidelity capabilities and (ii) attain an inherently time-varying fluid temperature field to drive the on-the-fly adaptivity. Images of the meshes used in each code are shown in Fig. 9. The NekRS mesh is composed of approximately 616,000 hexahedral elements of polynomial order $N = 7$ (about 315 million grid points), while the OpenMC mesh is composed of 35,186 tetrahedral elements. These two models are coupled, with OpenMC physics feedback in the form of (i) Doppler feedback to microscopic cross sections and (ii) density feedback to macroscopic cross sections. NekRS feedback is in the form of a power distribution evaluated by OpenMC.

As this analysis is intended to demonstrate on-the-fly geometry skinning, several simplifications are introduced. In the present work, the reactor is modeled at a 10% flow, 10% power condition (i.e. $300 \text{ MW}_{\text{th}}$ and 1893 kg s^{-1}) to be within an approachable range for LES.

Furthermore, Delayed Neutron Precursors (DNP) transport is neglected, though there is ongoing work to extend OpenMC to support DNP movement driven by velocity (e.g. from a CFD solver), which will relax this assumption in the future. The NekRS model is incompressible with constant thermophysical properties from Rouch et al. (2014). Density treatments including a Boussinesq buoyancy term or a low Mach formulation will be added in future work. In the OpenMC model, density feedback is instead approximated by evaluating a thermophysical property correlation $\rho(T)$ given temperature feedback from NekRS.

A number of approximations are made for geometric simplicity. The radial breeder blankets and axial reflectors are neglected. In addition, the heat exchanger and pump in each of the 16 legs are replaced by recycling inlet and turbulent outlet BCs for velocity. The recycling inlet BC applies the inlet velocity using the velocity distribution some distance upstream from the outlet boundary, essentially representing the inlet as fully-developed flow. The inlet temperature is then set to a uniform value of 898K. Vacuum BCs are applied to a bounding box enclosing the OpenMC mesh.

Within each Picard iteration, NekRS and OpenMC communicate via mesh-to-mesh interpolations through Cardinal. The NekRS temperature T is interpolated onto the OpenMC mesh, where the density is evaluated as $\rho(T)$. Then, the OpenMC model is re-generated on-the-fly according to contours in density and temperature. After completing the

following OpenMC solve, the fission power is tallied on an unstructured mesh and mapped with a mesh-to-mesh interpolation back to the NekRS mesh, where the heat source is applied as a volumetric power density. We emphasize that the meshes used by NekRS and OpenMC (shown in Fig. 9) are completely independent.

Each OpenMC solve is a steady, k -eigenvalue simulation. The NekRS solver, however, uses a transient LES solve (which is steady when averaged over a sufficiently long turbulence time window). In this analysis, the fluid temperature sent from NekRS to OpenMC is, for simplicity, taken as the instantaneous temperature field (as opposed to a moving-window time average). As such, conventional criteria for terminating Picard iterations (such as a relative L^2 norm between successive iterates) will never indicate coupled convergence except in the trivial limit of extremely loose tolerances. As this paper presents one of the first cases where LES is coupled to Monte Carlo transport, there remains much fundamental algorithmic questions to be answered but which are outside the scope of the present work. These questions include (i) the impact of time-averaging turbulent fields before passing to a neutron transport solver (and on what moving window to do so) and (ii) the ideal Picard iteration frequency considering computational cost and physical accuracy. These questions will be explored in future work. In the meantime, the present model is run for a fixed number of Picard iterations (10) with each Picard iteration defined to span 2000 NekRS CFD time steps. With these settings, the fluid completes at least one full flow-through time.

Despite these simplifications and caveats, this model is of sufficient complexity and realism to provide a useful demonstration of the adaptive unstructured mesh algorithms in Cardinal.

4.2. Results

The coupled simulation is run on the Summit supercomputer, requiring about 50 node hours per Picard iteration. Fig. 10 shows the (a) OpenMC fission power, (b) NekRS temperature, and (c) re-skinned OpenMC geometry on the last Picard iteration. Temperature is contoured into 15 bins (of width 20 K), while density is contoured into 15 bins (of width 20.6 kg m^{-3}). DAGMC cells must be contiguous in space, so in practice each Picard iteration results in a new OpenMC geometry with about 2500–3000 cells due to the complex turbulent flow field and the nature of the unstructured mesh. The black lines in Fig. 10(c) denote cell edges.

Fig. 11 shows the correspondence between the NekRS fluid temperature and the OpenMC cell temperature at two additional Picard iterates. Temperatures are shown on a shared color scale, ranging from 890 K to 1090 K. By careful inspection, the iteration-dependent nature of the OpenMC cells can be discerned by comparing the last column in Fig. 11. The geometry adaptivity automatically considers contours in temperature and density, eliminating the user ambiguity in defining CSG cells for multiphysics.

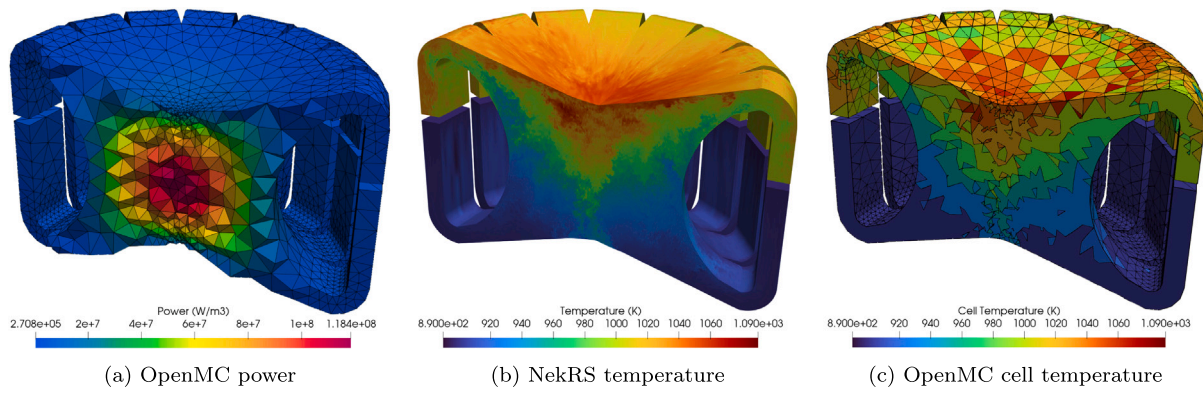


Fig. 10. (a) OpenMC power, (b) NekRS T , and (c) OpenMC T with cell edges in black, for the last Picard iteration.

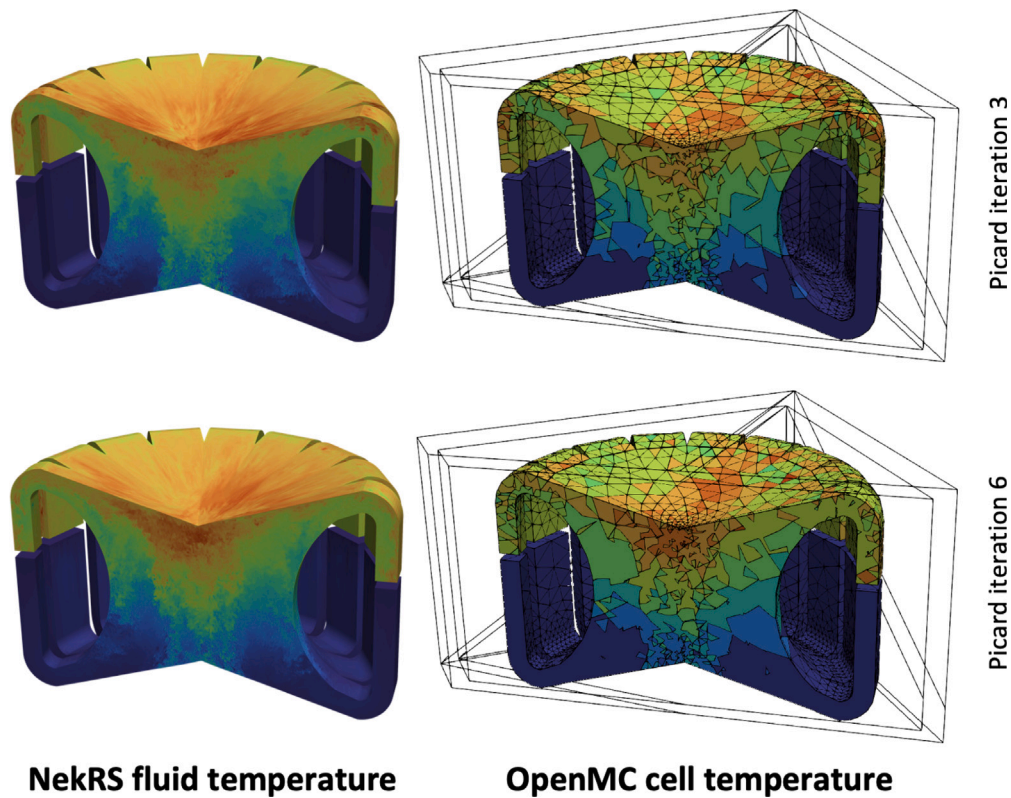


Fig. 11. NekRS fluid temperature (left column) and OpenMC cell temperature (right column) at two different Picard iterations (top row, bottom row).

For illustration only, Fig. 12 shows the same model, but using only 4 temperature bins (instead of 15). Due to the fine-scale turbulent LES, each temperature contour does not necessarily comprise a single contiguous region of space.

4.2.1. Mesh convergence

Next, we study the impact on k and the fission heat distribution as we vary the resolution of the temperature and density contours. Our objective is to illustrate how the on-the-fly skinning enables streamlined mesh refinement studies with a simple “tuning knob” (the width of the temperature/density bins), in contrast to the often highly manual geometry subdivisions required for refinement studies of CSG problems (which are therefore most often not performed/published). Here, the temperature and density distribution is “frozen” at the last Picard iteration (NekRS’s solve is disabled), and six separate OpenMC simulations are performed for increasingly fine resolution of temperature and density. Regardless of the skinning resolution, each of these six

OpenMC simulations has the same overall average temperature and density.

The details on these six runs, including numbers of cells and temperature/density resolution, are summarized in Table 1. For each case, we use the same number of bins for temperature and density using the temperature and density at the heat exchanger inlet/outlet conditions to bound this range. Each OpenMC run is performed with 50000 particles/batch, 500 inactive batches, and 1000 active batches. Table 1, as well as Fig. 13, shows the variation in k as the temperature and density distributions are refined. Mesh convergence is obtained by refining the contours. Note that even when using a fairly wide contouring interval, the small-scale details of LES result in high cell counts.

The temperature applied to each OpenMC model is shown in Fig. 14. As the contouring interval is refined, a more detailed temperature profile is applied to the neutron transport.

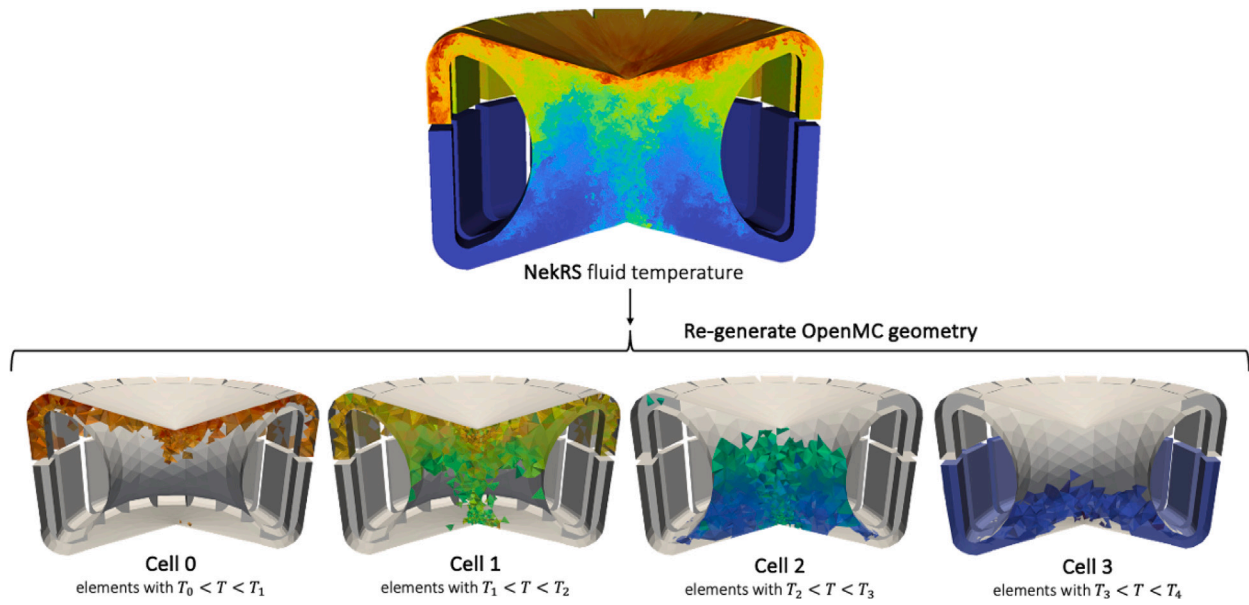


Fig. 12. Example grouping of mesh elements into four temperature bins.

Table 1
Skinning parameters used for mesh refinement study.

T Bin Width (K)	ρ Bin Width (kg m^{-3})	# OpenMC cells	k
350.0	308.7	1	0.94580 ± 4 pcm
175.0	154.3	399	0.94643 ± 4 pcm
87.5	77.2	442	0.94639 ± 4 pcm
43.8	38.6	1175	0.94515 ± 4 pcm
21.2	19.3	3080	0.94498 ± 4 pcm
10.9	9.6	7442	0.94496 ± 4 pcm

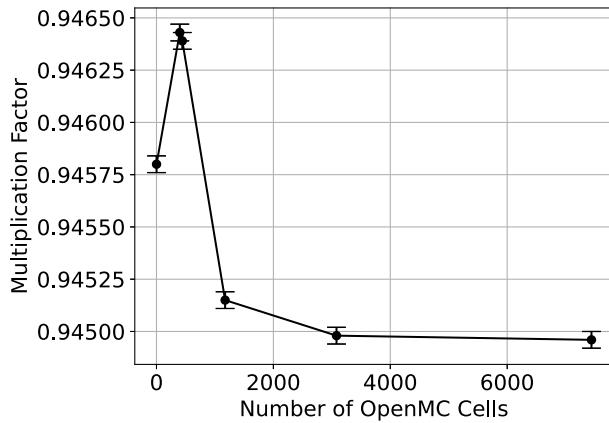


Fig. 13. Convergence of k as a function of refinement of OpenMC cells.

4.3. Computational performance

In Section 4.2, the temperature and density contouring intervals were selected based on a mesh refinement study, without consideration of their impact on runtime. The obvious physics choice for contouring interval would be to select very fine contours (e.g., 0.1 K) such that particles track through the temperature/density field without any additional numerical approximations induced by volume-averaging this temperature/density. However, it is expected that the particles/second rate simulated by OpenMC is influenced by the geometric detail. This section explores the impact of the contouring interval on OpenMC runtime in order to provide more detail on the selection of this parameter.

When skinning DAGMC models, an initially single-cell DAGMC geometry is split into smaller cells, which induces two competing effects on the runtime related to (i) sourcing a particle from the fission bank and (ii) the distance-to-surface calculation. When sourcing a particle from the fission bank, OpenMC does not currently store the particle's birth cell information. An OpenMC model with a higher number of cells has an increased cost of the “find cell” routine when sourcing a particle from the fission bank because there are more cells to check. The find-cell call is a point-containment operation, which in DAGMC currently scales as $\mathcal{O}(N)$, where N is the number of cells.

However, as each cell decreases in size, the ray fire operation accelerates because the number of triangular surface elements per cell is lower (faster distance-to-surface calculation, due to a shallower tree search and fewer bounding box intersection checks). The competing nature of these two effects suggests there may exist an optimal binning strategy (from the perspective of runtime, and not necessarily multi-physics accuracy — which would instead be assessed through a mesh convergence study such as in Section 4.2.1).

To explore these two find-cell and ray tracing hot spots further, the MSFR model was re-run with uniform T and ρ to eliminate any possible influence due to physical properties. Cardinal's object-oriented design allows the skinning to be driven by *any* field data, which we use to our advantage by simply dividing the MSFR mesh into a Cartesian grid. This is illustrated in Fig. 15(a) for a $2 \times 2 \times 2$ grid, which results in $\geq 2^3$ cells because the skinning operation will only create contiguous cells (and some cells here are “split” by the gap between the inlet and outlet of the 16 flow legs and by the unstructured mesh behavior along the grid edges).

The MSFR case is run for grids with $n = [1, 2, 4, 8, 12, 16, 24, 32, 40]$, which produce OpenMC models ranging from 1 to 17083 cells. The volume mesh contains 35186 elements, so this skinning is creating OpenMC models with between 35186 elements/cell and ~ 2.0 elements/cell (nearly equivalent to true unstructured mesh tracking). All runs are performed with 10^5 particles/batch, for a fixed number of batches, in serial — the only difference is the number of (materially-identical) cells. The simulation is also run without any active batches, to isolate runtime effects from particle tracking and remove additional effects from tally operations. The overall runtime of the simulation and functions are collected using additional timers for geometric operations in OpenMC. The next two sections delineate general observations regarding performance, followed by alternative find-cell and ray tracing algorithms tested in DAGMC.

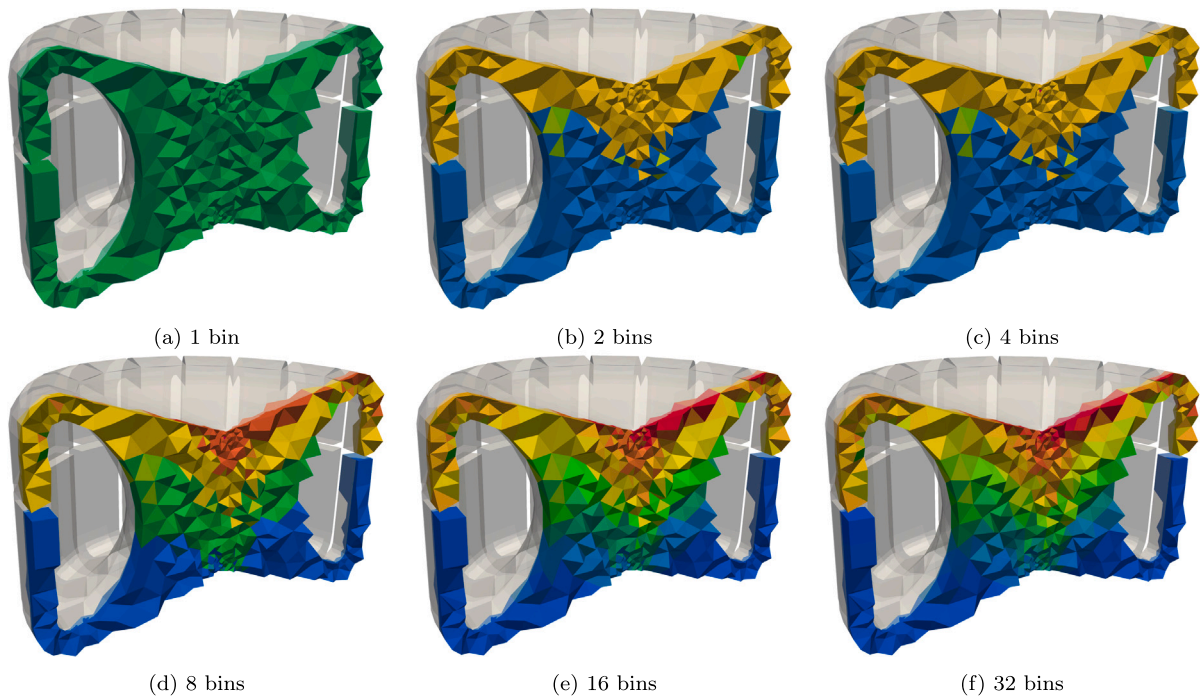


Fig. 14. Fluid temperature applied to the OpenMC model, adjusted by modifying the contouring interval. All images are shown on the same color scale, for $898 < T < 1073$ K.

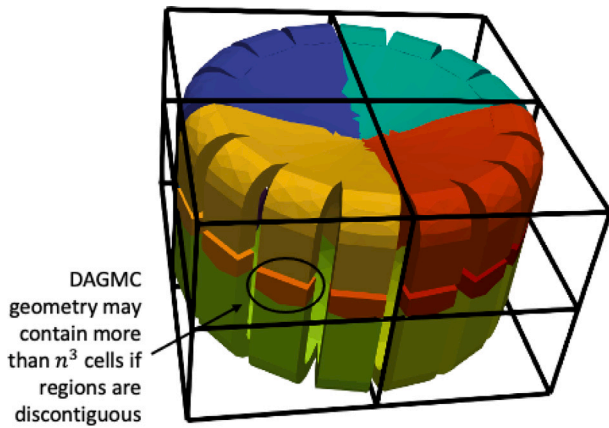


Fig. 15. Example skinning for an $2 \times 2 \times 2$ grid.

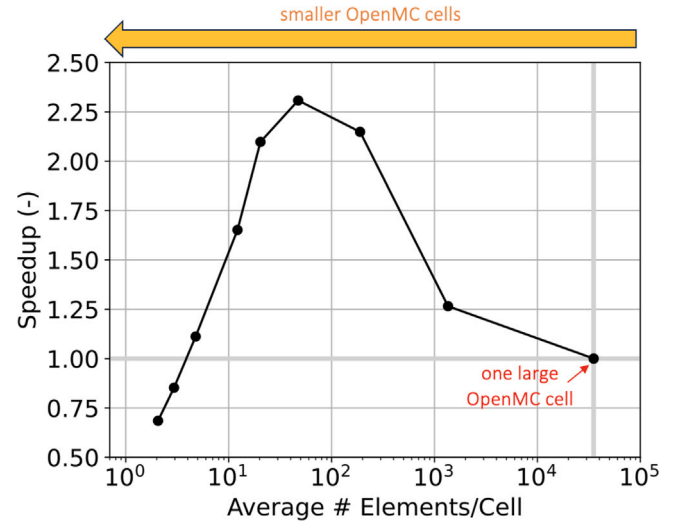


Fig. 16. Overall runtime and speedup of the OpenMC simulation as a function of OpenMC cell size.

4.3.1. General performance observations

Fig. 16 shows the speedup of the OpenMC simulation as a function of the average number of elements per OpenMC cell. The speedup is computed relative to a geometry consisting of a single large cell. A maximum speedup of 2.31 is observed for an OpenMC model with ~ 47 mesh elements/cell. A net loss in performance (speedup < 1) is only observed with very fine skinning (~ 3 or less mesh elements per cell).

Fig. 17 shows the runtime of the two dominant function calls (the find-cell and ray firing operations), as the OpenMC cell size is varied. As the number of cells increases, the runtime of (i) the “find cell” operation increases, while (ii) ray fire runtime initially decreases and eventually saturates. It is expected that the peak performance observed of ~ 47 elements/cell is strongly problem-dependent, and related to parameters such as (i) the ratio between the mean free path and cell size; (ii) cell complexity (e.g., surface area/volume ratio); and (iii) cell aspect ratio. In-depth study of these effects, and whether they can be optimized to enhance ray tracing performance in unstructured meshes, will be part of future work. However, it is interesting to note that the skinning algorithm can be used to “unlock” this performance gain,

which would otherwise not be accessible when using purely volumetric mesh tracking (one element/cell) or easily generated during the initial CAD-based geometry preparation.

It is also notable that, during a Cardinal run a skinned DAGMC .h5m geometry file can be output on each Picard iteration and loaded as geometry in a standalone OpenMC run, such that the skinning algorithm can also be used as a preprocessor to accelerate standalone OpenMC runs.

4.3.2. Performance studies

To further explore the nature of the particle tracking rate performance, we explored two modifications to DAGMC both individually and in concert with one another. First, an alternate algorithm for the find-cell call in DAGMC was implemented, in the aims of improving

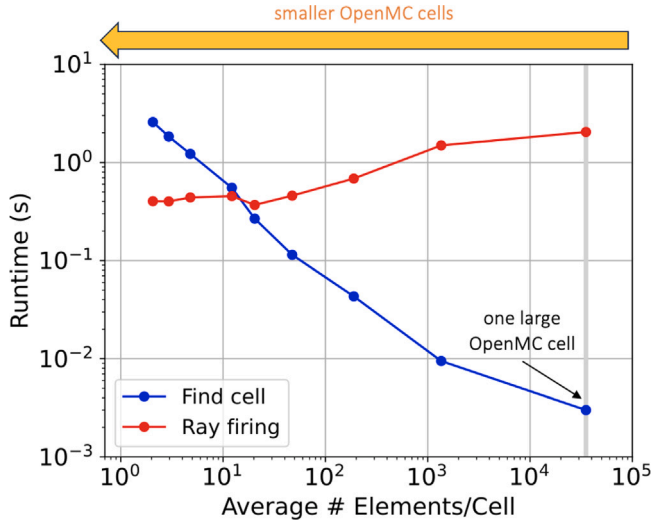


Fig. 17. Runtime associated with two dominant function calls in DAGMC as the OpenMC cell size is varied.

performance for the fine-scale cell geometries where this function call dominates runtime. The original find-cell operation in DAGMC fires a ray for each cell in the model, querying every volume for point containment. A new algorithm was implemented that performs an additional joining of all surfaces in a DAGMC model into a single Bounding Volume Hierarchy (BVH) tree. This additional global surface tree is used to accelerate the find-cell call because during particle flight, the particle's current containing cell BVH can be used for ray queries to exclude all irrelevant triangles from the query.

Both find-cell algorithms are shown conceptually in Fig. 18. In other words, an additional, global tree joins all surfaces to reduce the cost of a find-cell search in DAGMC from $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$, where N is the number of volumes in the model. With this data structure in place, a ray can be fired that may intersect with any surface in the problem. The surface intersection is then used to determine which volume the particle/ray is exiting via the topological information in the DAGMC model.

The nominal simulations performed in Figs. 16 and 17 used the native MOAB ray tracer, though DAGMC also supports hardware-accelerated ray tracing using Intel's Embree ray tracer (Shriwise et al., 2022b). The Embree ray tracer employs single-precision, vector operations, and axis-aligned bounding boxes, which can more efficiently leverage Single Instruction/Multiple Data (SIMD) operations. In order to study the effect of the ray tracer, the MSFR simulations were also repeated using the Embree ray tracer.

Fig. 19 shows the speedup from either (i) replacing the MOAB ray tracer by the Embree ray tracer (red line); (ii) replacing the find-cell call with the global tree algorithm (blue line); and (iii) applying both the changes simultaneously (black line). All speedup lines are computed

relative to the same baseline (the MOAB ray tracer with the original find-cell call). As expected, the modification to the ray tracer enhances performance for large, complex OpenMC cells with many triangles due to more efficient traversal of deeper tree structures when the number of DAGMC cells is low ($< 10^2$ cells). Conversely, the global tree find-cell call has a higher impact when a greater number of simpler DAGMC cells are present ($> 5 \times 10^3$ cells). Between these regions (10^2 to 10^3 cells), the tree depth decreases and the number of cells is not yet high enough to stress the original find-cell algorithm. When both modifications are applied simultaneously, the speedup for all cell counts is improved, but especially so for both the high- and low-cell count geometries. This inverted performance gain, relative to the speedup shown in Fig. 16, can be used to obtain more uniform performance numbers for these DAGMC simulations.

5. Conclusions

Open-source software can provide great benefits to the nuclear engineering community by allowing developments to be shared across project and institution boundaries. AURORA's innovative on-the-fly geometry re-generation process adaptively updates the OpenMC cells according to contours in temperature and density. This capability was ported into Cardinal and applied to a new and challenging high-fidelity multiphysics simulation of a fast-spectrum MSR. Full-core LES using NekRS was coupled with OpenMC neutron transport, presenting one of the first cases integrating unsteady turbulence modeling with radiation transport physics. As future work, the physical realism of the MSFR model will be improved by incorporating buoyancy terms in the NekRS model and delayed neutron precursor drift in OpenMC. Algorithmic questions related to Picard iteration strategies when using LES are also currently being explored.

The dynamic on-the-fly geometry regeneration reduces the barrier-to-entry for high-fidelity multiphysics by (i) eliminating the user ambiguity in defining cells for constant- Σ , surface tracking; (ii) enabling simple refinement studies which do not require additional model generation steps; and (iii) more closely integrating CAD workflows with Monte Carlo multiphysics. A performance study shows the impact of geometry subdivision on the find-cell and ray tracing calls, and suggests there may be performance improvements associated with the dynamic geometry regeneration. In the future, we will expand the performance study to include shared and distributed memory parallelism studies. We will also explore the influence of geometry (e.g. cell surface area/volume ratio) and physics (e.g. mean free path relative to cell size) on these performance numbers using a wider set of test cases in addition to the fast spectrum MSR studies in this paper. Additionally, future work should explore the possibility of body-fitted refinement. Such a development would require both (i) an adaptive baseline mesh (this work uses a fixed baseline mesh, and simply groups elements together according to temperature/density fields) and (ii) MOOSE's Adaptivity system to have knowledge of the underlying CAD geometry.

This research points towards many future lines of study. Because AURORA/Cardinal track on skinned versions of a MOOSE mesh, there

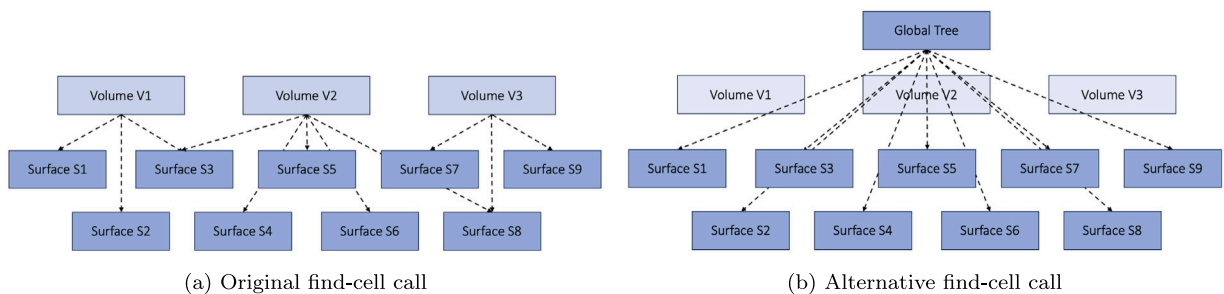


Fig. 18. Comparison between the (a) original and (b) improved find-cell call in DAGMC.

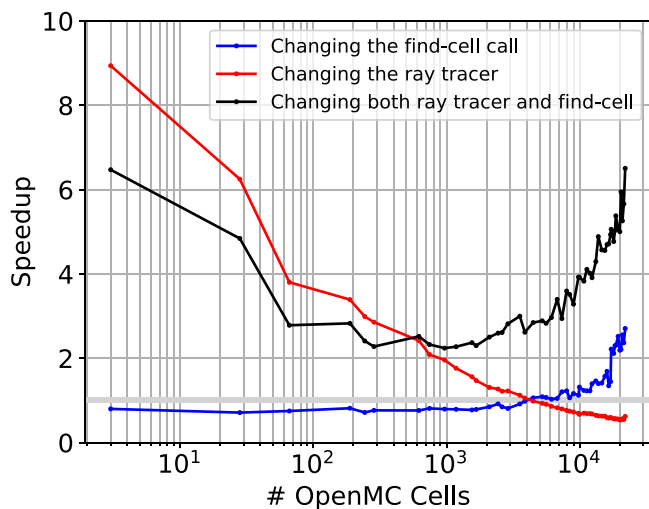


Fig. 19. Speedup in the MSFR case obtained when modifying the find-cell call, ray tracer, or both simultaneously.

is a natural opportunity for deforming mesh applications such as from thermal expansion. Ongoing work in the areas of fusion modeling, where complex CAD geometry is a necessity, will continue to expand the skinning algorithm's performance and add support for new features such as (i) in-line initialization of DAGMC via MOOSE, which will eliminate the need for duplicated mesh files (the .h5m surface mesh and an Exodus volume mesh); (ii) reflecting and periodic BCs; and (iii) multiphysics support for hybrid CSG/CAD geometry.

CRedit authorship contribution statement

A.J. Novak: Writing – original draft, Visualization, Software, Resources, Methodology, Investigation, Conceptualization. **H. Brooks:** Writing – review & editing, Software, Methodology, Conceptualization. **P. Shriwise:** Writing – review & editing, Writing – original draft, Investigation, Formal analysis, Data curation. **A. Davis:** Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The work of the ANL co-authors has been created by UChicago Argonne, LLC, Operator of ANL ("Argonne"). Argonne, a U.S. Department of Energy (DOE) Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. This material is based upon work supported by Laboratory Directed Research and Development (LDRD), United States of America funding from ANL, provided by the Director, Office of Science, of the U.S. DOE under Contract No. DE-AC02-06CH11357.

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. The authors also gratefully acknowledge Program Development funds from the ANL's Computational Sciences Division.

References

- Bennett, A., Bache, G., Martin, N., Senecal, J., 2019. BWR spacer grid modeling using serpent 2/STAR-CCM+ coupling. In: Proceedings of M&C.
- Bergeaud, V., Lefebvre, V., 2010. SALOME: A software integration platform for multi-physics, pre-processing and visualisation. In: Proceedings of SNA and M&C.
- Biondo, E., Davidson, G., Ade, B., 2022. Layered CAD/CSG geometry for neutronics modeling of advanced reactors. In: Proceedings of Physor.
- Boggs, P.T., Althsuler, A., Larzelere, A.R., Walsh, E.J., Clay, R.L., Hardwick, M.F., 2005. DART System Analysis. Technical report, SAND2005-4647.
- Brooks, H., Davis, A., 2023. Scalable multi-physics for fusion reactors with AURORA. Plasma Phys. Control. Fusion 65, 024002.
- Brooks, H., Dixon, S., Davis, A., 2022. Towards multiphysics simulations of fusion breeder blankets. In: Proceedings of Physor.
- Brown, F., Griesheimer, D., Martin, W., 2004. Continuously varying material properties and tallies for Monte Carlo calculations. In: Proceedings of PHYSOR.
- Chen, H., Wang, W., Wu, A., Zhang, K., Pan, R., Duan, W., Yang, G., Qin, C., Luo, X., 2023. Multi-physics coupling analysis of test heat pipe reactor KRUSTY based on MOOSE framework. Nucl. Eng. Des. 414, 112597.
- Chourdakis, G., Davis, K., Rodenberg, B., Schulte, M., Simonis, F., Uekermann, B., Abrams, G., Bungartz, H., Yau, L.C., Desai, I., Eder, K., Hertrich, R., Lindner, F., Rusch, A., Sashko, D., Schneider, D., Totounferoush, A., Volland, D., Vollmer, P., Kosemür, O.Z., 2022. preCICE v2: A sustainable and user-friendly coupling library. Open Res. Eur. 2 (51).
- Eltawila, M., Novak, A.J., Miao, Y., 2024. Coupled multiphysics modeling of heat pipe microreactors using cardinal, BISON, and sockeye. In: Proceedings of Physor.
- Ferraro, D., Alberto, P., Villarino, E., Doval, A., 2018. A multi-physics analysis for the actuation of the SSS in opal reactor. EPJ Nucl. Sci. Technol. 4, <http://dx.doi.org/10.1051/epjn/2018003>.
- Fischer, P., Kerkemeier, S., Min, M., Lan, Y., Phillips, M., Rathnayake, T., Merzari, E., Tomboulides, A., Karakus, A., Chalmers, N., Warburton, T., 2022. NekRS, a GPU-accelerated spectral element Navier-Stokes solver. Parallel Comput. 114, 102982.
- Große, D., Fischer, U., Kondo, K., Leichtle, D., Pereslavitsev, P., Serikov, A., 2013. Status of the mcdad geometry conversion tool and related visualization capabilities for 3D fusion neutronics calculations. Fusion Eng. Des. 88, 2210–2214.
- Gurecky, W., Schneider, E., 2016. Development of an MCNP6-ANSYS FLUENT multiphysics coupling capability. In: Proceedings of ICONE.
- Hansel, J.E., Berry, R.A., Andrs, D., Kunick, M.S., Martineau, R.C., 2021. Sockeye: A one-dimensional, two-phase, compressible flow heat pipe application. Nucl. Technol. 207, 1096–1117.
- Häußler, A., Fischer, U., Warmer, F., 2016. Verification of different Monte Carlo approaches for the neutronic analysis of a stellarator. In: Proceedings of the 29th Symposium on Fusion Technology.
- Hegazy, A., Novak, A.J., Uddin, R., 2024. Multiphysics simulation of TRISO fuel compacts and the effects of homogenization. In: Proceedings of Physor.
- Huxford, A., Leite, V.C., Merzari, E., Zou, L., Petrov, V., Manera, A., 2023. A hybrid domain overlapping method for coupling system thermal hydraulics and CFD codes. Ann. Nucl. Energy 189, 109842.
- Jiang, D., Xu, P., Hu, T., Jiang, X., Wang, L., Li, D., Zhang, X., Cao, L., 2023. Coupled Monte Carlo and thermal-hydraulics modeling for the three-dimensional steady-state analysis of the Xi'an pulsed reactor. Energies 16 (6046).
- Kirk, B.S., Peterson, J.W., Stogner, R.H., Carey, G.F., 2006. Libmesh: A C++ library for parallel adaptive mesh refinement/coarsening simulations. Eng. Comput. 22 (3–4), 237–254.
- Lee, C.H., Jung, Y.S., Park, H., Shemon, E.R., Ortensi, J., Wang, Y., Labouré, V.M., Prince, Z., 2021. Griffin Software Development Plan. Research Report INL/EXT-21-63185, ANL/NSE-21/23, Idaho National Laboratory, Argonne National Laboratory.
- Leppanen, J., 2010. Performance of woodcock delta-tracking in lattice physics applications using the serpent Monte Carlo reactor physics burnup calculation code. Ann. Nucl. Energy 37, 715–722.
- Leppanen, J., 2013. Modeling of nonuniform density distributions in the serpent 2 Monte Carlo code. Nucl. Sci. Eng. 174, 318–325.
- Leppanen, J., 2022. Methodology, applications and performance of the CAD-based geometry type in the serpent 2 Monte Carlo code. Ann. Nucl. Energy 176, 109259.
- Martz, R.L., 2017. The MCNP6 Book on Unstructured Mesh Geometry: User's Guide for MCNP 6.2. Technical Report LA-UR-17-22442, Los Alamos National Laboratory.
- Merzari, E., Yuan, H., Min, M., Shaver, D., Rahaman, R., Shriwise, P., Romano, P., Talamo, A., Lan, Y., Gaston, D., Martineau, R., Fischer, P., Hassan, Y., 2021. Cardinal: A lower length-scale multiphysics simulator for pebble-bed reactors. Nucl. Technol. 207.
- Novak, A.J., 2024. Cardinal: An open-source coupling of NekRS and OpenMC to MOOSE. <https://cardinal.cels.anl.gov>.
- Novak, A.J., Andrs, D., Shriwise, P., Fang, J., Yuan, H., Shaver, D., Merzari, E., Romano, P.K., Martineau, R.C., 2022a. Coupled Monte Carlo and thermal-fluid modeling of high temperature gas reactors using cardinal. Ann. Nucl. Energy 177, 109310.

- Novak, A.J., Brooks, H., Shriwise, P., Hegazy, A., Davis, A., 2023a. Multiphysics coupling of OpenMC CAD-based transport to MOOSE using cardinal and aurora. In: Proceedings of M&C.
- Novak, A.J., Carlsen, R.W., Schunert, S., Balestra, P., Reger, D., Slaybaugh, R.N., Martineau, R.C., 2021. Pronghorn: A multidimensional coarse-mesh application for advanced reactor thermal hydraulics. *Nucl. Technol.* 207, 1015–1046.
- Novak, A.J., Chaube, A., Shaver, D., Brooks, C., 2022b. Validation of nekrs-MOOSE conjugate heat transfer coupling for a 7-pin bare bundle. In: Proceedings of ANS.
- Novak, A.J., Shriwise, P., Romano, P.K., Rahaman, R., Merzari, E., Gaston, D., 2023b. Coupled Monte Carlo transport and conjugate heat transfer for wire-wrapped bundles within the MOOSE framework. *Nucl. Sci. Eng.*
- Permann, C.J., Gaston, D.R., Andrš, D., Carlsen, R.W., Kong, F., Lindsay, A.D., Miller, J.M., Peterson, J.W., Slaughter, A.E., Stogner, R.H., Martineau, R.C., 2020. MOOSE: Enabling massively parallel multiphysics simulation. *SoftwareX* 11, 100430.
- Poole, C.M., Cornelius, I., Trapp, J.V., Langton, C.M., 2012. Fast tessellated solid navigation in GEANT4. *IEEE Trans. Nucl. Sci.* 59 (4).
- Romano, P.K., Hamilton, S.P., Rahaman, R.O., Novak, A., Merzari, E., Harper, S.M., Shriwise, P.C., Evans, T.M., 2021. A code-agnostic driver application for coupled neutronics and thermal-hydraulic simulations. *Nucl. Sci. Eng.* 195, 391–411.
- Romano, P.K., Horelik, N.E., Herman, B.R., Nelson, A.G., Forget, B., Smith, K., 2015. OpenMC: A state-of-the-art Monte Carlo code for research and development. *Ann. Nucl. Energy* 82, 90–97.
- Rouch, H., Geoffroy, O., Rubiolo, P., Laureau, A., Brovchenko, M., Heuer, D., Merle-Lucotte, E., 2014. Preliminary thermal-hydraulic core design of the molten salt fast reactor (MSFR). *Ann. Nucl. Energy* 64, 449–456.
- Shriwise, P.C., Davis, A., Novak, A.J., Brooks, H.M., Romano, P.K., 2022a. Unstructured mesh capabilities and multiphysics simulation with openmc. In: Proceedings of the Technology of Fusion Energy.
- Shriwise, P.C., Wilson, P.P.H., Davis, A., Romano, P.K., 2022b. Hardware-accelerated ray tracing of CAD-based geometry for Monte Carlo radiation transport. *Comput. Sci. Eng.* 24 (2), 52–61.
- Shriwise, P.C., Zhang, X., Davis, A., 2020. DAG-openmc: CAD-based geometry in openmc. In: Proceedings of ANS.
- Slaughter, A.E., Permann, C.J., Miller, J.M., Alger, B.K., Novascone, S.R., 2021. Continuous integration, in-code documentation, and automation for nuclear quality assurance conformance. *Nucl. Technol.* 1–8.
- Song, J., Sun, G., Chen, Z., Zheng, H., Hu, L., 2014. Benchmarking of CAD-based SuperMC with ITER benchmark model. *Fusion Eng. Des.* 89, 2499–2503.
- Spencer, J.B., Alwin, J.L., 2019. Big 10 MCNP Unstructured Mesh Benchmark. Technical Report LA-UR-19-25731, Los Alamos National Laboratory.
- Spencer, B., Hoffman, W.M., Biswas, S., Jiang, W., Giorla, A., Backman, M.A., 2021. Grizzly and BlackBear: Structural component aging simulation codes. *Nucl. Technol.* 207, 981–1003.
- Tautges, T.J., Meyers, R., Merkley, K., Stimpson, C., Ernst, C., 2004. MOAB: A Mesh-Oriented Database. Technical Report SAND2004-1592, Sandia National Laboratories.
- Tautges, T.J., Wilson, P.P.H., Kraftcheck, J., Smith, B.M., Henderson, D.L., 2009. Acceleration techniques for direct use of CAD-based geometries in Monte Carlo radiation transport. In: Proceedings of M&C.
- Trammell, M.P., Jolly, B.C., Richardson, M. Dylan, Schumacher, A.T., Terrani, K.A., 2019. Advanced Nuclear Fuel Fabrication: Particle Fuel Concept for TCR. Technical Report ORNL/SPR-2019/1216, Oak Ridge National Laboratory.
- Turner, A., 2017. Investigations into Alternative Radiation Transport Codes for ITER Neutronics Analysis. Technical Report CCFE-PR(17)10, Culham Centre for Fusion Energy.
- Wang, X., Li, J., Wu, Z., Gao, S., Qiu, R., Deng, L., Li, G., 2020. CMGC: a CAD to Monte Carlo geometry conversion code. *Nucl. Sci. Tech.* 31.
- Wang, X., Yu, Y., Li, X., Feng, J., Zhang, B., Yan, C., Fu, Y., Xu, J., 2023. Development of a hybrid parallelism Monte Carlo transport middleware on mesh geometry. *Ann. Nucl. Energy* 190, 109872.
- Williamson, R.L., Hales, J.D., Novascone, S.R., Pastore, G., Gamble, K.A., Spencer, B.W., Jiang, W., Pitts, S.A., Casagrande, A., Schwen, D., Zabriskie, A.X., Toptan, A., Gardner, R., Matthews, C., Liu, W., Chen, H., 2021. BISON: A flexible code for advanced simulation of the performance of multiple nuclear fuel forms. *Nucl. Technol.* 207, 954–980.
- Wilson, P.P.H., Tautges, T.J., Kraftcheck, J.A., Smith, B.M., Henderson, D.L., 2010. Acceleration techniques for the direct use of CAD-based geometry in fusion neutronics analysis. *Fusion Eng. Des.* 85, 1759–1765.
- Yilmaz, S., Romano, P.K., Chierici, L., Knudsen, E.B., Shriwise, P.C., 2024. CAD and constructive solid geometry modeling of the molten salt reactor experiment with OpenMC. *Front. Nucl. Eng.* 3.