LA-UR-09-07320

*Approved for public release; distribution is unlimited.*

*Title:* Riding Bare-Back On Unstructured Mesh For 21st Century Criticality Calculations (U)

*Author(s):* Karen C. Kelley
Roger L. Martz
David L. Crane

*Intended for:* American Nuclear Society
PHYSOR 2010 Meeting
Pittsburgh, PA, May 9-14, 2010
and
Nuclear Science & Engineering

*PHYSOR 2010 – Advances in Reactor Physics to Power the Nuclear Renaissance*
Pittsburgh, Pennsylvania, USA, May 9-14, 2010, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2010)

# RIDING BARE-BACK ON UNSTRUCTURED MESHES FOR 21$^{ST}$ CENTURY CRITICALITY CALCULATIONS

**Karen C. Kelley, Roger L. Martz, and David L. Crane**
Los Alamos National Laboratory
PO Box 1663, Los Alamos, NM 87545
corzine@lanl.gov; martz@lanl.gov; dcrane@lanl.gov

## ABSTRACT

MCNP has a new capability that permits tracking of neutrons and photons on an unstructured mesh which is embedded as a mesh universe within its legacy geometry capability. The mesh geometry is created through Abaqus/CAE using its solid modeling capabilities. Transport results are calculated for mesh elements through a path length estimator while element to element tracking is performed on the mesh. The results from MCNP can be exported to Abaqus/CAE for visualization or other-physics analysis. The simple Godiva criticality benchmark problem was tested with this new mesh capability. Computer run time is proportional to the number of mesh elements used. Both first and second order polyhedrons are used. Models that used second order polyhedrons produced slightly better results without significantly increasing computer run time. Models that used first order hexahedrons had shorter runtimes than models that used first order tetrahedrons.

*Key Words*: Monte Carlo, CAD, CAE, unstructured mesh, geometry

## 1. INTRODUCTION

The Monte Carlo N-Particle (MCNP) transport code has a more general geometry capability than has been available in most combinatorial geometry codes [1]. In addition to the capability of combining several predefined geometrical bodies, as in a combinatorial geometry scheme, MCNP gives the user the added flexibility of defining geometrical regions from all the first and second degree surfaces of analytical geometry and elliptical tori and then of combining them with Boolean operators. This Constructive Solid Geometry (CSG) capability has been well-tested and verified and has exceptionally served users for decades. However, it has long been recognized that as the model complexity increases, the process of describing the geometry is difficult, tedious, time-consuming, and error-prone [2,3,4]. Consequently, innovators have taken on this task of developing a better way to construct geometries, not only for MCNP, but other particle transport codes as well.

To address the difficulty of building complex geometry models, three approaches have been taken which rely on a Computer Aided Design (CAD) system to create a geometry model. The first approach translates CAD surfaces into their equivalent representation using the Monte Carlo code's input specifications. The second approach uses the CAD model directly for Monte Carlo particle transport. Both of these approaches rely on particle tracking from surface to surface and have been driven by work in the fusion reactor and accelerator worlds. Numerous papers have been published in recent years describing these two approaches. The third approach creates a mesh representation of the solid geometry for the Monte Carlo code so that particles track directly on that mesh. This third approach is the one chosen for implementation in MCNP as a modular mesh-tracking library written in Fortran 90/95.

In very broad terms, Compute Aided Engineering (CAE) has been defined as the use of computer software to solve engineering problems. In this respect, CAE differs from CAD. For our purposes, we think of CAE to mean the computer solution of engineering problems with the assistance of interactive computer graphics to create a solid model representation that supports design, analysis, and simulation. The design, analysis, and simulation functionality generally relies on finite element methods which in turn require a mesh representation of the geometry.

Packages such as Abaqus/CAE [5] that are tightly integrated with finite element capabilities have the ability to generate an unstructured mesh representation of their solid models; generally, CAD programs do not have this functionality. The degree of fidelity between the CAE representation and the unstructured mesh is generally good and depends to a degree upon the user's willingness, ability, and need to refine the model. Neutral particles (neutrons and photons) can track directly on an unstructured mesh model in MCNP through a hybrid geometry environment that permits the existence of a mesh within the CSG representation. The amount of mixing of CSG and mesh geometries is at the discretion of the user, within certain limitations and restrictions.

This paper describes the third approach that uses a CAE mesh-model embedded into a CSG model, forming a hybrid one, to solve this problem. This approach is illustrated on a simple criticality benchmark problem; results are discussed. The paper is organized as follows: Section 2 discusses the hybrid geometry approach with emphasis on the background leading to this development, hybrid geometry features in MCNP, some details on unstructured mesh modeling, details on organizing the mesh data, and a discussion of tracking issues on the unstructured mesh. Section 3 presents results from the simple Godiva criticality benchmark problem. Section 4 contains conclusions from this work. Section 5 discusses future work.

## 2. THE HYBRID GEOMETRY APPROACH

### 2.1. Background

Computer-Aided Design and Computer Aided Engineering tools are used by engineers, designers, and analysts in many ways depending on the profession of the user and the type of software in question. The mechanical engineering community has invested heavily in coupling finite element analysis methods with these tools. For just the solid modeling aspects of these tools, much effort and money has been expended to ensure robust and easy to use products. Often, along with these products, advanced visualization of results capabilities exist. It is these latter two aspects that are attractive to and of primary importance to Monte Carlo users who need to construct complex geometries for their work.

Generally, CAD/CAE models start with the construction of individual parts, usually of a unique material, that are combined to form an assembly. A part may be used only once or repeated many times in different locations. In the construction of the assembly, gaps and overlaps of parts may exist. Current CAD/CAE systems use some type of boundary representation format to define the three-dimensional boundaries of the parts and assembly. This boundary representation is incompatible with the CSG requirements (discussed in the previous section) of a code like MCNP and other Monte Carlo particle transport codes.

As witnessed by the many papers that have been published in recent years, the attractiveness of the CAD/CAE tools in creating complex 3-D models has been so great that innovators have accepted the expense of building either CAD-to-MCNP convertors or developing approaches to track directly on CAD geometries. While these approaches have been successful in fulfilling a need and have helped to advance the state-of-the-art in particle transport code capability, they can only be considered band aids to the real problem – the inability of legacy geometry capabilities (that pre-date modern CAD/CAE tools) in Monte Carlo particle transport codes to meet the needs of 21$^{st}$ century particle simulation requirements, particularly when multi-physics analysis is needed. With a program like MCNP that has upwards of 230,000 lines of code, it would take man-years of effort to re-engineer and re-write completely with what some might consider a modern geometry capability that is tightly integrated with all code features. That is, if agreement could be found on what truly constitutes a modern capability.

It is unlikely that any one approach to representing geometry in a code like MCNP would fit the needs of everyone. Much effort was expended in developing, optimizing, and verifying the current CSG approach. When the geometry is fairly simple and comprised of homogeneous regions that can be easily represented with first and second degree analytic surfaces, this is the approach to use. For these situations with a reasonable number of cells, computer runtimes are good. At this time, it would be foolhardy to abandon this tried and true capability and force users to only one method that accommodates easy creation of complex, heterogeneous models at the expense of increased computational cost.

The solution to this quandary is a hybrid geometry capability for MCNP that lets a modern geometry capability co-exist with the legacy capability. Since most robust CAE tools like Abaqus/CAE that are integrated with finite element methods can generate an unstructured mesh representation of the solid model, we have chosen to implement a hybrid geometry capability in MCNP that permits the existence of the mesh as a mesh universe within the legacy CSG. Thus, each geometry type can be used where it is advantageous to do so.

## 2.2. Hybrid Geometry Features

As stated in the previous sub-section, the unstructured mesh is treated as a mesh universe using the existing CSG universe capabilities. The mesh universe must be among the lowest level universes (if there is more than one), meaning that no other universe can be embedded within it. Treating the mesh as a universe permits the usual MCNP CSG universe operations of rotation and translation. Currently, MCNP is restricted to reading only one mesh universe, but we expect to relax this limitation to permit multiple mesh descriptions for assemblies within the available computer memory.

Outside of the mesh universe, the CSG can be constructed in the legacy manner to the detail that is necessary to describe the problem. MCNP CSG lattices are compatible with the unstructured mesh feature provided the mesh cell resides entirely outside the lattice.

Currently, the mesh description must appear in the form of an Abaqus ascii input file. This input is separate from the MCNP input file and its existence is communicated to MCNP through new data-section input.

MCNP has many cell-based features, such as tallies and variance reduction techniques, which are desirable to use with the mesh. This is handled by tagging collections of mesh elements to create pseudo-cells and mapping these collections to special cell cards; more detail is provided in the next sub-section. This approach takes advantage of existing, proven capabilities that do not need to be re-engineered to work on the mesh.

Since multi-physics analyses are becoming more desirable, the MCNP unstructured mesh treatment was designed with this in mind. For example, a user may wish to calculate neutron energy deposition in a reactor component and then export this information back to a structural analysis program where finite element methods are used on an unstructured mesh for further analysis. To fulfill this requirement, it is necessary to calculate path length estimates of the flux, etc. for each mesh element through which the particles track. This dictates an element-to-element tracking approach that differs from the cell surface-to-surface tracking of the CSG geometry (either through a legacy model description or one created from a CAD-to-MCNP converter) or the surface-to-surface tracking of the direct-CAD tracking. In order to distinguish results collected on the mesh from the legacy tallies, the former are referred to as *edits*. Along with a simplified mesh geometry description, these edits are written to a special output file that can be used both for post-processing visualization and input to finite element analysis codes. It is not intended for the edit capability to re-engineer all of the functionality of the legacy tallies. For example, there are no statistical uncertainties calculated at the mesh element level. However, tally functionality can be used with collections of mesh elements through the pseudo-cell capability discussed previously. Providing full tally functionality at the mesh element level would prove to be more computationally expensive and might be unnecessary for many applications.

## 2.3. Unstructured Mesh Details

An unstructured mesh representation of a part can be created by tessellating it with simple shapes such as tetrahedra, pentahedra, or hexahedra to create corresponding mesh elements. Generally, a part contains only one mesh type, except when pentahedrons are mixed with hexahedrons due to difficulties in the geometry that the meshing algorithm has trouble resolving. These elements may be of first order where there are nodes only at the element vertices or of second order where there are nodes at the element vertices and at the mid-points of the edges connecting the vertices. When there are more than three nodes on a face, the nodes may be positioned such that there is a degree of curvature with that face. The 4-noded faces of the first order elements may be either planar or bi-linear. The 6-or-more-noded faces of the second order elements may be planar, bi-linear, or quadratic. Theoretically, less of the higher order elements are needed to accurately represent a curved surface. These six element types are available in MCNP; in fact, all six types may be used in the same model.

The modeling paradigm of creating an assembly of parts in a CAD/CAE tool can lead to overlapping of parts in some locations and gaps in others, regardless if it is the boundary representation of the solids or the equivalent mesh representation. One way for the user to avoid this is to create the model out of one part where the part has been sectioned into the appropriate material regions; often this is not practical while other times it may be the best choice.

Like most Monte Carlo transport codes, MCNP requires that all space be uniquely defined in CSG space. For those CAD-to-MCNP conversion routines, these gaps must be converted to cells. For the direct-CAD tracking approach, the gaps can be computed explicitly using Boolean operations commonly found in CAD systems or can be treated implicitly [2]. That is, they are implicitly defined from the explicit definition of the existing parts. The approach taken in MCNP for dealing with gaps in the unstructured mesh is to treat them implicitly.

Similarly, overlap regions must be reconciled. In MCNP, we have decided to take a very simple and straightforward approach: unless encountering a gap, a particle track stays in a region or element until it can definitely determine a new region or element in which to transport. On models tested to date, this approach has worked very well.

It should be noted that when the solid model is created, the user has control over the fidelity of the gaps and overlap regions; often, substantial effort may be required on the user's behalf to minimize the gaps and overlaps. We feel that this is the correct approach since MCNP should not be changing any model in a way that is opaque to the user. We also want MCNP to be robust enough to track in an expected manner on any reasonable mesh that may be less than perfect; however, the amount of deviation from perfect that can be tolerated for the unstructured mesh tracking has not been quantified.

## 2.4. Mesh Data Essentials

Information about the mesh must be organized in specific ways in order to support various functions. This sub-section discusses the data organization requirements.

### 2.4.1. Element sets

A collection of mesh elements is referred to as an element set (elset). A tool such as Abaqus/CAE allows creation of elsets. Creation of material elsets is essential in assigning material properties to mesh elements. For each material section of a part, a material elset must be created so that all elements are associated with some material.

If volume type tallies (cell-averaged flux, energy deposition, or fission energy deposition) are to be performed over a collection of mesh elements, these elements must be grouped into a statistic elset. The word statistic is chosen to denote that the full statistical treatment of the tally methodology applies to the set. However, all elements belonging to a statistic elset must also belong to the same material set. Thinking of a single part where there is one material set defined for that part, a statistic elset may be an exact duplicate of the material elset or it may comprise a subset of the material elset. A default statistic elset is created for any part where elements have not been assigned to one. It is this grouping of a material-statistic elset that is known as a *pseudo-cell* and must be mapped to the special MCNP cell cards. A table appears in the MCNP output file that describes all pseudo-cells that are explicitly or implicitly defined.

### 2.4.2. Surfaces

As with cell surfaces in the CSG capability, surfaces for the pseudo-cells and the elements that contribute one or more faces to a surface are important to aid in tracking. Somehow the pseudo-cell surfaces must be identified. While Abaqus/CAE does permit the user to tag surfaces, we have developed a routine to perform this function internally, thus alleviating additional user input. Details on how surface information is used in the tracking routine are discussed below.

## 2.5. Tracking Hierarchy

Two important pieces of information are needed for tracking on a mesh: 1) what element contains the particle? and 2) what surface will the particle intersect next? This sub-section discusses how these are handled.

### 2.5.1. Finding the element

If the current source or collision location in the mesh is known, but the element corresponding to that location is not, the element number must be found. MCNP takes the part and assembly information provided by the Abaqus input file and constructs a global mesh model where elements are numbered consecutively from one to the maximum number. The global elements with their corresponding axis-aligned bounding boxes are stored in a spatial kd-tree or skd-tree [6]. The skd-tree is a highly efficient data structure that can be searched in O(log n) time. Whenever a location must be associated with an element, the global skd-tree is searched for the approximate location and then verified that the location does indeed reside within a potentially trilinear or triquadratic element.

During the global model construction, all elements that belong to a cell-surface of a pseudo-cell are stored in a skd-tree; one for each pseudo-cell. When entry into a specific pseudo-cell is in question, these cell-surface skd-trees are searched in a manner similar to that for finding a location in the global skd-tree, but instead of looking for containment, a face intersection is sought.

### 2.5.2. Element-to-element tracking

Since path-length estimates of quantities are needed for each element, particles must track from element to element. Conceptually, this is tracking at a finer granularity than what is performed with the CSG surface to surface tracking. Given a mesh location and corresponding element number, if the location is not located on one of the element's faces, the code determines which face it intersects and the location of the intersection. Once the intersection location is known on the element's face, the code determines if there is a neighboring element sharing that face. If the element's nearest neighbor list exists it searches that list for a candidate. If the nearest neighbor list does not exist, the code creates the list on the fly and then checks it.

If the particle tracks to a surface element so that the current position is on the surface and the nearest neighbor list search is futile, the code searches all cell-surface skd-trees for a potential intersection. If this search is unsuccessful, the particle is leaving the mesh and normal CSG tracking resumes with the particle in the mesh cell's background region. During the searching, checks are performed to deal with gaps.

If the particle is tracking from a CSG cell into the mesh cell, the cell-surface skd-trees are searched to determine at what location and corresponding element number the intersection takes place. Normal CSG cell tracking takes place until the particle enters the mesh.

## 3. A GODIVA CRITICALITY BENCHMARK

As witnessed by recent publications dealing with CAD to Monte Carlo geometry conversion, most of the test problems presented there are of the shielding type. In testing their tetrahedral mesh capability in MONK/MCBEND, Bird [7] noted the difficulty in meshing with first order tetrahedra to accurately reproduce curved surfaces and hence volumes and masses for some components of interest; this impacted their results. Since obtaining accurate eigenvalues in criticality calculations is highly dependent on accurate volumes and masses, we wanted to test MCNP's new unstructured mesh capability in this regard. We chose to start with the simple Godiva criticality benchmark [8] since it is well understood and poses a challenge with its curved surface. This benchmark is a simple highly-enriched uranium sphere of radius 8.7407 cm.

All of the computer runs for these models were performed under the following set of conditions. ENDF/B-VII cross sections were used for all nuclides. Kcode calculations were performed and photons were followed in addition to the neutrons. Each calculation has 100 inactive cycles followed by 900 active cycles with 3000 histories per cycle. Calculations were performed with the sequential version of the code using an Intel Xeon X5450 clocked at 3.0 GHz. The operating system was 64-bit RedHat Enterprise Linux 5.

The cases considered for the simple sphere model are described in the tables. Both first and second order hexahedrons along with first and second order tetrahedrons were used. The mesh seed parameter presented in the table is an Abaqus/CAE parameter used to set the element size. Abaqus attempts to make elements with edges of this length; no attempt was made to force edges to this length. The number of total elements in the model is dictated by the seed number. Also present is a case where the combinatorial geometry was used and the actual experimental results, when known. Representations of the hexahedron and tetrahedron meshes are shown in Figure 1; one-quarter of the geometry is removed to aid visualization.

**Figure 1. Simple Godiva sphere meshed with tetrahedrons (left) and hexahedrons (right) using seed values of 1.75 cm.**

Figure 2 shows total energy deposition results for the 1.75 cm seed case where first order tetrahedrons are used. The energy deposition behavior is virtually identical to that of the total neutron flux. The component at the left of the figure displays the element average results. The component at the right of the figure shows contour results that were created from the element average results after applying Abaqus/CAE's built in smoothing function.

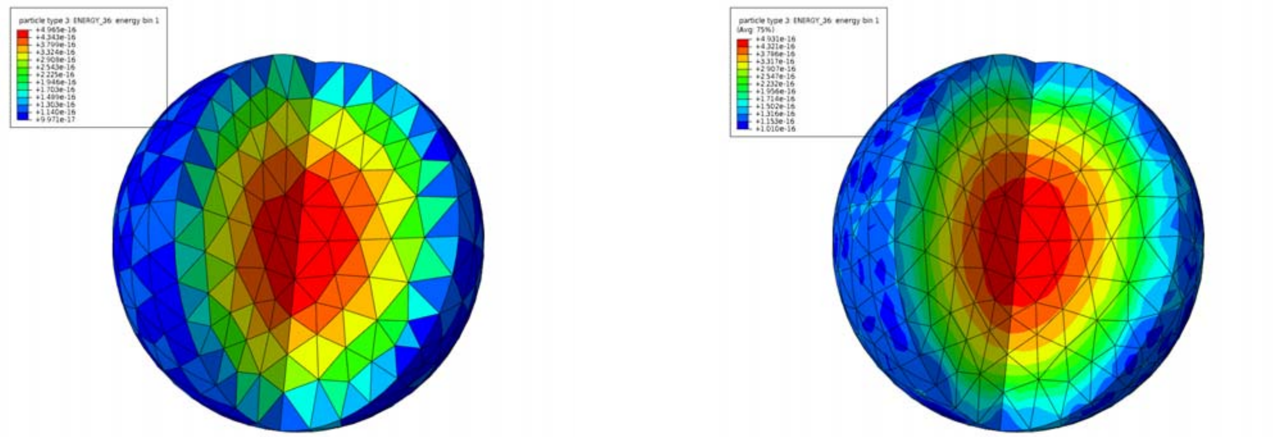

**Figure 2. Energy deposition results for the simple Godiva sphere meshed with tetrahedrons and a seed of 1.75 cm: Element averaged results (left). Contoured results (right).**

From Table I it can be seen that the smaller sized elements do a better job in terms of reproducing the actual volume. As expected, the second order elements do a better job than the first order elements in predicting volumes when compared to the combinatorial calculation. This is due to the ability of the second order elements to possess a higher degree of curvature.

**Table I. Some physical results for the simple Godiva sphere**

| Element Type | Mesh Seed (cm) | Number of Elements | Volume (% change from CSG or cm^3) | k-effective | k-effective (% change from CSG) | Total Flux (% change from CSG or n/cm^2) |
|---|---|---|---|---|---|---|
| 1st Order Hex | 0.875 | 7168 | -0.42 | 0.99850 | -0.14 | 0.28 |
| | 1.75 | 896 | -1.71 | 0.99524 | -0.46 | 1.24 |
| | 3.50 | 160 | -6.77 | 0.98018 | -1.97 | 5.11 |
| | 5.00 | 56 | -17.0 | 0.94660 | -5.33 | 14.0 |
| | 6.00 | 32 | -23.6 | 0.92461 | -7.53 | 20.9 |
| 1st Order Tet | 0.875 | 35453 | -0.46 | 0.99883 | -0.10 | 0.35 |
| | 1.75 | 4973 | -1.65 | 0.99565 | -0.42 | 1.22 |
| | 3.50 | 459 | -6.63 | 0.98060 | -1.93 | 5.00 |
| | 5.00 | 193 | -14.2 | 0.95702 | -4.28 | 11.6 |
| | 6.00 | 32 | -29.8 | 0.90002 | -9.99 | 28.0 |
| 2nd Order Hex | 0.875 | 7168 | 0.00* | 0.99937 | -0.05 | -0.07 |
| | 1.75 | 896 | -0.01 | 1.00006 | 0.02 | 0.02 |
| | 3.50 | 160 | -0.10 | 0.99964 | -0.02 | 0.05 |
| | 5.00 | 56 | -0.69 | 0.99768 | -0.22 | 0.45 |
| | 6.00 | 32 | -1.48 | 0.99393 | -0.59 | 0.89 |
| 2nd Order Tet | 0.875 | 35453 | 0.00* | 0.99975 | -0.01 | -0.04 |
| | 1.75 | 4847 | 0.00* | 0.99984 | 0.00 | 0.00 |
| | 3.50 | 459 | -0.05 | 0.99844 | -0.14 | -0.12 |
| | 5.00 | 193 | -0.27 | 0.99509 | -0.48 | -0.23 |
| | 5.50 | 64 | -1.49 | 0.97516 | -2.47 | -1.02 |
| | 6.00 | 32 | -1.50 | 0.97681 | -2.31 | -0.84 |
| CSG | n/a | n/a | 2.79722E+3 | 0.99986 | 0.00 | 2.42274E-3 |
| Actual | n/a | n/a | 2.79722E+3 | 1.000 +/- 0.001 | -- | -- |

* Value due to rounding a small number.

The three columns at the right of Table I and Figures 3 and 4, compare some physical results from the calculations. All calculated results have relative uncertainties less than 0.04%; the uncertainty in the actual bare Godiva sphere was 0.1%. Agreement is generally quite good when the element size is small, correlating with how well the volume and mass are reproduced. As expected, the second order elements do a better job (by an order of magnitude) than the first order elements in predicting results when compared to the combinatorial calculation. This is due to the ability of the second order elements to possess curvature. Note: even as the size of the second order elements become larger so that a smaller number of elements are used, the ability to accurately reproduce the volume and mass decreases with a corresponding effect in the Table I results.

Table II presents computer run times for these calculations as well as estimated memory requirements needed by the unstructured mesh. The last row in the table provides the run time for the CSG run and there are no mesh memory needs for it. Obviously, it takes longer to run the kcode calculation on an unstructured mesh compared to the legacy geometry in MCNP and the time increases with the number of mesh elements. When the same number of second order tetrahedrons and hexahedrons represent the volume equally well, the calculation with the second order tetrahedrons will run faster since the interpolation function and associated tracking routines for the second order tetrahedrons are not as complex as that for the second order hexahedrons. While a similar relationship holds for the first order interpolation functions, the difference is not as severe since the associated interpolation functions and tracking routines are much simpler for first order compared to second order. If the second order hexahedrons represent the volume more accurately than the second order tetrahedrons, given a fixed seed value, the calculations with the second order hexahedrons will outperform the second order tetrahedrons.

It can also be seen in Figure 5 that the convergence of k-effective is faster for second order elements than for first order elements for the same mesh seed. Memory requirements per mesh element are modest at between roughly 500 to 1100 bytes. The memory requirement per element decreases slightly as the number of elements increase due to amortization of basic mesh infrastructure costs over all the elements.

**Table II. Computer statistics for the simple Godiva sphere**

| Element Type | Number of Elements | Runtime (min)* | Mesh Memory (kb) | Mesh Memory Per Element (bytes) |
|---|---|---|---|---|
| 1st Order Hex | 7168 | 329 | 4476 | 624 |
| | 896 | 62 | 583 | 650 |
| | 160 | 24 | 108 | 678 |
| | 56 | 18 | 40 | 707 |
| | 32 | 15 | 23 | 729 |
| 1st Order Tet | 35453 | 1394 | 18334 | 517 |
| | 4973 | 209 | 2547 | 512 |
| | 459 | 37 | 254 | 552 |
| | 193 | 22 | 108 | 559 |
| | 32 | 12 | 21 | 655 |
| 2nd Order Hex | 7168 | 349 | 6964 | 972 |
| | 896 | 80 | 900 | 1005 |
| | 160 | 41 | 166 | 1039 |
| | 56 | 38 | 60 | 1075 |
| | 32 | 36 | 35 | 1101 |
| 2nd Order Tet | 35453 | 1397 | 17310 | 488 |
| | 4847 | 218 | 2416 | 499 |
| | 459 | 47 | 244 | 532 |
| | 193 | 34 | 104 | 539 |
| | 64 | 27 | 36 | 566 |
| | 32 | 23 | 21 | 653 |
| CSG | n/a | 3.4 | n/a | n/a |

* Runtimes are for un-optimized mesh tracking algorithm and coding.

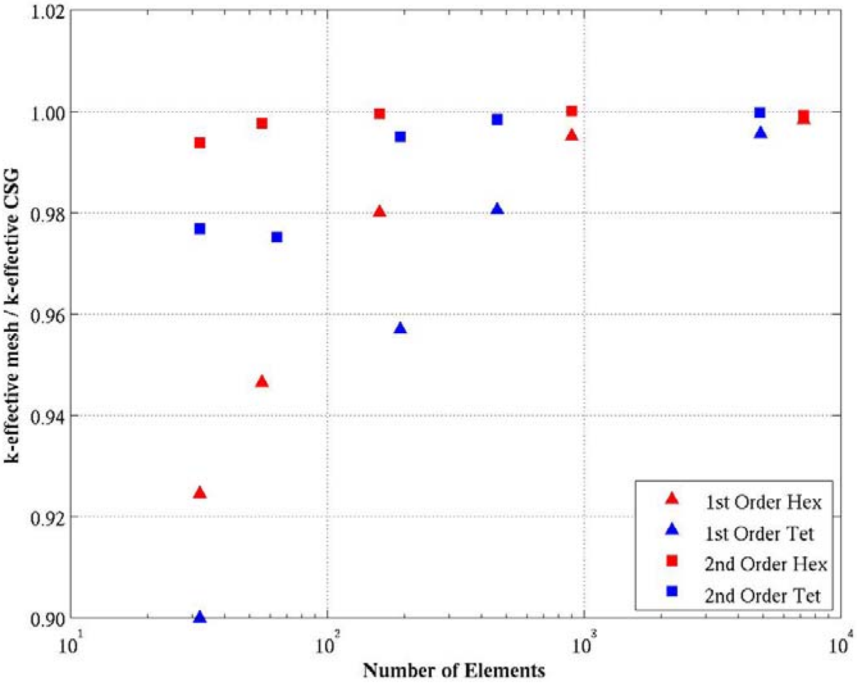

**Figure 3. Ratio of the Eigenvalue from the simple Godiva sphere for various mesh types and number of mesh elements to the eigenvalue from CSG geometry.**

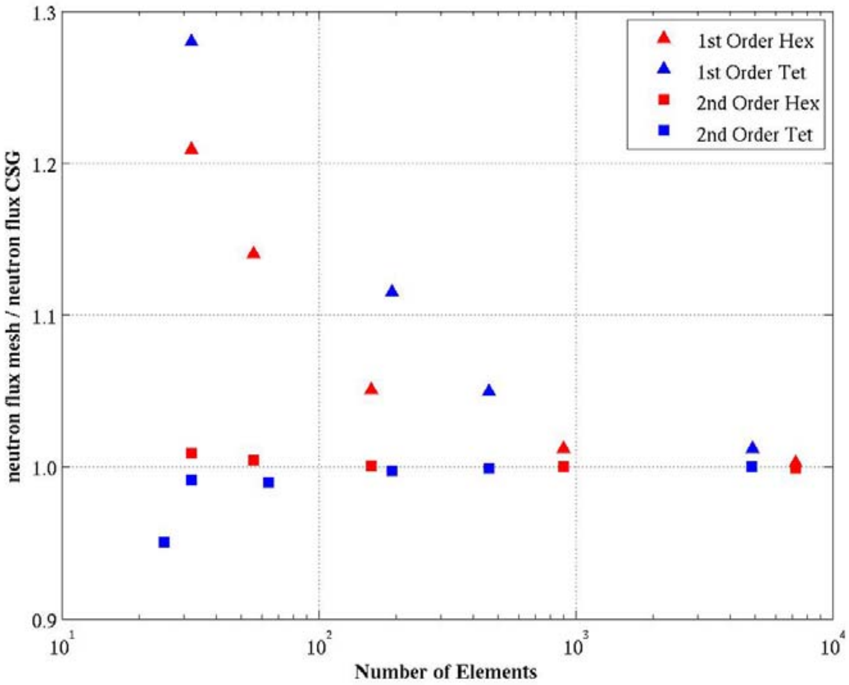

**Figure 4. Ratio of the total neutron flux from the simple Godiva sphere for various mesh types and number of mesh elements to the total neutron flux from the CSG geometry.**

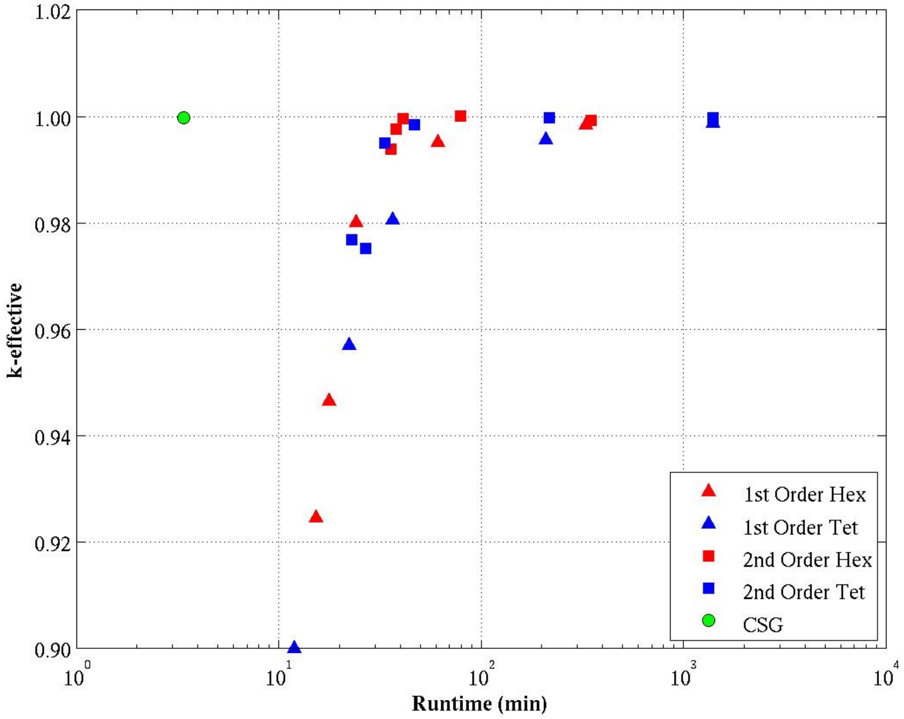

**Figure 5. K-effective convergence relative to runtime for various mesh types. (Runtime for un-optimized mesh tracking algorithms and coding.)**

## 4. CONCLUSIONS

The unstructured mesh tracking library for MCNP has been discussed in this paper. It is now possible to create a highly three-dimensional, heterogeneous geometry using Abaqus/CAE solid modeling tools, create an unstructured mesh representation of that geometry, import the mesh into MCNP where neutron and photons may be tracked in a hybrid arrangement, and export transport results such as flux and energy deposition back to Abaqus/CAE for visualization and multi-physics analysis.

We have shown that this new capability works quite well for criticality calculations where accurate volumes and masses are needed. However, appropriate refinement of the mesh is needed to achieve good answers; this results in increased, but not necessarily prohibitive, computational costs.

Our implementation accommodates both first and second order polyhedral element types. The second order polyhedrons function better than first order ones in accurately reproducing volumes and masses due to their ability to possess curved faces. Although no evidence was presented in this paper, it is expected that first order elements will out perform second order elements when objects they are representing possess essentially flat surfaces. The cost in terms of computer run time for using second order polyhedrons was comparable to that of using first order polyhedrons on the Godiva criticality benchmarks we tested, but yielded roughly an order of magnitude increase in accuracy.

## 5. FUTURE WORK

The development of the unstructured mesh library for Monte Carlo particle tracking in MCNP is ongoing. In the future, we hope to optimize the tracking algorithms and the associated coding. We also intend to complete a surface to surface tracking implementation where detailed element to element tracking and results are not required; this feature will be pseudo-cell selectable. We plan to continue verification and validation efforts for this feature by studying other benchmark problems.

## REFERENCES

1. X-5 MONTE CARLO TEAM, "MCNP – A General Monte Carlo N-Particle Transport Code, Version 5, Volume I: Overview and Theory," LA-UR-03-1987, Los Alamos National Laboratory (April 2003).
2. Timothy J. Tautges, Paul P. H. Wilson, Jason A. Kraftcheck, Brandon M. Smith, Douglass L. Henderson, "Acceleration Techniques For Direct Use Of CAD-Based Geometries In Monte Carlo Radiation Transport," *International Conference On Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3-7, 2009, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009).
3. Yican Wu, Qin Zeng, and Lei Lu, "CAD-Based Modeling For 3D Particle Transport," *International Conference On Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3-7, 2009, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009).
4. D. Groβe and H. Tsige-Tamirat, "Current Status of the CAD Interface Program McCad for MC Particle Transport Calculations," *International Conference On Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3-7, 2009, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009).
5. Dessault Systemes Simulia, Inc., "ABAQUS USER MANUALS, Version 6.9," Providence, RI (2009).
6. Beng Chin Ooi, *Efficient Query Processing in Geographic Information Systems*, Springer-Verlag, Berlin, Germany (1987) in the *Lecture Notes in Computer Science* series, G. Goos and J Hartmanis, editors.
7. Adam Bird, Julie Martin, Edmund Shuttleworth, and Andrew Cooper, "Testing of the Tetrahedral Mesh Import Capability for Importing Converted CAD Files for Shielding and Criticality Calculations with MONK and MCBEND*," International Conference on Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3-7, 2009, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009).
8. *International Handbook of Evaluated Criticality Safety Benchmark Experiments*, NEA/NSC/DOC(95) 03 Vol II, September 2009 Edition.