LA-UR-12-00795

Approved for public release;
distribution is unlimited.

Title:	Unstructured Mesh User's Startup Guide
Author(s):	Roger L. Martz
Intended for:	LANL MCNP Classes

MCNP6 Hybrid Geometry

Unstructured Mesh User's Startup Guide

Roger L. Martz

February 2012

Hybrid Geometry

Abstract

The presentation is intended to introduce MCNP users to the unstructured mesh capability by discussing the basic concepts needed to construct an unstructured mesh geometry. MCNP input cards specific to this feature are shown along with examples. The utility programs are presented for pre-processing input and post-processing output. Limitations of the current implementation are provided along with some performance results from two simple benchmark problems.

Hybrid Geometry

What is the MCNP Unstructured Mesh (UM) Capability?

It is part of the new hybrid geometry capability that lets MCNP users embed "other" geometry representations of their geometry (e.g., unstructured and structured mesh) in the "legacy" constructive solid geometry (CSG) so that all representations of the geometry work together seamlessly.

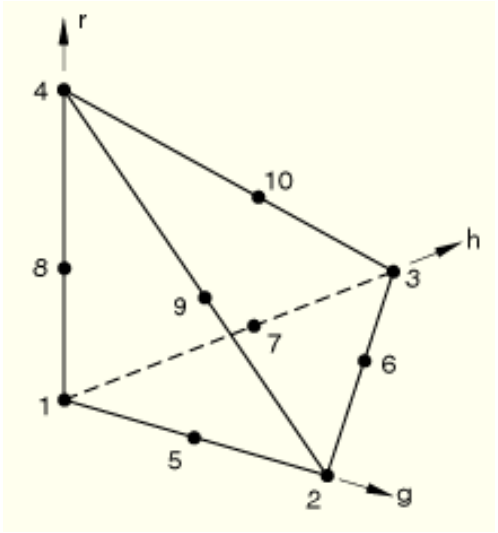
What is an UM geometry?

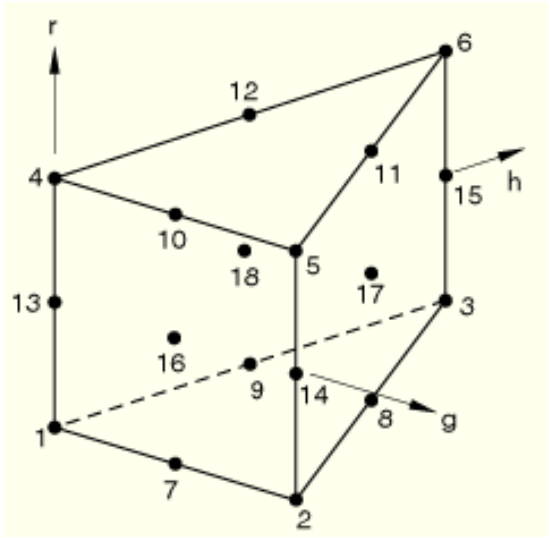
1. **A computer aided engineering (CAE) tool such as Abaqus/CAE or CUBIT can be used to create a 3-D solid model of the entity of interest.**
2. **An UM representation of that solid model can be created using one or all of the 6 supported finite element types:**
   - 1st order tetrahedra, pentahedra, hexahedra
   - 2nd order tetrahedra, pentahedra, hexadedra (not supported in CUBIT)
3. **The UM representation can coexist with other geometry types in MCNP in an hybrid arrangement.**

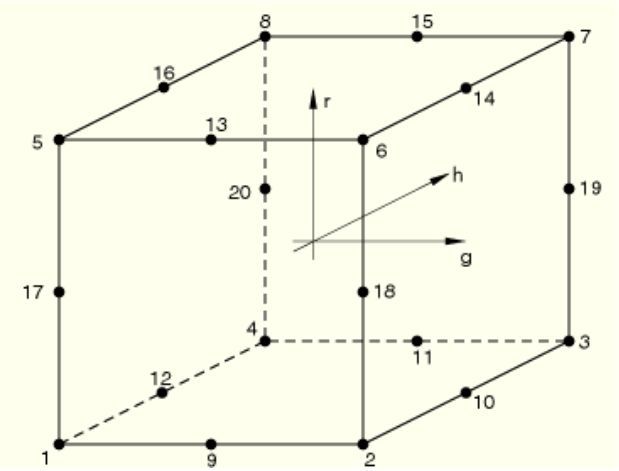
Objects and Definitions: elements or finite elements

The smallest building blocks

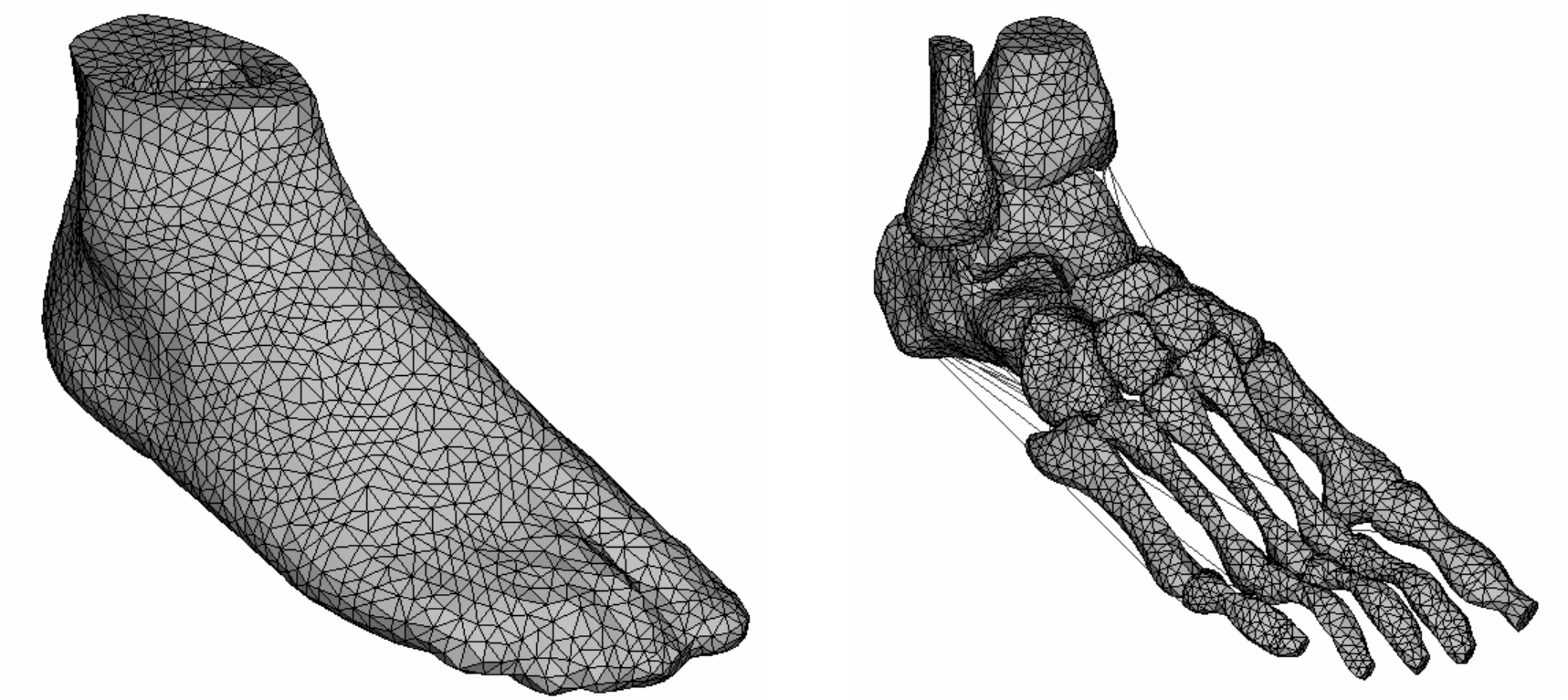
- **Unstructured polyhedrons with 4-, 5-, and 6-sides or faces generated by a program like the ABAQUS®/CAE. Surfaces may be bilinear or quadratic depending on the number of nodes.**







Example



Jason Tak-Man Cheung and Ming Zhang, "Finite Element Modeling of the Human Foot and Footwear," 2006 ABAQUS Users' Conference.

CAD vs. CAE: A Simple View

CAD – Computer Aided Design

CAE – Computer Aided Engineering

- **Both utilized solid modeling engines.**
- **Some CAD tools may generate a mesh.**
- **CAE tools generate a mesh since they are integrated with finite element methods that generally support thermo-mechanical design, analysis, and simulation.**
  - CAE is CAD on steroids

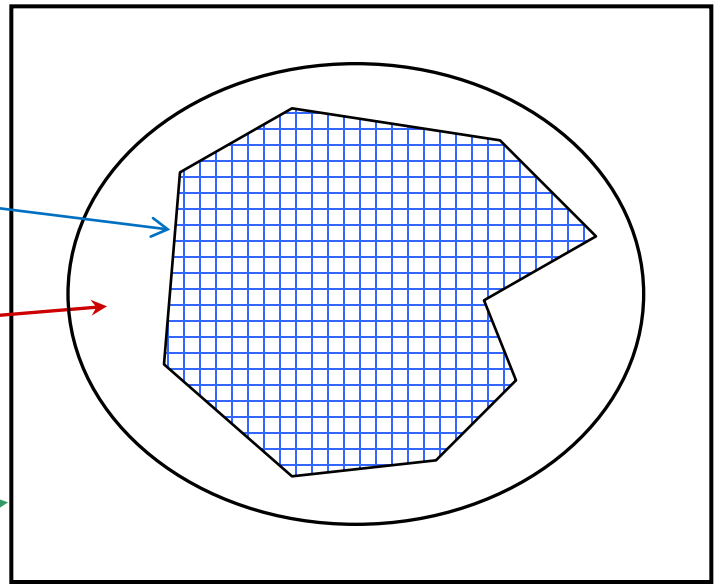
A Simple Hybrid Arrangement

Mesh universe with elliptical background cell and mesh

Unstructured mesh (imp=1)

Background cell (csg filling cell; imp=1)

CSG cell (imp=0)



Why use UM?

1) **Easier to create complex 3-D models with a state-of-the-art CAE tool.**
2) **Better geometry and results visualization.**
3) **Easier multi-physics integration with other mesh-based physics codes.**
4) **Better calculational performance for certain problems.**
5) **It's cool!**

Limitations

The unstructured mesh feature is still undergoing development and has not been fully integrated with all of MCNP's existing features. For example,

- **limited to only neutrons and photons**
- **unstructured mesh can not be placed inside a lattice**
- **a universe can not be placed within a mesh universe**
- **the MCNP plotter will not plot the unstructured mesh, but should plot the background cell**
- **DXTRAN, point detectors, and forced collisions are known not to work**
- **surface source reads and writes are not guaranteed to work with the unstructured mesh**
- **CSG surfaces must not clip or intersect the unstructured mesh**

See the complete list of limitations and restrictions later in these slides.

Paradigm Shift

- **Must recognize that this capability "marries" different "technologies".**
- **Each of these "technologies" was developed independently giving rise to terminology that one could consider unsettling because it is either new, different, conflicting, strange, weird, etc.**
  - Example: MCNP uses the concept of "cell" as the basic CSG building block. Abaqus uses the concept of "cell" when it refers to a 3-D region; a part may be a single cell or segmented into multiple cells.
- **One of the key concepts that was created to help the UM integrate with the MCNP world is the pseudo-cell or inferred-cell; these are defined later. In a nut shell, this is a mechanism to let a collection of mesh elements have MCNP cell-like properties (e.g., imp's) so that existing features are readily useable with the mesh.**

Objects and Definitions

Objects and Definitions

- **The following slides will attempt to define some terminology as it relates to objects associated with the unstructured mesh capability.**
- **NOTE: Some of the terminology may seem "different" from what you are accustomed. This arises due to the way the CAE / finite element users in the structural mechanics world have defined things. I have chosen to use their definitions where appropriate.**

Objects and Definitions: mesh

The collection of elements comprising the entire (meshed) model.

Objects and Definitions: edits

- **When element-to-element tracking on the unstructured mesh is requested (this is the default), results can be accumulated on each finite element through which particles pass.**
- **Results on the unstructured mesh are referred to as "edits" or "elemental edits" to distinguish them from the tally results in the CSG.**
  - Results go to a special output file
- **Edits are not intended to reproduce all of the tally functionality, such as**
  - Tally types F1, F2, F5
  - Statistical analysis, statistical checks, TFC's, empirical history score pdf, etc. (If these features are needed, use the appropriate statistical elset so that these things can be used with a collection of elements. See the discussion on statistical elsets below.)

Objects and Definitions: elsets

- **A collection of elements or a sub-set of the mesh associated with a specific tag, label, or name.**

Objects and Definitions: part

- **The smallest geometric object created in the Abaqus CAE/tool.**
  - Can contain a mesh representation
  - Can be assigned attributes such as material
  - Can be segmented into smaller pieces; these pieces are referred to as cells; if the part is not segmented then it is also a cell

Objects and Definitions: instance

- A copy of a part used when constructing an assembly

Objects and Definitions: assembly

- **The largest geometric object created in the Abaqus CAE/tool**
  - Constructed from instances of parts
  - A composite object

Objects and Definitions: pseudo-cell or inferred-cell

- An elset that has been mapped to an MCNP cell.
- This is a mechanism to let a collection of mesh elements have MCNP cell-like properties (e.g., imp's) so that existing features are readily useable with the mesh.
- In essence, a group of elements (elset) has cell-like properties, but is not a traditional MCNP cell.

Pseudo: in scientific use, denoting close or deceptive resemblance to

Objects and Definitions: background cell

- A cell that serves as the background (or container or holding) cell for the mesh.
- An MCNP cell into which the mesh has been placed and basically is the csg cell that is "filled" into the universe.

Objects and Definitions: mesh universe

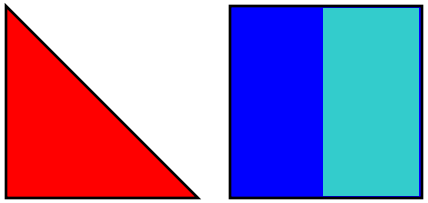
- **An MCNP universe comprised of the mesh and the background cell.**
  - May not contain any other lower universes

Constructing A Mesh Geometry

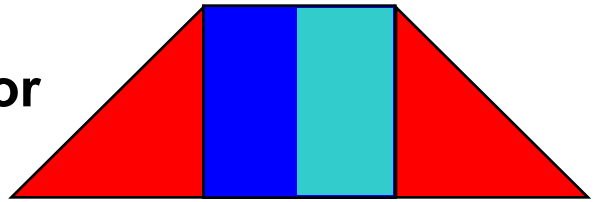
Constructing A Mesh Geometry: Parts & Assembly

Created with ABAQUS

- **The final model is an "assembly" constructed with "instances" of "parts"**
- **Each "part" can consist of**
  - a single segment of one homogeneous material
  - multiple segments of different homogeneous materials
- **Each "part" can be meshed independently**
- **The mesh representation in MCNP is for the assembly (i.e., the global model)**



parts



assembly

Constructing A Mesh Geometry: elsets

Created with ABAQUS

- **Elements must be "tagged" with information by creating element sets (elsets).**
- **Each part must contain two elsets of data for:**
  - material sets
  - statistic or tally sets
- **Any part may contain a volume source either in its entirety or in part. If so, there must be an additional elset(s):**
  - source set(s)

Constructing A Mesh Geometry: Material Names

Created with ABAQUS

- Material names must be defined so that elset's material numbers can be matched with a meaningful name in the MCNP output file.
- Currently, only 1 name is required and a warning is printed in the output if there are more material names expected but missing.

Constructing A Mesh Geometry: Naming Requirements

Elsets and Material Names

Name format: ????AAAA????%ZZZ

Where AAAA is one of the keywords:

material

statistic or tally

source

Where,

% can be either the underscore character, _ , or the hyphen, - .

ZZZ is the set number and must be at the end of the name.

???? Are any other character or groups of characters.

Constructing A Mesh Geometry: Naming Requirements

Multi-Function Elsets

- It is possible to construct one elset that has multiple functions.
- The elset number, ZZZ, must be an appropriate number for each function. (See the following discussion.) Sets will be specified for each keyword with that number.

Name format: ???AAAA%BBBB%CCCC???%ZZZ

Where AAAA, BBBB, and CCCC are one of the keywords and there should be at least AAAA :

material

statistic or tally

source

Constructing A Mesh Geometry: Materials

- **Material numbers should be unique in the UM model.**

  If part #1 is entirely lead and its material number is 1, part #2 should not use material number 1 if that part is uranium!

  If numbers aren't unique, the mesh output files (e.g., GMV file) & some utility program functions may not function properly.

- **Initially, material elset numbers (and statistic elset numbers) where to begin at 1 and continue sequentially to the maximum number of different materials (statistic sets) in the entire assembly (not the part).**
- **Initially, the material names were to be numbered in the same manner as the material elset numbers so that numbers could be matched with names in the output.**

Constructing A Mesh Geometry: Materials

- **Currently, material names need not include the material number.**
- **The procedure for matching the material name with the material number in the MCNP output is governed by the following "rules".**
  - 1st the code tries to match the material elset number with a material number in the material name.
  - 2nd the code assumes that the material names are alphabetized in a list and will match the material elset number with the sequential position in the list.
  - If both of these "rules" fail to produce a defined name, a message appears stating that the material name does not exist.

Constructing A Mesh Geometry: Statistic / Tally Sets

- **Statistic or tally elsets are the way to group collections of elements for the purpose of volume based MCNP tallies (i.e., F4, F6, F7 tallies).**
- **Each group of statistic elsets in a part should have a unique number.**
  - With Abaqus it is possible to partition a single part into different segments. Each segment can be a different material. Segments with different materials can not belong to the same statistical set and, hence, become different pseudo-cells. This helps enforce consistency with MCNP's concept of cells.
- **The statistic elset number along with the material elset number will be used internally by the code to establish a unqiue, internal, pseudo-cell number.**

Constructing A Mesh Geometry: Pseudo-Cells

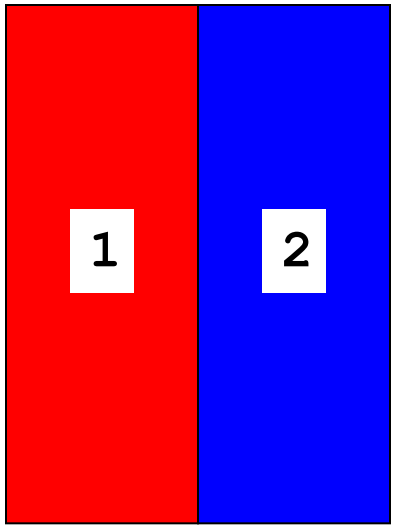
Pseudo-Cells or Inferred Cells

- Each homogeneous region in the mesh is referred to as a pseudo-cell.
- Want to map the mesh pseudo-cell to a legacy MCNP cell for things like cell-based tallies and variance reduction.

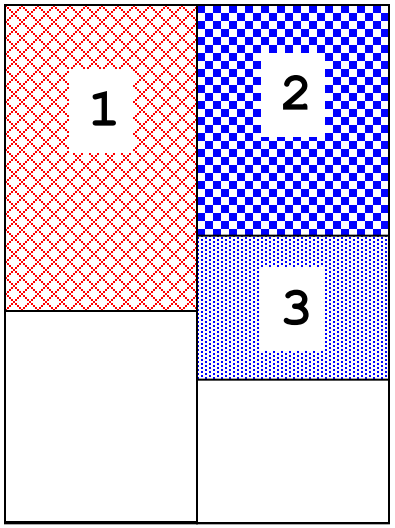
  Pseudo-cell could be called “mesh cell” if this terminology does not conflict with other meanings.
- Pseudo-cells are numbered internally starting at 1. A pseudo-cell cross reference table is printed to the output to aid the user in understanding material, tally, and part assignments between the csg and um worlds.

Constructing A Mesh Geometry

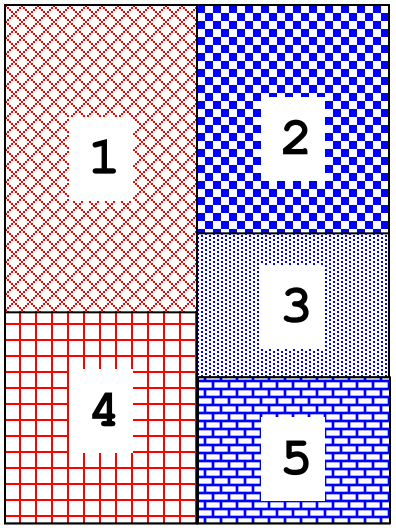
Pseudo-Cell Example



1 part with 2 materials



3 defined & 2 undefined statistical regions



5 pseudo-cells (always consecutively numbered internally from 1)

The code will tag untagged elements.

MCNP Output: Pseudo-Cell Cross Reference Table

Lists for each pseudo-cell that it knows about,

- **The corresponding MCNP cell number**
- **The instance number**
  - This is the same as the pseudo-cell number as long as there aren't any partitioned parts where partitions have different material or statistic elset numbers
- **The part number**
- **The material number**
- **The material name**

```
*****************************************************************************************************************
* Pseudo-Cell Cross Reference Table                                                                             *
*****************************************************************************************************************
  Pseudo-Cell #    MCNP6 Cell #    Instance #    Part #    Material #    Material Name
  -------------    ------------    ----------    ------    ----------    -------------
             1              11             1         2             2    material-cube_02
             2              12             2         3            10    material-source_10
             3              13             3         1             1    material-cube__01
*****************************************************************************************************************
```

MCNP Output: Pseudo-Cell Cross Reference Table

Example with a segmented part

```
*****************************************************************************************************************
* Pseudo-Cell Cross Reference Table                                                                             *
*****************************************************************************************************************
Pseudo-Cell #    MCNP6 Cell #    Instance #    Part #    Material #    Material Name
-------------    ------------    ----------    ------    ----------    -------------
           1              11             1         2             2    material-cube_material_02
           2              12             2         1             1    material-cube_material_01
           3              13             3         3            11    material-source_material_11
           4              14             3         3            12    material-source_material_12
*****************************************************************************************************************
```

MCNP Output: How The Parts Are Instanced

```
*****************************************************************************************************************
*  Building the Global Tracking Model                                                                           *
*****************************************************************************************************************

Adding Instance #     1    :  part-outer_shell-1                          [Part:]   part-outer_shell

       First 1st Order TET  element number:            0      Last  1st Order TET  element number:            0
       First 1st Order PENT element number:            0      Last  1st Order PENT element number:            0
       First 1st Order HEX  element number:            1      Last  1st Order HEX  element number:           56
       First 2nd Order TET  element number:            0      Last  2nd Order TET  element number:            0
       First 2nd Order PENT element number:            0      Last  2nd Order PENT element number:            0
       First 2nd Order HEX  element number:            0      Last  2nd Order HEX  element number:            0

                                                              Last GLOBAL element         :          56
                                                              Last GLOBAL node            :         124

       Translate:      0.000000000000000     0.000000000000000     0.000000000000000
       Rotate    :     0.000000000000000     0.000000000000000     0.000000000000000
                       0.000000000000000     0.000000000000000     0.000000000000000
                       0.000000000000000

Adding Instance #     2    :  part-source-block-1                         [Part:]   part-source-block

       First 1st Order TET  element number:            0      Last  1st Order TET  element number:            0
       First 1st Order PENT element number:            0      Last  1st Order PENT element number:            0
       First 1st Order HEX  element number:           57      Last  1st Order HEX  element number:           64
       First 2nd Order TET  element number:            0      Last  2nd Order TET  element number:            0
       First 2nd Order PENT element number:            0      Last  2nd Order PENT element number:            0
       First 2nd Order HEX  element number:            0      Last  2nd Order HEX  element number:            0

                                                              Last GLOBAL element         :          64
                                                              Last GLOBAL node            :         151

       Translate:      0.000000000000000     0.000000000000000    -0.500000000000000
       Rotate    :     0.000000000000000     0.000000000000000     0.000000000000000
                       0.000000000000000     0.000000000000000     0.000000000000000
                       0.000000000000000
```

MCNP Output: How The Parts Are Instanced

```
Adding Instance #     3    :  part-middle_shell-1                        [Part:]   part-middle_shell

        First 1st Order TET  element number:            0      Last  1st Order TET  element number:            0
        First 1st Order PENT element number:            0      Last  1st Order PENT element number:            0
        First 1st Order HEX  element number:           65      Last  1st Order HEX  element number:          188
        First 2nd Order TET  element number:            0      Last  2nd Order TET  element number:            0
        First 2nd Order PENT element number:            0      Last  2nd Order PENT element number:            0
        First 2nd Order HEX  element number:            0      Last  2nd Order HEX  element number:            0

                                                               Last GLOBAL element         :          188
                                                               Last GLOBAL node            :          367


        Translate:      0.000000000000000     0.000000000000000     0.000000000000000
        Rotate    :     0.000000000000000     0.000000000000000     0.000000000000000
                        0.000000000000000     0.000000000000000     0.000000000000000
                        0.000000000000000
```

Abaqus translate & rotate

```
Global Model Extents
Min X: -4.00000E+00   Max X:  4.00000E+00
Min Y: -4.00000E+00   Max Y:  4.00000E+00
Min Z: -4.00000E+00   Max Z:  4.00000E+00
```

Global Model Extents:

- Min & max for each mesh direction
- Background cell must be large enough to include these dimensions w/o clipping the mesh
- Doesn't reflect MCNP universe translations or rotations

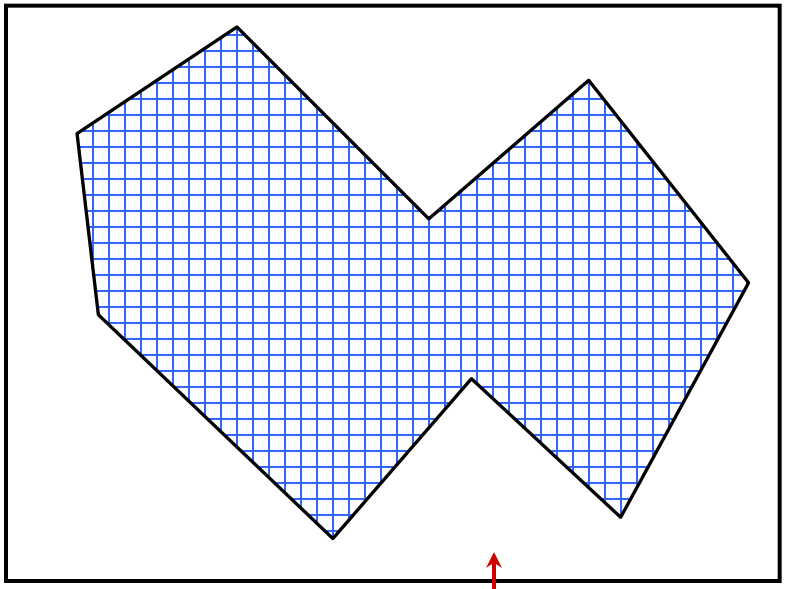
MCNP UM Input Cards

Embedded Mesh Universe

- **Geometry mesh are embedded as the lowest level universe.**
- **Cell card requirements:**
  - One cell card for each mesh “pseudo-cell” or “inferred-cell”.
    - Check the mcnp “outp” file for a table showing how the code expects the “pseudo-cells” to match with the mesh regions. Correct material names in this table are dependent upon the material rules cited above.
  - There must be a cell card that serves as a “background” for the embedded mesh. Think of this as the cell that substitutes or stands-in for the mesh. Another way of stating this is that this cell is the smallest cell into which the mesh can fit. (See next slide for examples.) This can be considered the background material in which the mesh resides and is essential for the unstructured mesh to work correctly. This must also have a “u” descriptor. See the “background” key word on the embed card.

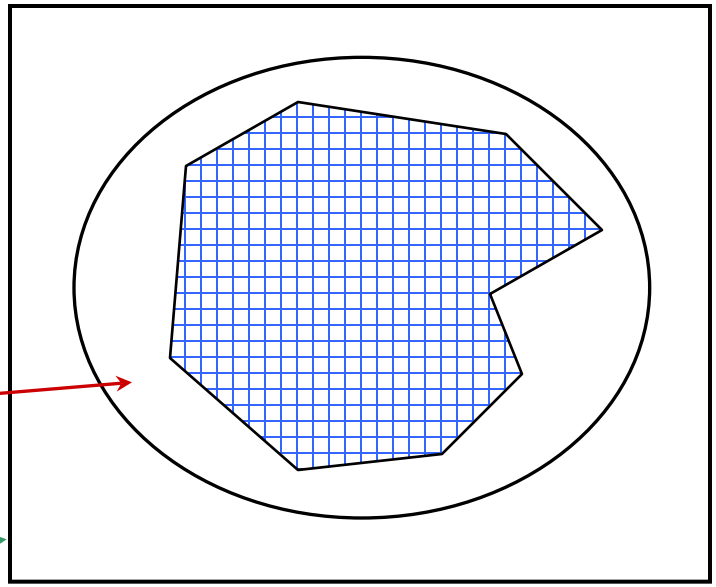
Embedded Mesh Universe

Mesh universe with rectangular background cell and mesh



Simple mesh universe examples.

Mesh universe with elliptical background cell and mesh



Background cell
(for filling the universe)

CSG cell

Embedded Mesh Universe: Cell Cards

- **Cell card requirements:**
  - There can be no cells outside of the background cell in the embedded mesh universe; put CSG cells outside the universe in order to build the remainder of the hybrid geometry.
  - The background cell and pseudo-cell descriptions appear as normal MCNP cell descriptions except they must contain a null surface; enter a zero (0) for the null surface description.

- **Example:**

```
c     *** Cell cards ***
c
      10      2   -7.8240    0                u=2    $ pseudo-cell
      11      1   -1.2230    0                u=2    $ pseudo-cell
      12      2   -7.8240    0                u=2    $ pseudo-cell
      13      3   -0.0012    0                u=2    $ pseudo-cell
      14      0              0                u=2    $ background cell
      20      0             -10     fill=2           $ fill the mesh universe
       6      0        10   -11
       7      0              11
```

Embedded Mesh Data Cards: Geometry

Embedded Mesh Control Card

```
EMBEDn  meshgeo=  mgeoin=  meeout=  meein= length=
        background=  matcell=  filetype=  gmvfile=
```

`n`	embedded mesh universe number n, must match a valid universe # from cell cards
`meshgeo`	mesh geometry type Current permitted values: abaqus, lnk3dnt
`mgeoin`	mesh geometry input file name
`meeout`	elemental edits output file name
`meein`	elemental edits input file name (valid only in continuation runs)

NOTE: all filenames must be lowercase

Embedded Mesh Data Cards: Geometry

Embedded Mesh Control Card (cont.)

```
EMBEDn  meshgeo=  mgeoin=  meeout=  meein=  length=
        background=  matcell=  filetype=  gmvfile=
```

`length`	conversion factor to centimeters for all mesh dimensions in input and output
`background`	cell number of the background cell for the mesh
`matcell`	pairs of numbers: 1st number = mesh material number (actually, the mesh pseudo-cell) + → element to element tracking – → surface to surface tracking (TBI) 2nd number = MCNP cell number (i.e., the pseudo-cell #)

Embedded Mesh Data Cards: Geometry

Embedded Mesh Control Card (cont.)

```
EMBEDn  meshgeo=  mgeoin=  meeout=  meein=  length=
        background=  matcell=  filetype=  gmvfile=
```

`filetype`	output eeout file type; on input, the code determines the correct type Current permitted values: binary, ascii
`gmvfile`	output file name for the gmv geometry only file

Embedded Mesh Data Cards: Edits

Elemental Edits Control Card

`EMBEEn:<pl>  embed=  energy=  time=`

`n`	elemental edit number ending in 4, 6, or 7; follows tally convention
`<pl>`	particle designator from particle list; current valid entries: n or p
`embed`	embedded mesh universe number; must correspond to a valid embed card #
`energy`	conversion factor from MeV/gm to jerks/gm for all energy related output
`time`	conversion factor from shakes for all time related output

Embedded Mesh Data Cards: Edits

Elemental Edit Energy Bins & Multipliers

EMBEBn B_1 B_2 ... B_k

n elemental edit number; 0 is not valid.

B_i monotonically increasing upper energy of the *i'th* bin.

EMBEMn M_1 M_2 ... M_k

n elemental edit number; 0 is not valid.

M_i monotonically increasing upper energy of the *i'th* bin.

Embedded Mesh Data Cards: Edits

Elemental Edit Time Bins & Multipliers

EMBTBn B_1 B_2 ... B_k

n elemental edit number; 0 is not valid.

B_i monotonically increasing upper time of the *i'th* bin. values in units of shakes (1 shake = 10^{-8} s)

EMBTMn M_1 M_2 ... M_k

n elemental edit number; 0 is not valid.

M_i monotonically increasing upper time of the *i'th* bin.

Embedded Mesh Data Cards: Volume Source

Volume Source Control

SDEF **POS= ???**

??? **x, y, z position for point sampling, or volumer for unstructured mesh volume source. Note that the last character "r" stands for sampling by rejection. (An alternative sampling algorithm is planned for the future.)**

Embedded Mesh Data Cards: Volume Source

Volume Source Sampling

- **A source elset or elsets are defined in Abaqus (see previous elset discussion).**
- **An element is selected proportional to the total source volume. That is, proportional to its volume divided by the total mesh source volume.**
- **A position is selected by rejection uniformly in the element's volume.**
  - Least efficient position sampling: tetrahedra & highly distorted elements
  - Most efficient position sampling: hexahedra

Parallel Input

Parallel Input

- **UM problem setup requires nested-loops in several parts of the code. These loops can be time-intensive.**
- **Can speed this up by running mpi.**
  - Want 1 mpi slave node for each instance. The minimum number of mpi processes to specify should be 1 + number of instances in the model.
  - Each instance or part will then have a dedicated processor for its input. At this time, multiple processors per instance or part are not implemented and there is no load balancing.
  - Still limited by the instance / part with the largest number of elements.
- **The most efficient parallel input processing takes place when all parts have approximately the same number of elements and there is more mpi processes than instances.**
  - 30,000 elements per part is a key # for efficient input processing.
  - When there are fewer processes than instances – round robin.

Limitations

Limitation & Restrictions

- limited to only neutrons and photons
- multiple instances of the same unstructured mesh are not functional
- multiple unstructured mesh files are not functional
- unstructured mesh can not be placed inside a lattice
- a universe can not be placed within a mesh universe
- the MCNP plotter will not plot the unstructured mesh, but should plot the background cell
- DXTRAN, point detectors, and forced collisions are known not to work
- surface source reads and writes are not guaranteed to work with the unstructured mesh

Limitation & Restrictions

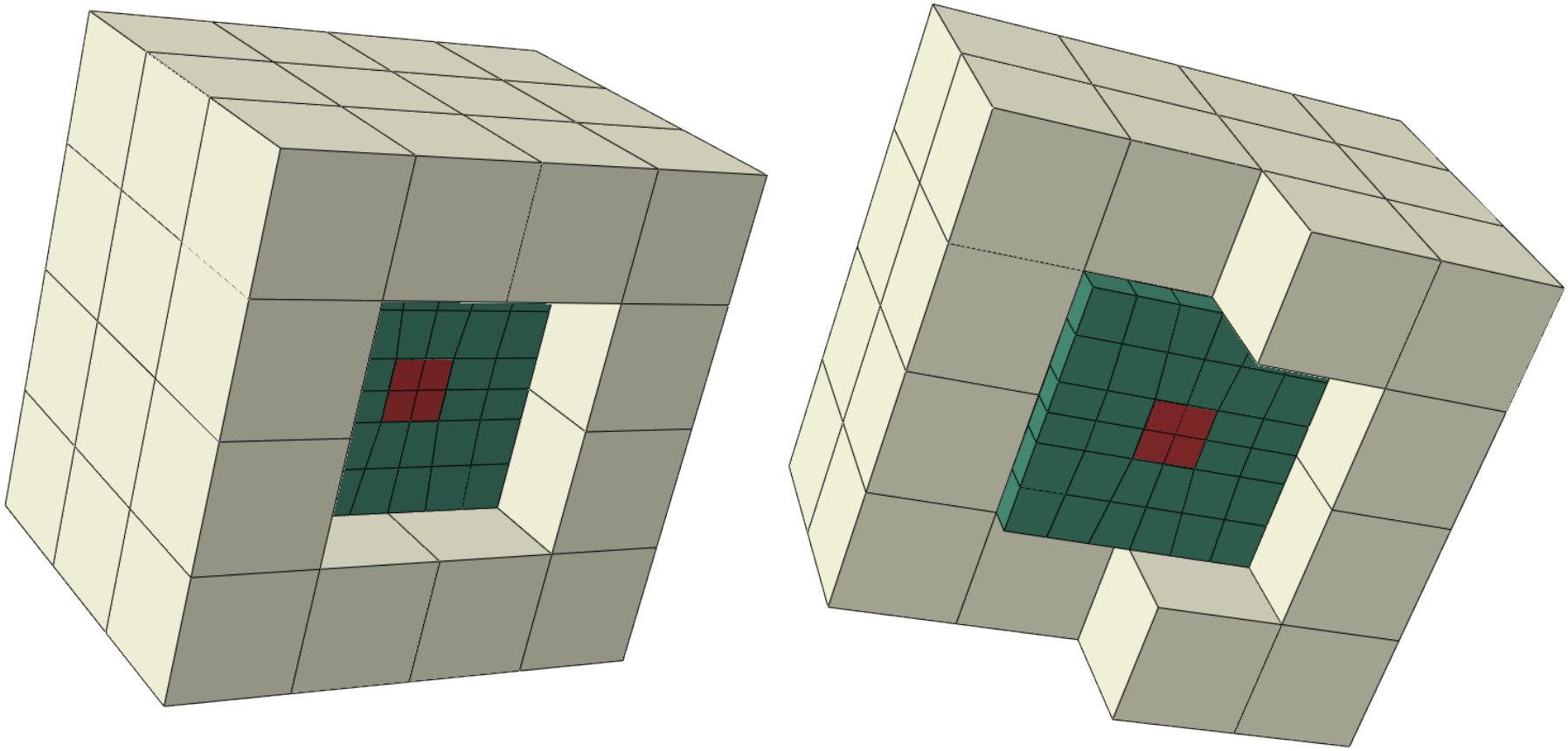
- reflecting and periodic boundary conditions are not guaranteed to work with the unstructured mesh
- source biasing of multiple volume sources in the mesh is not guaranteed to work with the unstructured mesh
- all gaps in the unstructured mesh are treated as voids
- source particles may not be started in mesh gaps
- surface tallies are not permitted in the unstructured mesh
- only pentahedra and hexahedra may appear together in a part; otherwise a part must contain only a single mesh type
- overlapping parts must not be severe; any single element may not be wholly contained within another element

Limitation & Restrictions

- there is no correction while tracking through overlapping elements; a particle tracks in an element until it finds a definite transition point in phase space (i.e., another element, gap, or background cell)
- even running parallel with mpi, problem setup may take considerable time if any one part has many (> 30,000) elements
- CSG surfaces must not clip or intersect the unstructured mesh

Examples

Concentric Cubes



Concentric Cubes: MCNP Input File

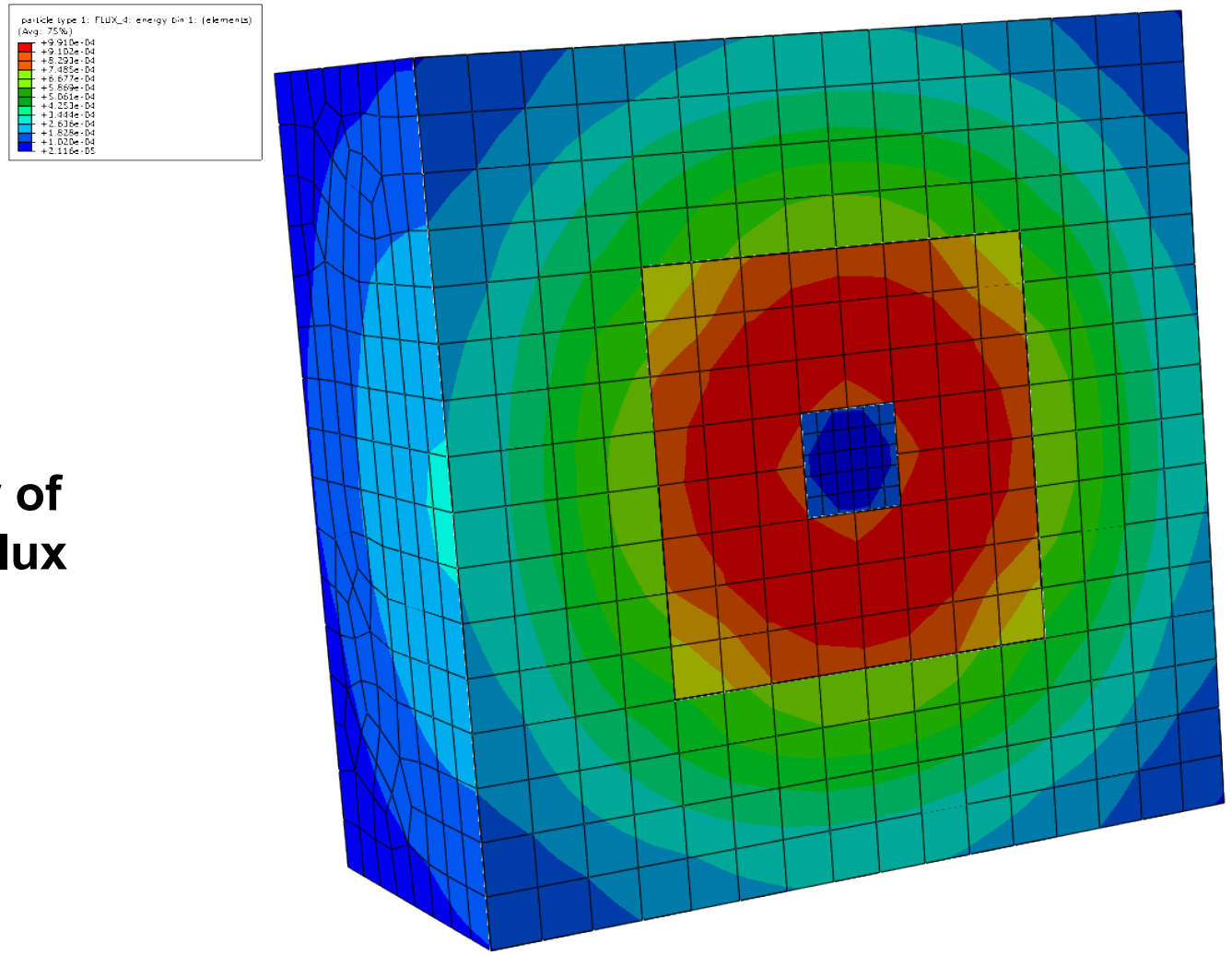
```
Concentric Cubes: 8x8x8 outer; 3 part, 1st order hexs
c
11   1  -2.03          0       u=2
12   2  -0.98          0       u=2
13   3   4.7984e-02    0       u=2
14   3   4.7984e-02    0       u=2
20   0                 0       u=2
30   0               -99       fill=2
40   0                99

99   sph    0.   0. 0. 10.

c  Pseudo-concrete
m1    1001 -0.02   8016 -0.60   14000 -0.38
c
c  Water
m2    1001  2    8016  1
c
c HEU (Godiva) atom density:  4.7984e-02 at/b-cm
m3    92235  4.4994e-02  92238  2.4984e-03  92234
4.9184e-04
c
sdef pos= volumer
kcode 2000 1 10 20 90000
c
mode n
imp:n  1  1  1  1  1  1  0
c
```

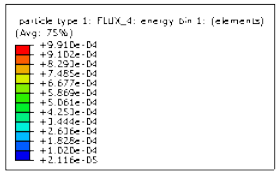
```
c Unstructured mesh data cards
c
embed2 meshgeo=abaqus
       mgeoin=um_8x8x8_3part_cube_s2s_hex_02.inp
       meeout=um_8x8x8_3part_cube_s2s_hex_02.eeout
       gmvfile=um_8x8x8_3part_cube_s2s_hex_02.gmv
       filetype=ascii
       background= 20
       matcell= 1 11  2 12  3 13  4 14
c
c
embee4:n  embed=1
embtb4  1e+39
embtm4  1.0
embeb4  1e+10
embem4  1.0
c
c Tallies
f14:n  11
```

Concentric Cubes: Abaqus/CAE Viz

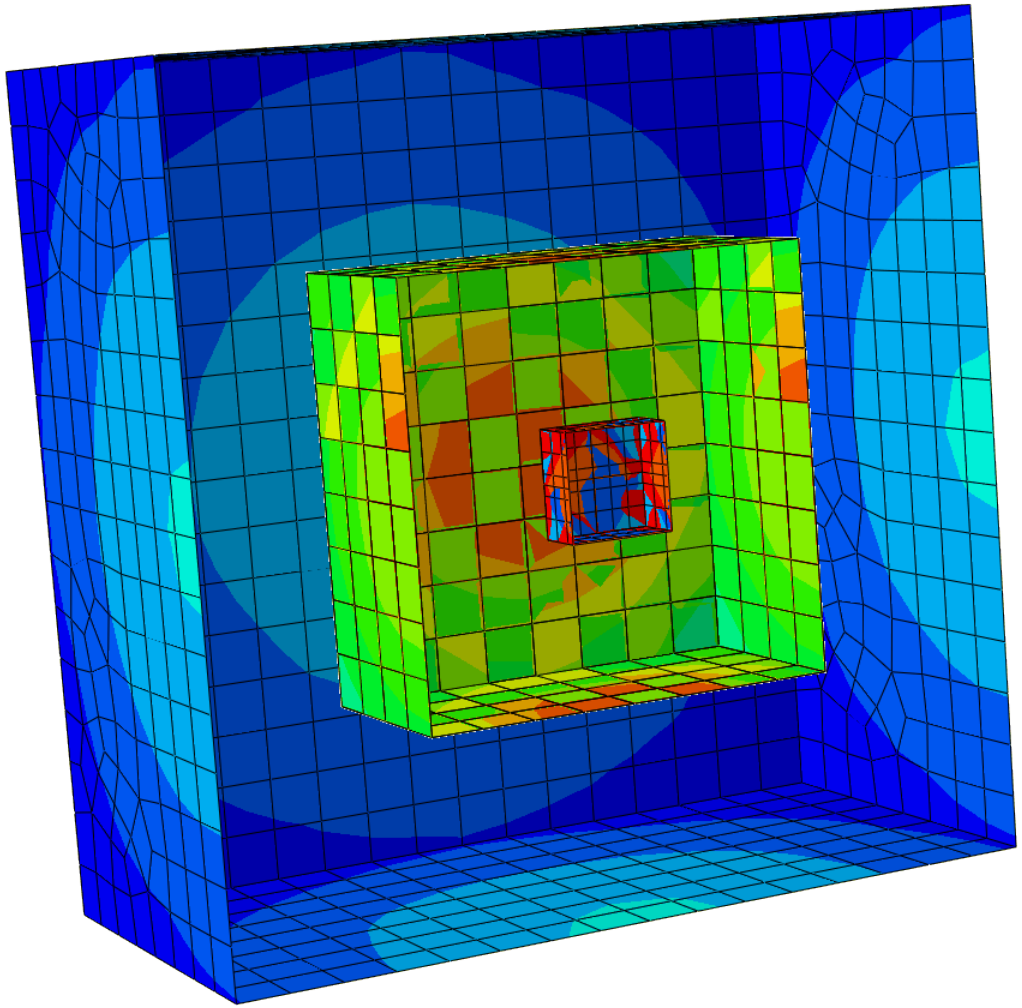


Cut away view of total neutron flux

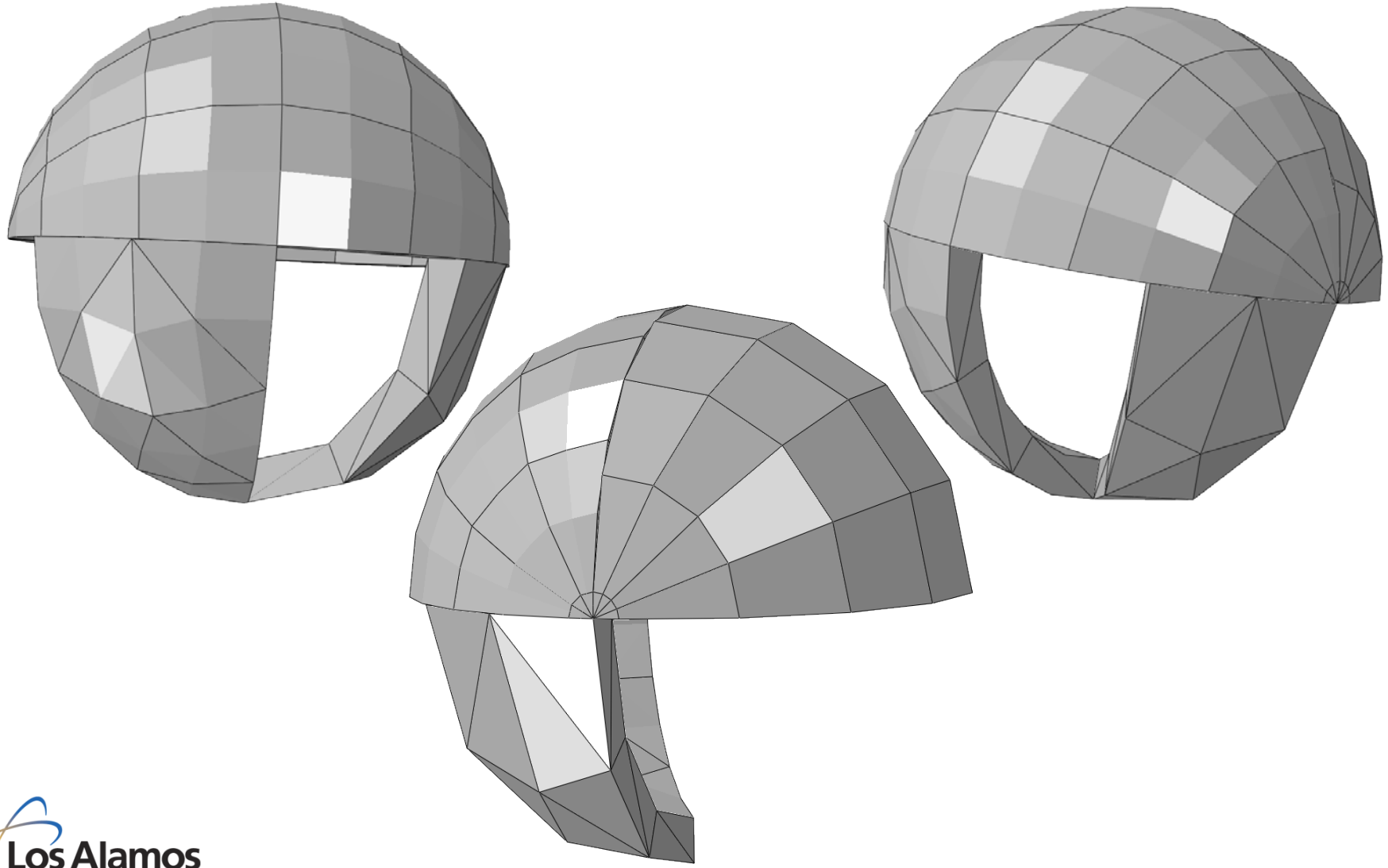
Concentric Cubes: Abaqus/CAE Viz



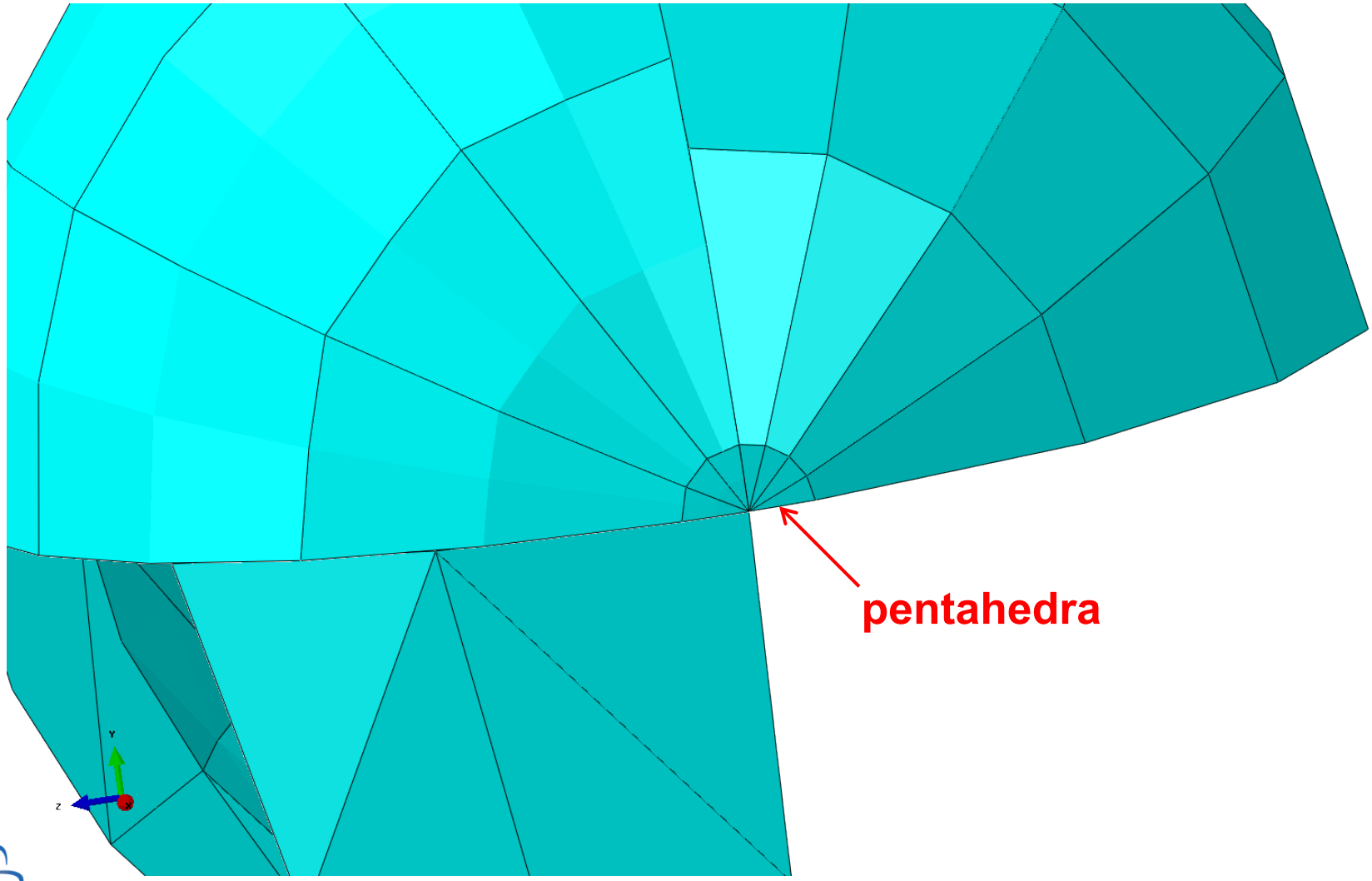
Cut away view of neutron flux on surfaces of the 3 parts



Helmet: Mixed Element Types



Helmet: Mixed Element Types



MCNP User's Checklist For Abaqus Modeling

- **Define materials**
  - Appropriate name & number plus density
- **Define sections**
  - (primarily needed to get geometry vizualization correct in Abaqus/CAE)
- **Create simple parts**
  - Appropriate elsets: material, statistic, source
  - Section assignments
  - Mesh
- **Create any complex parts**
  - Inherits elsets from simple parts at time of creation
  - Mesh
- **Create assembly**
  - Translations & rotations
- **Write job file (i.e., the Abaqus .inp file)**
  - File name for input to MCNP
  - Job description for MCNP command line

Utility Programs

Utility Programs

- **um_pre_op : pre-processing for MCNP input**
- **um_pre_op : post-processing of the eeout file**
  - Binary to ASCII file conversion
  - File merging
  - Writing VTK format

Utility Program: um_pre_op

- **The um_pre_op program is a utility program that performs various "manipulations" to generate a basic MCNP input file.**
- **This program is written in Fortran and uses various routines and data structures from the Revised Eolus Grid Library, REGL, in order to maintain consistency with MCNP.**
  - Ensures that the utility program uses the same routines that MCNP uses to read and generate its global mesh model.
- **um_pre_op is designed to run from the command line.**
- **A new version is automatically built every night to coincide with the latest MCNP6 bleeding edge version.**

Utility Program: um_pre_op

Functions

- Read the Abaqus .inp file and generate a “skeleton” input file for MCNP.
- Convert a CUBIT – Exodus file to an Abaqus .inp file – not fully implemented at this time.

Utility Program: um_pre_op

Command Line Arguments:

```
-h, --help          summary of features & arguments
-m, --mcnp          generate MCNP skeleton input file
-o, --output        output file name
-ex, --extension    output file extension
-dc, --datacards    data cards file to include
```

Utility Program: um_pre_op

Generating Basic MCNP Input file

- A basic MCNP input file called “mynewinput”. Contains pseudo-cells, minimal legacy cells, single surface that defines mesh universe, and minimal data cards.

```
um_pre_op -m -o mynewinput abaqus_geom.inp
```

Note that the first argument after the -o argument is interpreted as the output file name.

Utility Program: um_pre_op

Generating A More Complete MCNP Input file

- A complete MCNP input file called “mynewinput” can be created if the data cards file contains the appropriate information and the entire geometry of interest is represented by the unstructured mesh.

```
um_pre_op -m -o mynewinput abaqus.inp -dc dc_cards
```

Utility Program: um_pre_op

Restrictions

- A basic mesh universe is created that contains the UM without clipping it; if you need additional CSG cells, then create them by hand.
- Material numbers in the data cards file must be consistent with the material numbers used in Abaqus/CAE.
- If there is no active sdef card in the data cards file and there is a volume source specified in the .inp file, a skeleton sdef card is included for the volume source.
- If there are no imp cards in the data cards file, default imp cards will be generated based upon the particles found on the mode card in the data cards file.

Utility Program: um_pre_op

! CAUTION !

Do not expect the MCNP input file generated by um_pre_op to run if any of the known restrictions from the previous slide are not addressed.

The intent of this utility program is to make it easy for the user to quickly get up and running with the UM capability by generating a consistent set of pseudo-cell cards and matcell parameters on the embed card.

Utility Program: um_post_op

- **The um_post_op program is a utility program that performs various manipulations on MCNP's elemental edit output file, eeout.**
- **This program is written in Fortran and uses various routines and data structures from the Revised Eolus Grid Library, REGL, in order to maintain consistency with MCNP.**
  - Ensures that the utility program uses the same routines that MCNP uses to read and write the eeout file.
- **um_post_op is designed to run from the command line.**
- **A new version is automatically built every night to coincide with the latest MCNP6 bleeding edge version.**

Utility Program: um_post_op

Functions

- merge many eeout files into one
- convert binary files into ASCII files
- generate vtk files for VisIt visualization

Utility Program: um_post_op

Command Line Arguments:

```
-h, --help          summary of features & arguments
-m, --merge         merge multiple files
-o, --output        single output file name
-bc, --binconvert   convert binary file to ASCII
-ex, --extension    multiple output file extension
-vtk, --vtkfile     generate ascii visualization files
```

Utility Program: um_post_op

Merging Files

Example command line:

```
um_post_op -m -o my_merge_file eeout1 eeout2 ... eeoutN
```

Note that the first argument after the -o argument is interpreted as the output file name.

NOTE: Files are checked for consistency before merging takes place.

Utility Program: um_post_op

Converting Files

- Convert from binary format to ASCII.
- In performing this operation there is a loss of precision since all double precision reals are written with only six significant digits.
- Currently, there is no capability to convert from ASCII to binary.
- On the command line, one or many files may be specified for conversion.
- Why convert? The python results processer for Abaqus/CAE works much faster with an ASCII file.

Utility Program: um_post_op

Converting Files

- When the conversion request asks for only one file, the `-o` argument may be used. Example:

```
um_post_op -bc -o eeout.ascii eeout.binary
```

- It is also legitimate to use the `-ex` argument. Example:

```
um_post_op -bc -ex ascii eeout.binary
```

The resulting output file is named: `eeout.binary.ascii`

Utility Program: um_post_op

Converting Files

- When more than one file is to be converted, the `-ex` argument must be used. Example:

```
um_post_op -bc -ex asc eeout1 eeout2 ... eeoutN
```

- The resulting files appear with the names:

```
eeout1.asc eeout2.asc ... eeoutN.asc
```

Utility Program: um_post_op

Visualization Files

- Generates files in the vtk format for visualization.
- The geometry data and the edit information is taken from the *eeout* file and reformatted to be consistent with version 4.2 of the vtk standard and written to an ASCII file. (this has not been thoroughly tested)
- Details on the vtk file format and requirements can be found in the vtk documentation, available on the worldwide web and in text books.

Utility Program: um_post_op

Visualization Files

- When the generation request asks for only one file, the `-o` argument may be used. Example:

```
um_post_op -vtk -o eeout.vtk eeout1
```

- It is also legitimate to use the `-ex` argument. Example:

```
um_post_op -vtk -ex vtk eeout1
```

The resulting output file is named: `eeout1.vtk`

Utility Program: um_post_op

Visualization Files

- When more than one file is to be generated, the `-ex` argument must be used. Example:

```
um_post_op -vtk -ex vtk eeout1 eeout2 ... eeoutN
```

The resulting files appear with the names

```
eeout1.vtk eeout2.vtk ... eeoutN.vtk
```

Utility Program: um_post_op

Miscellaneous

- **The REGL routine that reads valid *eeout* files has the ability to detect whether the file it is reading is ASCII or binary. If it can't make a determination that the file is a valid *eeout* file, an error message appears in the terminal window. Therefore, when a list of files is specified on the command line, for either merging or generating vtk files, they may be a mixture of ASCII or binary.**

Performance

Performance

The next several slides are from

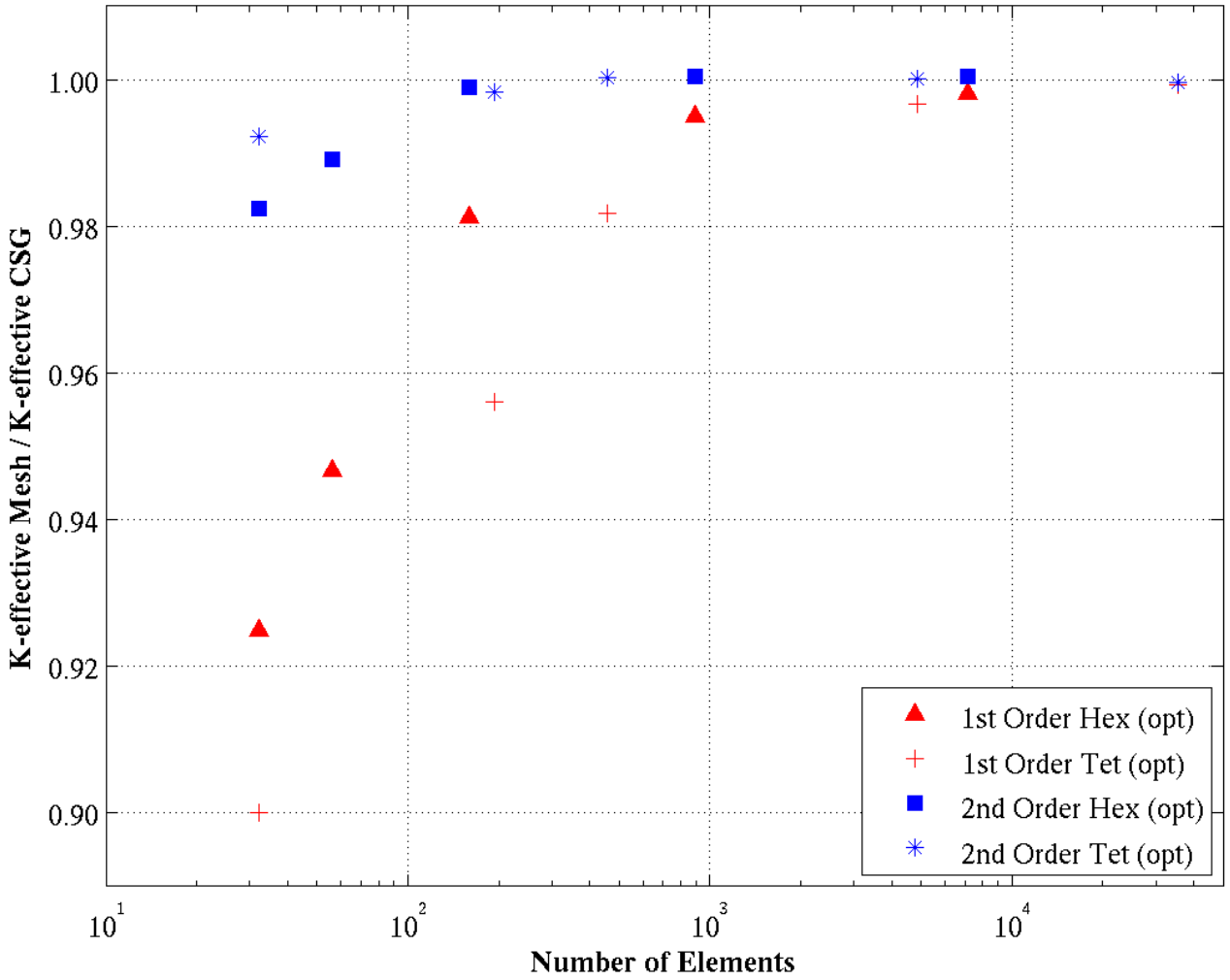
- **Karen C. Kelley, Roger L. Martz, and David L. Crane, *Riding Bare-Back On Unstructured Meshes For 21st Century Criticality Calculations*, PHYSOR 2010 – Advances in Reactor Physics to Power the Nuclear Renaissance, Pittsburgh, Pennsylvania, May 9-14, 2010, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009).**
- **Roger L. Martz, MCNP6 *Unstructured Mesh Initial Validation and Performance Results*, LA-UR-11-04657, Nuclear Technology (to be published).**

Performance

When preserving volumes & mass is important, use 2nd order elements. Fewer elements may be needed.

Example:

Godiva criticality

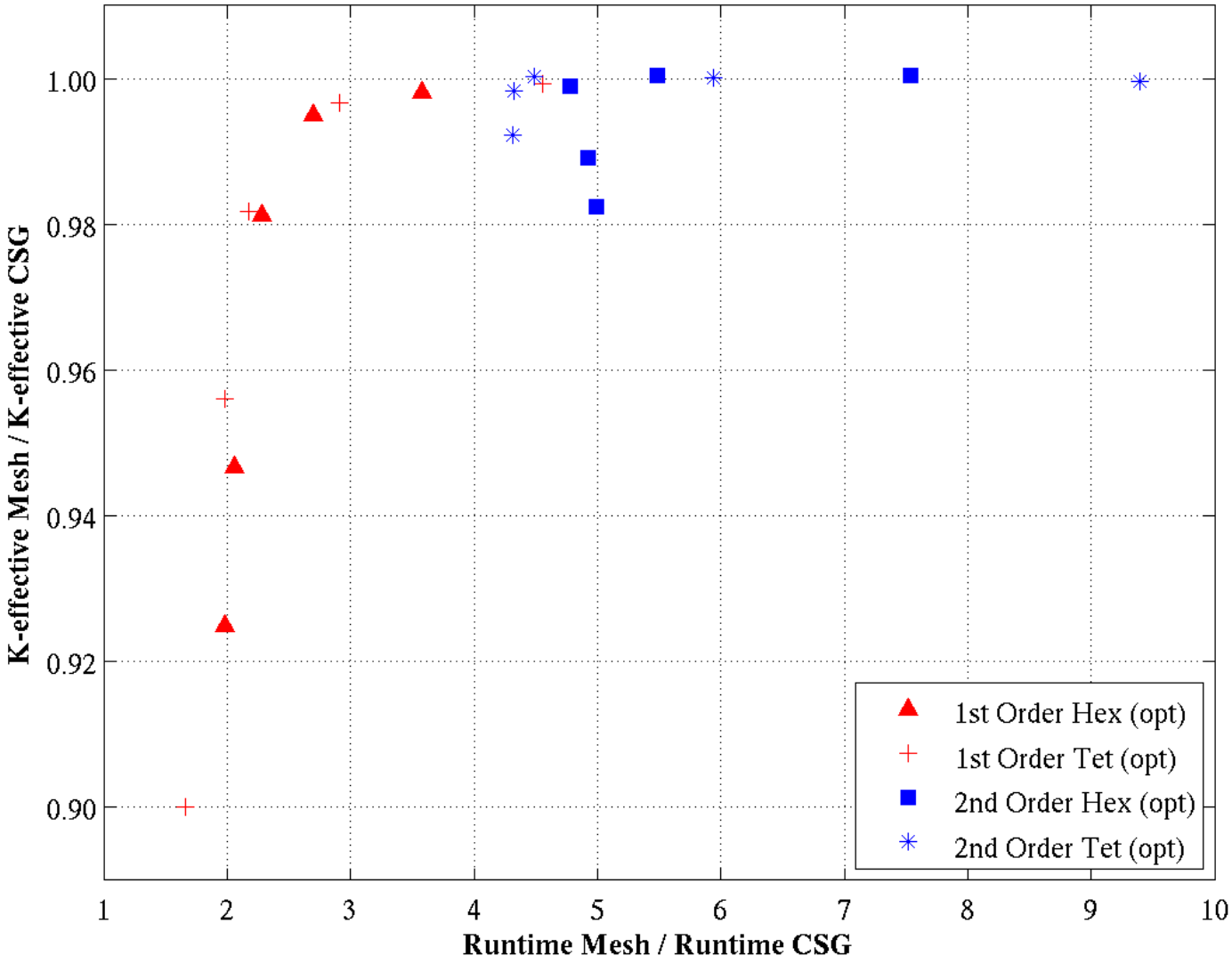


Performance

1st Order elements are faster.

Example:

Godiva criticality



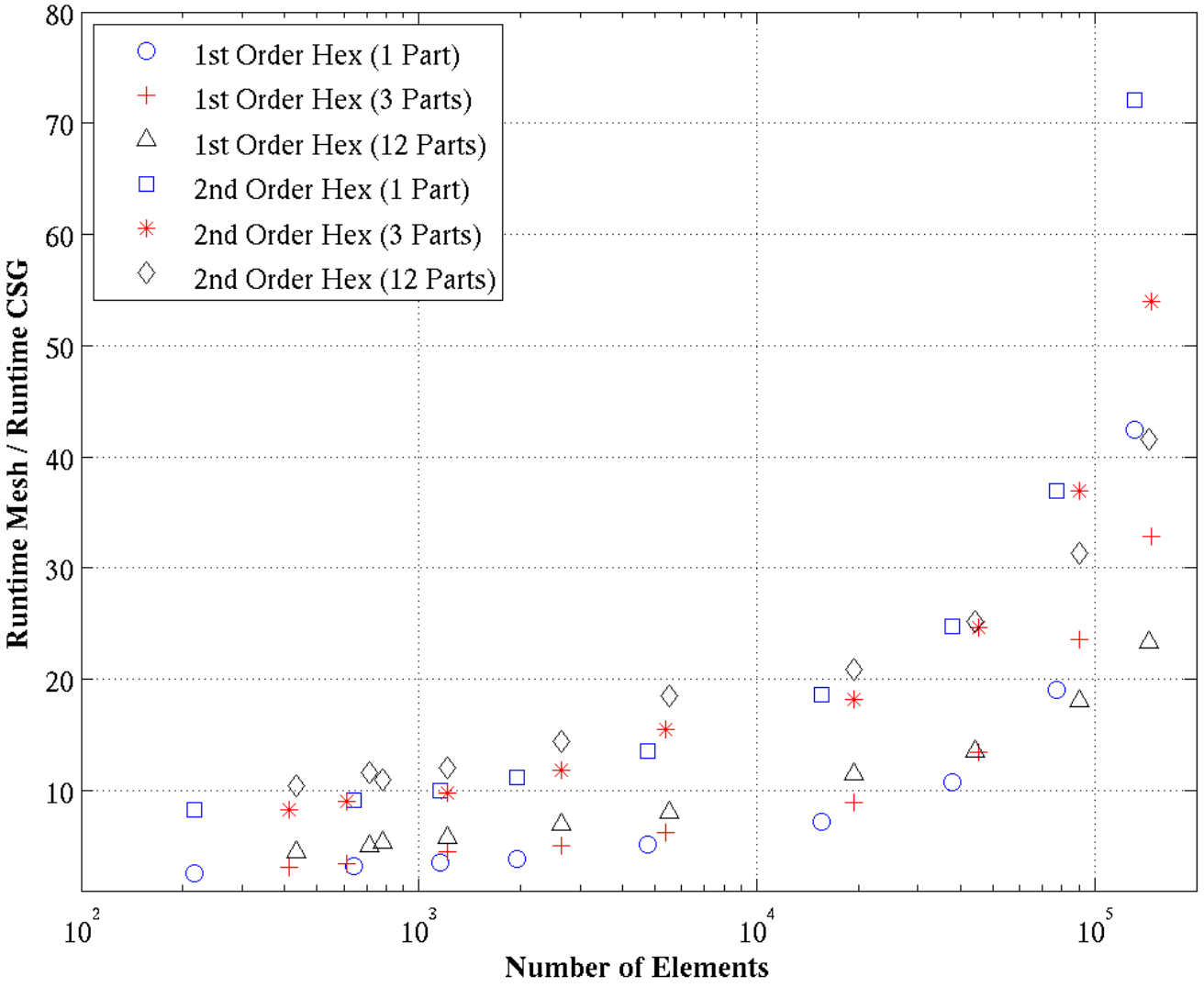
Performance

1st Order elements are faster.

# of parts and # of elements per part matter.

Example:

Osaka nickel sphere benchmark (hexs)



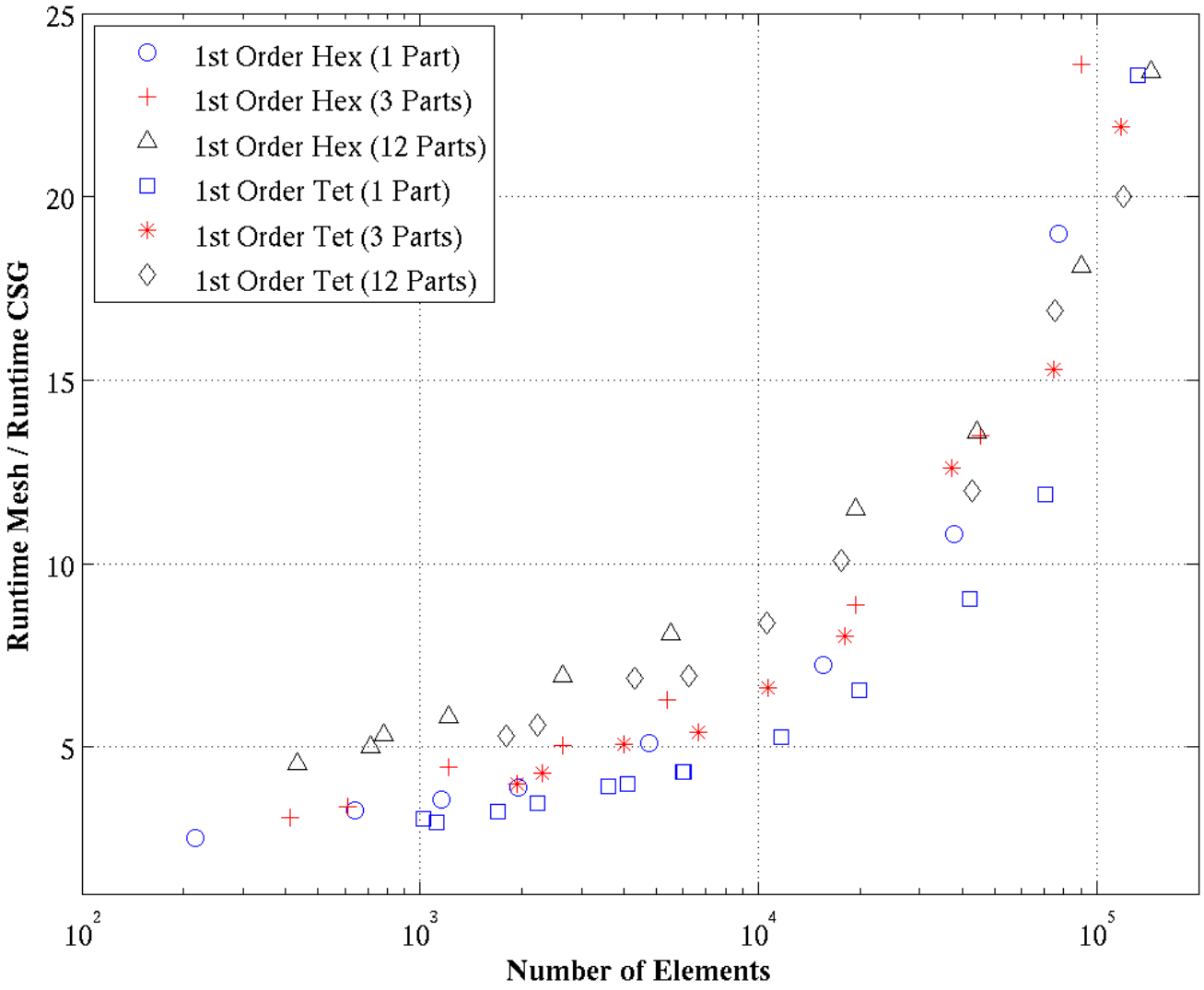
Performance

Tets elements are faster.

# of parts and # of elements per part matter.

Example:

Osaka nickel sphere benchmark (1st order)



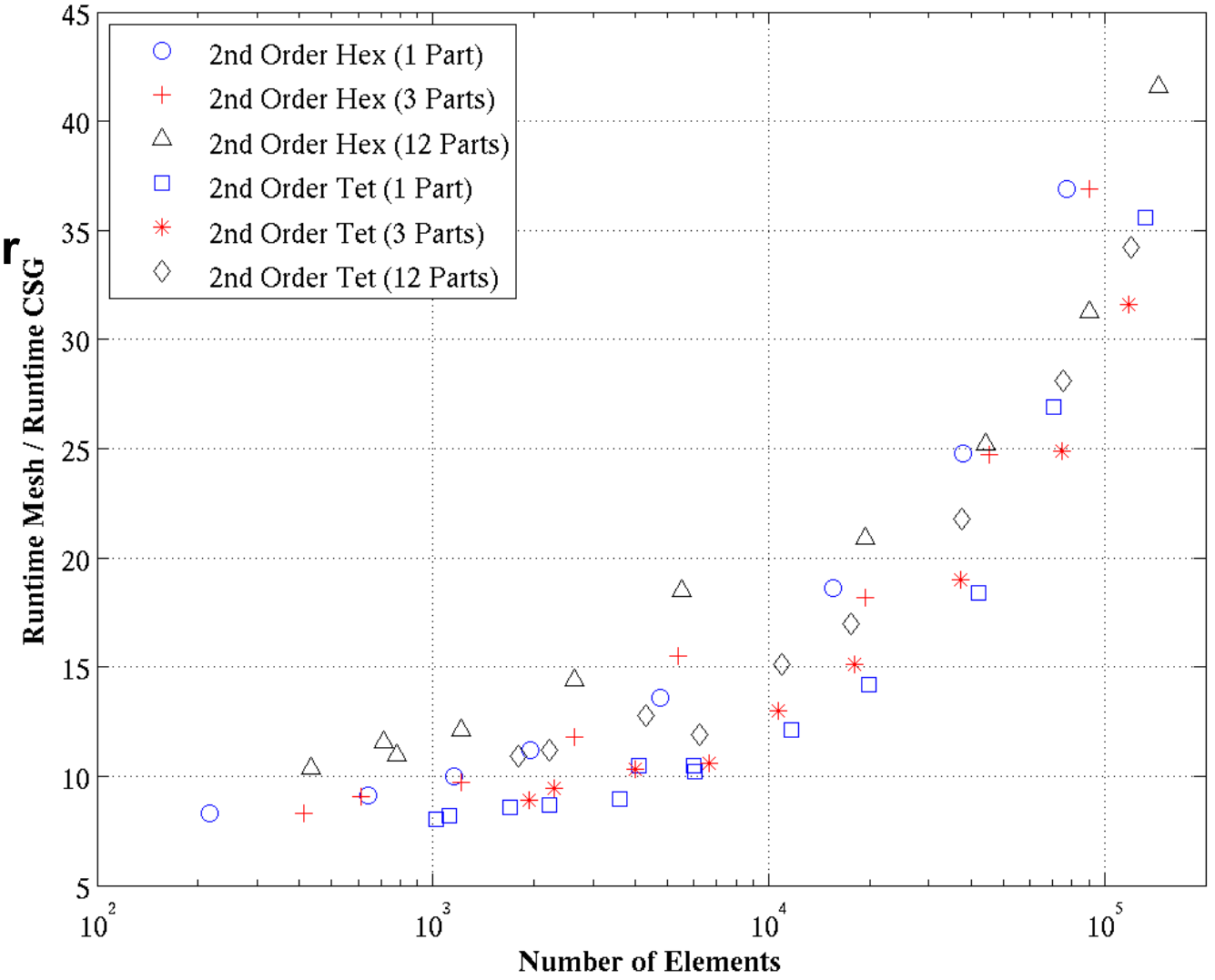
Performance

Tets elements are faster.

# of parts and # of elements per part matter.

Example:

Osaka nickel sphere benchmark (2nd order)



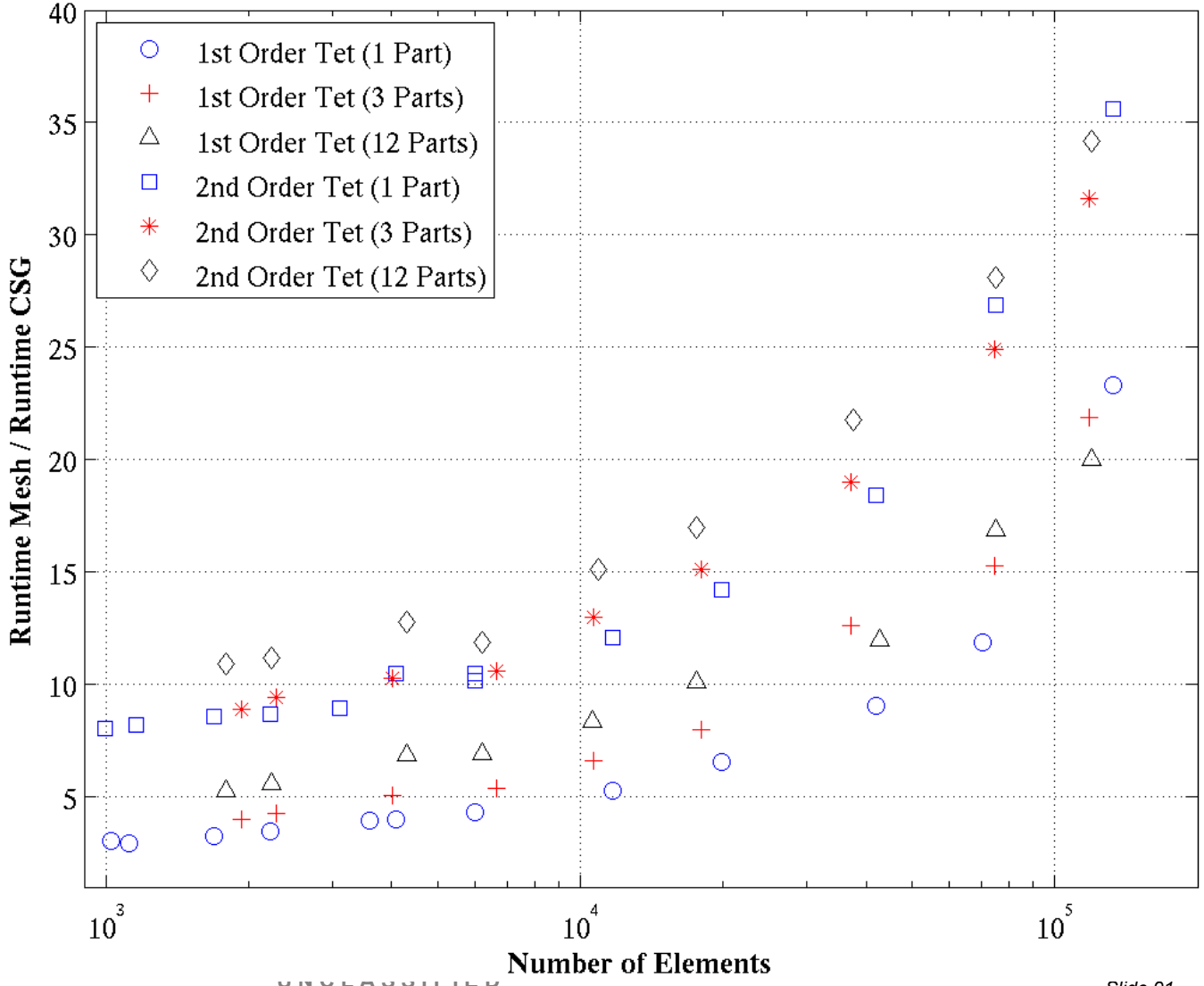
Performance

1st Order elements are faster.

# of parts and # of elements per part matter.

Example:

Osaka nickel sphere benchmark (tets)



Performance

Direct Comparison

The following slides show a comparison between CSG and UM for the Godiva and Osaka benchmarks where

- **The models use only 1st order tets.**
- **Each tet in the UM is modeled exactly using combinations of arb surfaces in CSG.**
  - Thanks to Kevin Marshall, et al., of the Radiation Science's Group, AWE Plc. for providing the program to convert the Abaqus .inp file into an MCNP CSG input deck.
- **The total number of histories were chosen so that the most detailed models could be run in a reasonable amount of time with 1 processor.**

Performance

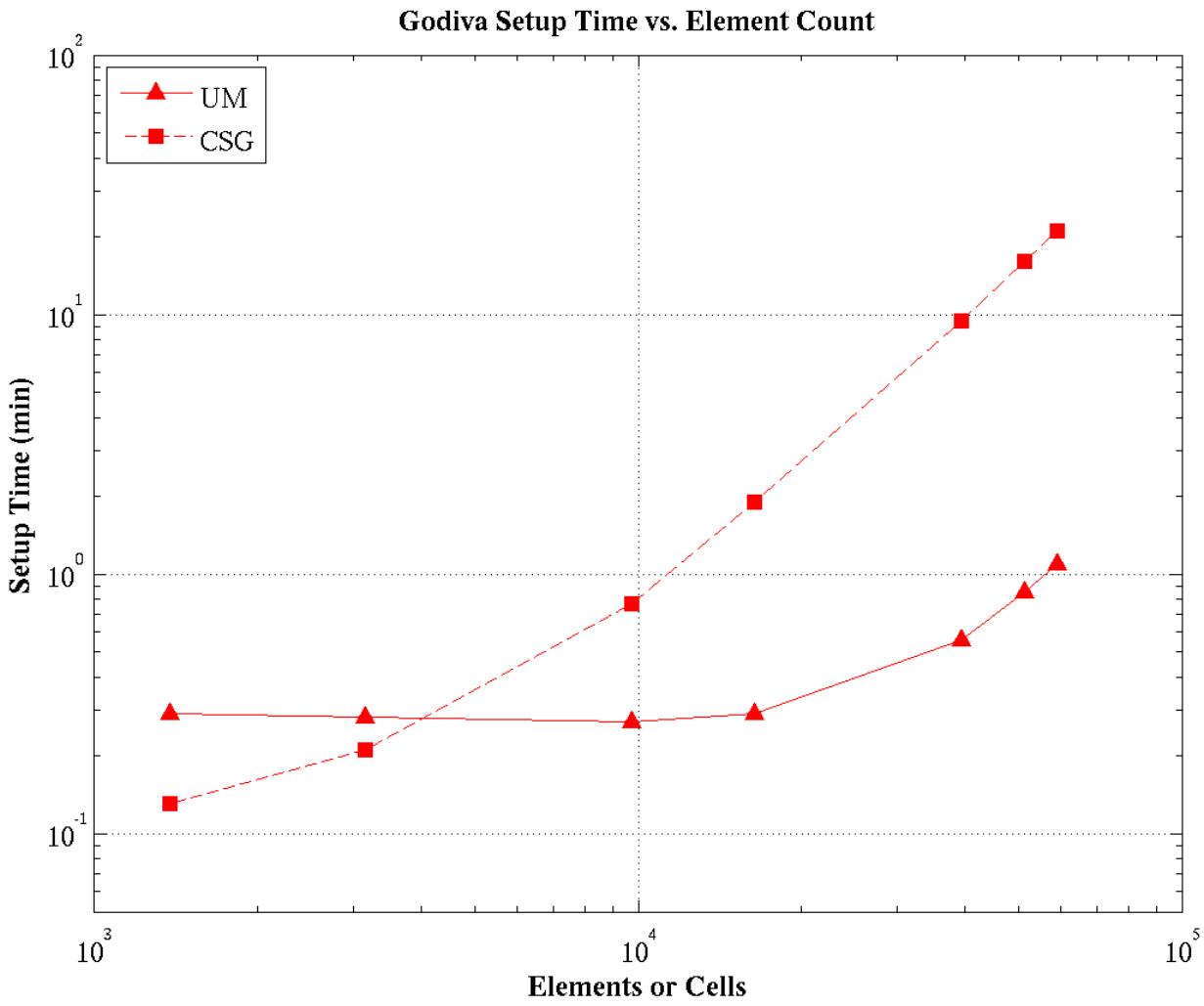
Direct Comparison

- **Godiva: 600,000 histories; 5000 histories per batch, 120 total batches with 20 inactive batches.**
- **Osaka: 100,000 histories.**

Performance

Godiva:
Setup Time vs.
Element Count

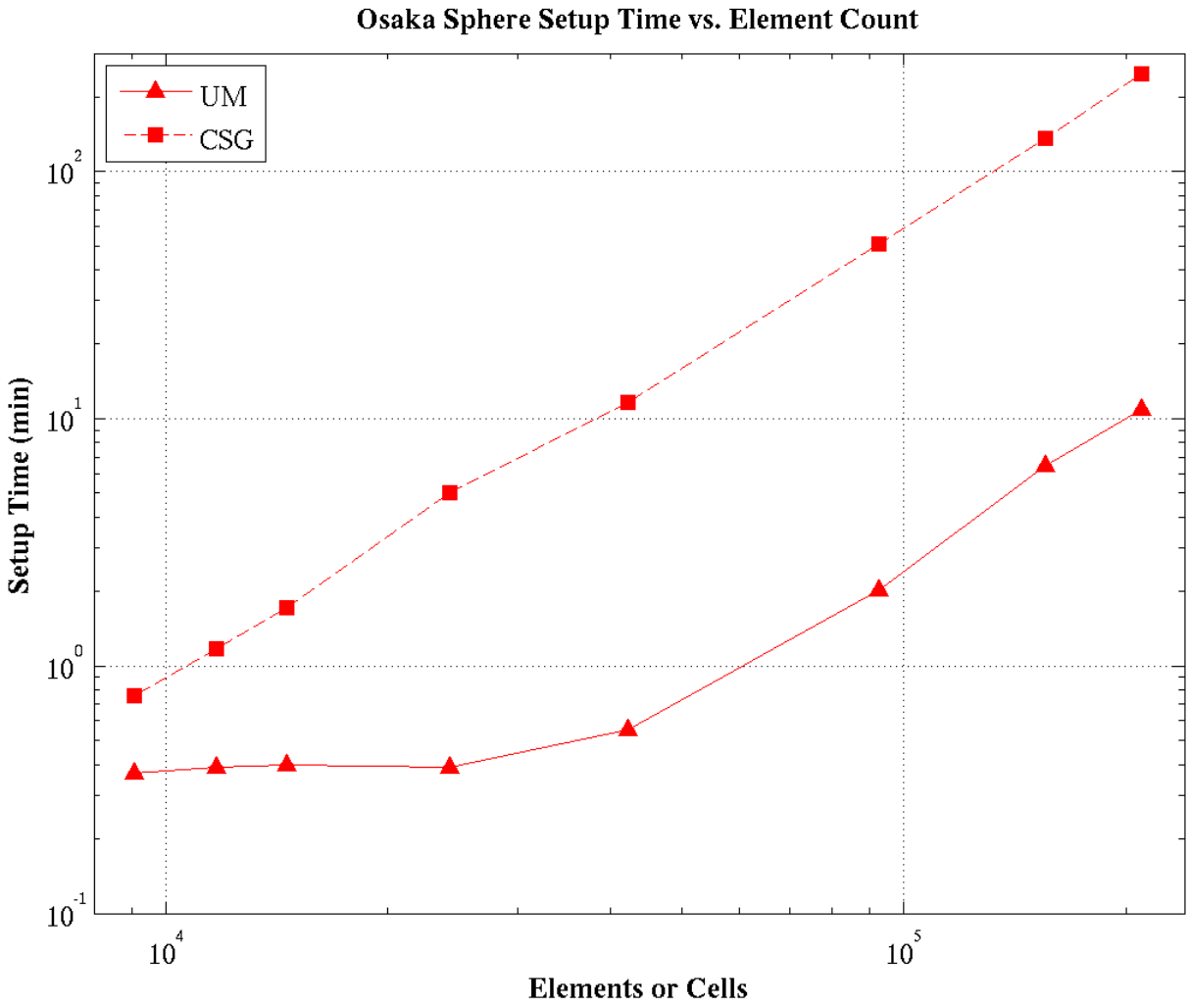
1 part in Abaqus model



Performance

Osaka:
Setup Time vs. Element Count

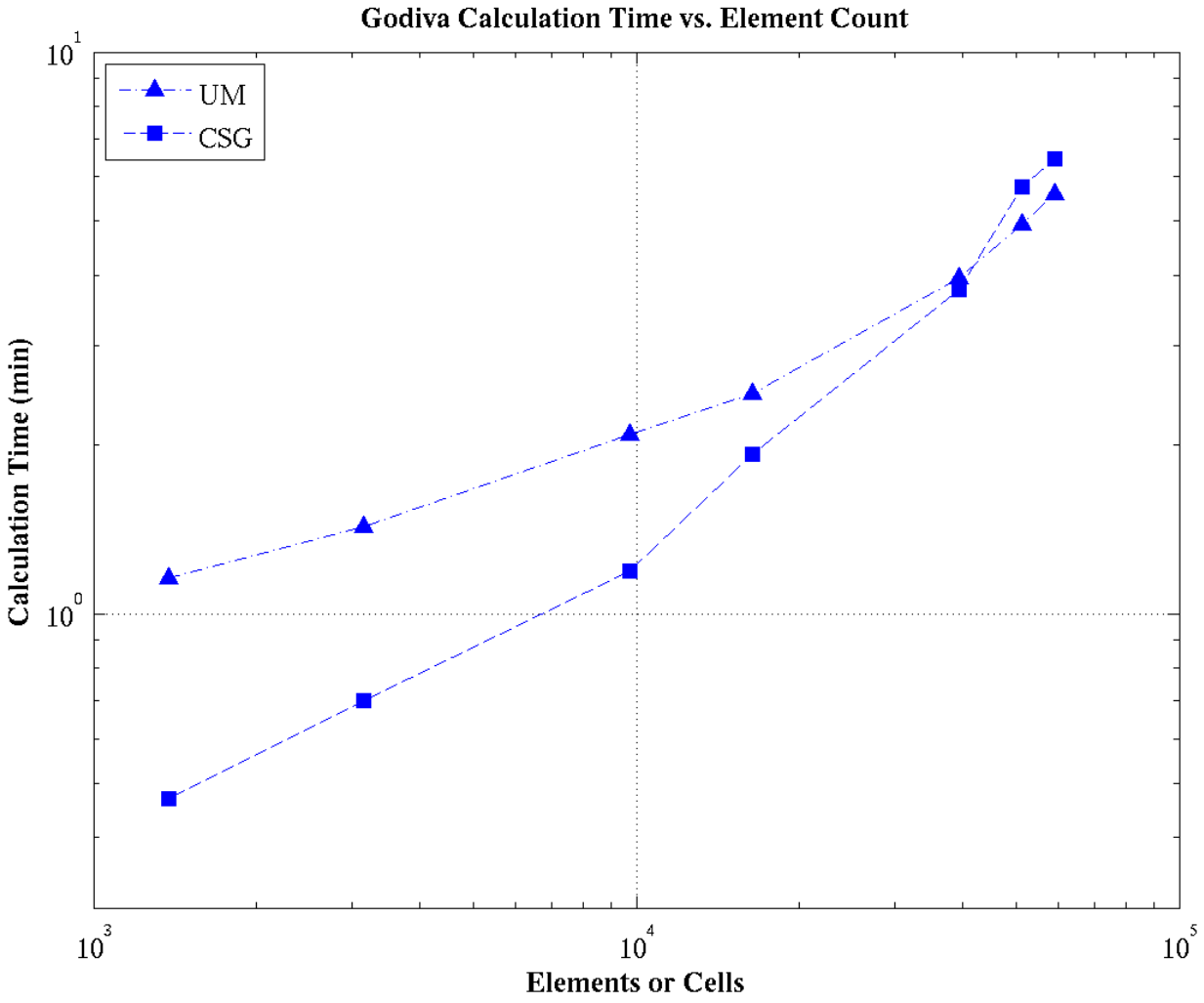
3 parts in Abaqus model



Performance

Godiva: Calculation Time vs. Element Count

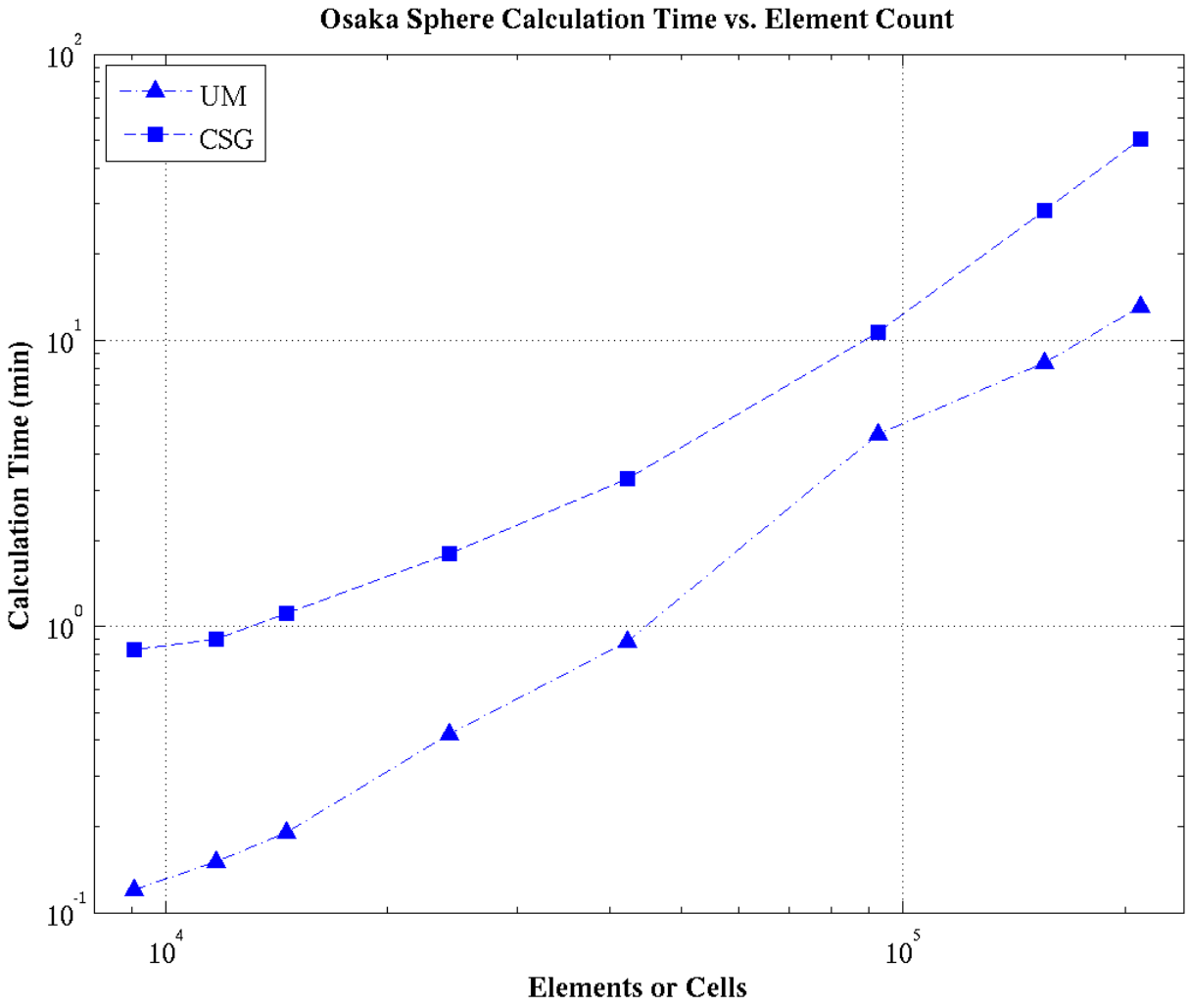
1 part in Abaqus model



Performance

Osaka: Calculation Time vs. Element Count

3 parts in Abaqus model



Performance

Preliminary Conclusions

While the number of problems that have been looked at so far has been limited and there is much more to understand, the following appears to be true:

- **As the detail increases (i.e. the element count rises), the setup time is significantly quicker with the unstructured mesh and the calculational component is faster.**

Expert Advice & Opinion

Expert Advice & Opinion

Why Abaqus/CAE ?

- **It is LANL's program of choice.**
- **Other CAE tools can write an Abaqus .inp file, but**
  - May not support 2nd order elements
  - May not be able to generate elsets or define materials as needed by MCNP
  - May not support visualization of results (a separate or different program maybe required)
  - May not allow partitioning of parts

Expert Advice & Opinion

Using somebody's already generated CAD file

- **May contain too many irrelevant features for the MCNP model**
  - Much time may be required to "de-feature" the existing model
- **May contain unsupported element types such as 2-D surface elements**
  - Totally un-useable without a great deal of work
- **Fidelity of the model may not be good enough for MCNP**
  - Dimensions may not be very accurate
  - Parts man not touch other parts exactly where they should
  - Much time may be required to fix

Expert Advice & Opinion

Mesh & The Monte Carlo Method

The mesh in MCNP is used for 2 basic purposes:

1. **Serve in defining boundaries between parts (i.e., cells)**
   - More elements or 2nd order elements may be required to accurately represent the boundary and the associated volume and mass that it encloses.
2. **Collect edit results for visualization**
   - Mesh granularity should be increased in areas of large gradients for better representation during visualization.

Expert Advice & Opinion

Think Through All Requirements Before Modeling

- **Fast input processing and calculation times point to parts with 30,000 elements or less.**
  - Can be accomplished with more, smaller parts or re-meshing.
- **Input processing time for models with many instances can be minimized when running with the mpi version of MCNP.**
  - Minimum number of mpi processes: 1 + number of instances in the model
- **Being able to use a certain element type may require partitioning the part.**
  - Important if hexahedra are desired.
- **Peeling away the outer layers of the model may require partitioning of the part such that removal of groups of elements result in a smooth surface.**

Extra Slides

Tracking On The Mesh

Unstructured Mesh Tracking

Eventually, there will be at least two types of tracking implemented on the unstructured mesh:

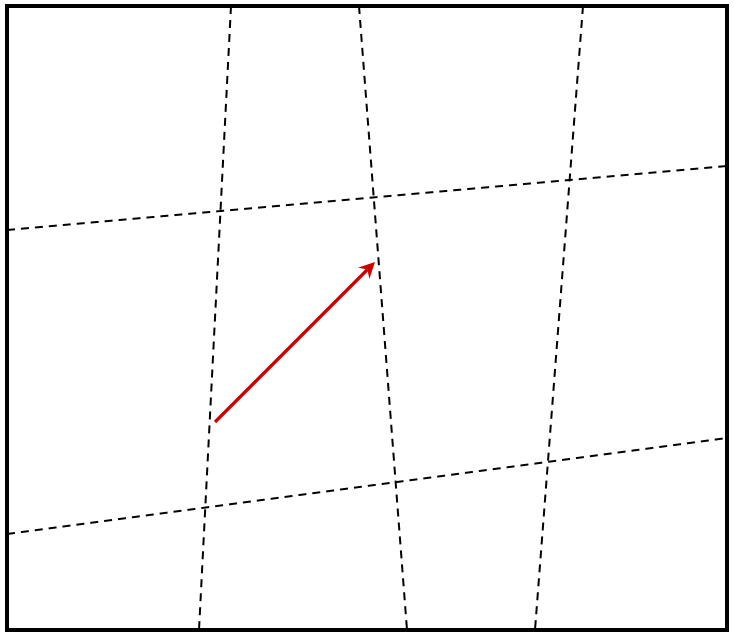
- **Element-to-element**
- **Surface-to-surface**

They primarily differ on the "granularity" of the geometry they use.

(see the next two slides for details)

Element-to-Element Tracking

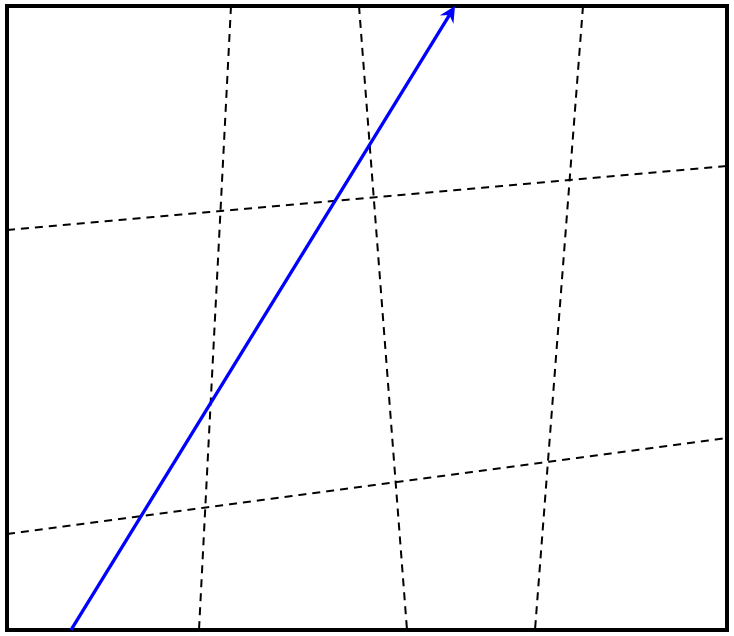
- **Tracking is from face-to-face of the element (barring a collision event or any other event that MCNP checks such as distance to dxtran surface).**
- **Path length results are collected for each element that the particle travels through.**



12-element part

Surface-to-Surface Tracking

- **Tracking is from surface-to-surface of the part (barring a collision event or any other event that MCNP checks such as distance to dxtran surface).**
- **No path length results are collected on the elements.**



12-element part