

LA-UR-26-27281

Approved for public release; distribution is unlimited.

Title: Initialization Improvements in MCNP6.4

Author(s): Swaminarayan, Sriram

Intended for: Meeting: 2026 MCNP Symposium, 2026-08-24 - 2026-08-26 (Los Alamos, New Mexico, UNITED STATES)

Issued: 2026-08-25 (rev.1)



Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Initialization Improvements in MCNP6.4

Sriram Swaminarayan

August 24, 2026

LA-UR-26-27281

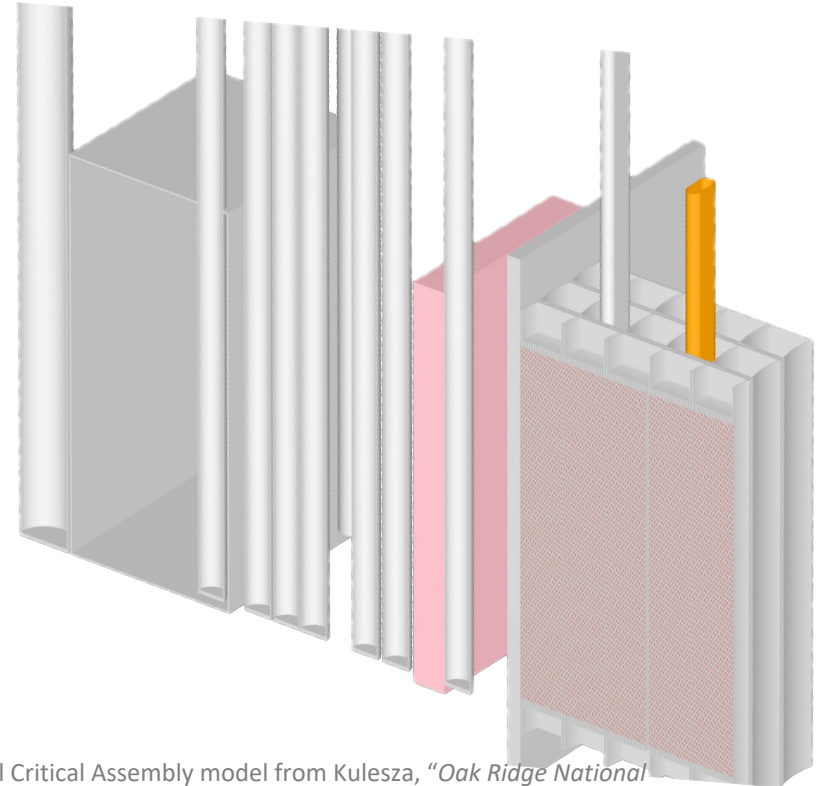
Motivation For This Work: Speed & Memory

Constructive Solid Geometry (CSG)
initialization can be slow

- Algorithms don't scale to complex geometries with over 10^3 surface / cell cards

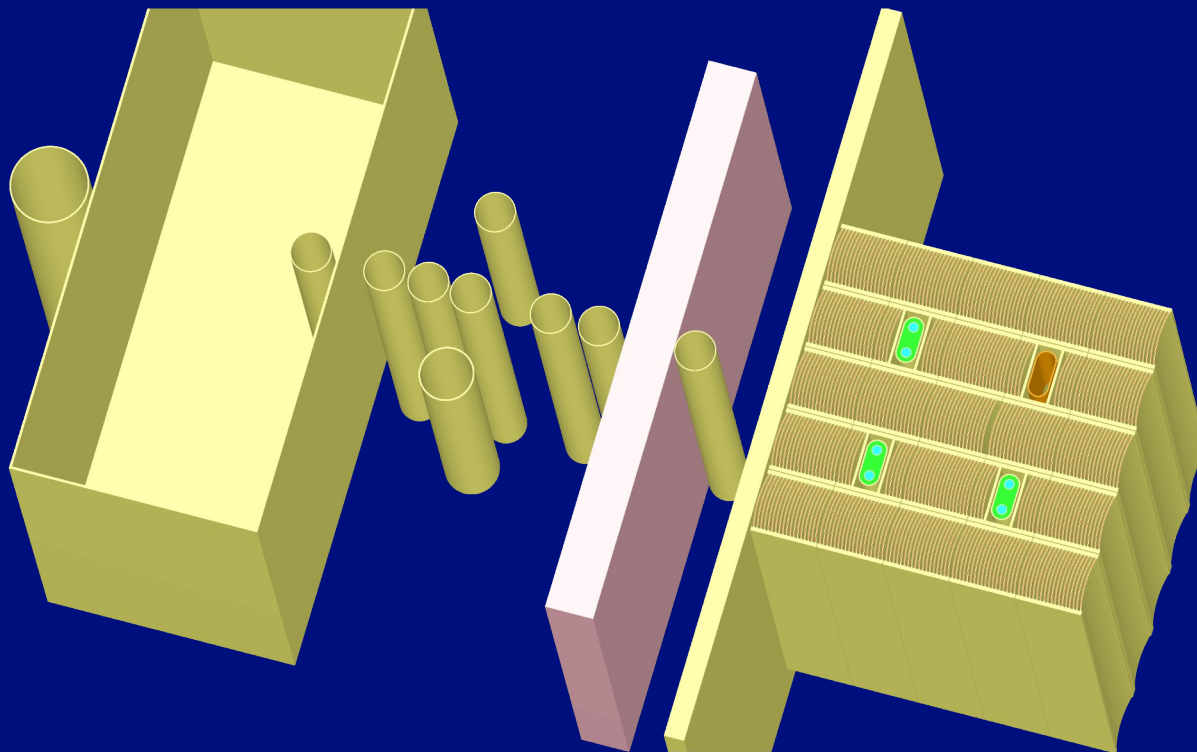
Unstructured Mesh (UM) initialization
exceeds runtime memory footprint

- Complex geometries with parts consisting of differing number of elements
- Running in parallel with multiple MPI ranks



Pool Critical Assembly model from Kulesza, "Oak Ridge National Laboratory Pool Critical Assembly MCNP6 Criticality Calculation Input File", doi: 10.2172/1601379

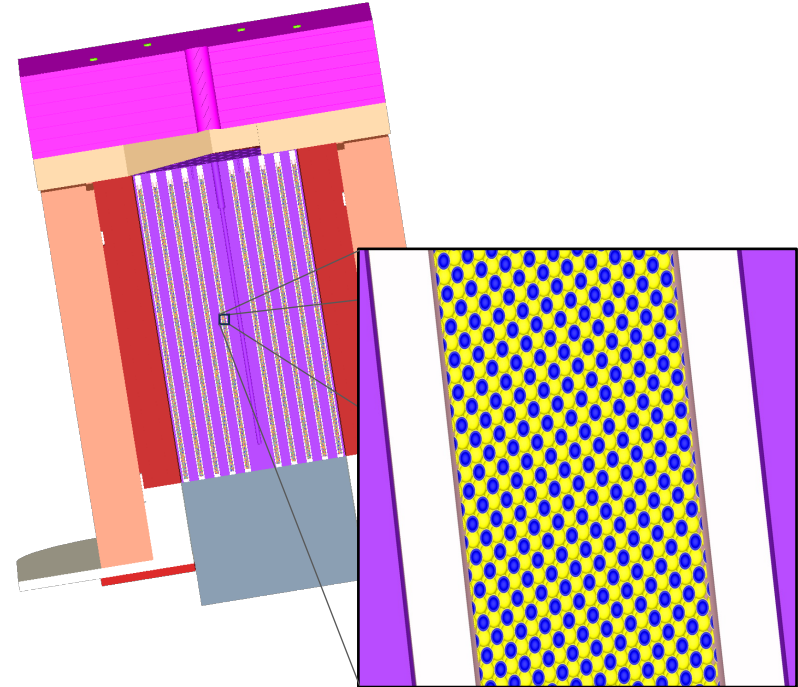
Part I: Constructive Solid Geometry



Pool Critical Assembly model from Kulesza, "Oak Ridge National Laboratory Pool Critical Assembly MCNP6 Criticality Calculation Input File", doi: 10.2172/1601379

Bottlenecks in CSG Initialization Routines

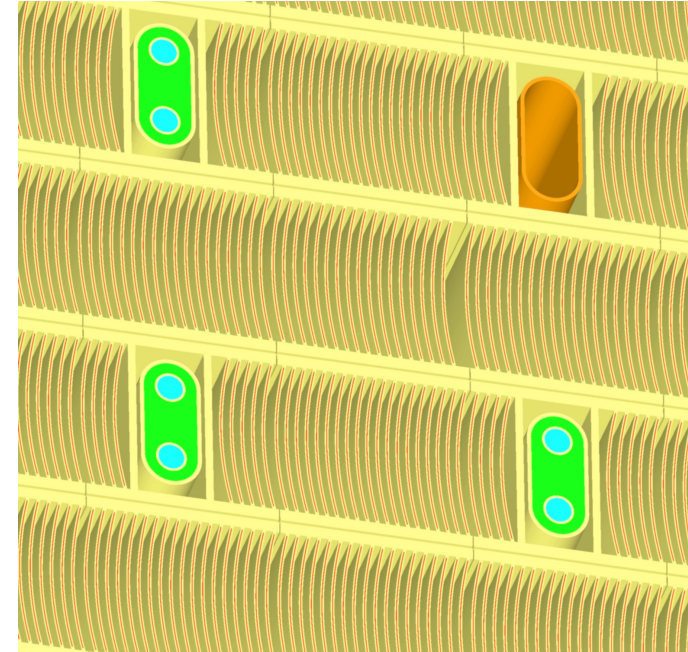
- Algorithms designed for a time when 1000 surfaces was “a lot”
- Some of these scale as $O(N_{Surf}^2)$ and a few as $O(N_{Surf}^2) * N_{Cell}$
 - Surface label translation – $O(N_{Surf}^2)$ search to convert back and forth between user provided labels and internal indices
 - Duplicate surface removal - $O(N_{Surf}^2)$ search through surface labels
 - Cell complexity reduction - $O(N_{Surf}^2) * N_{Cell}$
 - Duplicate lattice identification – $O(N_{Lattices}^2)$
- Geometry is central to all other functions making correctness essential
- The code is complex and has not been touched in over 20 years compounding the difficulty
- Code was restructured with new data structures and lower complexity algorithms



Deimos model from Brain, Cutler, Rising, Stolte, and Terricabras,
“Challenges in Explicit Modeling of TRISO in MCNP”, LA-UR-25-26827

Restructured Code Yields More Than 60× CSG Initialization Time Speedup on Large Geometries

Model	Cells	Surfaces	6.3 (s)	6.4 (s)	Speedup
Deimos	227	3.5k	1.2	0.23	5.2
PCA	1k	5.4k	18.8	2.3	8.2
Model A	7.8k	13k	12.5	3.6	3.5
Model B	37k	57k	994	52	19.1
Model C	70k	108k	3620	64	56.6
Model D	107k	166k	8460	134	63.3



Pool Critical Assembly model from Kulesza, “Oak Ridge National Laboratory Pool Critical Assembly MCNP6 Criticality Calculation Input File”, doi: 10.2172/1601379

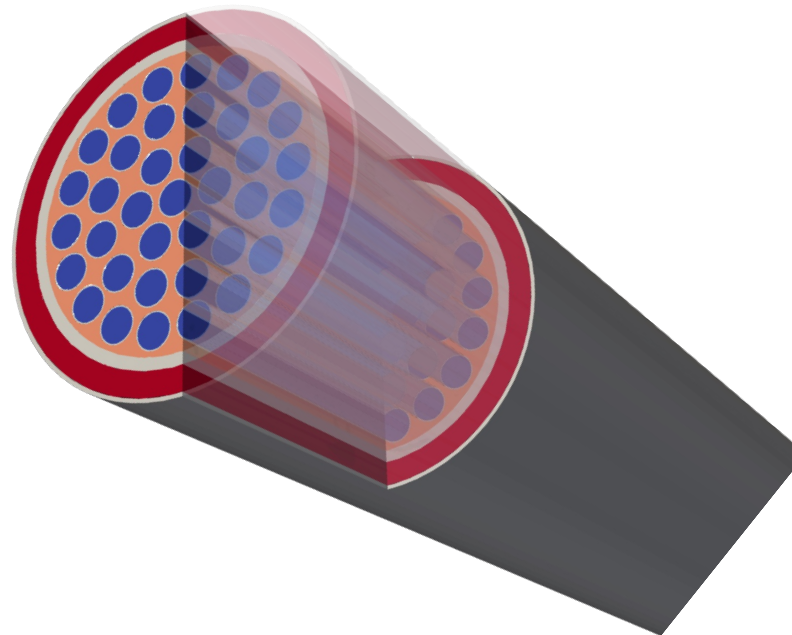
Part 2: Unstructured Mesh



Male phantom from ICRP Publication 145 (Adult
Mesh-type Reference Computational Phantoms),
doi: 10.1177/0146645319893605

Unstructured Mesh Implementation in MCNP Has High Memory Usage for Parallel Runs

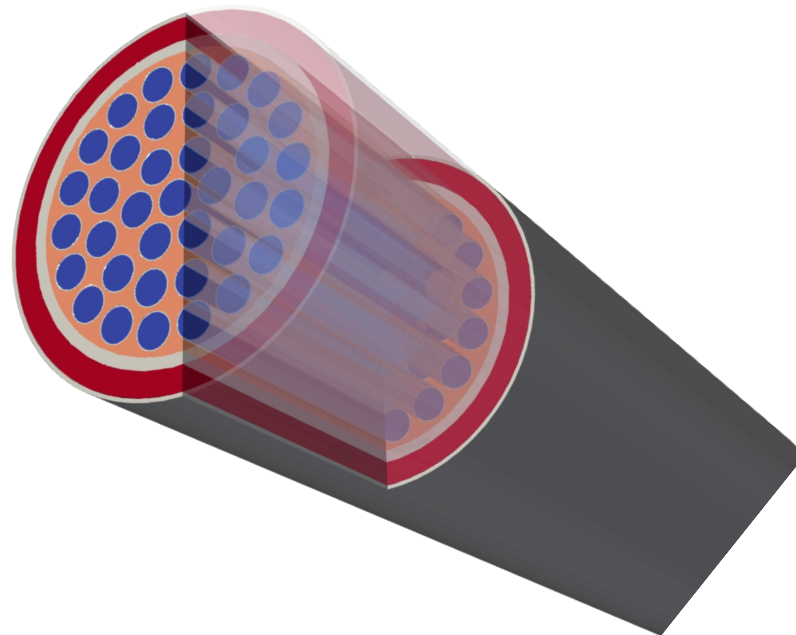
- Two prime offenders identified
 - Memory for parts is allocated based on largest part
 - Several input arrays are duplicated across all ranks
 - Initialization in parallel can quickly exhaust all available memory



CANDU model from Esteban, Armstrong and
Tutt, "MCNP6.3 Unstructured Mesh Verification:
GodivR and CANDU Models", LA-UR-22-33091

Unstructured Mesh Implementation in MCNP Has High Memory Usage for Parallel Runs

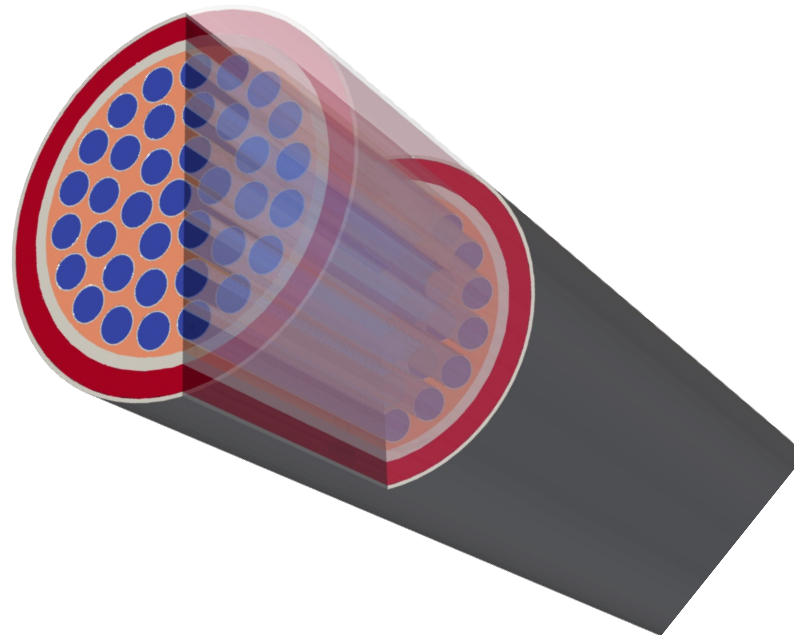
- **Memory for parts is allocated based on largest part**
 - Fixed by enabling per-part allocations
- Several input arrays are duplicated across all ranks
 - Initialization in parallel can quickly exhaust all available memory



CANDU model from Esteban, Armstrong and Tutt, "MCNP6.3 Unstructured Mesh Verification: GodivR and CANDU Models", LA-UR-22-33091

Unstructured Mesh Implementation in MCNP Has High Memory Usage for Parallel Runs

- Memory for parts is allocated based on largest part
 - Fixed by enabling per-part allocations
- **Several input arrays are duplicated across all ranks**
 - **Share memory across ranks on a node to reduce memory footprint**



CANDU model from Esteban, Armstrong and Tutt, "MCNP6.3 Unstructured Mesh Verification: GodivR and CANDU Models", LA-UR-22-33091

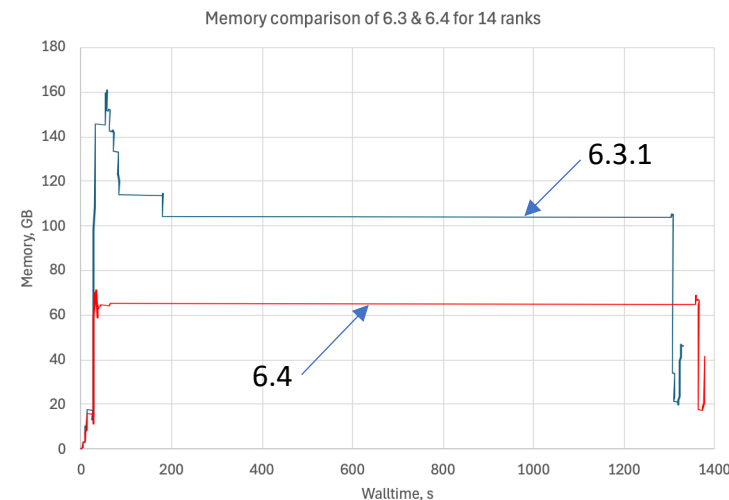
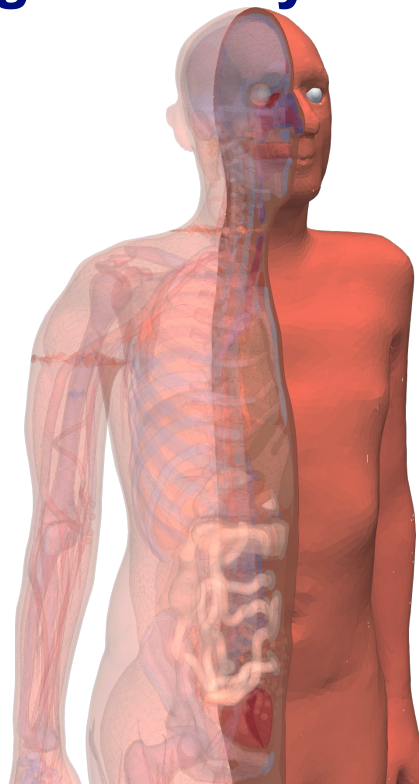
Moving Objects to MPI Shared Memory Windows Eliminates Redundant Copies

- Read-only data can be shared across all ranks on a node using MPI shared memory
- Created SHAMWOW library to simply transition
 - **SHared Memory Window Object Wrapper**
 - Provides a simple interface that enables using shared memory
 - Provides a straightforward Fortran interface
 - No overhead when running in serial
 - Implementation allows None, per-Numa, and per-Node shared memory
- Controlled via new MCNP command line keyword “shmem <location>” where location is *node*, *numa*, or *none*. Default is *node*.

Result: Memory Footprint for Large Unstructured Mesh Problem Dropped Significantly For Large Runs

Ranks	6.3.1 (GB)	6.4 (GB)
1	17.4	15.7
2	17.4	15.7
4	52.9	30.5
8	96.9	46.7
14	160.5	70.9

`srn -n <ranks> --cpu-bind=cores mcnp6.mpi`
`i=MRCP-AM_internal_input shmem node`



Male phantom from ICRP Publication 145 (Adult Mesh-type Reference Computational Phantoms),
doi: 10.1177/0146645319893605

Conclusions

- MCNP6.4 improves initialization time for CSG
 - Improved data structures for streamlined memory management
 - New algorithms for efficient computation
 - Reduced complexity of surface searches
- MCNP6.4 improves MPI initialization memory footprint for Unstructured Mesh
 - Improved data structures for smaller allocations
 - Created *SHAMWOW* library for sharing MPI memory with user control over the domain on which the memory is shared
 - No impact on serial runs

Questions?

- sriram@lanl.gov / mcnp_help@lanl.gov