Title:	Parallel Programming in MCNP6
Author(s):	Armstrong, Jerawan Chudoung
Intended for:	Report
Issued:	2025-09-22

Parallel Programming in MCNP6

Jerawan Armstrong

Introduction

Monte Carlo N-Particle (MCNP)[1] is a general-purpose Monte Carlo particle transport code developed by Los Alamos National Laboratory (LANL). To efficiently handle long simulations, MCNP version 6 (MCNP6) supports parallel execution using two primary programming models:

- **Shared-memory task-based threading using OpenMP (Open Multi-Processing)**
- **Distributed-memory calculations using MPI (Message Passing Interface)**

The OpenMP and MPI programming models enable MCNP6 to scale from desktop systems to high-performance computing (HPC) clusters, allowing users to run MCNP in one of three parallel modes:

- **OpenMP-only**
- **MPI-only**
- **Hybrid (MPI + OpenMP)**

The choice of parallelization mode depends on the underlying computer architecture and the characteristics of the simulation problem.

OpenMP-Only Mode

OpenMP is an Application Programming Interface (API) for shared-memory parallel programming in `C`, `C++`, and `Fortran`. It enables multi-threading, where multiple threads of a single program run in parallel, sharing the same memory space. OpenMP allows developers to write code that executes tasks concurrently across multiple CPU cores within a single machine or a single node of an HPC cluster. Developers identify which parts of the code should be parallelized using OpenMP directive, and the OpenMP runtime takes care of creating and managing threads.

Implementing OpenMP can be more complex than writing serial code because developers must ensure that the code is *thread-safe*, meaning that multiple threads do not simultaneously modify shared memory in a way that leads to errors or data corruption.

To enable OpenMP threading in MCNP6, use the `tasks` option in the execution command:

```
mcnp6 i=input tasks n
```

Where `n` is the number of OpenMP threads to use. Some features in MCNP6 are not OpenMP-threaded. If the input file uses features that are not threaded, MCNP6 will print a warning message and automatically fall back to single-threaded mode (equivalent to `tasks 1`).

OpenMP applications are intended for shared-memory systems and should run within a single compute node. Although MCNP6 may still execute if OpenMP threads are unintentionally distributed across multiple HPC nodes, performance will degrade significantly, causing the simulation to run extremely slowly and inefficiently.

[1]MCNP® and Monte Carlo N-Particle® are registered trademarks owned by Triad National Security, LLC, manager and operator of Los Alamos National Laboratory for the U.S. Department of Energy. Any third party use of such registered marks should be properly attributed to Triad National Security, LLC, including the use of the ® designation as appropriate. Any questions regarding licensing, proper use, and/or proper attribution of Triad National Security, LLC marks should be directed to trademark@lanl.gov. For the purposes of visual clarity, the registered trademark symbol is assumed for all references to MCNP within the remainder of this report.

MPI-Only Mode

MPI is an API for distributed-memory parallel computing. It allows programs running on multiple computers or multiple nodes of an HPC system to communicate and collaborate to solve large computational problems. MPI applications run as multiple processes, each with its own memory space. These processes exchange data using explicit communication, typically over high-speed interconnects in HPC systems.

An MPI parallelization model in MCNP6 is based on a manager-worker design:

- **Rank 0 (Manager)**: Does not perform particle transport calculations. It coordinates the simulation, distributes work, collects partial results from worker ranks, and computes and writes the final outputs.
- **Ranks 1 and higher (Workers)**: Perform the actual particle transport simulations.

To run MCNP in MPI mode, use an MPI launcher such as `mpirun` or `srun`. The following example launches 8 MPI ranks to run MCNP6 in MPI-only mode.

```
mpirun -np 8 mcnp6.mpi i=input
```

Note that the number of MPI ranks must be greater than 2. If you run with exactly 2 MPI ranks, MCNP6 will disable MPI and revert to sequential mode. That is:

```
mpirun -np 2 mcnp6.mpi i=input
```

is equivalent to:

```
mcnp6 i=input
```

When running MCNP using MPI, it uses domain duplication; this means that the geometry and input data are replicated across all MPI worker processes. This can lead to high memory usage, especially for large problems. In such cases, users may need to use fewer MPI ranks than the number of available cores to avoid exceeding memory limits.

Hybrid Mode (MPI+OpenMP)

In hybrid mode, MPI is used for distributing work across multiple nodes or NUMA domains, while OpenMP provides multi-core parallelism within each node or NUMA domain. NUMA stands for Non-Uniform Memory Access. The following example runs MCNP6 in hybrid mode:

```
mpirun -np m mcnp6.mpi i=input tasks n
```

Where `m` is the number of MPI processes (ranks), and `n` is the number of OpenMP threads per MPI rank.

Examples of SLURM Commands

SLURM is a workload manager used to launch jobs on HPC systems. On systems using SLURM, you must configure the appropriate `sbatch` directives in your job script. At LANL, `srun` is used to launch jobs on HPC systems. Requesting fewer resources from SLURM than the number of processes or threads launched by MCNP6 may result in job failure or the simulation hanging due to a resource mismatch. The following are example SLURM job submission scripts for running MCNP on HPC systems:

- OpenMP-only example script: 📎
- MPI-only example script: 📎
- MPI+OpenMP example script: 📎

For LANL users, additional information on running MCNP on HPC systems can be found at: https://ddw-confluence.lanl.gov/spaces/MCPUB/pages/401343029/MCNP+Use+on+HPC+and+ADX+LAN