Multilevel Parallel Strategy on Monte Carlo Particle Transport for the large-scale full-core pin-by-pin simulations

Baoyin Zhang, Gang Li, Wei Wang, Danhua Shangguan, Li Deng
Institute of Applied Physics and Computational Mathematics,
Beijing, P.R. China
zhangby1221@sina.com, li_gang@iapcm.ac.cn, wang_wei@iapcm.ac.cn,
sgdh@iapcm.ac.cn, deng_li@iapcm.ac.cn

Abstract

This paper introduces the Strategy of multilevel hybrid parallelism of JCOGIN Infrastructure on Monte Carlo Particle Transport for the large-scale full-core pin-by-pin simulations. The particle parallelism, domain decomposition parallelism and MPI/OpenMP parallelism are designed and implemented. By the testing, JMCT presents the parallel scalability of JCOGIN, which reaches the parallel efficiency 80% on 120,000 cores for the pin-by-pin computation of the BEAVRS benchmark.

Keywords: Multilevel h**ybrid parallelism, Monte Carlo, Particle transport, OpenMP, JCOGIN**.

1. Introduction

The advantages of the Monte Carlo method for reactor analysis are well known, but the full-core reactor analysis challenges the computational time and the computer memory[1]. Meanwhile, the exponential growth of computer power in the last 10 years is now creating a great opportunity for large scale parallel computing on the Monte Carlo full-core reactor analysis. However, programming adapting to the future computer architecture is becoming a great challenge[1], and will be even more challenging in the future[2].

The programming challenges mainly arise from two types of increasing complexity[3,4]. The first is the computer architecture. The memory wall is still the key bottleneck for realistic performance; the increasing number of cores in each CPU increases the seriousness of this bottleneck. To achieve EXAFLOPS performance, the heterogeneous architecture will be faced, such as MIC, GPU, cell processors.

The second complexity comes from the full-core reactor simulations. An estimate of the number of tallies needed for full-core reactor analysis was also given by Smith[5]. He assumed a reactor core with 70,000 fuel pins with 10 radial and 100 axial meshes per pin, and 300 isotopes to be tracked. Allowing additional tallies per isotope for both absorption and fission reactions and for the variance estimation, Smith estimated 1 TB of memory would be needed, assuming 8 bytes per tally for over 100 billion tallies. This is well beyond the capacity of current compute nodes, which tends to be in tens of GB or less, even for multi-core nodes[1]. The MC21 Monte Carlo code[6] has made big progress for the full-core reactor analysis. However, the traditional particle parallelism is not enough for the full-core reactor analysis. The new hybrid parallelism is necessary for the future Monte Carlo codes.

In this paper, we introduce the Strategy of multilevel hybrid parallelism of JCOGIN Infrastructure on Monte Carlo Particle Transport for the large-scale full-core pin-by-pin simulations. JCOGIN implements the hybrid parallelism of the spatial decomposition and the traditional particle parallelism on MPI and OpenMP to overcome the memory demand and the computational time.

Finally, JMCT code[7-9], which is a Monte Carlo neutron-photon transport code, is developed on JCOGIN. JMCT reaches the parallel efficiency of 80% on 120,000 cores for the pin-by-pin computation of the BEAVRS benchmark, and the number of the cells is up to 25 million, the tallies of the fluxes need over 20GB of memory and the number of particles is up to 240 billion.

2. JCOGIN infrastructure

JCOGIN is the abbreviation of “J COmbinatorial Geometry Monte Carlo transport INfrastructure”. Here, the capital J represents the Institute of Applied Physics and Computational Mathematics (IAPCM). JCOGIN is written primarily in C++ programming language, using the object-oriented feature. Figure 1 depicts the architecture of JCOGIN. Three layers are included.

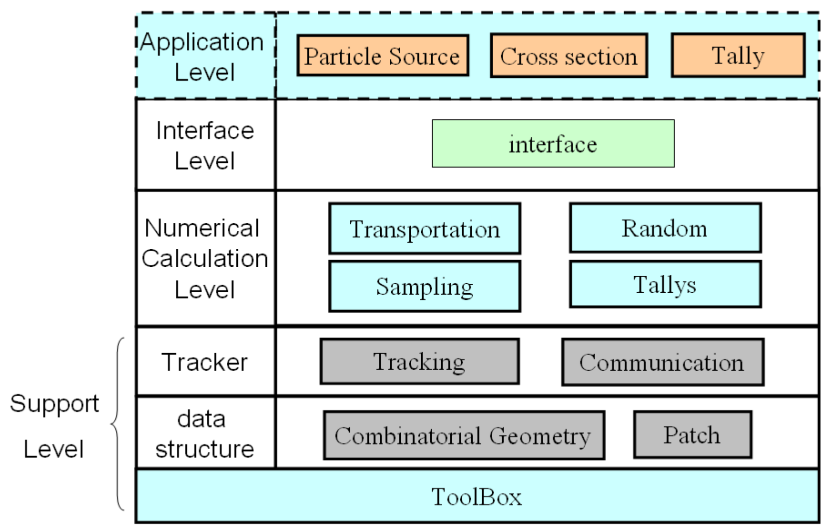

Figure 1 JCOGIN software architecture

The bottom layer mainly consists of modules supporting high performance computing for Monte Carlo transport. In this layer, modules are grouped into three sub-layers. The base sub-layer, Toolbox, contains the essential C++ classes for parameter input, memory management, restarting, and input/output interfaces with the HDF5[10]. The second sub-layer contains two modules, encapsulating the data structures of Combinatorial Geometry and Patch. The top sub-layer, Tracker, contains the components for the particle tracking and communications between the spatial decomposition domains.

The middle layer of JCOGIN contains the modules for the general numerical algorithms including random number generators, sampling for the general distributions, usual tallys, and so on. The top layer is a virtual layer consisting of C++ programming interfaces for Monte Carlo codes. Upon this layer, users can write particle source, cross section, some special tallys, variance-reduction methods, and so on, these subroutines constitute the application code.

Now, JCOGIN is portable for personal computers, high performance clusters, and massively parallel processing computers (MPP). On JCOGIN, after the program is rewritten on a personal computer, it can successfully run on parallel machines where JCOGIN is installed.

3. Multilevel parallel strategy

First, we introduce the datastructure of JCOGIN infrastructure. The datastructure is the base of the multilevel parallel strategy of JCOGIN. Figure 2 depicts the data structure of JCOGIN. “PatchLevel – Patch - Cell” is the kernel data structures of JCOGIN. Every PatchLevel includes the whole geometry information, so cloning the PatchLevels will implement the particle parallelism. Each PatchLevel can be decomposed into multi-Patches in space. So, spatial decomposition is supported by the kernel data structure of JCOGIN.

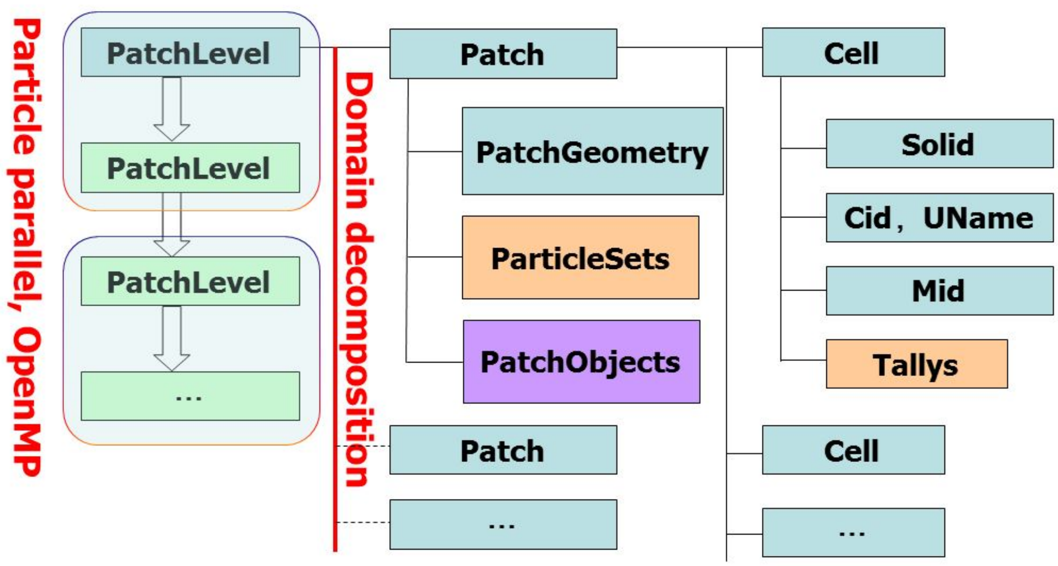

Figure 2. Data structure of JCOGIN

Every Patch includes PatchGeometry, ParticleSets, PatchObjects, and a part of cells. PatchGeometry is responsible for the management of geometry and the geometry computing. ParticleSets are responsible for storing the particles, and the size of the bank in the ParticleSet increases dynamically. PatchObjects are added for supporting multithreading programming with OpenMP. Every Cell includes Solid, Tallys, and so on. Solid is the geometry element of the cell. Tallys are responsible for storing the tallies of user defined.

Second, Spatial decomposition is used to solve the problem of memory overload of the full-core reactor analysis, but its parallel efficiency is lower than the traditional particle parallelism. Further, huge number of particles needs to be sampled to reach good statistics of tally results for pin-by-pin full-core reactor analysis in high-fidelity. So, JCOGIN implements the hybrid parallelism of the spatial decomposition parallelism and the traditional particle parallelism on MPI and OpenMP. These papers[11,12] introduce the spatial decomposition parallelism on JCOGIN in detail. Here, we give some features briefly.

In JCOGIN, geometry element is called solid, such as spheres, rectangles, cylinders, cones, and so on. Complex geometries can be made up by these elements using boolean operators: intersection, union, and complement. If a solid is filled with materials, it is called a cell. All cells are stored in computer memory in the form of tree. Each node on the tree corresponds to a cell. If a node has daughters, the cell corresponding to the node is made up of those cells corresponding to daughters. The root node stands for the whole geometry world.

In JCOGIN, the geometry domain is decomposed by dividing the whole tree into many subtrees. All the cells keep the shapes after the spatial decomposition. It is different from Mercury code[13], in which some cells are divided into parts by the added surfaces. A few cells are duplicated in all domains. In fact, there are usually several shells outside a reactor core and their volumes are very big. Further, the number of these cells is usually small, and they are duplicated in all domains, just like domain replication. Each domain can track particles in the shared cells.

Finally, in JCOGIN, the communication efficiency has been considered. The asynchronous communication is chosen for exchanging particles between processors, and it overlaps the particle communications and the transport computations of the particles. Figure 3 illustrates the communication topology of the multilevel parallel strategy.

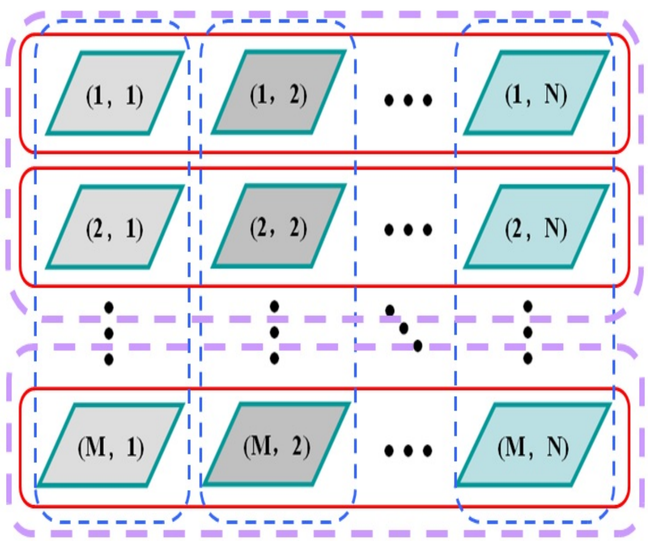

Figure 3. The communication topology.

4. JMCT and test of BEAVRS

JMCT is a combinatorial geometry Monte Carlo particle transport code developed on JCOGIN, which has been developed at the Institute of Applied Physics and Computational Mathematics over 30 person-years. It can do transport of neutrons and photons, including both multi-group and continuous energy treatments of cross-section.

A model of BEAVRS (Benchmark for Evaluation And Validation of Reactor Simulations) is constructed in JMCT. In the axial direction a fuel rod is divided into 398 segments. So the number of fuel cells reaches up to 22 million. First, we test the parallel efficiency of the multilevel parallelism. Spatial decomposition is by 2*2*2, and 8 domains are divided. Figure 4 illustrates the parallel efficiency of the large-scale computation. JMCT reaches the parallel efficiency of 80% on 120,000 cores for the pin-by-pin computation of the BEAVRS benchmark. Second, the flux of hot zero power is simulated. The tallies of the fluxes need over 20GB of memory. Testing runs 200 cycles with 120,000,000 particles in each cycle and the total number of particles is up to 240 billion. Figure 5 gives the hot zero power results of BEAVRS model.

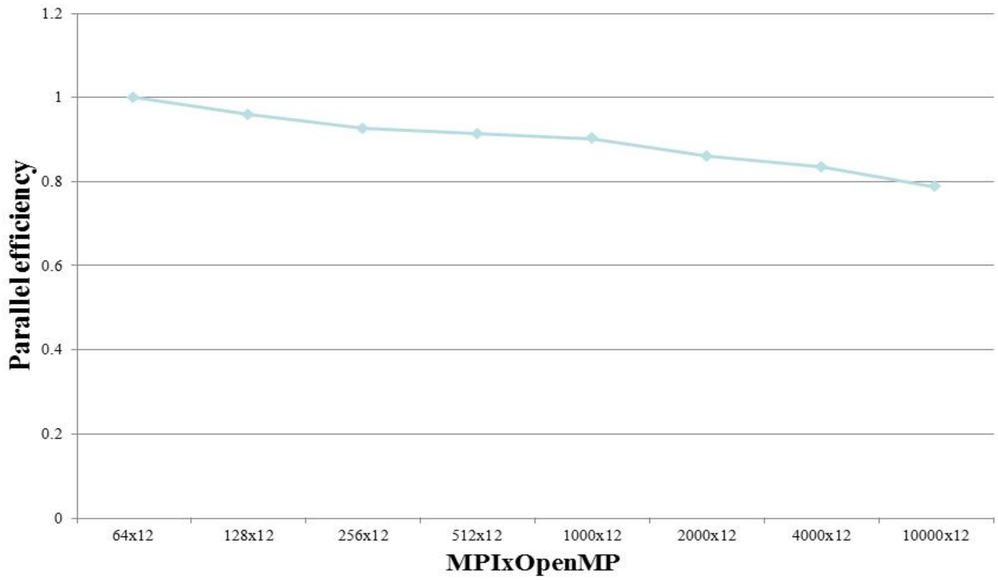

Figure 4. Efficiency of parallel computing

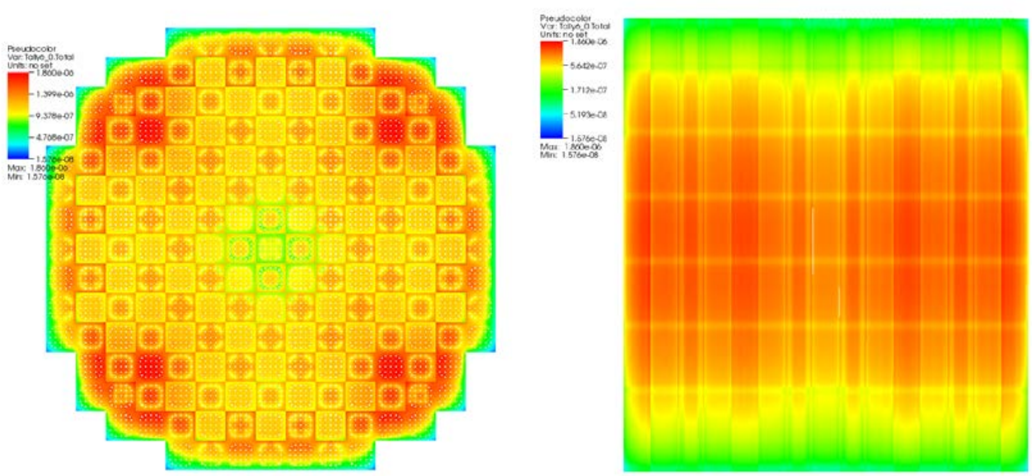

Figure 5. Flux results view of the BEAVRS model.

5. Conclusion

We introduce the Strategy of Multilevel hybrid parallelism of JCOGIN Infrastructure on Monte Carlo Particle Transport for the large-scale full-core pin-by-pin simulations. JCOGIN implements the hybrid parallelism of the spatial decomposition and the traditional particle parallelism on MPI and OpenMP to overcome the memory demand and the computational time.

JMCT code, which is a Monte Carlo neutron-photon transport code, is developed on JCOGIN. JMCT reaches the parallel efficiency of 80% on 120,000 cores for the pin-by-pin computation of the BEAVRS benchmark, and the number of the cells is up to 25 million, the tallies of the fluxes need over 20GB of memory and the number of particles is up to 240 billion.

6. References

[1] W. Martin, “Challenges and Prospects for Whole-Core Monte Carlo Analysis,” Nuclear Engineering and Technology, Vol. 44, No. 2, pp. 151-160, March 2012.
[2] V. Sarkar, W. Harrod W, A.Z. Snavelg, “Software challenges in extreme scale systems”. Journal of Physics: Conference Series, 2009, 180(1):012045.
[3] S. Zhu, “A brief report on scientific computing”, Physics, 2009, 38(8): 545–551 (in Chinese)
[4] Z. Mo, W. Pei, “Scientific computing application codes”, Physics, 2009, 38(8): 552–558 (in Chinese)
[5] K. Smith “Monte Carlo for Practical LWR Analysis: what’s needed to get to the goal?”, presented at the M&C Division Computational Roundtable, American Nuclear Society Summer Meeting, Hollywood, FL, June 27, 2011.
[6] D. J. Kelly, T. M. Sutton, T. H. Trumbull, and P. S. Dobreff, "MC21 Monte Carlo Analysis of the Hoogenboom-Martin Full-Core PWR Benchmark Problem," Proc. ANS Reactor Physics Division Topical Meeting, PHYSOR10, Pittsburgh, May 2010.
[7] L. Deng, G. Li, B. Zhang et al. “Simulation of the Full-core pin-model by JMCT Monte Carlo Neutron-Photon Transport Code”, M&C 2013, Sun Valley, Idaho, May 5-9, pp.427-433(2013)
[8] G. Li, B. Zhang, L. Deng, et al, “Development of Monte Carlo particle transport code JMCT, ”*High Power Laser and Particle Beams*, 25(1), pp:158-162 (2013)
[9] D. ShangGuan, G. Li, L. Deng, B. Zhang, “Tallying Scheme of JMCT—A General Purpose Monte Carlo Particle Transport Code”, ANS Transactions, 109, 1428-1430 (2013)
[10] HDF5. http://www.hdfgroup.org/HDF5/, Jan. 16, 2014
[11] G. Li, B. Zhang, L. Deng et al. Combinatorial Geometry Domain Decomposition Strategies for Monte Carlo Simulations, *M&C 2013*, Sun Valley, Idaho, May 5-9, pp.434-443(2013)
[12] G. Li, B. Zhang, L. Deng, “Domain Decomposition of Combinatorial Geometry Monte Carlo Transport Code JMCT”, ANS Transactions, 109, pp.1425-1427(2013)
[13] R. Procassini, D. Cullen, G. Greenman, et al, “New capabilities in MERCURY: a modern, Monte Carlo particle transport code,” *M&C+SNA*, Monterey California, April 15-19(2007).
[14] Z. Mo, A. Zhang, X. Cao, et al. “JASMIN: a parallel software infrastructure for scientific computting”. Front. Comput. Sci. China, 2010, 4(4): 480-488.