Chapter

Parallel Algorithm Analysis in Reactor Core Calculation

Pingzhou Ming

Abstract

The reactor core system consists of many materials, involving multi-physics processes, and can be analyzed and simulated at multi-scales. With the evolution of cluster computer, traditional programs and models could be translated into new program structure and modified in detail, so that more complex problems can be solved. Based on existing theory, programs of sub-channels analysis, two-dimensional (2D) method of characteristic (MOC), fuel temperature approximation, and three-dimensional (3D) discrete ordinate method (SN) are developed and analyzed. The different approach is that the reusable software structure of core calculation is established, with more well-defined storage of nuclear data, control layers, and more effective parallel algorithm for computation. The features of parallel algorithm for these programs are listed succinctly in the discussion. Additionally, the corresponding testing on parallel algorithm and computing results are given.

Keywords: reactor core, modeling and simulation, nuclear data, software structure, parallelism

1. Introduction

There are various heterogeneous phenomena in reactor core, which are related to multi-physics and multi-scales modeling and simulation [1]. The core program or software is the important approximation for the reactor core; therefore, mathematics models, numerical algorithms, preprocessing, post-processing, and visualization are sometimes managed in the similar software structure [2]. Based on the existing conditions of calculation, high-performance-computing technology is an important mean to accomplish the simulation of core programs by the cluster computer. For instance, there are several supercomputers (cluster) in ANL and ORNL that provide high-capacity resources for reactor engineering calculation and nuclear-related simulation [3]. Once the numerical algorithm is fixed, the time-to-solution becomes the necessary consideration during the research, and the approach pattern can be summed up in three steps [4] in this field:

1. Develop the single discipline program or multidisciplinary coupling code for the reactor core, which enable higher accuracy.

2. Propose and implement parallel algorithm for the program.

3. Use parallel hardware to run and carry out numerical experiments within acceptable requirements, and then match the reference solution or experimental data.

According to the modeling and simulation for reactor and the features of multi-physics and multi-scale models, the reactor calculation is classified into core calculation and out-core calculation [5]. The core calculation covers considerable subjects, for instance, reactor physics, thermal-hydraulics, dynamics, and fuel performance, for example. For the purpose of research and engineering, different discrete systems and solvers are implemented so as to reveal different phenomena in the core [6–8]. As an example, the DENOVO transport software of ORNL utilizes 160,000 processors to run the benchmark core example in Jaguar supercomputer; up to now the computer has been upgraded to TITAN [9].

On the one hand, the simulation could be accelerated through effective parallelization in different models [10]. On the other hand, the parallel computing makes researchers focus on more research fields except mathematics and physics [11], which constructs one view from nuclear data to top-level application. With the attempt of the M&S content, the core software could be ascribed as the expression and translation of the set of state variables and data structure [12]. Thus, the conjunction of the core problem and parallel computing is from the new requirements of application; furthermore it is driven by the underlying hardware innovation. The Berkeley Parallel Computing Laboratory reviews parallel computing and points out that the parallel elements of different algorithms are the methodology of application driven and reuse. Since various features exist in different layers of software, the parallel algorithm should be designed specifically so as to adapt the parallel hardware [13]. The chapter describes and explains the software structure, underlying nuclear data, and parallel algorithm from a systematic view in the domain of reactor core calculation. Section 2 abstracts the software structure for core calculation. Then, Section 3 explains the basic elements of nuclear data and its association according to the software structure. After that, Section 4 lists the concept and steps of parallel algorithm analysis for core problem in practice. Sample programs are analyzed and discussed concisely in Section 4, and conclusions are given in Section 5.

2. Software structure

As the complex system, the reactor core could be understood by various parts, such as concept models, numerical methods, model uncertainty, and model reuse. Algorithms and software are the center of digital simulation so that the calculation rule is regulated by the program technologies. Sometimes the fixed software structure could be abstracted, and common algorithms could be extracted to be close to the trend of reactor core simulation, which forms one united manner (also known as framework method [14]).

According to the description [12], one general software structure could be divided into seven layers, which covers all aspects from the application to the computer operating system by **Table 1**.

Then, the numerical calculation system is abstracted into multiple layers, which are implemented through controls, modularity, and dynamic interaction. So participants from different professional backgrounds pay more attention to their own business [15]. The core calculation learns from the above contents, and the software structure could be simplified in **Table 2**.

It is effective to develop numerical software based on reusable components or framework, so the discrete model could be modeled through fixed process. On the one hand, the usage leads to reuse. On the other hand, new programming pattern could be merged. For instance, with the help of the underlying framework, MC neutron transport combines the discrete model and algorithm analysis process and

Abstraction layer	Function
Model layer	The model code
Model framework layer	Collection of models for a single application area
Simulator layer	Provides the paradigm for simulation and synchronization usage
Component federation layer	Provides interface code to create one model with multiple sub-models
Load management layer	Within one parallel model execution, collects run information
Ensemble layer	Runs instances in a job and handles scheduling
Operating system/job scheduler layer	Runs independent jobs in parallel

Table 1.
General software structure.

Abstraction layer	Function	Examples
Numerical framework	Provides basic arithmetic function and express data	PETSc, Trilinos, HDF5 NetCDF, BLAS, MPI
Discrete model	The single component with the properties of time, spacing, parallelism, and synchronization	Structured grid Unstructured grid Discrete timeline
Component combination	Provides interface code to create model of multiple components	Geometry control Cross-sectional database Neutron transport solver
Algorithm analysis	Schedule the model in parallel, and process the run data and obtain running information	TAU VTK

Table 2.
Software structure for core calculation.

then uses numerical components and structure to accomplish the transport task easily in the reactor core [16]. Since this integrated software structure method restricts the programming manner of the core calculation and provides reuse mechanism, then in practical, the prototype software NAC4R is designed and constructed that contains basic data operation, basic algebra operation and linear system solver, and so on. As an example of the above data, NAC4R can be used for the parallel analysis process of sample programs that explains the effectiveness and efficiency of the corresponding software structure.

3. Nuclear data process

The underlying nuclear data is the basis of core calculation and also the fundamental elements in the proposed software structure. The reactor physics data is divided into four categories by A. Trkov, namely, (1) basic nuclear data, (2) evaluated nuclear data files, (3) processed nuclear data, and (4) averaged nuclear data [17]. They are the similar data in the thermal-hydraulics analysis, which correspond to the physical property information of the fluid, such as the light water property data in the core calculation [18]. These different types of data involve representation, storage, and computer algorithms. For example, the cross-sectional library of deterministic reactor physics calculation needs group-wise data, such as few-group homogenized cross section in diffusion calculation, with the affection of multiple state variables that include temperature, density, and so on [19]. The class of used nuclear data can be directly computed in the core calculation or be generated in advance. The pre-generated nuclear data (database) is commonly used in a two-step method of core calculation, one is multidimensional interpolation table, the other is parameterized library, and they have different effects on computation and parallel algorithm [20].

3.1 Multidimensional interpolation table

The table is the indirect format of cross-sectional parameters, with the grids inside the value range for each state variable. The core calculation uses interpolation algorithm to obtain the nuclear data.

Assume there is only one state variable ρ (such as coolant density), and its value range is $[a, b]$, then the grid of a single state variable is divided as in Eq. (1):

$$\{\rho_i\}_{i=1}^{N+1}, \rho_i \in [a, b] \tag{1}$$

where ρ_i is not necessarily uniform distribution. The corresponding parameters at each discrete grid can be obtained by the lattice program as follows in Eq. (2):

$$\left\{\sum_i\right\}_{i=1}^{N+1}, \sum_i := \sum(\rho_i) \tag{2}$$

In this way, the continuous parameter values can be calculated by polynomial interpolation method in the range of state variable. The coefficients of interpolation polynomials are stored in the table:

$$\tilde{\sum}(\rho) = P(\rho) = \sum_{i=0}^{N+1} a_i \rho^{i-1} \tag{3}$$

3.2 Parameterized library

The library method is the complex function model of nuclear data that is related to various state variables. To build an accurate model, many factors need to be considered, and the range of each state variable is also defined separately:

$$\sigma = f(\rho, T, P, \dots) \tag{4}$$

Generally, the tabulation method uses space for time, and the method is relatively simple, but the storage mode of nuclear data has a greater impact on the parallel algorithm of the core calculation. The method consumes more computing time, but the data storage is smaller, and it is easy to improve and modify the model so that there is less impact on the parallel algorithm.

The software implementation of the nuclear data corresponds to the model layer and model framework layer mentioned in Section 2. **Figure 1** shows two abstract forms of the nuclear data. The left side that represents the processing of each kind of nuclear data is independent, and the right side explains the design of consistent data attributes, and formats are independent underneath and dependent on top in the fixed software structure so that each type of nuclear data only needs format conversion and minor programming. This understanding of the underlying nuclear database focuses on software reuse and performance, which can fully reuse various

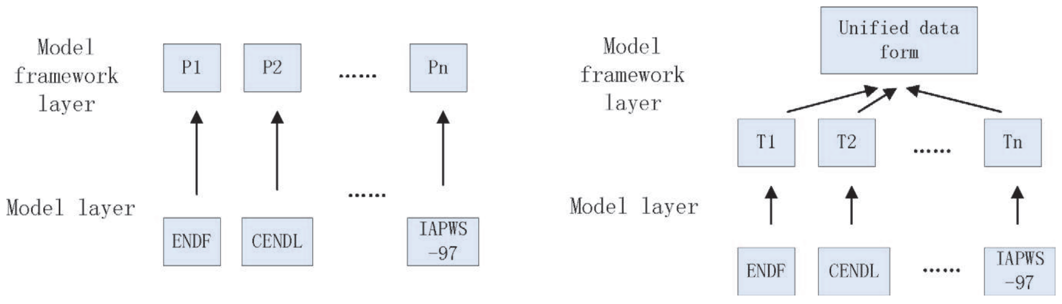

Figure 1.
The association between underlying nuclear data and software structure.

known nuclear data that have fine readability and correctness. This association also enables independent design and programming of core calculation at all levels, which makes the following parallel algorithm analysis match it.

4. Parallel algorithm analysis

Amdahl law stipulates the speedup ratio limit for parallelism, and Gustafson law quantifies the acceleration effect [21]. To be equivalent to Amdahl and Gustafson, measurability of calculation models the performance. Assume that the running time of one computing problem is $T(N, x)$, which represents the running time for the scale of problem x. Then the running efficiency could be utilized to prove the effectiveness of parallel core program from two categories.

4.1 Strong measurability

The change of computing time varies with the increase of processor cores, also is known as speedup ratio:

$$S_{strong} = \frac{T(1, x)}{T(N, x)} \tag{5}$$

4.2 Weak measurability

The problem size of each processor or process is fixed, and then the statistic of the running time varies as the processors change:

$$S_{weak} = \frac{T(1, x)}{T(N, x \cdot N)} \tag{6}$$

The deterministic calculation of the reactor core mainly makes the $S_{strong,N}$ as the variation of processor cores. In practical deterministic core, computation has features that the problem scale is related to the physical form. For instance, when the characteristic rays are fixed on every processor, the MOC transport would increase new characteristic rays and achieve new arrangement if the processors are added. The data expression and computational content may change with different program patterns. On the contrary, nondeterministic calculation of reactor core, such as the MC method, the computational features would remain unchanged so that the problem takes more emphasis on the statistics of weak measurability.

The steps proposed by Ian Foster is the most typical method for the parallel algorithm design and programming [22] (also known as PCAM method).

	Designed and implemented algorithm	Algorithm is not designed and implemented
Working step	S1: executes and obtains the statistical parameters of parallel performance on a fixed running platform S2: judges whether the requirements are met. If not, go to step 3. Otherwise, ends the analysis S3: designs or improves the existing parallel algorithm and then goes to step 1	S1: executes and obtains the statistical parameters of parallel performance on a fixed running platform S2: identifies the parallel features of the algorithm S3: determines whether the features have reusable patterns in the software structure (here refers to NAC4R). If so, reuses them directly, turns to step 5, if not, turns to step 4 S4: the new parallel algorithm is designed and programmed S5: runs on the fixed platform and obtains the statistical parameters S6: judges whether the requirements are met. If not, goes to step 3. Otherwise, ends the analysis and merges the new parallel algorithm into NAC4R as a reusable pattern
Key issues	1. The performance is closely related to the problem 2. The features depend on the abstraction of actual core numerical calculation that is often distinguished by the knowledge and experience of developers	

Table 3.
Two different analysis steps.

1. Partition identifies computing content that can be executed in parallel.

2. Communication determines the communication data of each parallel task.

3. Aggregation merges and adjusts parallel tasks and communication.

4. Mapping assigns the aggregated task to entities on which actual parallel programming techniques depend (process or thread).

At present, the cluster computers are often in heterogeneous development environment, and parallel algorithms are divided into many kinds such as pipeline parallelism, mater-slave parallelism, and so on. To facilitate engineering specifications, the usually analysis steps of parallel algorithm are divided into two cases in **Table 3**.

It can be concluded that the parallel analysis includes interpretation, design, and performance verification. The parallel analysis in reactor core calculation summarized here emphasizes to meet the application requirements, and then the common features of the algorithm are usually abstracted and recorded concisely, so that the analysis can be understood clearly and simply.

5. Samples research

Interconnections exist in the multi-physics process and software structure in reactor core. For solving according to physics-mathematics model, various core software has been developed. There are two main ideas for developing these procedures. One is to make full use of the existing programs in the world. The other is to redesign and program the program structure and function. The key parallel algorithms of single discipline sample program CORTH, KYLIN2, K-MOD, and Hydra are identified and programmed, which come from the typical examples of the reactor core system, computing work through the research career of the author.

5.1 Sub-channels analysis

The CORTH program utilizes the uniform flow model of four equations to process two-phase flow in the reactor core. The energy conservative equation could be abbreviated as the manner of structured grids, which is easy to use in finite difference and form the linear system. For the description, CORTH could be divided as components in **Table 4**.

The features of the parallel algorithm are listed summarily to make the program refinement practical.

1. The reactor core is divided along the radial direction so that each process stores the data of some sub-channels. The overlap method could also be used to reduce data communication.

2. The solution of linear system solving can be quick in the model. The coefficient matrix of the conservative system is symmetric, and it can be constructed simultaneously together with the right-hand vector of the system.

3. The PETSc and Aztec parallel solver can be ported and optimized inside NAC4R so that solvers could be used and hybrid programmed directly and easily.

The bottleneck of the parallel algorithm design is related to the components of the coefficient update, linear system solve, and mesh matching. Parallel scheme was realized in NAC4R with the features mentioned above. The ACP1000 model (177 fuel assemblies, there are 264 fuels and 25 guide tubes in each assembly, axial segments are 54, and there are 50868 sub-channels in radial direction) is used for performance validation. Cluster processors are chosen as 4, 6, 18, 27, 36, 54, and 108, which is the divisor of 50868 as the Aztec solver needs that the number of processes matches the tasks. The expected performance of strong measurability is improved by more than five times.

Performance conclusion: The trend of processor change shows that using about 30 processors, it has met the expectation in **Figure 2**, with the speedup ration about eight times. Not only the conservation system can be constructed in parallel, but the Aztec solver can also solve the matrix with the size of 50868 efficiently.

5.2 2D MOC

The KYLIN2 program carries out the 2D neutron transport calculation by MOC over the assembly region. In the condition of steady situation, multigroup neutron

Components	Model equation	Function
Preprocessing		Reads external input and translates into data structure in memory
Pre-computation		Calculates some parameters and provides initial values for iteration
Iterative control		Controls and manages cyclic calculation
Coefficient update	$f(G, H, \dots)$	Updates parameters of thermal-hydraulics
Linear system solve	$x = A^{-1}b$	Defines linear system and its computation
Mesh matching	$m_{i,j,k} = m_{i1,j1,k1} \cup \dots \cup m_{iN,jN,kN}$	Maps the sub-channel, meshes, and processes

Table 4.
Abstract components in CORTH.

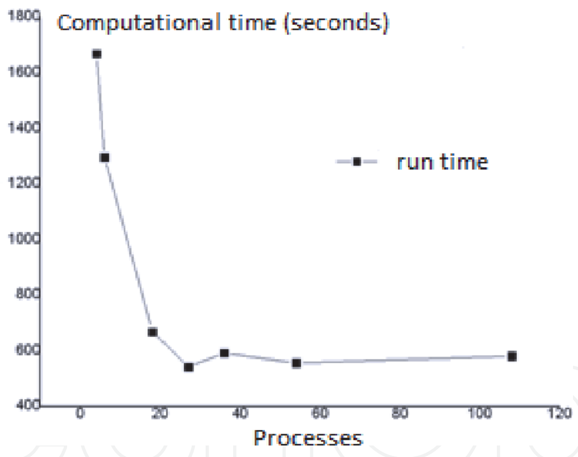

Figure 2.
Time statistics of model ACP1000 for CORTH.

transport equation could be expressed by the ODE along the characteristic ray. We can obtain the direct analytical solution of the ODE as in Eq. (7):

$$\psi_{out,m,i,k} = \psi_{in,m,i,k} e^{-\sum_i S_{m,i,k}} + \frac{Q_i}{\sum_i}\left(1 - e^{-\sum_i S_{m,i,k}}\right) \quad (7)$$

where $\sum_i$ and Q_i denote the macro-transport cross section and the neutron source, respectively. The length $S_{m,i,k}$ is the segment that is truncated with the position m-th azimuth angle, k-th characteristic ray, and the i-th mesh. When source iterative method is used, the iterative procedure will repeat the above calculation process after assigning initial guess values to the flux. As shown in **Table 5**, sample program KYLIN2 could be abstracted into different parts.

Sweep computation follows the loop structure, with each characteristic ray, neutron groups, azimuth angles, and polar angles. Moreover, the intermediate solution of KYLIN2 is related to the angular direction so that the structure can be described as follows.

```
set current←0
set mesh_angular_flux←0
for (g, ray, p) in (Groups, Rays, Polars):
    current ← computeForwardCurrent(g, ray, p)
    mesh_angular_flux ← sweepForward(g, ray, p, current)
    current ← computeBackwardCurrent(g, ray, p)
    mesh_angular_flux ← sweepBackward(g, ray, p, current)
end for
```

Components	Model equation	Function
Geometry description		Discrete result of 2D geometry
Nuclear data	$(\sum_i)$	Stores cross-sectional data with meshes
Characteristic rays	$(S_{m,i,k})$	Generates and manages rays
Sweep computation	$\overline{\psi}_{m,i,k} \leftarrow f\left(\alpha, e^{\beta}\right)$	Traverses rays and solves angular neutron flux
Source update	$Q_i \leftarrow f\left(\sum_i, \overline{\psi}_{m,i,k}, \omega\right)$	Updates the transport source
Task schedule		Controls the computational flow, such as BSP parallel model

Table 5.
Abstract components in KYLIN2.

The above components are redeveloped in NAC4R, and the features of the algorithm are listed here:

1. Iterate that each ray could be in parallel if some critical operations are added, so both MPI and OpenMP can be used for the sweep process.

2. If rays are non-modular arranged, the differences of working load of each ray are not obvious. Since the assembly problem sometimes uses the nuclear data with some energy groups, parallelism on neutron groups could be a better choice, and OpenMP can avoid data communication during the sweep procedure.

3. The callings of exponential function will affect the efficiency.

4. In order to adapt coarse mesh acceleration, reserved interfaces should be kept inside NAC4R so that parallel linear system solvers are used.

Here OpenMP guide sentences are utilized to implement parallelism based on loop structure. The critical operation exists in the cumulative sum of forward and backward sweep when each ray is accessed. C5G7-MOX assembly and ACP1000-UO2 assembly examples are used to test, and more than five times better performance are expected. The statistical calculation time is shown in **Table 6**.

Performance conclusion: OpenMP parallelism (multi-threads) can achieve the rational accelerated effect and meet the expectation. There are more speedup ratios for the ACP1000-UO2 as the neutron group parallelism strategy is used, and some fluctuation exists in the performance gain of shared memory parallelism since there are some management overheads by multi-threads.

5.3 Fuel temperature approximation

The calculation of the fuel temperature usually involves the temperature of the outer surface and the center, and then the effective temperature of the fuel could be obtained for rod-like geometry. The geometry is divided into different sections to solve the heat conduction equation in radial direction, with different approximations are taken account. The fuel rod of PWR is usually delimited as three sections in some programs, and the 1D conduction equation is constructed at each area. After knowing the boundary temperature, the Rowland's equation could be used to compute the effective temperature.

Fuel temperature approximation is necessary for the reactor core coupling calculation, but reasonable boundaries come from other discipline procedure. K-MOD coupling program has been realized, and according to the above description, fuel temperature approximation could be abstracted in **Table 7**.

Thermal feedback controller carries out the function of management and as an external module that will be called repeatedly in K-MOD. The features of the parallel algorithm are less obvious and can be listed in two points: (1) the computing

Example	One core	Four cores	Eight cores	Speedup ratio
C5G7-MOX	1655.28 s	562.20 s	295.01 s	5.61
ACP1000-UO2	19515.38 s	5091.92 s	2581.34 s	7.56

Table 6.
Parameters and time results of two assembly examples.

Components	Model equation	Function
Thermal feedback controller	$f\left(T_{coolant}, T_{fuel}, \phi\right)$	Main control component that processes computation in each discrete time step
Fuel temperature calculation	$T_{fuel} \leftarrow f(T_{coolant}, P, G, t)$	Solves the conduction equation by iterative method and obtains the temperature
Cross-sectional update interface	$\Sigma \leftarrow f\left(T_{fuel}, T_{coolant}, \phi\right)$	Data exchange interface for the temperature and cross section

Table 7.
Abstract components in K-MOD.

granularity is small and the single cyclic time is short and (2) the computing pattern could be abstracted and reused if lower level parallel method is researched; thus, SIMD technology is suitable for programming.

In the sample program K-MOD, fuel temperature calculation could be abstracted into three classes, such as op_basic represents the basic arithmetic, but op_math stands for the compound operation, such as square, logarithm, and so forth.

vector<int> a;
vector<int> b;
$c \leftarrow op(a, b)$
	vector<float> a;
	vector<float> b;
	$c \leftarrow op\_basic(a, b)$
		vector<float> a;
		vector<float> b;
		$c \leftarrow op\_math(a, b)$

compile option	icpc -O0 -march = native -mno-sse	
compile option with SIMD	icpc -O0 -xsse2	

The former deals with integer data, while the latter two deal with floating-point operations. The problem of rod inserting and lifting in real PWR core is tested and compared. The similar imitated structure solves 50,000 times the same as the K-MOD program, with the SIMD hand optimization, which is based on data structure named __m128i and __m128. The time results are shown in **Table 8**.

Performance conclusion: The statistical results of multiple runs show that SIMD scheme has performance bonus for the third class of calculation content; however, other classes get no gain. Looking up Intel-related documents, it is found that SIMD programming depends on the specific hardware and instruction set usage and depends on the compiler behavior, which is not suitable for manual SIMD rewriting with care in practice.

5.4 3D SN

The sweep operation of discrete ordinate method iterates every angle direction in each octant, which has famous KBA parallel sweep algorithm for structured grids. 3D SN method translates Boltzmann transport equation into the matrix system of flux moment as in Eqs. (8) and (9) in the sample program Hydra:

	First class	Second class	Third class
Serial	93 ms	89 ms	367 ms
SIMD parallelism	112 ms	135 ms	152 ms

Table 8.
Time results of three classes on parallel contents.

Components	Model equation	Function
Geometry storage		Manages structured meshes in XYZ coordinate system
Nuclear data	$(\sum_i)$	Records cross-sectional information
Operator process	(H, Γ)	Constructs mapping between the operator and other parameters
Sweep computation	$\phi \leftarrow H^{-1}Q$	Matrix operation by loop structures
Source update	$Q, S \leftarrow \Gamma\phi$	Updates the transport source

Table 9.
Abstract components in Hydra.

$$L\psi = MS\phi + \overline{Q} \tag{8}$$

$$\phi = D\psi \tag{9}$$

where $\overline{Q}$ and L denote the source term and the difference operator, respectively. Discrete operator of flux moment matrix M need to multiply the cross-sectional data, such as the scattering matrix S. The total meshes are (I, J, K) in each angle direction, and then every section (I_a, J_b, K) that are divided could be mapped into corresponding processes in parallel. According to the above description, SN transport calculation could be abstracted as in **Table 9**.

There are multiple sweeping levels in the SN method, and the section (I_a, J_b, K) could be represented by the loop structure in every octant, and the sweeping procedure can be described as follows:

```
I,J,K←0,1,2
  for k in K:
$\phi_{up,j} \leftarrow \phi_{in}$
  for (i, j) in $(I_a, J_b)$:
$\phi_{psi} \leftarrow \phi_{up}$
  call $T(i, j, k, \phi_{psi})$
$\phi_{up} \leftarrow \phi_{psi}$
  end for
$\phi_{out} \leftarrow \phi_{up,j}$
  end for
```

Since there is a lack of parallel programming for this loop structure in Hydra, the features of the algorithm are listed.

1. The extrapolation model, such as the diamond weight difference model, has an independent subroutine T, which is only related to mesh (i, j, k). Flux moment variables ϕ_{up} and ϕ_{psi} depend on the calculation order, so the sweep is one pipeline structure on the (I_a, J_b).

2. When the scale of loop structure is small, shared memory parallelism is fit for pipeline structure to improve efficiency, which has theoretical gain $\frac{ij}{i+j-1}$.

According to the identified features, OpenMP guide sentences are utilized, and lock array done [I][J] is used to ensure data consistency.

```
#pragma omp parallel for
for j in d_begin[J]:d_end[J]
  for i in d_begin[I]:d_end[I]
    wait(done[j][i])
$\phi_{psi} \leftarrow \phi_{up}$
    call $T\left(i, j, k, \phi_{psi}\right)$
    done[next(j)][i] = 1
  end for
end for
```

Benchmark	NAC4R (one thread)	NAC4R (eight threads)
KUCA reactor	3421.7 s	1935.8 s
IAEA pool reactor	5824.9 s	3551.3 s

Table 10.
Time results by the compile level -O0.

Table 10 collects the running time of two benchmark example, which utilizes the optimization level -O0, and only one MPI process is fixed during the experiment.

Performance conclusion: The speedup ratio is almost 1.8 and 1.6, which illuminates that the pipeline structure can gain performance bonus with small scale (I_a, J_b) for the SN method, such as the sample program Hydra. It can be further expanded to the parallel strategy of MPI + OpenMP.

6. Conclusion

Digital computers are used to simulate and analyze reactor core. For introducing parallel computing and promoting the efficiency, the core calculation with the integrated software structure is explored, which can provide the methodology of nuclear data, control layers, and parallel algorithms. The main software structure is discussed, and multiple parallel algorithms are analyzed concisely for sample core programs, with the corresponding conclusions as follows:

1. Integrated software structure (framework method) is benefited for the development of core software, and the similarity achieves the goal of reuse. For example, the underlying nuclear data could be managed as the effective manner and mapped to application. With the development of prototype software NAC4R, the research could be carried out in different control layers and demonstrate validity.

2. Parallel algorithms are refined as well as identify the parallel features that exist in the core model based on the existing theory. The features of four sample programs are described succinctly, which work well with the software structure. Parallel programming enables the expected efficiency improvement if some rules are obeyed, and the relevant performance results are given summarily.

It is necessary to establish systematic knowledge and suggestions for software approach and parallel algorithm development in reactor core calculation. On the one hand, it can integrate reusable patterns into computation, such as the instance of prototype software NAC4R. On the other hand, it can explore and record more specific parallel algorithms in this field.

Acknowledgements

The author would like to thank relevant scholars for their suggestions in NPIC, and also thank for the sample programs and computing resources provided by the 9th professional group in the design institute of NPIC. The research is supported by the discovery fund of NPIC (Number: 2019-06).

Nomenclature

A	coefficient matrix
x	length or position, m
k	effective multiplication factor
t	time, s
L	the matrix of Laplace operator
M	the matrix of flux moment
N	number of processors
Q	neutron source, $1/m^3s$
s	segment length of the characteristic ray, m
S	speedup ratio

Greek letters

Δ	difference
ρ	density, kg/m^3
ϕ	neutron flux density, $1/m^2s$
ψ	angular neutron flux density, $1/m^2s$
$\sum$	neutron macroscopic cross section, 1/m
ω	weight factor
ε	error threshold
Γ	operator

Subscripts/superscripts

s	serial
g	neutron energy group
p	parallel
i, j, k, z	mesh position
m	azimuth angle
v	volume
f	fission cross section
psi	temporary data
up	upstream data

Abbreviations

2D	two dimensional
3D	three dimensional
ACP1000	Advanced China Power 1000 MW

BLAS	Basic Linear Algebra Subprograms
CASL	Consortium for Advanced Simulation of Light water reactors
CORTH	CORE thermal-hydraulics
DENOVO	new three-dimensional parallel discrete ordinates code in SCALE
HDF	Hierarchical Data Format
IAEA	International Atomic Energy Agency
KBA	Koch-Baker-Alcouffe parallel algorithm
KYLIN2	advanced neutronics lattice code of NPIC
K-MOD	Kinetic Method of Demo
MOC	method of characteristics
MPI	message passing interface
NAC4R	New Architecture Computing for Reactor Core
NetCDF	scientific data NETwork Common Data Format
OpenMP	Open Multi-Processing
PETSc	Extensible Toolkit for Scientific Computation
PWR	Pressurized Water Reactor
SN	Discrete Ordinate method
SIMD	Single Instruction Multiple Data
TAU	Tuning and Analysis Utilities
VTK	Visualization ToolKit

Author details

Pingzhou Ming
Nuclear Power Institute of China, Chengdu, China

*Address all correspondence to: mingpz@mail.ustc.edu.cn

References

[1] Mylonakis AG, Varvayanni M, Catsaros N, et al. Multi-physics and multi-scale methods used in nuclear reactor analysis. Annals of Nuclear Energy. 2014;**72**:104-119. DOI: 10.1016/j.anucene.2014.05.002

[2] Arkadov GV, Zhukavin AP, Kroshilin AE, et al. The virtual digital nuclear power plant: A modern tool for supporting the lifecycle of VVER-based nuclear power units. Thermal Engineering. 2014;**61**(10):697-705. DOI: 10.1134/S0040601514100012

[3] Baker C, Davidson G, Evans TM, et al. High Performance Radiation Transport Simulations: Preparing for TITAN. SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. Salt Lake City, UT, USA: IEEE; 2012. DOI: 10.1109/SC.2012.64

[4] Weber DP, Sofu T, Yang WS, et al. High-fidelity light water reactor analysis with the numerical nuclear reactor. Nuclear Science and Engineering. 2007;**155**(3):395-408. DOI: 10.13182/NSE07-A2672

[5] Okumura K, Oka Y, Ishiwatari Y. Nuclear reactor calculations. In: Nuclear Reactor Design. Japan: Springer; 2014. pp. 49-126. DOI: 10.1007/978-4-431-54898-0_2

[6] Larsen EW. An overview of neutron transport problems and simulation techniques. In: Computational Methods in Transport. Berlin/Heidelberg: Springer; 2006. pp. 513-534. DOI: 10.1007/3-540-28125-8_26

[7] Saha P, Aksan N, Andersen J, et al. Issues and future direction of thermal-hydraulics research and development in nuclear power reactors. Nuclear Engineering and Design. 2013;**264**:3-23. DOI: 10.1016/j.nucengdes.2012.07.023

[8] Edsinger K, Stanek CR, Wirth BD. Light water reactor fuel performance: Current status, challenges, and future high fidelity modeling. JOM. 2011; **63**(8):49-52. DOI: 10.1007/s11837-011-0138-7

[9] Joubert W, Archibald R, Berrill M, et al. Accelerated application development: The ORNL Titan experience. Computers and Electrical Engineering. 2015;**46**:123-138. DOI: 10.1016/j.compeleceng.2015.04.008

[10] Gaston DR, Permann CJ, Peterson JW, et al. Physics-based multiscale coupling for full core nuclear reactor simulation. Annals of Nuclear Energy. 2015;**84**:45-54. DOI: 10.1080/00411450.2014.927364

[11] Prabhu P, Kim H, Oh T, et al. A survey of the practice of computational science. In: 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC). IEEE; 2011. pp. 1-12. DOI: 10.1145/2063348.2063374

[12] Carothers C, Ferscha A, Fujimoto R, et al. Computational challenges in modeling and simulation. In: Fujimoto R, Bock C, Chen W, Page E, Panchal JH, editors. Research Challenges in Modeling and Simulation for Engineering Complex Systems. Cham: Springer; 2017. pp. 45-74. DOI: 10.1007/978-3-319-58544-4.ch4

[13] Keutzer K, Massingill BL, Mattson TG, et al. A design pattern language for engineering (parallel) software: Merging the PLPP and OPL projects. In: Proceedings of the 2010 Workshop on Parallel Programming Patterns. ACM; 2010. p. 9. DOI: 10.1145/1953611.1953620

[14] Sbalzarini IF. Abstractions and middleware for petascale computing and beyond. International Journal of

Distributed Systems and Technologies. 2010;**1**(2):40-56. DOI: 10.4018/jdst.2010040103

[15] Mo Z, Zhang A, Cao X, et al. JASMIN: A parallel software infrastructure for scientific computing. Frontiers of Computer Science in China, Beijing. 2010;**4**(4):480-488. DOI: 10.1007/s11704-010-0120-5

[16] Bergmann RM, Vujić JL. Algorithmic choices in WARP–A framework for continuous energy Monte Carlo neutron transport in general 3D geometries on GPUs. Annals of Nuclear Energy. 2015;**77**:176-193. DOI: 10.1016/j.anucene.2014.10.039

[17] Trkov A. From basic nuclear data to applications. In: Workshop on Nuclear Data and Nuclear Reactors: Physics, Design and Safety; 2000

[18] Pavel T, Waltar A. Nuclear data and cross section processing. In: Fast Spectrum Reactors. Boston, MA: Springer; 2012. pp. 77-109. DOI: 10.1007/978-1-4419-9572-8.ch5

[19] International Atomic Energy Agency. Thermophysical Properties Database of Materials for Light Water Reactors and Heavy Water Reactors. IAEA TECDOC Series No. 1496(IAEA-TECDOC-1496). Vienna: International Atomic Energy Agency; 2006. Available from: https://www-pub.iaea.org/MTCD/Publications/PDF/te_1496_web.pdf

[20] Sánchez-Cervera S et al. Optimization of multidimensional cross-section tables for few-group core calculations. Annals of Nuclear Energy. 2014;**69**:226-237. DOI: 10.1016/j.anucene.2014.02.013

[21] Hill MD, Marty MR. Amdahl's law in the multicore era. Computer. IEEE; 2008;**41**(7):33-38. DOI: 10.1109/MC.2008.209

[22] Peter P. An Introduction to Parallel Programming. San Francisco: Morgan Kaufmann; 2011. 392 p. DOI: 10.1016/C2009-0-18471-4