Parallel calculation of the stationary diffusion equation

E O Soldatov, Yu N Volkov and M N Zizin

1 National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), 31 Kashirskoe shosse, 115409 Moscow, Russia
2 National Research Centre “Kurchatov Institute”, 1 Akademika Kurchatova pl., 123182 Moscow, Russia

E-mail: eo-soldatov@mail.ru

Abstract. Considering the complex neutron-physical systems with the number of calculation points than 1 million, the calculation time can be an hour or more, and in case of non-stationary process with the small step of time, the waiting can be a month. The aim of the study is to develop effective techniques for the stationary computational modeling using parallelization technology to imply in non-stationary calculations. The parallelization scheme with height decomposition for stationary model was developed. The calculated speedup on the parallel part of the code are in good agreement with the linear dependence. The theoretical possible acceleration was derived using Amdal's law. Comparing the obtained and theoretical curves demonstrates that the developed parallelization scheme has less acceleration due to losses. The losses and their relative contribution were identified.

1. Introduction
A high temperature gas cooled reactor (HTGR) based on fuel coated particles technology is one of the Generation IV prospective nuclear reactors for future sustainable and safe electricity and heat generation. The world currently are underway to develop a modular type HTGR (HTGR-M) with sphere type fuel cells and GT-MHR with prism type fuel assembly. Reactors of this type have a fundamental advantage on safety related to the lack of core melting at the accidents with loss of coolant.

A «zero power» ASTRA critical facility built in Kurchatov Institute [1] is intended for experimental investigations of neutronic peculiarities of modular HTGRs. Because of the «zero power» critical facility conditions, the uncertainties caused by material composition, temperatures and geometry are considerably lower than in a power reactor. In absence of temperature distribution uncertainties, burning, fission products accumulation, including xenon perturbation effects, the uncertainty of the fuel isotopic composition depends only on the technology of fuel fabrication. This feature allows using the results of experiments to verify neutronic codes used in HTGR calculations.

Experimental investigation at the ASTRA critical facility covers the following set of measurements:
- experimental investigation of spatial power density distributions (fission reaction rate measurements),
- measurements of the worth of safety rods mockups, their interference factor and calibration curves,
- measurements of neutron kinetic parameters (effective neutron lifetime),
- simulation of reactor physical startup.
The calculation analysis of stationary [2] and non-stationary [3] experiments demands the calculation by neutronic code to be able to solve time-dependent problems in accordance to space distribution of neutron field. One of the codes is SHIPR [4] software-based system, which contains space calculation modules including calculation of time-dependent problems.

The calculation model of ASTRA includes more than 1 million points (figure 1). In case of non-stationary process with the small step of time, the waiting can be a month. The development of modern computers towards a high-performance cluster architecture can significantly reduce the estimated time for many problems, but require a special restructuring of the numerical solution algorithm from sequential to parallel.

Therefore, the purpose of this work was to create a parallel algorithm and its software implementation for using on high-performance computing systems. At the first stage, it is necessary to work out the most effective methods of parallel computing on a stationary calculation.

Figure 1. The calculation model of ASTRA in SHIPR software-based system.

2. Methodology
Calculations of spatial fission rates distribution were carried out by utilization of the ShIPR code, preparation of macroconstants was carried out by WIMS-D/4 code [5].

ShIPR (Shell of Intelligent Package for Reactor) is an Integrated Development Environment of the Fortran codes application with automatic generation of main programs on the basis of the supply chain of computational modules, which implement the main stages of the neutronic calculation of nuclear reactors. The benefit of this system is the openness of the codes, respectively, ease in the modernization of neutronic calculation under the specific tasks. In SHIPR there is a set of modules that provide a matched solution for the 3D stationary and non-stationary tasks in multi-group diffusion approximation.

For solution of the time independent neutron diffusion equation, a finite difference method was used [6]. There are several methods for solving linear equation system such as Gauss elimination method, iteration method like Jacobi, Gauss-Seidel and SOR. Among iteration methods, SOR is much faster in
calculations. Thus, the derived system of linear equations can be solved by successive over-relaxation (SOR) method. [7]

The SOR method is an iterative method suitable for large number of unknowns due to its fast convergence rate. The method is ideal for saving memory and programming efforts, because the left-hand matrix of the linear system does not need to be constructed and the calculation is performed from mesh to mesh. This provides the opportunity to introduce parallelism to the solution process which significantly reduces the computational time. [8]

Because of the huge number of points (config. 1 – 1,099,952; config. 7A – 2,037,616) to be calculated in the space iterations, the MPI (Message Passing Interface) parallelization scheme was designed in the ShIPR module.

The main principle of the distribution of computational work over the nodes is the division of the computational domain into layers along the z axis. Each processor contains all the arrays that essential for the calculation, limited in space only by the size of their area (figure 2). During the iteration, data is exchanged between processors. At the end of this calculation, each processor sends to «zero» processor, the component of flux it has computed. Processor «zero» receives the components of flux from all processors and sums their contributions for each cell. It then takes the flux array from the previous iteration and compares them with the new flux by testing the pointwise relative difference against ε. If the ε criterion is satisfied, processor «zero» sends the flux distribution to the host, which successfully terminates execution.

![Figure 2. Decomposition of height layers into MPI architecture.](image)

3. Calculated results

Speedup calculation for the parallel part of the code is considered. The parallel part does not contain the sequential lines, which are calculated only in «zero» processor. Speedup is defined as the ratio of calculation time for the entire task without using the parallel interface and calculation time for the entire task using the parallel interface.

Figure 3 shows the speedup of the parallel part of code and comparison with linear dependence.
Figure 3. Speedup of the parallel part of code and comparison with linear dependence.

Speedup in parallel part of code calculated using MPI practically coincide with linear dependence. A good agreement in calculated speed up shows that MPI architecture was developed correctly. Deviation calculation dependence from linear can be explained by the computer delays.

In computer architecture, Amdahl's law is a formula which gives the theoretical speedup when using multiple processors. The theoretical speedup depends on number of processors and sequential code fraction.

Then the speedup for the entire task using the parallel interface was obtained. Comparing the obtained and theoretical curves demonstrates that the developed parallelization scheme has less speedup due to losses.

Figure 4 shows the comparison calculated dependence, Amdahl’s law and calculated dependence without losses.

Figure 4. Comparison calculated dependence, Amdahl’s law and calculated dependence without losses.
The developed parallelization scheme gives rise to eightfold speedup using 16 processors. The calculated dependence without losses shows agreement with Amdahl’s law. The losses and their relative contribution were identified. The losses are associated with data exchange between processors, increase in the number of iterations and deviation of calculated dependence from linear.

Table 1 demonstrates absolute share of losses in comparison with calculation time and relative contribution each component. It can be seen from table 1, that with an increase in the number of processors, the share of losses in the calculation time increases from 3.8 to 44.3 percent. Using 16 processors data exchange and an increase in the number of iterations make significant contribution to losses. The share of data exchange increases with an increase in the number of processors from 38.3 to 56.7 percent. The change in the number of iterations shows a lack of qualitative dependence, since the convergence coefficients were not corrected in parallel calculations.

| Losses                                      | Number of processors |
|---------------------------------------------|----------------------|
| Absolute share of losses in comparison with calculation time, % | 2        | 4        | 8        | 16        |
| Increase in the number of iterations, rel. % | 30.4                | 42.6                | 47.0                | 40.3                |
| Data exchange between processors, rel. %   | 38.3                | 28.9                | 41.2                | 56.7                |
| Deviation of calculated dependence from linear, rel. % | 31.3                | 28.5                | 11.8                | 3.1             |

4. Conclusion

MPI parallelization scheme was designed in the ShIPR module. The main principle of the distribution of computational work over the nodes is the division of the computational domain into layers along the z axis. Speedup calculation for the parallel part of the code is considered. Speedup in parallel part of code calculated using MPI practically coincide with linear dependence. A good agreement in calculated speed up shows that MPI architecture was developed correctly. Then the speedup for the entire task using the parallel interface was obtained. Comparing the obtained and theoretical curves demonstrates that the developed parallelization scheme has less speedup due to losses. The losses and their relative contribution were identified. The losses are associated with data exchange between processors, increase in the number of iterations and deviation of calculated dependence from linear. The share of losses in the calculation time increases from 3.8 to 44.3 percent. The share of data exchange increases with an increase in the number of processors from 38.3 to 56.7 percent. The change in the number of iterations shows a lack of qualitative dependence, since the convergence coefficients were not corrected in parallel calculations.

References

[1] Ponomarev-Stepnoi N N, Kukharkin N E and Glushkov E S 2003 Nucl. Eng. and Des. 222 pp 215–229
[2] Zizin M N, Boyarinov V F, Nevinitsa V A, Fomichenko P A, Volkov Yu N and Kruglikov A E 2020 Verification of the stationary module of the ShIPR software system for modelling experiments of the ASTRA HTGR type critical facility KERNTECHNIK 85(1) pp 4-8
[3] Volkov Y N, Kruglikov A E, Zizin M N, Boyarinov V F, Nevinitsa V A and Fomichenko P A 2018. Verification of the non-stationary neutronic calculation module in the ShIPR software system for modelling of experiments at the ASTRA HTGR type critical facility Journal of Physics: Conference Series 1133
[4] Zizin M N 2005 Intellectual software system ShIPRW for mathematical simulation of nuclear
reactors (Preprint IAE-6354/5)  

[5] Askew J R, Fayers F J, and Kemshell P B 1966 A general description of the code WIMS J. Br. Nucl. Energy Soc. 5 p 564  

[6] Zhuang P and Liu F 2007 Finite difference approximation for two-dimensional time fractional diffusion equation Journal of Algorithms & Computational Technology 1(1) pp 1–15  

[7] Nakamura S Numerical Methods with software 1991  

[8] Diethelm K 2011 An efficient parallel algorithm for the numerical solution of fractional differential equations Fractional Calculus and Applied Analysis 14(3) pp 475–490