Technique for Teaching Parallel Programming via Solving a Computational Electrodynamics Problems

Sergey Mosin, Nikolai Pleshchinskii, Ilya Pleshchinskii and ✉ Dmitrii Tumakov

Kazan Federal University, Kazan, Russia
dtumakov@kpfu.ru

Abstract. Three-dimensional problems of computational electrodynamics for the regions of complex shape can be solved within the reasonable time only using multiprocessor computer systems. The paper discusses the process of converting sequential algorithms into more efficient programs using some special techniques, including object-oriented programming concepts. The special classes for data storage are recommended to use at the first stage of programming. Many objects in the program can be destroyed when optimizing the code. Special attention is paid to the testing of computer programs. As an example, the problem of the electromagnetic waves diffraction by screens in three-dimensional waveguide structures and its particular cases are considered. The technique of constructing a parallel code for solving the diffraction problem is used in teaching parallel programming.

Keywords: Parallel programming teaching · Effective program · Computational electrodynamics.

1 Introduction

Parallel programming is becoming more widespread with increasing requirements for resources needed to solve problems and with development of computer technology. Currently, the three most widely used parallel programming technologies are OpenMP [1]–[3], MPI [3]–[5] and CUDA [6]–[8]. Each of them is oriented to a certain type of multiprocessor computer systems. All these three technologies are important for training professional programmers at bachelor and master levels. The programmer should be able to chose the most suitable technology and take into account a particular computer system for solving the considered problem.

There is a large amount of literature devoted to algorithms for the parallelization of classical problems, for example [9]–[12]. However, the issues of "correct" training in parallel programming technology remain relevant to this day. Students, who study parallel programming for the first time, deal with difficulty in understanding some specific features due to essential difference between implementations of basic methods using parallel and logical/structured programming. Different approaches to teaching are considered, starting from the students of

the first years [13] and ending with the Ph.D. students [14]. It is also obvious that the training should take into account the peculiarities in the preparation of students for certain specializations [15].

In this paper, we propose a methodology for teaching bachelor and master students the basics of parallel programming via the problems of electrodynamics. Computational electrodynamics contains a wide range of problems, for the numerical solution of which the methods of parallel programming are successfully used. The key problems of waveguide electrodynamics are problems for electromagnetic waves diffraction by thin conducting screens located in waveguide structures. In the three-dimensional case, the numerical solution of such problems is associated with high computational cost. We will show on a specific example (and its special cases) that (1) algorithms for numerical solution for the problems of the electromagnetic wave diffraction by screens allow effective parallelization and (2) development and debugging of computer programs for solving such problems contribute to the mastering the modern parallel programming technologies.

2 Infinite Systems of Linear Algebraic Equations in Diffraction Problems

The problem of wave diffraction by the screen in the general formulation is as follows. Suppose that an infinitely thin perfectly conducting screen is placed in the cross section of a cylindrical waveguide with metal walls. The electromagnetic wave falls on the screen. It is necessary to find the electromagnetic field that arises from its diffraction.

Any electromagnetic field in a cylindrical waveguide with conducting walls can be represented as an overlay of a superposition set of eigenwaves of TE- and TM-polarization [16]. Therefore, the problem of determining the diffracted electromagnetic field is reduced to determining the coefficients of the expansion of this field in the eigenvectors of the waveguide.

The work [17] shows that the diffraction problem under consideration is equivalent to two independent infinite systems of linear algebraic equations (ISLAE) with respect to the unknown coefficients A_k^-, b_k^- of the field expansion:

$$-a_k^- \lambda_k + \sum_{m=0}^{+\infty} a_m^- \gamma_m \lambda_m \sum_{n=0}^{+\infty} \frac{1}{\gamma_n} I_{m,n}^{\varphi} J_{n,k}^{\varphi} = \sum_{m=0}^{+\infty} a_m^0 \lambda_m I_{m,k}^{\varphi}, \quad k = 0, 1, \ldots \quad (1)$$

$$-b_k^- \chi_k + \sum_{m=0}^{+\infty} b_m^- \delta_m \chi_m \sum_{n=0}^{+\infty} \frac{1}{\delta_n} I_{m,n}^{\psi} J_{n,k}^{\psi} = \sum_{m=0}^{+\infty} b_m^0 \chi_m I_{m,k}^{\psi}, \quad k = 0, 1, \ldots \quad (2)$$

Here

$$I^{\varphi}_{m,n} = \iint\limits_{\mathcal{M}} \varphi_m(\xi,\eta)\,\varphi_n(\xi,\eta)\,d\xi\,d\eta, \quad J^{\varphi}_{m,n} = \iint\limits_{\mathcal{N}} \varphi_m(\xi,\eta)\,\varphi_n(\xi,\eta)\,d\xi\,d\eta,$$
$$I^{\psi}_{m,n} = \iint\limits_{\mathcal{M}} \psi_m(\xi,\eta)\,\psi_n(\xi,\eta)\,d\xi\,d\eta, \quad J^{\psi}_{m,n} = \iint\limits_{\mathcal{N}} \psi_m(\xi,\eta)\,\psi_n(\xi,\eta)\,d\xi\,d\eta, \tag{3}$$

where $\psi_m(x,y)$ and $\chi_m,\ m = 0,1,\ldots$ are eigenfunctions and eigenvalues of the eigenvalue problem:

$$\Delta\,\psi_m + \chi_m\,\psi_m = 0,\ \ \psi_m\big|_C = 0, \tag{4}$$

$\varphi_m(x,y)$ and $\lambda_m,\ m = 0,1,\ldots$ are eigenfunctions and eigenvalues of the eigenvalue problem:

$$\Delta\,\varphi_m + \lambda_m\,\varphi_m = 0,\ \ \frac{\partial\varphi}{\partial\nu}\Big|_C = 0, \tag{5}$$

$\gamma_m = \sqrt{k^2-\lambda_m},\ \ \delta_m = \sqrt{k^2-\chi_m}$, C is the boundary of the waveguide cross section where the cross section consists of two parts: $\mathcal{M}$ corresponds to the screen and $\mathcal{N}$ is the remaining part of the section.

Thus, the first obvious step when parallelizing the algorithm is the definition of TE- and TM-fields using independent computing nodes.

3 Parallel Calculations and Verification of Results of the Calculations

The diffraction problem is reduced to two independent ISLAE. An approximate solution of each of them can be found by the truncation method: it is necessary to retain a finite number of unknowns (assuming that the remaining ones are equal to zero) and the same number of equations. Thus, the solution of the original problem reduces to the solution of two systems of linear algebraic equations (SLAE).

Note that the truncation order can not be established a priori. Even if estimates of the approximate solution error were obtained (this could be done only in a few simple special cases), then they are very crude. Therefore, it is assumed that a series of calculations will be performed, during which it becomes possible to establish reasonable values for the truncation parameters.

The computational algorithm consists of three main steps during the numerical solving of the SLAE. Firstly, the integrals I and J are calculated. Then the values of the coefficients of the SLAE matrix and the vector of the right-hand sides are calculated. Finally, a solution to the SLAE is obtained. It is easy to see that each stage of the algorithm allows parallelization.

It should be expected that the preparation of coefficients matrices and vectors of the right-hand sides of the SLAE will require significantly more CPU time than solving the final SLAE. Therefore, the analysis of existing and the

development of new optimal parallel methods for solving the SLAE are not of primary importance.

The reliability of the computing results should be placed to the foreground. Therefore, the writing and debugging of the program are reasonable to start with the simple special cases of the general problem. Obtained result should be tested at each step of the computational experiment. Series of tests are developed for this purpose.

The following tests can be recommended for the computational problems of the class under consideration.

1. With increasing the order of approximation (that is, with an increase in the number of unknowns and the number of equations). The convergence of the sequence of approximate solutions to the exact solution must be observed. Since the exact value is unknown, it is to be expected that the sought values will change insignificantly as the dimension of the SLAE increases, beginning with some certain dimension.

2. The law of conservation of energy must be fulfilled: how much energy a wave brought from an external source to an inhomogeneity, the same amount in total must go from heterogeneity to infinity. The energy balance should improve when the dimension of the SLAE increases. The empirical experience shows that this happens when the accuracy of calculating the coefficients matrix of SLAE is increased.

3. In limiting cases, from physical considerations, it is possible to determine what solution of the problem should be like. For example, if the conducting screen in the section of the waveguide structure has a small area, then the field scattered from the screen should become insignificant. Conversely, if the screen occupies almost the entire cross-section, then the original wave should almost completely be reflected from the screen.

4. Finally, if we are dealing with problems of the electromagnetic waves diffraction by conducting bodies, then it is most important to check whether the boundary condition on the metal is satisfied (the tangential component of the electric vector must be zero).

The first three checks are based only on the necessary conditions for the reliability of the computing results. The fourth test is more important, because if we are seeking a solution of the diffraction problem in the form of expansions in the waveguide waves, then this guarantees that a number of requirements are automatically satisfied. These requirements include the Maxwell equations, the conditions on the waveguide walls, and the conditions for the coupling of the fields in the waveguide section outside the screen. In fact, the coefficients of these expansions must be found in such a way that the conditions on the conducting screen are satisfied. It is clear, that these conditions will never be fulfilled exactly due to approximate calculations .

4 Fast Programming or Effective Code?

There are two parameters which are considered at a program implementation of the computational algorithms. The first one is operationability of the proposed algorithm and the second one is effectiveness of the program providing minimal time cost on a computation.

The two-dimensional diffraction problem of a electromagnetic eigenwave by a transverse partition in a plane waveguide with metal walls is considered as a simple special case of the general diffraction problem. The problem is reduced to SLAE (after truncating the ISLAE) for the TE-polarization field.

$$-a_k + \sum_{n=1}^{N} a_n \gamma_n \sum_{m=1}^{M} \frac{1}{\gamma_m} I_{n,m} J_{m,k} = I_{l,k}, \quad k = 1..N, \tag{6}$$

$a_n,\ n = 1..N$ are the unknown coefficients. The propagation constants (complex)

$$\gamma_n = \sqrt{k^2 - \left(\frac{\pi n}{h}\right)^2} \tag{7}$$

are defined according to the following condition

$$\operatorname{Re}\gamma_n > 0, \quad or \quad \operatorname{Im}\gamma_n > 0. \tag{8}$$

$k = 2\pi/\lambda$ is the wavenumber, λ is the wavelength, h is the thickness of the waveguide. All these quantities must have the same order and $\lambda < 2h$ in order to have, at least, γ_1 being real. The integrals in (6) have the following form:

$$I_{m,n} = \int\limits_{\mathcal{M}} s_n(t)\, s_m(t)\, dt, \quad J_{m,k} = \int\limits_{\mathcal{N}} s_m(t)\, s_k(t)\, dt = \delta_{m,k} - I_{m,k}, \tag{9}$$

where

$$s_n(t) = \sqrt{\frac{2}{h}} \sin\frac{\pi n}{h} t. \tag{10}$$

Here $\mathcal{M}$ is the part of the segment $[0, h]$ occupied by the screen, and $\mathcal{N}$ is the remaining part of $[0, h]$. In the simplest case, $\mathcal{M} = [\alpha, \beta] \subset [0, h]$; in the more general case, $\mathcal{M}$ includes several segments. The numbers M and N are parameters of the method; l is the number of an eigenwave incident on the partition.

Taking this case study as a starting point, it is convenient to begin studying the features of using the OpenMP parallel programming technology.

The following procedure is recommended in the work [18]. C++ programming language is used for the algorithms description. All the initial data are declared as global constants. Functions which return the values $\gamma_n,\ I_n,\ J_n$ are considered at the first stage of programming. The standard C++ libraries for working with complex numbers may be used since almost all values in the program are the complex numbers. Alternatively the own proprietary software may be prepared

as a special library. For example, we need to describe the class "comp" for declaring complex-valued variables. We describe the necessary operations on the complex numbers as functions of the class members.

When using standard libraries, we should pay attention, for example, to how the branches of multi-valued functions are selected. For example, in our problem, either $\mathrm{Re}\,\gamma_n > 0$, or $\mathrm{Im}\,\gamma_n > 0$. Is this possible if we extract the root of a complex number using the standard function?

The classes "covect" and "comatr" are prepared in order to store the complex values of the elements of vectors and matrices. The indices of elements are changed in the same way as in the original formulas in order to reduce the probability of error in accessing the elements of such arrays. It also makes sense to overload the operator "()" to access these elements.

If the first version of the program confirms the operability of the algorithm, then the multiple calls of the functions $\gamma_n,\ I_n,\ J_n$ will be replaced by the calls to array elements with the same names. Of course, these arrays must be filled before calculating the coefficients of the SLAE matrix. At the same time, the computing time will be significantly reduced, but more RAM space will be required.

Now one can proceed to convert the serial code to parallel code. OpenMP technology allows to do this most simply. The first skills of parallel programming are easy to acquire if we use the directive "#pragma omp parallel for", understand the difference between common and private data and learn the reduction (when computing internal sums) in the example under consideration.

It should not be expected the significant decreasing of the computation time when the loops are parallelized in such a simple problem. Rather, it is even the other way around. In fact, the cost of creating the additional threads, allocating local memory and some other actions overlap the gain from acceleration when parallelizing loops with a number of iterations less than 2000.

The CUDA technology is also well mastered via this simple example. Since the solving SLAE requires significantly less time than preparing the coefficient matrix and the right-hand side vector, it makes sense to transfer only these preliminary calculations to the device. One or more cores should calculate the arrays $\gamma_n,\ I_n,\ J_n$ and calculate the elements of the coefficients matrix and the vector of the right-hand sides. Then, the matrix and the vector are sent to the host and the SLAE is solved.

The time for transferring data from the device's memory to the host's memory and back is the critical parameter for CUDA technology. The advisability of having the dimensions of arrays as multiples of the dimension for the threads (warp) has to be assessed. We also recommend teaching how to use the constant and shared memory of the device.

5 Benefit and Harm of Object Programming

As already mentioned, at the first stages of programming, it makes sense to define special classes for storing data with encapsulated operations on them. This approach significantly reduces the time spent on the debugging programs,

and simplifies the perception of their text. In addition, the probability of error is reduced, since the programmer does not have to rewrite (copy) several fragments of the code.

At the same time, the use of object programming ideas often increases the computing time and, therefore, reduces the effectiveness of programs. This is most noticeable when using CUDA technology. There is an opinion that object programming is not compatible with the technology of computing on a GPU. But this is not so; convincing examples are given in the book [6]. We only need to have two assignment-compatible class descriptions, one for the host, and one for the device.

If the problem of the electromagnetic wave diffraction in a rectangular waveguide is considered, then it is customary to enumerate the eigenfunctions and eigenvalues using two indices:

$$\begin{aligned} \psi_{mn} &= \sin\frac{\pi m x}{a}\,\sin\frac{\pi n y}{b}, \quad m,n = 1,2,\ldots, \\ \varphi_{mn} &= \cos\frac{\pi m x}{a}\,\cos\frac{\pi n y}{b}, \quad m,n = (0),1,\ldots, \end{aligned} \tag{11}$$

$$\chi_{mn} = \lambda_{mn} = \sqrt{k^2 - (\pi m/a)^2 - (\pi n/b)^2}. \tag{12}$$

Independent ISLAE in the diffraction problem have the same form as the system (1), (2), but all sums become double. In this case, it is necessary to provide two classes of constructors for the corresponding classes and to arrange an access to the elements of matrices and vectors in two ways: by double number and by number in the linear list. Here is one of the examples:

```
class covect {
    comp* p;
protected:
    int imi, ima, ile;
    int jmi, jma, kmi, kma, jle;
public:
    covect(int imii, int imaa);
    covect(int jmii, int jmaa, int kmii, int kmaa);
    ~covect() delete [] p;
    comp& operator ()(int i) return p[i-imi];
    comp& operator ()(int j, int k) return p[j-jmi+(k-kmi)*jle];
};
```

The main drawback of the object-oriented approach is related to the need to take care of address alignment when accessing global memory at optimization CUDA programs. Otherwise, the time required to transfer data becomes too large. In this sense, the structure of the arrays is better than the array of structures. In our case, this should be understood as follows: a pair of vectors

of real and imaginary parts of complex numbers will be processed faster than a vector of complex numbers.

Experience shows that object-oriented programming is very effective for testing the efficiency of a numerical method (especially if the programmer already has well-established libraries with a description of "live" data). But when we need to optimize the code, we have to "disassemble" the classes into separate parts. As a rule, after this we have to search for errors in the program for a long time.

Similar examples appear in the numerical solving of problems of diffraction by screens in layered media [20], by spherical screens [19] and by doubly periodic gratings [18].

6 Summary

Let us single out the main advices for parallel programming of electrodynamics problems.

1. Write a serial code for solving the problem.

2. Use object-oriented programming to improve understanding and facilitate debugging when writing the code.

3. Carry out experiments. For example, the influence of the truncation parameters on the solution of the problem has to be monitored at solving the ISLAE. Check the satisfaction of physical laws for the solution obtained.

4. Redefine the library functions for complex data types, if necessary. For example, it can be the extraction of the square root.

5. Provide an access to both of the two indices and one index (the number in the linear list) for problems with two-index arrays.

6. Abandon the object-oriented data, especially when using CUDA for parallelizing.

7. Use the array structure instead of an array of structures, especially in CUDA programs.

7 Conclusions

The advices for getting started with OpenMP technology and some recommendations for teaching CUDA programming are given in the paper. The teaching technology developed in parallel programming is used at the Department of Applied Mathematics of the Institute of Computational Mathematics and Information Technologies of Kazan Federal University when conducting classes with first-year undergraduates in the specialization "Applied Mathematics and Informatics". The total number of students is ∼30 people. The content of the lectures is in line with the textbooks [3], [8]. In the laboratory classes, various problems

are offered for unassisted solution. Parallel programming technology can be also used for R&D.

We recommend to use Visual Studio tools or open-source g++ compilers (MinGW package) for the Linux platform for mastering the parallel programming on the base of C++ language. Note also that you can improve the training program, supplementing it with the study of specialized patterns [21].

Acknowledgements

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

1. Antonov, A.S.: Parallel Programming using OpenMP technology: textbook. Izd-vo MGU, Moscow (2009) [in Russian]
2. Levin, M.P.: Parallel programming with OpenMP: textbook. BINOM: Laboratoriya znanij, Moscow (2012) [in Russian]
3. Pleshchinskii, N.B., Pleshchinskii, I.N.: Multiprocessor computing systems. Parallel programming technologies: textbook. Izd-vo Kazan. un-ta, Kazan (2018) [in Russian]
4. Korneev, V.D.: Parallel programming in MPI. IKI, Moscow-Izhevsk (2003) [in Russian]
5. Grishagin, V.A., Svistunov, A.N.: Parallel programming based on MPI. Textbook. Izd-vo NNGU im. N.I. Lobachevskogo, Nizhny Novgorod (2005) [in Russian]
6. Sanders, J., Kandrot, E.: CUDA by example an introduction to general-purpose GPU programming. Addison-Wesley (2010)
7. Boreskov, A.V., Kharlamov, A.A.: Parallel computing on the GPU. Architecture and software model of CUDA: textbook. Izd-vo MGU, Moscow (2012) [in Russian]
8. Tumakov, D.N., Chickrin, D.E., Egorchev, A.A, Golousov, S.V.: CUDA programming technology: textbook. Izd-vo Kazan. un-ta, Kazan (2017) [in Russian]
9. Herlihy, M., Shavit, N.: The Art of Multiprocessor Programming, Elsevier (2008)
10. Trobec, R., Vajteršic, M., Zinterhof, P.: Parallel computing. Numerics, applications, and trends, Springer (2009)
11. Voevodin, V., Voevodin, Vl.: Parallel computing. BKhV-Peterburg, SPb (2002) [in Russian]
12. Gergel, V.: High-performance computing for multi-processor multi-core systems, Izd-vo MGU, Moscow (2010) [in Russian]
13. Grossman, M., Aziz, M., Chi, H., Tibrewal, A., Imam, S., Sarkar, V.: Pedagogy and tools for teaching parallel computing at the sophomore undergraduate level. J. of Parallel Distrib. Comput. **105**, 18–30 (2017) https://doi.org/10.1016/j.jpdc.2016.12.026
14. Antonov, A., Popova, N., Voevodin, Vl.: Computational science and HPC education for graduate students: Paving the way to exascale. J. Parallel Distrib. Comput. (2018) https://doi.org/10.1016/j.jpdc.2018.02.023 [In Print]
15. Shemetova, A.: Techniques for parallel programming teaching. J. Appl. Informatics **11**(6), 43–48 (2016) [In Russian]

16. Samarskii, A.A., Tichonov, A.N.: The representation of the field in a waveguide in the form of the sum of TE and TM modes. Zhurn. Teoretich. Fiziki **18**(7), 971–985 (1948) [in Russian]
17. Pleshchinskii, N.B.: On boundary value problems for Maxwell set of equations in cylindrical domain. SOP Trans. Appl. Math. **1**(2), 117–125 (2014) https://doi.org/10.15764/AM.2014.02011
18. Pleshchinskii, I., Pleshchinskii, N.: Software implementation of numerical algorithms of solving the electromagnetic wave diffraction problems by periodical gratings. J. Fundam. Appl. Sci. **9**(1S), 1602–1614 (2017) https://doi.org/10.4314/jfas.v9i1s.809
19. Pleshchinskii, N.B., Tumakov, D.N.: A new approach to investigation of Maxwell equations in spherical coordinates. Lobachevskii J. Math. **36**(1), 15–27 (2015) https://doi.org/10.1134/S1995080215010114
20. Pleshchinskaya, I.E., Pleshchinskii, N.B.: On parallel algorithms for solving problems of scattering of electromagnetic waves by conducting thin screens in layered media. Vestnik Kazansk. gos. tekhnol. un-ta **16**(17), 38–41 (2013) [in Russian]
21. Capel, M.I., Tomeu, A.J., Salguero, A.G.: Teaching concurrent and parallel programming by patterns: An interactive ICT approach. J. Parallel Distrib. Comput. **105**, 42–52 (2017) https://doi.org/10.1016/j.jpdc.2017.01.010