04

Technical issues for kinetic modeling of radio-frequency gas discharge: KITe code

© L.A. Varshavchik, D.D. Galitsyn, E.A. Starovoitov, V.A. Bocharnikov, S.A. Nikitenko, E.E. Mukhin

Ioffe Institute, St. Petersburg, Russia
E-mail: lidia.varsh@mail.ioffe.ru

Received June 5, 2024
Revised September 27, 2024
Accepted October 1, 2024

The KITe code has been developed for kinetic 3D modeling of impurity atoms and ions transport in neutral gas, considering external magnetic field, and for capacitive-coupled radio-frequency gas discharge modeling. This paper describes the main technical issues: parallel computing implementation, managing the groups of particles, a win-win choice of particle parameters from distributions, bounding volume hierarchy (BVH) application to process particle collisions with walls and data parallelism (AVX instructions). This approach allows for modeling to be performed on average in one-two days of computing time.

Keywords: particle-in-cell, kinetic modeling, RF discharge, KITe.

DOI: 10.61011/TPL.2025.02.60624.20011

A capacitive-coupled radio-frequency (RF) gas discharge has a wide range of applications, such as film deposition, surface etching, material modification, and certain applications in space engineering. Most existing codes for RF discharge modeling operate in the hydrodynamic approximation at high pressures. The two-dimensional approximation utilizing symmetry of the model is common among kinetic codes applicable at low pressures. Our research group has developed the KITe [1] code, which allows one to perform fully three-dimensional modeling of a low-pressure (up to 10 Pa) capacitive-coupled RF discharge and may help develop and optimize relevant technologies. Owing to the high resource intensity and technical complexity of implementation of this challenge, only a few similar programs have been developed worldwide. Therefore, it appears important to provide a description of technical solutions and optimizations that ensure operability and speed of the discussed KITe code.

The classical particle-in-cell (PIC) method, wherein plasma is treated as a set of test particles with a certain statistical weight, is used for modeling. The time scale is divided into short intervals Δt, and the particle velocities and coordinates are recalculated in each of them. At the start of calculation, fluxes of identical test particles are specified in volume or on 2D surfaces. Energy distributions, particle flux densities, and a three-channel distribution for the velocity vector, where a single channel characterizes the distribution along axis x, y, or z, are set. To speed up the generation of particles, a distribution construction algorithm is applied in such a way that a produced particle is guaranteed to satisfy the required conditions. A distribution cache is used: if a distribution is being used already in some form in KITe, it is not constructed for the second time.

A particle manager was developed for working with test particles. Its primary purpose is to reduce the memory consumption in calculations. Particles are divided into several groups. This allows for unified handling of large numbers of particles: data common to a set of particles are not duplicated in each particle object. Particles are divided into four groups: tracked and untracked particles produced in a flux and having its characteristics; tracked and untracked particles of a certain element that are not associated with a flux (produced as a result of collision or wall sputtering). Some groups may be excluded from calculation. A garbage collector mechanism for lightly used groups is provided. The group structure includes the parameters of all particles, the number of elements (particles) in this group, the size of the particle object in memory, the size of a block in memory for charge storage, the belonging of particles to a flux, and the offset in writing to memory. The addition of new particles and iteration through them are done in parallel via a set of pre-initialized threads (*thread pool*); when a particle is created, writing to its group is blocked temporarily. The algorithm for extracting a particle from binary data differs from one group to the other. Old particles are not destroyed immediately: newly created particles are written in place of the destroyed ones.

An accelerating data structure is used to reduce the complexity of detection of collisions with objects in modeling of a particle–wall collision. Acceleration structures (geometric databases designed for efficient storage and fast retrieval of data) may be divided into two categories: structures with space partitioning and structures with object partitioning. When space is partitioned, the volume is divided into several regions, and a record is made of which objects belong to each region. During collision detection, a sequence of regions through which the object passes is calculated first,

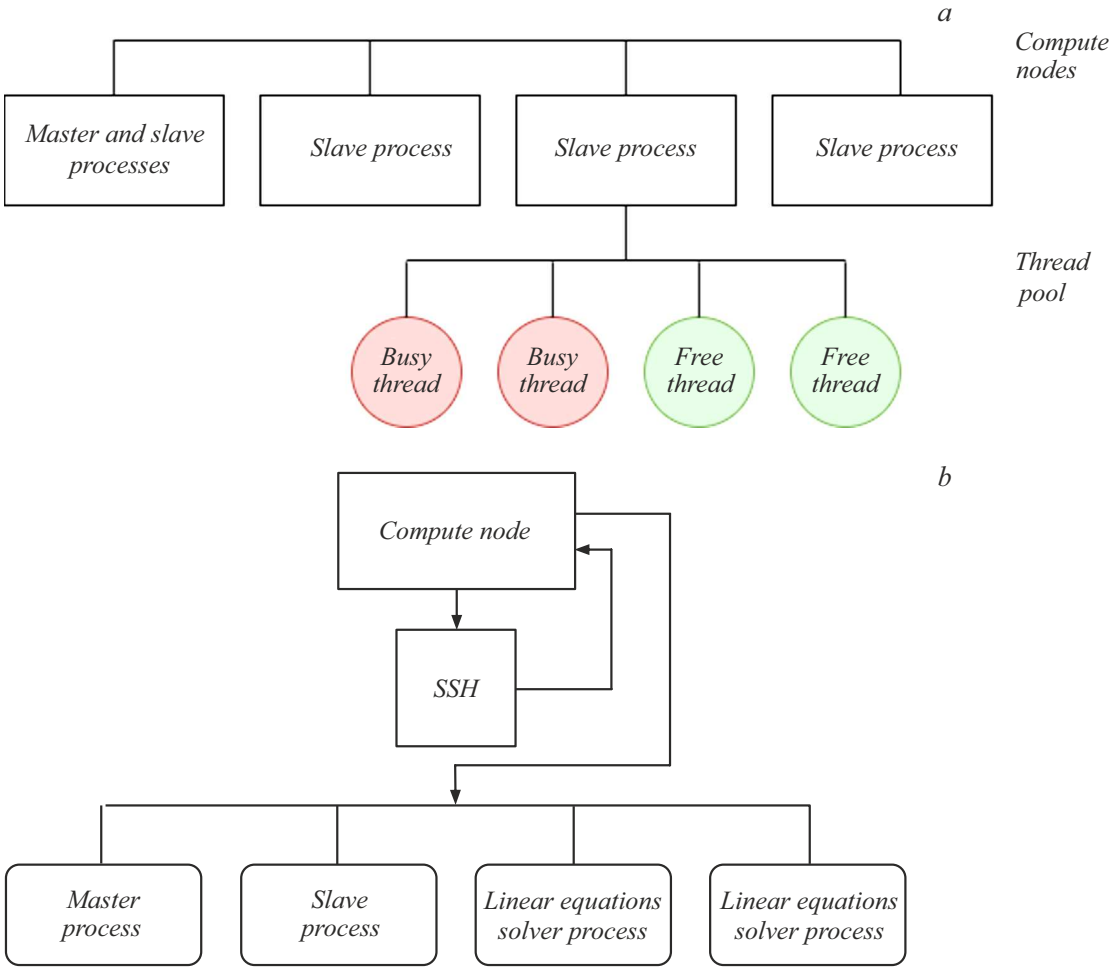

Figure 1. Diagram of parallel operation of KITe in particle transport modeling (a) and in the calculation of electric fields (b).

and then collisions with objects within the specified regions are detected. A k-dimensional tree is an example of such a structure. Object partitioning involves breaking down all geometry objects into smaller components of that geometry. One such structure is the bounding volume hierarchy (BVH) [2], which was chosen for KITe calculations.

The work with BVH in KITe is currently aimed at reducing the number of conditional branches, which helped ease the load on the prediction unit (which predicts whether a conditional branch will be executed or not) and reduce the probability of resetting the computational pipeline (if a prediction is incorrect, all preliminary calculations embedded in the pipeline are reset), and increasing the probability of caching of the next BVH segment and improving the ability to work with SIMD (see below). The use of this structure allows for a significant acceleration of the collection of statistics on particles adsorbed on the walls and the construction and optimization of a system of linear algebraic equations (linear system) for calculating electric fields and makes it possible to work with a dynamic wall with successively deposited material layers taken into account. In the future, BVH should allow the KITe code to be adapted easily to a computing cluster with a graphics processing unit (if needed).

When parallel code operation is organized, the simulation is divided into two stages: particle tracing and calculation of local electric fields. Parallelism „by particles“ (a different approach is parallelism „by space“) is used in tracing. When fields are calculated, parallelism is established in solving of the linear system to which the Poisson equation is reduced.

Since calculations using the KITe code are run on a computing cluster on a CPU, the Message Passing Interface (MPI) is used for passing messages between processes performing the same task. This technology provides an opportunity to run the KITe code on multiple nodes of a computing cluster simultaneously, distributing the execution of the code between them and, consequently, reducing the calculation time. This principle is implemented in the calculation of particle transport (Fig. 1, a): two processes (*master* and *slave*) are initiated on one node, and one slave process is initiated on all the other nodes. Each slave process (i.e., each computing node) uses its own „stationary“ thread pool, which persists until the entire code terminates, and has its own autonomous particle manager.

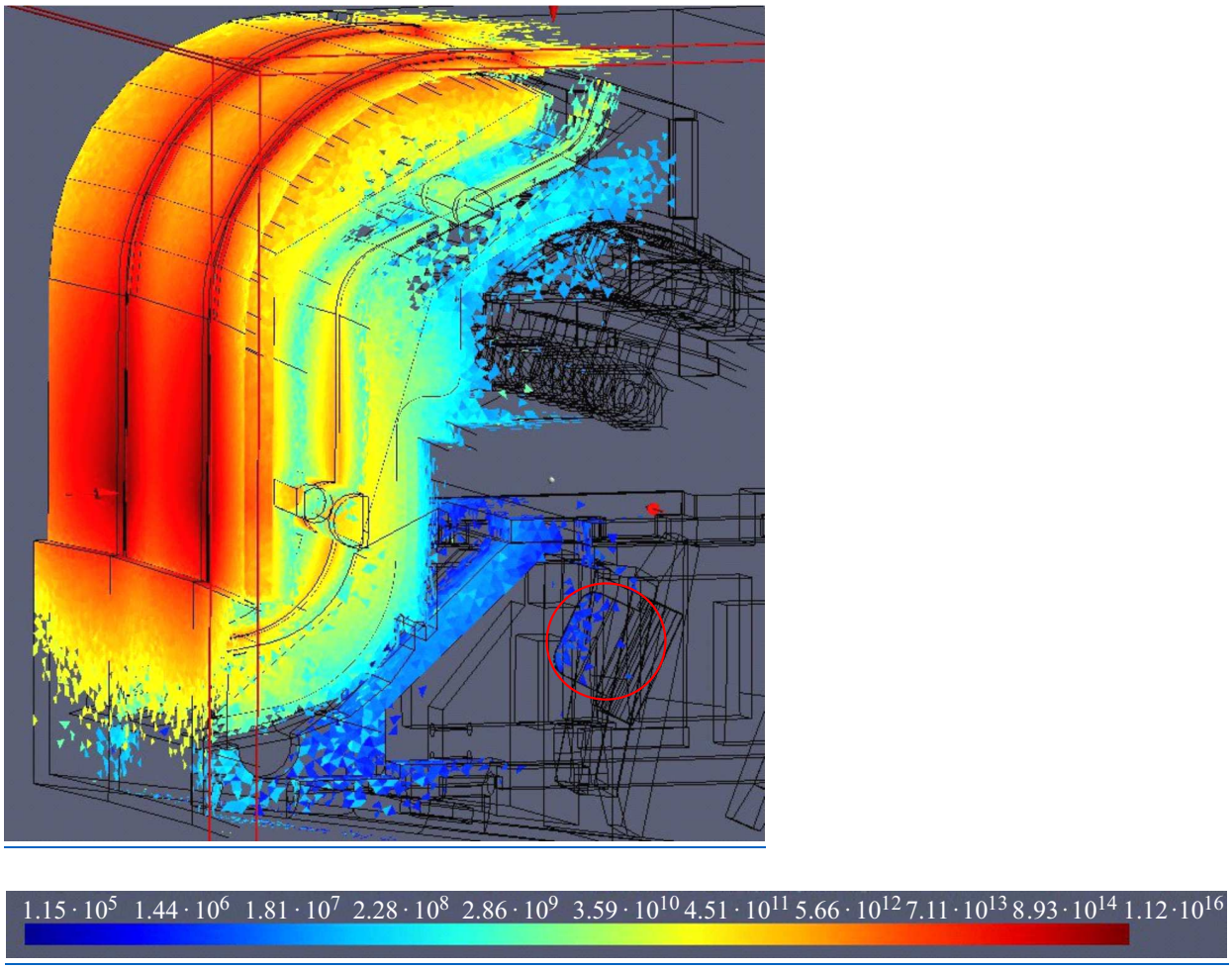

Figure 2. KITe calculation of the density [cm^{-2}] of beryllium atoms deposited on the walls of the ITER diverter Thomson scattering diagnostic system. The first mirror is marked with a red circle. A color version of the figure is provided in the online version of the paper.

Thread synchronization is performed at each iteration of the program.

At the same time, MPI is also used in the implementation of linear system solvers, since their implementation in libraries is designed for parallelism by processes (instead of threads) and, in most cases, is based on MPI. The MPI_Comm_spawn function, which creates the maximum possible number of copies of one process with MPI, should be used to form an additional group of processes belonging to the linear system solution. Thus, in addition to the processes created initially to perform the calculation, additional processes are created to solve the linear system. When the linear system is solved, the main group of processes is suspended; in turn, additional processes are suspended when control is returned to the main group. The ordering of stop and restart of processes is performed using named FIFO pipes.

The computing cluster of the Ioffe Institute utilizes the Slurm task management system that does not support the MPI_Comm_spawn function. The current workaround for this limitation (Fig. 1, *b*) allows the code to be run on the cluster, albeit on a single computing node only: KITe „rents“ one cluster node and, being in this node, connects to it via SSH, and MPI is started via SSH connection. This approach does not utilize fully the capabilities of the computing cluster, although the calculation is performed faster than on a PC or server. To solve this problem, one needs to rework the code architecture in such a way that the required number of processes is allocated straight away and auxiliary communicators are created afterwards to divide the groups of transport calculations and linear system solvers and establish communication between them.

Additional computational speedup may be achieved by combining the execution of same-type linear math operations and collision detection operations in BVH. The SIMD computing principle [3], which allows one to establish parallelism at the data management level by executing one instruction for a set of elements simultaneously, is utilized. Most current processors, including Intel ones used in the Ioffe Institute cluster, feature this instruction or extension. In the present case, the SIMD technology is implemented in the form of a set of Advanced Vector Extensions (AVX) instructions. AVX speeds up the process of command decoding, which is the most complex intra-processor stage.

The management of RAM by the processor is improved (these operations are orders of magnitude slower than intra-processor ones), since linear memory sections are fed to the input and the commands themselves are presented in a more compact form. A more efficient utilization of computing ports is achieved, since they no longer compete with each other.

A description of the methods used for RF discharge modeling, a verification of the KITe code, and a comparison of the results of transport calculations for boron and beryllium are being prepared for publication. Figure 2 illustrates the KITe code operation using calculations of the rate of contamination of the first mirror of the ITER divertor Thomson scattering diagnostic system by beryllium atoms sputtered from the first wall as an example. Modeling was performed at the Ioffe Institute cluster on one node (96 threads) and took 32.3 hours of computing time. The input data were the beryllium and deuterium flux densities at the entrance to the diagnostic channel and their energy distributions calculated in the SOLPS-ITER program. A total of $2.79 \cdot 10^8$ test particles (Be, D, and D_2) were included in the KITe calculation; $5.52 \cdot 10^5$ of them were Be atoms. In passage from the channel entrance to the first mirror, the particles were cloned thrice (50, 50, and 30 times with a corresponding reduction in weight). Only 18 test beryllium particles reached the first mirror, but they still provide an opportunity to estimate the rate of contamination of the mirror.

The implementation of the above principles in the KITe code has accelerated significantly the process of kinetic 3D modeling of transport and RF discharge, allowing one to obtain prompt modeling results with sufficient statistics for numerical assessments.

Funding

This study was supported in part by a grant from the Theoretical Physics and Mathematics Advancement Foundation „Basis“.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] L.A. Varshavchik, N.A. Babinov, P.A. Zatylkin, A.A. Chironova, Z.G. Lyullin, Al.P. Chernakov, A.M. Dmitriev, I.M. Bukreev, E.E. Mukhin, A.G. Raazdobarin, D.S. Samsonov, V.A. Senitchenkov, S.Yu. Tolstyakov, I.T. Serenkov, V.I. Sakharov, Plasma Phys. Control. Fusion, **63** (2), 025005 (2021). DOI: 10.1088/1361-6587/abca7e

[2] M. Pharr, W. Jakob, G. Humphreys, *Physically based rendering: from theory to implementation* (Elsevier/Morgan Kaufmann, Amsterdam–Boston, 2004), ch. 4.3.

[3] https://arstechnica.com/features/2000/03/simd/

Translated by D.Safin