DOI: 10.14529/jsfi140305

Co-design of Parallel Numerical Methods for Plasma Physics and Astrophysics

***Boris M. Glinskiy*[1,2], *Igor M. Kulikov*[1,2,3], *Alexey V. Snytnikov*[1,2], *Alexey A. Romanenko*[2], *Igor G. Chernykh*[1,2], *Vitaly A. Vshivkov*[1,2]**

Physically meaningful simulations in plasma physics and astrophysics need powerful hybrid supercomputers equipped with computation accelerators. The development of parallel numerical codes for such supercomputers is a complex scientific problem. In order to solve it the concept of co-design is employed. The co-design is defined as considering the architecture of the supercomputer at all stages of the development of the code. The use of co-design is shown by the example of two physical problems: the interaction of an electron beam with plasma and the collision of galaxies. The efficiency is 92 % with 500 Tesla GPUs at the Lomonosov supercomputer. The test computation involved 160 million of model particles.

Keywords: Co-design, hybrid supercomputers, Particle-In-Cell method, Godunov method, GPU.

Introduction

The main question in plasma simulation is the correct computation of the interaction of plasma particles with the waves in plasma. The waves are the basis of plasma instabilities. The correct simulation of plasma instabilities requires at least 16 grid nodes at the Debye length. The number of model particles cannot be small (less than 100 per cell). The simulations with small number of particles lead to poor quality results in well-known physical situations and to the lack of understanding in the new physical problems [1]. Moreover, the simulation of plasma instabilities is essentially a 3D probem.

The particular physical problem is the simulation of Langmuir waves excitation by the relativistic electron beam with plasma. One must notice that the simulation of the beam-plasma interaction is important for a lot of other plasma physics problems.

Beam-plasma interaction plays an important role in various physical phenomena, such as the transport of relativistic electrons in fast ignition scheme within inertial fusion, the gamma-bursts, solar radio bursts of II and III type, also in formation of collisionless shock waves in space plasma (see review [2] and references therein). Moreover, the collective processes in the beam-plasma system derermine the efficiency of turbulent plasma heating [3, 4] and also of electromagnetic radiation generation [5–8] in mirror traps.

The movement of galaxies in dense clusters turns the collisions of galaxies into an important evolutionary factor, because during the Hubble time an ordinary galaxy may suffer up to 10 collisions with the galaxies of its cluster. Observational and theoretical study of interacting galaxies is an indispensable method for studying their properties and evolution.

One of the most important computational problems within the study of galaxies is the ratio of the characteristics lengths. The size of a galaxy is 10^4 parsec, and the size of a star is about 10^{-7} parsec. Thus the solution of such problems requires the supercomputers from the top part of the Top500 list. Two of the first three (five of the first ten) supercomputers are equipped

[1]Institute of Computational Mathematics and Mathematical Geophysics, Novosibirsk, Russia
[2]Novosibirsk State University, Novosibirsk, Russia
[3]Novosibirsk State Technical University, Novosibirsk, Russia

with computation accelerators (either Nvidia GPUs of Intel Xeon Phi) in the most recent (June 2014) Top500 list.

The first Exaflops supercomputer will be most likely built on the basis of some computation accelerators. At present there are already codes for plasma physics adopted for both GPUs and Intel Xeon Phi accelerators [10, 11]. Still, the design of the codes for the hybrid supercomputers (i.e. the supercomputers equipped with accelerators) is not just a technical question. On the contrary, it is a complex scientific problem. The problem requires co-design of the algorithms at all stages of the solution of the problem from the physical statement of the problem to the software development tools. Within the numerical simulation context co-design is the design of physical and mathematical model, of the numerical method, of the parallel algorithm and of the implementation using the architecure of the hybrid supercomputer efficiently.

The need for top supercomputers in plasma physics exist because of the recent trend of using more precise models instead of simplified ones such as hydrodynamical. The more precise models are based on the Vlasov equation. The Vlasov equation is solved either by the explicit numerical methods in the 6D space of by the Particle-In-Cell (PIC) method [12]. If the PIC method is used then the quantitatively correct physical result may be obtained only with the large number of model particles(from 1 to 10 thousands per cell).

The simulations were conducted of the relativistic electron beam interacting with plasma [32]. These simulations resulted in the correct value of the two-stream instability increment. In order to obtain the correct value the number of model particles was increased up to 1000 per cell. The number of grid nodes is $120 \times 4 \times 4$, domain size is 1.2 (in non-dimensional units). The solution time is about 3 days with 256 Intel Xeon cores (64 4-core Intel Xeon processors). The full 3D simulation will need at least 100 nodes in both transverse dimensions and several times more nodes in the longitudinal dimension keeping the number of model particles per cell, resulting in hundreds of millions of model particles as a whole. It means the increase of computational workload in several thousand times. Let us use 5000. Thus, in order to keep the wallclock time of the simulation (3 days) one has to use 256×5000=1.2 million cores!!! It is the size of the IBM BlueGene/Q, the present number 3. But since it is just one single simulation of at least 10 usually needed to solve a physical problem, it is strongly necessary to do such simulations with lower number of processors and less electric power. That is why using hybrid supercomputers for plasma simulation is inevitable. This requires co-design of the parallel algorithms.

In recent two decades two approaches are being mainly used for the solution of non-stationary 3D astrophysical problems. First, it is the Lagrangian approach mainly represented by SPH method [13, 14] (Smoothed Particle Hydrodynamics) and Eulerian approach with adaptive meshes, or AMR [15] (Adaptive Mesh Refinement). Within the Lagrangian approach the following codes were developed: Hydra [16], Gasoline [17], GrapeSPH [18], GADGET [19] on the basis on SPH method. Within Eulerian approach the following codes were developed: NIRVANA [20], FLASH [21], ZEUS [22], ENZO [15], RAMSES [23], ART [24], Athena [25], Pencil Code [26], Heracles [27], Orion [28], Pluto [29], CASTRO [30], GAMER [31].

The statement of the problem for the galactic objects dynamics is the solution of the equation of the gravitational magnetic gas dynamics for the gas component considering magnetic field and self-gravity and also the solution of the N-body problem for the collisionless component. It is well-known that the N-body problem is hard to solve with supercomputers especially with hybrid architecture. Thus it is necessary to find a method to describe the collisionless component, the method being suitable for efficient parallel implementation. In [33] such a model was developed

on the basis of the first moment of the Boltzmann equation. Such an approach makes it possible to use the same numerical algorithm [34] for the solution of magnetic gas dynamics equations, and also for the solution of the first moments of the Boltzmann equation [35]. This approach was efficiently implemented for two types of hybrid supercomputers: for the supercomputer with GPUs [35], and the supercomputer with Intel Xeon Phi accelerators [11].

1. Co-design of parallel numerical methods

The essence of co-design is the analysis of the efficiency of the parallel implementation at all stages of the development of the algorithm. The stages are the following:

- Physical consideration of the model
- Mathematical statement of the problem: differential equations
- Numerical method
- Parallel algorithm
- Architecture of the supercomputer
- Parallel programming tools

1.1. Co-design of parallel numerical methods for plasma physics

In the present case co-design begins at the stage of the physical consideration of the problem. It is known from experiment that the plasma density modulation cannot exceed 300 %. It means that the number of model particles cannot be increased without a limit. Thus there is no need for dynamic load balancing, and the absence of dynamic balancing improves the reliability and efficiency of the parallel implementation.

At the stage of the numerical method design the field evaluation method was chosen that is built on the basis of Faraday and Ampere laws. In such a way there is no need to solve Poisson equation. Instead, the equations that represent the Faraday and Ampere laws in the numerical form are solved by the Langdon-Lasinski scheme. This results in the field solver with virtually unlimited scalability.

At the stage of supercomputer architecture selection the PIC method details are taken into account. In order to evaluate the new values of position and impulse of a particle it is necessary to know the values of electric and magnetic fields at the present position of the particle. Each of the three components of both electric and magnetic field is stored in a separate 3D array. In such a way six 3D arrays are accessed at each time step for each particle. Since the particles are situated randomly in the domain, the access to the arrays is also unordered. It means that the use of the cache memory can not reduce the computation time. If a part of the field array was fetched to the cache in the process of computation of the particle movement, it would be impossible to use this part of the field array for the computation with the next particle, because it is (most likely) situated in a completely different part of the subdomain. Since the cache memory can not store all the six arrays for fields, one has to access the RAM (Random Access Memory) for computation of the particle movement. And since the performance of the processor is usually limited by the memory bandwidth, it is the memory bandwidth that determines the speed of the computation with particles and the performance of the program as a whole (particles take from 60 % to 90 % of the total time). This fact determines the transition to the supercomputers equipped with GPUs because CUDA has a lot of tools to accelerate memory use for the PIC method.

At the stage selection of software design tools is the folowing. For the PIC method with a very big number of independently processed elements (the model particles) the use of CUDA technology is very efficient. Other parallel technologies for hybrid supercomputers such as OpenCL, OpenMP, OpenACC could be also used but it is CUDA that gives the possibility to use the highest number of parallel processes and to get the highest performance.

The last stage of co-design is the adoptation of the algorithm to the GPU architecture. The traditional PIC method implementation (all the particles stored in one very big array) is unacceptable for GPUs because it prevents the use of the advantages of GPU memory. Thus the particles are distributed between the cells and the computation is conducted as follows: one block of threads for one cell.

1.2. Co-design of parallel numerical methods for astrophysics

In the design of astrophysics models we shall follow the idea of co-design that means the design of parallel numerical technologies considering all the aspects of parallelism. For example, in [33] a new approach was proposed for the simulation of the collisionless component of the galaxies. The approach is based on the first moments of the collisionless Boltzmann equation. This approach has some limitations if it is necessary to trace a single particle and not a group of particles (for example, in the study of planet formation). But for the simulation of interacting galaxies this approach was successfully tested [35]. As it was already said in the introduction, the statement of the problem for the galactic objects dynamics is the solution of the equation of the gravitational magnetic gas dynamics for the gas component considering magnetic field and self-gravity and also the solution of the equations for the first moments of the collisionless Boltzmann equations. During the last two decades for the solution of non-stationary 3D problems

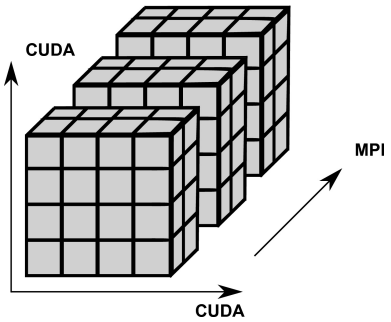

Figure 1. Domain decomposition for GPU-based supercomputers

two approaches are mostly used from the wide range of gas dynamics methods. This is the Lagrangian approach mostly represented by the SPH method and Eulerian approach with the use of adaptive meshes or AMR. The main weak point of SPH is the bad simulation of high gradients and discontinuities, suppression of instabilities, difficult choice of the smoothing kernel and the need for artificial viscosity. A large standalone problem is the local entropy decrease in the SPH method. The main weak point of the mesh-based methods is the non-invariance with respect to rotation, or Galilean non-invariance. This invariance appears because of the mesh. But this problem can be solved by means of the various approaches to the design of numerical schemes [9]. One of the common weak points of both Lagragian and Eulerian approaches is the lack of scalability. In recent time the mixed Lagrangian-Eulerian methods are employed for the solution of astrophysical problems. These methods combine the advantages of both Lagrangian and Eulerian approaches. For a number of years the authors have been developing

the Lagrangian-Eulerian approach on the basis of the combination of the Fluid-In-Cells method and Godunov method [9, 11, 34, 35]. The system of gas dynamics equations is solved in two stages. During the first, Eulerian stage, the system is solved without the advection terms. During the second, Lagrangian stage, the advection is taken into account. The separation into two stages facilitates the elimination of the Galilean non-invariance. The use of Godunov method at the Eulerian stage enables the correct simulation of discontinuos solutions.

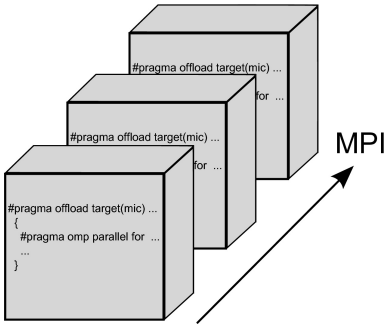

Figure 2. Domain decomposition for Xeon Phi-based supercomputers

The use of uniform Cartesian grids makes it possible to choose an arbitrary Cartesian domain decomposition. In fig. 1 and fig. 2 the two ways of the domain decomposition for hybrid supercomputers are given: for the GPU-based supercomputer (fig. 1) and for the supercomputer equipped with Intel Xeon Phi (fig. 2).

2. Speedup and efficiency

2.1. Co-design for astrophysics: the codes GPUPEGAS and AstroPhi

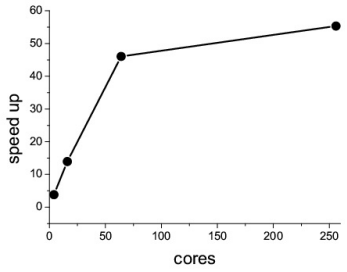

Figure 3. Speedup of the GPUPEGAS code within a single GPU

The use of such domain decomposition enables the most efficient use of the computational accelerators. In fig. 3 and fig. 4 the speedup is given with the single GPU (fig. 3) and a single Intel Xeon Phi accelerator (fig. 4) in the different modes.

With the use of 60 GPUs and 32 Intel Xeon Phi accelerators the parallel efficiency exceeded 95 %. The wallclock time for the AstroPhi code is the following: 1 Xeon core 10,664 sec., 1 Xeon Phi core in offload mode 81,708 sec. (same as native mode), 60 Xeon Phi cores in offload mode 2,960 sec. (28 times speedup), 60 Xeon Phi cores in native mode 1,547 sec. (53 times speedup). For the GPUPEGAS code 1 Xeon core 10,664 sec., 1 GPU core 14,575 sec., 256 GPU cores 0,265 sec. (55 times speedup).

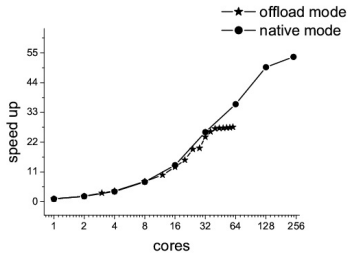

Figure 4. Speedup of the AstroPhi code within a single Intel Xeon Phi

2.2. Co-design for plasma physics: the code for beam-plasma interaction simulation

The use of co-design for the development of the PIC code for plasma simulation resulted in the 42 times speedup with Nvidia Kepler. Speedup here is given with respect to 4-core Intel Xeon processor (all 4 cores employed). The efficiency of the program measured with MVS-100K is over 60 % with up to 2048 Intel Xeon cores (in the figure they are called processors since here each core acts as a stanalone processor, from the MPI point of view). It is shown in fig. 5.

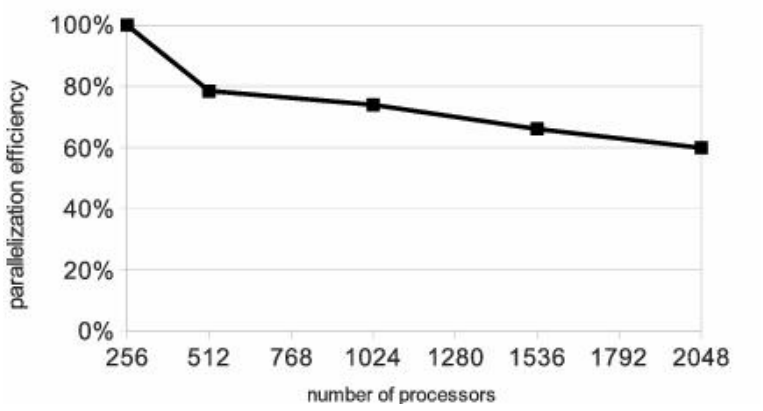

Figure 5. Parallelization efficiency measured with cluster named MVS-100K, Joint Supercomputer Centre of the RAS. The grid size along Y and Z is 64 nodes, grid size along X is equal to the number of processors, 150 particles per cell for all the cases.

The parallel program was developed primarily for the simulation of beam interaction with plasma on large computational grids and with large numbers of particles. It is interesting to know, what will the worktime be for a larger problem. For example, if there is a small problem solved with a small computer in some time, is it possible to solve a larger problem with a larger computer in the same time? Or will the time be larger? How much? That is why parallelization efficiency was computed in the following way.

$$k = \frac{T_2}{T_1} \times 100\% \tag{1}$$

here T_1 - is the computation time with N_1 processors, T_2 is the computation time with N_2 processors. Here the workload of a single processor is constant. This definition of efficiency is inversely proportional to the standard weak scale efficiency, but the above efficiency definition is better since it clearly shows how the things get worse with a lot of processors (with weak scale efficiency one notices that the plot stops growing, but the values are still high, so it seems to be

good, though it is not). In the ideal case the computation time must remain the same (the ideal $k = 100\%$).

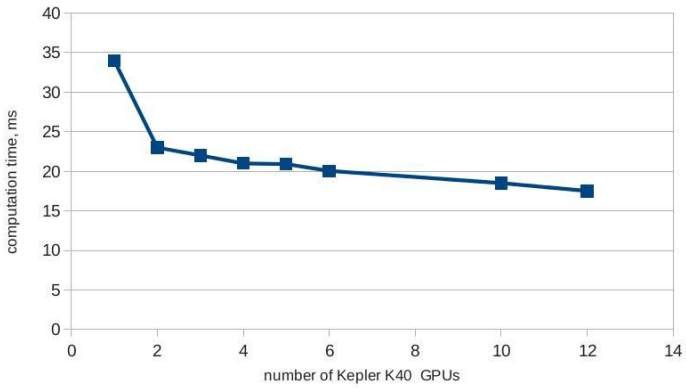

Figure 6. Computation time for the cluster with Kepler K40 GPUs. Here the size of the solved problem is constant and the workload of a single GPU (the number of particles per one GPU) is decreased with the increase of the number of GPUs

Also the speedup was measured for two clusters equipped with GPUs: first cluster with Tesla M2090 and the second with Kepler K40. A single timestep is computed in 34 ms with one Kepler K40 and 17.5 ms with 12 Kepler K40 GPUs (same grid size, model particles are distributed between GPUs), the speedup is shown in fig. 6. Parallel efficiency is over 90 %.

2.2.1. Efficiency for large number of GPUs

The behaviour of the code with large number of GPUs is of special interest. Due to this reason the special perfomance tests were conducted with the hybrid part of the Lomonosov supercomputer. First, the speedup was measured fig. 7. The speedup is relatively small because of the small number of model particles, still one can see that the plot has not reached saturation and is growing. It means that the problems of bigger size will be accelerated more effectively, with higher speedup. At the Lomonosov supercomputer with one Tesla GPU a single timestep

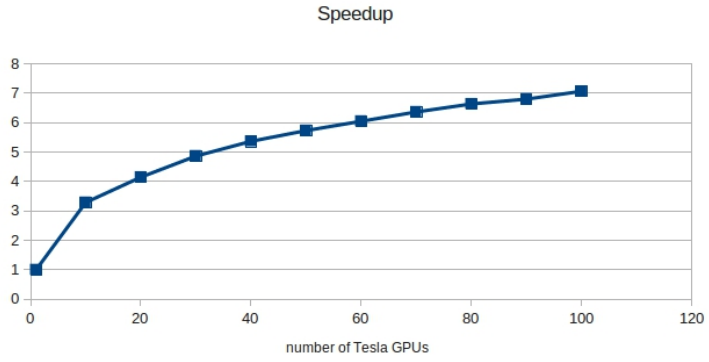

Figure 7. Speedup measured with the Lomonosov supercomputer.Here the size of the solved problem is constant and the workload of a single GPU (the number of particles per one GPU) is decreased with the increase of the number of GPUs

is computed (without communications) in 109 ms.

Now let us turn to the efficiency for the large number of GPUs (that is, up to 500), fig. 8. Here the number of model particles per one GPU (the workload of a GPU) is constant. It means that the size of the whole problem is growing with the number of GPUs. The sense of the test is to show what is the communication overhead for computations with large number of GPUs. Here, is there is no overhead at all and with 100 GPUs a 100 times bigger problem is solved in exactly the same time, the efficiency is 100 %. If the overhead is big, and with large number of GPUs most time is spent for communications then the efficiency will be low.

The efficiency in fig. 8 means that the time spent for a timestep does not change significantly with the number of GPUs. The preparation times such as initialization time, the time for copying the data to GPUs is not considered here. This is because in the real computation which lasts tens of thousands of timesteps all the preparation times will play no role, they will be very low in comparison with total time.

Let us remind that the workload of a single GPU (number of model particles) here is constant. The only communication between the GPUs is just the summation of currents: every GPU evaluates the current in the whole domain only with the particles residing on the GPU, after that one has to sum all the partial currents through all the GPUs. Thus only one MPI_Allreduce operation is performed per timestep and nothing more. Such a small number of MPI operations is because of co-design of the parallel algorithm.One must also notice that the average MPI_Allreduce time measured with Intel Trace Analyzer & Collector does not change significantly with the number of GPUs (at least up to 500): for 10 GPUs the MPI_Allreduce time was 29 ms and for 500 just 41 ms. And this is not because of co-design but because of the Lomonosov hardware and MPI installation.

This result is important not only as a performance test. Each GPU computed the motion of 320000 model particles, it means 160 million model particles as a whole. In such a way the code is really capable of doing physically meaningful simulations (as mentioned in the Introduction) with the Lomonosov supercomputer.

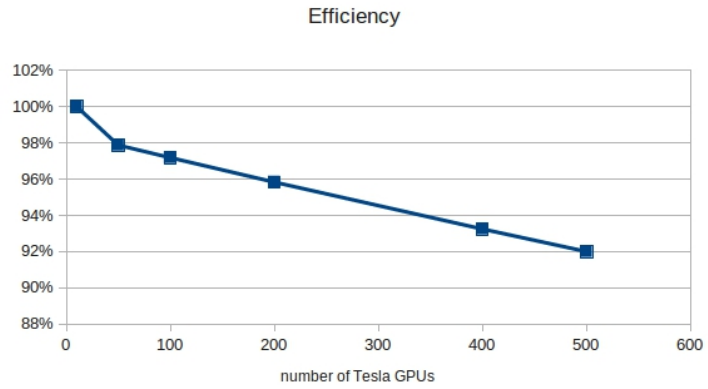

Figure 8. Parallelization efficiency measured with the Lomonosov supercomputer.

3. Simulation results

Finally let us consider the results of the two computational experiments in fig. 9 showing the simlation of the collision of two galaxies according to the described two-phase model. The details of the problem statement and simulation results can be found in [35].

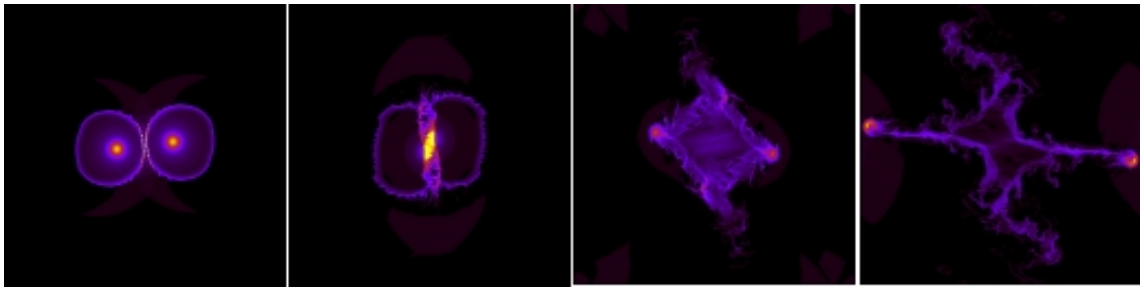

Figure 9. The runaway of galaxies after the central collision

4. Conclusion

The development of the efficient software for the hybrid supercomputers is a standalone complex scientific problem that requires the use of co-design at all stages from the physical statement of the problem to the software development tools. Within the numerical simulation context co-design is the design of physical and mathematical model, of the numerical method, of the parallel algorithm and of the implementation using the architecure of the hybrid supercomputer efficiently.

The use of co-design for the code for beam-plasma interaction simulation gives hope that the full 3D computations will be feasible. If one looks at the efficiency for 2048 processor elements (60 %) and the computation time for a single Kepler K40 GPU (34 ms), then, using about one thousand Kepler GPUs in a computational experiment one might obtain the necesary computational power for a full 3D simulation.

The co-design of the parallel numerical technology of the solution of the astrophysics problems resulted in 55 times speedup within a single GPU and 96% efficiency when using 60 GPUs, and also 27 times speedup in offload mode and 53 times speedup in native mode within a single Intel Xeon Phi and 94 % efficiency when using 32 MICs. Such an efficient use of the accelerators enable to conduct simulations with the grid size 1024^3 in acceptable time and to model "fine" effects in the collision of galaxies.

The present work was supported by the RFBR with grants 14-07-00241, 13-07-00589, 14-01-31088, 14-02-00294, 15-01-508, 15-31-20150 and 14-01-31199 and also by the Grant of the President of the Russian Federation for the support of young scientists number MK 6648.2015.9. We would like to thank Siberian Supercomputer Center department of ICMMG SB RAS for support of this work.

References

1. C.K. Birdsall, A.B Langdon, Plasma Physics via Computer Simulation. CRC Press, 01.10.2004 - 504 p.
2. A. Bret, L. Gremillet, and M.E. Dieckmann, Phys. Plasmas **17**, 120501 (2010).
3. I.V. Timofeev, K.V. Lotov, Phys. Plasmas **13**, 062312 (2006).
4. A. Burdakov, A. Arzhannikov, V. Astrelin, V. Batkin, A. Beklemishev, V. Burmasov, G.

Derevjankin, V. Ivanenko, I. Ivanov, M. Ivantsivskiy, I. Kandaurov, V. Konyukhov, I. Kotelnikov, K. Kuklin, S. Kuznetsov, K. Lotov, I. Timofeev, A. Makarov, M. Makarov, K. Mekler, S. Popov, S. Polosatkin, V. Postupaev, A. Rovenskikh, A. Shoshin, I. Shvab, S. Sinitsky, Yu. Suliaev, V. Stepanov, Yu. Trunyov, L. Vyacheslavov, V. Zhukov, and Eh. Zubairov, Fusion Sci. Technol. **55**(2T), 63 (2009).

5. M.K.A. Thumm, A.V. Arzhannikov, V.T. Astrelin, A.V. Burdakov, I.A. Ivanov, P.V. Kalinin, I.V. Kandaurov, V.V. Kurkuchekov, S.A. Kuznetsov, M.A. Makarov, K.I. Mekler, S.V. Polosatkin, S.A. Popov, V.V. Postupaev, A.F. Rovenskikh, S.L. Sinitsky, V.F. Sklyarov, V.D. Stepanov, Yu.A. Trunev, I.V. Timofeev, L.N. Vyacheslavov, Journal of Infrared, Millimeter, and Terahertz Waves, DOI: 10.1007/s10762-013-9969-3 (2013).
6. V.V. Postupaev, A.V. Burdakov, I.A. Ivanov, V.F. Sklyarov, A.V. Arzhannikov, D.Ye. Gavrilenko, I.V. Kandaurov, A.A. Kasatov, V.V. Kurkuchekov, K.I. Mekler, S.V. Polosatkin, S.S. Popov, A.F. Rovenskikh, A.V. Sudnikov, Yu.S. Sulyaev, Yu.A. Trunev and L.N. Vyacheslavov, Phys. Plasmas **20**, 092304 (2013).
7. I.V. Timofeev, Phys. Plasmas **19**, 044501 (2012).
8. A.V. Arzhannikov and I.V. Timofeev, Plasma Phys. Control. Fusion **54**, 105004 (2012).
9. V.A. Vshivkov, G. Lazareva, A. Snytnikov, I. Kulikov, A. Tutukov "Hydrodynamical code for numerical simulation of the gas components of colliding galaxies" // The Astrophysical Journal Supplement Series. 2011. V. 194, 47. 12 pp.
10. Francesco Rossi, Pasquale Londrillo, Andrea Sgattoni, Stefano Sinigardi, Giorgio Turchetti. Towards robust algorithms for current deposition and dynamic load-balancing in a GPU particle in cell code. 2012. ADVANCED ACCELERATOR CONCEPTS: 15th Advanced Accelerator Concepts Workshop
11. I. Kulikov, I. Chernykh, A. Snytnikov, B. Glinskiy, A. Tutukov "AstroPhi: A code for complex simulation of the dynamics of astrophysical objects using hybrid supercomputers" // Computer Physics Communications. 2015. V. 186. P. 71-80. DOI: 10.1016/j.cpc.2014.09.004
12. Yu.N. Grigoryev, V.A. Vshivkov, M.P. Fedoruk, Numerical particle-in-cell methods. Theory and applications. VSP, 2002.
13. Gingold, R.A., Monaghan, J.J., 1977. Smoothed particle hydrodynamics - Theory and application to non-spherical stars. MNRAS, 181, 375-389.
14. Lucy, L.B., 1977. A numerical approach to the testing of the fission hypothesis. ApJ. 82, 1013-1024
15. O'Shea, B., Bryan, G., Bordner, J., Norman, M., Abel, T., Harkness, R., Kritsuk, A., 2005. Adaptive Mesh Refinement - Theory and Applications. Lect. Notes Comput. Sci. Eng. 41, 341-350.
16. Pearcea, F.R., Couchman, H.M.P., 1997. Hydra: a parallel adaptive grid code. New Astronomy. 2, 411-427.
17. Wadsley, J.W., Stadel, J., Quinn, T., 2004. Gasoline: a flexible, parallel implementation of TreeSPH. New Astronomy. 9, 137-158.
18. Matthias, S., 1996. GRAPESPH: cosmological smoothed particle hydrodynamics simulations with the special-purpose hardware GRAPE. MNRAS. 278, 1005-1017.

19. Springel, V., 2005. The cosmological simulation code GADGET-2. MNRAS. 364, 1105-1134.

20. Ziegler, U., 2005. Self-gravitational adaptive mesh magnetohydrodynamics with the NIRVANA code. A&A. 435, 385-395.

21. Mignone, A., Plewa, T., Bodo, G., 2005. The Piecewise Parabolic Method for Multidimensional Relativistic Fluid Dynamics. ApJ. 160, 199-219.

22. Hayes, J., Norman, M., Fiedler, R. et al., 2006. Simulating Radiating and Magnetized Flows in Multiple Dimensions with ZEUS-MP. ApJS. 165, 188-228.

23. Teyssier, R., 2002. Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. A&A. 385, 337-364.

24. Kravtsov, A., Klypin, A., Hoffman, Y., 2002. Constrained Simulations of the Real Universe. II. Observational Signatures of Intergalactic Gas in the Local Supercluster Region. ApJ. 571, 563-575.

25. Stone, J. et al., 2008. Athena: A New Code for Astrophysical MHD. ApJS. 178, 137-177.

26. Brandenburg, A., Dobler, W., 2002. Hydromagnetic turbulence in computer simulations. Comp. Phys. Comm. 147, 471-475.

27. Gonzalez, M., Audit, E., Huynh P., 2007. HERACLES: a three-dimensional radiation hydrodynamics code. A&A. 464, 429.

28. Krumholz, M.R., Klein, R.I., McKee, C.F., Bolstad, J., 2007. ApJ. 667, 626.

29. Mignone, A. et al., 2007. PLUTO: a Numerical Code for Computational Astrophysics. ApJS. 170, 228.

30. Almgren, A. et al., 2010. CASTRO: A New Compressible Astrophysical Solver. I. Hydrodynamics and Self-gravity. ApJ. 715, 1221.

31. Schive, H., Tsai, Y., Chiueh, T. 2010. GAMER: a GPU-accelerated Adaptive-Mesh-Refinement Code for Astrophysics. ApJ. 186, 457-484.

32. K.V.Lotov, I.V.Timofeev, E.A.Mesyats, A.V.Snytnikov, V.A.Vshivkov. Note on quantitatively correct simulations of the kinetic beam-plasma instability. http://arxiv.org/abs/1410.5617

33. N. Mitchell, E. Vorobyov, G. Hensler "Collisionless Stellar Hydrodynamics as an Efficient Alternative to N-body Methods" // Monthly Notices of the Royal Astronomical Society. 2013. Vol. 428, I. 3. P. 2674-2687.

34. I. Kulikov "PEGAS: Hydrodynamical code for numerical simulation of the gas components of interacting galaxies" // The Book Series of the Argentinian Association of Astronomy. 2013. Vol. 4. P. 91-95.

35. I. Kulikov "GPUPEGAS: A New GPU-accelerated Hydrodynamic Code for Numerical Simulations of Interacting Galaxies" // The Astrophysical Journal Supplement Series. 2014. V. 214, 12. 12 pp. Arxiv: http://arxiv.org/abs/1311.0861