The Parallel Hydrodynamic Code for Astrophysical Flow with Stellar Equations of State

Igor Kulikov[✉1], Igor Chernykh[1], Vitaly Vshivkov[1],
Vladimir Prigarin[2], Vladimir Mironov[3] and Alexander Tutukov[4]

[1] Institute of Computational Mathematics and
Mathematical Geophysics SB RAS, Novosibirsk, Russia
[2] Novosibirsk State Technical University, Novosibirsk, Russia
[3] Lomonosov Moscow State University, Moscow, Russia
[4] Institute of Astronomy RAS, Moscow, Russia
`kulikov@ssd.sscc.ru,chernykh@parbz.sscc.ru,vsh@ssd.sscc.ru,`
`vovkaprigarin@gmail.com,vmironov@lcc.chem.msu.ru,atutukov@inasan.ru`

Abstract. In this paper, a new calculation method for numerical simulation of astrophysical flow at the supercomputers is described. The co-design of parallel numerical algorithms for astrophysical simulations is described in detail. The hydrodynamical numerical model with stellar equations of state (EOS), numerical methods for solving the hyperbolic equations and a short description of the parallel implementation of the code are described. For problems using large amounts of RAM, for example, the collapse of a molecular cloud core, our code upgraded for Intel Memory Drive Technology (IMDT) support. In this paper, we present the results of some IMDT performance tests based on Siberian Supercomputer Center facilities equipped with Intel Optane Memory. The results of numerical experiments of hydrodynamical simulations of the model stellar explosion are presented.

Keywords: Computational Astrophysics, Intel Xeon Phi, Numerical Methods.

1 Introduction

The Type Ia supernova progenitor problem is one of the most exciting problems in astrophysics, requiring detailed numerical modeling to complement observations of these explosions. One possible progenitor is the white dwarf merger scenario, which has the potential to naturally explain many of the observed characteristics of SN Ia [1].

During the last three decades two main approaches have been used to solve astrophysical problems: the Eulerian adaptive mesh refinement (AMR) methods, and the Lagrangian smoothed particle hydrodynamics method (SPH). The Lagrangian approach (SPH method) is used in Hydra [2], Gasoline [3], GrapeSPH [4], GADGET [5] packages. The Eulerian approach (including AMR) is used in NIRVANA [6], FLASH [7], ZEUS [8], ENZO [9], RAMSES [10], ART [11], Athena [12], Pencil Code [13], Heracles [14], Orion [15], Pluto [16], CASTRO [17] codes. Eulerian approach with use of AMR was for the first time used on the hybrid supercomputers equipped with graphic accelerators in a GAMER code [18]. BETHE-Hydro [19], AREPO [20], CHIMERA [21], GIZMO [22] codes are based on combination of Lagrangian and Eulerian approaches. The advantages and disadvantages of the method are described in detail in papers [23, 24], we provide below a brief comparison. In the last decade Lagrangian-Eulerian methods have been actively used to solve astrophysical problems. These methods unite advantages of both the Lagrangian, and Eulerian approaches. During the past years, us has developed the hybrid Eulerian-Lagrangian approach based on operator splitting method for solving the astrophysical problems [23–29]. In this paper we describe the novel approach to the co-design of the numerical model for astrophysical flows with stellar equations of state, that can be implemented on the overscalable Peta- and Exascale supercomputer complex.

Table 1. Advantages and disadvantages of methods

SPH Advantages:	AMR Advantages:
Robustness of the algorithm	Approved numerical methods
Galilean-invariant solution	No artificial viscosity
Simplicity of implementation	Higher order shock waves
Flexible geometries of problems	Resolution of discontinuities
High accurate gravity solvers	No suppression of instabilities
SPH Disadvantages:	AMR Disadvantages:
Artificial viscosity is parameterize	The complexity of implementation
Variations of the smoothing length	The effects of mesh
The problem of shock wave and discontinuous solutions	Problem of the minimal mesh resolution
The problem of discontinuous solutions	Problem of the minimal mesh resolution
Instabilities suppressed	Not galilean-invariant solution
The method is not scalable	The method is not scalable

2 Numerical Method

The method is based on the idea of solving the hydrodynamics equations in two stages. At the first, Eulerian, stage the system is solved without advection terms, at the second, Lagrangian, stage the advective transportation is taken into account. The division into stages allows us to solve problems of Galilean non-invariance and other mesh effects efficiently, and to simulate discontinuous solutions and shock waves correctly. The use of regular meshes does not cause new mesh problems and at the same time allows us to use various Cartesian topology of communication for supercomputers and for accelerators. The equations are written on entropy formulation.

In a couple of works in field of numerical hydrodynamics the entropy equation is used instead of nonconservative equation for internal energy written in conservative form [27]. The undeniable advantage of such approach is guarantee of non-decreasing entropy and effect of negative pressure, and ability to correctly describe supersonic flows. But there are some restrictions in usage of this approach (see the discussion in [30]). The undeniable advantage of this notation of equations of hydrodynamics is in their representation in vector conservative form.

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho\boldsymbol{u}\\ \rho S\\ \varepsilon+\rho\frac{\mathbf{u}^2}{2}\end{pmatrix}+\bigtriangledown\cdot\begin{pmatrix}\rho\boldsymbol{u}\\ \rho\boldsymbol{u}\otimes\boldsymbol{u}+p\\ \rho S\boldsymbol{u}\\ \left(\varepsilon+\rho\frac{\mathbf{u}^2}{2}+p\right)\boldsymbol{u}\end{pmatrix}=\begin{pmatrix}0\\ \rho\bigtriangledown\Phi\\ 0\\ (\bigtriangledown\Phi,\rho\boldsymbol{u})\end{pmatrix} \tag{1}$$

$$\triangle\Phi = 4\pi G\rho \tag{2}$$

where ρ is density of the gas, $\boldsymbol{u}$ is velocity, S is entropy, $p = p(\rho, S)$ is pressure, ε is internal energy, γ is adiabatic index, Φ is gravity, G is gravity constant. For numerical simulation in this papers we should use next form of equation of state:

$$p = (\gamma - 1)\,\varepsilon = S\rho^{\gamma} \tag{3}$$

of course, equation (1) allow to take into account any kind of EOS. On figure (1) block diagram of the numerical scheme was shown. The details of numerical method were described in [30].

3 Parallel Implementation

3.1 Domain Decomposition

Using a uniform mesh in Cartesian coordinates to solve the equations of hydrodynamics makes it possible to use an arbitrary Cartesian topology for decomposition of the computational domain. Such organization of computing has potentially infinity scalability. In this paper, we use the geometric decomposition of the computational domain. On one coordinate, the external one-dimensional cutting takes place using the MPI technology by means FFTW library. (see fig. 2). This decomposition is related to the topology and architecture of all supercomputers that were used for computational experiments.

3.2 Parallel Algorithm

In this section, we will consider the parallel implementation of each box in the computational scheme on figure (1).

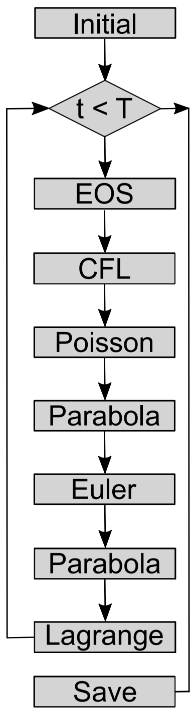

Fig. 1. The block diagram of numerical method

EOS Procedure The blocks are computing in everyone cells an equation of state, and velocity vector:

1. computing pressure by means equation of state (for ideal gas in this a paper)
$$p = p(\rho, S) = S\rho^{\gamma} \tag{4}$$
2. computing velocity vector by means median value of the density on 27-point local stencil
$$\boldsymbol{u} = \frac{\rho\boldsymbol{u}}{[\rho]} \tag{5}$$

CFL Procedure The blocks are computing in everyone cells a time step by means Courant–Friedrichs–Lewy condition:

1. computing local time step for everyone cell by means of equation:
$$\tau = \sqrt{\gamma p/\rho} + \|\boldsymbol{u}\| \tag{6}$$
2. computing global time step for mesh by means MPI_Allreduce function.

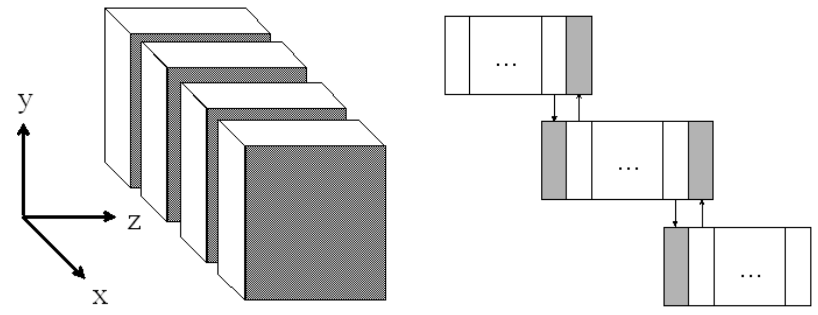

Fig. 2. The geometric domain decomposition

POISSON Procedure The blocks are solving Poisson equation for gravity:

1. formation the right part of the Poisson equation
$$\triangle\Phi = 4\pi G\rho \tag{7}$$
2. on everyone cells are computing the mass of cell:
$$m = \rho \times h^3 \tag{8}$$
where h is size of cells.
3. computing mass of the gas in computational domain by means MPI_Allreduce function.
4. on boundary cell are computing boundary condition by means fundamental solution:
$$\Phi_{boundary} = -\frac{m}{r} \tag{9}$$
where r is a distance from boundary to center of domain.
5. computing σ is the forward Fast Fourier Transform for the density by means fftwnd_mpi FFTW function (on base MPI_Alltoallv function).
6. solving Poisson equation in harmonic space by means equations:
$$\phi_{jmn} = \frac{\frac{2}{3}\pi h^2 \sigma_{jmn}}{1 - \left(1 - \frac{2sin^2\frac{\pi j}{I}}{3}\right)\left(1 - \frac{2sin^2\frac{\pi m}{K}}{3}\right)\left(1 - \frac{2sin^2\frac{\pi n}{L}}{3}\right)} \tag{10}$$
7. computing Φ is the inverse Fast Fourier Transform for the gravity in harmonic space ϕ by means fftwnd_mpi FFTW function (on base MPI_Alltoallv function).

PARABOLA Procedure The blocks are construct parabolas for numerical scheme. We construct the piecewise-parabolic function $q(x)$ on regular mesh with step size is h, on interval $[x_{i-1/2}, x_{i+1/2}]$. In general form the parabola can be written on next equation:
$$q(x) = q_i^L + \xi\left(\triangle q_i + q_i^{(6)}(1-\xi)\right) \tag{11}$$

where q_i is the value in center of cell, $\xi = (x - x_{i-1/2})h^{-1}$, $\triangle q_i = q_i^L - q_i^R$ and $q_i^{(6)} = 6(q_i - 1/2(q_i^L + q_i^R))$ by means conservation laws:

$$q_i = h^{-1} \int_{x_{i-1/2}}^{x_{i+1/2}} q(x)dx \tag{12}$$

to construct $q_i^R = q_{i+1}^L = q_{i+1/2}$ we should use the interpolation function of 4-th order of accuracy:

$$q_{i+1/2} = 1/2(q_i + q_{i+1}) - 1/6(\delta q_{i+1} - \delta q_i) \tag{13}$$

where $\delta q_i = 1/2(q_{i+1} - q_{i-1})$. Input value for construct of the parabola is q_i. Output procedure are all parameters of parabola on each interval $[x_{i-1/2}, x_{i+1/2}]$.

1. We construct $\delta q_i = 1/2(q_{i+1} - q_{i-1})$, and no extreme regularization:

$$\delta_m q_i = \begin{cases} min(|\delta q_i|, 2|q_{i+1} - q_i|, 2|q_i - q_{i-1}|)sign(\delta q_i), \\ (q_{i+1} - q_i)(q_i - q_{i-1}) > 0 \\ 0, (q_{i+1} - q_i)(q_i - q_{i-1}) \leq 0 \end{cases} \tag{14}$$

2. We do the exchange of overlapping domain by means MPI_Send/MPI_Recv functions (details of implementation You can found in appendix).
3. Computing boundary values for parabola:

$$q_i^R = q_{i+1}^L = q_{i+1/2} = 1/2(q_i + q_{i+1}) - 1/6(\delta_m q_{i+1} - \delta_m q_i) \tag{15}$$

4. Reconstruct of the parabola by means equations:

$$\triangle q_i = q_i^L - q_i^R \tag{16}$$

$$q_i^{(6)} = 6(q_i - 1/2(q_i^L + q_i^R)) \tag{17}$$

for the monotone of parabola we should use for boundary values q_i^L, q_i^R next equations:

$$q_i^L = q_i, q_i^R = q_i, (q_i^L - q_i)(q_i - q_i^R) \leq 0 \tag{18}$$

$$q_i^L = 3q_i - 2q_i^R, \triangle q_i q_i^{(6)} > (\triangle q_i)^2 \tag{19}$$

$$q_i^R = 3q_i - 2q_i^L, \triangle q_i q_i^{(6)} < -(\triangle q_i)^2 \tag{20}$$

5. Make a finally upgrade of parameters of parabola

$$\triangle q_i = q_i^L - q_i^R \tag{21}$$

$$q_i^{(6)} = 6(q_i - 1/2(q_i^L + q_i^R)) \tag{22}$$

EULER Procedure The blocks are solve equations on Euler stage:

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho\boldsymbol{u}\\ \rho S\\ \varepsilon+\rho\frac{\mathbf{u}^2}{2}\end{pmatrix}=\begin{pmatrix}0\\ \rho\bigtriangledown\Phi\\ 0\\ (\bigtriangledown\Phi,\rho\boldsymbol{u})\end{pmatrix} \tag{23}$$

For approximation of pressure and velocity we should use next equations:

$$U=\frac{u_L(-\lambda t)+u_R(\lambda t)}{2}+ \tag{24}$$

$$\frac{p_L(-\lambda t)-p_R(\lambda t)}{2}\sqrt{\frac{(\sqrt{\rho_L}+\sqrt{\rho_R})^2}{\frac{\gamma_L+\gamma_R}{2}(\rho_L^{3/2}+\rho_R^{3/2})(p_L\sqrt{\rho_L}+p_R\sqrt{\rho_R})}}$$

$$P=\frac{p_L(-\lambda t)+p_R(\lambda t)}{2}+ \tag{25}$$

$$\frac{u_L(-\lambda t)-u_R(\lambda t)}{2}\sqrt{\frac{\frac{\gamma_L+\gamma_R}{2}(\rho_L^{3/2}+\rho_R^{3/2})(p_L\sqrt{\rho_L}+p_R\sqrt{\rho_R})}{(\sqrt{\rho_L}+\sqrt{\rho_R})^2}},$$

where

$$\lambda=\sqrt{\frac{\frac{\gamma_L+\gamma_R}{2}(p_L\sqrt{\rho_L}+p_R\sqrt{\rho_R})}{\rho_L^{3/2}+\rho_R^{3/2}}} \tag{26}$$

$$q_L(-\nu t)=q_i^R-\frac{\nu t}{2h}\left(\triangle q_i-q_i^6\left(1-\frac{2\nu t}{3h}\right)\right) \tag{27}$$

$$q_R(\nu t)=q_i^L+\frac{\nu t}{2h}\left(\triangle q_i+q_i^6\left(1-\frac{2\nu t}{3h}\right)\right) \tag{28}$$

After computing of Euler stage we do the exchange of overlapping domain by means MPI_Send/MPI_Recv functions.

LAGRANGE Procedure The blocks are solve equations on Lagrange stage:

$$\frac{\partial f}{\partial t}+\nabla\cdot(f\boldsymbol{v})=0 \tag{29}$$

where f is the some conservation variable (density, moment of impulse, entropy or total energy). For computing of the flux $F=f\boldsymbol{v}$ by means $\lambda=|\boldsymbol{v}|$ using equations:

$$F=v\times\begin{cases}f_L(-\lambda t), v\geq 0\\ f_R(\lambda t), v<0\end{cases} \tag{30}$$

where $f_L(-\lambda t)$ and $f_R(\lambda t)$ is piecewise-parabolic function for f and velocity cells interface are computing by means Roe average method:

$$v=\frac{v_L\sqrt{\rho_L}+v_R\sqrt{\rho_R}}{\sqrt{\rho_L}+\sqrt{\rho_R}} \tag{31}$$

After computing of Lagrange stage we do the exchange of overlapping domain by means MPI_Send/MPI_Recv functions (details of implementation You can found in appendix).

3.3 Parallel Code Efficiency

To understand the behavior of the paralleled code, we provide the results of scalability tests. In our case, we did weak scalability tests due to a strong memory bounded problem. Typical arithmetic intensity for PDE (Partially Differential Equations) solvers is ranged from 0.1-1 FLOPS per byte [31–33]. For our tests, we use all Intel Xeon X5670 CPU based nodes of Siberian Supercomputer Center's NKS-30T cluster. Each core of CPU was loaded by 256^3 mesh. Figure 3 shows the efficiency T on a different cores number. The efficiency T is

$$T = \frac{Total_1}{Total_p} \tag{32}$$

where $Total_1$ is computations time on one core when using one core, $Total_p$ is computations time on one core when using p cores. The 93 % efficiency was

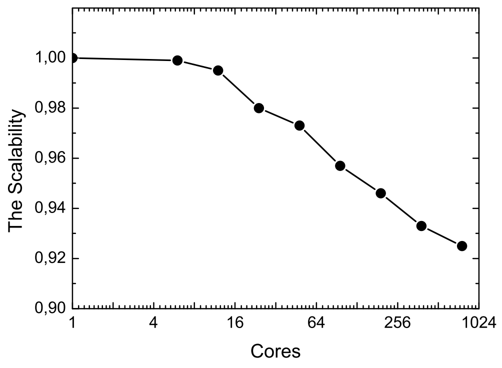

Fig. 3. Scalability of parallel code on the cluster NKS-30T SSCC.

achieved on 768 computational cores (see fig. 3). In numerical experiment the mesh with size 256^3 was computed at each core. As we said before, our problem is the memory bounded problem. Modern CPUs such as Intel Xeon Scalable processors have up to 28 cores with up to 8 sockets on a platform.

From our point of view, the main challenge for all kinds of PDE solvers is to take full computational power from all CPU's cores. In our case, it will be reasonable to put 2048^3 mesh into the RAM of each computational node with 2x28 cores CPU. We need at least 3TB memory for each computational node for holding this size of mesh in DRAM. It is hard to find this kind of supercomputer's nodes, but we hope that the evolution of Intel Memory Drive Technology (IMDT) [36] and Intel Optane technologies can help to build this kind of systems. In the next chapter, we will present the first results of our solver tests on Intel Optane memory.

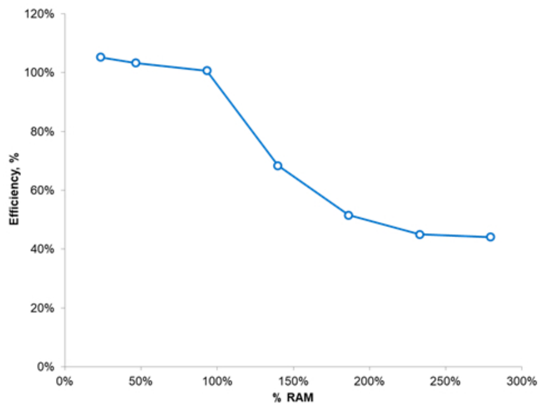

Fig. 4. An efficiency of the Lagrangian stage using Intel Optane SSDs.

3.4 Intel®Optane™Performance Tests

In the recent years the capacity of system memory for high-performance computing (HPC) systems has not been kept with the pace of the increased CPU power. The amount of system memory often limits the size of problems that can be solved. System memory is typically based on DRAM. DRAM prices have significantly grown up in the recent year. In 2017, DRAM prices were growing up approximately 10-20% quarterly [34]. As a result, memory can contribute up to 90% to the cost of the servers. A modern memory system is a hierarchy of storage devices with different capacities, costs, latencies, and bandwidths intended to reduce price of the system. It makes a perfect sense to introduce yet another level in the memory hierarchy between DRAM and hard disks to drive price of the system down. SSDs are a good candidate because they are cheaper than DRAM up to 10 times. What is more important, over the last 10 years, SSD based on NAND technology emerged with higher read/write speed and Input/Ouput Operations per Second (IOPS) metric than hard disks. Siberian Supercomputer Center SB RAS has one RSC Tornado [35] node equipped with Intel Optane memory (2xPCI-E 375GB P4800X SSDs). Intel Optane SSDs are working as an expansion of 128 GB node's DRAM. IMDT is working as a driver, and transparently integrates the SSDs into the memory subsystem and makes it appear like DRAM to the OS and applications. On the Fig. 4 we presented the efficiency plot of Lagrangian step of the gas dynamic simulation, which is the most time-consuming step ($>$ 90% compute time). This step describes the convective transport of the gas quantities with the scheme velocity for our problem. The number of FLOP/byte is not very high and the efficiency plot follows the trend we described above. We observe a slow decrease in efficiency down to $\approx$50% when the data does not fit into DRAM cache of IMDT. Otherwise, the efficiency is close to 100%. We achieved this results on an unoptimized code.

One of the approaches for improving the performance of IMDT based systems is the more effective usage of Intel Optane's cache which is held in DRAM

4 Numerical Simulation for Explosion Model

As a model problem of a supernova explosion, let us consider a hydrostatically equilibrium density profile:

$$\rho(r) = \begin{cases} 2r^3 - 3r^2 + 1, & r \leq 1, \\ 0, & r > 1. \end{cases} \tag{33}$$

and pressure profile:

$$p(r) = \begin{cases} -\pi r^8/3 + 44\pi r^7/35 - 6\pi r^6/5 - 4\pi r^5/5 & \\ \qquad +8\pi r^4/5 - 2\pi r^2/3 + \pi/7, & r \leq 1, \\ 0, & r > 1. \end{cases} \tag{34}$$

In center of domain injected pressure is equal to $P = 500$. For the velocity the Gauss distribution was used. The results of the simulation are shown in the figure (5). The result of numerical experiments have a dense ring (like a Sedov

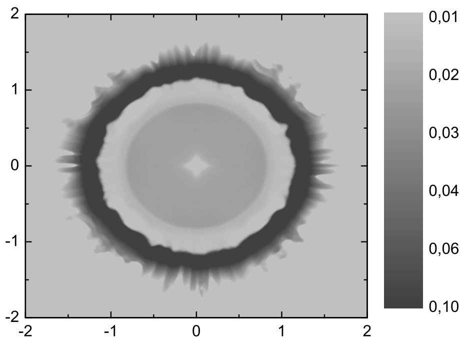

Fig. 5. The nondimensional density in time is equal to t = 0.01

test). A small velocity perturbation leads to unevenness of the ring and has a potential for rupture.

5 Conclusion

In this paper, a novel computation technique for numerical simulations of astrophysical flow at the supercomputers was described. The co-design of parallel numerical algorithms for astrophysical simulations was described in detail.

The hydrodynamical numerical model with stellar equations of state, numerical methods for solving the hyperbolic equations and a brief description of the parallel implementation of the code was described. This code is a classical memory bounded partially differential equations solver. We achieved more than 93% weak scalability for 1024 CPU cores. For detailed numerical simulation of our problem, we need to use a large amount of RAM (more than 1TB) on each node. At this moment, it is hard to find supercomputing facilities with large RAM computational nodes. The evolution of Intel Optane SSDs as well as Intel Memory Drive Technology giving expectations that future systems will be equipped with DRAM and Intel Optane SSDs which are working as an extension of DRAM. First tests of Intel Optanes shows that unoptimized PDE solver working twice slower on SSDs than a DRAM, but it is possible to optimize memory operations for improving performance. One of the approaches for improving the performance of IMDT based systems is more effective usage of Intel Optane's cache which is held in DRAM. This optimization will be done in our future work.

This work is a part of the common joint "Hydrodynamical Numerical Modelling of Astrophysical Flow at the Peta- and Exascale", developed by our team at the Siberian Supercomputer Center ICMMG SB RAS in collaboration with the researchers from the Institute of Astronomy RAS, the Institute of Astronomy of Vienna University and the Southern Federal University, supported by RSC Group.

6 Acknowledgements

The research work was supported by the Grant of the Russian Science Foundation (project 18-11-00044).

References

1. Katz, M., Zingale, M., Calder, A., Douglas Swesty, F., Almgren, A., Zhang, W.: White dwarf mergers on adaptive meshes. I. methodology and code verification. The Astrophysical Journal. 819, 2, 94 (2016)
2. Pearcea, F.R., Couchman, H.M.P.: Hydra: a parallel adaptive grid code. New Astronomy. 2, 411 (1997).
3. Wadsley, J.W., Stadel, J., Quinn, T.: Gasoline: a flexible, parallel implementation of TreeSPH. New Astronomy. 9, 137 (2004).
4. Matthias, S.: GRAPESPH: cosmological smoothed particle hydrodynamics simulations with the special-purpose hardware GRAPE. Monthly Notices of the Royal Astronomical Society. 278, 1005 (1996).
5. Springel, V.: The cosmological simulation code GADGET-2. Monthly Notices of the Royal Astronomical Society. 364, 1105 (2005).
6. Ziegler, U.: Self-gravitational adaptive mesh magnetohydrodynamics with the NIRVANA code // Astronomy & Astrophysics. 435, 385 (2005).
7. Mignone, A., Plewa, T., Bodo, G.: The Piecewise Parabolic Method for Multidimensional Relativistic Fluid Dynamics. The Astrophysical Journal. 160, 199 (2005).

8. Hayes, J., Norman, M., Fiedler, R., et al.: Simulating Radiating and Magnetized Flows in Multiple Dimensions with ZEUS-MP. The Astrophysical Journal Supplement Series. 165, 188 (2006).
9. O'Shea, B., Bryan, G., Bordner, J., Norman, M., Abel, T., Harkness, R., Kritsuk, A.: Adaptive Mesh Refinement - Theory and Applications. Lectures Notes of Computer Science Engineering. 41, 341 (2005).
10. Teyssier, R.: Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. Astronomy & Astrophysics. 385, 337 (2002).
11. Kravtsov, A., Klypin, A., Hoffman, Y. Constrained Simulations of the Real Universe. II. Observational Signatures of Intergalactic Gas in the Local Supercluster Region. The Astrophysical Journal. 571, 563 (2002).
12. Stone, J., Gardiner, T., Teuben, P., Hawley, J., Simon, J.: Athena: A New Code for Astrophysical MHD. The Astrophysical Journal Supplement Series. 178, 137 (2008).
13. Brandenburg, A., Dobler, W.: Hydromagnetic turbulence in computer simulations. Computer Physics Communications. 147, 471 (2002).
14. Gonzalez, M., Audit, E., Huynh, P.: HERACLES: a three-dimensional radiation hydrodynamics code. Astronomy & Astrophysics. 464, 429 (2007).
15. Krumholz, M.R., Klein, R.I., McKee, C.F., Bolstad, J.: Equations and Algorithms for Mixed-frame Flux-limited Diffusion Radiation Hydrodynamics. The Astrophysical Journal. 667, 626 (2007).
16. Mignone, A., Bodo, G., Massaglia, S., Matsakos, T., Tesileanu, O., Zanni, C., Ferrari, A.: PLUTO: a Numerical Code for Computational Astrophysics. The Astrophysical Journal Supplement Series. 170, 228 (2007).
17. Almgren, A., Beckner, V., Bell, J., et al.: CASTRO: A New Compressible Astrophysical Solver. I. Hydrodynamics and Self-gravity. The Astrophysical Journal. 715, 1221 (2010).
18. Schive, H., Tsai, Y., Chiueh, T.: GAMER: a GPU-accelerated Adaptive-Mesh-Refinement Code for Astrophysics. The Astrophysical Journal. 186, 457 (2010).
19. Murphy, J., Burrows, A.: BETHE-Hydro: An Arbitrary Lagrangian-Eulerian Multidimensional Hydrodynamics Code for Astrophysical Simulations. The Astrophysical Journal Supplement Series. 179, 209 (2008).
20. Springel, V.: E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. Monthly Notices of the Royal Astronomical Society. 401, 791 (2010).
21. Bruenn, S., Mezzacappa, A., Hix, W., et al.: 2D and 3D core-collapse supernovae simulation results obtained with the CHIMERA code. Journal of Physics. 180, 012018 (2009).
22. Hopkins, P.: A new class of accurate, mesh-free hydrodynamic simulation methods. Monthly Notices of the Royal Astronomical Society. 450, 53 (2015).
23. Kulikov, I.: GPUPEGAS: A New GPU-accelerated Hydrodynamic Code for Numerical Simulations of Interacting Galaxies. The Astrophysical Journal Supplement Series. 214, 1, 12 (2014)
24. Kulikov, I.M., Chernykh, I.G., Snytnikov, A.V., Glinskiy, B.M., Tutukov, A.V.: AstroPhi: A code for complex simulation of dynamics of astrophysical objects using hybrid supercomputers. Computer Physics Communications. 186, 71-80 (2015)
25. Tutukov, A., Lazareva, G., Kulikov, I.: Gas Dynamics of a Central Collision of Two Galaxies: Merger, Disruption, Passage, and the Formation of a New Galaxy. Astronomy Reports. 55, 9, 770-783 (2011)

26. Vshivkov, V., Lazareva, G., Snytnikov, A., Kulikov, I.: Supercomputer Simulation of an Astrophysical Object Collapse by the Fluids-in-Cell Method. Lecture Notes of Computer Science, 5698, 414-422 (2009)
27. Godunov, S., Kulikov, I.: Computation of Discontinuous Solutions of Fluid Dynamics Equations with Entropy Nondecrease Guarantee. Computational Mathematics & Mathematical Physics. 54, 1012-1024 (2014)
28. Vshivkov, V., Lazareva, G., Snytnikov, A., Kulikov, I., Tutukov, A.: Hydrodynamical code for numerical simulation of the gas components of colliding galaxies. The Astrophysical Journal Supplement Series. 194, 47, 1-12 (2011)
29. Vshivkov, V., Lazareva, G., Snytnikov, A., Kulikov, I., Tutukov, A.: Computational methods for ill-posed problems of gravitational gasodynamics. Journal of Inverse and Ill-posed Problems. 19, 1, 151-166 (2011)
30. Kulikov, I., Vorobyov, E.: Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows. The Journal of Computational Physics. 317, 318-346 (2016)
31. Roofline Performance Model. `https://crd.lbl.gov/departments/computer-science/PAR/research/roofline/`
32. Glinskiy, B., Kulikov, I., Chernykh, I.: Improving the Performance of an AstroPhi Code for Massively Parallel Supercomputers Using Roofline Analysis. Communications in Computer and Information Science, vol. 793, pp. 400-406 (2017)
33. Kulikov, I., Chernykh, I., Glinskiy, B., Protasov, V.:An Efficient Optimization of Hll Method for the Second Generation of Intel Xeon Phi Processor. Lobachevskii Journal of Mathematics, 2018, Vol. 39, No. 4, pp. 543-550 (2018)
34. Markets analytics. `https://www.trendforce.com/`
35. RSC Tornado architecture. `http://www.rscgroup.ru/en/our-solutions`
36. Intel Memory Drive Technology. `https://www.intel.ru/content/www/ru/ru/solid-state-drives/optane-ssd-dc-p4800x-mdt-brief.html`