Content of ontology for solving compute-intensive problems of the cosmic plasma hydrodynamics

Anna Sapetina¹, Boris Glinskiy¹ and Galina Zagorulko²

¹ Institute of Computational Mathematics and Mathematical Geophysics SB RAS, 6 Akad. Lavrentieva pr., Novosibirsk, Russia
² A.P. Ershov Institute of Informatics System SB RAS, Novosibirsk, 6 Akad. Lavrentieva pr., Novosibirsk, Russia
E-mail: afsapetina@gmail.com

Abstract. Modern mathematical modeling on supercomputer systems requires that researchers have both a view of the main aspects of a studied phenomena and a knowledge base of numerical methods and supercomputer technologies for solving natural science problems. Development of intelligent support systems for solving such problems can greatly facilitate the researcher work by providing convenient tools for constructing an efficient scheme of problem solution on a target computational architecture and, if possible, implementing it using already developed program codes.

This paper describes an implementation of this idea for the compute-intensive problems of the cosmic plasma hydrodynamics. A knowledge base has been developed, including ontological descriptions of studied astrophysical phenomena, their mathematical representations, numerical methods for solving set problems and supercomputer architectures that can be used for calculations. The paper presents patterns describing the main classes of the constructed ontology and their instances. The ontology can be expanded by a generation system of computational codes for the selected schemes of a problem solution.

1. Introduction

The main method of theoretical study of astrophysical objects and processes is mathematical modeling. A large number of codes have been developed to solve astrophysical problems. Two approaches are mainly used to solve dynamic 3D astrophysical problems. These are Lagrangian approach presented by Smoothed Particle Hydrodynamics (SPH) [1, 2] and Eulerian approach using Adaptive Mesh Refinement [3].

In terms of Lagrangian approach based on the SPH method, such software packages have been developed as Gasoline [4], GrapeSPH [5], GADGET [6]. In terms of Eulerian approach, such software packages have been developed as ZEUS [7], RAMSES [8], Athena [9], Pluto [10], CASTRO [11]. The combination of Lagrangian and Eulerian approaches is an unconditional trend in modern computational astrophysics. BETHE-Hydro [12], AREPO [13], GIZMO [14], PEGAS [15], GPUPEGAS [16], AstroPhi [17] packages have been developed based on this approaches combination.

Each of these codes is focused on solving a certain class of problems and is implemented for a certain supercomputer architecture. The use of these codes for solving many problems requires improvement and the transition to a new supercomputer architecture further complicates the
task. For example, the MPI-AMRVAC code has some flexibility for solving the problems of stellar evolution [18], but nevertheless it has a number of limitations when going to problems of the galaxies evolution or supernova explosions. An intelligent system for generating astrophysical code has not yet been created, although there are attempts to develop such system [19, 20].

Thus, there is a set of astrophysical codes implemented on various computing architectures using various software tools and packages. It can be difficult for a researcher solving an original problem to understand the existing set of methods and computational architectures for constructing an efficient solution scheme. This requires intelligent support.

In papers [21, 22], a conception of intelligent support for solving compute-intensive problems of mathematical physics on supercomputers is considered. The conceptual framework for such support is provided by the following components: a knowledge base, an information-analysis web resource, an expert system, a library of software components, a simulation modeling module and a code generation module. This concept is based on the ontological approach. The use of an ontology of numerical methods and parallel algorithms and an ontology of parallel architectures and technologies with specified inference rules allow one to select an efficiency numerical method, parallel algorithm and computational architecture for solving a problem.

In this work, we specify and expound the structure of the ontology proposed in [21, 22] by developing a class hierarchy and defining the concept properties, and we also fill the developed classes with necessary instances for solving compute-intensive problems of the cosmic plasma hydrodynamics.

2. Ontology structure
Sharing common understanding of the structure of information among people or software agents is one of the most general goals of ontology development. Detaching the main stages of solving compute-intensive problems of mathematical physics allows us to propose the ontology structure shown in figure 1.

![Ontology Structure Diagram](image)

**Figure 1.** The top-level ontology for solving compute-intensive problems of mathematical physics.
In this work, we focus on the Astrophysical Problems, although in the presented scheme, there can be any others instead. It is assumed that a potential ontology user is sufficiently familiar with the physical features of the problems; otherwise, the Astrophysical problem class can be additionally described by structuring class objects through the concepts of the physical phenomenon, object and law.

For the numerical solution of an Astrophysical Problem, it is necessary to formalize it using a Mathematical Model, which we associate with an Equation System describing the problem, since the equation system selection basically determines the model in our understanding. The Equation System is solved by a suitable Numerical Method implemented by a Parallel Algorithm. The Parallel Algorithm is optimized for the Architecture of a target computer, on which a Code that encodes the Parallel Algorithm is executed. The Code uses Parallel Programming Technologies assigned by the Target Architecture and is tested on a Test Problem, which is a subclass of the Astrophysical Problem class. Thus, the scheme for solving compute-intensive astrophysical problems, taking into account supercomputer architectures, is closed.

3. Class hierarchy, properties and instances

Following the co-design methodology [23], when developing high-performance scientific codes, it is necessary to take into account features of the computational architecture at all stages of solving the problem. Therefore, it is important to single out the properties of each developed ontology class that are necessary for this. These also will be used in inference rules for constructing an efficient scheme for solving the problem. It is also important to develop the ontology that is intuitive, easily extensible, and easy to maintain.

Guided by these principles, an ontological description of the top-level classes presented in figure 1 has been compiled, this are shown schematically in figures 2–6.

All schemes are made in a single notation format:

- Top-level classes are marked in orange;
- Subclasses are marked in yellow;
- Instances are marked in purple;
- Object properties are marked in blue;
- Data properties are marked in green;
- The light gray rectangles represent the data-value type;
- The dark gray rectangles represent the enumerated data type.

All subclasses of a class are expected to inherit the properties of that class. Let’s give comments for each description.

3.1. The astrophysical problem

The set of problems under consideration dictates the content of all classes of the ontology. In figure 2 subclasses of Astrophysical problems are distinguished, in which the hydrodynamics of cosmic plasma is the main process. For example, there is a subclass of problems related to the Stars Evolution, an instance of which is the study of a supernovae Ia explosion. It can also include other classes of Astrophysical problems in which hydrodynamic processes do not play a key role (these are marked with a dotted line), for example, Formation of Planetary Systems, but to describe their solution, additional instances will need to be added to main classes of the ontology.

Separately, we note the subclass of Test Problems, in which problems with description of their exact solution (for codes testing) are presented.
3.2. The equation system
Figure 3 shows the typical components of the Equation System of mathematical physics: Partial Differential Equations (their subclasses are the Parabolic, Elliptic and Hyperbolic equations or reducible to them), State Equations and Boundary Conditions.

The initial conditions can be very variable, so we leave them as a specification of a certain problem at the discretion of the ontology user. In the case of Test Problems, the initial configuration was added to their description to close the problem statement and obtain a numerical solution (figure 2).

Since the initial problems of the space plasma hydrodynamics are unstable and their formulations are incorrect, a Solution Adjustment subclass containing additional equations to control the numerical solution correctness is also added.

The main instances for solving problems of cosmic plasma hydrodynamics are introduced into the Equation System class. As the list of problems expands, we can easily extend this class with other subclasses of equations. For example, to account for chemical kinetics, it will be necessary to add a subclass of Ordinary Differential Equations (ODE).

3.3. The numerical method
Figure 4 presents main data properties to describe the Numerical Method for solving astrophysics problems. They allow us to get necessary information about class instances with regard to accuracy and computational complexity.

Also the class hierarchy and basic methods for the numerical solution of elliptic equations (Poisson equation) and hyperbolic equations (basic conservation laws) are presented on the basis of the widely used Godunov Method. Approaches to solving the Riemann problem (the main structural element of Godunov method) and high-order realizations are singled out.
Figure 3. The Equation System class description with instances.

Figure 4. The Numerical Method class description with instances.
3.4. The parallel algorithm and the code

The further we move from the formulation of the problem to its parallel implementation, the more important it is to highlight the specific properties of classes that allow us to estimate the efficiency of a selected problem solution scheme for computing on a target architecture. This is especially important for describing the Parallel Algorithm and Code classes. For brevity, only the Code description scheme is presented (figure 5).

For a qualitative presentation of the Parallel Algorithm, in addition to the Description and Implementation Scheme, we separately highlight such data properties as Type of Parallelism (distributed or dependent), Data Structure, dimension of the computational Domain Decomposition and Overlap Width, Distribution of operations among processes or threads, Operations Number and Transactions Number. All of them significantly affect the software implementation, performance and scalability of the Code associated with the algorithm.

Efficiency of the Code can be estimate by Performance, Scalability and Energy Efficiency. Also the Code can be verified by checking the quality of the numerical solution on Test Problems. Instances of the Code class can be both existing multifunctional codes (for example, AstroPhi code) and some of Program Modules that implement a certain Numerical Method for a specific type of Computing Device and/or Memory Access Type.

In paper [20] authors propose a concept for creating a system of generating the grid parallel codes for the numerical solution of cosmic plasma hydrodynamic problems based on a generalized representation of the magnetic hydrodynamics equations using the Godunov method. The code generation is based on the comprehensive structure of the “main” function, which maps the upper level of the algorithm, combined with program modules that correspond to the problem solution scheme. The generated code allows one to simulate the dynamics of cosmic plasma for a wide range of problems.

Figure 5. The Code class description.

3.5. The target architecture

The structure of Target architecture class (figure 6) allows us to describe instances of different technological complexity, ranging from some of Computing Devices (KNL, Broadwell, Tesla
M2090) to hybrid supercomputer systems with a large number of Nodes (NKS-1P cluster of the Siberian Supercomputer Center). Specific values or estimates of numerical data properties (such as Bandwidth, Memory Size, Peak Performance, etc.) make it possible to understand whether instances of the class satisfy demands that the Code imposes on the Architecture. Due to the wide variety of computing systems, it is necessary to head on modern trends in the development of supercomputer technologies, focusing on the Top500 list (www.top500.org).

Figure 6. The Target Architecture and The Parallel Programming Technology classes description with instances.

4. Conclusion
The paper provides a detailed description of the developed ontology of intelligent support for solving compute-intensive problems of mathematical physics and examples of its class instances in the field of space plasma hydrodynamics. Brief comments on the structures of the considered ontology classes are given.
The developed ontology allows one to build efficient schemes for solving a wide range of compute-intensive astrophysics problems taking into account the architecture of a supercomputer. Ontology is designed in such a way that other problems of mathematical physics can be added easily to the ontology.

Acknowledgments
This research was conducted within the framework of the budget project No. 0315-2019-0009, for ICMMG SB RAS, and supported in part by the Russian Foundation for Basic Research [grant No. 19-07-00085].

References
[1] Gingold R A and Monaghan J J 1977 Mon. Not. R. Astron. Soc. 181 375–89
[2] Lucy L B 1977 Astrophys. J. 82 1013–24
[3] O’Shea B, Bryan G, Bordner J, Norman M, Abel T, Harkness R and Kristsuk A 2005 Lect. Notes Comput. Sci. Eng. 41 341–50
[4] Wadsley J W, Stadel J and Quinn T 2004 New Astronomy 9 137–58
[5] Matthias S 1996 Mon. Not. R. Astron. Soc. 278 1005–17
[6] Springel V 2005 Mon. Not. R. Astron. Soc. 364 1105–34
[7] Hayes J, Norman M, Fiedler R, et al 2006 Astrophys. J., Suppl. Ser. 165 188–228
[8] Teyssier R 2002 A&A 385 337–64
[9] Stone J M, Gardiner T A, Teuben P, Hawley J F and Simon J B 2008 ApJS 178 137–77
[10] Mignone A, Bodo G, Massaglia S, Matsakos T, Tesileanu O, Zanni C and Ferrari A 2007 ApJS 170 228–42
[11] Almgren A, et al 2010 Astrophys. J. 715 1221–38.
[12] Murphy J W and Burrows A 2008 ApJS 179 209–41
[13] Springel V 2010 Mon. Not. R. Astron. Soc. 401 791–851
[14] Hopkins P F 2015 Mon. Not. R. Astron. Soc. 450 53–110
[15] Vshivkov V, Lazareva G, Snytnikov A, Kulikov I and Tutukov A 2011 Astrophys. J., Suppl. Ser. 194 1–12
[16] Kulikov I 2014 ApJS 214 12
[17] Kulikov I M, Chernykh I G, Snytnikov A V, Glinskiy B M and Tutukov A V 2015 CPC 186 71–80
[18] Goedbloed J, Keppens R and Poedts S 2003 Space Science Reviews 107 63
[19] Frutos-Alfaro F and Carboni-Mendez R 2016 Rev. Mat 23 41–61
[20] Sapetina A, Ulyanichev I and Glinskiy B 2019 J. Phys.: Conf. Ser. 1336 012012
[21] Glinskiy B, Zagorulko Y, Zagorulko G, Kulikov I and Sapetina A 2019 CCIS 1129 427–38
[22] Zagorulko G, Zagorulko Y, Glinskiy B and Sapetina A 2019 CCIS 1093 363–75
[23] Glinskiy B, Kulikov I, Snytnikov A, Romanenko A, Chernykh I and Vshivkov V 2014 Supercomputing Frontiers and Innovations 3 88–98