The automated construction of a scheme for solving compute-intensive problems based on the ontological approach and Semantic Web technologies

B M Glinskiy1,2, A F Sapetina1,2, A V Snytnikov3,4, Y A Zagorulko5 and G B Zagorulko5

1 Institute of Computational Mathematics and Mathematical Geophysics SB RAS, 6 Akad. Lavrentieva pr., Novosibirsk, Russia
2 Novosibirsk State University, 1 ul. Pirogova, Novosibirsk, Russia
3 Institute of Automation and Electrometry SB RAS, 1 Akad. Koptyuga pr., Novosibirsk, Russia
4 Novosibirsk State Technical University, 20 K. Marksa pr., Novosibirsk, Russia
5 A.P. Ershov Institute of Informatics Systems SB RAS, 6 Akad. Lavrentieva pr., Novosibirsk, Russia

E-mail: gal@iis.nsk.su

Abstract. This paper describes the tools for supporting researchers in the development of a parallel code. The tools are based on the ontology of the knowledge area "Support for solving compute-intensive problems of mathematical physics on supercomputers". The main result of these tools operation is a scheme for solving the problem, built according to its specification provided by the user. The scheme includes the most suitable mathematical models for solving the problem, numerical methods, algorithms, and parallel architectures, links to available fragments of a parallel code that the user can use when developing his own code. The scheme construction is carried out on the basis of ontology and expert rules built using the Semantic Web technology.

1. Introduction

Solving compute-intensive problems on supercomputers is an urgent task that makes researchers highly qualified both in the field of their main scientific and practical interests and in the field of parallel computing, supercomputer technologies, and architectures. For this problem solution, various tools are being developed to support the interaction of users with supercomputers: information and reference systems, systems for generating parallel codes, systems for supporting parallelization of program codes, and systems for solving a limited (specific) class of problems on supercomputers.

This paper describes software and methodological tools designed to assist researchers in the development of a parallel code. These tools are developed within the framework of an intelligent system (IS) [1]. The purpose of the IS is to systematize information about the area of knowledge "Support for solving compute-intensive problems of mathematical physics on supercomputers" and to provide the content-based access to this information, to publications and resources dealing with this problem, to available parallel codes and development of the tools for such codes. The tools proposed are based on the ontology of the considered area of knowledge. The main result of the system is a scheme for solving the problem of mathematical physics on supercomputers, built according to the
specification of the problem provided by the user. The scheme includes the most suitable mathematical models for solving the problem, numerical methods, algorithms, parallel technologies and architectures, links to available fragments of a parallel code that the user can use when developing his own code. The construction of the scheme is performed based on expert rules. Semantic Web technologies to construct ontology and rules are used.

The paper is organized as follows. Section 2 describes related published works. Section 3 presents the IS, its architecture, and a brief description of the system components. Section 4 describes the process of constructing a scheme for solving compute-intensive problems on supercomputers. The rules used in the construction of the scheme, and the rules intended for populating the knowledge base, are presented in Section 5. The conclusion contains a summary of the results of this study and emphasizes its merits.

2. The related published works

Let us consider published works that have similarities in the general statements of the problem (simplification or automation of the development of parallel programs), in the main method used (the use of a set of expert competencies to support the user with the development of a parallel program), or in the implementation of actions performed at one of the levels of the ontological approach (architecture, software, level of a mathematical model).

The fragmented programming system LuNA [2] is based on the concept of active knowledge, i.e. automatic or semi-automatic conversion of a set of expert knowledge about the subject knowledge a correctly working program for a supercomputer. The active knowledge concept allows us at the conceptual level to consider the LuNA system as an analog of the ontology-based approach presented in this paper.

The LuNA project is aimed at excluding parallel programming from the development of large-scale numerical models. From this point of view, the LuNA project solves only part of the problems that are posed and solved within the ontological approach framework. Namely, the LuNA system works at the level of software implementation and partly at the level binding to the architecture of a supercomputer [3] but does not affect the level of constructing a mathematical model. This is the main feature and an advantage of the ontological approach.

Let us also add some details regarding working with the LuNA system. This system allows one to take as input a computational model, a set of input and output variables, and the desired computational algorithm specified as a recursively enumerable set of functional terms. The required program that implements the algorithm derived from the presented functional terms is automatically created.

An analog of the ontological approach, not from a conceptual point of view, but in terms of the presence of an agreed approach to the implementation of the main stages of a large-scale computational experiment, is the methodology of the basic modeling system (BSM [4,5,6]). A computational experiment using the BSM methodology consists of the following stages: geometric and functional modeling, grids, approximation of the original equations, and solving algebraic problems. Some of the above steps are implemented using external software (mesh generation, solving systems of algebraic equations), but all these steps based on a unified approach that ensures the quality of the final solution are performed.

The DVM system also solves the problem of increasing the reliability of parallel programs [7,8]. It allows one to naturally implement various types of parallelism and to easily create portable programs because the system-dependent features of the program are automatically generated. The system user interface is based on a markup language similar to OpenMP but much more adapted to solving scientific problems. The system supports all major communication libraries, allows one to work on both homogeneous and heterogeneous computing systems, including graphics accelerators. The ability to dynamically balance the load has also been implemented. Let us compare the approach implemented in the DVM system with the ontological approach. The DVM is beyond the limits of a purely software implementation, also affecting the level of mathematical models, but it does not include the issues of constructing mathematical models in any way. Moreover, due to the interface
implementing in a certain way, the DVM makes it easy to implement models only of a certain type, namely, grid models with a deterministic and constant computation volume.

The above short list of tools is aimed at simplifying the development of applications for solving large computational problems and increasing the reliability of parallel programs is an additional confirmation of the relevance of the developed ontological approach to creating programs for mathematical modeling on a supercomputer.

3. The architecture of the intelligent support system for solving compute-intensive problems of mathematical physics

Let us consider the main conceptual blocks of the developed intelligent support system for solving compute-intensive problems of mathematical physics (the system for SCIPMP) in relation to a researcher using this system (figure 1).

Firstly, the support system for the SCIPMP provides the researcher with the access to an information-analytical block, which includes a knowledge base presented in the form of related ontologies (this is an ontology of numerical methods and parallel algorithms, and an ontology of parallel architectures and technologies) supplemented by inference rules. The knowledge base includes information (with links to details) about the mathematical description of the presented branch of mathematical physics (mathematical models, systems of equations, etc.), information about the necessary numerical solution methods and their parallel implementation on a computing system with links to the current parallel computing architectures and technologies as well as links to available program codes and their descriptions. A detailed description of the construction of ontologies of the SCIPMP with examples for astrophysics, plasma physics, and geophysics is given in [9-12].

For a researcher, the access to this block is provided through an information-analytical Internet resource that allows viewing all the information provided in the form of a drop-down ontology tree. With the help of the resource, one can trace all connections between the presented objects, form a clear view of their main properties, and follow the link with a detailed description of an object, if necessary. That allows the user to receive a systematic understanding of the knowledge domain.

Secondly, a researcher can turn to the expert system, which proposes one or several schemes for its solution according to the specification of his problem. The access to this system is organized through a web interface, which allows one to provide all the information about the user’s problem necessary for the system. Based on this information, the rule engine constructs the corresponding solution schemes, using ontological objects and inference rules that connect them. More details about constructing solution schemes are in the next sections.

From the solution scheme of the problem, the researcher can directly proceed to the development of parallel program implementation. To do this, one can use an interactive code generator, which would provide the access to selecting available ready-made software modules according to the solution scheme. In the absence of necessary modules, the researcher will have to develop them himself, guided by the problem-solution scheme. So far, the code generation module in the developed support system for the SCIPMP is presented only conceptually without any practical implementation. A conceptual description of a code generator structure for solving problems of the space plasma hydrodynamics is presented in [13].

One way or another, the researcher will receive a parallel code to solve his problem for a selected (recommended) computing system. At this stage, one can turn to a module of the simulation modeling. This module simulates the code execution on a large cores number based on the description of the parallel processes interaction during the code execution and real delays collected during the test runs on a small cores number. That makes it possible before the start of large-scale calculations to assess the scalability of the studied code and the actual amount of computing system resources required for a well-timed solution. Although the simulation modeling requires significantly fewer resources than solving actual problems of mathematical physics, for it the resources of a computing cluster (supercomputer), to which the support system for SCIPMP must have access, are needed.
4. Constructing a scheme for solving problems

Let us describe in more detail the conceptual representation of the problem-solution scheme (PSS). To do this, we will consider the scheme constructing process for solving a compute-intensive problem of mathematical physics, relying on the standard approach of researchers involved in this field.

First, it is important to identify the key specifications of a problem that the researcher wants to solve. Therefore, in the general case, the user of an expert system needs to indicate which Physics of Processes or Objects, the Dimension of the problem and the Solution Accuracy he is interested in, the Stationary or Time-dependent problems he will solve, and what Geometry of the computational domain he is considering. We distinguish these parameters as the main ones for the construction of the PSS. But these can be extended with additional parameters that depend on a specific branch of mathematical physics. The interface for entering the specifications must offer the user a choice from the presented options linked to the ontology.

Next, let us describe the main blocks that compound the PSS and the groups of rules necessary for their determination by the inference engine (figure 2).

The modeling Physics, the Space Dimension of the problem, and its Time-dependent properties determine the Equations System, which needs to be numerically solved. In the ontology, the class of Equations Systems is associated with the Test Problems class, on which the user can verify his potential solutions.
numerical solution subsequently. These are the first two blocks of the PSS. To infer them, we need to set a group of rules for determining the Equations System.

The chosen Equations System as well as the Dimension, the desired Solution Accuracy, and the Geometry of the computational domain influence the choice of the Numerical Method. That is the next block of the PSS. Hence, it is necessary to set a group of rules responsible for choosing a numerical method from those presented in the ontology. These rules must also be taken into account in the optimal choice from the point of view of further parallelization.

The considered Equations System and Numerical Method, as well as the specified Geometry, determine the Principles of Parallel Implementation of the numerical method, which must be guided by the development of an efficient parallel code. This block provides information on a suitable decomposition selection of the computational domain, a recommendable data structure, a relevant distribution of computations among processes, etc. Based on this knowledge, one can get an idea of the algorithm parallelization for solving the problem.

Further, using a group of rules for obtaining the qualitative and quantitative Parallel Algorithm Properties, the next block of the PSS is inferred. It can include estimates of the number and volume of transactions, arithmetic operations number, data re-use in a parallel algorithm, etc. This information allows the researcher to evaluate the performance of considered approaches to the numerical solution from different angles if the expert system offers several configurations of the PSS. Also, this information helps to determine the choice of the Target Architecture and the Parallel Programming Technologies associated with it in the ontology. For this, it is also necessary to formulate an appropriate group of rules. The Target Architecture is some class of computing architectures that are most suitable for solving the problem using the selected methods. It can be supplemented with specific computing systems examples available in the computing centers described in the ontology.

Note that in addition to the described groups of rules, there may be some other rules. They restrict the choice of any subsequent blocks at one or another stage of constructing the PSS. Also, if the expert
system cannot make a deterministic choice among several configurations of the PSS, then the user is provided with a description of possible ways. Additionally, the user can intervene in the Equations System choice or a Numerical Method, guided by his own considerations. In any case, the system will provide him with recommendations for the parallel implementation of the selected solution method.

Thus, the PSS suggests a wide range of recommendations for the numerical solution of the user’s problems on parallel computing systems with allowance for their features.

5. The inference rules
The inference rules are used to check the consistency of the knowledge bases and to infer information that is not explicitly presented in intelligent systems. Such rules are developed in close collaboration between the subject matter experts and the knowledge representation specialists. These rules are formalized using the SWRL language [14]. The rule execution is provided by one of the standard inference engines (Pellet, Hermit, FaCT++). The considered IS SCIPMP contains two types of rules: general rules that serve to populate the knowledge base of the system, and the rules intended for constructing a scheme for solving a specific problem according to its specification provided by the user. Let us take a closer look at these rules.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

5. The inference rules

The inference rules are used to check the consistency of the knowledge bases and to infer information that is not explicitly presented in intelligent systems. Such rules are developed in close collaboration between the subject matter experts and the knowledge representation specialists. These rules are formalized using the SWRL language [14]. The rule execution is provided by one of the standard inference engines (Pellet, Hermit, FaCT++). The considered IS SCIPMP contains two types of rules: general rules that serve to populate the knowledge base of the system, and the rules intended for constructing a scheme for solving a specific problem according to its specification provided by the user. Let us take a closer look at these rules.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

5. The inference rules

The inference rules are used to check the consistency of the knowledge bases and to infer information that is not explicitly presented in intelligent systems. Such rules are developed in close collaboration between the subject matter experts and the knowledge representation specialists. These rules are formalized using the SWRL language [14]. The rule execution is provided by one of the standard inference engines (Pellet, Hermit, FaCT++). The considered IS SCIPMP contains two types of rules: general rules that serve to populate the knowledge base of the system, and the rules intended for constructing a scheme for solving a specific problem according to its specification provided by the user. Let us take a closer look at these rules.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

5. The inference rules

The inference rules are used to check the consistency of the knowledge bases and to infer information that is not explicitly presented in intelligent systems. Such rules are developed in close collaboration between the subject matter experts and the knowledge representation specialists. These rules are formalized using the SWRL language [14]. The rule execution is provided by one of the standard inference engines (Pellet, Hermit, FaCT++). The considered IS SCIPMP contains two types of rules: general rules that serve to populate the knowledge base of the system, and the rules intended for constructing a scheme for solving a specific problem according to its specification provided by the user. Let us take a closer look at these rules.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.

General rules make it possible to represent in the system the knowledge of experts about the subject area as a whole. For example, rule (1) says that a kinetic model is required for a low-temperature plasma with a low density. Rules (2) and (3) determine the most suitable model depending on the distribution function. If the form of the distribution function is unknown, then it is necessary to solve the Vlasov equation. If the distribution function is in equilibrium, then the equations of magneto-hydrodynamics can be used.
'Problem solving Scheme'(?, 'has Scheme'(?, ?), 'suitable for'('Vlasov equation', ?)) -> 'includes Architecture Element'(?, GPU, 'includes Technology'(?, 'NVIDIA CUDA')

Figure 3 shows a fragment of the ontology and inference rules presented in the Protégé editor. The expert rules are shown in the left area of the figure. The right area shows the result of their work - an object of the class “Scheme for solving the problem of modeling magnetized plasma” and some elements of this scheme.

Figure 3. The inference rules and the result of their work in the Protégé editor

6. Conclusion
The paper discusses an intelligent system designed to support researchers in the development of a parallel code for physical process modeling. The system is based on the developed ontology of the knowledge area "Support for solving compute-intensive problems of mathematical physics on supercomputers". The main result of the system work is a scheme for solving a specific problem, built according to its specification provided by the user. The scheme includes a description of all components required for the solution from mathematical models to the most suitable parallel architectures as well as links to available fragments of a parallel code that the user can re-use. The scheme construction based on ontology and expert rules built using the Semantic Web technology is carried out.

The distinctive features of the system under consideration, in comparison with the systems (software packages) that are similar in the formulation of the problem, the methods used, and the capabilities provided, are the following:

- Access to well-structured information on solving problems of modeling physical processes.
- Openness of access: the possibility to see what is "under the hood" of the proposed solution and, if necessary, replace one or another component.
- Possibility to consider problems, methods, or architectures not presented in existing packages.
- Possibility to use our system for training.
- Free access and the absence of specific license conditions in comparison with paid software packages.

Acknowledgments
The research has been supported by Russian Foundation for Basic Research (project no. 19-07-00085).
References

[1] Zagorulko G, Zagorulko Y, Glinskiy B and Sapetina A 2019 Ontological approach to providing intelligent support for solving compute-intensive problems on supercomputers Communications in Computer and Information Science 1093 363–75

[2] Malysshkin V 2015 Active knowledge, LuNA and literacy for oncoming centuries Lecture Notes in Computer Science 9465 292–303

[3] Malysshkin V E and Perepelkin V A 2011 LuNA fragmented programming system, main functions and peculiarities of run-time subsystem. Lecture Notes in Computer Science 6873 53–61

[4] Il’in V P and Skopin I N 2016 About performance and intellectuality of supercomputer modeling Programming and Computer Software 42(1) 5–16

[5] Il’in V P 2016 Fundamental issues of mathematical modeling Herald of the Russian Academy of Sciences 86(2) 118–126

[6] Il’in V P 2018 The conception, requirements and structure of the integrated computational environment Communications in Computer and Information Science 965 653–65

[7] Konovalov N A, Kryukov V A, Mikhajlov S N and Pogrebtsov A A 1995 FORTRAN DVM language for developing portable parallel programs Programmirovanie 21(1) 49–54

[8] Bakhtin V A, Kryukov V A, Chetverushkin B N and Shil’nikov E V 2011 Extension of the DVM parallel programming model for clusters with heterogeneous nodes Doklady Mathematics 84(3) 879–881

[9] Glinskiy B M, Zagorulko Yu A, Zagorulko G B, Kulikov I M, Sapetina A F, Titov P A and Zhernyak G F 2021 Building ontologies for solving compute-intensive problems J. Phys.: Conf. Ser. 1715 012071

[10] Sapetina A, Glinskiy B and Zagorulko G 2020 Content of ontology for solving compute-intensive problems of the cosmic plasma hydrodynamics J. Phys.: Conf. Ser. 1640 012019

[11] Snytnikov A V, Glinskiy B M, Zagorulko G B and Zagorulko Y A 2020 Ontological approach to formalization of knowledge in computational plasma physics J. Phys.: Conf. Ser. 1640 012013

[12] Glinskiy B M, Zhernyak G F and Titov P A 2020 System of intelligent support for solving geophysical problems Russian Supercomputing Days: Proc. of the Int. Conf., ed Vl Voevodin (Moscow: MAKS Press) pp 11–18

[13] Sapetina A, Ulyanichev I and Glinskiy B 2019 The grid codes generation for solving problems of the cosmic plasma hydrodynamics on supercomputers J. Phys.: Conf. Ser. 1336 012012

[14] SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission 21 May 2004. http://www.w3.org/Submission/SWRL/