Bull. Nov. Comp. Center, Comp. Science, 44 (2020), 23–29
© 2020 NCC Publisher

The ontology design for solving computational plasma physics problems on supercomputers*

A.V. Snytnikov

Abstract. Computational plasma physics problems are a wide field with its own hardware, software and scheduling strategies. There is a set of physical phenonena, mathematical equations, numerical approaches, programming strategies and architecture concepts directly followed by each other. This means that an efficient code for the plasma physics must involve all the subjects, which require a clear representation of the relationships amongst physical, mathematical and computer science concepts — which is, the ontology. In this paper it is shown how physics determines equations being used, how the equations define numerical methods, and how methods enable programming strategies to form an architecture-efficient implementation.

Introduction

Ontology, as defined in Wikipedia [1], is somewhat encompassing a representation, formal naming and definition of the categories, properties and relations amongst the concepts, data and entities that substantiate a domain of discourse, which is, the computational plasma physics. The ontology is a way of showing the properties of a subject area and how they are related by defining a set of concepts and categories that represent the subject.

The necessity of developing an ontology for the computational plasma physics is to give a formal, standard way of choosing mathematical and numerical models for a given physical problem, and, as a more recent challenge, to select an appropriate target architecture of a supercomputer.

The correct choice of architecture is really the key question in recent times. It is important, first, to avoid the waste of computing time and of energy resources, and second, to get physically meaningful results, which might be really impossible with a wrong architecture.

1. Components of the subject area

1.1. Individuals. Instances or objects (the basic or "ground level" objects). The basic or the ground level object in the computational plasma physics [2] ontology is a sort of plasma [3]. In Figure 1, they are listed at level 1 in the top of picture. Plasma is present in space (the solar crown,

*Supported by RFBR under Grant 19-07-00085.

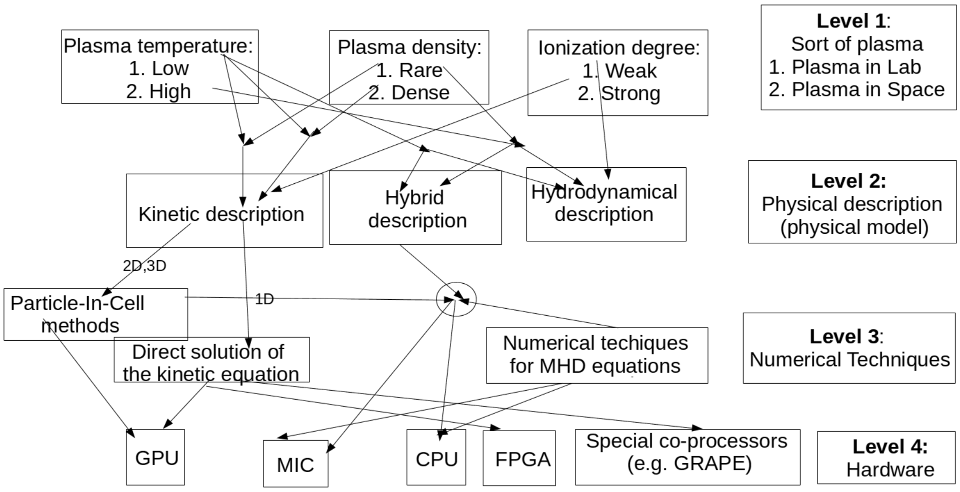

Figure 1. Computational plasma physics ontology

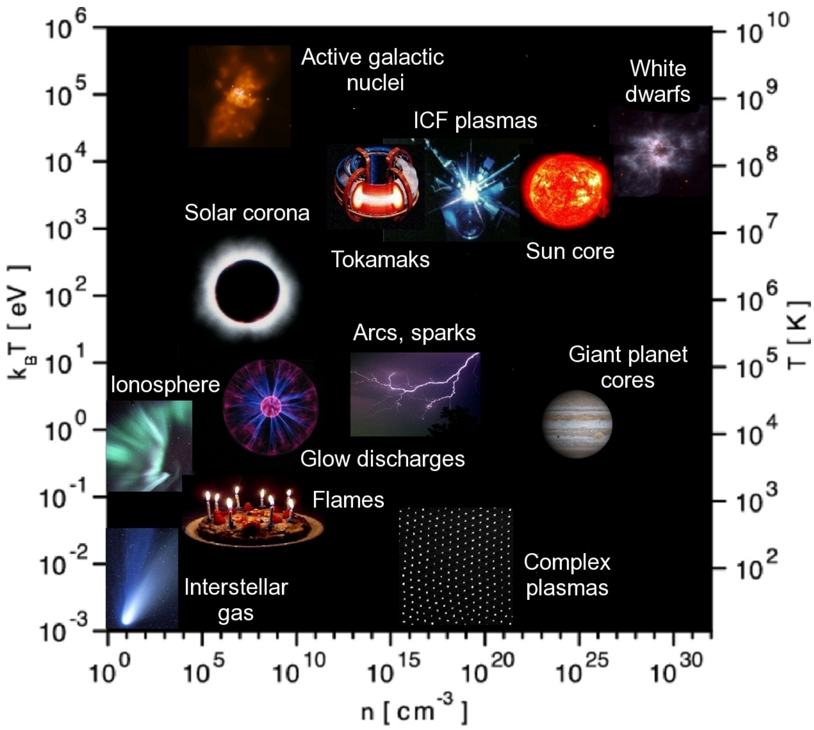

Figure 2. Sorts of plasma with respect to density (n, in particles per cubic centimeter, cm^{-3}) and temperature (T), in degrees (Kelvin). The picture is taken from the page of Wigner Research Centre for Physics

ionized gas in the interstellar medium, etc.) and in laboratory conditions (fusion devices and plasma reactors used in microelectronics). Plasma might be cold (e.g. 1000 K) or hot (over 1 million degrees). Plasma also differs in density from 10^{15} cm^{-3} in fusion devices like tokamak to 1 particle per cubic centimeter (or even less) in interstellar gas. This diversity is shown in Figure 2.

Since the physical properties of plasma under study determine everything else, all the arrows in the figure show downward, and never upward. Neither equations, nor numerical techniques, nor hardware affect plasma. If they do, then the simulation starts being physically incorrect.

1.2. Classes are plasma models on physical level, e.g., collision or non-collision plasma, kinetic or hydrodynamic plasma modeling and also on sets of equations describing plasma in various cases.

Plasma is generally described by the Vlasov equation [4], which is the phase volume conservation law:

$$\begin{gathered}\frac{\partial f_{i,e}}{\partial t} + \vec{v}\frac{\partial f_{i,e}}{\partial \vec{r}} + \vec{F}_{i,e}\frac{\partial f_{i,e}}{\partial \vec{p}} = 0,\\ \vec{F}_{i,e} = q_{i,e}\left(\vec{E} + \tfrac{1}{c}[\vec{v},\vec{B}]\right).\end{gathered} \tag{1}$$

Here f is the distribution function (the number of real particles at some definite point with some definite velocity), or the phase density. The indices i and e refer to the ion and electron components of plasma, $\vec{r}$ is the coordinate and $\vec{v}$ is the velocity, $\vec{p}$ gives momentum, $\vec{F}$ is the force acting on each particle, and depending on the electric field $\vec{E}$ and the magnetic field $\vec{B}$. Note that the right-hand-side is 0, which means no collisions (collisions are neglected).

The Vlasov equation is the most common way which is correct in all cases. But since the numerical solution of the Vlasov equation in real 2D or 3D cases is extremely time-consuming, a simpler way is required. It is the hydrodynamic description: if the local distribution function is close to Maxwellian, then the Vlasov equation could be integrated by the velocities resulting in the hydrodynamic set of equations. Since it also involves the magnetic field equations, it is called magnetohydrodynamic (MHD) equations.

1.3. Attributes of the classes appear on mathematical level at level 3 in Figure 1. The attributes mean the details of the physical plasma models. For example, kinetic modeling might be performed with Particle-In-Cell method [5,6] or by means of a direct solution [7] of the Vlasov equation, or using Monte-Carlo methods [8]. The attributes might overlap, for example, the solution method might implement both Particle-In-Cell, or Monte Carlo techniques [9,10].

1.4. Relations. Basically, in the computational plasma physics there is just one relation: "is determined" or "is followed". This happens when an individual, a sort of plasma, or its two sorts determines a description to be used. For example, the low temperature and rare density plasma strongly requires a kinetic description. Or, on the contrary, the strongly ionized plasma allows a hydrodynamic description to be used. This type of relation, "is determined", is shown by an arrow in Figure 1. When something is simultaneously determined by two individuals we use joining arrows.

In the same way, a numerical technique is determined by a physical model, which is, a kinetic or a hydrodynamic description.

1.5. Function terms. More complex relations, involving two or more classes, appear at level 3, the numerical techniques level, when a physical model, from level 2, together with one of the numerical techniques from level 3, could be implemented for more than one of the hardware architectures. In Figure 1, it is shown by the circle with joining arrows inside.

1.6. Restrictions. There is only one restriction in the computational plasma physics ontology: no arrow (meaning determination relation) can go over a level. This means that an arrow is able to link objects only at the adjacent levels, and the arrows show only down, never up. In the real simulation of plasmas this means that the physical nature of plasma does not directly affect numerical techniques, it affects them only through physical models. Moreover, the physical nature of plasma does not directly determine architectures being used, only through two intermediate levels.

1.7. Rules. The present subsection is the most important in the ontology. Here we state that a numerical technique strictly determines the choice of an architecture.

Here, the MHD equations solution techniques have to be implemented for classic CPUs (central processor units or, simply, processors). It is because these techniques require rather a big amount of RAM together with large cache memory and random access. It is sometimes possible to implement these techniques also with MIC (Many Integrated Cores, a powerful co-processor from Intel), though as Kulikov [11] showed, it demands a great amount of low-level programming.

As Figure 1 shows, the direct solution of the kinetic equation (an explicit numerical scheme) could be efficiently implemented by all the abovementioned hardware, including FPGA (Field-programmable gate array, [12]). But it must be remembered that the direct solution of the Vlasov equation [13] requires a very big amount of memory. In fact, it needs one more level in the ontology to consider the speed of interprocessor communications.

The Particle-In-Cell method, another way of the kinetic treatment of plasma problems, is shown to be efficiently implemented for GPUs (graphical processors units), [14–18].

2. Graphical representation of ontology

Following the idea given in [19], let us give the conceptual scheme (Figure 3) that shows the place of the above-discussed ontology in the development of the HPC plasma simulation code.

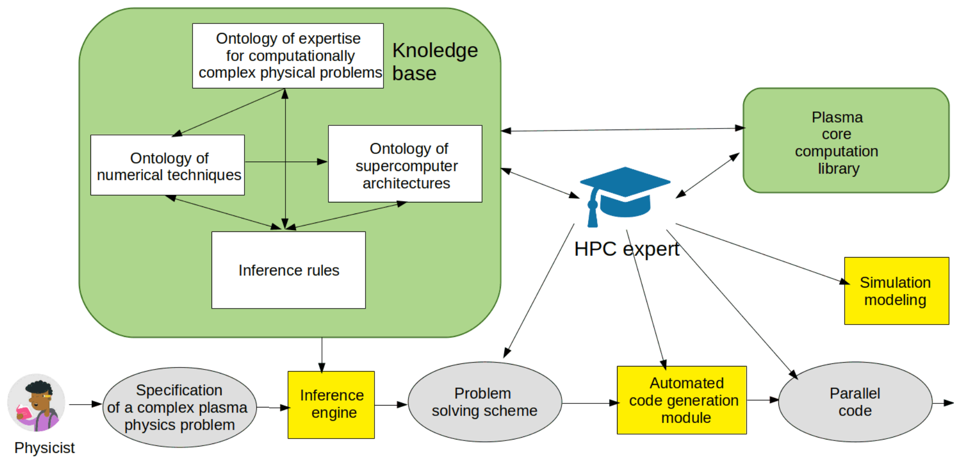

Figure 3. A conceptual scheme of the Intelligence Support system for solving compute-intensive plasma physics problems

Conclusion

The ontology description for the computational plasma physics is given. The ontology includes individuals, classes, attributes, relations, restrictions and rules. The feature of the computational plasma physics ontology is that all the relations go from a higher level down (from physics to mathematics and from mathematics to programming), and relations exist only between the adjacent levels. The practical use of the above text is the restriction that a hardware architecture must be chosen according to the numerical technique used. The future study is to extend the ontology in order to take the peculiarities of the numerical techniques into account and to facilitate the connection between the numerical techniques and hardware used.

References

[1] Ontology (information science). — https://en.wikipedia.org/wiki/Ontology_(information_science).

[2] Birdsall C.K., Langdon A.B. Plasma Physics via Computer Simulation. — Bristol: Institute of Physics Pub., 2005.

[3] Nicholas A. Krall, Alvin W. Trivelpiece. Principles of Plasma Physics. — 1973.

[4] Vlasov A.A. Theory of Vibrational Properties of an Electron Gas and Its Applications. — 1945.

[5] R.W Hockney R.W., Eastwood J.W. Computer Simulation Using Particles. — CRC Press, 1988.

[6] Grigoryev Yu.N., Vshivkov V.A., Fedoruk M.P. Numerical "Particle-in-Cell" Methods. — VSP, Utrecht–Boston, 2002.

[7] Fijalkow E. A numerical solution to the Vlasov equation // Computer Physics Communications, 1999. — Vol. 116, Iss. 2–3. — P. 319–328.

[8] Hammersley J. M., Handscomb D.C. Monte Carlo Methods. — London: Methuen, 1975. ISBN 978-0-416-52340-9.

[9] Birdsall .K. Particle-in-cell charged-particle simulations plus Monte Carlo collisions with neutral atoms, PIC-MCC. // IEEE Trans.Plasma Sci. — 1991. — Vol. 19, No. 2. — P. 65–83.

[10] Vahedi V., Surendra M. A Monte Carlo collision model for the particle-in-cell method: applications to argon and oxygen discharges // Computer Physics Communications. — 1995. — Vol. 87. — P. 179–198.

[11] Kulikov I.M., Chernykh I.G., Glinsky B.M. AstroPhi: a hydrodynamical code for complex modelling of astrophysical objects dynamics by means of hybrid architecture supercomputers on Intel Xeon Phi base // Vestn. YuUrGU. Ser. Vych. Matem. Inform. — 2013. — 2:4. — P. 57–79 (In Russian).

[12] Wiśniewski R. Synthesis of Compositional microprogram control units for programmable devices. — Zielona Gora: University of Zielona Gora. ISBN 978-83-7481-293-1.

[13] Cohen B.I., Barnes D.C., Dawson J.M., et al. The numerical tokamak project: simulation of turbulent transport // Computer Physics Communications. — 1995. — Vol. 87, Iss. 1–2. — P. 1–15.

[14] Burau H., Widera R., Honig W., et al. PIConGPU: a fully relativistic particle-in-cell code for a GPU cluster // IEEE Transactions on Plasma Science — 2010. — Vol. 38, Iss. 10. — P. 2831–2839. DOI: 10.1109/TPS.2010.2064310.

[15] Rossi F., Londrillo P., Sgattoni A., et al. Towards robust algorithms for current deposition and dynamic load-balancing in a GPU particle in cell code // AIP Conference Proceedings. — 2012. — Vol. 1507, Iss. 1. — P. 184–192. DOI: 10.1063/1.4773692.

[16] Kong X., Huang M., Ren Ch., Decyk V. Particle-in-cell simulations with charge-conserving current deposition on graphic processing units // J. Comput. Phys. — 2011. — Vol. 230, Iss. 4. — P. 1676–1685. DOI: 10.1016/j.jcp.2010.11.032.

[17] Rieke M., Trost T., Grauer R. Coupled Vlasov and two-fluid codes on GPUs // J. Comput. Phys. — 2015. — Vol. 283. — P. 436–452. DOI: 10.1016/j.jcp.2014.12.016.

[18] Lotov K.V., Timofeev I.V., Mesyats E.A., et al. Note on quantitatively correct simulations of the kinetic beam-plasma instability // Physics of Plasmas. — 2015. — Vol. 22, Iss. 2. — P. 024502. DOI: 10.1063/1.4907223.

[19] Glinskiy B., et al. J. Phys.: Conf. Ser. 1392 012052. — 2019.