Application of the InKS programming model to the Dynamico code

Supervisor : Thomas Dubos (LMD) dubos@lmd.polytechnique.fr, Tel : +33 (0)1 69 33 51 43
Co-supervisor : Julien Bigot (MdlS) julien.bigot@cea.fr, Tel : +33 (0)1 69 08 01 75
Localization : Maison de la Simulation (MdlS)

MdlS Maison de la Simulation, bât. 565, CEA Saclay, 91191 Gif-sur-Yvette Cedex
LMD Laboratoire de Météorologie Dynamique, Département de Mécanique, École Polytechnique, 91128 Palaiseau

Abstract

InKS is a new programming model being designed at the "Maison de la Simulation". Dynamico is a code intended to model earth and planetary atmospheres by solving the equations of compressible fluid motion subject to the gravity and Coriolis forces. The Ph.D. proposed here aims to to apply the InKS model to express parts of the Dynamico code critical in term of parallel performance. In this context, you will most likely have to improve InKS so as to take into account the constraints that appear when using it in a real code. This new approach should enable to solve limitations of the Dynamico code by improving its readability, by optimizing it for various target architectures and by easing the choice between various versions of the physical equations implemented.

Context

The **Maison de la Simulation** (MdlS) is a joint laboratory between CEA, Inria, CNRS, the Universities of Versailles Saint-Quentin and Paris-Sud located on the Saclay Plateau near Paris. The lab hosts research activities in high performance computing (*HPC*) science, applied mathematics and engineering connected to scientific simulation codes for application domains. Especially, research related to programming models is developed at MdlS.

Laboratoire de Météorologie Dynamique(LMD) is a joint laboratory between CNRS, École Polytechnique, École Normale Supérieure and Université Pierre et Marie Curie. Il hosts research activity on climat and atmospheric dynamics, leaning on theory, observation and numerical modelling. In collaboration with the Laboratoire des Sciences du Climat et de l'Environnement (LSCE) within Institut Pierre Simon Laplace (IPSL), it especially develops research work on high-performance modelling of atmospheric dynamics.

InKS (*INdependant Kernel Scheduling*) is a programming model being designed at MdS with the aim of decoupling the functional specification of HPC codes from their optimization. InKS can be regarded as a domain-specific language (DSL) targeting purely computational scientific simulation applications, whose result is entirely determined by a set of input values. The computations to be performed can the be described in terms of a task graph within which data flows as in a Makefile, typically by applying the same computation on many elements of an array. A InKS program consists therefore in a set of data carried by meshes and in computational kernels computing one or several values of these arrays according to specified dependencies. Automatic execution of such a program is possible, but in order to extract reasonable performance it is necessary to specify additionnally : 1) the order in which to execute the computations and possibly their distribution among computing cores as well as 2) the memory allocation of data as a function of time and possibly their distribution among computing nodes.

Dynamico[16] is a global dynamical core aimed at modelling terrestrial and planetary atmospheres. Dynamico solves the equations of motion of a compressible fluid subject to gravity and Coriolis forces. For realistic applications it is complemented by a set of so-called physical parameterizations describing the statistical effect of small-scale processes such as convection and matter-radiation interaction. The degree of parallelism of the full model is essentially limited by that of the dynamical core, physical parameterizations being embarassingly parallel since they do not couple neighboring atmospheric columns. Dynamico is formulated on a quasi-uniform spherical mesh and reaches a high degree of MPI/OpenMP parallelism. It is written in FORTRAN 90 following a computational structure and a memory layout designed for simplicity and efficiency on scalar architectures. Physical and computational aspects being tightly intricated in the code, exploiting efficiently emerging computational architectures (GPU, Intel Xeon Phi) while continuing to add new physical functionalities is a challenge.

Goals

During the PhD, you will apply the InKS programming model to express certain performance-critical parts of Dynamico. You will possibly introduce evolutions of InKS in order to take into account the constraints of a production code. This new approach should allow to overcome some limitations of Dynamico by improving its readability, optimizing it for various target machines and by facilitating the switch between several versions of the physical equations to be solved.

In the longer run you can aim at covering all the performance-critical parts of Dynamico. You should then encounter elements of the code that fit less well in the InKS framework. You will then determine how to take into account these aspects. A direction could be to use the PDI[10] library which provides a way to interface with third-party libraries.

Another current limitation of InKS that you could relax is that this model is limited to very regular codes, in terms of mesh as well as time iteration. Supporting unstructured meshes within InKS would provide a particularly useful extension of Dynamico to such meshes.

If time allows, it will be possible to continue the work in various directions, such as :

— formalisation and implementation of a language dedicated to the programming model,
— exploration of a wide array of tools for optimisation,
— analysis of the validity of the manually-specified scheduling.

Work related to InKS

InKS is a new programming model, but many of its underlying ideas are already present in existing models.

The idea to describe operations and their dependencies in a dedicated language in order to schedule them is at the heart of the concept of Makefile [2], a similar idea is found in the IRP programming method implemented in IRPF90 [23]. In a HPC framework, various models allow to define tasks within a code and automatically

schedule them, as with OpenMP 4 [22], Cilk++ [20], StarPU [8], XKAAPI [18], PaRSEC [12] or Legion [9] for instance.

The approcah consisting in decoupling the description of computing kernels from the iteration space to which they should be applied is found in many languages and libraries among which Cuda [4], Nabla [13], Kokkos [17] or Raja [1] for instance. This approach is particularly present in the category of codes called 'Stencil' "Stencil" [6] for which some languages propose to optimize a specific kernel, e.g. PATUS [14], Pochoir [24], OP2 [19] or Liszt [15] ; more recent work aims at optimizing a full application, e.g. MSL [11]. Finally the category of languages with partitioned global address space [3] (*PGAS*) allow an distributed across several nodes to be considered as a single array ; it is the case among others of Co-array Fortran [21] and *unified parallel C (UPC)* [7].

Describing computations as acting on logical data possessing a well-defined value rather than variables representing memory space whose content and semantics may vary as the computation proceeds is to be related to the notion of static single assignment [5] (*SSA*) or dynamic single assignment [25] (*DSA*).

Finally, one may mention BOAST [26], a tool allowing to vary the optimizations applied to a computing kernel according to various parameters, especially the target machine.

Liens et références / Links and references

Maison de la Simulation http://www.maisondelasimulation.fr/
Laboratoire de Météorologie Dynamique http://www.lmd.jussieu.fr/
Dynamico http://forge.ipsl.jussieu.fr/dynamico/

[1] Cuda. https://en.wikipedia.org/wiki/CUDA. Accessed : 2016-02-01.

[2] Makefile. https://en.wikipedia.org/wiki/Makefile. Accessed : 2016-02-01.

[3] Partitioned global address space. https://en.wikipedia.org/wiki/Partitioned_global_address_space. Accessed : 2016-02-01.

[4] Raja : Managing application portability for next-generation platforms. http://computation.llnl.gov/projects/raja-managing-application-portability-next-generation-platforms. Accessed : 2016-02-01.

[5] Static single assignment form. https://en.wikipedia.org/wiki/Static_single_assignment_form. Accessed : 2016-02-01.

[6] Stencil code. https://en.wikipedia.org/wiki/Stencil_code. Accessed : 2016-02-01.

[7] Unified parallel c. https://upc-lang.org/. Accessed : 2016-02-01.

[8] Cédric Augonnet, Samuel Thibault, Raymond Namyst, and Pierre-André Wacrenier. StarPU : A unified platform for task scheduling on heterogeneous multicore architectures. *Concurrency and Computation : Practice and Experience, Special Issue : Euro-Par 2009*, 2011.

[9] Michael Bauer, Sean Treichler, Elliott Slaughter, and Alex Aiken. Legion : Expressing locality and independence with logical regions. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 66 :1–66 :11, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[10] Julien Bigot. PDI, the parallel data interface. https://gitlab.maisondelasimulation.fr/jbigot/pdi. Accessed : 2016-02-01.

[11] Julien Bigot, Hélène Coullon, and Christian Pérez. From DSL to HPC Component-Based Runtime : A Multi-Stencil DSL Case Study. In *WOLFHPC 2015 (Fifth International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing)*, page 10, co-located with SC'15, Austin, Texas, United States, November 2015.

[12] George Bosilca, Aurelien Bouteiller, Anthony Danalis, Thomas Hérault, Pierre Lemarinier, and Jack Dongarra. DAGuE : A generic distributed dag engine for high performance computing. *Parallel Computing*, 2012.

[13] Jean-Sylvain Camier. Improving performance portability and exascale software productivity with the &nabla ; numerical programming language. In *Proceedings of the 3rd International Conference on Exascale Applications and Software*, EASC '15, pages 126–131, Edinburgh, Scotland, UK, 2015. University of Edinburgh.

[14] M. Christen, O. Schenk, and H. Burkhart. PATUS : A Code Generation and Autotuning Framework for Parallel Iterative Stencil Computations on Modern Microarchitectures. In *Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 676–687. IEEE, May 2011.

[15] Z. DeVito, N. Joubert, F. Palacios, S. Oakley, M. Medina, M. Barrientos, E. Elsen, F. Ham, A. Aiken, K. Duraisamy, E. Darve, J. Alonso, and P. Hanrahan. SC '11, pages 9 :1–9 :12, New York, NY, USA, 2011. ACM.

[16] T. Dubos, S. Dubey, M. Tort, R. Mittal, Y. Meurdesoif, and F. Hourdin. Dynamico-1.0, an icosahedral hydrostatic dynamical core designed for consistency and versatility. *Geoscientific Model Development*, 8(10) :3131–3150, 2015.

[17] H. Carter Edwards and Daniel Sunderland. Kokkos array performance-portable manycore programming model. In *Proceedings of the 2012 International Workshop on Programming Models and Applications for Multicores and Manycores*, PMAM '12, pages 1–10, New York, NY, USA, 2012. ACM.

[18] Thierry Gautier, Joao Vicente Ferreira Lima, Nicolas Maillard, and Bruno Raffin. Locality-aware work stealing on multi-cpu and multi-gpu architectures. In *Workshop on Programmability Issues for Heterogeneous Multicores (MULTIPROG)*, 2013.

[19] M. B. Giles, G. R. Mudalige, Z. Sharif, G. Markall, and P. H.J. Kelly. Performance Analysis of the OP2 Framework on Many-core Architectures. *SIGMETRICS Perform. Eval. Rev.*, 38(4) :9–15, March 2011.

[20] Charles E. Leiserson. The cilk++ concurrency platform. In *Proceedings of the 46th Annual Design Automation Conference*, DAC '09, pages 522–527, New York, NY, USA, 2009. ACM.

[21] Robert W. Numrich and John Reid. Co-array fortran for parallel programming. *SIGPLAN Fortran Forum*, 17(2) :1–31, August 1998.

[22] OpenMP Architecture Review Board. OpenMP application programming interface version 4.5, November 2015.

[23] Anthony Scemama. IRPF90 : a programming environment for high performance computing. *CoRR*, abs/0909.5012, 2009.

[24] Y. Tang, R.A. Chowdhury, B.C. Kuszmaul, C-K Luk, and C.E. Leiserson. The pochoir stencil compiler. In Lance Fortnow and Salil P. Vadhan, editors, *SPAA*, pages 117–128. ACM, 2011.

[25] Peter Vanbroekhoven. Dynamic single assignment. ESSES PhD Student Meeting at the 9th International Static Analysis Symposium, Madrid, Spain, September 15-21, 2002, 2002.

[26] Brice Videau, Vania Marangozova-Martin, and Johan Cronsioe. BOAST : bringing optimization through automatic source-to-source transformations. In *International symposium on embedded multicore/manycore system-on-chip (MCSoC)*, 2013.