Computer Physics Communications 220 (2017) 188–204

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

Atlas: A library for numerical weather prediction and climate modelling

Willem Deconinck [*], Peter Bauer, Michail Diamantakis, Mats Hamrud,
Christian Kühnlein, Pedro Maciel, Gianmarco Mengaldo, Tiago Quintino,
Baudouin Raoult, Piotr K. Smolarkiewicz, Nils P. Wedi

European Centre for Medium-Range Weather Forecasts (ECMWF), Shinfield Park, Reading RG2 9AX, United Kingdom

ARTICLE INFO

Article history:
Received 10 January 2017
Received in revised form 31 May 2017
Accepted 5 July 2017
Available online 22 July 2017

Keywords:
Numerical weather prediction
Climate
Earth system
High performance computing
Meteorology
Flexible mesh data structure

ABSTRACT

The algorithms underlying numerical weather prediction (NWP) and climate models that have been developed in the past few decades face an increasing challenge caused by the paradigm shift imposed by hardware vendors towards more energy-efficient devices. In order to provide a sustainable path to exascale High Performance Computing (HPC), applications become increasingly restricted by energy consumption. As a result, the emerging diverse and complex hardware solutions have a large impact on the programming models traditionally used in NWP software, triggering a rethink of design choices for future massively parallel software frameworks. In this paper, we present *Atlas*, a new software library that is currently being developed at the European Centre for Medium-Range Weather Forecasts (ECMWF), with the scope of handling data structures required for NWP applications in a flexible and massively parallel way. *Atlas* provides a versatile framework for the future development of efficient NWP and climate applications on emerging HPC architectures. The applications range from full Earth system models, to specific tools required for post-processing weather forecast products. The *Atlas* library thus constitutes a step towards affordable exascale high-performance simulations by providing the necessary abstractions that facilitate the application in heterogeneous HPC environments by promoting the co-design of NWP algorithms with the underlying hardware.

1. Introduction

In the early days of HPC, computational power was gained through taking advantage of vector instructions on large single-processor computers. With little modification to existing code bases, shared memory parallelisation was introduced when more and more processors were added to one computer, bringing forth multi-core units. However, when multi-node/multi-core architectures (many computers linked together) became widespread in the 1990s, the required effort to port existing codes to make use of distributed memory was significant, and often meant rewriting or redesigning large parts of the codes. Over a decade later, many NWP codes, such as the ECMWF Integrated Forecasting System (IFS), have grown in size to millions of lines, making use of hybrid parallelisation with distributed and shared memory on multi-node/multi-core architectures [1]. Here performance was gained through increasing the CPU flop-rates and the number of nodes of the HPC system.

In today's HPC landscape, it has become unfeasible to boost computational performance by increasing the CPU's clock speed as the Dennard scaling [2] has broken down since around 2006, so the increase in computational performance has to be obtained largely by introducing more parallelism. Enlarging the HPC systems with more nodes of current CPU technology will ultimately result in unaffordable energy costs for many NWP operational centres. For these reasons, and with the goal to develop solutions suitable for the next generation exascale systems, hardware vendors introduced many-core processors (also known as accelerators), such as Graphic Processing Units (GPUs) and Intel's Many Integrated Core (MIC) architectures. These computing technologies have a higher flop-rate and lower power consumption than traditional multi-core CPU processors, and they have less stringent cooling requirements. On the other hand, they suffer from lower single-thread performance than traditional multi-core CPU processors. To exploit their higher degree of parallelism, a substantial effort in redesigning existing codes is required, as with the advent of multi-node/multi-core architectures in the 1990s.

---

[*] Corresponding author.
E-mail address: willem.deconinck@ecmwf.int (W. Deconinck).

http://dx.doi.org/10.1016/j.cpc.2017.07.006

While there is no definitive answer regarding what the optimal solution in terms of performance and energy requirements is, many NWP centres are looking at hybrid strategies, having both traditional multi-core and many-core units within the same HPC system and inter-operate them exploiting at best the advantages of the two [3]. In the next few decades, there might be a complete transition to many-core architectures and some NWP centres have already adopted this solution [4,5]. Interoperating across different hardware is a key factor for a sustainable development of Earth system models and for an efficient software/hardware co-design [6]. In order to achieve these goals, it is essential to have full flexibility on the underlying data structure. This involves not only having flexibility in terms of algorithmic design and memory layout most suited for different hardware, but also enabling the development of different numerical modelling strategies for solving the partial differential equations (PDEs) describing the atmosphere and ocean dynamics.

A good example for the importance of the latter point is the current numerical modelling infrastructure adopted at ECMWF. The IFS at ECMWF solves the system of governing PDEs through a spectral-transform-based approach [7]. This approach requires discrete transformations between physical space (grid-points) and spectral space (spherical-harmonics or Fourier coefficients). In general, spectral-transform-based approaches require data-rich global communications that become the main limiting factor for extreme-scale computations [8]. In addition, the IFS has been developed targeting traditional multi-node/multi-core computing technologies while its portability and efficiency on emerging many-core hardware is still under investigation [9]. Research may show that alternative numerical strategies offer better scalability and efficiency for certain hardware configurations, such as compact-stencil grid-point methods that only require nearest-neighbour communication.

ECMWF is developing a library called *Atlas*, with the primary goals to exploit the emerging hardware architectures becoming available in the next few decades, and to support the development of alternative numerical algorithm strategies in operational NWP. These developments apply not only to the forecast model, but also to the post-processing of model output to generate *products*.[1]

Atlas is also expected to facilitate the coupling of an increasing number of Earth system components, such as the atmosphere, ocean, wave, surface, or sea-ice, and could effectively enhance existing couplers such as OASIS [10]. The challenge of coupling Earth system components has been addressed in the Earth System Modelling Framework (ESMF) [11] and the Earth System Prediction Suite [12]. ESMF and *Atlas* both provide similar fundamental building blocks for data structures and model development. However, *Atlas* has the distinct primary goal of accelerating novel numerical algorithm development for emerging hardware architectures, compared to ESMF's effort to enhance collaborative Earth system model development. More specifically, novel discretisation methods using hybrid unstructured meshes require specialised data structures currently not available in ESMF. Atlas does not aim to be used as an alternative to a coupler's "super structure", designed to couple different models or model components, but rather intends to be a flexible toolkit of components that can be combined to create custom parallel data structures.

The implicit assumption behind the design of *Atlas* is the ability to exploit the structure that may be present in a physical system. In a global NWP or climate model for example, there is a strong asymmetry of large horizontal and small vertical scales with dominant hydrostatic balance in the Earth's atmosphere. Moreover, there is a need to efficiently model the statistical effect of sub-grid-scale processes. The latter are typically arranged in independent vertical columns constituting what is called *physical parametrisation*, coupled via process-splitting to the atmospheric flow itself, called the *dynamical core*. The dynamical core in global NWP integrates the PDEs on the sphere whereas limited-area models (LAMs) solve the PDEs on a particular area of the sphere [13] to which lateral boundary conditions are supplied by a global model. Alternatively a LAM may be directly embedded into a global domain [14]. For more information on NWP and climate models, refer to [15].

Historically the development of an operational NWP model takes about 10 years. Therefore, it is imperative for *Atlas* to remain flexible and maintainable, aiming towards substantially reducing this development time. Given the aforementioned assumptions and restrictions, *Atlas*' aims are:

- Target massively parallel global and limited-area NWP and climate applications such as new dynamical core developments, and pre- and post-processing tools.
- Offer flexibility of choices in new numerical strategies and algorithmic paradigms under the same software platform to explore emerging hardware such as many-core architectures.
- Facilitate the implementation of different structured and unstructured point distributions on the sphere (global grids) and on limited areas of the sphere (non-global grids).
- Support different spatial discretisation strategies, such as the spectral transform approach currently used at ECMWF [7], compact-stencil finite volume methods [16–18], discontinuous spectral element methods [19–21] and possibly others, to solve the set of PDEs forming the dynamical core.
- Provide an array-type container to store variables (or fields) that is parallel-enabled variables (or fields) and provides support for domain-specific languages (DSL) such as GridTools [22]. The latter is achieved through an advanced data storage layer (for instance, Kokkos [23] or the GridTools native storage layer [22]). This core aim tries to ensure optimal use of emerging hardware such as many-core architectures.
- Support different programming languages, including Fortran and C++, providing object-oriented (OO) designs and data structure flexibility, so that *Atlas* can be used to update existing and support new code infrastructures.
- Provide object-oriented programming interfaces enhancing multi-disciplinary collaboration at multiple levels ranging from e.g. high-level mathematical operators, typically developed by domain scientists, to low-level data-storage abstractions, typically maintained by computer scientists.

The development of the *Atlas* library is part of the wider 'Scalability Programme' ongoing at ECMWF and *Atlas* represents one of the core strategic software infrastructure tools that ECMWF has initiated during the FP7 funded project on Collaborative Research into Exascale Systemware, Tools and Applications (CRESTA, http://www.cresta-project.eu). A first public version of the *Atlas* library is intended to be delivered as part of ESCAPE (http://www.hpc-escape.eu), a H2020 funded initiative that aims at finding energy-efficient numerical solution for Exascale computations [8], for which *Atlas* capabilities are used in the implementation of alternative numerical discretisations on emerging hardware.

---

[1] *Products* are fields that are disseminated upon request and post-processed to satisfy a customer's specific requirements.