Crossing the Chasm: Towards common Software Infrastructure for Earth System Model development.

IS-ENES2 is hosting a workshop in Reading (UK) on the 24th/25th of October (midday to midday) to discuss broad principles for aligning coding efforts "in the chasm" between hardware aware libraries (such as MPI) and the scientific libraries needed to build weather and climate models. The workshop aims to bring together those working on these future technologies and is intended to inform current and future developments so as to maximise community resources for the computing challenges ahead. The workshop will precede a larger workshop on the infrastructure (including software) needed for the next decade of climate science, being held at the same venue.

A registration website will be made available shortly as this meeting will be limited to 30 participants. In the mean time, interested participants should hold the dates.

Bryan Lawrence (Meeting Organiser)

Background

The next generations of high performance computing are likely to involve significant heterogeneity: both in terms of memory structures and underlying compute (graphical processing units, Intel multi-core, Power, ARM, FPGAs etc). This heterogeneity will be accompanied by rapid change in the hardware offerings at any given time, meaning that systems procured more than a few years apart may have significantly different architectures. This is very different from the past decades of relatively slow change of homogeneous systems, which allowed community codes to evolve slowly on much slower timescales than the hardware. It is unlikely that these existing coding practices will allow science codes to evolve fast enough to exploit new hardware (and possibly not even to evolve fast enough to survive).

The European weather and climate community are not well prepared to meet this challenge, primarily because the need to prepare for future computing has to be tensioned against the need to meet present day scientific challenges, and no part of the community has had enough resource to fully support both objectives, and the tactical imperative to "get science done" has dominated. There are of course notable exceptions discussed below, but these are still relatively small initiatives compared to the scale of the problem.

The general solution to this problem has been some combination of (1) significant re-engineering for hybrid parallelisation in the hope that a future directive based compiler will handle the problem; (2) exploiting new programming models which attempt to further abstract the hardware away from the science code (e.g. Mozdzynski et al 2015[1]); and (3) the use of Domain Specific Languages (in weather and climate notable examples are Gung-Ho and COSMO[2]). While these are all sensible approaches, it can be argued that no institute can explore all of them, and a more community focused approach is necessary.

One of the reasons why the scientific community has been able to exploit the massively parallel computing which dominates the HPC landscape today has been the pervasive availability of high-performance Message Passing Interface (MPI) libraries. It has been argued[3] that the weather and climate community need to establish similar libraries at a higher-level in the stack ("crossing the chasm") with support for common problems such as stencil generation, kernel execution, and memory mapping. Of course these are very similar to the tools that have been developed in the Gung-Ho and COSMO projects (op cit) and possibly in more generic libraries such as Kokkos[4], and the argument is similar to that espoused by Schultess (2015)[5]. The aim of this workshop is to explore commonalities in these approaches and the potential of more community based working, possibly beginning with more sharing of common practice.

---

[1] A PGAS implementation of the ECMWF IFS, Int. J. HPC. Apps., 10.1177/1094342015576773

[2] Gung-Ho: Ashworth et al, 2015, http://goo.gl/obOQkW; COSMO: Fuhrer et al, 2014, https://goo.gl/lgaWSB.

[3] On the path to exascale; will we be hungry? Rezny 2016, https://goo.gl/RWggM9.

[4] Manycore performance-portability: Kokkos multidimensional array library. Edwards et.al. 2012. 10.3233/SPR-2012-0343

[5] Programming Revisited, Schulthess (2015), 10.1038/nphys3294