Middleware for Memory and Data-Awareness in Workflows

Run time: 3-year project, started in September 2018

Current status: Description of the requirements of partners’ applications and workflows in order to design the first specification of the Maestro middleware API

Next steps: Finalize the specifications of the Maestro middleware architecture and design the demonstrators for the ECMWF and SIRIUS use cases

Motivation

• HPC and HPDA workloads are more and more I/O-intensive
• Performance bottlenecks are usually in the memory and storage systems
• Reducing and minimising data movement is very hard in general
• The HPC software stack was designed in a different era, to solve a different problem
• Few abstractions exist that capture data semantics of applications, so reasoning about data movement and memory in software is impossible
• Few useful models of memory systems and data movement exist, so estimation of cost of data movement is hard
• The memory-storage hierarchy is becoming more heterogeneous and complex, so a unified API and automatic promotion are needed

Approach and Methodology

• Co-design: ascertain data movement and access requirements of target applications
• Develop new data-aware abstractions:
  • Used in any level of software (compiler, runtime, application)
  • Relevant for any type of data (array, file, unspecified)
• Design a middleware and library that enables:
  • Modelling of memory hierarchy
  • Reasoning based on cost of moving data objects
  • Automatic movement and promotion of data in memories
  • Powerful data transformations and optimisation
• Explore data-based performance portability of Maestro applications

Selected Co-Design Use Cases

• ECMWF workflow: acquire and assimilate observations, produce numerical forecasts, post-process output and deliver products to customers.
  • Large quantity (both number and size) of data transferred from forecast to post-processing produces a bottleneck, indexing and consistency challenge.
  • 100 TiB of 3 MiB fields indexed, and transferred, in one-hour time-critical window.
• ETHZ’s SIRIUS library: a domain specific library for electronic structure codes
  • Beyond data movement at a workflow level, Maestro will be capable of performing smart I/O within an application: for instance, from DRAM to GPU’s high bandwidth memory

Figure 1: Design of the Maestro middleware. The CDO (Core Data Object) is at the heart of Maestro’s design. It is used to encapsulate data and meta-data.

Figure 2: ECMWF workflow for weather forecast with and without the Maestro middleware to manage data movement.

Figure 3: Offloading on GPU in the SIRIUS library with and without Maestro.