Extended software development workshop in intelligent high throughput computing for scientific applications

Location: CECAM-IT-SIMUL, Politécnico de Torino, Turin, Italy
Dates: July 16, 2018 to July 20, 2018
Organizers: A. O'Cais, D.W.H. Swenson, L. Rondoni, A. Di Carlo

1 State of the art

High throughput computing (HTC) is a computing paradigm focused on the execution of many loosely coupled tasks. It is a useful and general approach to parallelizing (nearly) embarrassingly parallel problems. Distributed computing middleware, such as Dask or COMP Superscalar (COMPSs), can include tools to facilitate HTC, although there may be challenges extending such approaches to the exascale.

Across scientific fields, HTC is becoming a necessary approach in order to fully utilize next-generation computer hardware. As an example, consider molecular dynamics: Excellent work over the years has developed software that can simulate a single trajectory very efficiently using massive parallelization. Unfortunately, for a fixed number of atoms, the extent of possible parallelization is limited. However, many methods, including semiclassical approaches to quantum dynamics and some approaches to rare events, require running thousands of independent molecular dynamics trajectories. Intelligent HTC, which can treat each trajectory as a task and manage data dependencies between tasks, provides a way to run these simulations on hardware up to the exascale, thus opening the possibility of studying previously intractable systems.

This workshop aimed to produce four or more software modules related to intelligent HTC, and to submit them, with their documentation, to the E-CAM software module repository. These included modules adding HTC support to existing computational chemistry codes, where the participants brought the codes they are developing. They may also include modules adding new middleware or adding features to existing middleware that facilitate the use of HTC by the computational chemistry community. This workshop involved training both in the general topic of designing software to interface with HTC libraries, and in the details of interfacing with specific middleware packages.

The range of use for intelligent HTC in scientific programs is broad. For example, intelligent HTC can be used to select and run many single-point electronic structure calculations in order to develop approximate potential energy surfaces. Even more examples can be found in the wide range of methods that require many trajectories, where each trajectory can be treated as a task, such as:

* rare events methods, like transition interface sampling, weighted ensemble, committor analysis, and variants of the Bennett-Chandler reactive flux method;

* semiclassical methods, including the phase integration method and the semiclassical initial value representation;

* adaptive sampling methods for Markov state model generation;

* approaches such as nested sampling, which use many short trajectories to estimate partition functions.

The challenge is that most developers of scientific software are not familiar with the way such packages can simplify their development process, and the packages that exist may not scale to exascale. This workshop will introduce scientific software developers to useful middleware packages, improve scaling, and provide an opportunity for scientific developers to add support for HTC to their codes.

2 Training provided

In practice, many scientific programmers are not aware of the range of middleware to facilitate parallel programming. When HTC-like approaches are implemented as part of a scientific software project, they are often done manually, or through custom scripts to manage SSH, or by running separate jobs and manually collating the results. Using the intelligent high-level approaches enabled by distributed computing middleware can simplify and speed up development. Major topics that were covered included

* Concepts of HTC; how to structure code for HTC,
* Accessing computational resources to use HTC,
* Interfacing existing C/C++/Fortran code with Python libraries,
* Specifics of interfacing with Dask/PyCOMPSs,
* Challenges in using existing middleware at extreme scale.

Furthermore, middleware frameworks can meet the needs of many different computing infrastructures. For example, in addition to working within a single job on a cluster, COMPSs includes support for working through a cluster's queueing system or working on a distributed grid. Moreover, architecting a software package such that it can take advantage of one HTC library will make it easy to use other HTC middleware. Having all of these possibilities immediately available will enable developers to quickly create software that can meet the needs of many users.

This E-CAM Extended Software Development Workshop (ESDW) focused on intelligent HTC as a technique that crosses many domains within the molecular simulation community in general, and the E-CAM community in particular. Teaching developers how to incorporate middleware for HTC matches E-CAM's goal of training scientific developers on the use of more sophisticated software development tools and techniques. The primary goals were:

1. To help scientific developers interface their software with HTC middleware.
2. To benchmark, and ideally improve, the performance of HTC middleware as applications approaching extreme scale.

The second portion of the workshop focused exclusively on HTC enabled by Dask, and on the related libraries dask-jobqueue and jobqueue_features. The latter library is developed by E-CAM, largely in the context of the ESDW, and focusses on enabling MPI-aware tasks within Dask.

Lectures from this workshop were recorded and stored on E-CAM's training portal at https://training.e-cam2020.eu/collection/5b3b5f27e4b0d62a7508ccd2

3 List of software development projects

The software package "jobqueue_features" (https://github.com/E-CAM/jobqueue_features) was entirely developed by E-CAM in collaboration with PRACE. There are a number of modules submitted to the E-CAM repository related to development work for this package:
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/83
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/84
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/85
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/86
with more related modules currently being prepared.

There are also user-related modules submitted
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/50
* https://gitlab.e-cam2020.eu/e-cam/E-CAM-Library/merge_requests/150
again with more modules under preparation.

4 Future plans

The second face-to-face meeting took place at the beginning of July 2019. The workshop was 3.5 days long consisting of 1.5 days with three different Python libraries related to Dask:

*Dask: https://docs.dask.org/en/latest/
*Dask_jobqueue: https://dask-jobqueue.readthedocs.io/en/latest/
*jobqueue_features: https://github.com/E-CAM/jobqueue_features

The last library is something that was developed between the two workshops by E-CAM. It allows the user to create tasks that call out to MPI programs, and easily configure the tasks to run on different types of resources (CPU/GPU/KNL). The final 2 days were a hackathon where people could work on their own use case with technical assistance. Lectures from the second part of this workshop were recorded and stored on E-CAM's training portal at https://training.e-cam2020.eu/collection/5d1a5ddfe4b06ec7bbfe4d15 .

Beyond this, future developments would be to continue to support jobqueue_features and expand its capabilities and resilience.

5 Participant list

Organizers

O'Cais, Alan
Jülich Supercomputing Centre, Germany
Swenson, David
École Normale Supérieure de Lyon, France

Adamska, Lyudmyla - University of Padova, Italy
Badia, Rosa - Barcelona Supercomputing Centre, Spain
Conejero Bañón, Francisco Javier - Barcelona Supercomputing Centre, Spain
De Angelis, Paolo - Politecnico di Torino, Italy
Giulini, Marco - University of Trento, Italy
Kirmizialtin, Serdal - NewYork University Abu Dhabi , United Arab Emirates
Liang, Yanyan - ICAMS, Ruhr-Universität Bochum, Germany
Malis, Momir - CECAM, EPFL, Switzerland
Meinke, Jan - Forschungszentrum Juelich, Germany
Menon, Sarath - ICAMS, Ruhr University Bochum, Germany
Roet, Sander - Norwegian University of Science and Technology,
Shaidu, Yusuf - International School for Advanced Studies, Italy
Troncoso, Javier - QUB, United Kingdom
Uchronski, Mariusz - Wroclaw Centre for Networking and Supercomputing, Poland
van Dijk, Marc - VU Amsterdam, The Netherlands
Vitale, Valerio - University of Cambridge, United Kingdom
Wlodarczyk, Adam - Wroclaw Centre for Networking and Supercomputing, Poland
Zapata, Felipe - Netherlands eScience Center, The Netherlands