Toward improved scientific software productivity on leadership facilities: An Argonne Leadership Computing Facility View

Venkatram Vishwanath, Thomas Uram, Lisa Childers, Hal Finkel, Jeff Hammond,
Kalyan Kumaran, Paul Messina, Michael E. Papka
Argonne National Laboratory
<venkat, turam, childers, hfinkel, kumaran, messina, papka>@anl.gov

Foster a stronger and deeper engagement for extreme-scale software development between application and library developers and DOE leadership facilities staff

In order to successfully conduct science at extreme-scales a deep understanding of both science goals and technology constraints will be needed. Computational and computer scientists at DOE leadership facilities have expertise in the development of production-scale scalable software and algorithms for scientific computing. Additionally, these scientists have deep interactions with the application projects using the leadership resources. They also have early access to new hardware, and have been involved with several generations of supercomputers, thus, gaining a deep understanding of the underlying abstractions needed for scalable and performance portable software. Engaging these best-of-the-breed computational scientists and including them as part of computational science teams and collaborations would be of paramount importance toward scalable science on DOE systems.

Additionally, access to a test and development infrastructure at leadership facilities would ensure that software libraries are ported to the DOE computational facilities. A testing framework at the facilities would help users understand the state of the various libraries and their robust features. Facility experts would also interface these testing frameworks with the schedulers and likely develop relevant policies for unit tests, among others, at scale on the leadership systems.

Design abstractions and a performance projection framework to adapt science codes to disruptive architectures

Several libraries, including BLAS and LAPACK, have been successfully ported to a wide variety of architectures, and have achieved scalable performance on these systems. A primary reason for this success has been the ability to abstract out the interface and separate the implementation while keeping the interface common across systems. The latter hides the complexity of the underlying system, and thus enables productive science. As we move to future disruptive systems, a concerted effort in designing usable abstractions for software libraries will be needed. This is akin to development of intuitive user-interfaces for devices. Interfaces (APIs) have a longer life than any given implementation, as they are more easily reused and refined to meet emerging requirements. Non-intrusive techniques using mechanisms such as library interposition will reduce the barrier of integrating new extreme-scale libraries with applications.

Adapting a library to a disruptive architecture requires significant investments in both rewriting the code and in several cases the underlying algorithm. A performance projection framework that characterizes and transforms scientific applications for improved performance on extreme-scale systems would be of paramount importance. This framework would need to abstract application behavior, model hardware architectures, and develop the necessary transformations and project the performance of the extreme-scale architectures. This would enable us as a community to understand where one should invest resources, including re-implementation or new algorithms, to transform science onto extreme-scale machines.

Manage extreme-scale data from scientific simulations for increased productivity

Scientific simulations generate enormous amounts of data that must be captured, cataloged, and analyzed. Unless this is formally done, monitoring such simulations, tracking and reproducing old ones, and

analyzing and archiving their output, can be haphazard and idiosyncratic. A simulation management and analysis system would solve some of these problems by offering an intuitive single point of control and analysis, a metadata database, plugins to incorporate application-specific tools and functions, and a set of tools that automate some of what scientists have been doing by hand. Investments to foster research and development towards this effort would lead to improved scientific productivity as well as enable reproducible results and papers.

Enable new interaction modalities for computational science campaigns on extreme-scale systems

- Develop abstractions to facilitate seamless access to remote resources and repositories, and provide capabilities to perform computations on these distributed resources. These abstractions would need to account for the trends in scientific computing wherein the teams are increasingly collaborative and geographically distributed.
- Exploratory science involves a scientist making interactive queries as well as modifying simulation parameters at run-time. Interfaces to enable scientists-in-the-loop will be critical for improved productivity.
- To handle the volume and complexity of data together with the increasing mismatch in capabilities between computation and storage, we will need to develop interfaces and infrastructure to incorporate simulation-time analyses and visualization for science, aiding efficient use of large scale I/O infrastructure and networks.

Improve user productivity by providing high-performance, standards-compliant, and familiar programming and runtime environments (leveraging open-source software where possible)

User productivity on extreme-scale systems can be significantly improved by providing software tools and environments on par with what is available on the workstations and small clusters available to most science-code developers. By investing in standards-compliant and open-source software for leadership-class systems, we will significantly reduce the associated learning curve, and increase the probability that users will be able to get up and running quickly. Increased user productivity will enable more time to be spent on scientific investigation using the resources we provide, and will make it easier to attract new users to our facilities. To achieve this goal, one requires a close collaboration between vendors, open-source tools and libraries, leadership-facility staff, and application developers. The MPICH and LLVM/CLANG projects are examples of successful open-source collaborations on leadership systems.

Develop a Knowledge Base for extreme-scale computational science

Many new users will face a steep learning curve mastering the techniques needed to transform their science goals into codes suitable for execution at extreme-scale. A DOE-wide knowledge base capturing best practices for developing computational science software on leadership systems would significantly lower the barrier for new users and software. The knowledgebase would include:

- Workshops and tutorials covering the methodologies and approaches to using leadership systems and achieving performance portability.
- A set of micro-benchmarks, mini-applications, and full applications, and the steps involved, including algorithms, to improve the performance at scale.
- Performance provenance of software on a wide variety of platforms. This will help one understand how a change in the software affects the performance at scale on a variety of systems.
- Reproducible publications together with results and source code.
- Best practices for developing portable and scalable software.