Kokkos/RAJA

Exascale systems are characterized by computer chips with a large number of cores, a smaller amount of memory, and a range of various architectures, which can result in decreased productivity for library and application developers who need to write specialized software for each system. The Kokkos/RAJA project provides high-level abstractions for expressing the necessary parallel constructs that are then mapped onto a runtime to achieve portable performance across current and future architectures, freeing developers who adopt these technologies of the burden of writing specialized code for each system.

Library and application developers are confronted with the challenges of inventing new parallel algorithms for many-core chips while learning the different programming mechanisms for each architecture and creating and maintaining specialized performant code for each. Adapting libraries and application software as the architectures evolve and become more complex to attain improved performance is a large time investment. The purpose of the Kokkos/RAJA project is to provide portable abstractions that can be adopted by developers to reduce or eliminate this overhead and improve developer productivity.

Kokkos provides a C++ parallel programming model for performance portability that is implemented as a C++ abstraction layer including both parallel execution and data management primitives. RAJA provides various C++ abstractions for parallel loop execution and supports constructs to reorder, aggregate, tile, and partition loop iterations and complex loop-kernel transformations. RAJA's companion projects Umpire and CHAI provide portable memory management and smart data motion capabilities. Application and library developers can implement their code using Kokkos/RAJA, which will map their parallel algorithms onto the underlying execution mechanism using existing parallel programming models, such as OpenMP.

The Kokkos/RAJA team is focused on developing and optimizing backends to support the Aurora and Frontier systems. These backends will ensure that libraries and applications built with the Kokkos/RAJA abstractions will run and achieve high performance on these exascale systems without requiring the library and application developers to change their code, even if these architectures require their own custom programming mechanism.

PI: Christian Trott, Sandia National Laboratories

Collaborators: Sandia National Laboratories, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Oak Ridge National Laboratory

Progress to date

- The Kokkos team developed a parallel programming model with flexible enough semantics that it can be mapped on a diverse set of exascale architectures including current multi-core CPUs and massively parallel GPUs.
- The Kokkos library implementation consists of a portable Application Programming Interface (API) and architecture-specific backends, including OpenMP, Intel Xeon Phi, and CUDA on NVIDIA GPUs.
- The RAJA team produced a collection of C++ software abstractions that enable architecture portability for exascale applications using standard C++11 features and provided support for multiple backends including OpenMP, CUDA, Intel TBB, and AMD GPUs.
- The Kokkos/RAJA team developed training material and held training events to enable adoption of their abstractions.