PaRSEC: Distributed Tasking at Exascale

One difficulty associated with programming exascale systems is expressing the tasks comprising a scientific simulation and then mapping them to the heterogenous computational resources on that system, while achieving high performance. PaRSEC supports the development of domain-specific languages and tools to simplify and improve the productivity of scientists when using a task-based system and provides a low-level runtime that seamlessly leverages the combined computing power of accelerators and manycore processors at any scale when executing the tasks.

PaRSEC helps application developers express dataflow parallelism using domain-specific languages and tools and then maps and executes the resultant program on exascale systems with heterogenous computational and memory resources. The team's interaction with scientists includes building domain-specific languages that both suit their needs and facilitate the expression of algorithmic parallelism with familiar constructs. The runtime maps the resultant tasks to the hardware and supports heterogeneous architectures and accelerators and data transfers between different memory hierarchies.

The PaRSEC team focuses on (1) increasing programming flexibility using domain-specific languages benefiting from optimized runtime components, architecture-aware coverage of all target architectures, and reduction of overheads; (2) extending the programming system to new composable paradigms; and (3) providing a production-quality runtime with documentation, testing, packaging, and deployment. This work enables libraries and applications developed by the ECP to efficiently use exascale systems in a pure dataflow programming environment, whereas the domain scientists focus mainly on algorithmic aspects and leave the architectural details and optimizations, such as overlapping of communication/computation and data movement, to the runtime supporting the programming paradigm.

The PaRSEC team has improved their runtime on multiple levels. At the low level, key elements have been modularized and exposed for end-user control. Node-level task schedulers and GPU managers have been designed that support hyperthreading to offload scheduling decisions. The communication subsystem has been extended to take advantage of remote memory access hardware support and to improve the general performance of distributed applications. Critical limitations on the internal representation of the tasks tracking and dependencies tracking have been removed by opting for scalable, efficient, open addressable data structures suitable for shared memory parallelism on many-core architectures. Support for heterogeneous hardware has been improved and includes better memory management strategies, which allow tackling problems many times larger than the available memory on the accelerators without a significant performance penalty. Proof-of-concept integrations with libraries and applications supported by the ECP show promising performance at large scale.

PI: George Bosilca, University of Tennessee – Knoxville

Collaborators: University of Tennessee – Knoxville

Progress to date

- The PaRSEC runtime has been continuously improved to support the exascale architectures and has been integrated with other program models/frameworks and with performance and correctness tools. These new capabilities have been evaluated in a distributed heterogeneous environment.
- The team has also designed programmatic interfaced-to-prefetch data on accelerators that provide memory management advice to the accelerator engine to improve scalability and performance.
- The team has dedicated effort to improve to software quality and usability. Tutorial material has been created to facilitate user adoption, along with developer and user documentation. To ensure that users have reliable access to all capabilities of the runtime system, continuous integration tools have been streamlined in the development process.