Runtime Systems and Programming Environments for the Future

DOE X-Stack PI Meeting
May 28-29, 2014

Lauren Smith
DOD
lauren.l.smith18.civ@mail.mil

Goals and Objectives

To ensure that there are productive programming environments and runtime systems for future extreme-scale systems that enable performance, resilience and power efficiency for mission applications

Programming Models

Chapel

- Enabling model for future applications on future HPC Systems
- Multi-year on-going engagement
- Make Chapel to be in a “production-grade” state
- Strong focus on increasing performance, scalability
- Developing heterogeneous compute support (eg. GPUs, Knights XXX, etc.)
- http://chapel.cray.com

Programming Models

OpenSHMEM

- Focus on maturing API towards modern HPC capability (e.g. richer AMOs)
- Focus on improving programmer productivity (e.g. PE Teams)
- Universal Common Communications Substrate (UCCS) Library
- http://openshmem.org

Unified Parallel C (UPC)

- Vendor-developed extensions for improving performance critical communication primitives
- http://upc.gwu.edu or http://upc.lbl.gov or www.gccupc.org

Enabling Programming

Qthreads

Enabler for high-performance task parallelism in Chapel
https://code.google.com/p/qthreads/

GASNet

Communications layer enabling portable implementations for Chapel, UPC and OpenSHMEM
http://gasnet.lbl.gov

Global Name Space Productivity

Exploring higher-productivity programming environments (e.g. Python, C++) for PGAS development

Tools

Enhancements as needed

Runtime Systems Research

Advanced Computing Initiative BAA issued through Army Research Office

- Programming Environment
- Runtime Environment
- Machine Environment

Objectives

- Performance
- Reliability
- Energy Efficiency

Advanced Computing Runtime

- Build non-blocking runtime features in support of dynamic code/data placement and movement to accommodate power efficiency, performance and reliability

- Exploit COTS hardware features to support runtime monitoring and control

- Exploit automated adaptive techniques and explicit hints from the programming environment

- Rice University/Intel

Throughput Oriented Runtimes for Large Scale Manycore Systems: THOR

- Develop throughput oriented runtimes for future many-cores; and lay the foundations of system level adaptive power management
- Develop micro-benchmarks, metrics, control theory for system wide power/energy
- Develop portable runtime able to provide optimal network throughput
- Demonstrate for Chapel/UPC runtimes at scale
- LBL/Cray

Programmer-Guided Reliability and Tradeoffs with Energy and Performance

- Develop a better understanding of the efficacy of error detection and correction mechanisms at the application level, and their impacts on energy and performance.

- Develop techniques to automate the insertion of detectors/correctors, and to manage their use to meet energy/performance goals.

- Proof of concept using Chapel

- ORNL/Cray

Integrated Compiler and Runtime Autotuning Infrastructure for Power, Energy and Resilience

- Integrated compiler and runtime decision framework to minimize the energy consumed by complex mission applications on next generation systems
- Auto-tuning graph/linear algebra analytics framework with offline analysis and online runtime introspection
- Just in-time compilation based on online power information
- Machine learning-base techniques for adaptive execution scenarios
- PNNL, Ohio State, U. Delaware, Stanford, UNC/RENCI

FAIL-SAFE: Fault Aware Intelligent Software for Exascale

- Demonstrate the use of both domain-specific application-specific knowledge to substantially increase the resilience of applications expected to run on extreme-scale HPC platforms in the 2020 time frame

- Users will be provided with access to adaptive, application-oriented controls for fault tolerance

- USC/ISI, LLNL, JPL

Summary

Complementary to DOE X-Stack and OS/Runtime Programs

Tackling many of the same issues related to
performance, productivity, energy and resiliency

Opportunities for collaboration