2015 ECI Runtime Systems Workshop – Questions and Issues

Dates: March 11 - 13, 2015
Location: Rockville Hilton, Rockville, MD

Workshop Overview

Future runtime system software must achieve significant improvements in efficiency and scalability in the context of user productivity, performance portability, and operational uncertainty due to asynchrony.

The primary purpose of this workshop is to enable a detailed technical exchange among key thinkers within the exascale system research and development community on the topics of runtime systems.

At this workshop, we need to discuss previous research experiences and technical strategies used in the research and development of runtime system prototypes. We need to determine the required characteristics of future extreme scale runtime systems and discuss the vision for them. Finally, there is need to determine the research questions that must be addressed and the roadmap for R&D in the context of the Exascale Computing Initiative, including the plan to produce high quality runtime systems software packages to be deployed on extreme scale systems in the 2023 timeframe.

We need to formulate a conceptual framework for evaluating prototype runtime systems (e.g., metrics, measures, benchmarks, test plans), as well as the integration framework with programming models, and other software system layers.

This is not an attempt to formulate a vision by a consensus building exercise, but rather an attempt to enable an open and technically charged context for discussions to inform future planning.

Workshop Objectives

1. Propose, discuss, and determine the required characteristics of future extreme scale runtime systems
2. Devise metrics, measurements, benchmarks, and other means for testing and evaluation for prototypes of runtime systems,
3. Identify research questions that need to be resolved within the context of current experience and knowledge,
4. Discuss a research and development roadmap that will result in one or more high quality runtime system software packages that could be deployed in the 2023 timeframe, on extreme scale systems.

Workshop Questions and Issues

The following two focus areas will be addressed by the speakers in their talks and discussed during the breakout sessions with all participants. The detailed issues and their associated questions define the general focus areas within the scope of current understanding. System Architecture refers to the entire system, hardware and software, in which the runtime system is integrated. Runtime System Design refers to the properties and components of the runtime system itself. Together, context and construct cover the breadth and depth of future runtime systems within exascale computing.
Focus Area 1: System Architecture

- **Execution model**: Discuss the governing principles of the strategy of computation that guides the runtime system (together with the other system component layers) in addressing the challenges and issues discussed here. Discuss the necessary features of a cogent, comprehensive, and self-consistent execution model upon which to base a runtime system design.

- **Asynchrony**: Discuss the dramatic increase on variability of latency of communication. Considering shared resources such as networking, memory banks, discuss the increased uncertainty of action/task response time requiring adaptive methods managed by the runtime. Discuss the semantics and control strategy in the presence of the uncertainty imposed by asynchrony. Identify the hardware support that facilitates task management. Identify the differences between supporting synchronous and asynchronous programming models.

- **System fragmentation**: Discuss whether a runtime system should operate across the total system span (or sub-span of the app) or if it should operate on separate local domains (nodes).

- **Relationship between OS and runtime system**: Given that the OS is persistent and is the surrogate for the hardware resources and that the runtime reflects the workload demands and properties of the application program, discuss the relative roles and responsibilities of these two in cooperation and what are the logical interfaces and protocols by which current information is exchanged.

- **Relationship between Programming models and runtime systems**: Establish the basic requirements of future runtime systems in terms of their relationships with the programming interfaces and compilation strategies. Explore the nature of programming models that will employ runtime systems for management of dynamic adaptive execution to guide co-design of programming and runtime methods.

- **Compile time information, guidance, and constraints**: Discuss new forms of compile time information that is fed to the runtime to help it better perform its responsibilities adaptively? This is a major new area of consideration, perhaps among the biggest innovations of future system design.

- **Evaluation**: Discuss the metrics, measures, benchmarks, and means for testing and evaluation of the runtime system architecture.

Focus Area 2: Runtime System Design

- **Memory models, name space, and address space**: Discuss the abstraction of data storage and how does it extend across the system for the parallel application. To what extent is the management of user memory and virtual addressing done by the runtime system and how? What does it assume are the resources provided by the operating system and underlying hardware?

- **Introspection interfaces, policies, and control**: A major motivating factor for the incorporation and exploitation of runtime software is to achieve substantial improvement in efficiency through dynamic adaptive techniques and greater scalability through parallelism discovery. Discuss introspection, control mechanisms, and advanced means of describing and asserting policies to adjust to changing conditions and requirements. Identify system components (hardware and/or
software) that must provide the necessary information to the runtime and how interfaces must be presented to minimize overhead.

- **Contribution to tools:** Discuss the development of new ways to debug ultra-scale programs. The role of the runtime system may be crucial in contributing to debugging as such errors from flawed problems will occur on the fly among many other simultaneously executing tasks. In addition, as the community grapples with new architectures, performance analysis plays even a more important role in application development and porting. Discuss the approaches that the runtime system, working with the compiler, can use to diagnose faults and performance issues.

- **Parallelism forms, granularity, and synchronization:** Discuss how the runtime system manages parallelism of activity and determines the granularity of size of the threads to balance the anticipated overheads. How does the runtime actually make this work and adapt between lazy and eager evaluation? Discuss support of communicating processes, fork-join parallelism, and other support for increased forms of parallelism.

- **Contribution and responsibility to reliability:** Future methods of sustaining execution in the presence of faults (soft and hard) will have to extend beyond pure hardware solutions and conventional, coarse-grained checkpoint-restart methods. With increases in number of cores per sockets, number of sockets per node, network communication channels, DRAM chips and banks, and the potential use of near threshold logic (for lower power), the MTTI is likely to reach minutes or even tens of seconds instead of hours or days. Discuss the solutions to the fault tolerance problem that engage specific capabilities of the runtime system software. Discuss the precise role, strategy, and mechanisms of the runtime system to address the challenge of resiliency.

- **Contribution and responsibility to energy/power management:** Dynamic control of power consumption through a number of available mechanisms including voltage level control, clock rate control, reconfiguration (turning pieces on and off), and relative locality of data and tasks (communication power) provide means by which the runtime system can influence the amount of power the system consumes, the cost of energy used, and the thermal properties of the physical elements. Discuss the state-of-the-art in strategies and methods that the runtime system, in conjunction with both the operating system and the compiler, needs to utilize in order to optimize total time to solution while minimizing energy costs.

- **System Hardware Control:** Discuss the opportunity to dramatically reduce the programmer’s time by assuming much of the burden of system hardware control conventionally done explicitly by the programmer. The uses of the runtime to establish the invariants of the application program, which are independent of the underlying execution platform, contribute to productivity and performance portability. Articulate the advantages for the programmer when optimizing the end-user code for each different machine differing in scale, type, or generation. How can these advantages be quantified/measured?

- **Evaluation:** Discuss the performance models, metrics, measures, benchmarks, and means for testing and evaluation of the runtime system design.
**Workshop Structure**

The workshop will consist of the collection of talks and breakout discussion sessions. The invited runtime system speakers will address as much as possible the two focus areas above.

The following is a list of the breakout sessions:

- Sessions I & II: Runtime Systems Architecture
- Sessions III & IV: Runtime Systems Design
- Sessions V & VI: Runtime Systems Research Questions
- Sessions VII & VIII: Runtime R&D Roadmap

The questions listed above for Focus Area 1 will be discussed in the runtime system architecture breakout sessions (Parallel Sessions I and II).

The questions listed above for Focus Area 2 will be discussed in the runtime system design breakout sessions (Parallel Sessions III and IV).

In the Research Questions breakout sessions (Parallel Sessions V and VI), participants are asked to identify research questions that need to be resolved within the context of current experience and knowledge. The questions listed on Focus Areas 1 and 2 above can be helpful in guiding the discussion.

The roadmap discussions (Parallel Sessions VII and VIII) should leverage conclusions reached from discussing all of these issues and in addition should:

- Define strategy for sustaining multiple concepts/approaches within the scope of an overall vision, and
- Create a straw-man example of what an effective research agenda would look like in terms of accomplishments, useful intermediate research results, demonstrations, and application advances through increased scaling and efficiency.