Runtime System Software - Summit

April 9, 2014

American Geophysical Union (AGU)

2000 Florida Ave. NW, Washington, D.C. 20009

Summit Goal

The purpose of this summit is to discuss current challenges in the area of runtime systems and to generate a roadmap for generating a unified runtime systems *architecture* for Exascale systems that has broad community acceptance and that leverages investments DOE and other government agencies have made in the last several years in this area.

We expect to leverage recent reports and community input on challenges that we face in the area of runtime systems to create a complete set of challenges and discuss the current state-of-the-art to deal with them. Expected outcome of this summit is a roadmap for runtime systems research going forward. As a starting point, we will develop a set of questions that must be answered. An initial set of questions is included in this document.

This summit is not an opportunity for participants to promote their current research agenda, but to take a fresh look into the future needs for Exascale runtime systems. The goal for this summit is to develop a unified runtime architecture with common components, not a single unified runtime software.

Motivation

A recent ASCAC report [1] has discussed some of challenges facing the research community in runtime systems, namely:

- For the different execution models, key abstractions need to be identified and jointly supported by the runtime system, compilers, and hardware architecture.
- A large number of lightweight tasks and their coordination will need runtime support that is capable of dealing with system heterogeneity and with end-to-end asynchrony.
- Locality-aware, dynamic task scheduling will need runtime support so that it is possible to continuously optimize when code or data should be moved.
- Task coordination/synchronization primitives that are best suited to support exascale systems need to be identified.
- Load imbalances created by a large number of sources for non-uniform execution rates will require runtime support to dynamic load balancing.

DOE Exascale Runtime Investments

ASCR has made a number of investments in runtime system software research for Exascale. In the 2012 X-Stack program [2], projects include application-driven runtime systems support. Research in this area is concerned with maximizing concurrency efficiency, properly dealing with asynchrony of computation and communication, exploiting data locality, minimizing data movement, managing faults, and the needed support for heterogeneous computing elements, sound semantics for programmability, support for novel programming models, and delivery of an efficient execution environment to application developers. A number of runtime systems are currently being pursued: OCR, HPX, ARTS, SEEC, GVR runtime [2] and runtimes to support advance/extended MPI and Global Arrays [3].

Research of system-driven runtime systems, supported by the 2013 OS/R Program [4], is concerned with mechanisms, including semantics, of "common runtime services," described in the OS/R report [5]. Examples of such mechanisms are thread management, low-level communication services, and resource management. Tight interaction among the different runtime service components has been identified as essential in order to deal with challenges of resilience, asynchronous computations, and locality of computation. A number of approaches of systems-driven runtime are currently being pursued in the ARGO, HOBBES, and X-ARCC projects.

Research on applications-driven and systems-driven runtime systems are addressing challenges of resilience, power, hierarchical memory management, unprecedented parallelism, heterogeneity of hardware resources, locality and affinity management. DOE ASCR has insisted that focus on self-aware, dynamical systems should guide most of the research solutions in these two categories. DOE ASCR has also strongly recommended that close coordination be established among the various runtime system research projects. However, a community-wide involvement in defining runtime architecture for Exascale computing remains elusive.

Initial Set of Summit Questions

1. Runtime system software architecture
    a. What are the principal components? What are the semantics of the components? What are the differences for the different execution models?
    b. What are the performance models or metrics that provide a means of comparison between alternative architectures or components?
    c. What are the mechanisms for managing processes and data?
    d. What is the role of a global address space or a global name space?
    e. What types of programming models can be supported by the runtime architecture?
    f. What is the runtime support needed for different programming models?
    g. What OS support should be assumed by Exascale runtime systems?
2. Community buy-in:
    a. How do we achieve community buy-in to an envisioned runtime architecture and semantics?
    b. We need a process to continuously evaluate refine the envisioned runtime architecture and semantics while keeping focus on achieving an Exascale runtime system. What should this process be?

Participants

Ron Brightwell

Kathy Yelick

Maya Gokhale

Wilf Pinfold

Andrew Chien

Sanjay Kale

Vivek Sarkar

Richard Lethin

Thomas Sterling

Vijay Saraswat

Sonia Sachs

Bill Harrod

Proposed Report Outline

1. Introduction
    1. Charter and Goal
    2. Vision
2. Technical Challenges (brief + metrics)
    1. Resilience (Andrew, Sanjay)
    2. Power/Energy (Vivek, Wilf)
    3. Memory Hierarchy (Milind, Thomas)
    4. Parallelism (Thomas, Rich)
    5. Communications (Kathy, Vijay)
    6. Scheduling and execution (Rich, Kathy)
    7. Dynamic resource allocation (Rich, Sanjay)
    8. Locality (Vivek, Milind)
    9. Legacy Support (Maya, Wilf)
    10. Sustainability and Portability (Wilf, Andrew)
3. Requirements
    1. Hardware Support (includes monitoring and control) (Wilf, Thomas)
    2. Operating System Services (Ron, Maya)
4. Services (challenges are detailed here)
5. Components (merge with services?)
    1. Computation Manager and Scheduling

    2. Address allocation/translation
    3. Location Manager
    4. Power Manager
    5. Memory Manager
    6. Threads/Tasks Manager
    7. Introspection Manager
    8. Persistent Data Store Manager
    9. Load Balancers
    10. Adaptive Controller
    11. Composability Manager
    12. failure detection/recovery management
6. Interfaces
    1. Hardware Abstraction Layer (Node): Vivek, Wilf
    2. Hardware Abstraction Layer (System): Thomas, Ron
    3. Operating System Interface: Ron, Maya
7. Major Services (some runtimes may not need these)
    1. Scheduling Service (Schedule and execute threads/tasks/work unit, including code generation): Vivek, Ron
    2. Resource management service (allocation, give me resources dynamically, as needed, release resources, including networks. heterogeneity, virtualization): Sanjay, Ron
    3. Introspection services (info about power, performance, and heterogeneity. Variability): Sanjay, Andrew
    4. Naming service: Sanjay, Thomas
    5. Communication of data and code, including synchronization (event-oriented). Migration services. Move work, move data. Is not separate from the communication services, it is composed with. (Kathy)
    6. Concurrency control service: isolation, atomics, event, synchronization (Rich, Thomas)
    7. Location and Affinity/Locality services: map to some things that are mentioned above. Provides information and does binding. (Milind, Vivek)
    8. Resilience Services: Express/configure error checking/detection and recovery at multiple levels. Allows specification of resilience properties. Both to computation and data and hardware resources. Message recovery, checkpoint-restart (Andrew, Sanjay)
    9. Load balancing service. Data Distribution and Redistribution (Kathy, Sanjay)

References

[1] Top Ten Exascale Research Challenges, DOE ASCAC Subcommittee Report, February 10, 2014.

[2] X-Stack Program: see the Extreme Scale Software Stack wiki, https://www.xstackwiki.com/index.php/Extreme_Scale_Software_Stack

[3] MPI and Global Arrays:
https://collab.mcs.anl.gov/display/mpiexascale/Evolving+MPI+to+Address+the+Challenges+of+Exascale+Systems;jsessionid=02076F47AB1AD12B97987FE748609B1C

[4] OS/R Program: see the Extreme Scale Software Stack wiki,
https://www.xstackwiki.com/index.php/Extreme_Scale_Software_Stack

[5] Exascale Operating Systems and Runtime Software Report:
http://science.energy.gov/%7E/media/ascr/pdf/research/cs/Exascale%20Workshop/ExaOSR-Report-Final.pdf