Providing Efficient Mechanisms for Scheduling in High Performance Virtualized Multicore Environment
Abstract

Oscar Mondragon

March 11, 2014

Resource allocation and scheduling in next-generation HPC systems is expected to be much more complex and dynamic than in current systems, encompassing multiple collaborating applications that coordinate activities within and across nodes. In current architectural designs, a high-level provisioner allocates nodes to applications based on a potentially diverse set of scheduling criteria These could include, optimization of performance given power constraints, or optimization of energy consumption given performance constraints. These policies must be translated and communicated to node-level schedulers which implement and enforce these policies.

In this presentation, I describe the research I am conducting on the node-level interfaces and mechanisms to support this scheduling architecture. My basic approach uses of synchronized per-core earliest deadline first (EDF) schedulers to address these challenges. EDF scheduling provides a simple mechanisms for enforcing wide range of high-level policies provided by the provisioner, but raises a number of important research questions.

First, I am researching interfaces for specifying key high-level provisioning use-cases and how to translate them into appropriate EDF schedules. High-level provisioning requests could be tightly or loosely specified. For tight provisioning use-cases, scheduling mechanisms are offered for applications with physical resources already allocated to them, while for the loosely provisioning case, local node provisioning must be provided by our framework.

Second, I am researching mechanisms for coordinating EDF scheduling across cores and nodes to handle application gang scheduling demands. Efficient synchronization mechanisms must be provided in order to meet performance and power constraints. Using an EDF scheduler will guarantee that minimum service levels are provided to applications.

Third, I am researching interfaces and mechanisms for cooperatively scheduling applications running on the same node, both on regular CPUs and on GPUs. Mechanisms to avoid performance degradation of high priority applications while space sharing resources with lower priority applications must be considered. On the other hand, special considerations must be taken in account for resources which are non-preemptive or where preemption is expensive, such as GPUs, as missed deadlines appear in this case, even at low utilization levels.

Finally, I am researching the feedback that the node-level scheduler will need to provide to the provisioner to support dynamic workload changes. Meaningful statistics must be provided in order to make the provisioner aware of the performance of the low level mechanisms, through efficient metrics. Since this monitoring information is helpful to dynamically adapt high level policies, it is important to properly identify those metrics.