Power Management Framework for Extreme-Scale Computing

Masaaki Kondo
Graduate School of Information Science and Technology, The University of Tokyo.
Information Technology Center, The University of Tokyo.

Introduction: Power consumption is expected to be a first class design constraint for developing future extreme-scale computing systems. To achieve exa-flops performance with realistic power provisioning of 20-30 megawatts, about 10x power-performance efficiency improvement over today's most power-efficient supercomputers is necessary. In order to maximize effective performance within a power constraint, we need a paradigm shift from the worst-case design strategy to the power-constraint adaptive system design (P-CAS), which allows the system's peak power to exceed maximum power provisioning with adaptively controlling power-knows equipped in hardware components so that effective power consumption at runtime is under the power constraint (Fig 1). This concept is known as overprovisioned system design. To realize extreme-scale systems with the P-CAS concept, we are conducting research and development of a software framework for code optimization and power management which adaptively controls power-performance knobs under a given power constraint [1]. Several key challenges that we need to address are as follows:

1) Framework to maximize application performance under a given power constraint.
2) Power aware job scheduling to maximize total system throughput and to minimize under-utilized power budget.
3) Power-performance simulation and analysis framework for extreme-scale applications.
4) Standardized API for power monitoring and control.

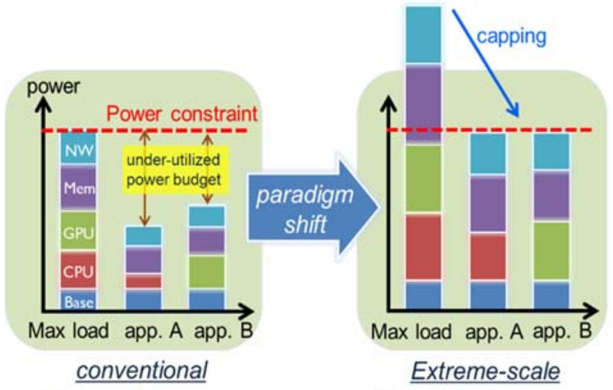
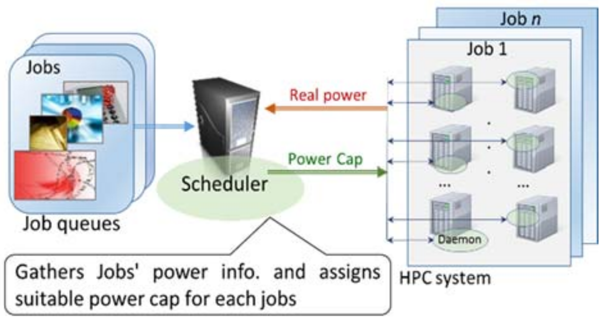


Fig. 1: Power constraint adaptive system.


Jobs
Job queues
Scheduler
Real power
Power Cap
Job n
Job 1
Daemon
HPC system
Gathers Jobs' power info. and assigns suitable power cap for each jobs
Fig. 2: Power-aware job scheduler.

Framework to maximize application performance under a given power constraint: There is a strong demand for maximizing application performance under a given power constraint. In fact, power provisioning demand to each hardware component is very divergent among different applications. For examples, compute-intensive applications prefer allocating much power for CPUs while memory-intensive applications prefer to invest power resources in memory bandwidth. To optimize power allocation among hardware component by application developers, we need to develop an optimizing framework which assists the following steps: i) collecting power-performance information with various power settings, ii) optimizing power allocation with power-performance knobs, iii) modifying the application code to control power-performance knobs appropriately. To this end, our framework contains a compilation framework which automatically generates application code for power-performance profiling and control. Moreover, it provide an interface between the framework and power optimization method given by application developers or system designers.

Power aware job scheduling: In overprovisioned systems, power budget allocated for each node should be controlled by power-knobs such as dynamic voltage and frequency scaling (DVFS) or a power capping mechanisms. One of the challenges is to develop a runtime system for optimizing and controlling power budget allocated to each job as well as scheduling jobs according to available power budget of the entire system so that the total system throughput is maximized. Since effective power consumption differs application by application or phase by phase within an application, it is not easy to determine optimal power allocation. An easy way to allocate power budget for each job is to set power-cap statically so that total power consumption is almost within the system's power constraint. This static power capping approach is not optimal since runtime power of an application changes throughout the execution. Moreover, quality of services (QoS) demand is not constant for all the jobs. Therefore, a power-aware job scheduler with power manager which dynamically control power budget of running jobs according to their power-demand shown in Fig. 2 is necessary.

Power-performance simulation and analysis framework: Optimizing power-efficiency of an application relies on knowledge about its runtime characteristics. In extreme-scale applications, it is difficult to obtain such characteristics by profiling since users have few chances to execute a job with large number of nodes. Therefore, low-cost and highly flexible simulation environment in which the users can obtain performance and power behavior of their applications without actually execute them in large-systems. Moreover, we need to develop some tools which make it possible for users to well understand the static (or steady-state) and dynamic (or transient) power behavior.

Standardized API for power monitoring and control: In power constrained systems, there are several actors that monitors and controls power consumption at various levels. To deal with power monitoring and controlling seamlessly in various systems, a standardized API is indispensable. One such effort is High Performance Computing Power Application Programming Interface Specification developed by Sandia National Laboratories [2]. In order to make the common API useful, community level efforts for maintaining standardized power API are necessary.

[1] http://www.hal.ipc.i.u-tokyo.ac.jp/research/pompp/
[2] http://powerapi.sandia.gov/