Many-core systems and their challenges for operating systems

Klaus Danne
Microprocessor and Programming Research, Intel Labs

Technology scaling continuous and the number of transistors that can be placed on a chip doubles about every two years. This enabled dramatic processor performance gains by increasing clock frequency and instruction level parallelism over the last two decades. Recently, thermal design power constraints have begun to limit the rate at which processor frequency can be increased and industry shifted towards multi-core processors that deliver high performance gains by employing task and thread level parallelism.

This talk will discuss the performance and energy efficiency of a heterogeneous many-core approach, which employs a mix of few big cores, many small cores and special purpose cores. It than focuses on two potential challenges for the operating system to efficiently employ such devices.

First, the task scheduling and load balancing for such devices becomes challenging. The scheduler should account for the heterogeneity, such as the different supported instruction sets, performance of the cores and cache hierarchy characteristics. For instance, it could schedule the performance critical sequential parts of applications on the big cores, while assigning the parallel tasks to smaller cores.

Second, the OS should play a bigger role in power management. It is likely that in the future thermal design power limitations will not allow us employing all cores at full speed at the same time. While quick responsive control will be done at the hardware level, more sophisticated management should be done at the OS level. For example, performance uncritical tasks (or tasks with memory-bounded performance) could be executed at lower frequencies but with higher energy efficiency. On the other hand, tasks that block others could be speed up.

To deal with those challenges, the OS should be aware of hardware characteristics concerning the cores, caches and interconnect as well as the dynamic and static power consumption. Further, it should observe task behavior at runtime, estimate and measure power at fine granularity, and control finer power states at core granularity via new interfaces.

References

[1] Jim Held, Jerry Bautista, and Sean Koehl. From a few cores to many: A tera-scale computing research overview. *Intel White Paper*, 2006.

[2] Shekhar Borkar. Thousand core chips: a technology perspective. In *DAC '07: Proceedings of the 44th annual Design Automation Conference*, pages 746–749, New York, NY, USA, 2007. ACM.

[3] V. Siddha, S.and Pallipadi and A Mallick. Process scheduling challenges in the era of multi-core processors. *Intel Technology Journal*, 2007.

[4] Michael D. Powell, Arijit Biswas, Joel S. Emer, Shubhendu S. Mukherjee, Basit R. Sheikh, and Shrirang M. Yardi. Camp: A technique to estimate per-structure power at run-time using a few simple parameters. In *HPCA*, pages 289–300, 2009.

[5] Rob Knauerhase, Paul Brett, Barbara Hohlt, Tong Li, and Scott Hahn. Using os observations to improve performance in multicore systems. *IEEE Micro*, 28(3):54–66, 2008.

[6] Krishna K. Rangan, Gu-Yeon Wei, and David Brooks. Thread motion: fine-grained power management for multi-core systems. *SIGARCH Comput. Archit. News*, 37(3):302–313, 2009.

[7] Tong Li, Dan Baumberger, David A. Koufaty, and Scott Hahn. Efficient operating system scheduling for performance-asymmetric multi-core architectures. In *SC '07: Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, pages 1–11, New York, NY, USA, 2007. ACM.