Intl. Symposium on Principles and Practice of Parallel Programming, PPoPP 2018
February 24 – 28, 2018 Viena, Austria

Full Day Tutorial

An Introduction to Intel® Threading Building Blocks (Intel® TBB) and its Support for Heterogeneous Programming

Rafael Asenjo, Jim Cownie, Aleksei Fedotov

Abstract:
Due to energy constraints, high performance computing platforms are becoming increasingly heterogeneous, achieving greater performance per watt through the use of hardware that is tuned to specific computational kernels or application domains. It can be challenging for developers to match computations to accelerators, choose models for targeting those accelerators, and then coordinate the use of those accelerators in the context of their larger applications. This tutorial starts with a survey of heterogeneous architectures and programming models, and discusses how to determine if a computation is suitable for a particular accelerator. Next, Intel® Threading Building Blocks (Intel® TBB), a widely used, portable C++ template library for parallel programming is introduced. TBB is available as both a commercial product and as a permissively licensed open-source project at http://www.threadingbuildingblocks.org. The library provides generic parallel algorithms, concurrent containers, a work-stealing task scheduler, a data flow programming abstraction, low-level primitives for synchronization and thread local storage and a scalable memory allocator. The generic algorithms in TBB capture many of the common design patterns used in parallel programming. While TBB was first introduced in 2006 as a shared-memory parallel programming library, it has recently been extended to support heterogeneous programming. These new extensions allow developers more easily to coordinate the use of accelerators such as integrated and discrete GPUs, attached devices such as Intel® Xeon Phi co-processors, and FPGAs into their parallel C++ applications. This tutorial will introduce students to the TBB library and provide a hands-on opportunity to use some of its features for shared-memory programming. The students will then be given an overview of the new features included in the library for heterogeneous programming and have a hands-on opportunity to convert an example they developed for shared-memory into one that performs hybrid execution on both the CPU and an accelerator. Finally, students will be provided with an overview of the TBB Flow Graph Analyzer tool and shown how it can be used to understand application inefficiencies related to utilization of system resources.