Schedule of IISWC Tutorials and Workshops

Morning Session, September 29 (Saturday)

Tutorial I: STM in Managed Runtimes

The influential article "The Landscape of Parallel Computing Research: A View From Berkeley" suggests the only path toward significantly faster CPUs is chip multiprocessing. Programmers will simply have to adapt by writing concurrent code, regardless of any consequential problems with threads. An important trend in application development is that new applications are being developed based on a Managed Runtime Environment (MRTE) such as Java or .NET. These languages provide features such as garbage collection and dynamic class loading that require a managed runtime system (i.e. the Java Virtual Machine (JVM) for Java and .NET for C#).

Software transactional memory (STM) promises to alleviate the difficulty of programming using conventional mechanisms such as locks in a threaded environment. In this tutorial we will describe the various mechanisms that are available for software developers to implement transactional applications in Managed Runtimes.

We will then use a case study to delve into the workload characteristics when run under a Software Transactional memory system. The case study will be based on workloads and a runtime developed using the Apache Harmony Environment and the High Performance McRT (Multicore Runtime) STM. Apache Harmony (http://harmony.apache.org) is the open source Java SE implementation done under Apache License v2. McRT STM is a high performance runtime environment. The workloads we will characterize and demonstrate include simple micros as well as large multithreaded Java applications.

Speakers:

Suresh Srinivas is a Principal Engineer in the Software Solutions Group within Intel Corporation where he is currently focused on new technology development using open source technologies. He has 15 years of experience developing language runtimes, tools, and compilers. During his career he has published over 15+ Peer Reviewed Articles and has 7 Patents pending. Prior to Intel he was at SGI leading their JVM efforts. He obtained his PhD in Computer Science from Indiana University specializing in tools and runtimes for parallel languages. While not working you can find him volunteering, cooking or hiking in the beautiful Pacific Northwest.

Vijay Menon is a senior research scientist in the Programming Systems Lab at Intel investigating new programming technologies for multi-core systems. His primary areas of a research include compilers, managed runtimes, and transactional memory. Vijay holds a Ph.D. in Computer Science from Cornell University and a B.S. in Electrical Engineering and Computer Science from the University of California at Berkeley.

Afternoon Session, September 29 (Saturday)

Tutorial II: Using the Pin Instrumentation Framework for Workload Characterization

With several new emerging application domains, understanding the requirements of these applications is essential in designing future high performance processors. Such workload characterization and exploratory studies require fast and efficient techniques that can determine the behavior of these emerging workloads. This workshop will illustrate the use of Pin to conduct workload characterization and performance studies.

Pin is a dynamic instrumentation system provided by Intel (http://rogue.colorado.edu/Pin) that has become widely used throughout academia and industry. Pin allows code (C/C++) to be injected at arbitrary places in an executable while it is running. The injected code is used to observe the behavior of the program, and can be used to write a variety of workload characterization tools such as application profilers and trace generators. Pin provides a rich API that abstracts away the underlying instruction set idiosyncrasies and allows context information such as register contents to be passed to the injected code as parameters. Pin automatically saves and restores the registers that are overwritten by the injected code so the application continues to operate normally. Pin makes it easy to do studies on complex real-life applications, which makes it a useful tool for enabling workload characterization studies.

This tutorial targets researchers, students, and educators alike, from the novice Pin user to the expert Pinhead. The tutorial will provide a brief background on Pin and describe how to build simple Pin tools that can help in workload characterization.

Organizers:

Aamer Jaleel is a Hardware Engineer at Intel. He received his Ph.D. in Computer Engineering from the University of Maryland, College Park in 2005. His current research focuses on workload characterization and memory system optimizations for high performance processors.

Harish Patil is a Staff Engineer at Intel, where he works on the Pin project. His current interests include workload characterization and simulation of parallel programs. He received his Ph.D. in Compilers and Programming Languages from University of Wisconsin, Madison in 1996. He then worked with Hewlett Packard's Massachusetts Language Lab for two years mainly doing research in debugging of optimized code. He later joined Compaq where he worked on Spike, a simulation framework [ASIM], static branch prediction, and SPEC performance.