SpeedCode: Software performance engineering education via Coding of didactic exercises

Abstract

Software performance engineering is challenging both to do and to teach. This tutorial will present key challenges in software performance engineering and showcase SpeedCode, a prototype online programming platform for learning how to write fast code. SpeedCode allows users to develop fast multicore solutions to short programming problems and evaluate the performance and scalability of their solutions in a quiesced cloud environment, with no local installation required. SpeedCode supports parallel programming using OpenCilk, a task-parallel computing platform that is open-source and easy to program, teach, and use for research. This tutorial is suitable for anyone with experience programming in C or C++. Participants are encouraged to bring ideas for programming exercises in software performance engineering for the final parts of the tutorial, which will include a discussion on future directions for the SpeedCode platform.

Outline

- **Introduction:** Teaching software performance engineering: key concepts and challenges. **(15 min)**
- **SpeedCode walkthrough**: Overview and live demonstration of SpeedCode. This demonstration will walk participants through an example SpeedCode problem, in which a user starts from a correct, serial implementation of a small program and is tasked with making it run faster on a modern multicore computer. Participants are encouraged to follow along on their laptops to become acquainted with SpeedCode and how to use its tools and facilities to tackle performance-engineering problems. Participants can also try out a selection of other SpeedCode problems on the platform. **(60 min)**
- **Discussion about future plans for SpeedCode**: Structured discussion about future directions for SpeedCode, such as new problems; additional performance-engineering tools; new system capabilities; and community contributions. **(40 min)**

Prerequisites

- Some experience with C/C++ programming.
- A laptop that can connect to the internet.

Swag

The first 10 attendees will receive a free T-shirt!

Bios

The instructors for this tutorial share a common research agenda: to make parallel programming accessible so that every programmer, especially the non-experts, can write fast code for commodity multicore hardware in the post-Moore era.

- Tao B. Schardl is Chief Architect of OpenCilk and a research scientist in the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL).
- Tim Kaler is Lead Developer of SpeedCode and a postdoctoral associate in MIT CSAIL.
- I-Ting Angelina Lee is Runtime Architect of OpenCilk and an associate professor in the Computer Science and Engineering department in Washington University in St. Louis.
- Charles E. Leiserson is a budding software performance engineer and Professor of Computer Science and Engineering at MIT.