USC Viterbi School of Engineering
Ming Hsieh Department of Electrical Engineering

Seminar Announcement

Wonsun Ahn

Postdoctoral Research Scientist
University of Illinois at Urbana-Champaign

High-Performance JIT-Compiled Frameworks: Hardware/Compiler Co-Optimization

Thursday, April 17, 2014 at 10:30AM
Electrical Engineering Building (EEB 248)

Hosted by Prof. Michel Dubois

JIT-compiled frameworks are gaining increasing use for their cross-platform portability, performance portability, and runtime adaptability. In particular, scripting languages such as JavaScript, Python, and R are gaining wide acceptance. In these emerging frameworks, there is great opportunity for performance improvement through hardware/compiler co-optimization. In this talk, I present a few novel techniques that I have developed to improve performance.

First, I show how the compiler can use Hardware Transactional Memory (HTM) support to enforce high-performance Sequential Consistency (SC) for programmability and security. The idea is to wrap large sections of code inside a transaction, and then optimize the code inside each transaction without concern for memory-consistency-model restrictions. The optimizations speculate that any violation of the memory model will not be seen by other threads; otherwise, the transaction is aborted. Using this approach, the compiler can even outperform current compilers by a significant margin by allowing optimization across synchronization boundaries.

Next, I also show how the compiler can use the same HTM support to perform alias speculation. The approach consists of performing optimizations assuming the alias relationships that are true most of the time, and using the hardware to detect when such relationships are found not to hold through runtime checks. If the assumptions are correct, the code experiences good speedups; otherwise, the transaction is aborted.

Lastly, I show a compiler enhancement for JavaScript. A key feature of scripting languages that gives them their flexibility is dynamic typing. However, the absence of declared types makes it very challenging for the compiler to generate efficient code. Advanced compilers cope with it by introducing type systems of their own behind the scenes, and maintaining the type of each object at runtime as metadata. In this work, I focus on the Google Chrome V8 JavaScript compiler, and show that its type system is too brittle. While it works well for applications that display static behavior, it causes type specialization to fail in real website code. I go on to modify V8's type system to match the more dynamic behavior of real websites, and show significant savings in execution time, energy, and memory consumption.

Overall, these three approaches allow JIT compilers to achieve high performance while still maintaining programmability and security.

Bio:
Wonsun Ahn is a Postdoctoral Research Scientist at the University of Illinois at Urbana-Champaign. His research interests are parallel computer architecture and compilation systems. He is currently the co-PI of an NSF grant on improving the performance of scripting languages. He received a PhD in Computer Science from the same university in 2012. His PhD work was recognized by an IEEE Micro's Top Picks award publication. He has (co-)authored 12 journal and conference papers that have appeared in top compiler and architecture venues, and has two industry patents. He has served in the program and organizational committees of conferences, and is a member of the Samsung Frontier Membership.