Micro ToT Award 2016

Onur Mutlu

Micro-49

Taipei

October 18, 2016

The Winner

Iterative Modulo Scheduling: An Algorithm For Software Pipelining Loops

B. Ramakrishna Rau

Abstract

Modulo scheduling is a framework within which a wide variety of algorithms and heuristics may be defined for software pipelining innermost loops. This paper presents a practical algorithm, iterative modulo scheduling, that is capable of dealing with realistic machine models. This paper also characterizes the algorithm in terms of the quality of the generated schedules as well the computational expense incurred.

Published in Micro 1994

A Few Words on the Award Paper

- Courtesy of Scott Mahlke
- With input from Tom Conte and Wen-mei Hwu

Background: Software Pipeliening

- Back in the late 1980's and 1990's, software pipelining was one of the hottest topics at Micro with countless different approaches published each year.
- Many researchers focused on this problem due to the emergence of VLIW multicomputers like Multiflow and Cydra-5.
- Simple VLIW hardware was used to construct wide issue processors that were dependent on the compiler to produce efficient instruction schedules that could uncover enough instruction-level parallelism to maximize performance of the hardware.
- Software pipelining was particularly effective because it focused on program hotspots, namely loops, and it was capable of hiding long arithmetic and memory latencies that were common in these high performance designs.

IMS: Contribution

- Iterative modulo scheduling or IMS was the culmination of Bob's nearly 2 decades of work on what was originally called polycyclic scheduling, then software pipelining, which is the name that has stuck in the broader community, and finally modulo scheduling.
- IMS was the perfect combination of technical elegance and engineering excellence, which are the true hallmarks of Bob himself.
- IMS solves the complex problem of overlapping the execution of multiple iterations of a loop as simply generating the schedule for just a single iteration under a constraint that the schedule will repeat itself in a fixed number of cycles.
- This simplification brought sanity to the chaotic world of software pipelining that had traditionally always thought about multiple loop iterations.

IMS: Idea and Impact

- IMS with hardware support for predicated execution eliminated all the code expansion of traditional software pipelining.
- The pipeline is filled, drained, and executes arbitrary numbers of iterations with just a single copy of every instruction in the loop.
- The beauty of the approach is the mathematical formulation of starting with bounds on the maximum throughput the loop could achieve, and the backing away until a near-optimal solution is found.
- The code generation schema was extremely systematic, enabling rigorous implementation and testing that enabled it to be part of many commercial compilers.
- Such rigor is a rare find in the complex world of compiler backends and made IMS the defacto way to implement software pipelining.
- IMS was a central part of the original Cydra-5 compiler, one of the best strengths of the Itanium compilers built by Intel and HP, and is also standard in any modern VLIW DSP compiler from companies like TI and STMicro.

IMS: Final Words

- IMS stands the test of time because it is still in use 20 years after being published and will be for the foreseeable future.
- It translated software pipelining from a research concept to an engineering reality that could be implemented in production compilers.

- Bob passed away Dec 10, 2002, but his memory lives on in all of our hearts for his unparalleled technical excellence as well as being a true gentleman who treated even the most novice graduate students with kindness and respect.
- He was an integral part of the MICRO community for many years and someone who will never be forgotten.

Micro ToT Award 2016

Iterative Modulo Scheduling:
An Algorithm For Software Pipelining Loops

B. Ramakrishna Rau