Fast Cycle-Approximate Instruction Set Simulation

Björn Franke
Institute for Computing Systems Architecture
School of Informatics
University of Edinburgh
Outline

• Motivation & Background

• Basic Idea

• Linear Regression Analysis & Prediction

• Evaluation & Empirical Results

• Conclusion
Motivation

Cycle-Accurate Simulation: 8071s

Functional Simulation: 531s

15.2x

Equiv. Time on 206.4MHz Host: 104.8s
Simulator Landscape

Timing Accuracy

- RTL Simulation
- RTL-to-C Simulation
- Statistical Simulation
- Micro-Profiling
- Functional Simulation
- FPGA Prototyping

Speed
Simulator Landscape

Timing Accuracy

- RTL Simulation
- RTL-to-C Simulation
- "Conventional" Statistical Sim., Hybrid Analytical-Statistical Sim, SimPoint
- Statistical Simulation
- Micro-Profiling
- Functional Simulation
- FPGA Prototyping

Speed
Fast and Timing-Accurate Simulation

- ASIP Design Space Exploration
  - Cycle-Accurate Simulation Limited to Small Kernels
  - Video Codecs Already Beyond the Scope of the Possible
- Iterative Compilation - “Compiler in the Loop”
  - Possibly Thousands of Program Executions
  - Execution forms a Bottleneck
Basic Idea

Training Stage

- Benchmark Programs
- Cycle-Accurate Simulator
- Training Data

Deployment Stage

- New Program
- Functional Simulator
- Regression Solver
- Predictor

Cycle Count
Linear Regression Modelling

\[ y = \beta_0 + \sum_{i=1}^{N} \beta_i x_i + \epsilon \]

Matrix Form: \[ y = X\beta + \epsilon \]

Then, Minimise:

\[ S(\beta) = \sum_{i=1}^{m} \left( y_i - \beta_0 - \sum_{j=1}^{N} \beta_j x_{i,j} \right)^2 \]

To solve for estimates of regression coefficients \( \beta \).
Experimental Setup

- SimIt-ARM v2.1 Simulators
  - Cycle-Accurate & Functional ARM v5 ISA Simulators
  - 32bit, 5-stage pipeline, 16k instr. & 8k data caches, MMU, no FP
- 183 Applications from 6 Embedded Benchmark Suites
  - DSPstone, UTDSP, SWEET WCET, MediaBench, Pointer-Intensive Codes, Other (Cryptography, Software Defined Radio, Audio Processing)
- GCC 3.3.2 Compiler (-O3)
## Overview of Counters

<table>
<thead>
<tr>
<th>Counters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1 \ldots x_{30}$</td>
<td>Instruction counters for <code>mov</code>, <code>mvn</code>, <code>add</code>, <code>adc</code>, <code>sub</code>, <code>sbc</code>, <code>rsb</code>, <code>rsc</code>, <code>and</code>, <code>eor</code>, <code>orr</code>, <code>bic</code>, <code>cmp</code>, <code>cmn</code>, <code>tst</code>, <code>teq</code>, <code>mla</code>, <code>mul</code>, <code>smull</code>, <code>umull</code>, <code>ldr_imm</code>, <code>ldr_reg</code>, <code>str_imm</code>, <code>str_reg</code>, <code>ldm</code>, <code>stm</code>, <code>syscall</code>, <code>br</code>, <code>bl</code>, <code>fpe</code></td>
</tr>
<tr>
<td>$x_{31}, x_{32}$</td>
<td>Total instructions, nullified instructions</td>
</tr>
<tr>
<td>$x_{33}$</td>
<td>Total 4K memory pages allocated</td>
</tr>
<tr>
<td>$x_{34}, x_{35}$</td>
<td>Total I-Cache reads, read misses</td>
</tr>
<tr>
<td>$x_{36}, x_{37}$</td>
<td>Total I-TLB reads, read misses</td>
</tr>
<tr>
<td>$x_{38} \ldots x_{41}$</td>
<td>Total D-Cache writes, write misses, reads, read misses</td>
</tr>
<tr>
<td>$x_{42}, x_{43}$</td>
<td>Total D-TLB reads, read misses</td>
</tr>
<tr>
<td>$x_{44}$</td>
<td>Total BIU accesses</td>
</tr>
<tr>
<td>$x_{45}, x_{47}$</td>
<td>Total allocated OSMs, retired OSMs</td>
</tr>
<tr>
<td>$y$</td>
<td>Total cycles</td>
</tr>
</tbody>
</table>
Leave-One-Out Cross Validation

5.72% Mean Absolute Error for Instruction, Cache & TLB Counters
Error Distribution

![Error Distribution Chart]

- Error in %: 0-1, 2-3, 4-5, 6-7, 8-9, 10-11, 12-13, 14-15, 16-17, 18-19, 20-21, 22-23, 24-25
- Relative Frequency: 0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50

The chart above shows the distribution of errors in percentages. The Y-axis represents the relative frequency, while the X-axis represents the error range in percentages.
Subset Selection

<table>
<thead>
<tr>
<th>Counters/Parameters</th>
<th>Mn. Abs. Error</th>
<th>Std. Dev.</th>
<th>Max. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructions</td>
<td>38.9%</td>
<td>57.7</td>
<td>518%</td>
</tr>
<tr>
<td>Instructions &amp; Cache</td>
<td>10.5%</td>
<td>13.0</td>
<td>66.4%</td>
</tr>
<tr>
<td>Instruction, Cache &amp; TLB</td>
<td>5.72%</td>
<td>7.12</td>
<td>26.31%</td>
</tr>
<tr>
<td>All (incl. OSMs)</td>
<td>5.44%</td>
<td>7.37</td>
<td>44.66%</td>
</tr>
</tbody>
</table>
Prediction of 10 Largest Programs Based on $N=90,100,...,170$ Smallest Programs
Domain Specialisation

- Training on Regular, Embedded Benchmarks
- Test Against Irregular, General-Purpose Pointer-Intensive Applications
- 2.73% Average Error (0.17% to 7.1%)
Future Work

• Methodology

  • Confidence Intervals to Describe “Uncertainty” of Prediction

  • Subset Selection Algorithms for Feature Selection

  • Generalised Linear Regression Subject to Constraints

  • Continuous Learning in Hybrid Simulator

• Evaluation on Broader Range of Processors & Benchmarks

  • Including Extensible and VLIW Processors
Conclusions

- Enhance Functional Simulators with a Cycle-Approximate Timing Model
  - Learn mapping function from high-level event counters to cycle counters
- High Prediction Accuracy
  - Between 0.0-1.0% error for 50% of all programs
  - But, maximum error is still high (confidence intervals?)
- Simplicity & “Analysability”
  - No synthetic traces, immediately applicable to further analysis
Questions?