A Tool Suite for Simulation Based Analysis of Memory Access Behavior

ICCS 2004, 9 June 2004
Krakow, Poland

Josef Weidendorfer*, Markus Kowarschik**, Carsten Trinitis*

* Technische Universität München, Germany
** Universität Erlangen-Nürnberg, Germany
Overview

- Introduction
  Scope, Motivation
- Simulation Based Profiling
  Techniques, Results
- Visualization
  Requirements, Implementation
- Future Work
Introduction

- Scope of this Work: Project DiME
  - Cache-Optimized Library for Sequential PDE Solvers (Multigrid)
  - Best Parameters for Standard Techniques (Blocking/Padding) searched at Compile Time for given Platform
  - Benefit on Alpha around +100%, on Intel P4 +20% (far from Peak Perf.)
Introduction

- Scope of this Work: Project DiME
- Current Approaches
  - Try to understand Memory Access Behavior on recent HW (OOO, Speculation, HW Pf.)
  - Separate Effects (using Simulation):
    1. Simulation with Simple Cache Model
    2. + HW/SW Prefetching
    3. Real Hardware
  - New Techniques (e.g. aware of HW Pf.)
Introduction

• Scope of this Work: Project DiME
• Current Approaches
• Here:
  Presentation of Tools developed so far
   – Focus on Analysis of Sequential Code
     (Profiling, no Tracing)
   – General Usability
Simulation Based Profiling

• Trapping Memory Access Events
  – Via Runtime Instrumentation (Valgrind/Linux on x86)
  – No recompilation needed
  – „Instrument on demand“ (JIT like)
  – Switchable (at function enter / interactive)
    • Delivery Off (Slowdown factor 3 on HPC-Code)
    • Delivery On (Slowdown factor 30 – 100)
Simulation Based Profiling

- Trapping Memory Access Events
- Cache Simulator
  - Read/Write Accesses, Hits/Misses on L1/L2
  - Misses because of HW/SW Prefetching
Simulation Based Profiling

- Trapping Memory Access Events
- Cache Simulator
- Event Collection/Processing (on the fly)
  - Dynamic Context (Caller Chains) Call Graph
  - Cache Events attributed to Context, Thread
Simulation Based Profiling - Results

Amount of Profiling Data can get Huge!

<table>
<thead>
<tr>
<th>Command</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>5</th>
<th>10</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>bzip2 libm.so.6</td>
<td>408</td>
<td>850</td>
<td>1004</td>
<td>1329</td>
<td>1332</td>
<td>1132</td>
</tr>
<tr>
<td>(Compressor)</td>
<td>538</td>
<td>688</td>
<td>861</td>
<td>1113</td>
<td>1113</td>
<td>1113</td>
</tr>
<tr>
<td>cc1 ct_main-i.c</td>
<td>1519</td>
<td>5157</td>
<td>8352</td>
<td>22060</td>
<td>41164</td>
<td>44899</td>
</tr>
<tr>
<td>(C compiler)</td>
<td>6741</td>
<td>10881</td>
<td>15905</td>
<td>34282</td>
<td>52191</td>
<td>54722</td>
</tr>
<tr>
<td>konqueror</td>
<td>21500</td>
<td>55829</td>
<td>91713</td>
<td>251449</td>
<td>420871</td>
<td>507507</td>
</tr>
<tr>
<td>(KDE Browser)</td>
<td>51052</td>
<td>90629</td>
<td>147958</td>
<td>315838</td>
<td>470032</td>
<td>544487</td>
</tr>
</tbody>
</table>
Simulation Based Profiling - Results

Multigrid Code:

- „Standard“ Strategies work (Blocking on 2 Iterations: Half L2 Misses)
- Cache Events comparable to Reality

<table>
<thead>
<tr>
<th></th>
<th>Simulation</th>
<th>Real Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Instr. exec.</td>
<td>L2 Misses</td>
</tr>
<tr>
<td>Standard</td>
<td>11 879 M</td>
<td>751 421 K</td>
</tr>
<tr>
<td>Optimized</td>
<td>11 666 M</td>
<td>361 336 K</td>
</tr>
</tbody>
</table>
Visualization - Requirements

- GUI enabling fast Browsing
- Zoomability from Overview to Details
- Visualizations focusing on Performance Problems
- Language/Tool Independence
- Extensibility
  (Input Data/Visualization Types)
Visualization - KCachegrind

• „Fast“ Browsing: Currently C++/QT
• Zoomability: Library/Function/Line/Instruction
• Visualizations:
  Call Graph, Tree Map, Annotations
• Independence:
  – Use Debug Info + Standard Tools (Disass.)
  – Tailored to Simulation, but Converters exist for Data from Statistic Sampling (OPROfile, PFMon)
• Extensibility: Component Architecture
Future Work

• Simulation
  – Locality Metrics (temporal, spatial, reuse dist.)
  – Relation to Data Structures, Compression Techniques (e.g. Address differences)

• According Visualizations

Thanks for Listening!
Any Questions?