SimTrace: Capturing Over Time Program Phase Behavior

Steven Flolid – UT Austin
Emily Shriver – Intel Labs
Zachary Susskind – UT Austin
Benjamin Thorell – UT Austin
Lizy John – UT Austin
Proxy Motivation

• Pre-silicon design emulators are prohibitively slow

<table>
<thead>
<tr>
<th></th>
<th>Silicon</th>
<th>Emulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run Time (1x)</td>
<td>1 second</td>
<td>4.7 hours</td>
</tr>
<tr>
<td>Time to Complete (x)</td>
<td>1 minute</td>
<td>11.8 days</td>
</tr>
</tbody>
</table>

• Current techniques create proxies that capture key performance and power metrics

• It is beneficial to use the same workload throughout all stages of system design
Prior Art: Limitations & Research

Limitations

- CPU Centric
- Limited automation, requires hand-tuning
- Not modeling over-time behavior of metric (e.g. IPC, dynamic capacitance, cache miss)

Open Research Questions

- Capture over time behavior
- Increase automation
Problem with Average Proxies

- Existing techniques do not capture over time variation within a program
- When a system runs multiple proxies, shared resources may not be utilized correctly
- Power management algorithms use over time behavior
Overview of Simpoint

- Simpoint [1] breaks a workload into equal sized regions (100 Million Instructions)
- Regions are profiled based on micro-architecture independent Basic Block Vectors (BBV)
- Similar regions are clustered together based on the BBV using K-means
- A single region is simulated to represent each cluster

The SimTrace Technique

• Simpoint possesses an over time cluster trace but does not use it

• A single representative proxy could be created for each cluster

• Replaying the proxies in the cluster trace order results with a SimTrace

• This technique could serve as a baseline for future over time proxies
Measuring Over Time Similarity

- A Program’s average error is often used for accuracy measures

\[ \text{Average Error} = \left| \frac{x_{\text{experimental}} - x_{\text{reference}}}{x_{\text{reference}}} \right| \]

- Over time comparisons require more powerful techniques

- Various options in literature to measure time series similarity[1]
  - Point by point Mean Abs Error
  - Euclidean Distance
  - Correlation
  - Dynamic Time Warping
  - Kolmogorov-Smirnov Test
  - Many Others

Classification: https://arxiv.org/abs/1401.3973

We used metrics recommended by [1]
Simtrace Results

**Exchange2 Simtrace**
- MAPE = 1.6%

**MCF Simtrace**
- MAPE = 10.4%

**Leela Simtrace**
- MAPE = 1.2%

Graphs show dynamic instruction count (100m Instructions) for each application with performance metrics.
Simtrace Results (Cont..)

- Program’s with similar average IPC can have vastly different over time behavior
- Simtrace naturally removes complexity from a program’s performance
- Simtrace follows regular trends more accurately than irregular trends

XZ

MAPE=6.2%

GCC

MAPE=5.4%
## Similarity Results

<table>
<thead>
<tr>
<th>IPC</th>
<th>Trace Length</th>
<th>MAPE</th>
<th>Avg Err</th>
<th>Pear Cor</th>
<th>Euc Dist</th>
<th>DTW / simpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>leela 49,459</td>
<td>1.2%</td>
<td>-0.0015</td>
<td>0.87</td>
<td>3.65</td>
<td>0.01</td>
<td></td>
</tr>
<tr>
<td>exchange2 66,589</td>
<td>1.6%</td>
<td>-0.0006</td>
<td>-0.07</td>
<td>26.35</td>
<td>0.04</td>
<td></td>
</tr>
<tr>
<td>gcc 17,817</td>
<td>5.4%</td>
<td>-0.0005</td>
<td>0.88</td>
<td>28.99</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td>xz 45,718</td>
<td>6.2%</td>
<td>-0.0094</td>
<td>0.96</td>
<td>34.29</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>mcf 22,460</td>
<td>10.4%</td>
<td>0.0044</td>
<td>0.72</td>
<td>20.92</td>
<td>0.10</td>
<td></td>
</tr>
</tbody>
</table>

- SimTrace performs well for Point by Point error metrics (MAPE, Avg Err)
- Each technique captures some characteristics of over time performance

![Exchange2 Simtrace Comparison](image)

Smaller is better
Conclusions and Next Steps

Conclusions:

• Promising initial results, but more investigation is needed
• DTW and Euclidean are useful for comparison but are difficult to interpret without normalization

Next Steps:

• Create Simtraces for other benchmarks of SPEC CPU 2017
• Explore Simtrace’s ability to capture over time behavior of micro-architecture independent metrics (Imix, branches, footprint, etc)
• Normalizing Euclidian distance and Dynamic Time Warping