HOTL: A Higher Order Theory of Locality

Xiaoya Xiang, Chen Ding, Hao Luo
Department of Computer Science
University of Rochester

Bin Bao
Adobe Systems
Background

- Memory system of computer is organized as a hierarchy
- Locality metrics are used to manage and optimize the use of memory hierarchy
  - E.g. footprint, miss rate, reuse distance

*How these different locality metrics are related to each other?*
What is HOTL?

• **A set of formulas**: to convert between five locality metrics ("Filmer" metrics)
  – Footprint
  – Inter-miss time
  – Volume fill time
  – Miss ratio
  – Reuse distance

• **Reuse-window hypothesis**: the conditions for correctness
HOTL - Contributions

• Filmer metrics are mutually derivable
• Complements and extends the working set theory
• Using the new theory, the paper presents a technique of real-time locality measurement through sampling
  – Measured in real-time
  – Good accuracy (compared with simulation, hardware counters)
  – Much faster (compared with simulation)
• Cache Interference prediction
About Filmer metrics

• The metrics are defined on a sequential data access trace
• The time is logical and counted by the number of data accesses from the start of execution
• The cache is fully associative and uses the LRU replacement
• Miss ratio if time is logical, Miss rate if time is physical
Average Footprint

• A footprint is the amount of data accessed in a time window.

• Average footprint \( fp(l) \) is the average footprint size in all windows of length \( l \)

\[
fp(l) = \frac{\sum fp_w}{n - l + 1}
\]

• Example: “abbb”
  – 3 length-2 windows: “ab”, “bb”, “bb”
    • footprints: 2, 1, 1
  – \( fp(2) = \frac{(2+1+1)}{3} = \frac{4}{3} \)
Miss Ratio

• Footprint to Miss Ratio conversion

\[
mr(c) = mr(fp(x)) = \frac{fp(x + \Delta x) - fp(x)}{\Delta x}
\]

\(x\) and \(x+\Delta x\) are two consecutive window sizes, and cache size \(c \in [fp(x), fp(x + \Delta x)]\)
Miss Ratio (continued)

- Footprint to Miss Ratio conversion

\[ mr(c) = mr(fp(x)) = \frac{fp(x + \Delta x) - fp(x)}{\Delta x} \]
Reuse Distance

• The number of distinct data used between successive access to the same data

• Capacity miss ratio, \( mr(c) \), is the total fraction of reuse distance greater than cache size

\[
mr(c) = P(rd > c)
\]

• Probability function for reuse distance

\[
P(rd = c) = mr(c - 1) - mr(c)
\]
Example

- Consider the trace “xyzxyz...”
  - Assume it infinitely repeating

<table>
<thead>
<tr>
<th>t</th>
<th>fp(t)</th>
<th>c</th>
<th>mr(c)</th>
<th>P(rd = c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
The Higher Order Relations

<table>
<thead>
<tr>
<th>locality metrics</th>
<th>formal property</th>
<th>useful characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd order: footprint, volume fill time</td>
<td>concave/convex</td>
<td>linear-time, amenable to sampling, composable (dynamic locality)</td>
</tr>
<tr>
<td>2nd order: miss ratio, inter-miss time</td>
<td>monotone</td>
<td>machine specific, e.g. cache size/associativity (cache locality)</td>
</tr>
<tr>
<td>1st order: reuse distance</td>
<td>non-negative</td>
<td>decomposable by code units and data structures (program locality)</td>
</tr>
</tbody>
</table>

- To compute a lower order metric, HOTL conversion takes the difference of function of a higher order metric
- To compute a higher order metric, integrate the function of a lower order metric
Reuse-time conversion & Xiang formula

• Footprint conversion

\[ mr(c) = mr(fp(x)) = \frac{fp(x + \Delta x) - fp(x)}{\Delta x} \]

• Reuse-time conversion
  – Reuse window starts and ends with two access to the same data with not intervening reuses
  – Reuse time is the length of a reuse window
  – Cache size \( c = fp(l) \) (average footprint for all length-\( l \) reuse windows)

\[ mr_{rt}(fp(l)) = P(rt > l) = \sum_{t=l+1}^{\infty} P(rt = t) \]

• Xiang formula [PACT’11], simplified for \( n \gg w \)

\[ fp(w) \approx m - \sum_{t=w+1}^{n-1} (t - w)P(rt = t) \]
Footprint and reuse-time conversion equivalence

• **Theorem 2.2** For long execution ($n \gg w$), the footprint conversion and the reuse time conversion produce equivalent miss-ratio predictions

“But, the two predictions being the same does not mean that they are correct. They may be both wrong.”
Correctness Condition

• Corollary 2.3 (reuse-window hypothesis) The footprint-based conversions are accurate if the footprints in all reuse windows have the same distribution as the footprints in all windows, for every reuse-window length $l$
Sampling-based Locality Analysis

• Footprint can be analyzed through sampling
  – Set the system timer to interrupt at preset interval
  – Interrupt handler forks a sampling task and attaches the binary rewriting tool Pin to collect data access trace

\textbf{Require}: This handler is called whenever a program receives the timer interrupt

1: \texttt{pid} \leftarrow \texttt{fork}()
2: \texttt{if pid = 0 then}
3: \hspace{1em} \text{Attach the Pin tool and begin sampling until seeing c distinct memory accesses}
4: \hspace{1em} \text{Exit}
5: \hspace{1em} \texttt{else}
6: \hspace{2em} \texttt{Reset the timer to interrupt in k seconds}
7: \hspace{2em} \texttt{Return}
8: \hspace{1em} \texttt{end if}
Evaluation

• Experiment setup
  – 29 benchmarks from SPEC 2006
    • Trace length ranges from 20 billion to 2.1 trillion
    • Amount of data ranges from 3MB to 1.7GB
  – 8 benchmarks from PARSEC v2.1
  – Employ different approaches to measure miss-ratio: simulation, rd-prediction, fp-prediction, sampling

• Objective
  • Evaluate the accuracy and speed of miss-ratio prediction, made by footprint conversion and locality sampling
Results: Sequential Program

• Analysis speed
  – Average slowdown: simulation 38 times; reuse distance 153 times, footprint 23 times

<table>
<thead>
<tr>
<th>benchname</th>
<th>sim</th>
<th>rd</th>
<th>fp</th>
<th>samp</th>
<th>cov</th>
</tr>
</thead>
<tbody>
<tr>
<td>400.perlbench</td>
<td>49</td>
<td>219</td>
<td>34</td>
<td>0.24%</td>
<td>3.1%</td>
</tr>
<tr>
<td>401.bzip2</td>
<td>34</td>
<td>139</td>
<td>24</td>
<td>0.73%</td>
<td>1.5%</td>
</tr>
<tr>
<td>403.gcc</td>
<td>24</td>
<td>88</td>
<td>15</td>
<td>0.55%</td>
<td>0.1%</td>
</tr>
<tr>
<td>410.bwaves</td>
<td>57</td>
<td>196</td>
<td>35</td>
<td>2.14%</td>
<td>0.5%</td>
</tr>
<tr>
<td>416.gamess</td>
<td>62</td>
<td>286</td>
<td>40</td>
<td>0.29%</td>
<td>2.9%</td>
</tr>
<tr>
<td>429.mcf</td>
<td>10</td>
<td>56</td>
<td>6</td>
<td>0.14%</td>
<td>0.03%</td>
</tr>
<tr>
<td>433.milc</td>
<td>21</td>
<td>74</td>
<td>9</td>
<td>1.53%</td>
<td>0.04%</td>
</tr>
<tr>
<td>434.zeusmp</td>
<td>25</td>
<td>102</td>
<td>14</td>
<td>0.81%</td>
<td>0.06%</td>
</tr>
<tr>
<td>435.gromacs</td>
<td>40</td>
<td>142</td>
<td>19</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>436.cactusADM</td>
<td>40</td>
<td>167</td>
<td>21</td>
<td>0.00%</td>
<td>1.1%</td>
</tr>
<tr>
<td>437.leslie3d</td>
<td>42</td>
<td>131</td>
<td>23</td>
<td>0.00%</td>
<td>0.01%</td>
</tr>
<tr>
<td>444.namd</td>
<td>44</td>
<td>155</td>
<td>24</td>
<td>0.00%</td>
<td>2.2%</td>
</tr>
<tr>
<td>445.gobmk</td>
<td>35</td>
<td>130</td>
<td>23</td>
<td>0.22%</td>
<td>1.1%</td>
</tr>
<tr>
<td>447.dealII</td>
<td>54</td>
<td>209</td>
<td>34</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>450.soplex</td>
<td>13</td>
<td>52</td>
<td>7</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Results: Sequential Program

- Accuracy of the miss-ratio prediction
Results: Sequential Program

• Accuracy of the miss-ratio prediction
Results: Sequential Program

- Accuracy of the miss-ratio prediction
Results: Parallel Program

• Analysis speed
  – Did not profile reuse distance for PARSEC because it took too long
  – Average slowdown: footprint 113 times

• Accuracy of miss-ratio prediction
  – Across the 3 different input sizes, the predicted miss ratio matches closely with the simulated miss ratio
Results: Predicting Cache Interference

- Measurement
  - 20 SPEC 2006 benchmark programs
  - 190 different pair runs, total of 380 executions
  - ~9 days of CPU time
Summary of HOTL

• Filmer locality metrics
  – Miss rate in hardware
  – Reuse distance and footprint in a program

• Formulas for mutual conversion

• Correctness
  – Theoretical: reuse-window hypothesis
  – Empirical
Backup Slides
Volume Fill Time

• $vt(v)$: the time a program takes to access a given amount of data with volume as $v$
• Cache fill time: $vt(c)$, $c$ is cache size
• HOTL formula

$$vt(c) = \begin{cases} \frac{pp^{-1}(c)}{m}, & \text{if } 0 \leq c \leq m \\ \infty, & \text{if } c > m \end{cases}$$

Where $m$ is the total amount of program data
Inter-miss Time

- Time interval $vt(c+1) - vt(c)$ is the miss-free period when the program uses only data in cache

$$im(c) = \begin{cases} 
vt(c + 1) - v(t), & \text{if } 0 \leq c < m \\
n/m, & \text{if } c \geq m 
\end{cases}$$
Results: Parallel Program

• Accuracy of the miss-ratio prediction

(a) 8-way, 32KB cache
Results: Parallel Program

• Accuracy of the miss-ratio prediction
Results: Parallel Program

- Accuracy of the miss-ratio prediction