Precise Low-Overhead Performance Measurements

Lode Vandevenne and Jan Wassenberg
Highwayhash Profiler

- [https://github.com/google/highwayhash](https://github.com/google/highwayhash)
- profiler.h
- tsc_timer.h
Rapidly accelerating use of Deep Learning at Google

Number of directories containing model description files

<table>
<thead>
<tr>
<th>Year</th>
<th>2012</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>10</td>
<td>1500</td>
</tr>
</tbody>
</table>

Used across products:

Android
Apps
Gmail
Maps
Photos
Speech
Search
Translation
YouTube
and many others...
Highwayhash Profiler

- Software profiler for C++
- Flat or call-graph output (cumulative elapsed time per zone)
- Instrumentation based (not statistical like gprof or simulating like cachegrind)
- Manual: PROFILER_ZONE at each scope to measure, PROFILER_PRINT_RESULTS() at end of program
Why another?

- Realistic
- Fast
- High resolution
- Multi-core
- Measure arbitrary zones
Implementation

- Event-driven tracing
- Write combining
- Fences
- Time Stamp Counter
- Delta-coding string literals
Event-driven tracing
Write combining
Fences
Time Stamp Counter
Delta-coding string literals
Results
Event-driven tracing

- Zone entry and exit events
- Small fast event packets (64 bits)
- Call-graph computation deferred to the end
- Self-organizing list
Event-driven tracing

Capture events

Entry Event → Entry Event → Exit Event → Exit Event → ...

Analyze: Stack, Self-Organizing List

Zone Total → Zone Total → Zone Total → Zone Total → Zone Total

swap
Event-driven tracing
Write combining
Fences
Time Stamp Counter
Delta-coding string literals
Results
Write combining

- When writing events
- Avoid polluting the cache (only 64 bytes used)
- Write-Combine buffer (512-bit)
- Requires aligned pointers to cache line size
Event-driven tracing
Write combining
Time Stamp Counter
Fences
Delta-coding string literals
Results
Fences (Barriers)

- Avoid instruction reordering
Compiler Fence

- Compiler can reorder code
- Don't let measured code escape!
- Compiler specific keywords, e.g. gcc:
  
  ```asm volatile("": : : "memory")```
CPU can reorder instructions: out of order execution

Don't let measured code escape!

Avoid with serializing instruction

Load Fence: Serializes load operations

Faster and less variability than CPUID

Platform specific
SFENCE

- Store Fence: Serializes store operations
- To flush write-combine buffers: read correctly when printing results
Event-driven tracing
Write combining
Fences
Time Stamp Counter
Delta-coding string literals
Results
Time Stamp Counter

- Precise: fixed at nominal CPU frequency
- RDTSC, RDTSCP
- tsc_timer.h
- Per socket, not per core
### Time Stamp Counter

<table>
<thead>
<tr>
<th>Time Stamp Counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Instructions (outside)</td>
</tr>
<tr>
<td>LFENCE</td>
</tr>
<tr>
<td>RDTSC</td>
</tr>
<tr>
<td>LFENCE</td>
</tr>
<tr>
<td>Capture Event</td>
</tr>
<tr>
<td>Program Instructions To Measure</td>
</tr>
<tr>
<td>RDTSCP</td>
</tr>
<tr>
<td>LFENCE</td>
</tr>
<tr>
<td>Capture Event</td>
</tr>
<tr>
<td>Program Instructions (outside)</td>
</tr>
</tbody>
</table>

excluded self-overhead
Event-driven tracing
Write combining
Time Stamp Counter
Fences
Delta-coding string literals
Results
Delta-coding string literals

- Macros using `__func__` in zone names
- Need small fast key for zone
- Offset of address of string to a known string literal
- Assume all others are close to it in the binary
- Event: 25 bits offset, 39 bits timestamp

```
<table>
<thead>
<tr>
<th>data segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foo</td>
</tr>
<tr>
<td><em><strong>#</strong></em></td>
</tr>
<tr>
<td>Bar</td>
</tr>
</tbody>
</table>
```

unique origin

delta

delta
Event-driven tracing
Write combining
Time Stamp Counter
Fences
Delta-coding string literals

Results
## Results

**Mixed:**  
10000 x 996177 = 3831.453102 ms

**RunNoInline:**  
1 x 136544 = 0.052517 ms

**RunInline:**  
1 x 123894 = 0.047652 ms

<table>
<thead>
<tr>
<th>% cumulative</th>
<th>self</th>
<th>self</th>
<th>total</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>seconds</td>
<td>seconds</td>
<td>calls</td>
<td>s/call</td>
</tr>
<tr>
<td>55.06</td>
<td>2.10</td>
<td>2.10</td>
<td>5001</td>
<td>0.00</td>
</tr>
<tr>
<td>45.10</td>
<td>3.83</td>
<td>1.72</td>
<td>5000</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>3.83</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
</tr>
</tbody>
</table>
## Results - Variability

<table>
<thead>
<tr>
<th>% cumulative</th>
<th>time</th>
<th>cumulative</th>
<th>self</th>
<th>self</th>
<th>total</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>44.45</td>
<td>0.20</td>
<td>0.20</td>
<td>6874417</td>
<td>0.00</td>
<td>0.00</td>
<td>ZopfliFindLongestMatch</td>
</tr>
<tr>
<td>25.00</td>
<td>0.26</td>
<td>0.13</td>
<td>6874417</td>
<td>0.00</td>
<td>0.00</td>
<td>ZopfliFindLongestMatch</td>
</tr>
<tr>
<td>31.38</td>
<td>0.16</td>
<td>0.16</td>
<td>6874417</td>
<td>0.00</td>
<td>0.00</td>
<td>ZopfliFindLongestMatch</td>
</tr>
<tr>
<td>26.54</td>
<td>0.29</td>
<td>0.13</td>
<td>6874417</td>
<td>0.00</td>
<td>0.00</td>
<td>ZopfliFindLongestMatch</td>
</tr>
<tr>
<td>31.92</td>
<td>0.15</td>
<td>0.15</td>
<td>6874417</td>
<td>0.00</td>
<td>0.00</td>
<td>ZopfliFindLongestMatch</td>
</tr>
</tbody>
</table>
Summary

- Low-overhead, fast, precise, flexible profiler
- Achieved with several low-level techniques
Questions? Comments?

https://github.com/google/highwayhash

lode@google.com
janwas@google.com
## Child Overhead

<table>
<thead>
<tr>
<th>Program Instructions (outside)</th>
<th>before</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>RDTSC</td>
<td></td>
</tr>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>Capture Event</td>
<td></td>
</tr>
<tr>
<td>Program Instructions To Measure</td>
<td>child overhead</td>
</tr>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>RDTSC</td>
<td></td>
</tr>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>Capture Event</td>
<td>child and its self overhead</td>
</tr>
<tr>
<td>Program Instructions (child)</td>
<td>child overhead</td>
</tr>
<tr>
<td>RDTSCP</td>
<td></td>
</tr>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>Capture Event</td>
<td></td>
</tr>
<tr>
<td>Program Instructions To Measure</td>
<td>child overhead</td>
</tr>
<tr>
<td>RDTSCP</td>
<td></td>
</tr>
<tr>
<td>LFENCE</td>
<td></td>
</tr>
<tr>
<td>Capture Event</td>
<td>after</td>
</tr>
<tr>
<td>Program Instructions (outside)</td>
<td></td>
</tr>
</tbody>
</table>