How to Write Fast Numerical Code

Fall 2016

*Lecture:* Roofline model

**Instructor:** Torsten Hoefler & Markus Püschel

**TA:** Salvatore Di Girolamo

*ETH*

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich
Roofline model (Williams et al. 2008)

Resources in a processor that bound performance:
• peak performance [flops/cycle]
• memory bandwidth [bytes/cycle]
• <others>

Platform model

```
mem

Bandwidth β [bytes/cycle]

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
P_1  | P_p  |

Peak performance π [ops/cycle]
```

Algorithm model (n is the input size)

Operational intensity \( I(n) = \frac{W(n)}{Q(n)} = \frac{\text{number of flops (cost)}}{\text{number of bytes transferred between memory and cache}} \) [ops/bytes]

\( Q(n) \): assumes empty cache; best measured with performance counters

Notes
In general, Q and hence W/Q depend on the cache size \( m \) [bytes]. For some functions the optimal achievable W/Q is known:
- FFT/sorting: \( \Theta(\log(m)) \)
- Matrix multiplication: \( \Theta(\sqrt{m}) \)

Roofline model

Example: one core with \( \pi = 2 \) and \( \beta = 1 \) and no SSE ops are double precision flops

```
performance [ops/cycle]

bound based on β

\( \pi = 2 \)
\( β = 1 \)
\( \frac{1}{2} \)
\( \frac{1}{4} \)

bound based on π

some function run on some input

operational intensity [ops/bytes]
```

Bound based on \( β \)?
- assume program as operational intensity of \( x \) ops/byte
- it can get only \( β \) bytes/cycle
- hence: performance = \( y \leq βx \)
- in log scale: \( \log_2(y) \leq \log_2(β) + \log_2(x) \)
- line with slope 1; \( y = β \) for \( x = 1 \)

Variations
- vector instructions: peak bound goes up (e.g., 4 times for AVX)
- multiple cores: peak bound goes up (p times for p cores)
- program has uneven mix adds/mults: peak bound comes down (note: now this bound is program specific)
- accesses with little spatial locality: operational intensity decreases (because entire cache blocks are loaded)
Roofline Measurements

- Tool developed in our group
  
  (G. Ofenbeck, R. Steinmann, V. Caparros-Cabezas, D. Spampinato)

  http://www.spiral.net/software/roofline.html

- Example plots follow

- Get (non-asymptotic) bounds on I:
  
  - daxpy: \( y = \alpha x + y \)
  - dgemv: \( y = Ax + y \)
  - dgemm: \( C = AB + C \)
  - FFT
Roofline Measurements

Core i7 Sandy Bridge, 6 cores
Code: Intel MKL, sequential
Cold cache

What happens when we go to parallel code?
Roofline Measurements

Core i7 Sandy Bridge, 6 cores
Code: Intel MKL, parallel
Cold cache

What happens when we go to warm cache?
Roofline Measurements

Core i7 Sandy Bridge, 6 cores
Code: Intel MKL, **sequential**

Warm cache
Roofline Measurements

Core i7 Sandy Bridge, 6 cores
Code: Various MMM

Cold cache

MMM: Try to guess the basic shapes
Summary

- Roofline plots distinguish between memory and compute bound
- Can be used on paper
- Measurements difficult (performance counters) but doable
- Interesting insights: *use in your project!*
References

- Samuel Williams, Andrew Waterman, David Patterson
  *Roofline: an insightful visual performance model for multicore architectures*
  Communications ACM 55(6): 121-130 (2012)

- Georg Ofenbeck, Ruedi Steinmann, Victoria Caparros, Daniele G. Spampinato and Markus Püschel
  *Applying the Roofline Model*

- Victoria Caparros and Markus Püschel
  *Extending the Roofline Model: Bottleneck Analysis with Microarchitectural Constraints*