

Advanced Systems Lab

Spring 2025

Lecture: Roofline model

Instructor: Markus Püschel

TA: Tommaso Pegolotti, several more



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

1

Operational Intensity Again

Definition: Given a program P, assume cold (empty) cache

Operational intensity: $I(n) = \frac{W(n)}{Q(n)}$

#flops (input size n) → $W(n)$
#bytes transferred cache ↔ memory (for input size n) → $Q(n)$

Asymptotic bounds on $I(n)$

- *Vector sum:* $y = x + y$ $O(1)$
- *Matrix-vector product:* $y = Ax$ $O(1)$
- *Fast Fourier transform* $O(\log(n))$ $O(\log(\gamma))$ ← (not explained)
- *Matrix-matrix product:* $C = AB + C$ $O(n)$ $O(\sqrt{\gamma})$

Cache lecture
 γ = size LLC (last level cache)
Known to be optimal

2

2

Compute/Memory Bound

A function/piece of code is:

- **Compute bound** if it has high operational intensity
- **Memory bound** if it has low operational intensity

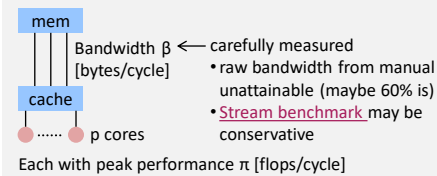
The roofline model makes this more precise

3

3

Roofline model/plot (Williams et al. 2008)

Platform model

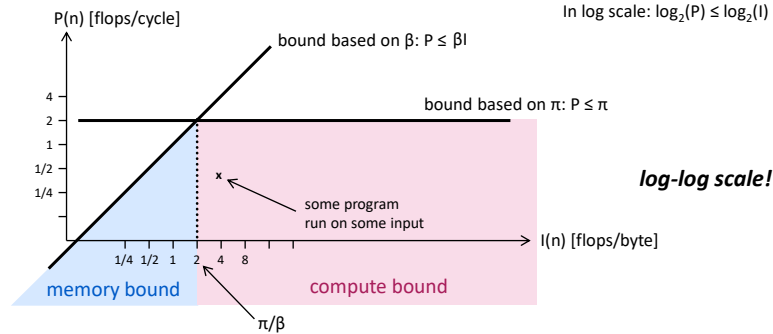


Algorithm/program model (n is the input size)

Work	$W(n)$ [flops]
Data movement: (mem \leftrightarrow cache)	$Q(n)$ [bytes]
Runtime:	$T(n)$ [cycles]
Derived:	
Operational intensity	$I(n) = W(n)/Q(n)$ [flops/byte]
Performance:	$P(n) = W(n)/T(n)$ [flops/cycle]

Example: one core, $\pi = 2$, $\beta = 1$, no SIMD

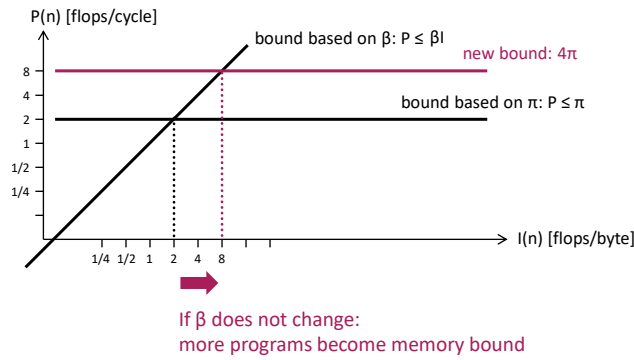
Bound based on β :
 $\beta \geq Q/T = (W/T)/(W/Q) = P/I$
 In log scale: $\log_2(P) \leq \log_2(I) + \log_2(\beta)$



4

Roofline Plots

What happens if we introduce 4-way SIMD?

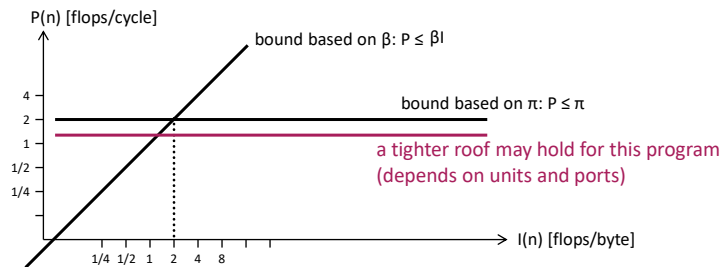


5

5

Roofline Plots

What if a program has an uneven mix of operations (e.g., 20% mults and 80% adds)?



6

6

Roofline Measurements

Tool developed in our group (code may need an update)
 (G. Ofenbeck, R. Steinmann, V. Caparras-Cabezas, D. Spampinato)
<http://www.spiral.net/software/roofline.html>

Example plots follow

Estimate operational intensity $I = W/Q$ (cold cache):

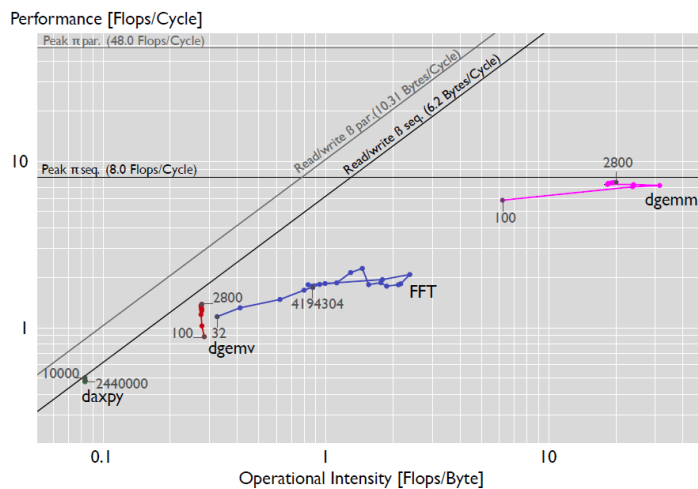
- *daxpy*: $y = \alpha x + y$ $W = 2n$ $Q = 3n$ doubles = 24n bytes $I = 1/12$
- *dgemv*: $y = Ax + y$ $W = 2n^2$ $Q \approx n^2$ doubles = $8n^2$ bytes $I \approx 1/4$
- *dgemm*: $C = AB + C$ $W = 2n^3$ $Q \geq 4n^2$ doubles = $32n^2$ bytes $I \leq n/16$
- *FFT*

Note:

- For *daxpy* and *dgemv*, Q is determined by compulsory misses.
- For *dgemm*, more misses than compulsory misses occur for larger sizes. If $3n^2 \leq \gamma$ (cache size), equality should hold above.

Roofline Measurements

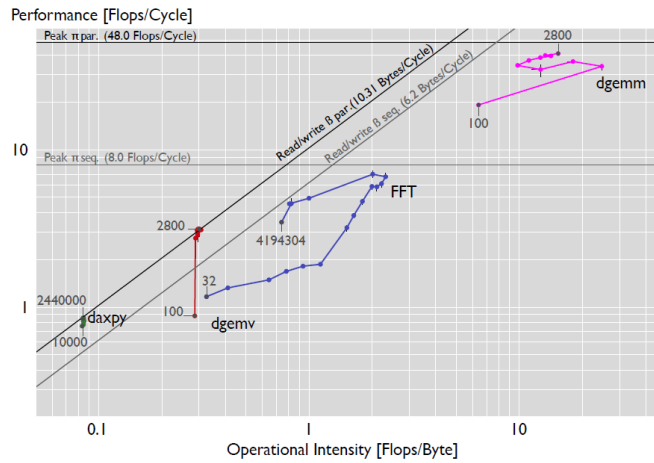
Core i7 Sandy Bridge, 6 cores
 Code: Intel MKL, *sequential*
 Cold cache



What happens when we go to parallel code?

Roofline Measurements

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



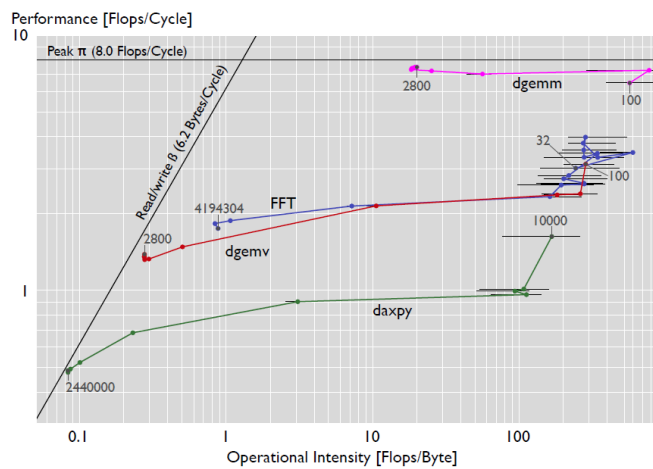
What happens when we measure with warm cache?

9

9

Roofline Measurements

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

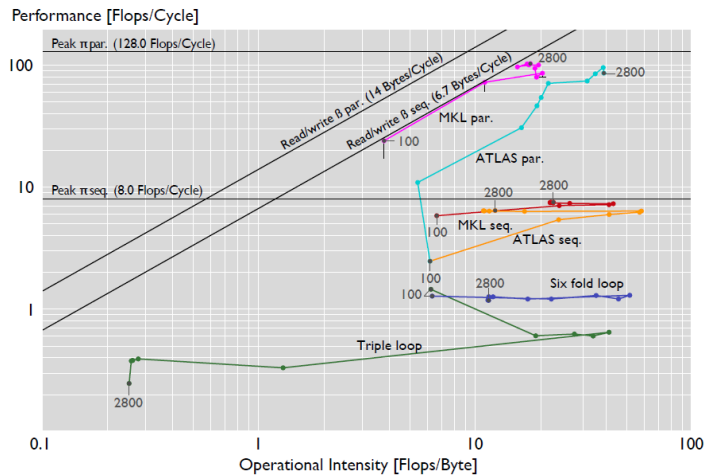


10

10

Roofline Measurements

Xeon E5 Sandy Bridge, 16 cores
Code: Various MMM
Cold cache



MMM: Different implementations

11

11

Cache-Aware Roofline Model (CARM)

Changes the definition of Q to count all loads from the memory hierarchy, i.e., from memory and all levels of caches.

Shows throughput bounds for all levels of caches

Intel Advisor:

- Uses CARM as standard
- Also provides the standard model, called MLR, but has to be used properly, e.g., you have to ensure cold cache and not include initialization
- Examples: next slides

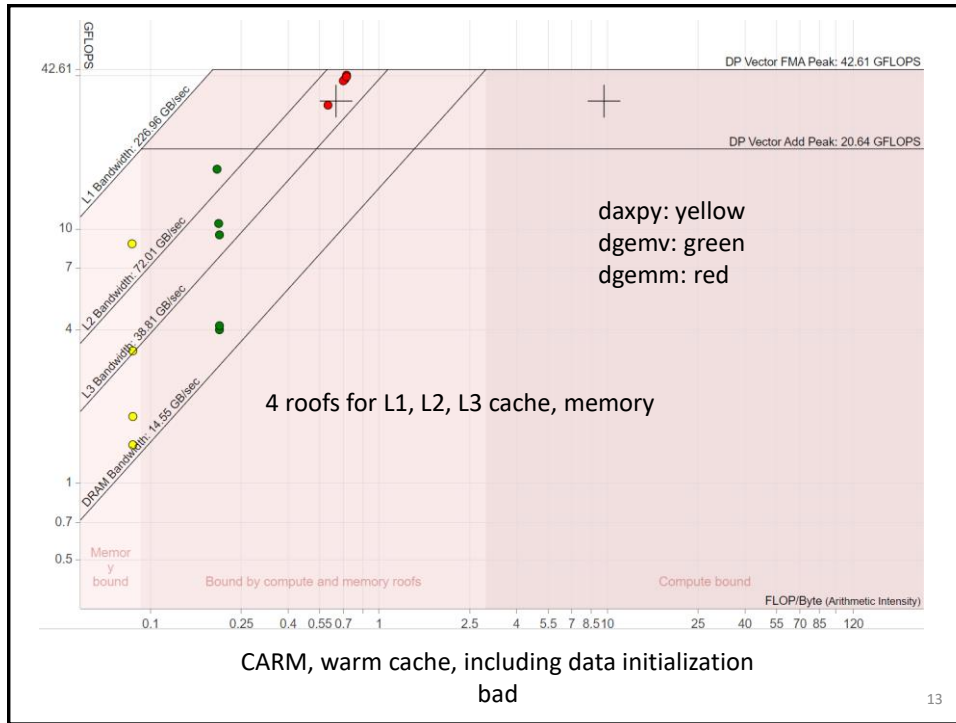
In your project (if roofline is used):

- Do measurements yourself, or
- Use Intel Advisor but choose MLR and use it properly to reflect the standard roofline model
- Explain in the report how you did the roofline plot/measurements

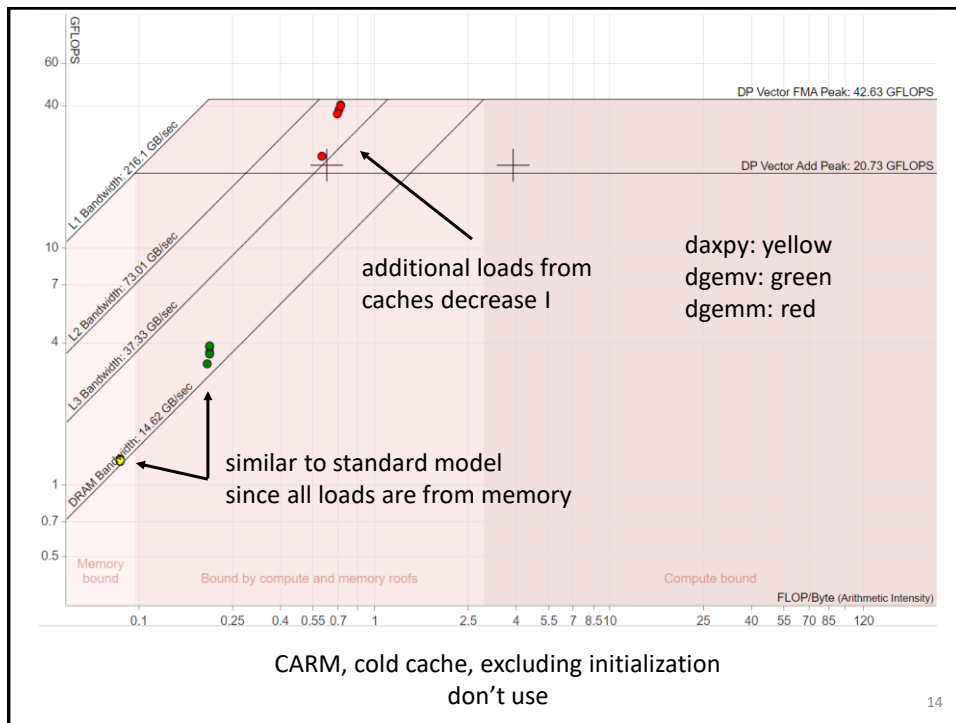
A. Ilic, L. Sousa, Cache-aware Roofline model: Upgrading the loft, IEEE Computer Architecture Letters, 13(1), pp. 21–24, 2014

12

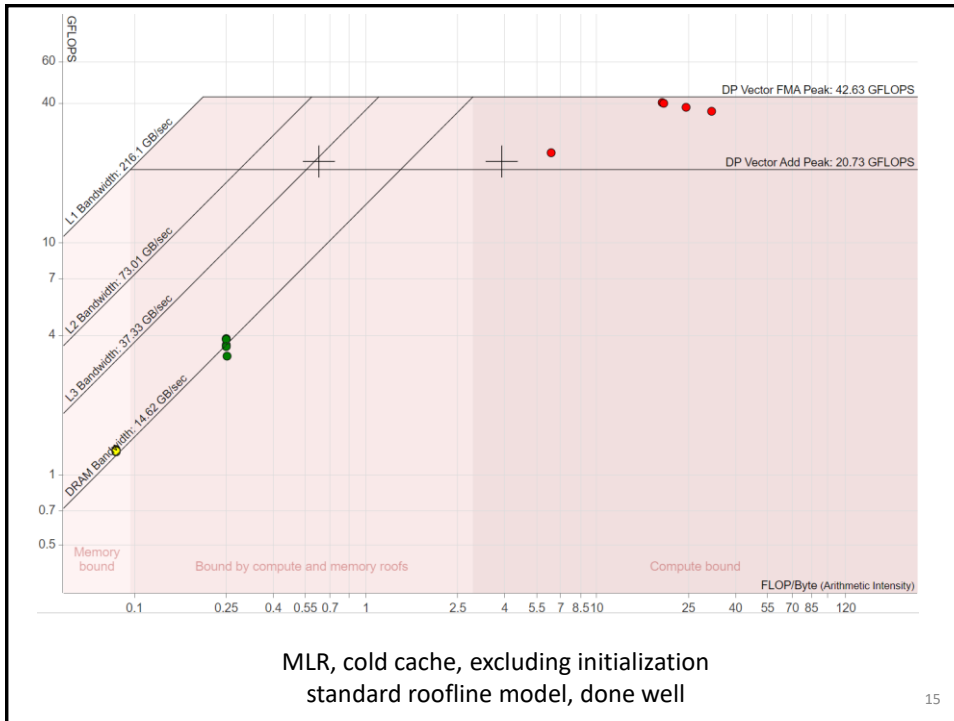
12



13

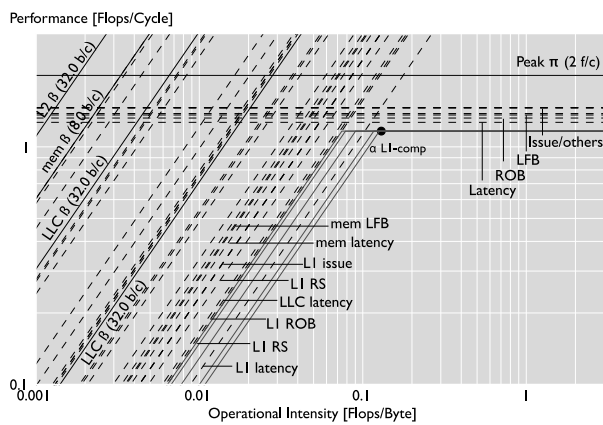


14



15

Generalized Roofline Model



Website and tool: <https://acl.inf.ethz.ch/research/ERM/>

Caparrós Cabezas and P. Extending the Roofline Model: Bottleneck Analysis with Microarchitectural Constraints, Proc. IISWC, pp. 222-231, 2014

16

Summary

Roofline plots distinguish between memory and compute bound at the memory/cache boundary

Can be instantiated for different scenarios (e.g., computation on GPU, data in CPU memory)

Can be used on for back-of-the-envelope computations on paper

Measurements difficult (performance counters) but doable

Easier variant just consider reads: $Q_{\text{read}}, \beta_{\text{read}}$

Interesting insights: *use in your project, but if so, do properly and explain how you did it!*

17

17