Extending the Roofline Model: Bottleneck Analysis with Microarchitectural Constraints

Victoria Caparrós Cabezas
Markus Püschel

Computer Science
ETH Zürich

Part of PhD work

Measuring Runtime
Measuring Runtime

Is this code fast?

Measuring Performance

Performance [Flops/Cycle]
Measuring Performance

Why does it not reach peak performance?

Roofline Plot

Operational Intensity [Flops/Byte]

Goal

Extended Roofline Model (ERM) to include additional hardware-related bottlenecks

Performance bottlenecks:
1. Latency of flops
2. Latency of L1 accesses

Current limitation: single core (with SIMD) only

Applications
- Code optimization
- Assess the impact of hardware upgrades

Example Challenge: Latency

```
int n = 1000;
x=(float*)malloc(...);
//Vector sum reduction
for(int i=0; i<n; i++)
y+=x[i];
```

Challenge: What if the latency only affects some of the operations?
Our solution:

Detailed breakdown of the computation time through a DAG-based performance model
Utilization-Based Bottleneck Modeling

Where are execution cycles spent so that peak performance is not reached?

Utilization-Based Bottleneck Modeling

break down underutilization by reason

$P = \frac{W}{T}$
Utilization-Based Bottleneck Modeling

$T_{issue} = \text{Number of cycles in which nodes are issued}$

$T_{issue} = 4$

Utilization-Based Bottleneck Modeling

$T_{issue} = \text{Number of cycles in which nodes are issued}$

$T_{lat} = \text{Number of cycles in which nodes are executed, but not issued}$

$T_{issue} = 4$

$T_{lat} = 8$
Utilization-based Bottleneck Modeling

Similar procedure for each level of the memory hierarchy

Finally, compute overlap (with compute) lines

Merging Roofline Plots for Memory Hierarchy

Original roofline model:

\[
I_x = \frac{W}{Q_x} \\
P_x \leq \min(I_x \beta_x, \pi)
\]

Redefinition of operational intensity:

\[
I = \frac{W}{Q} = \frac{W}{Q_{L1} + Q_{L2} + Q_{L3} + Q_{mem}} \\
I_x = I \frac{Q}{Q_x} \\
P_x \leq \min(I_x \beta_x, \pi) = \min(I \frac{Q}{Q_x} \beta_x, \pi)
\]
Some Results

Vector Sum Reduction (cold cache, n = 5 \times 10^6)

Performance [Flops/Cycle]

Latency/RS

Operational Intensity [Flops/Byte]

© Markus Püschel
Computer Science, ETH Zurich, 2018
FFT NR (warm cache)—Increasing Size

Size $2^{10}$

Size $2^{20}$

MMM Warm Cache: Triple Loop vs. Blocked ($n = 500$)

Triple loop

Blocked for L1 ($b = 50$)
Followup Work — Balancing Processor for Kernel

Hill climbing algorithm to find balanced processors (= good match SW/HW)

(a) Livermore loop 1 (small), initial configuration Sandy Bridge.
(b) Livermore loop 1 (small), initial configuration Cortex-A9.

**LL 1: Hydrodynamics computation fragment**

Effect on Roofline Plot (Example)

On Sandybridge

On a balanced processor

Rooflines are tighter
Conclusions

Code: DAG from LLVM interpreter
Processor: 30+ parameters

Basic Idea: Schedule and analyze DAG

Tool ERM:
https://github.com/caparro/llvm-performance

Paper:
P = 1.856 flops/cycle

\[ C = \frac{N}{D_{\text{eff}}} + m_{\text{prod}}(C_{\text{rea}} + C_{\text{rea}}) + \sum_i m_{\text{HL}} C_i \]

\[ P = \frac{W}{T} = T_{\text{issue}} + T_{\text{lat}} \]

\[ P(n) \leq \frac{W(n)}{D(n)} \]
Simple Example: Overlap

**Perfect overlap**

\[ T = \max(T_{\text{comp}}, T_{\text{mem}}) \]

\[ P = \min(\beta I, \pi) \]

**Overlap 0 \leq \alpha \leq 1**

\[ T = T_{\text{comp}} + T_{\text{mem}} - \alpha \min(T_{\text{comp}}, T_{\text{mem}}) \]

\[ P = \frac{I}{\alpha + \frac{1}{\beta}} - \alpha \min(\frac{1}{\alpha}, \frac{1}{\beta}) \]