Georgia Tech College of Computing
Computational Science and Engineering

A Roofline Model of Energy

Jee Whan Choi
Georgia Institute of Technology

Richard Vuduc
Georgia Institute of Technology

Introduction

- Energy-based analogue of the time-based roofline model.
- Directly connects properties of an algorithm with architectural time and energy costs.
- Contributions:
  - Energy cost model.
  - Energy "arch line."
  - Energy-balance.
  - Time-energy balance gap.
  - Work-communication trade-off.
  - Experimental results.
- Targeted at algorithm designers and performance tuners.
- More information:

  - J. Choi and R. Vuduc, "A Roofline Model of Energy," to appear in *Proceedings of the 27th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2013.
  - J. Choi and R. Vuduc, "A Roofline Model of Energy," Georgia Institute of Technology, GA, USA, Tech. Rep. GT-CSE-2012-01, 2012.
  - Daniel Bedard, Min Yeol Lim, Robert Fowler, and Allan Porterfield. PowerMon 2: Fine-grained integrated measurement. Technical report, RENaissance Computing Institute, University of North Carolina, Chapel Hill, NC, USA, 2009.

Energy Cost Model

- **Time**

$$T = \max\left(W\tau_{flop}, Q\tau_{mem}\right) = W\tau_{flop}\cdot\max\left(1, \frac{B_\tau}{I}\right)$$

- **Energy**

$$E = W\epsilon_{flop} + Q\epsilon_{mem} + \pi_0 T = W\epsilon_{flop}\cdot\left(1 + \frac{B_\epsilon}{I} + \frac{\pi_0}{\epsilon_{flop}}\frac{T}{W}\right) = W\cdot\epsilon_{flop}\cdot\left(1+\frac{\hat{B}_\epsilon(I)}{I}\right)$$

$$\hat{B}_\epsilon(I) = \eta_{flop}B_\epsilon + (1-\eta_{flop})\max(0, B_\tau - I)$$

- **Power**

$$P = \frac{\pi_{flop}}{\eta_{flop}}\left[\frac{\min(I, B_\tau)}{B_\tau} + \frac{\hat{B}_\epsilon(I)}{\max(I, B_\tau)}\right]$$

Variable W	Description
Q	# of useful compute operations
I	# of main memory operations
τ_{flop}	Intensity, or W/Q (e.g., flops per byte)
τ_{mem}	Time per work (arithmetic) operation
B_τ	Time per memory operation τ_{mem}/τ_{flop}
ϵ_{flop}	Balance in time, or (e.g., flops per byte)
ϵ_{mem}	Energy per work (arithmetic) operation
B_ϵ	Energy per memory operation $\epsilon_{mem}/\epsilon_{flop}$
π_0	Balance in energy, or (e.g., flops per joule)
ϵ_0	Constant power $\pi_0\cdot\tau_{flop}$
$\hat{\epsilon}_{flop}$	Constant energy per flop, $\epsilon_{flop}+\epsilon_0$
η_{flop}	Minimum energy to execute one flop, $\epsilon_{flop}/(\epsilon_{flop}+\epsilon_0)$
π_{flop}	Constant-flop energy efficiency, $\epsilon_{flop}/\tau_{flop}$
	Baseline power per flop (excluding constant power),

Rooflines in Time, Energy, and Power

- **Experimental Results**
  - **Target Systems**

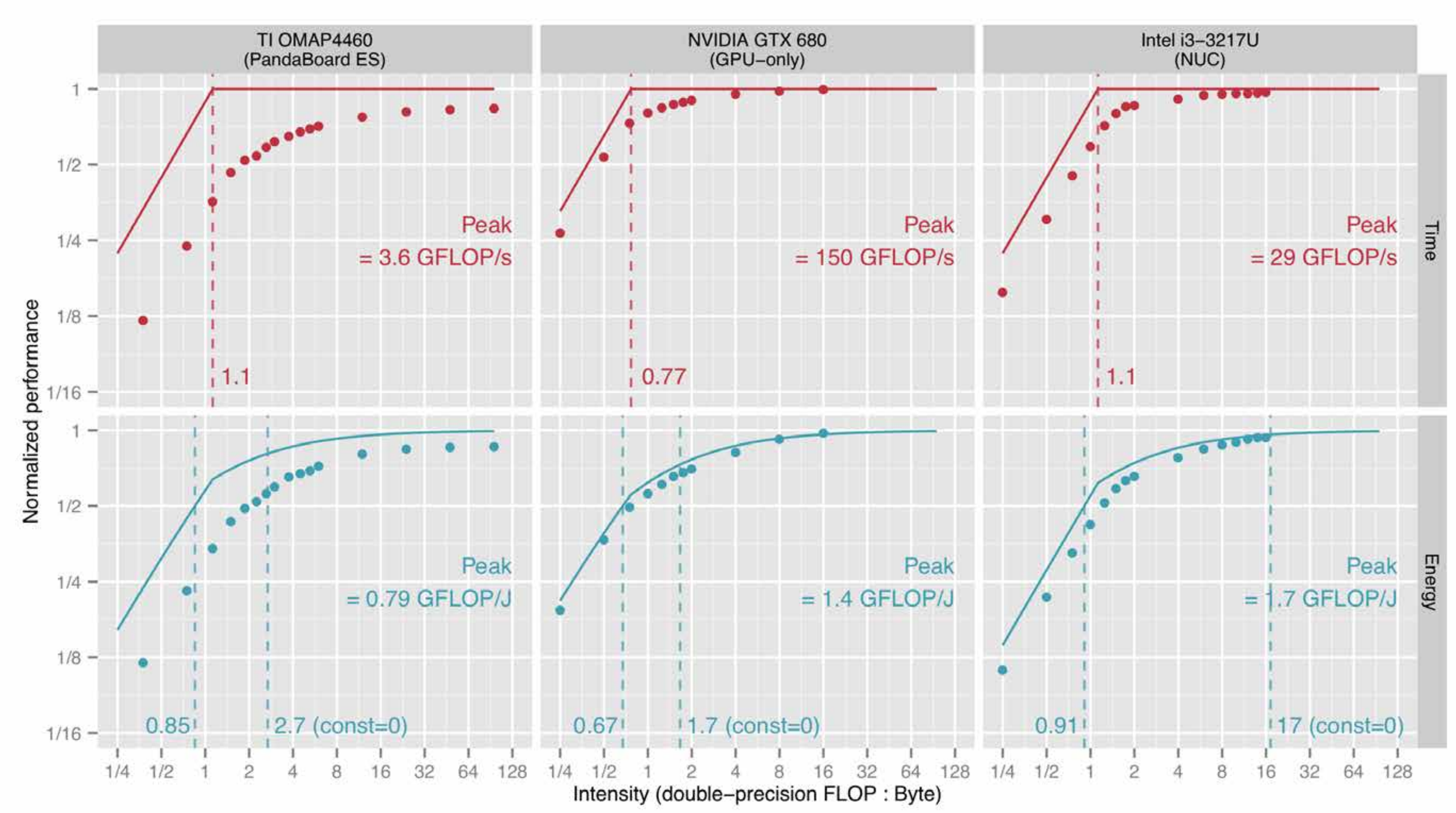
Architecture	Model	Peak performance Single (Double) GFLOP/s	Peak memory bandwidth GB/s	TDP Watts
Cortex A9	TI OMAP 4460	9.6 (3.6)	3.2	0.6
Kepler	GTX 680	3532.8 (147.2)	192.2	190[1]
Ivy Bridge	i3-3217U	57.7 (28.8)	25.6	17

[1]Board-level

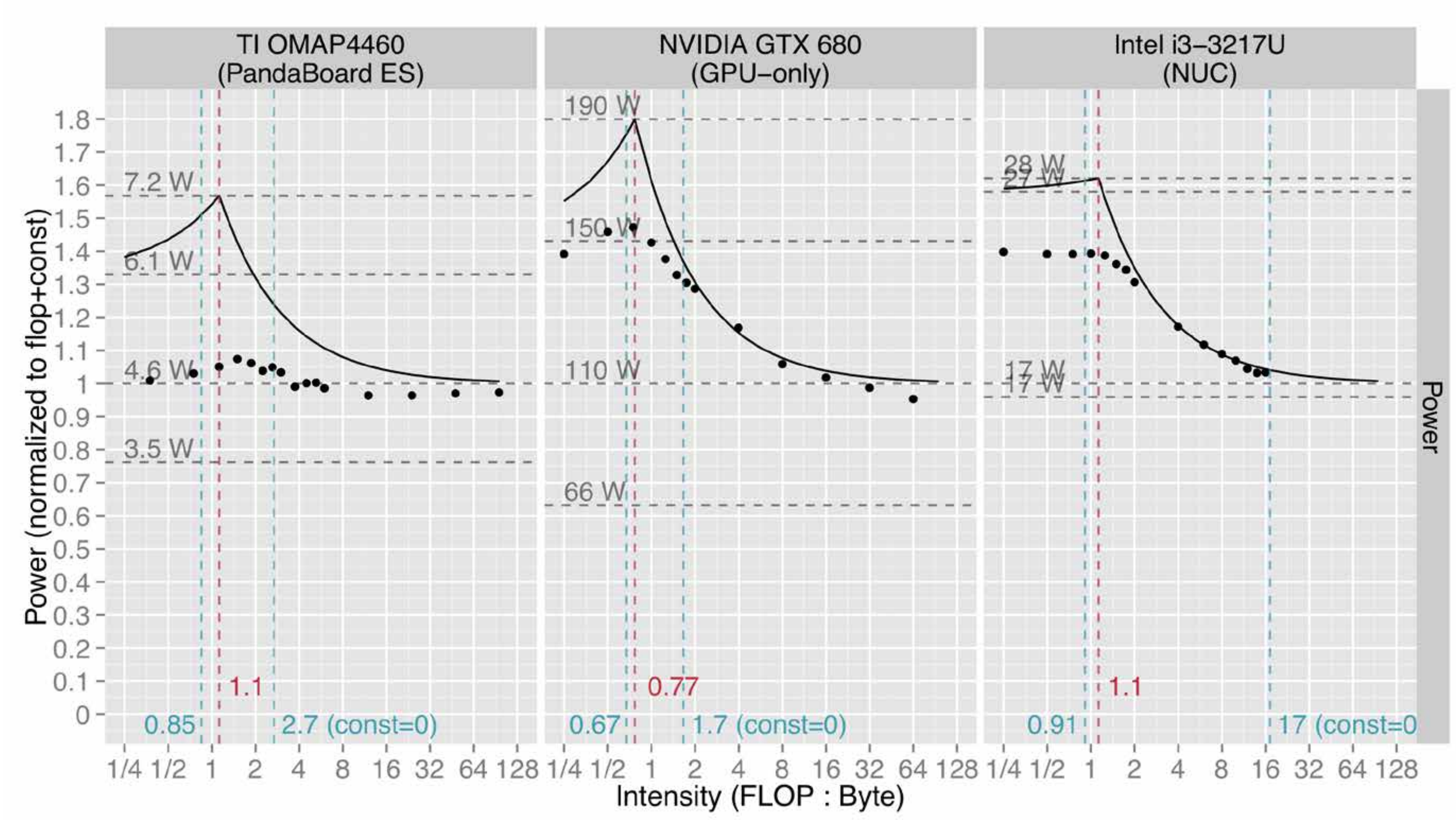
  - **Tools**
    - **PowerMon 2 intercepts and measures DC voltage and current**
    - **PICe Interposer intercepts signals coming from the motherboard's PCIe connector**
    - **Highly optimized μbenchmarks achieves high % of peak performance**
    - **Model instantiated using linear regression to find ϵ_{flop}, ϵ_{mem}, and π_0**

Architecture	Model	$\epsilon_{flop(single)}$	$\epsilon_{flop(double)}$	ϵ_{mem}	π_0
Cortex A9	TI OMAP 4460	16.85 pJ/FLOP	280.15 pJ/FLOP	814.7 pJ/Byte	3.5 W
Kepler	GTX 680	43.2 pJ/FLOP	262.9 pJ/FLOP	437.5 pJ/Byte	66.37 W
Ivy Bridge	i3-3217U	14.72 pJ/FLOP	24.297 pJ/FLOP	417.7 pJ/Byte	16.52 W

- **Rooflines in time and energy ("arch line")**



- **Power lines**



Work-Communication Trade-off

- Can we save energy by trading off work (FLOPs) for communication?
  - Assume a baseline algorithm that consumes energy of $E_{1,1}$
  - A "new" algorithm that reduces communication by a factor of m in exchange for increasing work by a factor of f consumes $E_{f,m}$
- We define a new energy-efficiency metric "*Greenup*" as

$$\Delta E = \frac{E_{1,1}}{E_{f,m}}$$

- We can now compare three cases in relating speedup ΔT and greenup ΔE

Case 1 **Baseline and new algorithm are memory-bound in time**

$$I < B_\tau$$
$$1 < \Delta T = m$$
$$\frac{1+\frac{I}{B_\epsilon}}{1+\frac{B_\tau}{B_\epsilon}} < \Delta E < \frac{1+\frac{B_\epsilon}{I}}{1+\frac{B_\epsilon}{B_\tau}}$$

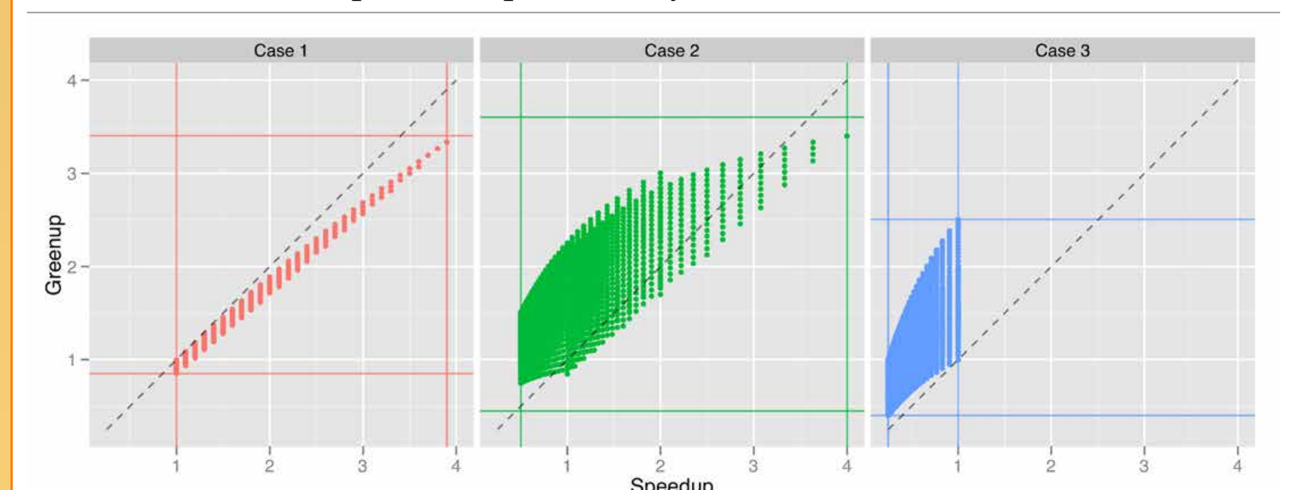
Case 2 **Baseline is memory-bound in time while the new algorithm is compute-bound**

$$I < B_\tau$$
$$\Delta T = \frac{1}{f}\frac{B_\tau}{I}$$
$$\Delta T\cdot\frac{1+\frac{I}{B_\epsilon}}{1+\frac{B_\tau}{B_\epsilon}} < \Delta E < m\cdot\frac{1+\frac{I}{B_\epsilon}}{1+\frac{B_\tau}{B_\epsilon}}$$

Case 3 **Baseline and new algorithm are compute-bound in time**

$$B_\tau < I$$
$$\frac{1}{f} = \Delta T < 1$$
$$\Delta T\cdot\frac{1+\frac{B_\epsilon}{I}}{1+\frac{1}{f}\frac{B_\epsilon}{I}} < \Delta E < \frac{1+\frac{B_\epsilon}{I}}{1+\frac{1}{m}\frac{B_\epsilon}{I}}$$

- Visualized
  - Each point corresponds to $(\Delta T, \Delta E)$ pair
  - Indicates that improving time while incurring a loss in energy-efficiency is unlikely
  - Converse is possible, particularly in case 2 and case 3.



- Future Work
  - Test our theory on real applications
  - Fast Multipole Method (FMM)
  - Convolution (FFT vs. stencil)
  - 2D vs. 2.5D vs. 3D matrix multiplication
  - Quantum Chromodynamics (QCD)