Lower Bounds on Algorithm Energy Consumption: Current Work and Future Directions

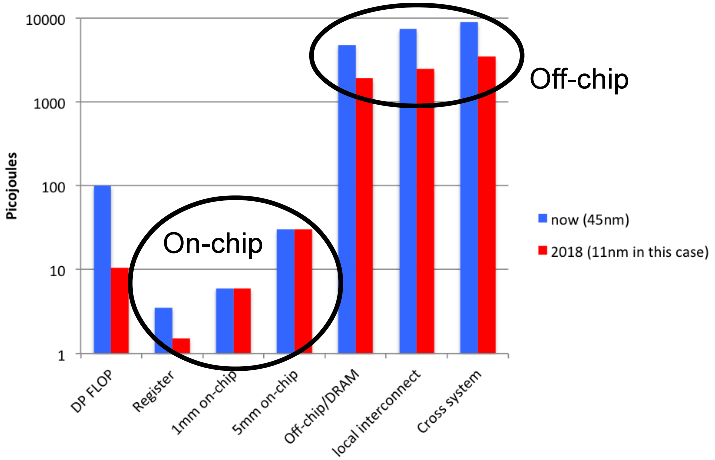
James Demmel, **Andrew Gearhart**, Benjamin Lipshitz and Oded Schwartz

Electrical Engineering and Computer Sciences
University of California, Berkeley

March 1, 2013

Energy is the current hot ticket item...

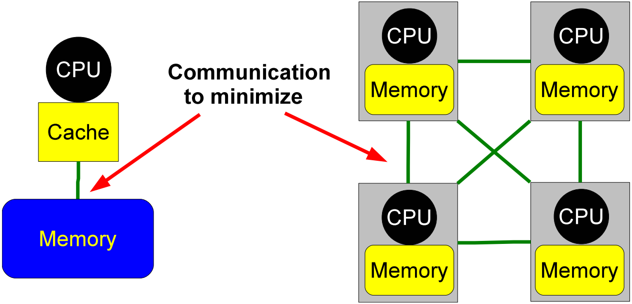
- Problem in both client and cloud
- UCB ASPIRE project
  - Algorithms and Specializers for Provably Optimal Implementations with Resilience and Efficiency
  - 5-year project with funding from DARPA and industry
  - **Primary Goal**: Energy efficiency in hardware and software!
- This work is an initial foray into the **Provably Optimal** portion of ASPIRE



- **Hypothesis: Reducing communication via communication-avoiding (CA) algorithms can reduce energy/task**

What do we mean by communication?

- Communication defined as the number of words and messages transferred
- Sequential and parallel distributed machine models



- These can be composed hierarchically (more later on this)

- Communication lower bounds for many linear algebra problems [BDHS11]

Sequential	$\Omega\left(\frac{\#\text{flops}}{M^{1/2}}\right)$
Parallel	$\Omega\left(\frac{\#\text{flops}}{pM^{1/2}}\right)$

where M is fast memory size and p is the number of processors.

- Bounds for messages moved (latency-cost) obtained by dividing by largest message size m $(m \leq M)$

- 2.5D matrix multiplication replicates input data c times to reduce communication in the distributed model (still $O(n^3)$ flops)

- Communication lower bounds ($M = \frac{cn^2}{p}, 1 \leq c \leq p^{1/3}$):

$$\text{\# Words} = \Omega\left(\frac{n^2}{(cp)^{1/2}}\right), \quad \text{\# Messages} = \Omega\left(\frac{p^{1/2}}{c^{3/2}}\right)$$

- 2.5D matrix multiply algorithm has a range of perfect strong scaling
  - i.e. increase # of procs by factor c (w/ problem size n constant)...and runtime decreases by c while energy is constant (details later)

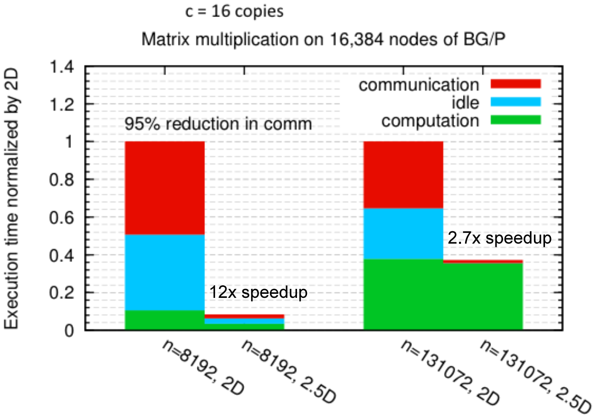
- 2.5D matrix multiplication replicates input data c times to reduce communication in the distributed model (still $O(n^3)$ flops)

- Communication lower bounds ($M = \frac{cn^2}{p}, 1 \leq c \leq p^{1/3}$):

$$\text{\# Words} = \Omega\left(\frac{n^2}{(cp)^{1/2}}\right), \quad \text{\# Messages} = \Omega\left(\frac{p^{1/2}}{c^{3/2}}\right)$$

- 2.5D matrix multiply algorithm has a range of perfect strong scaling
  - i.e. increase # of procs by factor c (w/ problem size n constant)...and runtime decreases by c while energy is constant (details later)

- 2.5D Matmul on BG/P, 16K nodes/64K cores (Distinguished Paper Award at EuroPar'11) [SD11]



- Can we now say something about the minimal amount of energy needed to compute a problem?
- Assume the distributed memory machine mentioned earlier
- Model runtime, then apply to a model of energy

- Model runtime T as

$$T = \gamma_t F + \beta_t W + \alpha_t S$$

- where
  - F = flops performed and γ_t = sec/flops
  - W = words transferred and β_t = sec/word
  - S = messages sent and α_t = sec/msg

- Model total energy E as

$$E = p(\gamma_e F + \beta_e W + \alpha_e S + \delta_e MT + \epsilon_e T)$$

- where for p processors and M words of mem/node
  - γ_e,β_e,α_e = joules/flop, joules/word, joules/msg
  - δ_e = joules/word/sec
  - ϵ_e = joules/sec
  - T = runtime

- The first 3 terms represent the energy directly required to perform flops and move data

Modeling runtime and energy

- Model total energy E as

$$E = p(\gamma_e F + \beta_e W + \alpha_e S + \delta_e MT + \epsilon_e T)$$

- where for p processors and M words of mem/node
  - γ_e,β_e,α_e = joules/flop, joules/word, joules/msg
  - δ_e = joules/word/sec
  - ϵ_e = joules/sec
  - T = runtime

- $\delta_e MT$ is the energy cost to store data in memory

Modeling runtime and energy

- Model total energy E as

$$E = p(\gamma_e F + \beta_e W + \alpha_e S + \delta_e MT + \epsilon_e T)$$

- where for p processors and M words of mem/node
  - γ_e,β_e,α_e = joules/flop, joules/word, joules/msg
  - δ_e = joules/word/sec
  - ϵ_e = joules/sec
  - T = runtime

- $\epsilon_e T$ is for other energy components
  - leakage
  - cooling fans (45W+ on some servers!)
  - memory idle power
  - other fixed energy costs

- Add a processor, use the additional memory
- Start with the minimal number of procs: $pM = 3n^2$
- Scale p (and total memory) by factor c ($c \leq p^{1/3}$)
- Recall:
  - γ_t,β_t,α_t = sec/flop, sec/word moved, sec/msg sent
  - γ_e,β_e,α_e = joules for same operations
  - δ_e = joules/word/sec
  - ϵ_e = joules/sec

$$T(cp) = \frac{n^3}{cp}\left(\gamma_t + \frac{\beta_t}{M^{1/2}} + \frac{\alpha_t}{mM^{1/2}}\right) = \frac{T(p)}{c}$$

$$E(cp) = cp\left[\frac{n^3}{cp}\left(\gamma_e + \frac{\beta_e}{M^{1/2}} + \frac{\alpha_e}{mM^{1/2}}\right) + \delta_e MT(cp) + \epsilon_e T(cp)\right]$$
$$= E(p)$$

Matmul perfect strong scaling-in Time and Energy (with .5D)

- This is what we mean by perfect strong scaling

$$T(cp) = \frac{T(p)}{c}$$

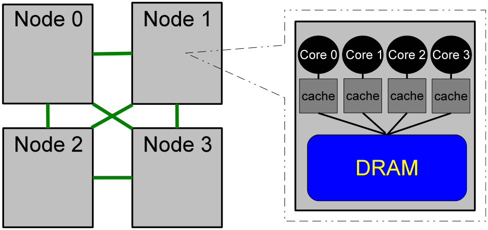
$$E(cp) = E(p)$$

- Not true for algorithms that don't replicate data (2D)...

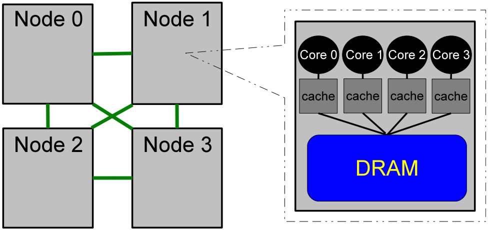
More energy lower bounds

- In the distributed model, energy lower bounds for:
  - classical $O(n^3)$ and Strassen matrix multiplication
  - LU factorization
  - Fast Fourier Transform (FFT)
  - direct n-body problem ($O(n^2)$ and with cutoff)
- Perfect energy strong scaling by using more memory via *.5D in
  - Bandwidth: Classical/Strassen Matmul, direct n-body, LU
  - Latency: Classical/Strassen Matmul, direct n-body

- Energy models are flexible...can be generated for more machines
- An example 2-level model:



- An example 2-level model: "Level 0" is internode, and "Level 1" is intranode



- Energy parameters have e superscript, subscripts show level

$$E_0 \geq p_0 \left[p_1 \left(\gamma_1^e \frac{n^3}{p_0 p_1} + \beta_1^e \frac{n^3}{p_0 p_1 M_1^{1/2}} + \alpha_1^e \frac{n^3}{p_0 p_1 M_1^{3/2}} + \delta_1^e M_1 T_1 + \epsilon_1^e T_1 \right) + \beta_0^e \frac{n^3}{p_0 M_0^{1/2}} + \alpha_0^e \frac{n^3}{p_0 M_0^{3/2}} + \delta_0^e M_0 T_0 + \epsilon_0^e T_0 \right]$$

Future work

- Accurate measurement and validation of models and parameters
  - Initial work involves C benchmarks for bandwidth, latency and tuned linear algebra code (Intel's MKL)
  - Measurement with wall power meters, on-chip firmware power meters
- Use energy bounds to aid hardware design-space exploration
  - HW/SW cotuning in ASPIRE
  - Tuned computational kernels + specialized hardware = energy efficiency

Future work

- Accurate measurement and validation of models and parameters
  - Initial work involves C benchmarks for bandwidth, latency and tuned linear algebra code (Intel's MKL)
  - Measurement with wall power meters, on-chip firmware power meters
- Use energy bounds to aid hardware design-space exploration
  - HW/SW cotuning in ASPIRE
  - Tuned computational kernels + specialized hardware = energy efficiency

Future work

- Use models to consider interesting problems:
  - Minimize energy to compute problem
  - Minimize energy w/ runtime bound
  - Minimize time w/ energy bound
  - Minimize avg. power w/ runtime bound
  - Given an algorithm and target efficiency (GFLOPS/W), can we determine a set of optimal architectural parameters?
  - Others?

The End. Questions?

We acknowledge funding from Microsoft (Award #024263) and Intel (Award #024894), and matching funding by U.C. Discovery (Award #DIG07-10227). Additional support comes from ParLab affiliates National Instruments, Nokia, NVIDIA, Oracle and Samsung, as well as MathWorks. Research is also supported by DOE grants DE-SC0004938, DE-SC0005136, DE- SC0003959, DE-SC0008700, and AC02-05CH11231, and DARPA grant HR0011-12-2-0016. Approved for public release; distribution is unlimited. The content of this presentation does not necessarily reflect the position or the policy of the US government and no official endorsement should be inferred.

References I

G. Ballard, J. Demmel, O. Holtz, and O. Schwartz.
Minimizing communication in numerical linear algebra.
SIAM J. Matrix Analysis Applications, 32(3):866–901, 2011.

James Demmel, Andrew Gearhart, Benjamin Lipshitz, and Oded Schwartz.
Perfect Strong Scaling Using No Additional Energy.
In *Proceedings of the 2013 IEEE 27th International Parallel and Distributed Processing Symposium*, IPDPS '13. IEEE Computer Society, 2013.

Edgar Solomonik and James Demmel.
Communication-optimal parallel 2.5d matrix multiplication and lu factorization algorithms.
In *Proceedings of the 17th international conference on Parallel processing - Volume Part II*, Euro-Par'11, pages 90–109, Berlin, Heidelberg, 2011. Springer-Verlag.

Backup Slides

Comm. Type	Now (45nm)	2018 (11nm)	Reference
DP Flop	100	10.6	Tensilica XPG @ 1Ghz
Register	3.5	1.5	Tensilica XPG
1mm on-chip	6	6	ORION-2 model
5mm on-chip	30	30	ORION-2 model
off-chip/DRAM	4800	1920	Micron Inc (JEDEC roadmap)
local interconnect	7500	2500	Finisar optical cable roadmap
cross system	9000	3500	Finisar optical cable roadmap

Table: Sources for Communication Energy Figure (John Shalf, LBNL)