International Conference on
Energy-Aware High Performance Computing

Are our Dense Linear Algebra Libraries Energy-Friendly?

Time-Power-Energy Trade-Offs in BLAS and LAPACK

Jose I. Aliaga[1], Maria Barreda[1], Manuel F. Dolz[2],
Rafael Mayo[1], Enrique S. Quintana-Ortí[1]

1 Universitat Jaume I

2 Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

September 1st – 2nd, 2014, Dresden (Germany)

Motivation

- High performance computing
  - Optimization of algorithms applied to solve scientific complex problems
- Technological advance ⇒ Performance improvement
  - Higher number of cores per socket (multicore processors)
  - Use of accelerators (GPUs) and coprocessors (e.g., Intel Xeon Phi)
- High performance computing data centers ⇒ High energy consumption!
  - Growth of the Total Cost of Ownership (TCO)
  - Power wall towards exascale computing

Outline

Introduction

- Why linear algebra?
  - Linear algebra libraries reside at the bottom of the scientific computing "foodchain"
  - Its performance is key to many HPC applications, also its energy efficiency!
  - Insights can be extended to other areas
- Numerical scientific applications can be decomposed in simple kernels "dwarfs"
  - Implementations from MKL, GotoBLAS, OpenBLAS, BLIS, LAPACK, etc.
  - **Problem**: they are not deeply analized from the performance, power and energy perspectives

BLAS routines as the base of DLA

- BLAS-2 (matrix-vector operations):
  - $O(n^2)$ operations on a $= O(n^2)$ data
  - FLOPS limited by memory access
  - Presumedly memory bound operations
- BLAS-3 (matrix-matrix operations):
  - $O(n^3)$ operations on a $> O(n^2)$ data
  - FLOPS rate close to processor's peak!
  - CPU-bound operations

BLAS-2 and BLAS-3 have, by definition,
different behaviour from time-power-energy balance!

- Study case:
  - `dsymv`: BLAS-2 symmetric matrix-vector product
  - `dsyr2k`: BLAS-3 symmetric rank-2k update

BLAS routines

`dsyr2k`: BLAS-3 symmetric rank-$2k$ update

$$C := \beta C + \alpha A B^T + \alpha B A^T$$

- C: symmetric matrix of size $n \times n$
- A, B: factor matrices of size $n \times k$
- α, β: scalars
- Updates only the lower (or upper) triangular part of C

Costs:

- $2n^2k$ operations on $n^2/2 + 2nk$ DP elements
- CPU-bound with ($n \approx k$) but memory-bound as $k \to 1$

BLAS routines

`dsymv`: BLAS-2 symmetric matrix-vector product

$$y := \beta y + \alpha A x$$

- A: symmetric matrixs of size $n \times n$
- x, y: vectors of size n
- α, β: scalars

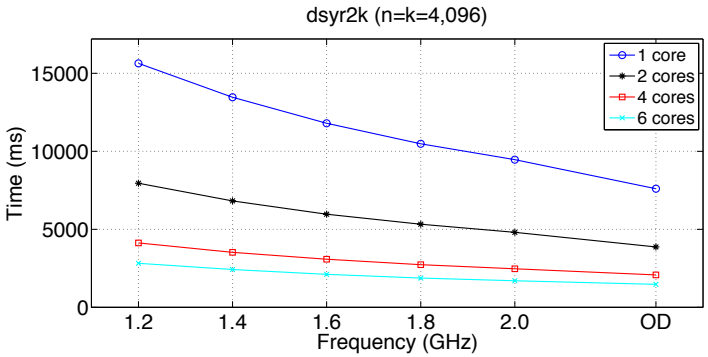
Costs:

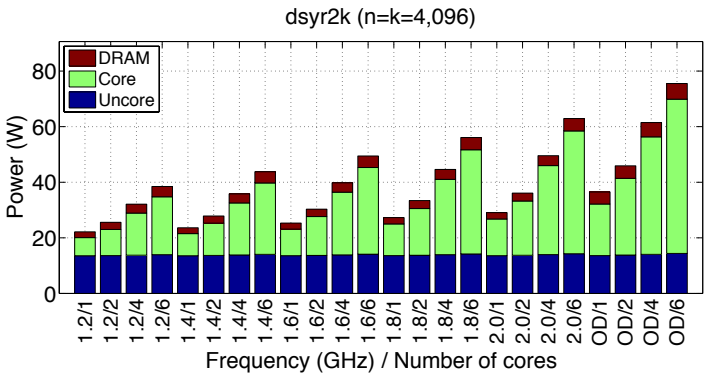
- $2n^2$ operations on $n^2/2$ DP elements
- Ratio of 4 flops per matrix element read, thus is a memory-bound operation

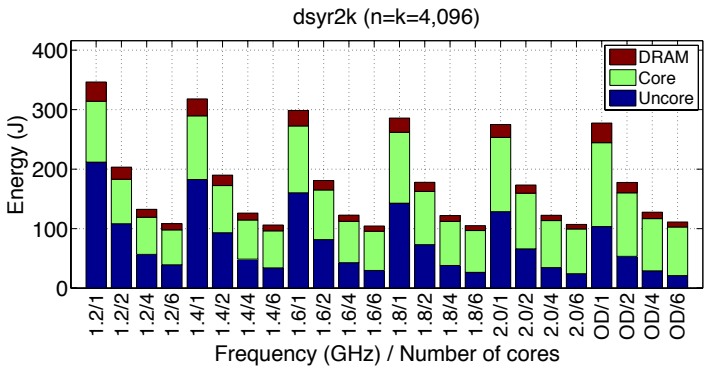
Environment setup

- Intel Xeon E5-2620 Sandy Bridge (6 cores) at 2.0 GHz with 32 Gbytes of DDR3 RAM (1.3 GHz)
  - Runs using 1, 2, 4 and 6 cores
  - {1.2,1.4,1.6,1.8,2.0} GHz (`userspace` governor)
  - 2.3 GHz turbo frequency (`ondemand` governor)
- Power measures obtained via the RAPL interface (MSR):
  - Core, Uncore, DRAM components
- Use of the reference LAPACK:
  - Double-precision (DP) kernels from OpenBLAS 0.2.92
  - $n = 4,096$ to prevent the problem fitting into L3 cache (15 MB)

Performance-Power-Energy in BLAS-3



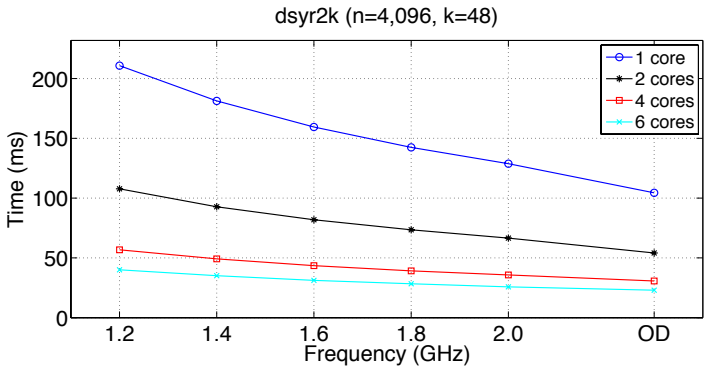


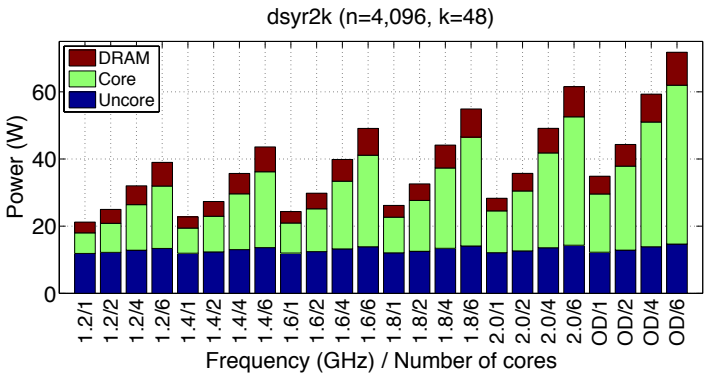


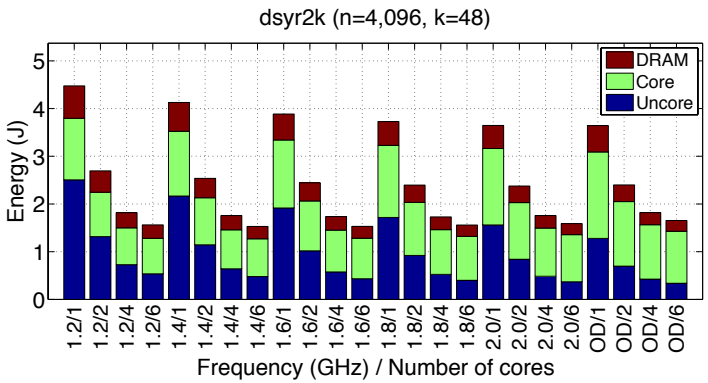
`dsyr2k`
(n=k=4,096)

- **Execution time**
  - Decreases when the number of cores and CPU frequency increases
  - Best option: OD/6 cores!
- **Average power**
  - Increases with the number of cores and CPU frequency
  - Almost only Core power changes!
- **Energy consumption**
  - The greenest is 1.6 GHz/6 cores
  - The fastest is OD/6 cores
  - Performance: 1.43×; Energy efficiency: 0.93×

Performance-Power-Energy in BLAS-3





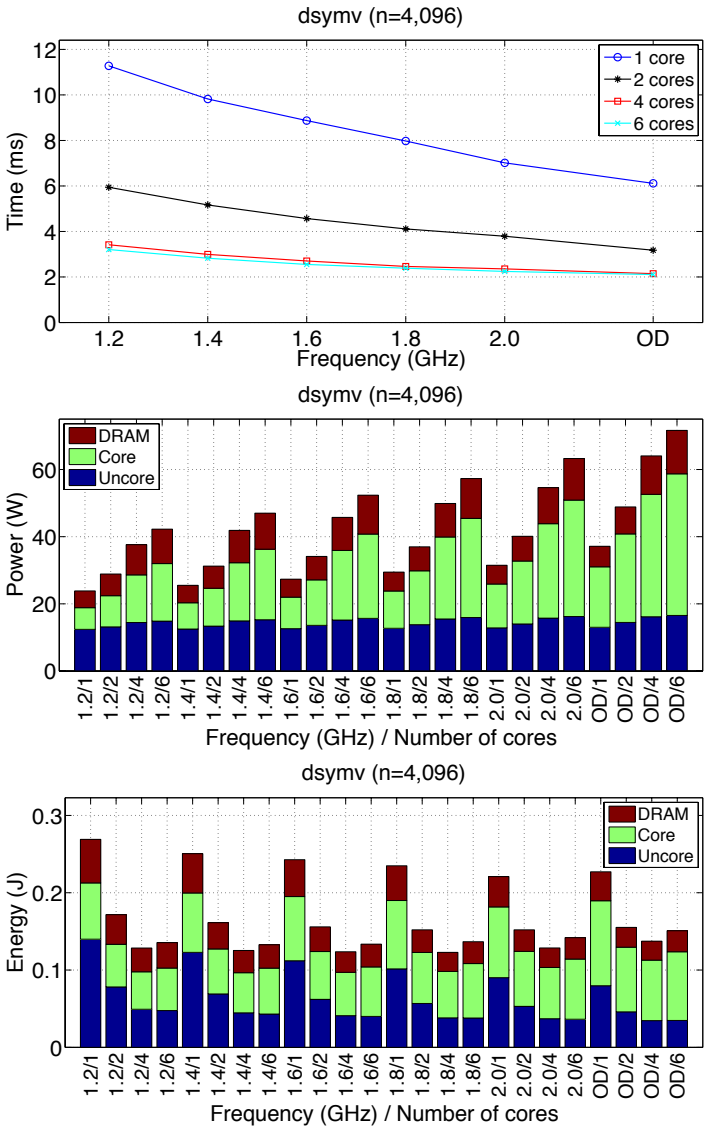


dsyr2k

(n=4,096, k=48)

- **Execution time**
  - Decreases when the number of cores and CPU frequency increases
  - Best option: OD/6 cores!
- **Average power**
  - Increases with the number of cores and CPU frequency
  - Core and DRAM power changes!
- **Energy consumption**
  - The greenest is 1.4 GHz/6 cores
  - The fastest is OD/6 cores
  - Performance: 1.52×; Energy efficiency: 0.92×

Performance-Power-Energy in BLAS-2



dsymv
(n=4,096)

- **Execution time**
  - Decreases when the number of cores grows up to 4
  - Memory-bottleneck or unbalanced OpenBLAS for 6 cores
  - Memory bandwith is proportional to the CPU frequency
- **Average power**
  - Increases with the number of cores and CPU frequency
  - Core and small DRAM power changes
- **Energy consumption**
  - The greenest is 1.8 GHz/4 cores
  - The fastest is OD/6 cores
  - Performance: 1.16×; Energy efficiency: 0.81×

Performance/power/energy trade-offs

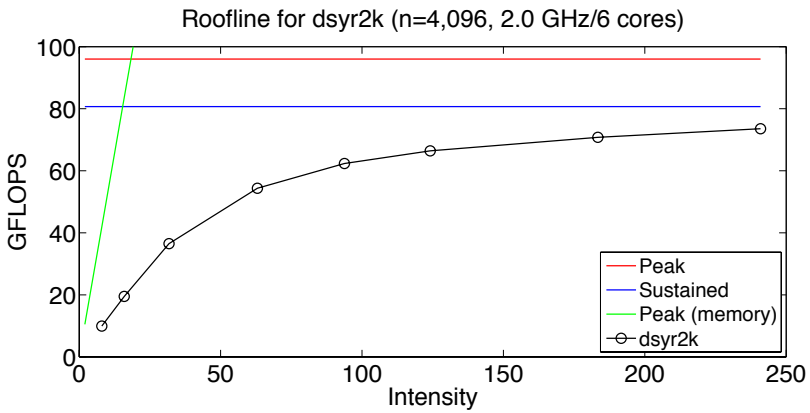
When is it crucial to perform a longer execution and decrease the power/energy consumption?

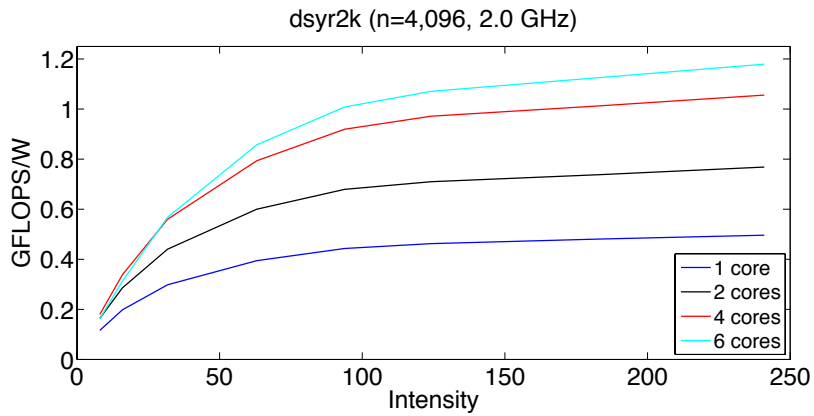
- Look at the *arithmetic intensity* of operations:
  - Ratio of flops and memory operations
    - `dsyr2k`: with $k = 48 \ll n$, then $2n^2k/n^2 \approx 2k = 96$
    - `dsymv`: 4
  - For some problems this depends on input parameters! (e.g., k)
- Studying the memory- vs. CPU-bound in more depth:
  - **Peak**: theoretical peak of the platform

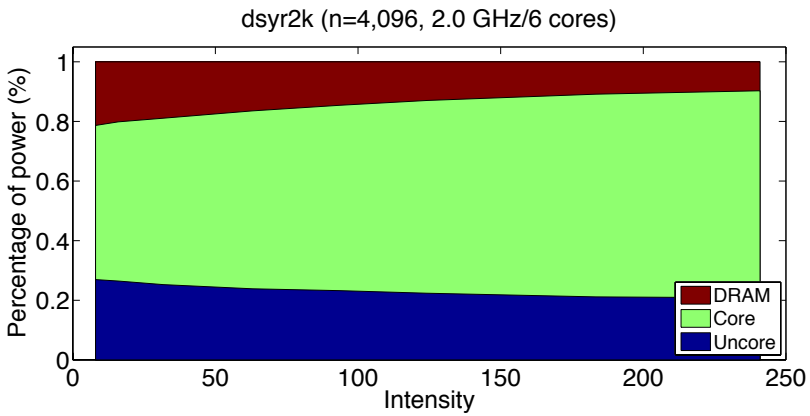
    6 cores at 2.0 GHz:
    8 DP flops/cycle $\times$ 2.0 GHz $\times$ 6 cores = 96 DP GFLOPS
  - **Sustained peak**: obtained by executing the kernel on a much larger problem size
  - **Peak (memory)**: obtained using the theoretical bandwith of the processor, 42 Gbytes/sec.

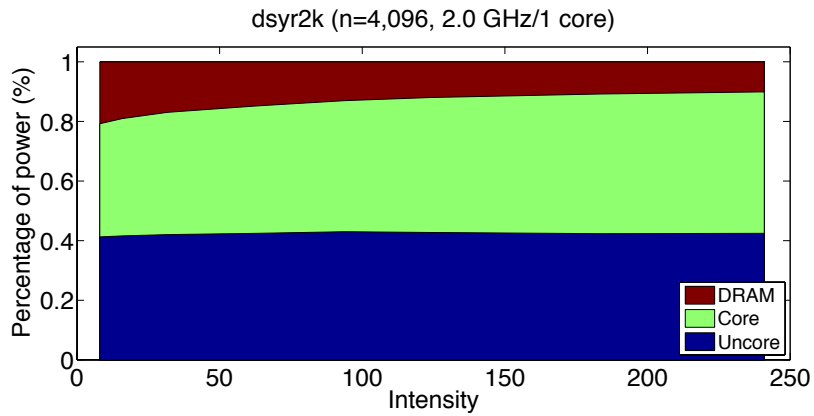
Arithmetic intensity of BLAS-3

Experiments with `dsyr2k` varying the arithmetic intensity $k = \{4, 8, 16 \ldots, 128\}$



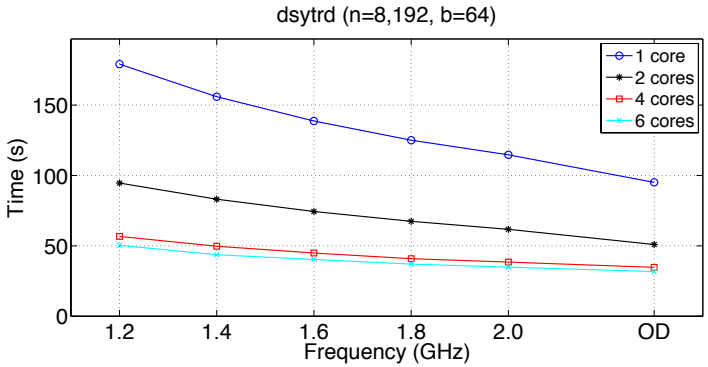






- Core power grows with the arithmetic intensity and DRAM is reduced (better locality)
- For 1 core, Uncore power is constant (40 %)
- GFLOPS/W grows with the *arithmetic intensity* and core count (multithread BLAS)

Arithmetic intensity of LAPACK



`dsytrd`

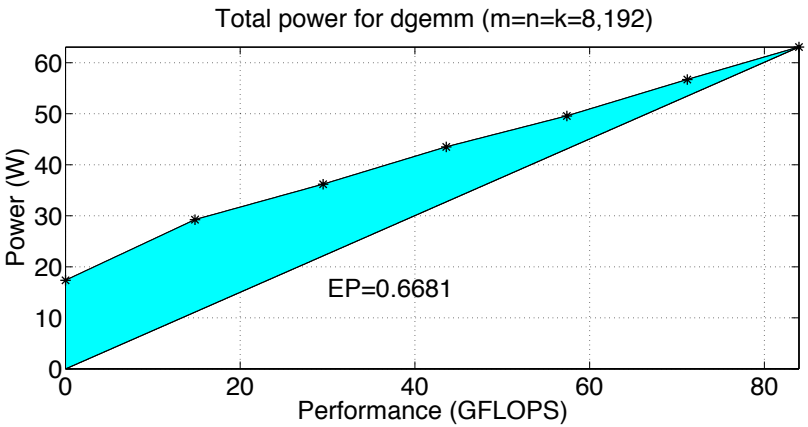
(n=8,192, b=64)

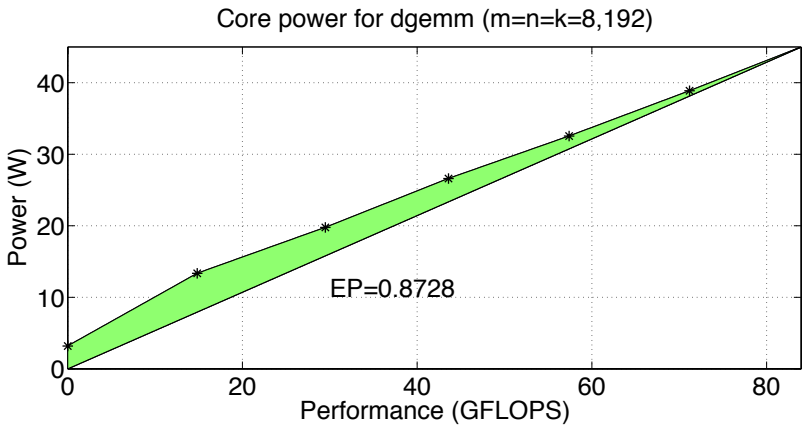
- LAPACK routine for symmetric eigenproblems
  - 50 % of flops are BLAS-2 (`dsymv`)
  - 50 % of flops are BLAS-3 (`dsyr2k`)
  - As the comptution proceeds, the size of the:
    - `dsyr2k` decreases in k steps
    - `dsymv` decreases in unit steps
- **Energy consumption**
  - The greenest is 1.4 GHz/6 cores
  - The fastest is OD/6 cores
  - Performance: 1.37×; Energy efficiency: 0.90×
- Dynamic DCT/DVFS is sometimes a delicate issue and requires further investigation

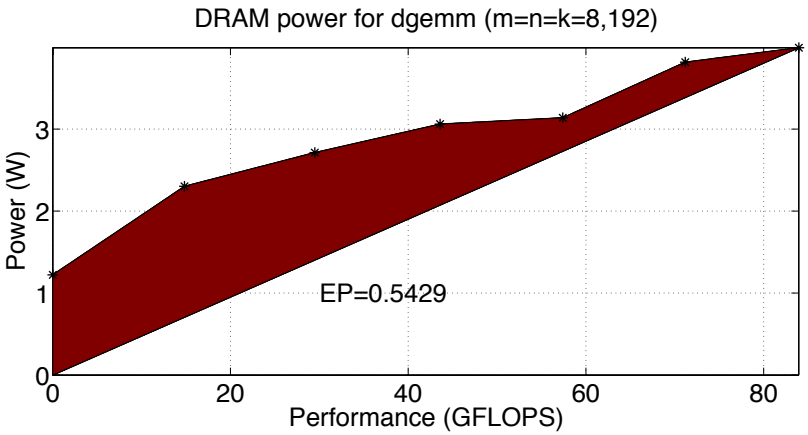
What is energy proportionality?

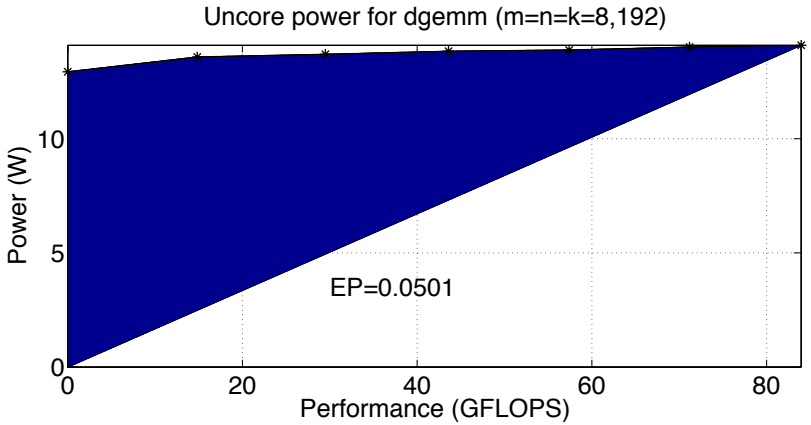
- **Energy proportionality**:
  - Power consumption grows linearly with the amount of work being performed
  - Null activity $\rightarrow$ zero power
  - Maximum throughput $\rightarrow$ maximum power
- For the DLA domain...
  - we consider FPUs and DDR power consumption and,
  - GFLOPS to measure the throughput
- Examples:
  - `dgemm`: BLAS-3 matrix-matrix product
    - CPU-bound operation: $2n^3$ flops on $3n^2$ DP elements
  - `dgemv`: BLAS-2 matrix-vector product
    - Memory-bound operation: $2n^2$ flops on n^2 DP elements

Energy proportionality of BLAS-3



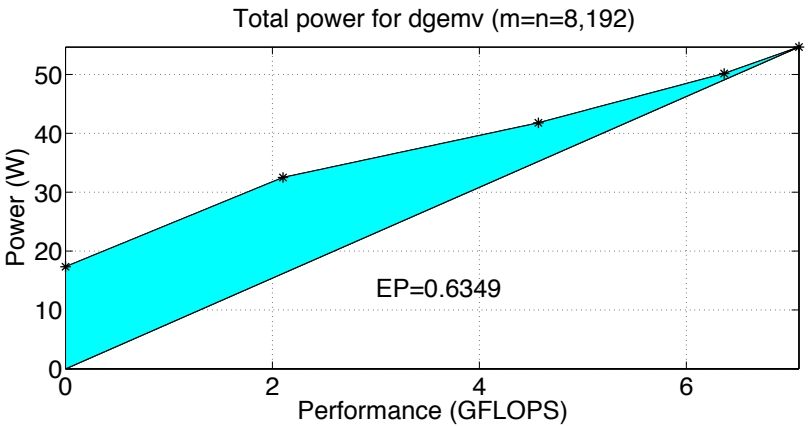


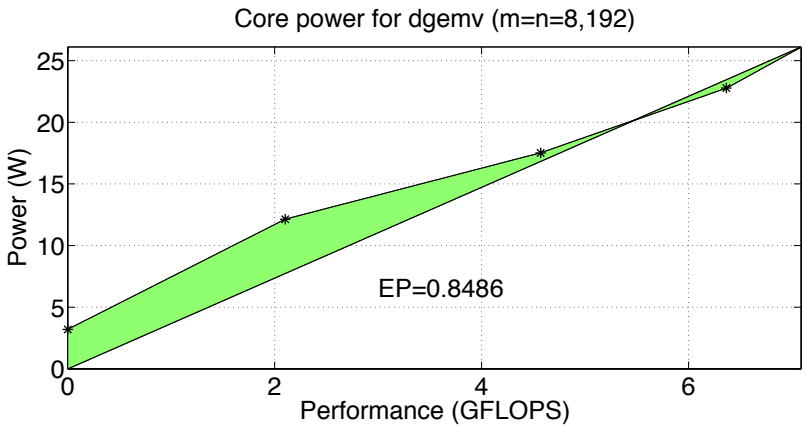


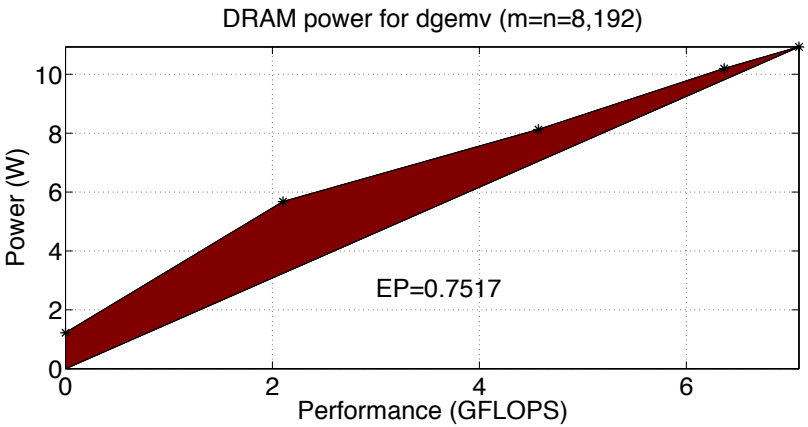


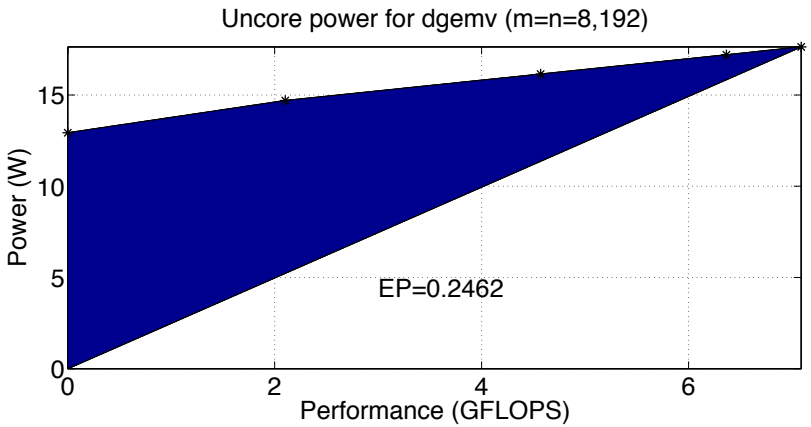
- dgemm with $m = n = k = 8,192$ running at 2.0 GHz
- Runs from 1 to 6 cores
- EP: 1 minus the integral of the curve divided by integral of the diagonal
  1 stands for a perfect case, 0 for the opposite

Energy proportionality of BLAS-2









- dgemv with $m = n = 8,192$ running at 2.0 GHz
- Runs from 1 to 4 cores
- EP: 1 minus the integral of the curve divided by integral of the diagonal
  1 stands for a perfect case, 0 for the opposite

Conclusions and future work

- Investigation between performance-power-energy of BLAS/LAPACK operations on a Intel Xeon E5 Sandy Bridge:
  - For performance and:
    - CPU-/memory-bound operations $\rightarrow$ run at the highest frequency with all cores
    - On this processor memory bandwith varies with the clock frequency!
  - For energy consumption:
    - Best DCT/DVFS combination depends on problem parameters (size, intensity, etc.)!
- Turning DLA routines energy-aware is not straight-forward!
  - Requires specific study on the target platform!
  - Dynamic DCT/DVFS is a solution but is a delicate issue (future work)
- We can have more energy savings if future architectures tend to proportional computing!

Thanks for your attention!

Questions?

Dr. Manuel F. Dolz
manuel.dolz@informatik.uni-hamburg.de