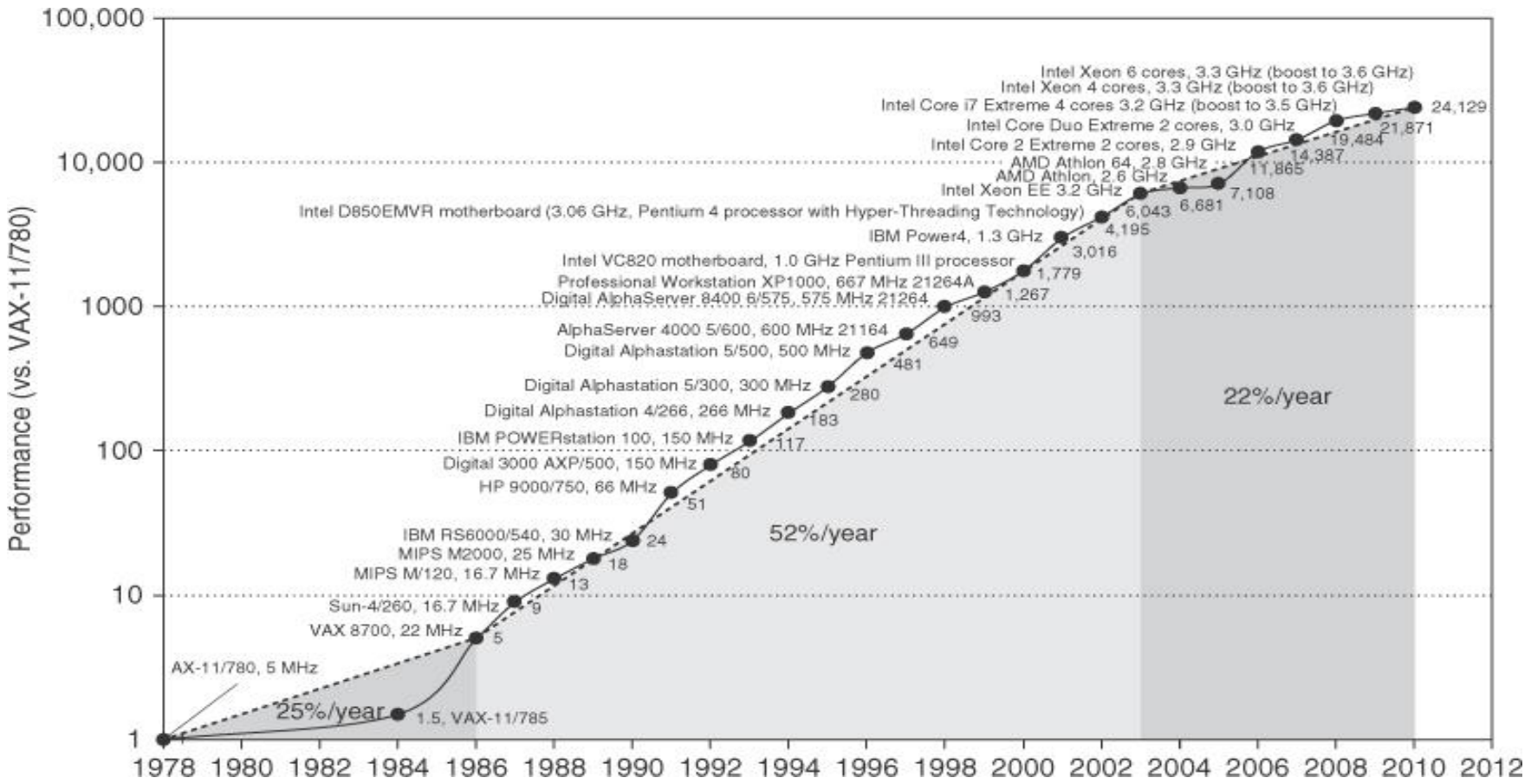
Energy Efficiency in Scientific Computing

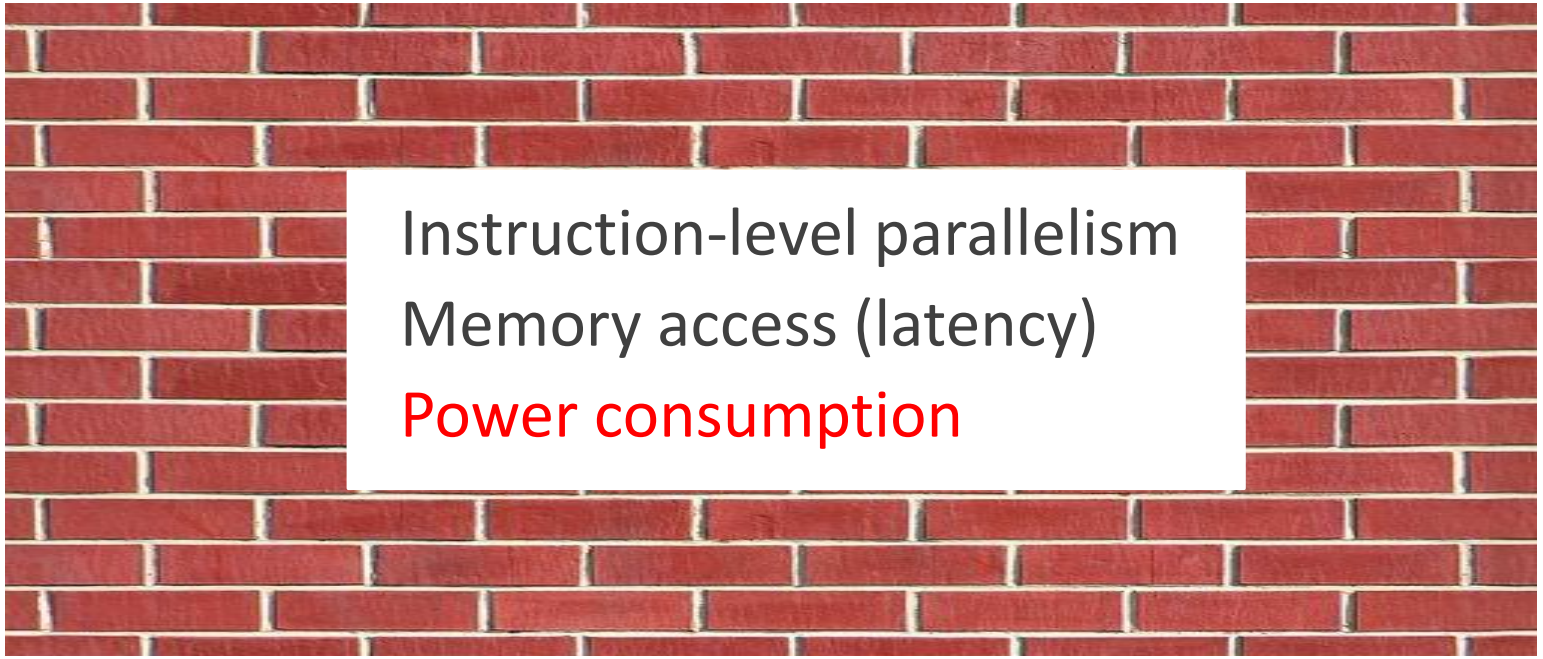
Enrique S. QUINTANA-ORTÍ
Professor of Computer Architecture
Group leader High Performance Computing & Architectures (HPC&A) group
http://www.uji.es/~quintana

Motivation



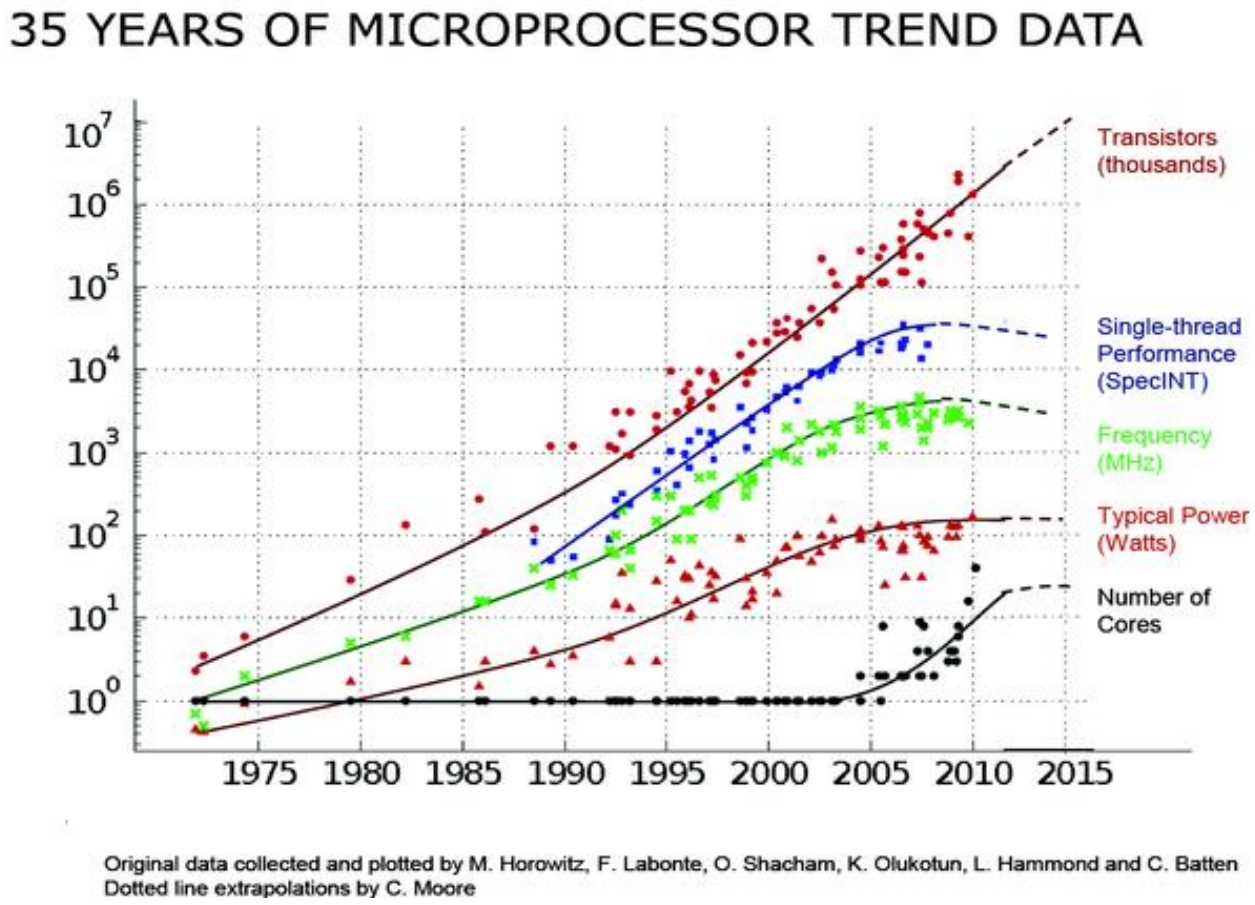
"Computer Organization and Design". D. A: Patterson, J. L. Hennessy, 2014

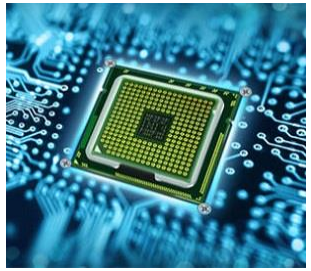
Motivation


Instruction-level parallelism
Memory access (latency)
Power consumption

Motivation

- Dennard's scaling vs Moore's Law





IBM Power8 (Q3'15)
- 22 nm
- 3.12 GHz
- TDP 190-200 W
- 12 cores/96 threads

Intel Xeon E7-8890 v4 (Q2'16)
- 14 nm
- 2.2 GHz
- TDP 165 W
- 24 cores/48 threads

Motivation

- 5 nm in about 4-6 years:
  - Faster
  - More transistors
  - … but only 10% simultaneously active (dark silicon)

Specialization!

Motivation
Concurrency and energy efficiency

- Top500 vs Green500 (June 2016)

Rank Top/Green	Site	Technology	MFLOPS/W
1/3	Sunway TaihuLight – National Supercomputing Center	Intel Xeon E5 8C 2.3 GHz + PEZY-SCnp	6,051
94/1	Shoubu - RIKEN	Sunway SW26010 260C 1.45GHz	6,673

Motivation
Concurrency and energy efficiency

- Top500 vs Green500 (June 2016)

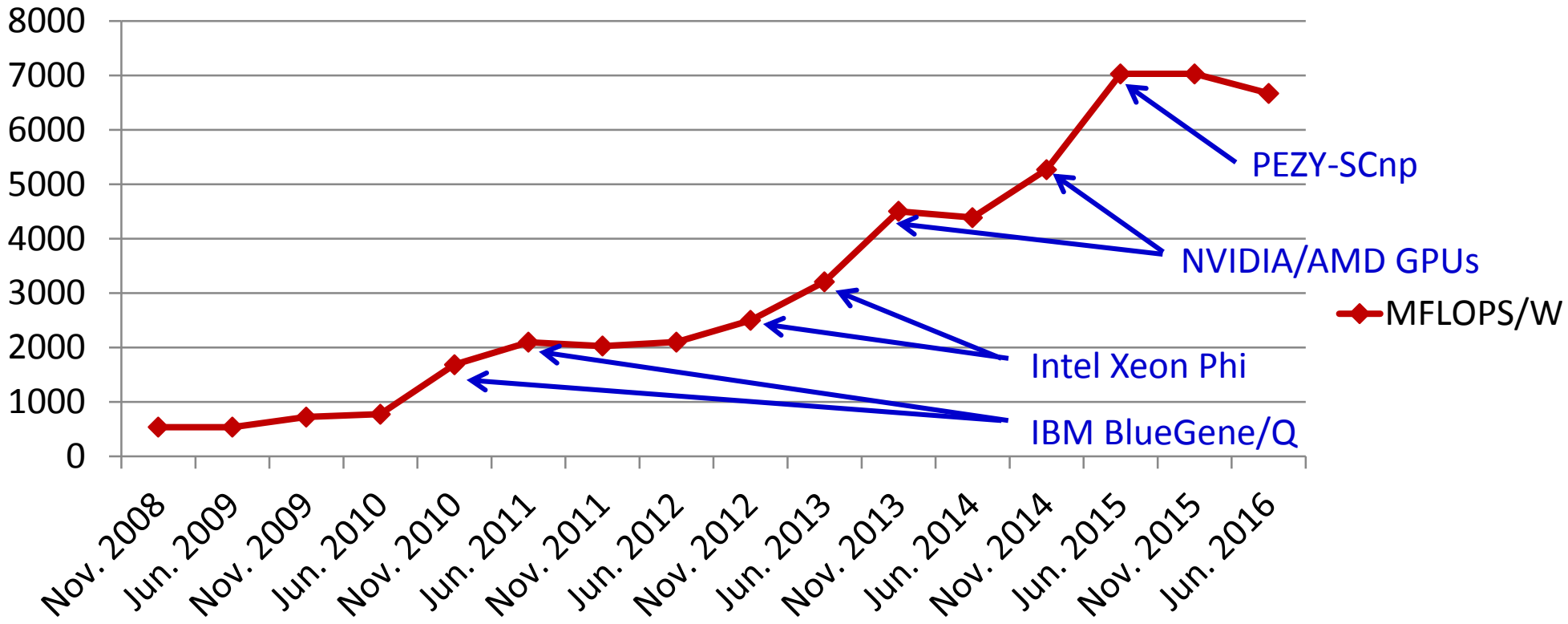
Rank Top/Green	Site	Technology	MFLOPS/W	MW to EXAFLOPS?
1/3	Sunway TaihuLight – National Supercomputing Center	Intel Xeon E5 8C 2.3 GHz + PEZY-SCnp	6,051	165
94/1	Shoubu - RIKEN	Sunway SW26010 260C 1.45GHz	6,673	149

Oskarshamn Nuclear Power Plant

- Most powerful in Sweden
- ABB-II: 1,400 MWe

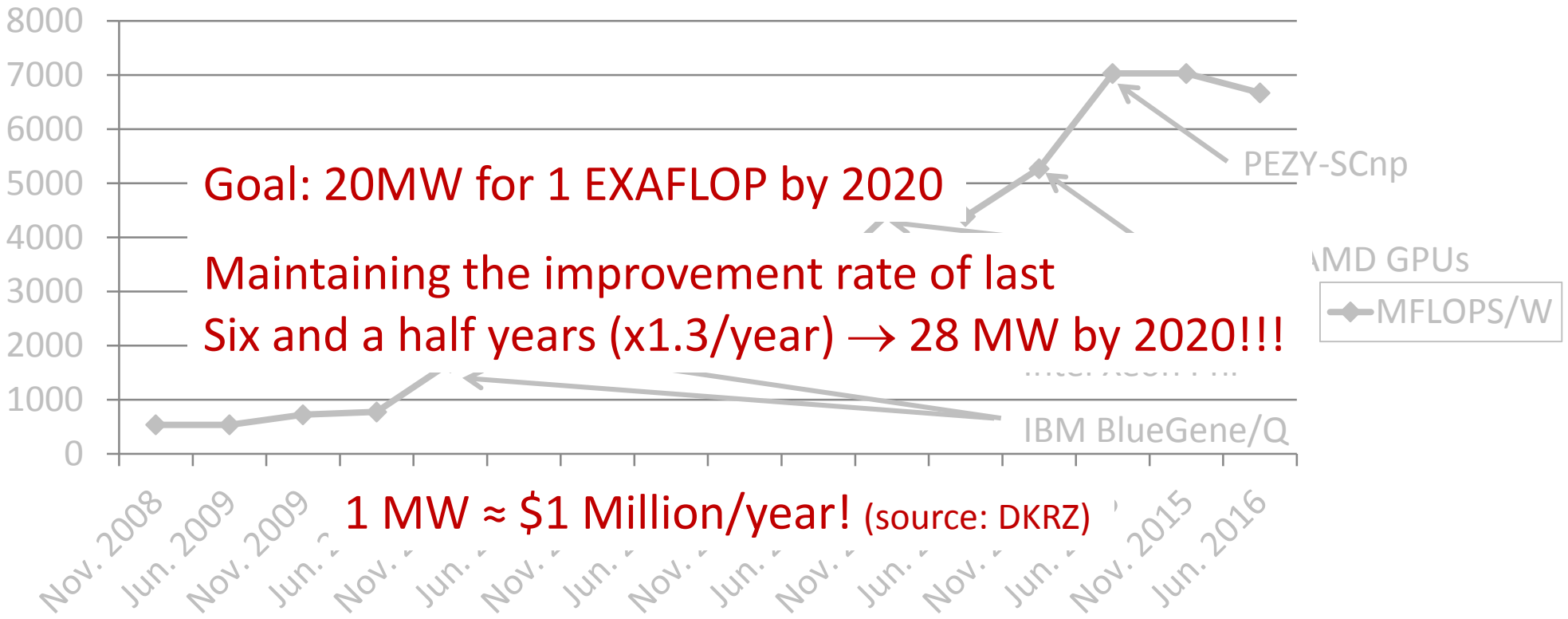
Motivation
Concurrency and energy efficiency

- System ranked #1 in Green500



Motivation
Concurrency and energy efficiency

- System ranked #1 in Green500



Motivation

- Reduce energy consumption!
  - Costs over lifetime of an HPC facility often exceed acquisition costs
  - Hazard for health and environment
  - Heat reduces hardware reliability

- Personal view
  - Hardware features some power-saving mechanisms (from mobile/embedded to desktop/server)
  - Scientific apps. are in general energy-oblivious

Outline

- What can I do? … A recipe to saving energy:

Optimize performance!!!

1. Choose the “right” hardware
2. Dynamic Voltage-Frequency Scaling (DVFS)
3. Dynamic Concurrency Throttling (DCT)
4. Avoid polling
5. Approximate computing/adaptive precisión
6. Near Threshold Voltage Computing (NTVC)
7. Energy-proportional hardware
8. Virtualization of HPC resources

A Recipe to Saving Energy

- DISCLAIMER
  - Sorry, most of the examples come from linear algebra:
    - Solution of dense/sparse linear systems via direct/iterative methods
    - Solution of eigenvalue problems
  - …but the message carries over to many other math kernels for scientific and engineering applications

A Recipe to Saving Energy
1. Choose the right hardware

- The Conjugate Gradient (CG) method is a representative of the performance/energy efficiency attained by real scientific applications (HPCCG benchmark)
- Performance depends on:
  - Target architecture: frequency-voltage setting, #cores, arithmetic floating-point precision, etc.
  - Compiler optimizations
  - Sparsity pattern
  - Storage format
  - Programmer's optimization effort

A Recipe to Saving Energy
1. Choose the right hardware

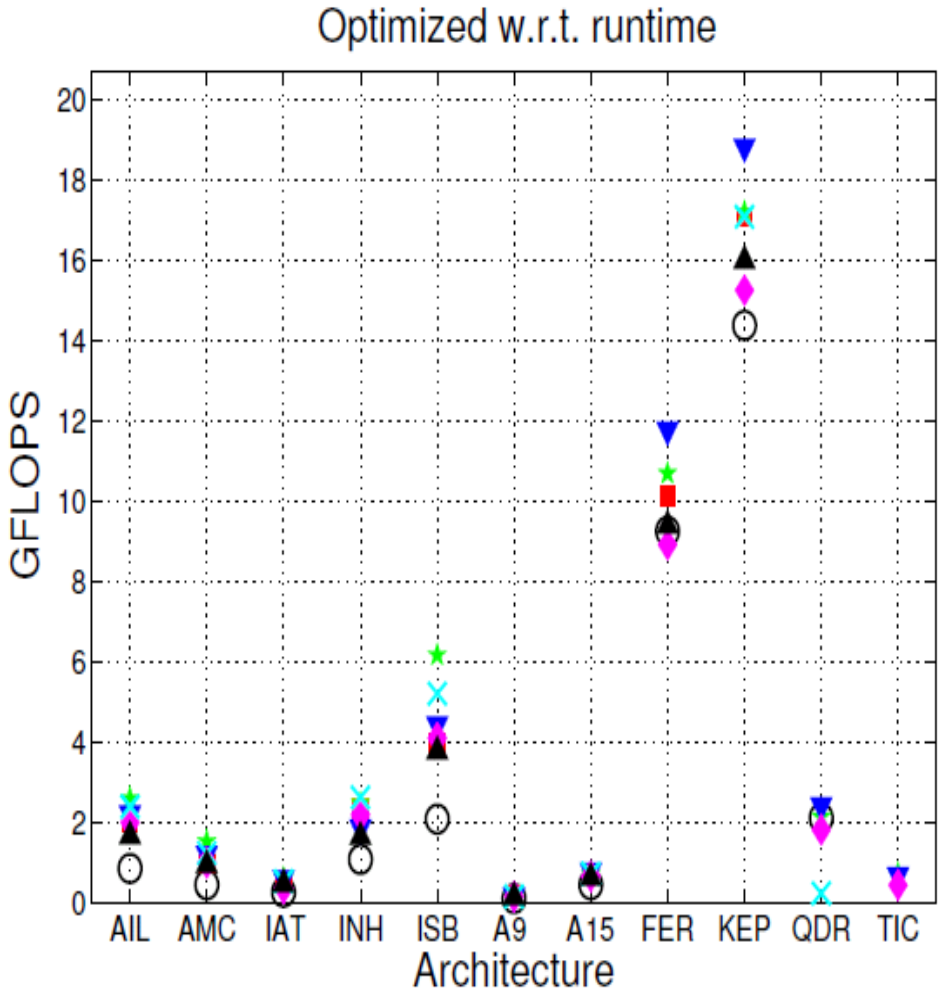
Acron.	Architecture	Total #cores	Frequency (GHz) – Idle power (W)	RAM size, type	Compiler
AIL	AMD Opteron 6276 (Interlagos)	8	1.4–167.29, 1.6–167.66 1.8–167.31, 2.1–167.17 2.3–168.90	64GB, DDR3 1.3GHz	`icc` 12.1.3
AMC	AMD Opteron 6128 (Magny-Cours)	8	0.8–107.48, 1.0–109.75, 1.2–114.27, 1.5–121.15, 2.0–130.07	48GB, DDR3 1.3GHz	`icc` 12.1.3
IAT	Intel Atom S1260	2	0.6–41.94, 0.90–41.93, 1.30–41.97, 1.70–41.95 2.0–42.01	8GB, DDR3 1.3GHz	`icc` 12.1.3
INH	Intel Xeon E5504 (Nehalem)	8	1.60–33.43, 1,73–33.43, 1.87–33.43, 2.00–33.43	32GB, DDR3 800MHz	`icc` 12.1.3
ISB	Intel E5-2620 (Sandy-Bridge)	6	1.2–113.00, 1.4–112.96, 1.6–112.77, 1.8–112.87, 2.0–112.85	32GB, DDR3 1.3GHz	`icc` 12.1.3
A9	ARM Cortex A9	4	0.76–10.0, 1.3–10.1	2GB, DDR3L	`gcc` 4.6.3
A15	Exynos5 Octa (ARM Cortex A15 + A7)	4+4	0.25–2.2, 1.6–2.4	2GB, LPDDR3	`gcc` 4.7
FER	Intel Xeon E5520 NVIDIA Tesla C2050 (Fermi)	8 448	1.6–222.0, 2.27–226.0 1.15	24GB, 3GB, GDDR5	`gcc` 4.4.6 `nvcc` 5.5
KEP	Intel Xeon i7-3930K NVIDIA Tesla K20 (Kepler)	6 2,496	1.2–106.30, 3.2–106.50 0.7	24GB, 5GB, GDDR5	`gcc` 4.4.6 `nvcc` 5.5
QDR	ARM Cortex A9 NVIDIA Quadro 1000M	4 96	0.120–11.2, 1.3–12.2 1.4	2GB, DDR3L 2GB, DDR3	`gcc` 4.6.3 `nvcc` 5.5
TIC	Texas Instruments C6678	8	1.0–18.0	512MB, DDR3	`cl6x` 7.4.1

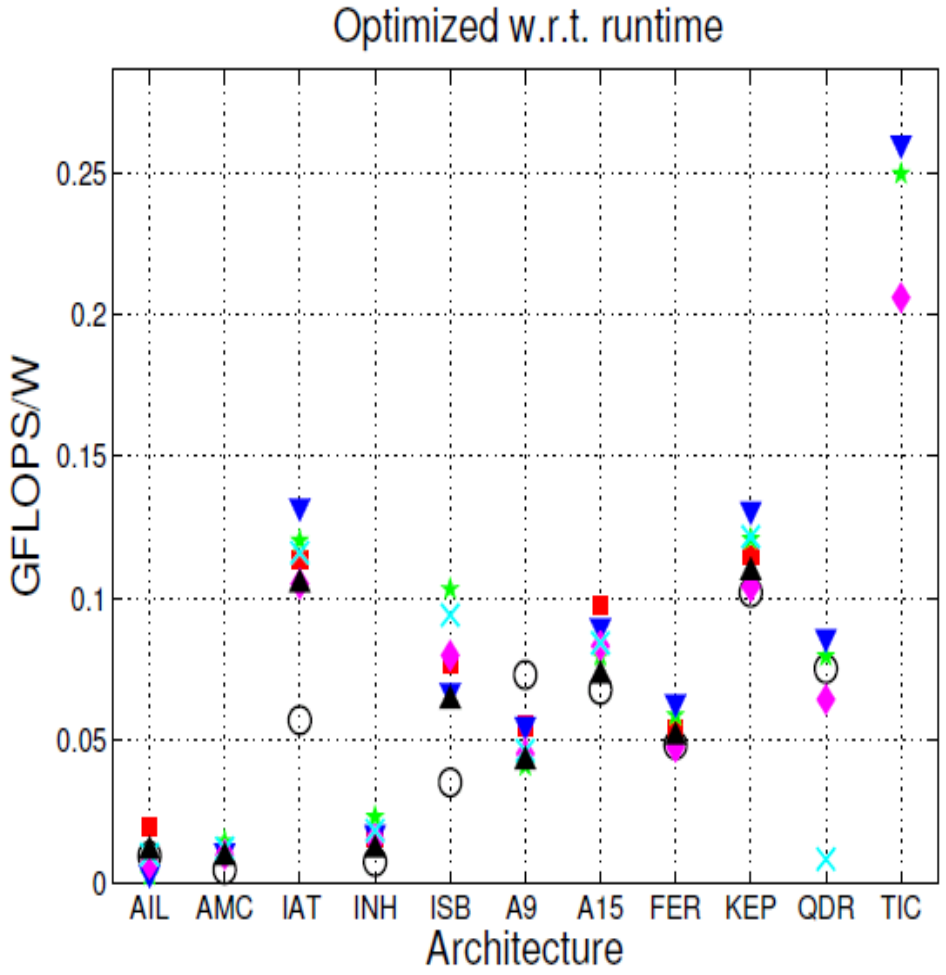
A Recipe to Saving Energy
1. Choose the right hardware

- Optimization effort:
  - Multicore x86-based: Intel MKL with CSR and BCSR, and CSB library
  - Other multicore: CSR+OpenMP
  - GPUs: ELLPACK & SELL-P, with further optimizations (described in last block)

A Recipe to Saving Energy

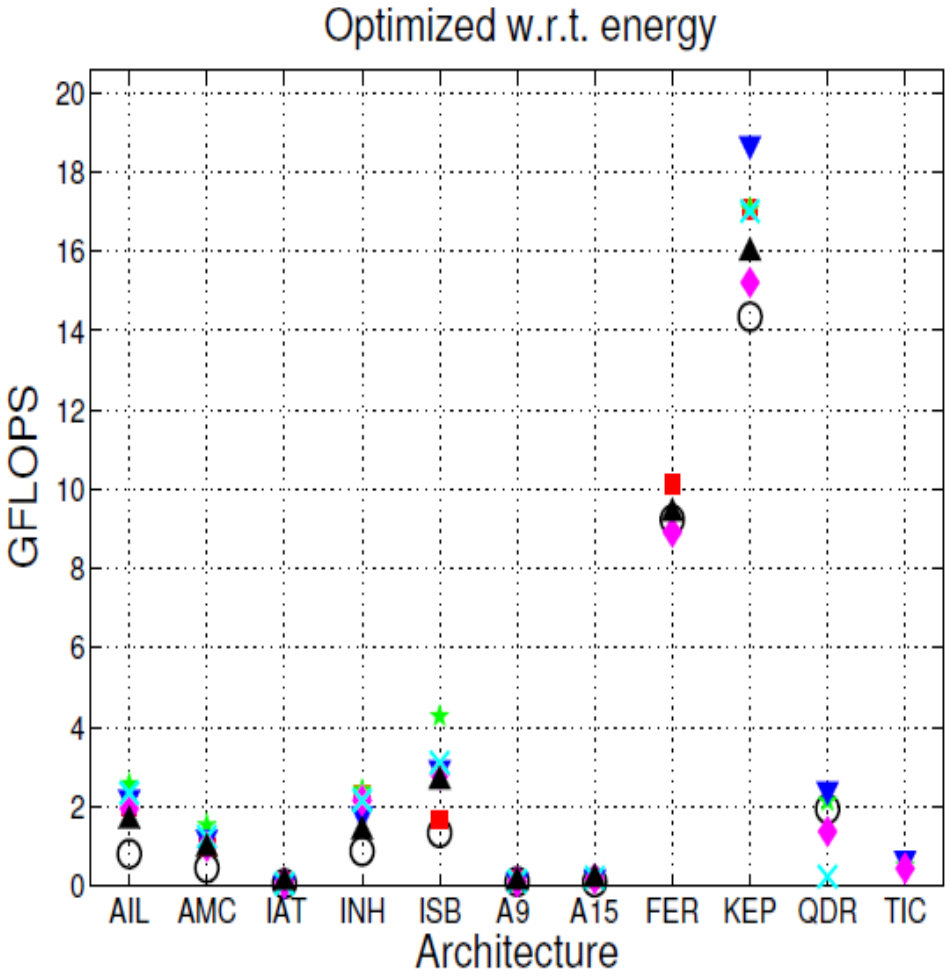
1. Choose the right hardware

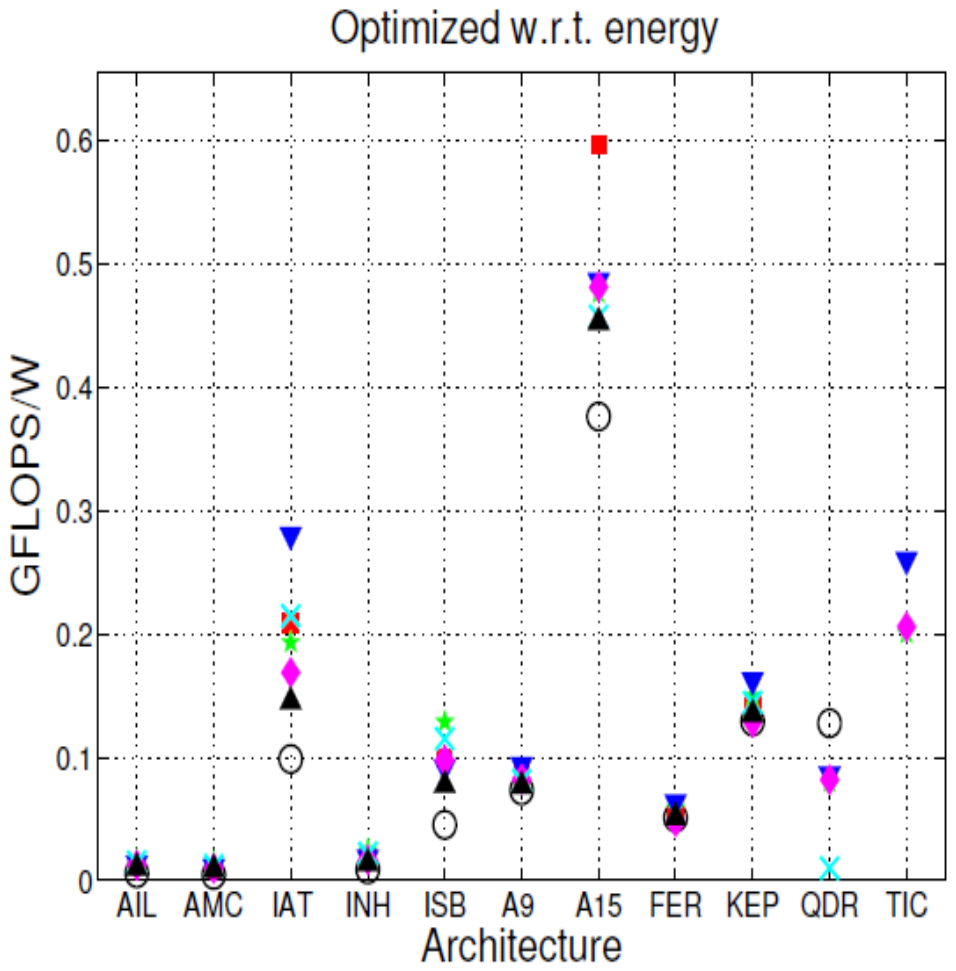




A Recipe to Saving Energy

1. Choose the right hardware





A Recipe to Saving Energy
2. DVFS

- Dense linear algebra kernels are the building blocks for many scientific and engineering applications: _GEMV, _GEMM
- (Dense) LU factorization is the basis for the LINPACK benchmark (Top500/Green500 lists): _GETRF
- Routines are highly optimized as part of vendor implementations of BLAS/LAPACK (Intel MKL, AMD ACML, IBM ESSL, NVIDIA CuBLAS, etc.)

A Recipe to Saving Energy
2. DVFS

- Current processors adhere to the ACPI (Advanced Configuration and Power Interface) standard:
  - P-states:
    - Adjust voltage-frequency to the workload in execution
    - Control by the Linux kernel or user
  - C-states:
    - Suspend processor components to save energy
    - Can waste energy if CPU needs to be activated soon
    - No control by the user

A Recipe to Saving Energy
2. DVFS

- P-states

Conf.	ARM Cortex-A7		ARM Cortex-A15		ARM Cortex-A53		ARM Cortex-A57	
	Freq.	Voltage	Freq.	Voltage	Freq.	Voltage	Freq.	Voltage
C_1	0.200	0.913	0.200	0.912	0.450	0.820	0.450	0.810
C_2	0.400	0.913	0.400	0.912	0.575	0.860	0.625	0.850
C_3	0.600	0.951	0.600	0.912	0.700	0.910	0.800	0.900
C_4	0.800	1.026	0.800	0.925	0.775	0.960	0.950	0.950
C_5	1.000	1.101	1.000	0.973	0.850	1.010	1.100	1.000
C_6	1.200	1.176	1.200	1.023	–	–	–	–
C_7	1.400	1.273	1.400	1.062	–	–	–	–
C_8	–	–	1.600	1.115	–	–	–	–
C_9	–	–	1.800	1.191	–	–	–	–
C_{10}	–	–	2.000	1.318	–	–	–	–

A Recipe to Saving Energy
2. DVFS

- P-states

Architecture	Configuration	GEMV			GEMM			GETRF		
		G	P	EE	G	P	EE	G	P	EE
ARM Cortex-A7	C_1	0.271	0.064	4.238	0.758	0.072	10.473	0.554	0.060	9.259
	C_2	0.501	0.115	4.360	1.573	0.140	11.204	1.193	0.119	9.984
	C_3	0.677	0.166	4.086	2.355	0.217	10.834	1.809	0.186	9.733
	C_4	0.802	0.233	3.440	3.258	0.328	9.937	2.398	0.279	8.603
	C_5	0.911	0.319	2.853	4.070	0.483	8.426	2.998	0.409	7.323
	C_6	0.999	0.417	2.395	4.893	0.672	7.278	3.536	0.561	6.303
	C_7	1.000	0.541	1.848	5.630	0.943	5.968	3.980	0.784	5.078
ARM Cortex-A15	C_1	0.381	0.188	2.028	3.502	0.496	7.067	2.205	0.388	5.679
	C_2	0.718	0.335	2.141	7.109	0.970	7.328	4.619	0.761	6.068
	C_3	0.997	0.471	2.115	10.652	1.436	7.418	7.067	1.132	6.245
	C_4	1.227	0.582	2.108	14.165	1.926	7.356	9.232	1.506	6.130
	C_5	1.396	0.768	1.817	17.757	2.686	6.611	11.702	2.120	5.519
	C_6	1.539	0.981	1.568	21.145	3.632	5.822	13.700	2.793	4.905
	C_7	1.648	1.182	1.394	24.344	4.562	5.336	15.856	3.532	4.489
	C_8	1.756	1.489	1.179	27.710	5.978	4.635	17.152	4.539	3.779
	C_9	1.728	1.855	0.931	–	–	–	–	–	–
	C_{10}	1.744	2.569	0.679	–	–	–	–	–	–
ARM Cortex-A53	C_1	0.877	0.198	4.425	7.461	0.359	20.787	4.901	0.272	18.148
	C_2	1.008	0.259	3.887	9.488	0.510	18.594	6.221	0.374	16.818
	C_3	1.106	0.327	3.387	11.271	0.685	16.447	7.453	0.494	15.204
	C_4	1.148	0.399	2.880	12.533	0.855	14.658	8.164	0.620	13.161
	C_5	1.191	0.470	2.536	13.629	1.045	13.040	8.883	0.727	12.333
ARM Cortex-A57	C_1	0.733	0.270	2.715	6.159	0.536	11.482	4.189	0.509	8.222
	C_2	0.972	0.404	2.408	8.491	0.843	10.072	5.801	0.781	7.429
	C_3	1.163	0.560	2.077	10.812	1.214	8.903	7.375	1.157	6.373
	C_4	1.286	0.709	1.814	12.698	1.586	8.006	8.702	1.490	5.839
	C_5	1.375	0.858	1.603	14.538	2.059	7.059	9.982	1.952	5.115

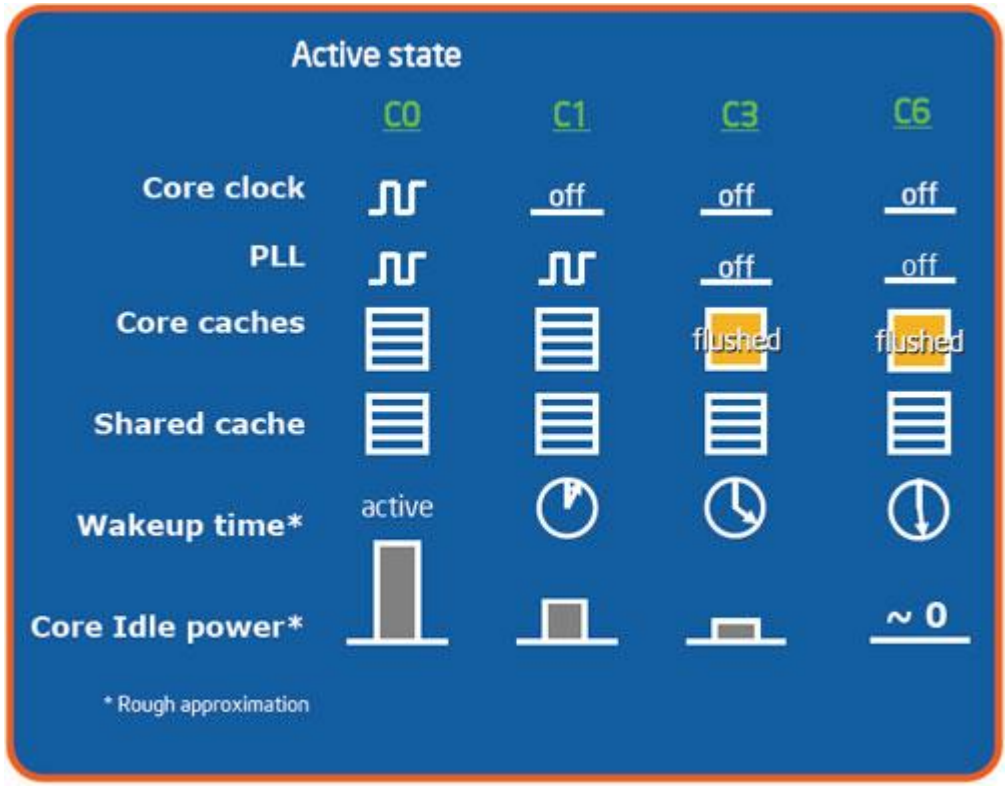
A Recipe to Saving Energy
2. DVFS

- Control of P-states by user possible via cpufrequtils, but too slow:
  - 225 µseconds in Intel E5-2620
  - Directly writing in MSR (in µseconds):

f_T \ f_S	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
1.2	–	28.98	29.11	29.07	28.98	29.14	29.57	29.08	29.32
1.3	34.10	–	29.14	28.90	28.99	29.23	28.98	29.13	28.78
1.4	35.33	34.14	–	28.96	29.00	28.65	29.04	29.26	28.90
1.5	35.60	34.68	33.29	–	28.39	28.57	28.83	28.99	28.68
1.6	35.61	35.09	34.16	33.63	–	28.08	28.12	28.34	28.50
1.7	35.63	35.41	34.74	34.30	32.35	–	28.27	28.33	28.21
1.8	36.69	36.30	35.40	35.05	34.50	33.70	–	28.36	28.23
1.9	36.89	36.21	36.02	35.29	34.63	34.19	32.52	–	27.70
2.0	37.22	36.65	36.27	35.49	35.24	34.31	33.54	32.54	–

A Recipe to Saving Energy
2. DVFS

- C-states (Core i7-Nehalem, similar for others)



A Recipe to Saving Energy
3. DCT

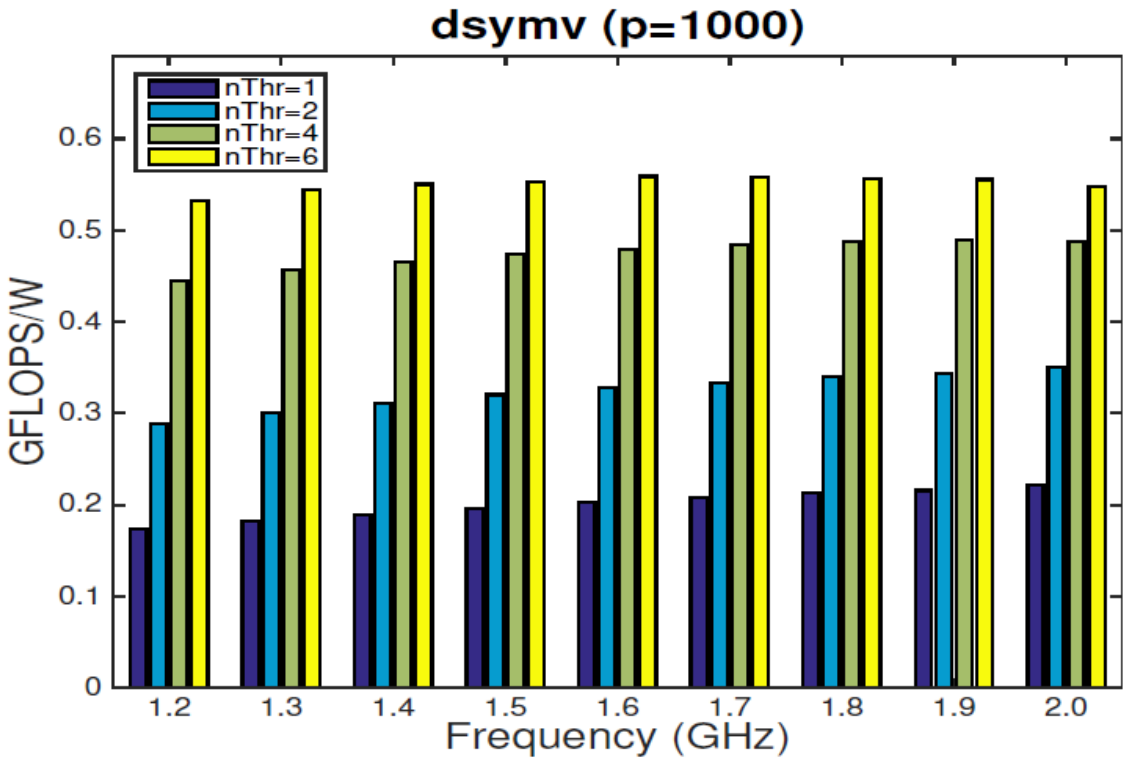
- Solution of eigenvalue problems is one of the cornerstones for scientific/engineering applications
- In many cases, the problem is dense and presents a symmetric structure

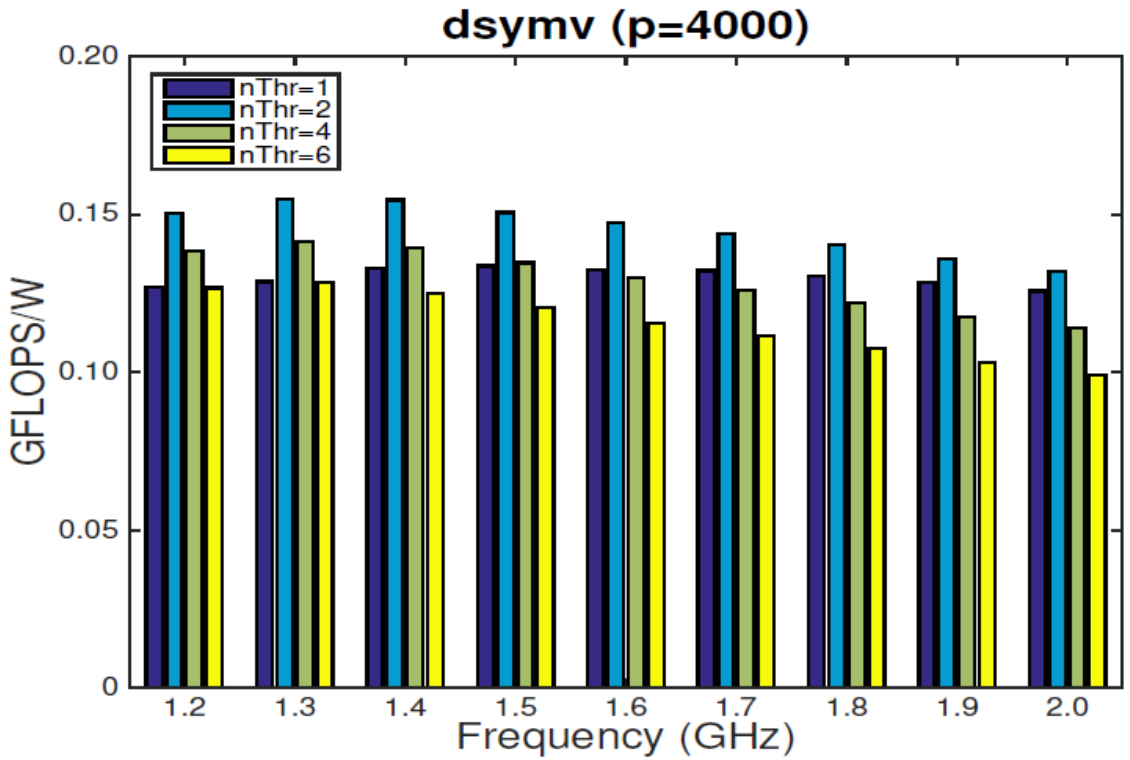
A Recipe to Saving Energy
3. DCT

- Control the number of threads in execution
  - More threads does not necessarily mean faster
  - Even if (slightly faster, or at least not slower), it may not be more energy efficient

A Recipe to Saving Energy
3. DCT

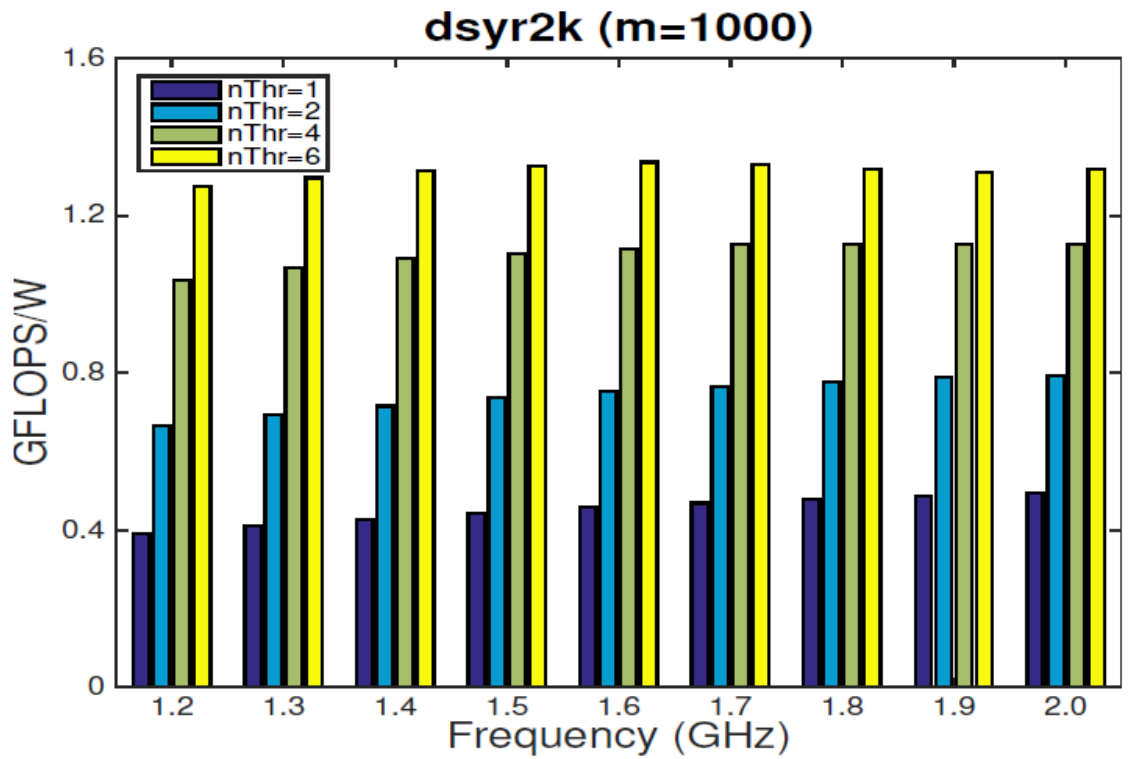
- Intel E5-2620. DSYMV:

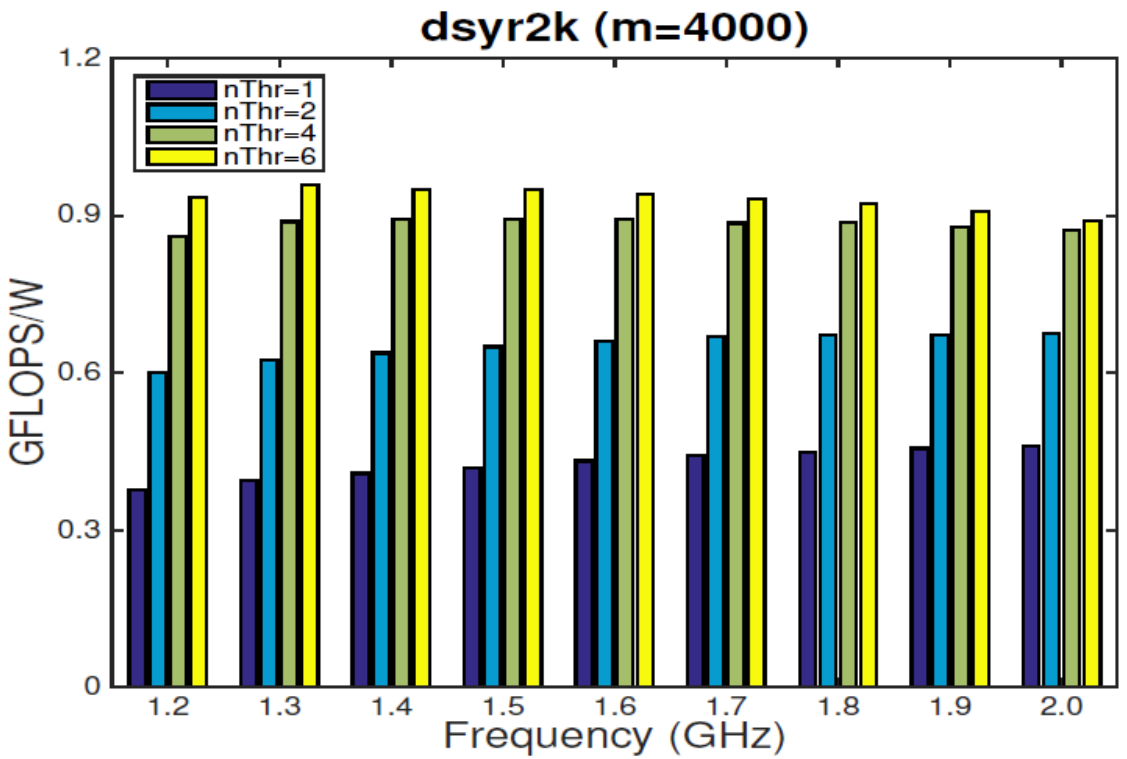




A Recipe to Saving Energy
3. DCT

- Intel E5-2620. DSYR2K:





A Recipe to Saving Energy
3. DCT

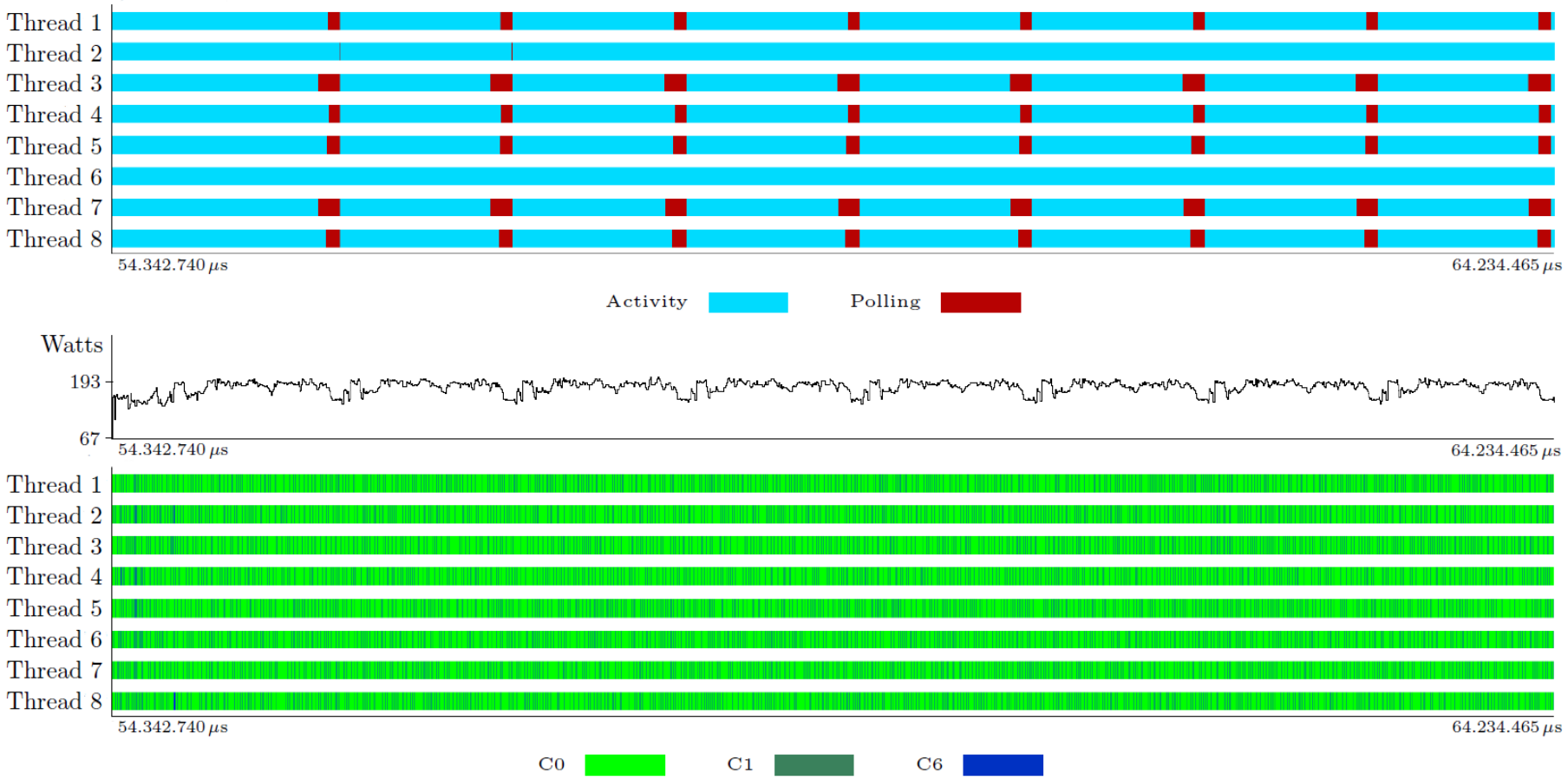
- Control with fine granularity may be necessary:
  - Reduction to tridiagonal from via _SYTRD (key for the solution of dense eigenvalue problems) spends half of its flops in _SYMV and the other half in _SYR2K
  - Subproblems become progressively smaller, till they fit into the cache

A Recipe to Saving Energy
4. Avoid polling

- Do nothing well!
- Polling ensures a rapid reaction of CPU to status changes, but prevents it from entering energy-saving C-states
  - Wait for other tasks to complete (task-parallelism, synchronization)
  - CPU-GPU execution
  - MPI blocking routines

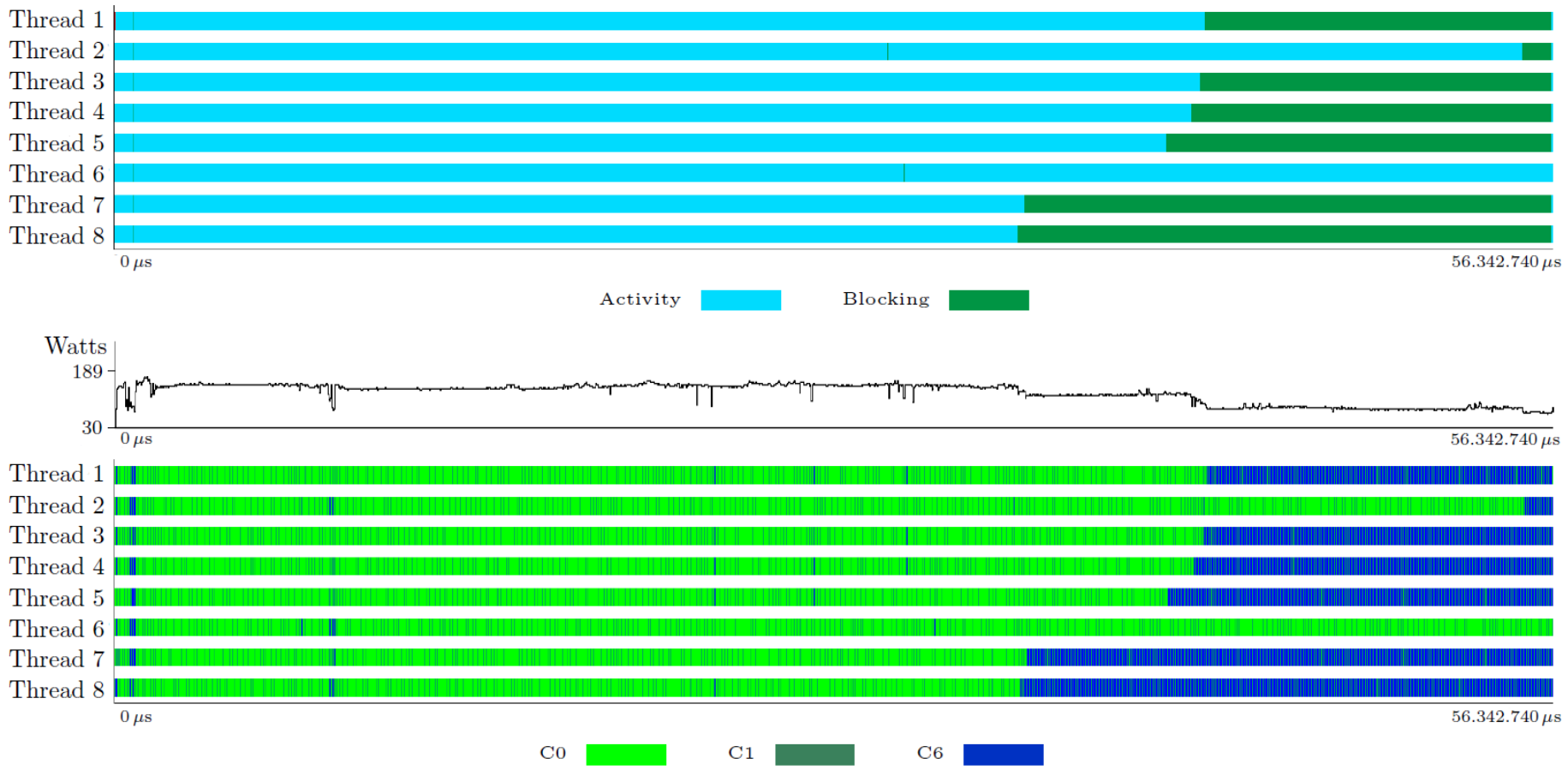
A Recipe to Saving Energy
4. Avoid polling

- ILUPACK's PCG on Intel Xeon E5504 (2x4 cores)



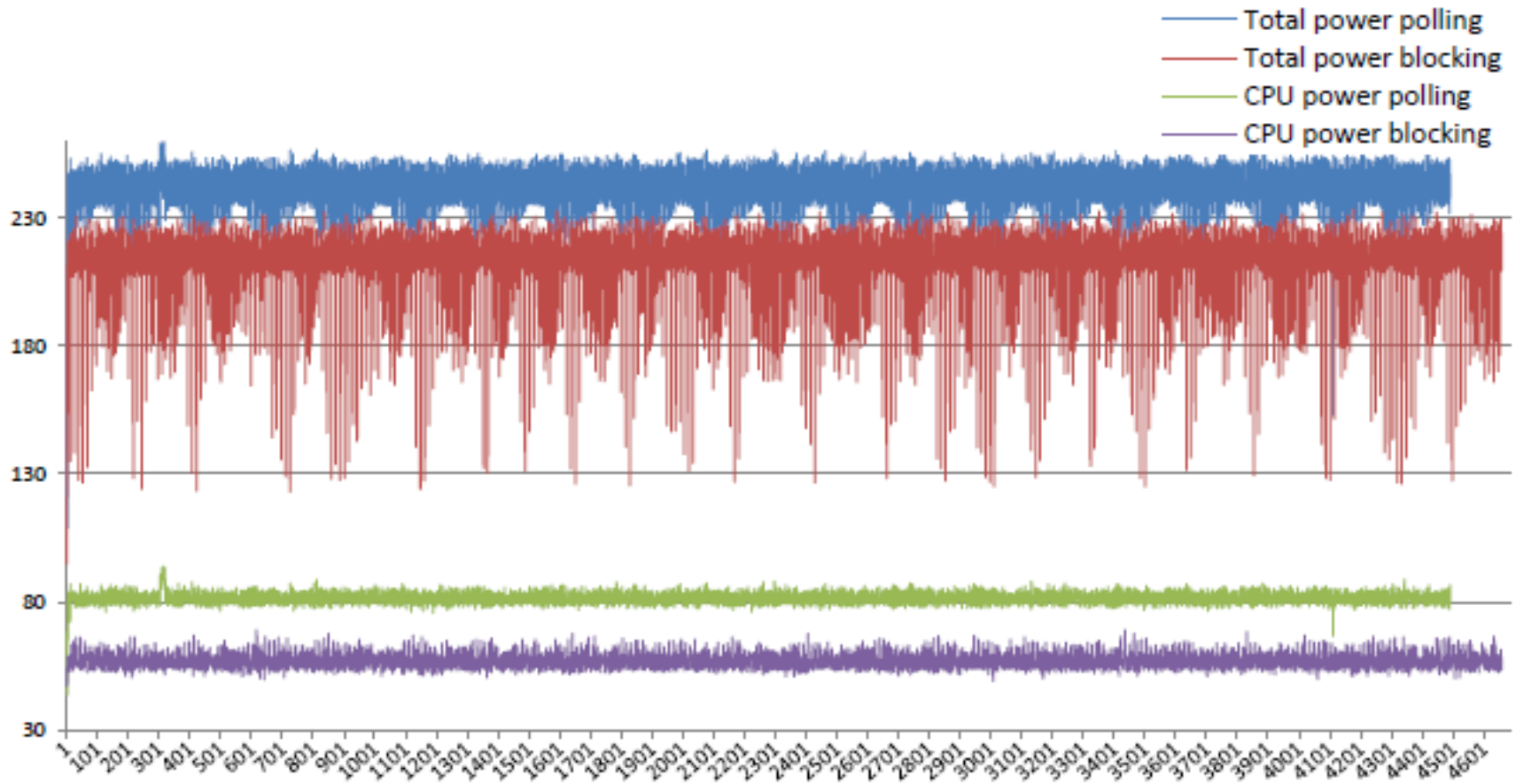
A Recipe to Saving Energy
4. Avoid polling

- ILUPACK's PCG on Intel Xeon E5504 (2x4 cores)



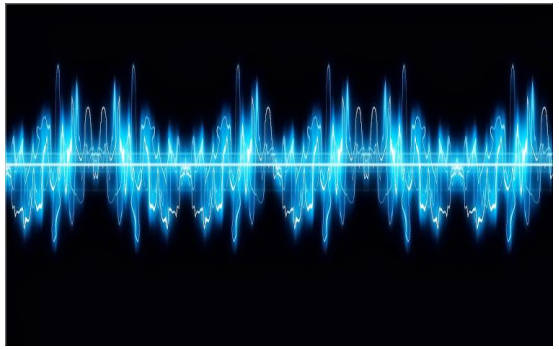
A Recipe to Saving Energy
4. Avoid polling

- CG on GPU: Intel Core i7-3770K + NVIDIA GeForce GTX480

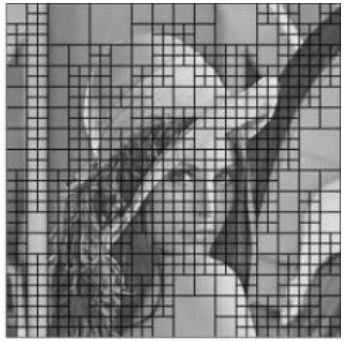


A Recipe to Saving Energy
5. Approximate computing

- Some applications do not need a “fully accurate” answer:
  - Signal & video processing
  - Probabilistic inference
  - Service profiling
  - Monte Carlo simulation
  - Machine learning



(a)

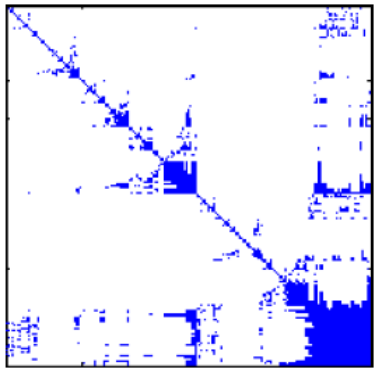


(b)

- Trade off accuracy for energy

A Recipe to Saving Energy
5. Approximate computing

- Numerical linear algebra for scientific computing?
  - Tiny errors (round-off) can rapidly “aggregate”
  - Double precision is the standard



- Can we work in reduced precision (most of the time), but still compute a full-precision solution?
  - Adaptive precision

A Recipe to Saving Energy

5. Approximate computing

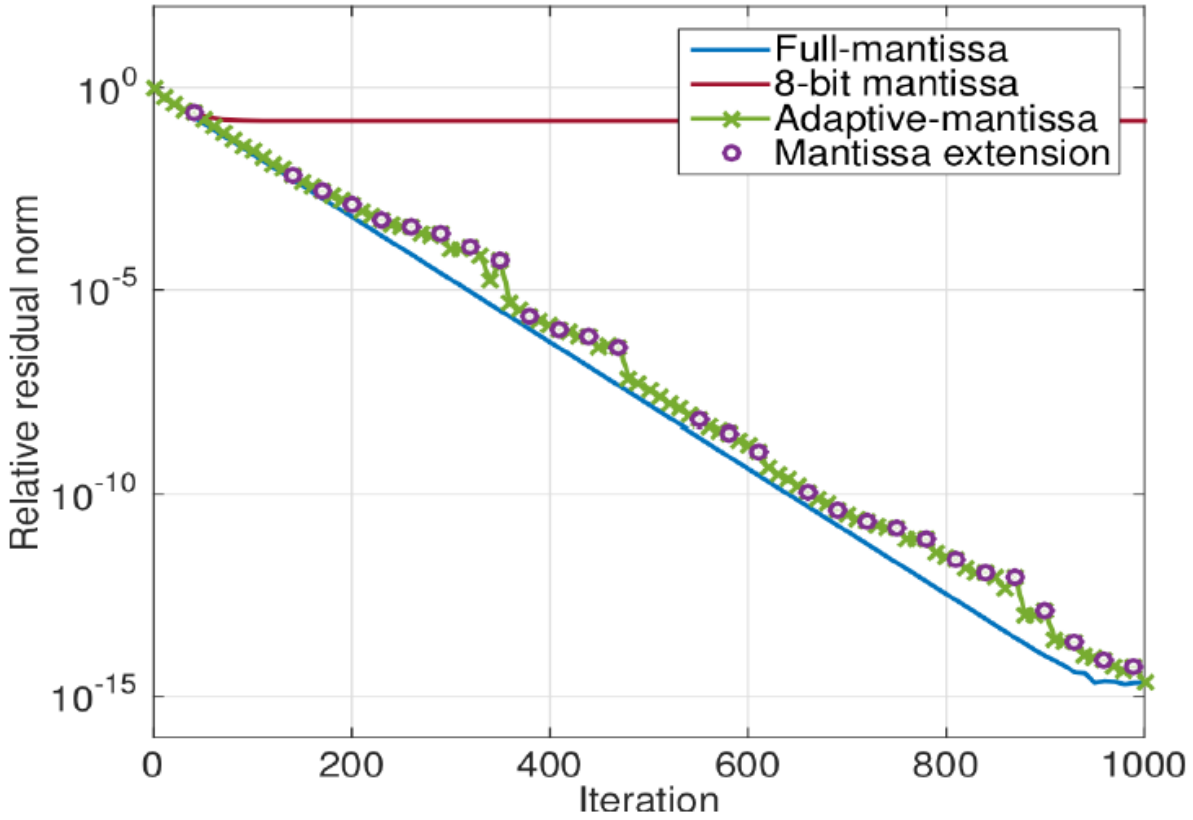
- For linear systems, mixed precision+iterative refinement is a long-known technique with good results:

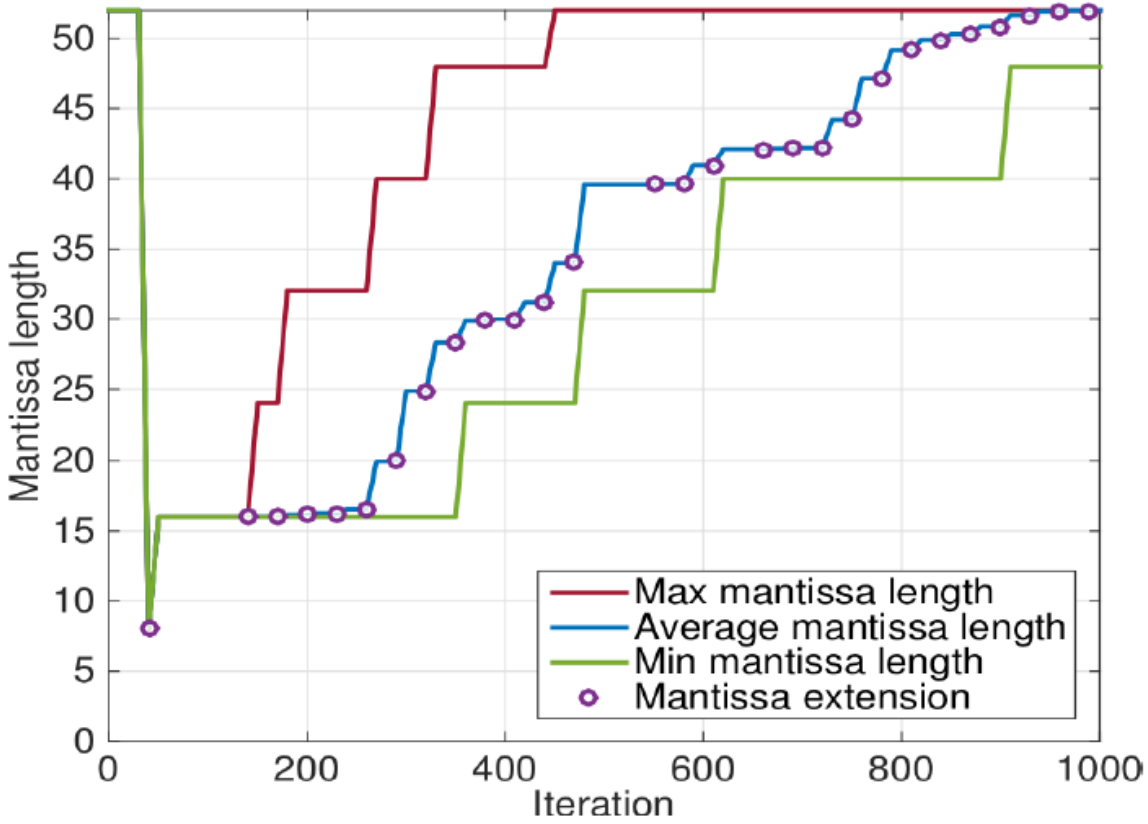
A = LU	O(n^3) flops	SINGLE
x = L\(U\b)	O(n^2) flops	SINGLE
r = b-A*x	O(n^2) flops	DOUBLE
while || r || not small enough	O(n^2) flops	DOUBLE
z = L\(U\r)	O(n^2) flops	SINGLE
x = x + z	O(n) flops	DOUBLE
r = r-A*x	O(n^2) flops	DOUBLE
end		

- On conventional hardware, SP is twice as fast as DP. In addition, it can save energy by reducing data communication to half
- On GPUs, the difference between SP and DP can be as large as 26x
- NVIDIA “Pascal” GPUs support half (16-bit) precision

A Recipe to Saving Energy
5. Approximate computing

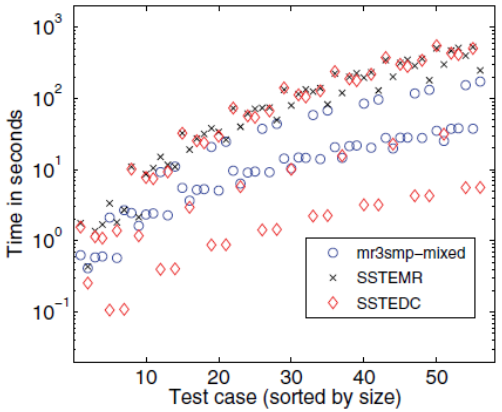
- In Jacobi-based solvers for linear systems, precision can be adapted component-wise: FPGAs!



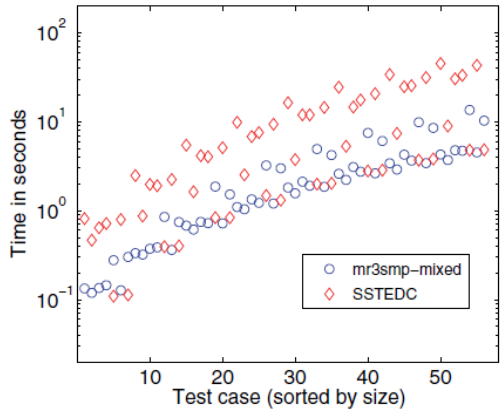


A Recipe to Saving Energy
5. Approximate computing

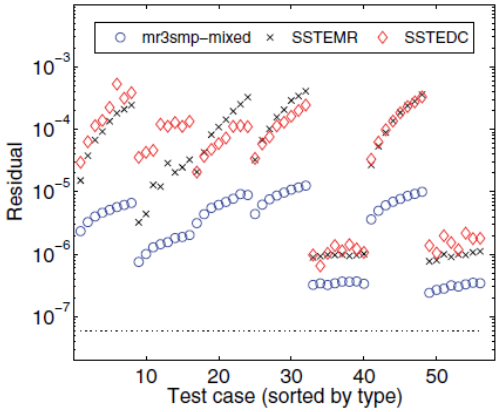
- For MR^3 dense eigensolver, extended precision in a few suboperations is key to calculate "correct" solution at high speed



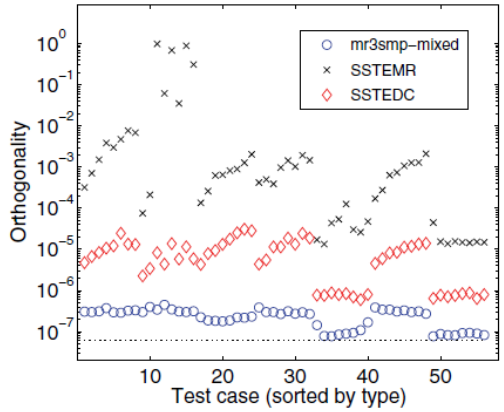
(a) Execution time: sequential.



(b) Execution time: multithreaded.



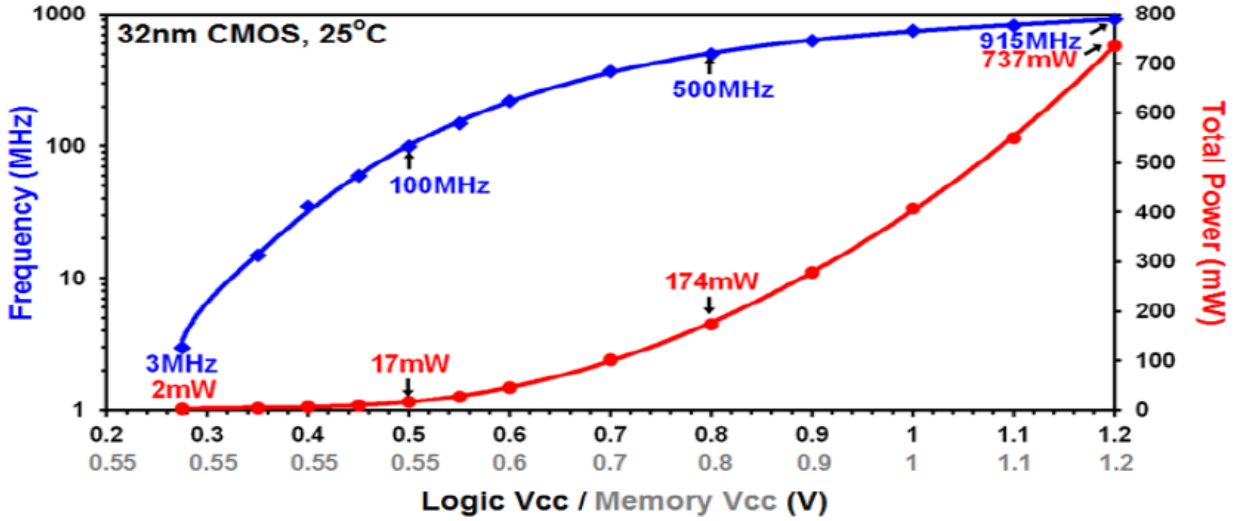
(c) Largest residual norm.



(d) Orthogonality.

A Recipe to Saving Energy
6. NTVC

- Dynamic power is proportional to *V^2 f*



  - Undervolting: Reduce *V*, but maintain *f*
  - NTVC: Reduce *(V,f)* in the same proportion
    - For some applications, reducing *f* does not impact performance
    - For others, a linear decrease in performance is expected

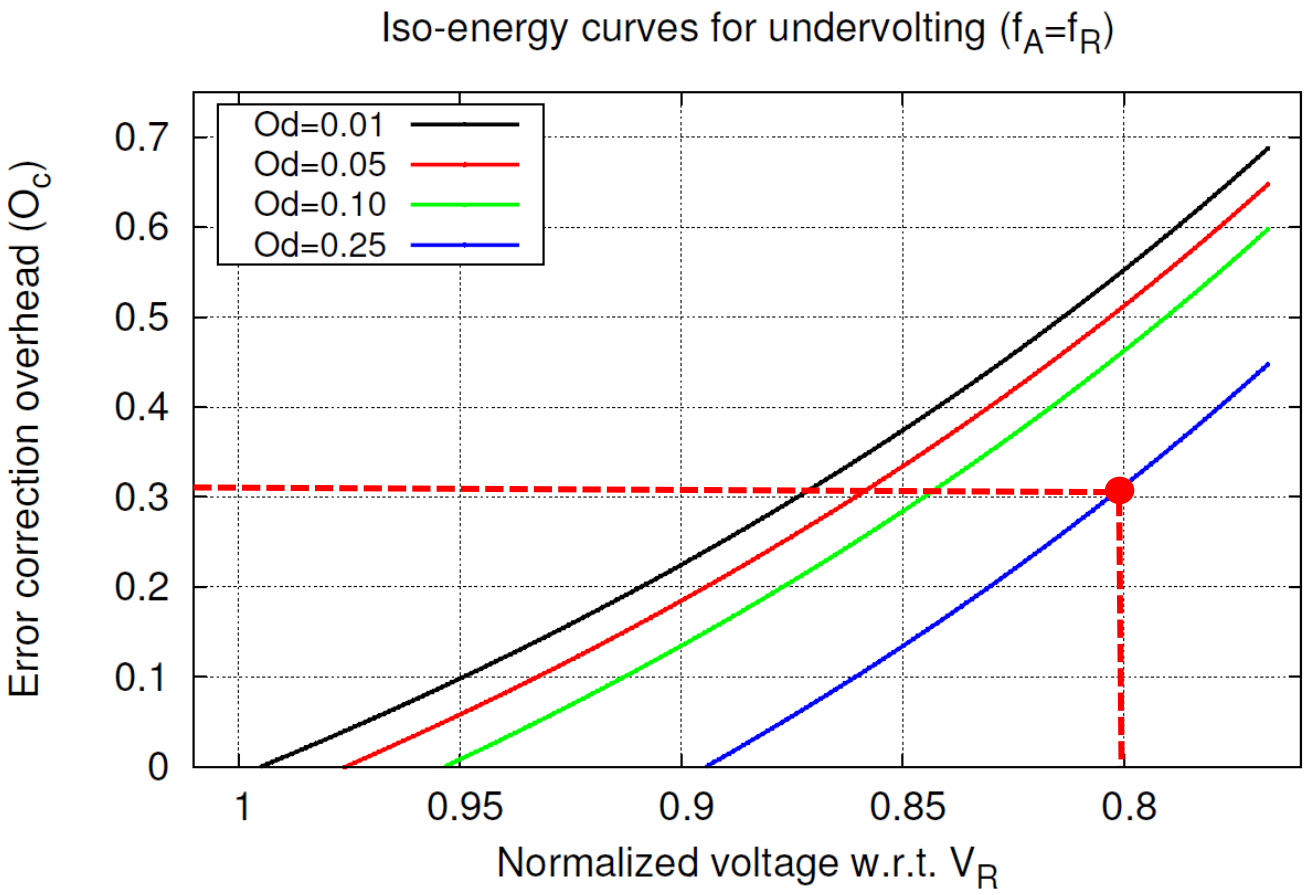
A Recipe to Saving Energy
6. NTVC

- Operating near the voltage threshold may introduce errors
- Integrate fault-tolerance into software (applications)
  - Check-point + restart
  - Modular redundancy
  - Algorithmic-based fault tolerance (ABFT)
- What is the energy trade-off?
  - Move from error-free (V_R, f_R) → *error-prone* (V_A, f_A)
  - Detection overhead $\mathcal{O}_d$ (even if no errors occur)
  - Correction overhead $\mathcal{O}_c$ (proportional to error rate)

A Recipe to Saving Energy
6. NTVC

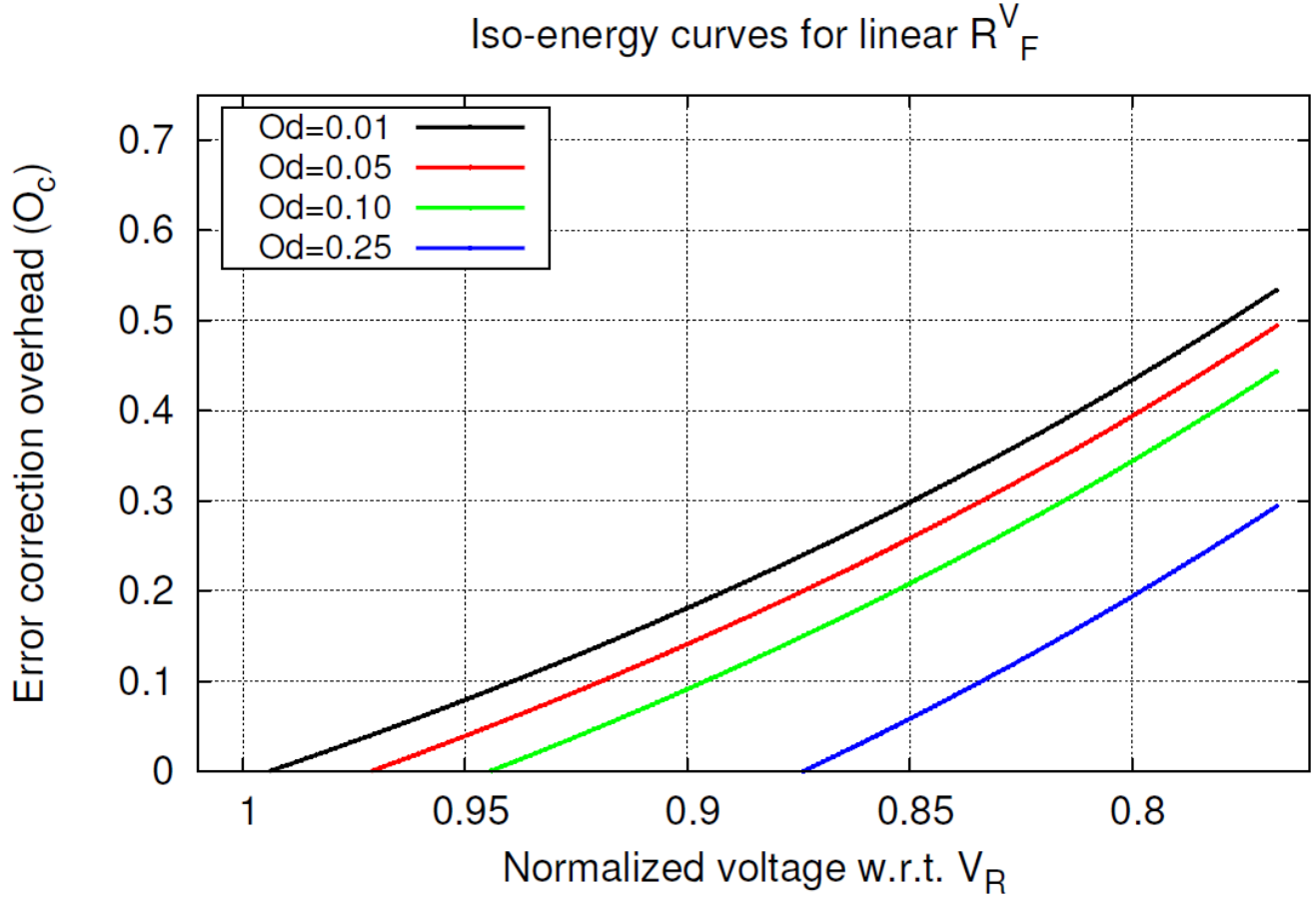
- For undervolting, any DLA and architecture:

$$\mathcal{O}_c^{iso} = \left(\frac{V_R}{V_A}\right)^2 - (1 + \mathcal{O}_d)$$



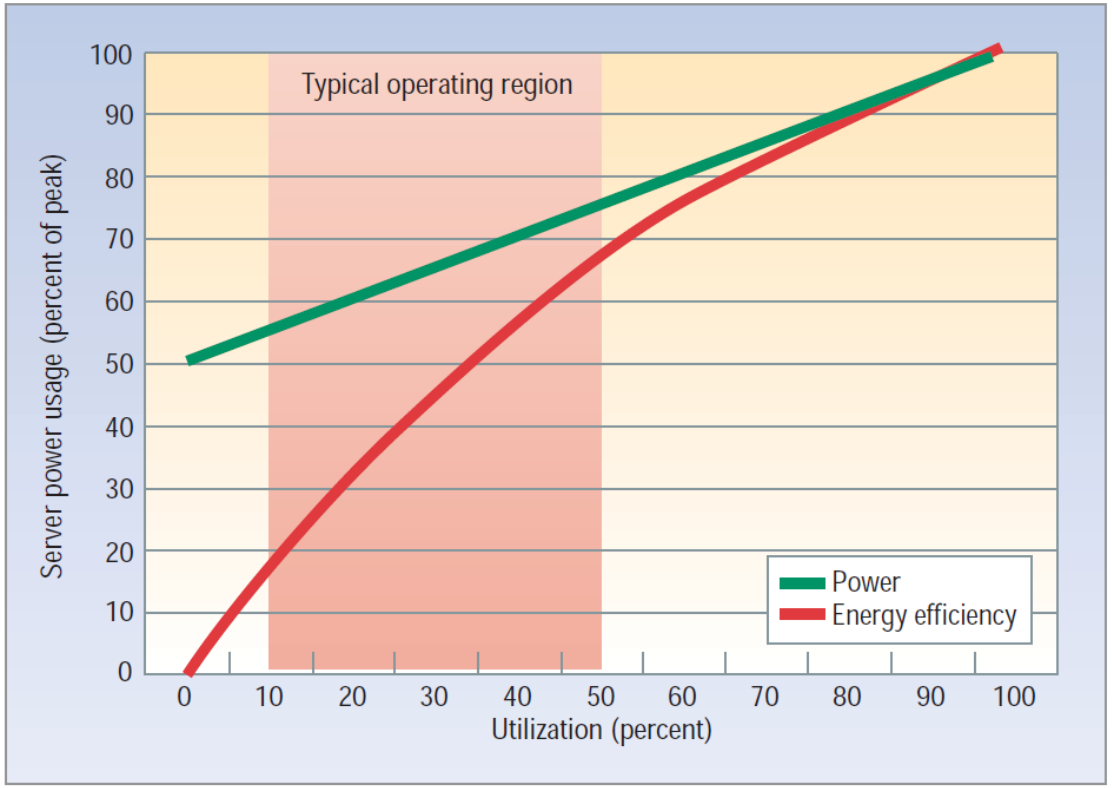
A Recipe to Saving Energy
6. NTVC

- For NTVC, assuming linear relation between performance and *f*:
  - Same relation for compute-bound kernels
  - …but iso-energy more difficult for memory-bound kernels



A Recipe to Saving Energy
7. Energy-proportional hardware

- Power consumption should be proportional to use of resources



"The case for energy-proportional computing". L. A. Barroso, U. Hölzle, IEEE Computer, 2007

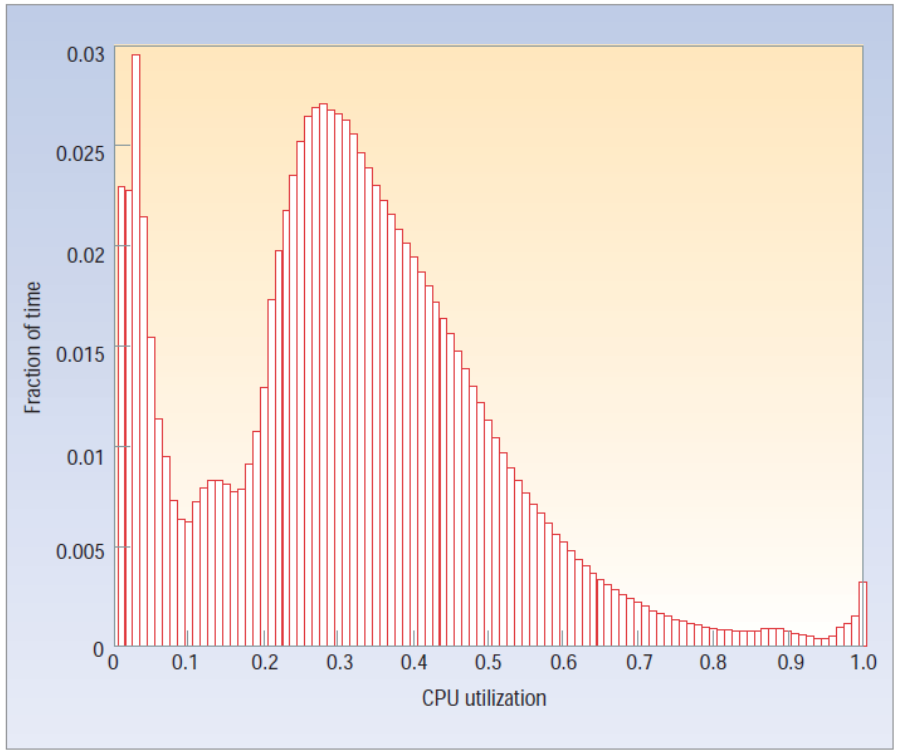
A Recipe to Saving Energy
7. Energy-proportional hardware

- Stencil computation on Intel Xeon E5-2620 (2x6 cores)

#cores	Power (W)
1	135.1
2	144.1
3	150.2
4	155.9
5	162.8
6	169.4
7	171.4
8	180.0
9	186.0
10	188.4
11	190.6
12	195.1

A Recipe to Saving Energy
8. Virtualization of HPC resources

- Servers seldom operate at 100% of their maximum utilization level

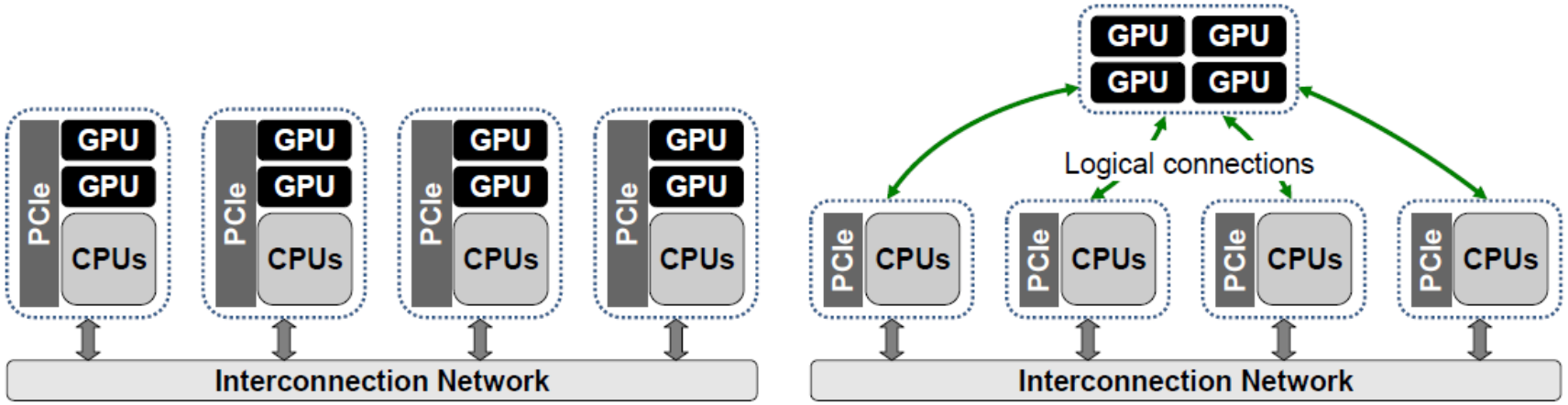


Average CPU utilization of more than 5,000 servers during a six-month period

"The case for energy-proportional computing". L. A. Barroso, U. Hölzle, IEEE Computer, 2007

A Recipe to Saving Energy
8. Virtualization of HPC resources

- Same for GPUs in a cluster:
  - Not all applications can run on a GPU
  - Not all parts of application's code benefit from a GPU
- Virtualization of accelerators



(a) Traditional GPU-equipped cluster. (b) Virtual GPU pool

A Recipe to Saving Energy
8. Virtualization of HPC resources

- Transparent view of remote GPUs
- …but slower performance for some applications

Outline

- A recipe to saving energy: Optimize performance!!!

    1. Choose the “right” hardware
    2. Dynamic Voltage-Frequency Scaling (DVFS)
    3. Dynamic Concurrency Throttling (DCT)
    4. Avoid polling
    5. Approximate computing/adaptive precisión
    6. Near Threshold Voltage Computing (NTVC)
    7. Energy-proportional hardware
    8. Virtualization of HPC resources

- … but other “chefs” may propose a different recipe

Energy Efficiency in Scientific Computing

Enrique S. QUINTANA-ORTÍ
Professor of Computer Architecture
Group leader High Performance Computing & Architectures (HPC&A) group
http://www.uji.es/~quintana