# Analysis and Optimization of Power Consumption in the Iterative Solution of Sparse Linear Systems on Multi-core and Many-core Platforms

H. Anzt, V. Heuveline
Karlsruhe Institute of Technology, Germany

J.I. Aliaga, M. Castillo, J.C. Fernández, R. Mayo, E.S. Quintana-Ortí
Universidad Jaume I, Spain

# Motivation

- Reduce energy consumption!
  - Costs over the lifetime of an HPC facility in the range of acquisition costs
  - Produces carbon dioxide, a risk for the health and the environment
  - Produces heat which reduces hardware reliability
  - It gave us a reason to meet in nice Orlando ;-)

## Personal view

- Hardware features mechanisms and modes to save energy
- Software, in particular, scientific apps are in general power oblivious

## Target scientific application

- Sparse linear systems

$$Ax = b$$

  arise in many apps. that involve PDEs modeling physical, chemical or economical processes

- Low-cost iterative Krylov-based solvers for large-scale systems: $A$ s.p.d. $\rightarrow$ Conjugate Gradient (CG), Preconditioned CG (PCG)

## CG (Matlab)

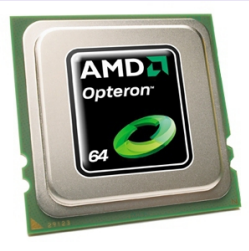
```
1 function [x] = cg(A,b,x,tol)   % BLAS    SBLAS    Arch.
2                                %-------------------------
3   r=b-A*x;                     %         spmv     CPU/GPU
4   p=r;
5   rsold=r'*r;                  % dot               CPU
6
7   for i=1:size(A,1)
8     Ap=A*p;                    %         spmv     CPU/GPU
9     alpha=rsold/(p'*Ap);       % dot               CPU
10    x=x+alpha*p;               % axpy              CPU
11    r=r-alpha*Ap;              % axpy              CPU
12    rsnew=r'*r;                % dot               CPU
13    if sqrt(rsnew)<tol
14      break;
15    end
16    p=r+rsnew/rsold*p;         % axpy              CPU
17    rsold=rsnew;
18  end
19 end
```

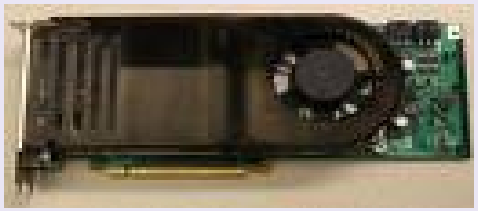
# Outline

# 2. Experimental setup

## Hardware platform

- AMD Opteron 6128 (8 cores)@2.0 GHz with 24 GBytes of RAM
- NVIDIA Tesla C1060 (240 cores). Disconnected during CPU-only experiments!
- PCI-Express (16×)

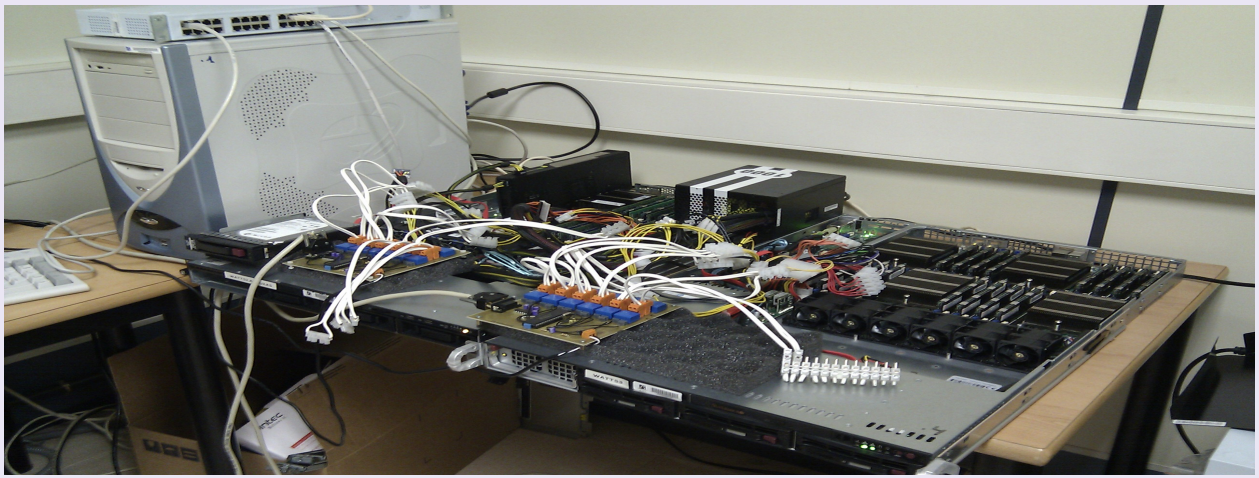


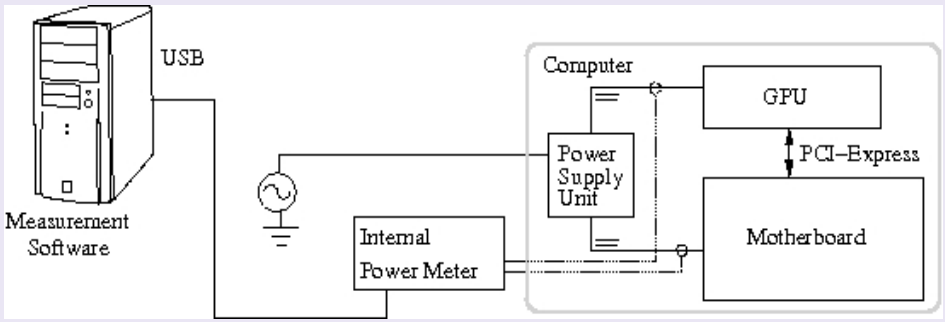



## Software implementation of CG, PCG

- AMD: Intel MKL (11.1) for BLAS-1 and own implementation of `spmv`
- NVIDIA: CUBLAS (3.0) and implementation of `spmv` based on Garland and Bell's approach
- `gcc -O3` (4.4.3) and `nvcc` (3.2)

# Measurement setup






- ASIC with sampling frequency of 25 Hz

## Linear systems

| Matrix name | Size ($n$) | Nonzeros ($nnz$) |
|---|---|---|
| A318 | 32,157,432 | 224,495,280 |
| APACHE2 | 715,176 | 4,817,870 |
| AUDIKW_1 | 943,695 | 77,651,847 |
| BONES10 | 914,898 | 40,878,708 |
| ECOLOGY2 | 999,999 | 4,995,991 |
| G3_CIRCUIT | 1,585,478 | 7,660,826 |
| LDOOR | 952,203 | 42,493,817 |
| ND24K | 72,000 | 28,715,634 |

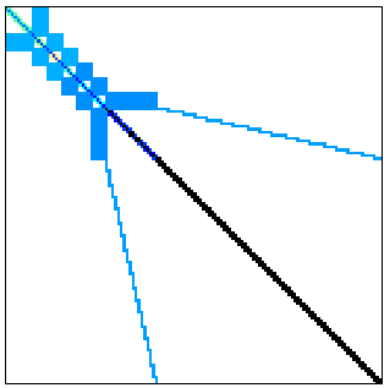
## Solvers $Ax = b$

- Iterative: $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_n \approx x$
- Stopping criterion: $\varepsilon = 10^{-10}\|r_0\|_2$
- Initial solution: $x_0 \equiv 0$

# 3. Analysis of power consumption

## Experiment #1

- Power consumption of CG and PCG on CPU (1T, 2T, 4T, 8T on 1, 2, 4, 8 cores) and hybrid CPU (4T)+GPU
- G3_CIRCUIT (moderate dimension, complex sparsity pattern)



## CG method

| Hardware | # iter | Time [s] | Energy consumption [Wh] | | |
|---|---|---|---|---|---|
| | | | Chipset | GPU | Total |
| CPU 4T | 21,424 | 1,076.97 | 42.18 | - | 42.18 |
| GPU 4T | 21,467 | 198.43 | 8.04 | 3.44 | 11.48 |

- Hybrid CPU-GPU code clearly outperforms CPU one in both performance ($5\times$) and energy ($4\times$)
- Energy gap mostly from reduction in execution time:

| CPU 4 T | GPU 4 T |
|---|---|
| $\frac{42.18}{1,076.97} \cdot 3,600 = 140.0$ W | $\frac{11.48}{198.43} \cdot 3,600 = 208.2$ W |

# PCG method (Jacobi preconditioner)

| Hardware | # iter | Time [s] | Energy consumption [Wh] | | |
|---|---|---|---|---|---|
| | | | Chipset | GPU | Total |
| CPU 4T | 4,613 | 348.79 | 13.31 | - | 13.31 |
| GPU 4T | 4,613 | 46.28 | 1.89 | 0.83 | 2.72 |

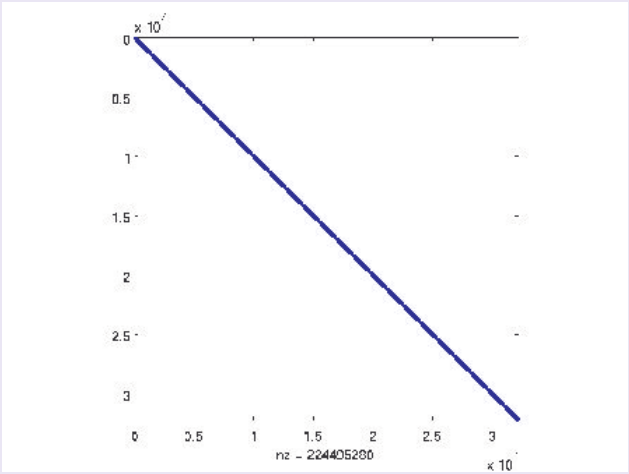
- Important reduction in #iterations: 21,424 $\rightarrow$ 4,613
- Time/iteration and energy/iteration not significantly increased (preconditioning this matrix only requires diagonal scaling):

| CG GPU 4 T | PCG GPU 4 T |
|---|---|
| $\frac{198.43}{21,467} = 0.0092$ s/iter | $\frac{46.28}{4,613} = 0.0100$ s/iter |
| $\frac{11.48}{21,467} = 5.34 \cdot 10^{-4}$ Wh/iter | $\frac{2.72}{4,613} = 5.89 \cdot 10^{-4}$ Wh/iter |

# 4. DVFS

## Experiment #2

- For memory-bounded operations, in general a decrease of the processor operation frequency can yield energy savings
  - Memory-bounded or I/O-bounded?
  - Decreasing processor frequency impacts memory latency?
- The sparse matrix-vector product is indeed memory-bounded: $2nnz$ flops vs. $nnz$ memops
- AMD Opteron 6128: 800 MHz – 2.0 GHz
- $\mathrm{A318}$ (large size to match powermeter sampling rate)



## CG method

| Hardware | Freq. [MHz] | Time [s] | Power/Energy consumption Chipset [Avg. W] | GPU [Avg. W] | Total [Wh] |
|---|---|---|---|---|---|
| CPU 4T | 2,000 | 1441.78 | 123.99 | - | 49.66 |
| CPU 4T | 800 | 1674.62 | 108.11 | - | 50.29 |
| GPU 4T | 2,000 | 253.22 | 149.04 | 61.89 | 14.84 |
| GPU 4T | 800 | 254.25 | 138.50 | 61.45 | 14.12 |

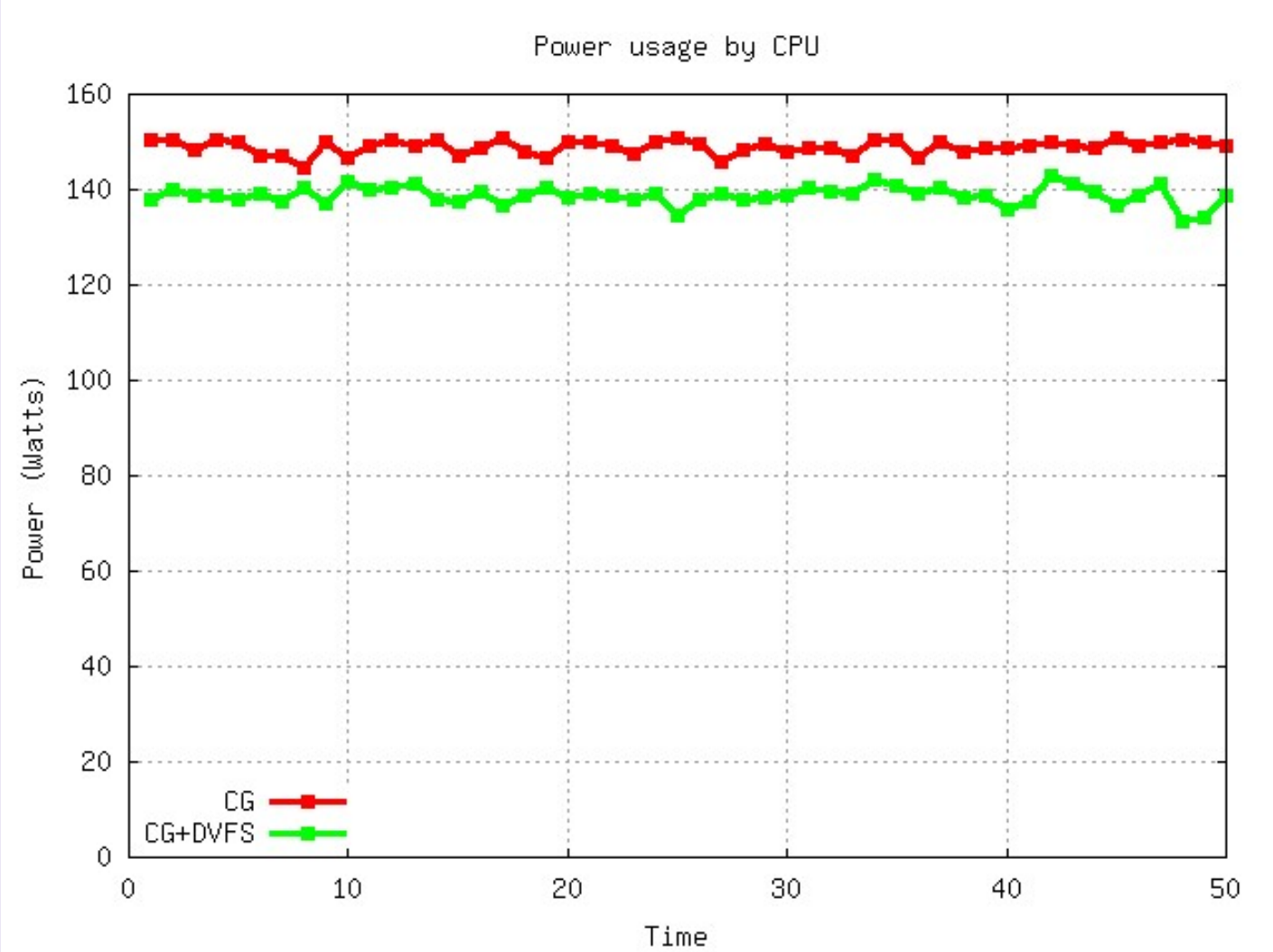
- For the CPU solver, lowering the processor frequency increases the execution time, which blurs savings in power consumption
- For the hybrid CPU-GPU solver, as the computationally intensive parts are executed on the GPU, lowering the frequency yields some energy savings... Why not larger?

## Experiment #3

- GPU and CPU operate in asynchronous mode but... when the GPU is executing a kernel, and the CPU encounters a call to a second kernel, it enters into a *polling* loop
- In the polling state, the power usage of the CPU is as high as that of a fully-loaded processor!
- Solution: use DVFS (actually, static VFS) to adjust CPU frequency while in the polling loop
- Alternatives:

  (i) Plain solver
  (ii) Solver + DVFS during GPU execution

# Power-friendly CPU modes



**CG method:** Energy consumption of chipset+GPU

| matrix | Energy consumption [Wh] | | improvement [%] |
|---|---|---|---|
| | (i) | (ii) | (i)→(ii) |
| A318 | 14.84 | 14.12 | 5.1 |
| APACHE2 | 1.98 | 1.99 | -0.5 |
| AUDIKW_1 | no convergence | | - |
| BONES10 | no convergence | | - |
| ECOLOGY2 | 2.30 | 2.27 | -1.3 |
| G3_CIRCUIT | 11.48 | 11.11 | 3.3 |
| LDOOR | no convergence | | - |
| N24K | 26.43 | 25.42 | 3.97 |

- A moderate gain, in some cases a loss...

**PCG method:** Energy consumption of chipset+GPU

| matrix | Energy consumption [Wh] | | improvement [%] |
|---|---|---|---|
| | (i) | (ii) | (i)→(ii) |
| A318 | 14.84 | 14.12 | 5.1 |
| APACHE2 | 1.75 | 1.76 | -0.6 |
| AUDIKW_1 | 47.98 | 38.15 | 5.2 |
| BONES10 | 157.32 | 150.16 | 4.8 |
| ECOLOGY2 | 2.51 | 2.45 | 2.4 |
| G3_CIRCUIT | 2.71 | 2.38 | 3.0 |
| LDOOR | 43.22 | 41.18 | 5.0 |
| N24K | 34.62 | 32.97 | 5.0 |

- Moderate but more consistent gain... Why not larger?

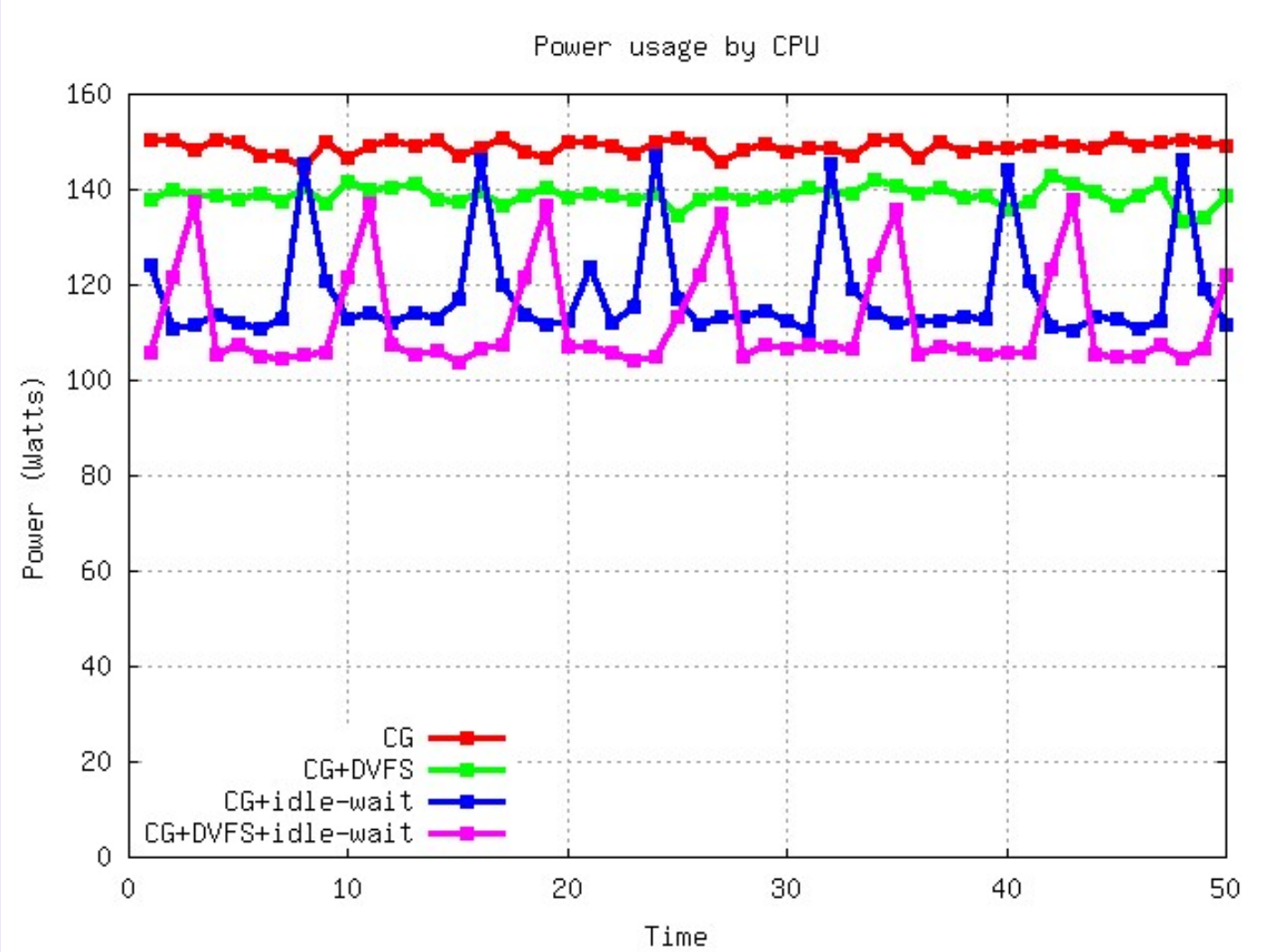
# 5. Idle-wait

## Experiment #3

- Solution: set the CPU to "sleep" during the execution of the GPU kernels: Execution time of GPU `spmv` can be measured and accurately adjusted
- Use of `nanosleep()` function from `sys/time.h`
- Alternatives:

  (i) Plain solver
  (ii) Solver + DVFS during GPU execution
  (iii) Solver + DVFS + idle-wait during GPU execution

# Power-friendly CPU modes



## **CG method:** Energy consumption of chipset+GPU

| matrix | energy consumption [Wh] | | | improvement [%] | |
|---|---|---|---|---|---|
| | (i) | (ii) | (iii) | (i)→(ii) | (i)→(iii) |
| A318 | 14.84 | 14.12 | 12.18 | 5.1 | 21.8 |
| APACHE2 | 1.98 | 1.99 | 1.82 | -0.5 | 8.8 |
| AUDIKW_1 | no convergence | | | - | - |
| BONES10 | no convergence | | | - | - |
| ECOLOGY2 | 2.30 | 2.27 | 2.09 | -1.3 | 10.0 |
| G3_CIRCUIT | 11.48 | 11.11 | 10.10 | 3.3 | 13.7 |
| LDOOR | no convergence | | | - | - |
| N24K | 26.43 | 25.42 | 21.17 | 3.97 | 24.8 |

**PCG method:** Energy consumption of chipset+GPU

| matrix | energy consumption [Wh] | | | improvement [%] | |
|---|---|---|---|---|---|
| | (i) | (ii) | (iii) | (i)→(ii) | (i)→(iii) |
| A318 | 14.84 | 14.12 | 12.18 | 5.1 | 21.8 |
| APACHE2 | 1.75 | 1.76 | 1.64 | -0.6 | 6.7 |
| AUDIKW_1 | 47.98 | 45.61 | 38.15 | 5.2 | 25.8 |
| BONES10 | 157.32 | 150.16 | 125.78 | 4.8 | 25.1 |
| ECOLOGY2 | 2.51 | 2.45 | 2.29 | 2.4 | 9.6 |
| G3_CIRCUIT | 2.71 | 2.63 | 2.38 | 3.0 | 13.9 |
| LDOOR | 43.22 | 41.18 | 34.79 | 5.0 | 24.2 |
| N24K | 34.62 | 32.97 | 27.64 | 5.0 | 25.3 |

# 5. Conclusions

- The concurrency of `spmv` enables the efficient usage of GPUs, that render important savings in execution time and energy consumption
- For memory-bounded operations, DVFS can potentially render energy savings...
  but the busy-wait of the host system during the kernel calls still consumes about 80 % of full-demand power
- The use of GPU-accelerated HPC-systems combined with power-saving techniques leads to more reduced energy consumption of all test problems without impacting the performance

- IGCC 2012...

# 5. Conclusions

- The concurrency of `spmv` enables the efficient usage of GPUs, that render important savings in execution time and energy consumption
- For memory-bounded operations, DVFS can potentially render energy savings...
  but the busy-wait of the host system during the kernel calls still consumes about 80 % of full-demand power
- The use of GPU-accelerated HPC-systems combined with power-saving techniques leads to more reduced energy consumption of all test problems without impacting the performance

- IGCC 2012 Please, in a warmer, more power-friendly hotel

# Analysis and Optimization of Power Consumption in the Iterative Solution of Sparse Linear Systems on Multi-core and Many-core Platforms

H. Anzt, V. Heuveline
Karlsruhe Institute of Technology, Germany

J.I. Aliaga, M. Castillo, J.C. Fernández, R. Mayo, E.S. Quintana-Ortí
Universidad Jaume I, Spain