Karlsruhe Institute of Technology

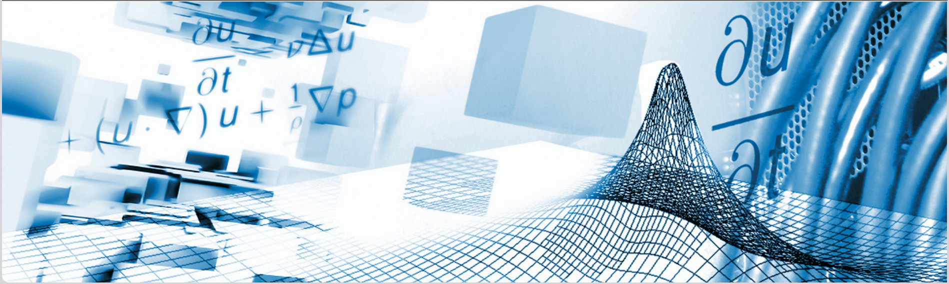
Energy-Efficient High-Performance-Computing

Hardware Aware Numerics for Scientific Simulations

Hartwig Anzt

16/09/2011

Engineering Mathematics and Computing Lab (EMCL)



Motivation

Karlsruhe Institute of Technology

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

Current state

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

Motivation

Karlsruhe Institute of Technology

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: Conjugate Gradient (CG), Preconditioned CG (PCG), GMRES and P-GMRES

EMCL

Experimental setup

Karlsruhe Institute of Technology

Hardware platform

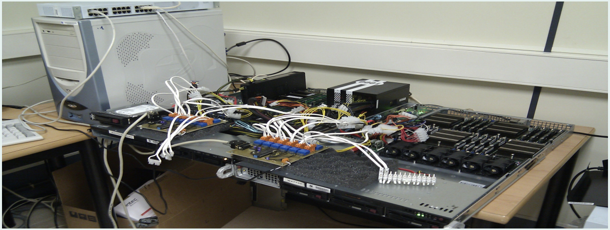
- AMD Opteron 6128 (8 cores)@2.0 GHz with 24 GBytes of RAM
- NVIDIA Tesla C1060 (240 cores). Disconnected during CPU-only experiments!
- PCI-Express (16×)

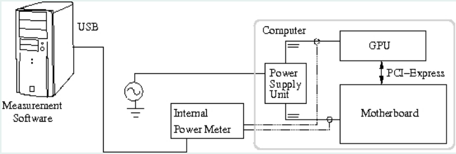
Software implementation of CG, PCG

- 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)

Experimental setup

Measurement setup





- ASIC with sampling frequency of 25 Hz

Experimental setup

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

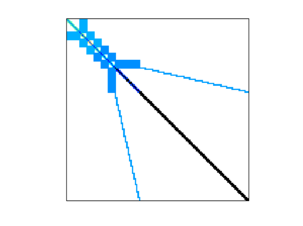
Solvers $Ax = b$

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

Analysis of power consumption

Experiment #1

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



CG method

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

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

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

PCG method (Jacobi preconditioner)

Karlsruhe Institute of Technology

Hardware	# iter	Time [s]	Energy consumption [Wh] Chipset	GPU	Total
CPU 4c	4,613	348.79	13.31	-	13.31
GPU 4c	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 c	PCG GPU 4 c
$\frac{198.43}{21,467} = 0.0092 \text{ s/iter}$	$\frac{46.28}{4,613} = 0.0100 \text{ s/iter}$
$\frac{11.48}{21,467} = 5.34 \cdot 10^{-4} \text{ Wh/iter}$	$\frac{2.72}{4,613} = 5.89 \cdot 10^{-4} \text{ Wh/iter}$

DVFS

Experiment #2

- In general, for memory-bounded operations 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
- 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 4c	2,000	1441.78	123.99	-	49.66
CPU 4c	800	1674.62	108.11	-	50.29
GPU 4c	2,000	253.22	149.04	61.89	14.84
GPU 4c	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?

Problems with DVFS

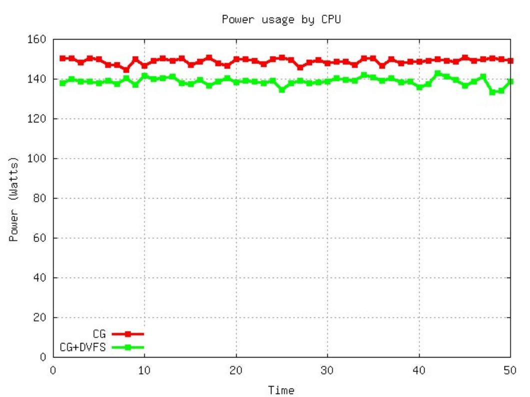
Karlsruhe Institute of Technology

- CPU computations slower when using DVFS
- GPU and CPU operate in asynchronous mode but for GPU kernel calls, the CPU is set into *busy wait* which is very energy-inefficient
- Alternatives:
  - (i) Plain solver
  - (ii) Solver + DVFS during GPU execution

DVFS

Karlsruhe Institute of Technology

Power-friendly CPU modes



EMCL

CG: Energy of chipset+GPU

Karlsruhe Institute of Technology

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: Energy of chipset+GPU

Karlsruhe Institute of Technology

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

Idle-wait

Karlsruhe Institute of Technology

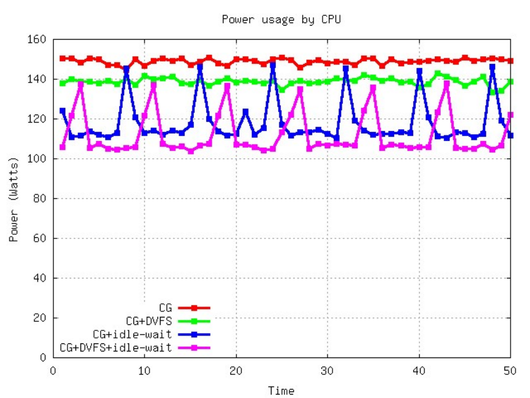
Experiment #3

- 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

Idle Wait

Power-friendly CPU modes



CG: Energy of chipset+GPU

Karlsruhe Institute of Technology

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: Energy of chipset+GPU

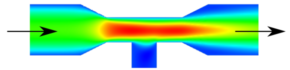
Karlsruhe Institute of Technology

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

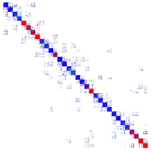
GMRES / Preconditioned GMRES

CFD-application

- Numerical simulation of fluid flow in ventury nozzle
- Navier-Stokes 2D
- Discretized with FEM & linearized
- large, sparse, unsymmetric & ill conditioned linear problem



Example	*n*	*nnz*
CFD1	395,009	3,544,321
CFD2	634,453	5,700,633
CFD3	1,019,967	9,182,401



GMRES / Preconditioned GMRES

Karlsruhe Institute of Technology

Ex.	Solver	Time	Energy
CFD1	GMRES	292.23	16.89
	P-GMRES	194.26	11.30
	gain (%)	33.5	33.1
CFD2	GMRES	1104.02	66.66
	P-GMRES	601.93	36.19
	gain (%)	45.5	45.7
CFD3	GMRES	3377.03	231.23
	P-GMRES	1391.16	86.12
	gain (%)	58.5	62.8

- Strong improvements by adding Jacobi Preconditioner
- Almost linear dependency between time and energy

Mixed-Iter Variants

Karlsruhe Institute of Technology

Ex.	Solver	Time	Energy Chipset	Energy GPU	Energy Total
CFD1	GMRES	292.23	12.00	4.89	16.89
	MPIR GMRES	154.30	6.34	2.71	9.05
	gain (%)	47.2	47.2	44.6	46.4
	P-GMRES	194.26	8.02	3.28	11.30
	MPIR P-GMRES	122.43	5.05	2.30	7.35
	gain (%)	37.0	37.0	29.9	35.0
CFD2	GMRES	1104.02	46.53	20.13	66.66
	MPIR GMRES	640.84	26.89	11.91	38.80
	gain (%)	42.0	42.2	40.8	41.8
	P-GMRES	601.93	25.19	11.00	36.19
	MPIR P-GMRES	416.42	17.47	8.46	25.93
	gain (%)	30.8	30.6	23.1	28.4
CFD3	GMRES	3777.03	160.98	70.25	231.23
	MPIR GMRES	2459.84	104.28	47.74	152.02
	gain (%)	34.9	35.2	32.0	34.2
	P-GMRES	1391.16	59.38	26.74	86.12
	MPIR P-GMRES	1520.79	64.47	28.15	92.62
	gain (%)	- 9.3	-8.6	-5.3	-7.5

Idle Wait for GMRES

Ex.	Solver	Energy (Wh) Plain	Idle-wait	gain (%)
CFD1	GMRES	16.88	15.65	7.31
	MPIR GMRES	12.38	11.62	8.75
	P-GMRES	9.01	8.22	6.09
	MPIR P-GMRES	7.35	6.61	9.99
CFD2	GMRES	66.66	62.03	6.95
	MPIR GMRES	36.19	33.83	10.22
	P-GMRES	38.79	34.83	6.51
	MPIR P-GMRES	25.94	23.39	9.80
CFD3	GMRES	231.23	217.48	5.95
	MPIR GMRES	86.12	80.88	8.70
	P-GMRES	152.02	138.80	6.08
	MPIR P-GMRES	92.62	84.51	8.75

- smaller improvements compared to CG / PCG why?

Idle Wait for GMRES

Karlsruhe Institute of Technology

Ex.	Solver	Energy (Wh)		
		Plain	Idle-wait	gain (%)
CFD1	GMRES	16.88	15.65	7.31
	MPIR GMRES	12.38	11.62	8.75
	P-GMRES	9.01	8.22	6.09
	MPIR P-GMRES	7.35	6.61	9.99
CFD2	GMRES	66.66	62.03	6.95
	MPIR GMRES	36.19	33.83	10.22
	P-GMRES	38.79	34.83	6.51
	MPIR P-GMRES	25.94	23.39	9.80
CFD3	GMRES	231.23	217.48	5.95
	MPIR GMRES	86.12	80.88	8.70
	P-GMRES	152.02	138.80	6.08
	MPIR P-GMRES	92.62	84.51	8.75

- smaller improvements compared to CG / PCG

GPU kernel calls "short" compared to iteration

$\rightarrow$ **merge computations into one kernel**

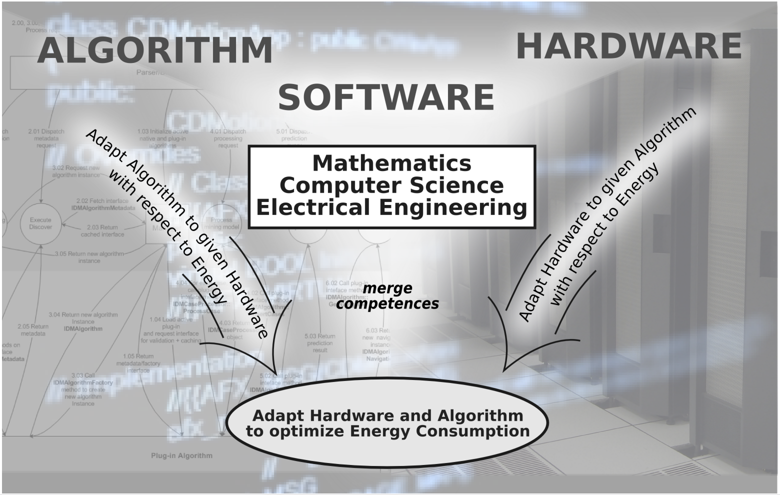
Conclusions

Karlsruhe Institute of Technology

- 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

- Merging more operations into a GPU kernel would trigger longer idle-wait & larger energy savings
- Multi-Iteration kernels would enhance energy efficiency furthermore

Future Work


ALGORITHM
HARDWARE
SOFTWARE
Mathematics
Computer Science
Electrical Engineering
Adapt Algorithm to given Hardware with respect to Energy
Adapt Hardware to given Algorithm with respect to Energy
merge competences
Adapt Hardware and Algorithm to optimize Energy Consumption

Karlsruhe Institute of Technology

Energy-Efficient High-Performance-Computing

Hardware Aware Numerics for Scientific Simulations

Hartwig Anzt

16/09/2011

Engineering Mathematics and Computing Lab (EMCL)