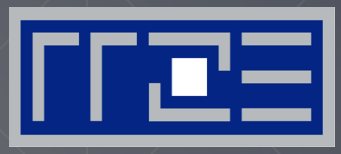
ERLANGEN REGIONAL COMPUTING CENTER



Making Sense of Performance Numbers

Georg Hager
Erlangen Regional Computing Center (RRZE)
Friedrich-Alexander-Universität Erlangen-Nürnberg

OpenMPCon 2018
Barcelona, 2018-09-25

Agenda

- Performance Engineering
- Analytic performance models
- A simple example: Sparse matrix-vector multiplication
- Another example: Sparse matrix-transpose vector multiplication
- An advanced chip-level model: ECM
- Yet another example: Composite modeling of a (P)CG solver

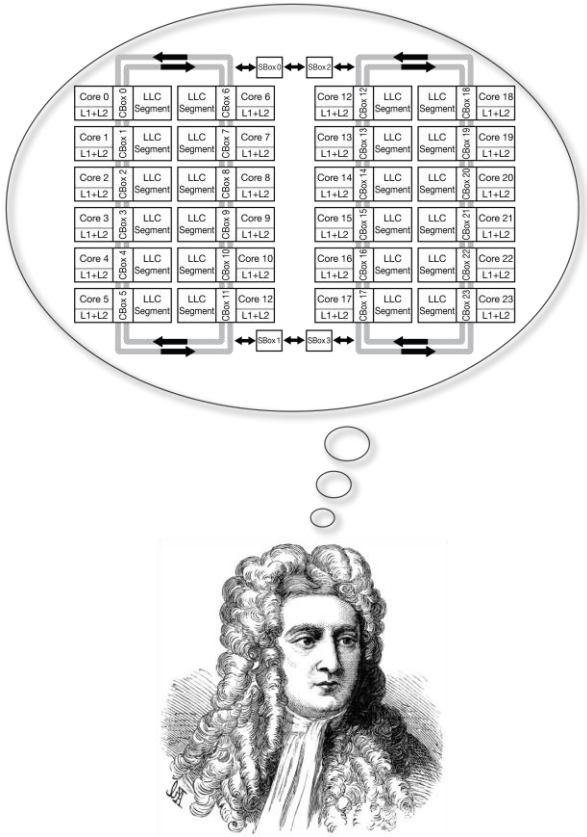
Performance Engineering

- Performance Engineering (PE): a structured process based on analytic (white-/gray-box) models to optimize/parallelize codes

- Basic questions addressed by analytic performance models
  - **What is the bottleneck?** → optimization technique
  - What is the next bottleneck? → performance potential of the optimization
  - **Am I done?** What about other hardware?
  - Impact of processor frequency and socket scalability
    → Appropriate execution parameters, energy-optimized operating point

- Engineering for Performance in High Performance Computing (Bill Gropp; PASC 2015): https://www.youtube.com/watch?v=sadfSARXSC0

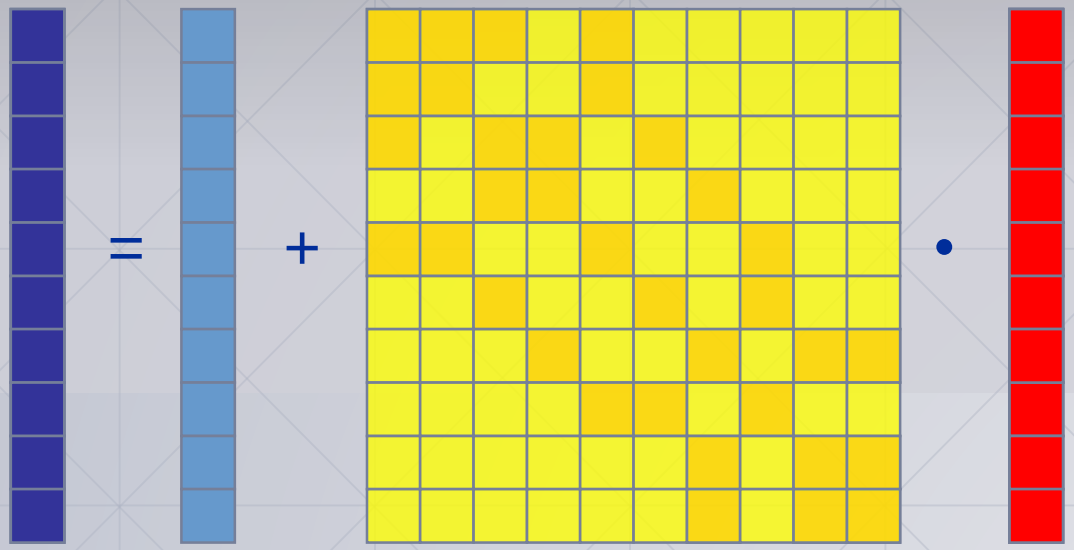
Motivation for analytic modeling



- Advantages of analytic models
  - Identification of universality
  - Identification of governing mechanisms
  - Insight via model nature
  - Insight via model failure

- Performance models
  - Microarchitecture analysis: Determine bottlenecks and influencing factors
  - Design space exploration: What would happen if resource *X* were improved?

NODE LEVEL PERFORMANCE MODEL FOR SPARSE MATRIX (TRANSPOSE) VECTOR MULTIPLY



SpMV – Roofline Model (Comp. Intensity)

Sparse MVM in
double precision
with CRS data storage:
(N_{nzr} : avg. non-zeros per row)

```
do i = 1,N_r
  do j = row_ptr(i), row_ptr(i+1) - 1
    C(i) = C(i) + val(j) * B(col_idx(j))
  enddo
enddo
```

Double precision computational intensity

$$I_{CRS}^{DP} = \frac{2}{8 + 4 + 8\alpha + 20/N_{nzr}} \frac{\text{flops}}{\text{byte}}$$

- α quantifies traffic for loading RHS (B)
  - $\alpha = 0$ → RHS is in cache
  - $\alpha = 1/N_{nzr}$ → RHS loaded once
  - $\alpha = 1$ → no cache
  - $\alpha > 1$ → Houston, we have a problem!
- "Expected" performance = b_S x I_{CRS} (Roofline model)

See
W. D. Gropp, D. K. Kaushik, D. E. Keyes, and B. F. Smith, "Towards realistic performance bounds for implicit CFD codes," in Proceedingsof Parallel CFD99. Elsevier, 1999, pp. 233–240.
G. Schubert et al. Parallel Processing Letters 21(3), 339-358 (2011). DOI: 10.1142/S0129626411000254
M. Kreutzer et al.: SIAM Journal on Scientific Computing 36(5), C401–C423 (2014). DOI: 10.1137/130930352,

SpMV – Quantifying RHS impact (α)

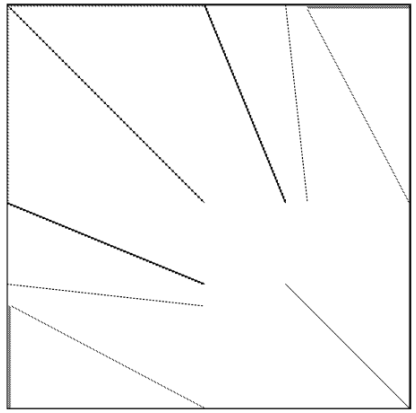
$$I_{CRS}^{DP} = \frac{2}{8+4+8\alpha+20/N_{nzr}} \frac{\text{flops}}{\text{byte}} = \frac{N_{nz} \cdot 2 \text{ flops}}{V_{meas}}$$

V_{meas} is measured overall memory data traffic (using, e.g., likwid)

$$\alpha = \frac{1}{4}\left(\frac{V_{meas}}{N_{nz} \cdot 2 \text{ bytes}} - 6 - \frac{10}{N_{nzr}}\right)$$

Example: kkt_power matrix from the UoF collection

- Intel Xeon Sandy Bridge (8c; 20 MB L3)
- $N_{nz} = 14.6 \cdot 10^6$, $N_{nzr} = 7.1$
- $V_{meas} \approx 258$ MB → $\alpha = 0.36$, $\alpha N_{nzr} = 2.5$
- → RHS is loaded 2.5 times from memory



- but: $\frac{I_{CRS}^{DP}(1/N_{nzr})}{I_{CRS}^{DP}(\alpha)} = 1.11$

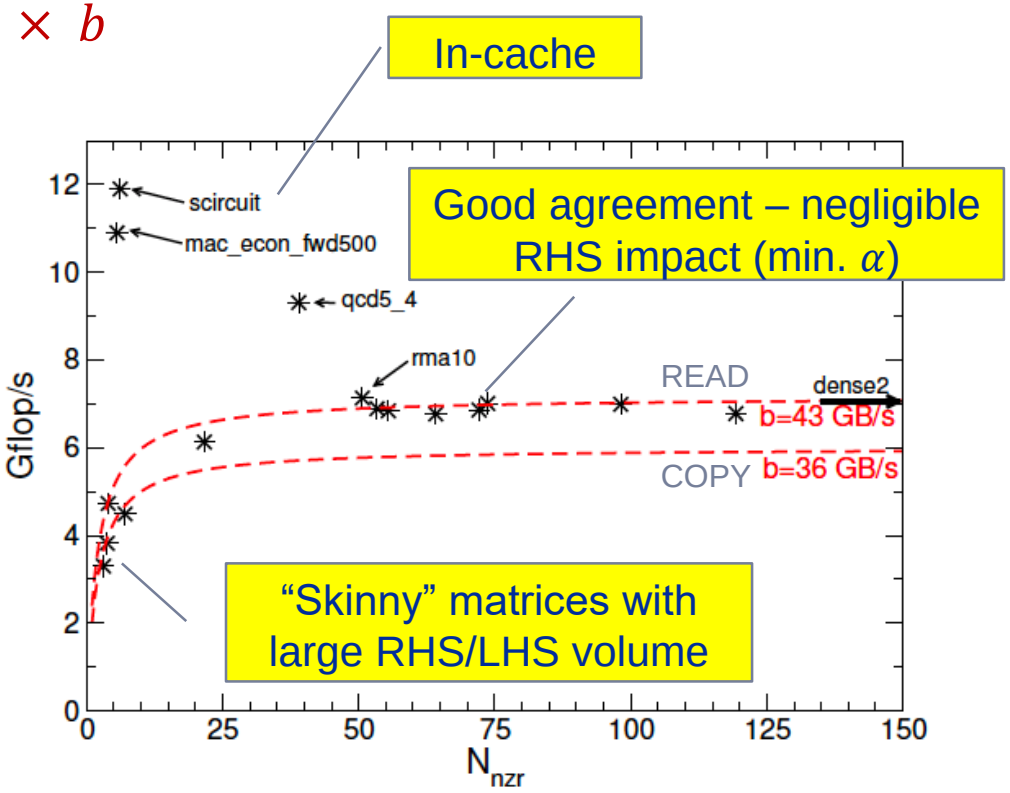
11% extra traffic → optimization potential!

SpMV – understanding performance

Intel Xeon E5-2680 (**8c**@2.7 GHz; 20 MB L3)

Assumption: $\alpha = {}^{1}/_{N_{nzr}} \rightarrow P = I_{CRS}^{DP} \times b$

#	Test case	N	N_{nz}	N_{nzr}
1	RM07R	381,689	37,464,962	98.16
2	kkt_power	2,063,494	14,612,663	7.08
3	Hamrle3	1,447,360	5,514,242	3.81
4	ML_Geer	1,504,002	110,879,972	73.72
5	pwtk	217,918	11,634,424	53.39
6	shipsec1	140,874	7,813,404	55.46
7	consph	83,334	6,010,480	72.13
8	pdb1HYS	36,417	4,344,765	119.31
9	cant	62,451	4,007,383	64.17
10	cop20k_A	121,192	2,624,331	21.65
11	rma10	46,835	2,374,001	50.69
12	mc2depi	525,825	2,100,225	3.99
13	qcd5_4	49,152	1,916,928	39.00
14	mac_econ_fwd500	206,500	1,273,389	6.17
15	scircuit	170,998	958,936	5.61
16	rail4284	4,284× 1,092,610	11,279,748	2,632.99
17	dense2	2,000	4,000,000	2,000.00
18	webbase-1M	1,000,005	3,105,536	3.11



4 corner case matrices from UoF collection

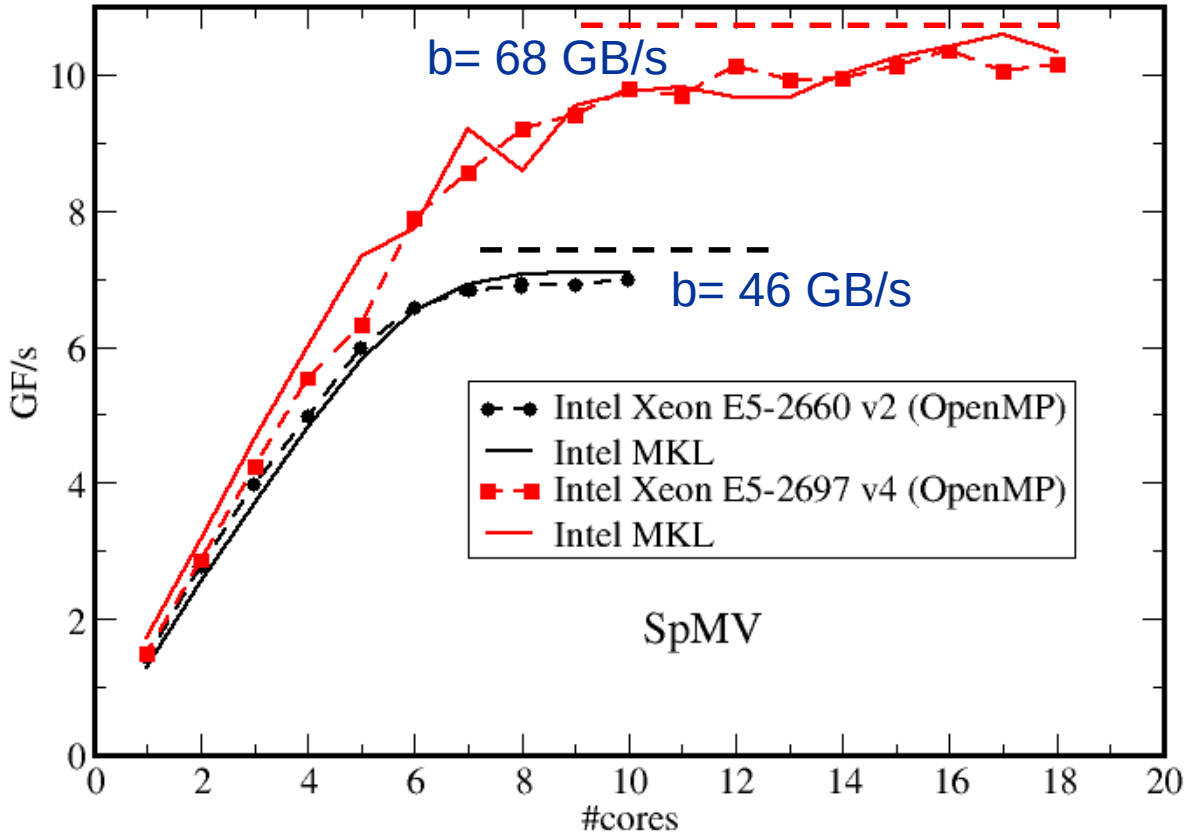
Williams collection http://www.nvidia.com/content/NV_Research/matrices.zip

M. Kreutzer et al.: A Unified Sparse Matrix Data Format for Efficient General Sparse Matrix-Vector Multiplication on Modern Processors with Wide SIMD Units. SIAM Journal on Scientific Computing 2014 36:5, C401-C423

SpMV – understanding performance

Socket scaling – more recent architectures

PWTK matrix (N_{nzr}= 53 & $\alpha = 1/N_{nzr}$ → $I_{CRS}^{DP} = \frac{2\,F}{12.5\,B}$ → $P = I_{CRS}^{DP} \times b$)



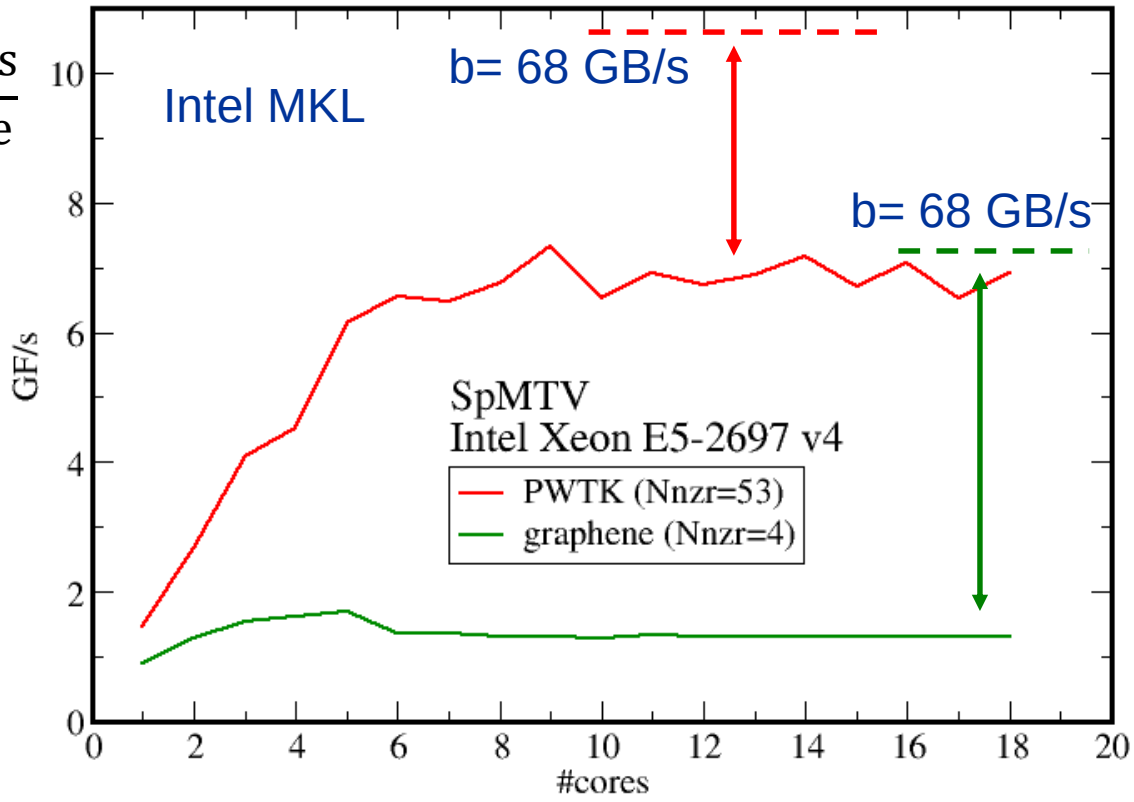
Model covers saturation regime – full chip!

SpMTransposeV – understanding performance

```
do i = 1,N_r
 do j = row_ptr(i), row_ptr(i+1) - 1
  C(col_idx(j)) = C(col_idx(j)) + val(j) * B(i)
 enddo
enddo
```

$$I_{CRS}^{DP} = \frac{2}{8+4+16\alpha+12/N_{nzr}} \frac{\text{flops}}{\text{byte}}$$

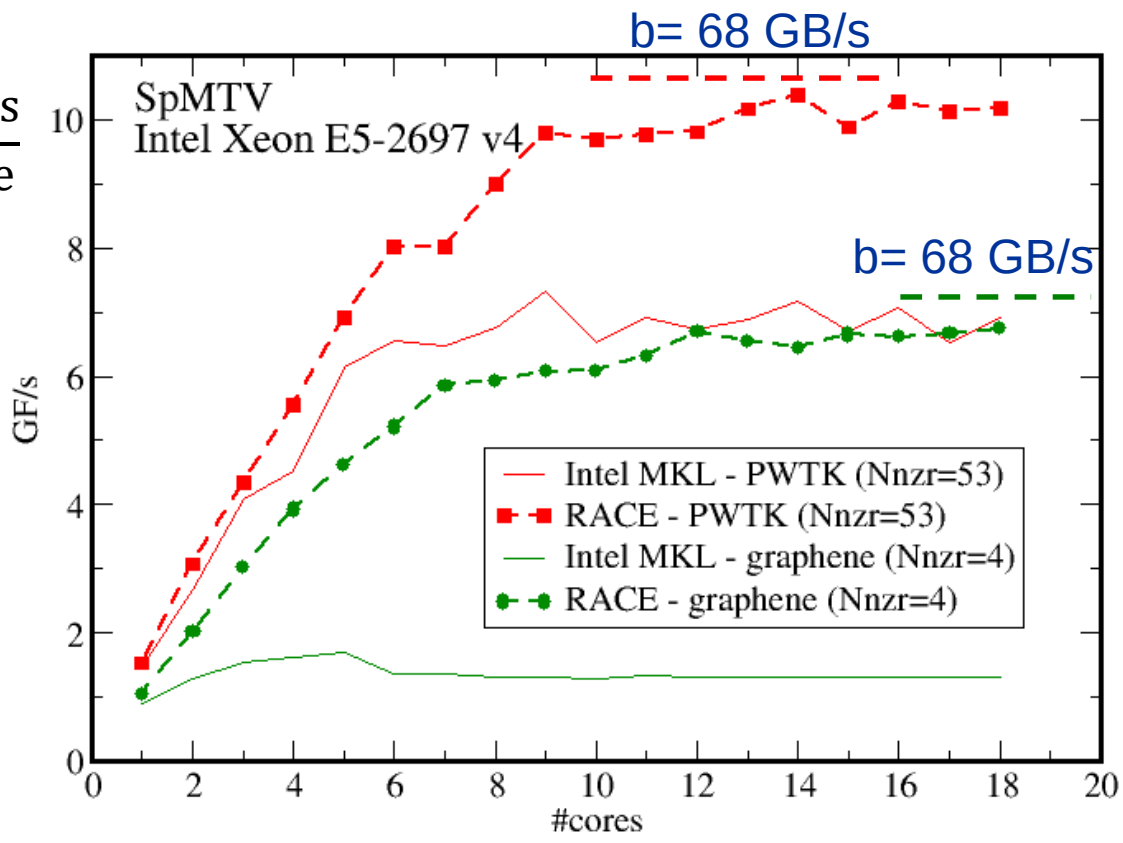
- Assume $\alpha = 1/N_{nzr}$
- Does the model fail?
- Parallelization ?!
  (write conflicts!)



SpMTV – understanding performance

```
do i = 1,Nr
 do j = row_ptr(i), row_ptr(i+1) - 1
  C(col_idx(j)) = C(col_idx(j)) + val(j) * B(i)
 enddo
enddo
```

$$I_{CRS}^{DP} = \frac{2}{8 + 4 + 16\alpha + 12/N_{nzr}} \frac{\text{flops}}{\text{byte}}$$



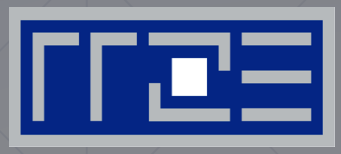
- New Parallelization Approach:

  "Recursive Algebaric Coloring Engine" (RACE)

  by C.L. Alappatt (FAU)

- Publication in preparation

THE ECM PERFORMANCE MODEL



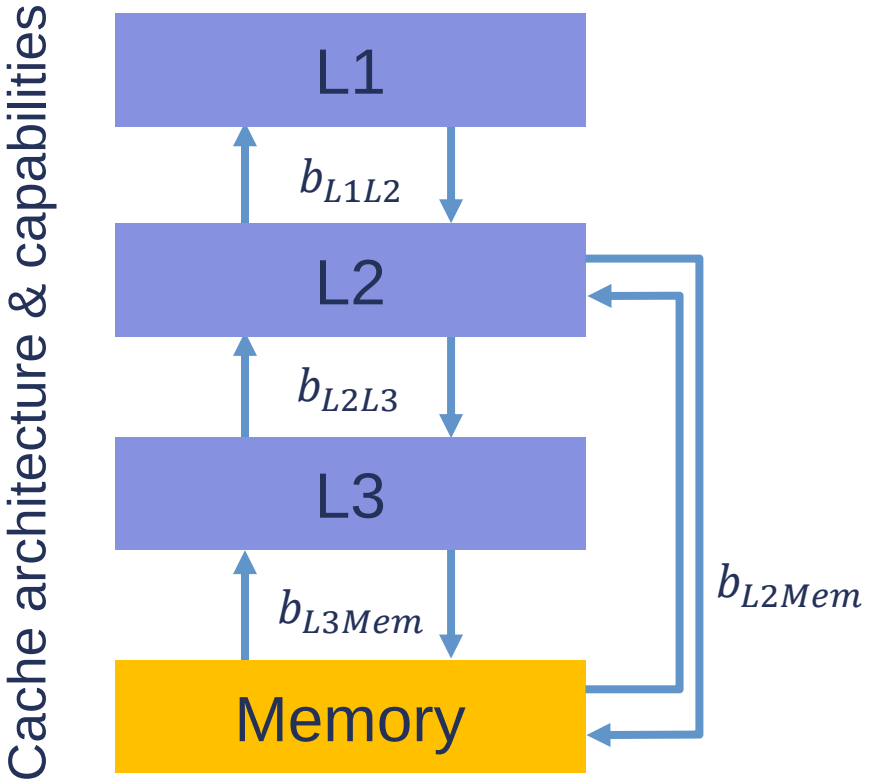
A quick walk-through

ECM model components: Data transfer times

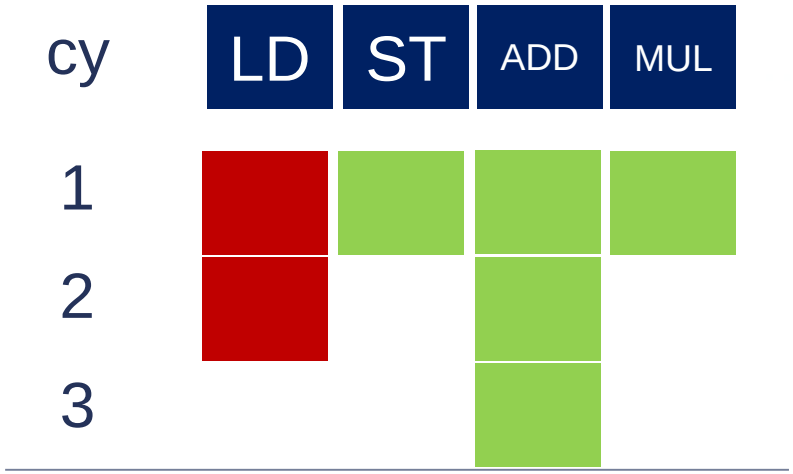
- Optimistic transfer times through mem hierarchy
- $T_i = \frac{V_i}{b_i}$
- Transfer time notation for some given loop kernel:

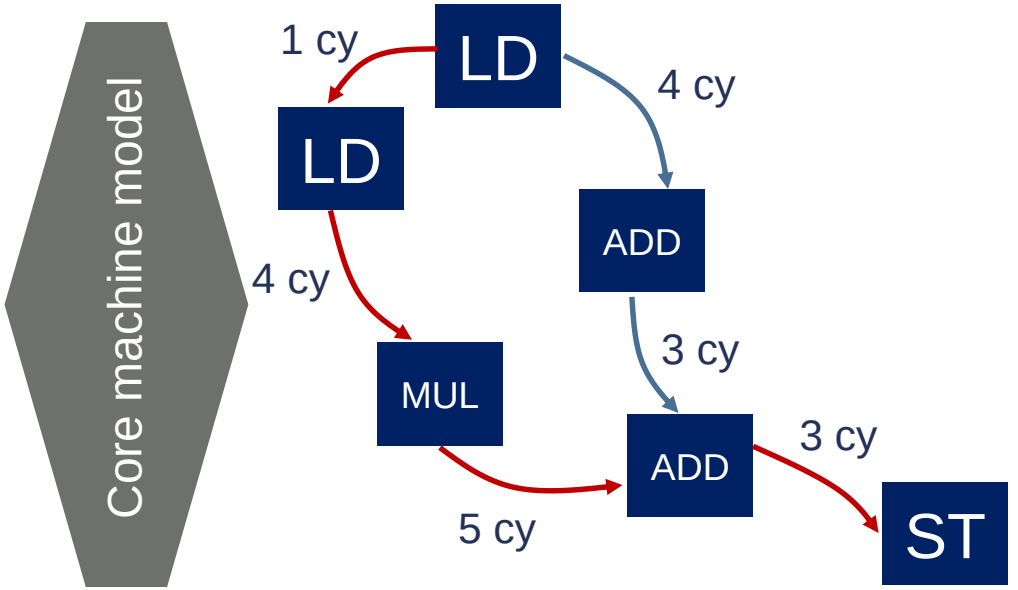
$$\{T_{L1L2} | T_{L2L3} | T_{L3Mem}\} = \{ 4 \mid 8 \mid 18.4 \}\,\mathrm{cy}/8\,\mathrm{iter}$$

- Input:
  - Cache properties
  - Application data transfer prediction



ECM model components: In-core execution





Best case: max throughput

Worst case: critical path

$$T_{\text{core}}^{\min} = \max(T_{\text{nOL}}, T_{\text{OL}})$$

$$T_{\text{core}}^{\max} = T^{\text{CP}}$$

T_{nOL} interacts with cache hierarchy, T_{OL} does not

ECM model components: Overlap assumptions

- Notation for model contributions

$$\{T_{\mathrm{OL}} \,||\, T_{\mathrm{nOL}} | T_{L1L2} | T_{L2L3} | T_{L3Mem}\} = \{\, 7 \,||\, 2 \mid 4 \mid 8 \mid 18.4 \,\}\,\mathrm{cy}/8\,\mathrm{iter}$$

- Most pessimistic overlap model: no overlap

$$T_{ECM}^{Mem} = \max(T_{\mathrm{OL}}, T_{\mathrm{nOL}} + T_{L1L2} + T_{L2L3} + T_{L3Mem}) \text{ for in-mem data}$$

Appropriate for most Intel Xeon CPUs up to and including Broadwell

ECM model:
Notation for runtime predictions

$$\{T_{\mathrm{L1}}^{ECM} \rceil T_{L2}^{ECM} \rceil T_{L3}^{ECM} \rceil T_{Mem}^{ECM}\}$$

Example: no-overlap model

$$\{\max(T_{\mathrm{OL}}, \mathrm{T_{nOL}}) \rceil$$
$$\max(T_{\mathrm{OL}}, \mathrm{T_{nOL}} + T_{L1L2}) \rceil$$
$$\max(T_{\mathrm{OL}}, \mathrm{T_{nOL}} + T_{L1L2} + T_{L2L3}) \rceil$$
$$\max(T_{\mathrm{OL}}, \mathrm{T_{nOL}} + T_{L1L2} + T_{L2L3} + T_{L3Mem})\}$$

L1
L2
L3
Memory

data in…

ECM model: (Naive) saturation assuption

- Performance is assumed to scale across cores until a shared bandwidth bottleneck is hit

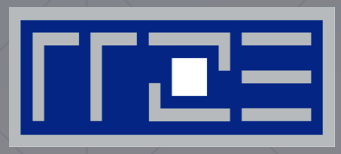
$$T_{ECM}(n) = \max\left(\frac{T_{Mem}^{ECM}}{n}, T_{L3Mem}\right) \quad \Longrightarrow \quad n_S = \left\lceil \frac{T_{ECM}^{Mem}}{T_{L3Mem}} \right\rceil$$

Roofline bandwidth ceiling

- This is (sometimes) too optimistic near the saturation point. For improvements see

J. Hofmann, G. Hager, and D. Fey: *On the accuracy and usefulness of analytic energy models for contemporary multicore processors*. Proc. ISC High Performance 2018. DOI: 10.1007/978-3-319-92040-5_2

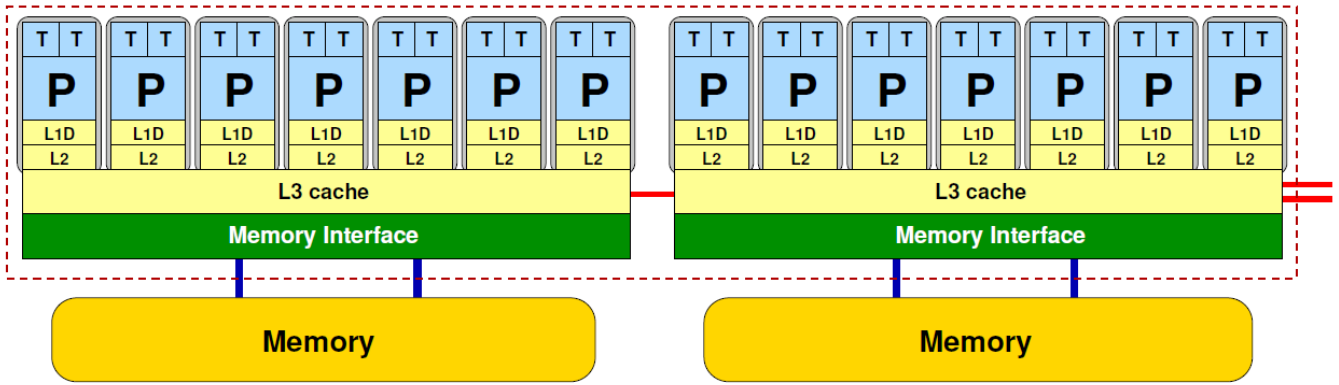
MODELING A CONJUGATE-GRADIENT SOLVER



Building a model from components

A matrix-free CG solver

- 2D 5-pt FD Poisson problem
- Dirichlet BCs, matrix-free
- N_x x N_y = 40000 × 1000 grid
- CPU: **Haswell E5-2695v3 CoD mode**



Machine characteristics

- 2.3 GHz (fixed core & Uncore)
  - AVX2, 2 x FMA per cycle
  - 2 load & 1 store per cycle (in practice: 1+1)

- Cache characteristics
  - Inclusive, non-overlapping
  - $b_{L1L2} = 43$ Byte/cy (gray)
  - $b_{L2L3} = 32$ Byte/cy (white)

- Memory bandwidth per ccNUMA domain (saturated)
  - $b_{read} = 32.3$ GByte/s $= 11.3$ Byte/cy
  - $b_{copy} = 26.1$ GByte/s $= 14.0$ Byte/cy

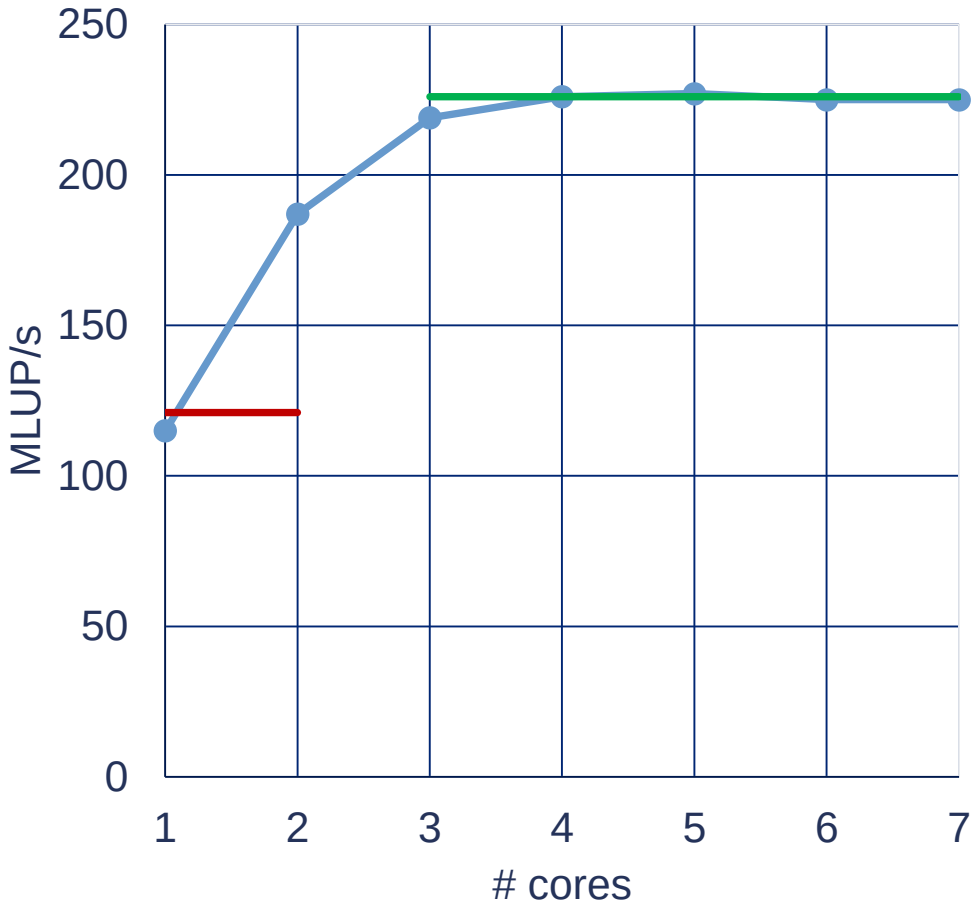
ECM model composition

Naive implementation of all kernels (omp parallel for)

while(α_0 < tol):	T_x [cy/8 iter]	T_{Mem}^{ECM} [cy/8 iter]	n_s [cores]	Full domain limit [cy/8 iter]
$\vec{v} = A\vec{p}$	{ 8 || 4 | 6.7 | 10 | 16.9 }	37.6	3	16.9
$\lambda = \alpha_0 / \langle \vec{v}, \vec{p} \rangle$	{ 2 || 2 | 2.7 | 4 | 9.1 }	17.8	2	9.11
$\vec{x} = \vec{x} + \lambda\vec{p}$	{ 2 || 4 | 6 | 16.9 }	29.0	2	16.9
$\vec{r} = \vec{r} - \lambda\vec{v}$	{ 2 || 4 | 6 | 16.9 }	29.0	2	16.9
$\alpha_1 = \langle \vec{r}, \vec{r} \rangle$	{ 2 || 2 | 1.3 | 2 | 4.6 }	9.90	3	4.56
$\vec{p} = \vec{r} + \frac{\alpha_1}{\alpha_0}\vec{p}$, $\alpha_0 = \alpha_1$	{ 2 || 4 | 6 | 16.9 }	29.0	2	16.9
	Sum	**152**		**81.3**

CG performance – 1 core to full socket

- Multi-loop code well represented
- Single core performance predicted with 5% error
- Saturated performance predicted with < 0.5% error
- Saturation point predicted approximately
  - Can be fixed by improved ECM model



Barrier cost

- Does the OpenMP barrier impact the performance?
- At which problem size can this be expected?

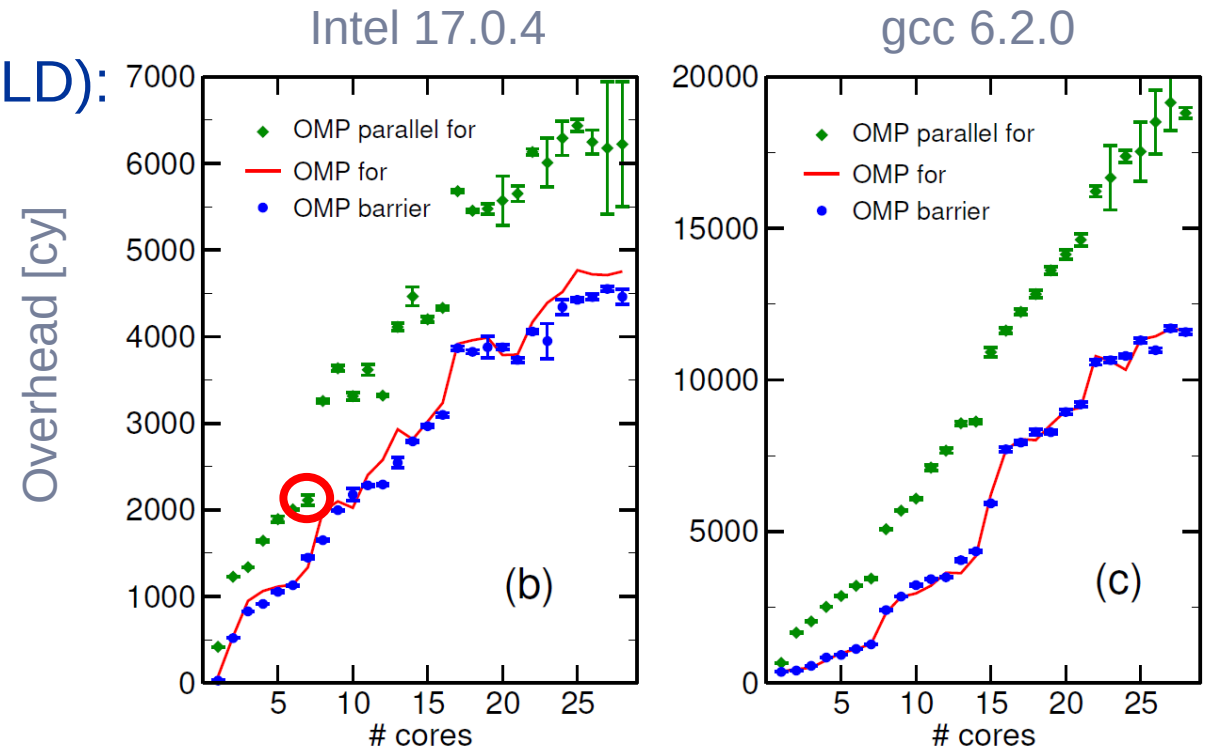
Overhead (1 NUMA LD):

6 x 2000 cy

Time:

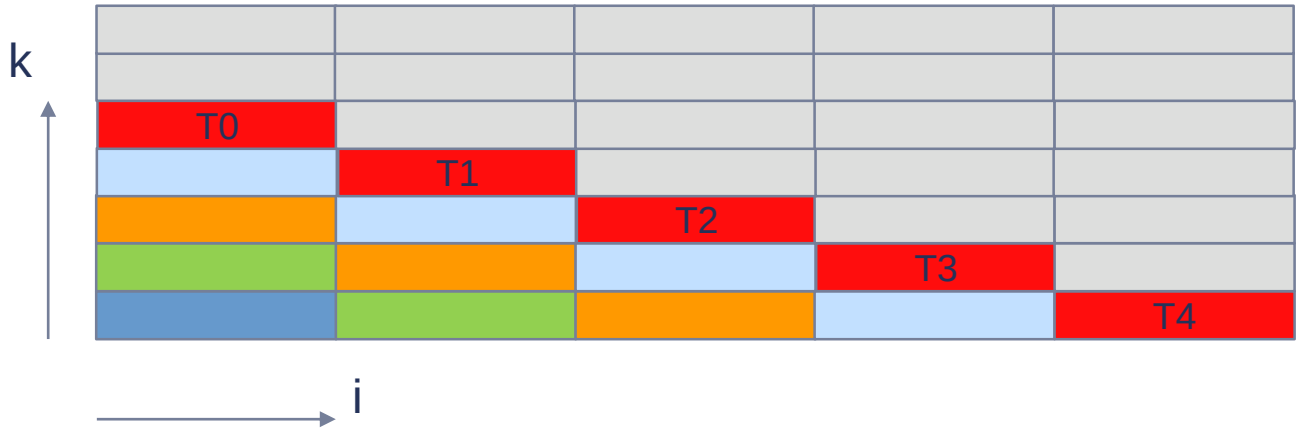
10 x $4x10^8$ cy

→ No problem for any out of cache data set



CG with GS preconditioner: Naïve parallelization

Pipeline parallel processing: OpenMP barrier after each wavefront update (ugh!)



CG with GS preconditioner: additional kernels

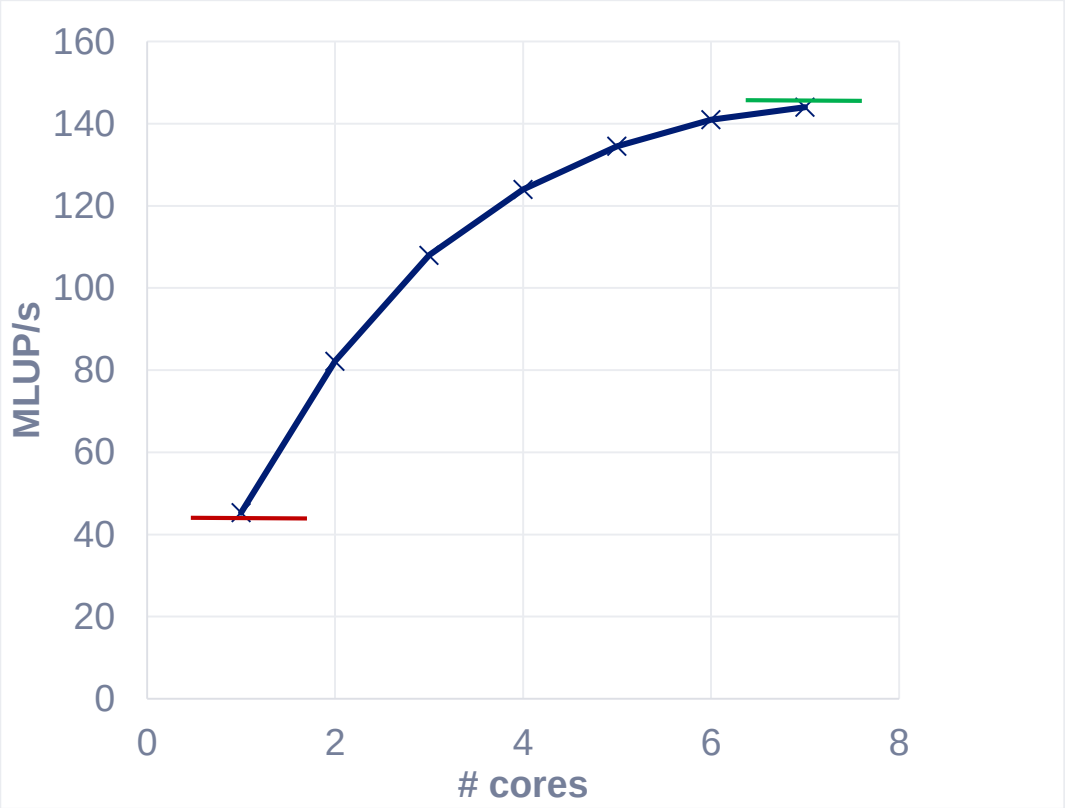
Intel IACA

	T_x [cy/8 iter]	T_{Mem}^{ECM} [cy/8 iter]	n_s [cores]	Full domain limit [cy/8 iter]
Non-PC model		**152**		**81.3**
$\vec{z} = P\vec{r}$ (fw)	{ 108 || 16 | 5.4 | 8 | 16.9 }	108	7	16.9
$\vec{z} = P\vec{r}$ (bw)	{ 138 || 16 | 4.0 | 6 | 11.3 }	138	13	19.7
$\alpha = \langle \vec{r}, \vec{z} \rangle$	{ 2 || 2 | 2.7 | 4 | 9.1 }	17.8	2	9.1
	Sum	**416**		**127**

- Back substitution does not saturate the memory bandwidth!
  - → full algorithm does not fully saturate
- Impact of barrier still negligible overall, but noticeable in the preconditioner

PCG measurement

- <2% model error for single threaded and saturated performance
- Expected large impact of barrier at smaller problem sizes in x direction



Conclusions

- Analytic modeling is worth the effort
  - Even if it‘s inaccurate
- Even Roofline can yield amazing insights
- Analytic modeling
  - is not just for “simple kernels”
  - is composable
  - can cover programming model overhead, too
- … and yes, it is real work.
- No, it does not always work.

http://tiny.cc/kerncraft

Automatic Roofline/ECM modeling tool