Erlangen Regional Computing Center

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Zum Verständnis des Performanceverhaltens moderner Prozessoren für dünn besetzte lineare Algebra

C.L. Alappat, G. Hager, J. Laukemann, G. Wellein

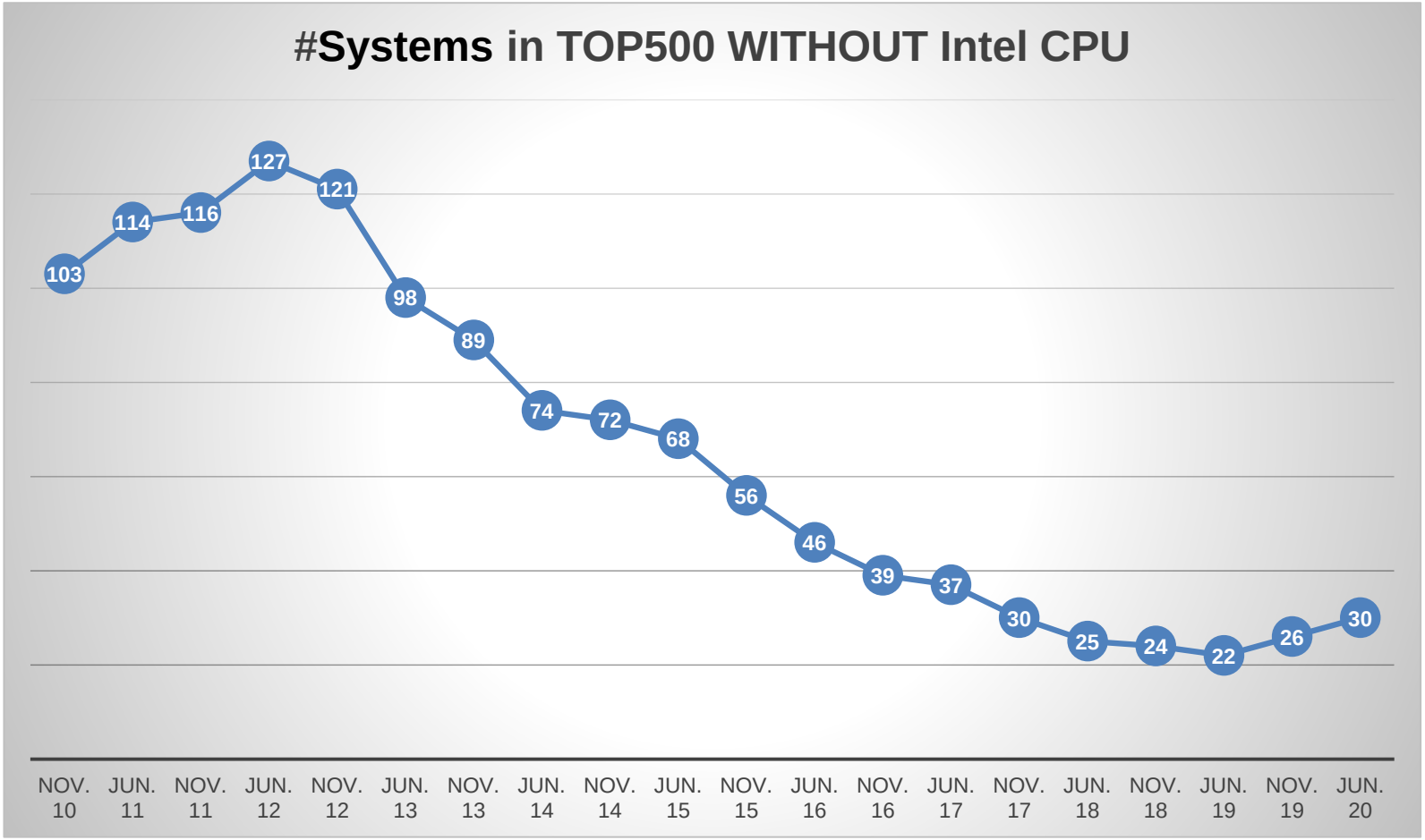
(RRZE / FAU Erlangen-Nürnberg)

N. Meyer, T. Wettig

(Department of Physics, Univ. Regensburg)

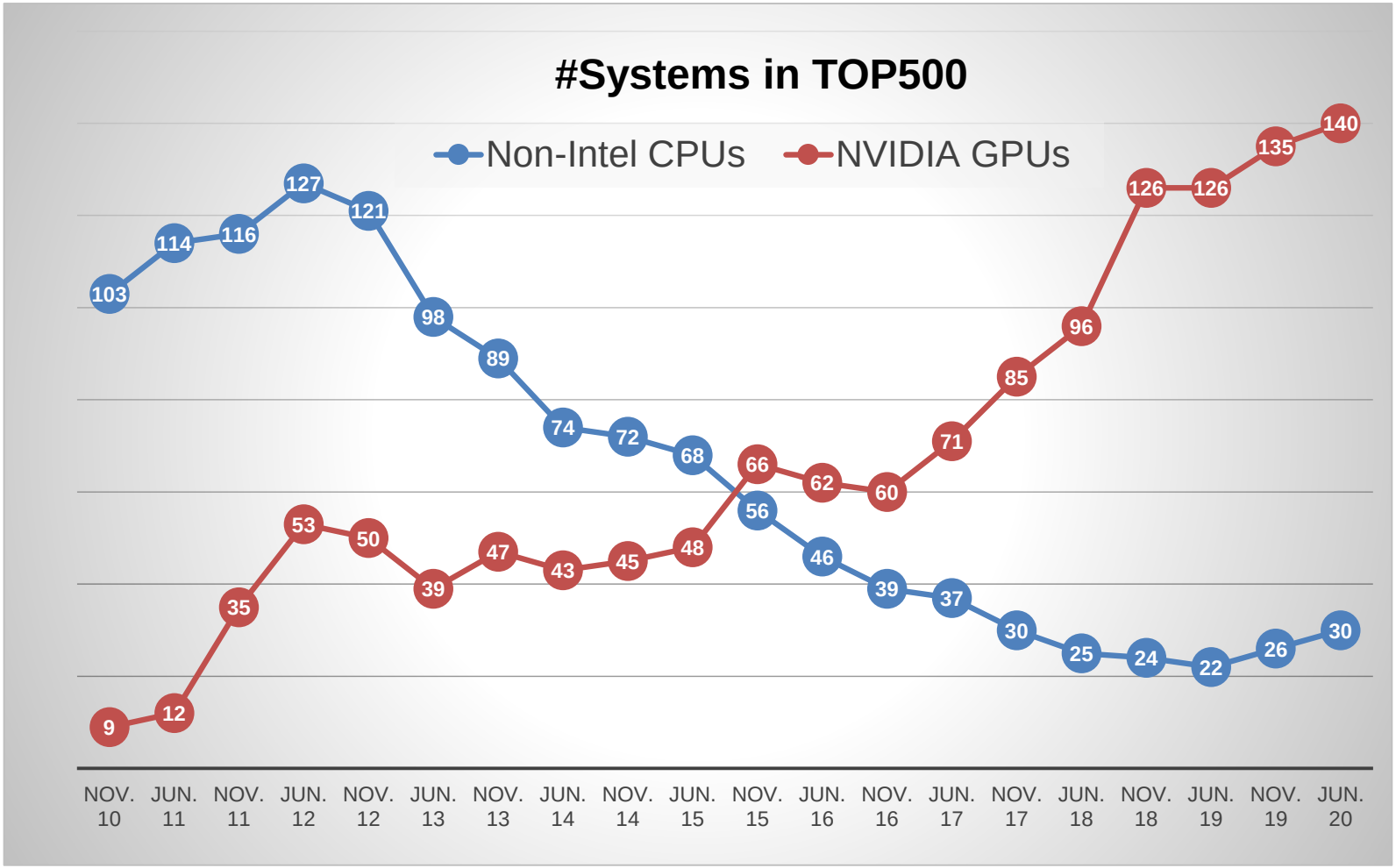
HPC High Performance Computing

The HPC universe is blue…..



Data: https://www.top500.org/statistics/list/

… almost



Data: https://www.top500.org/statistics/list/

But there is hope…

Cambrian Explosion[1] also on CPU/accelerator architecture?

Intel
*-Lake / Sapphire R.

NVIDIA
Volta / Ampere

Marvell
Thunder **X2**/X3/**X?** arm

AMD
Rome/Milano

AMD
Radeon Instinct

Fujitsu
A64FX arm

NEC
Tsubasa

EPI
arm + RISC V arm

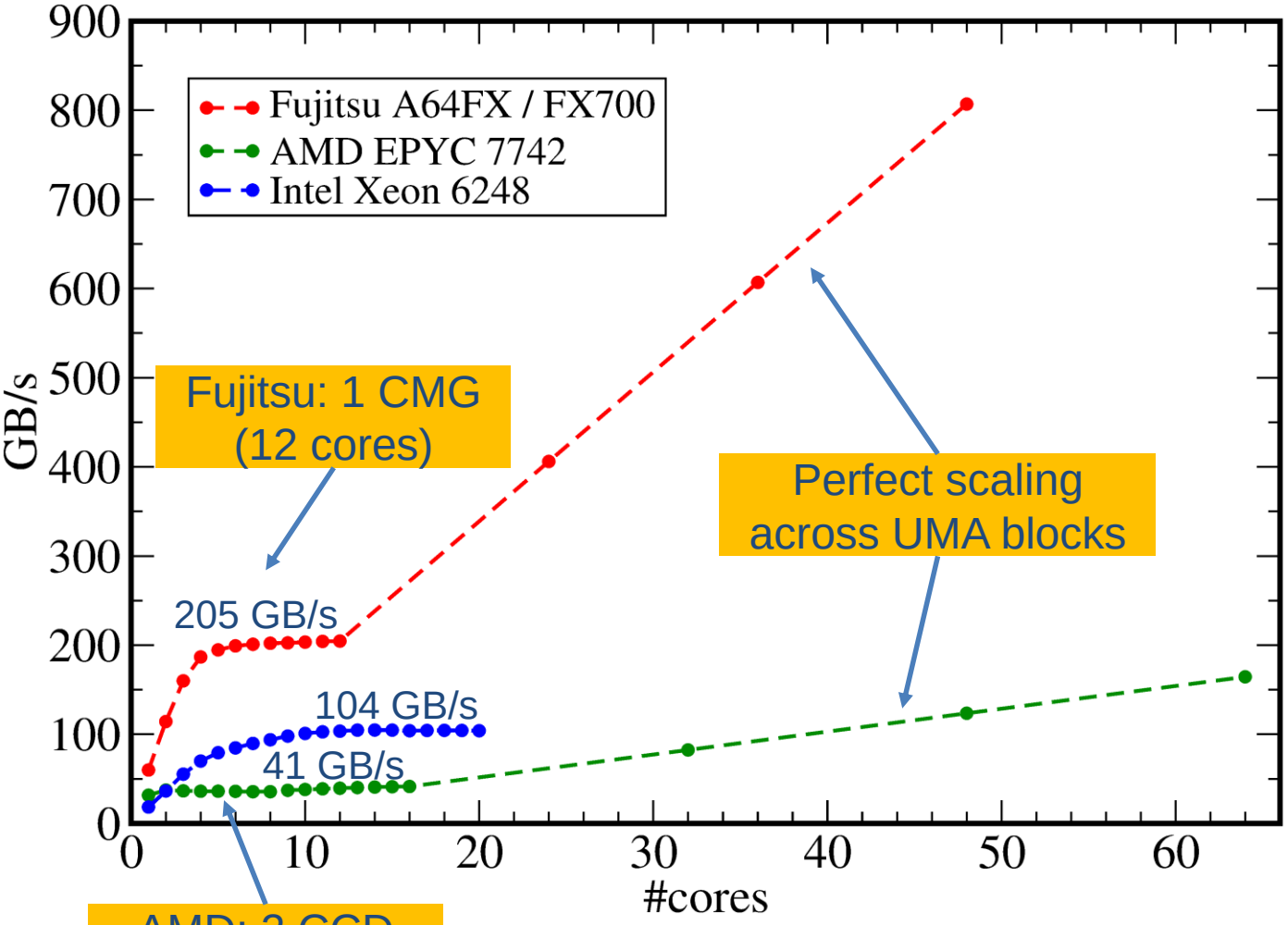
Many more:
IBM Power, China,...

[1]S. Matsuoka, Cambrian explosion of computing and big data in the post-moore era, https://doi.org/10.1145/3208040.3225055

Systems under investigation

	Fujitsu A64FX/FX700 (U. Regensburg)	AMD EPYC 7702 (HLRS)	Intel Xeon 6248 (RRZE)
Clock	1.8 GHZ	2.25 GHz	2.5 GHz
Flops/cy	32 (2xFMA512b)	16 (2xFMA256b)	32 (2xFMA512b)
#cores	48	64	20
L1 cache	48 x 64kiB	64 x 32kiB	20 x 32 kiB
L2 cache	4 x 8 MiB	64 x 0.5 MiB	20 x 1 MiB
L3 cache	---	16 x 16 MiB	1 x 27.5 MiB
MemBW	920 GB/s (HBM2)	204.8 GB/s (DDR4)	140.8 GB/s (DDR4)
Settings	---	NPS=4	CoD=off
Compiler	gcc-10.1	Intel-19	Intel-19

Memory Performance: „Socket“ Scaling



Benchmark

- STREAM Triads: A(i)=B(i)+s*C(i)
- Compiler code
- Standard Stores
- Corrected for WA
- Compact pinning

- Four UMA blocks: AMD & Fujitsu
- Attainable BW: 80% - 90% Peak

Erlangen Regional Computing Center

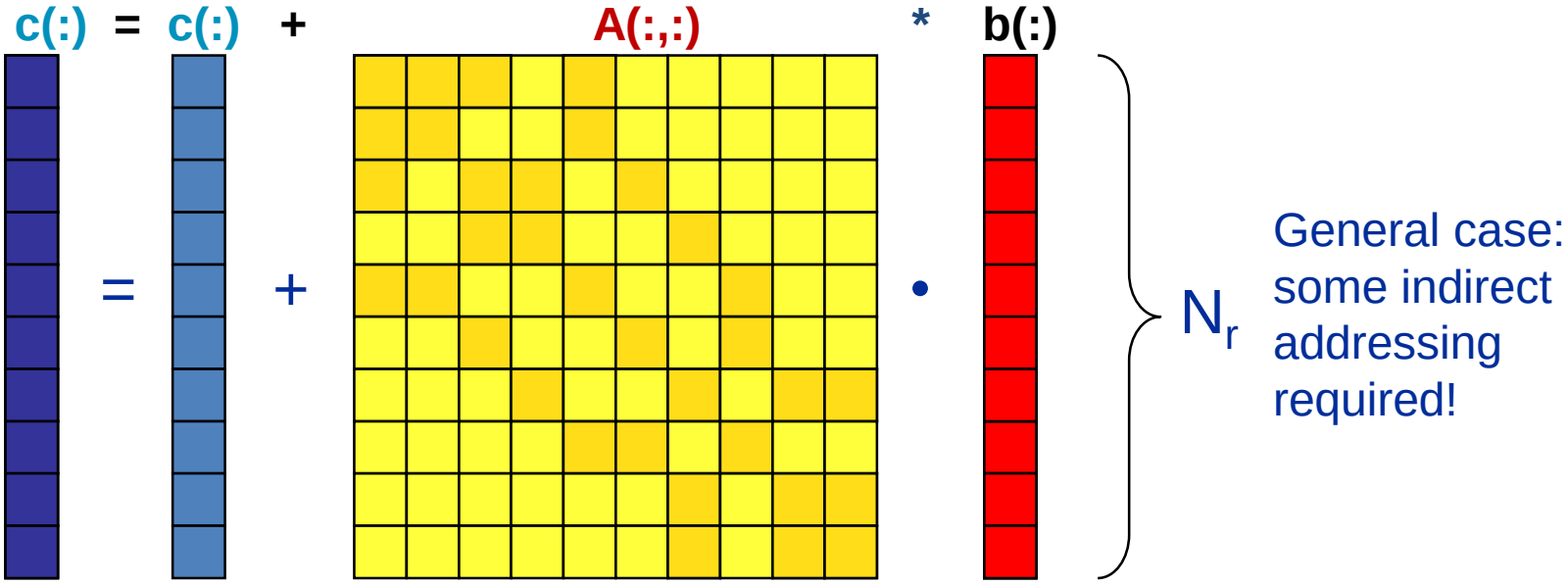
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Sparse Matrix Vector Multiplication (SpMVM)

High Performance Computing

Sparse Matrix Vector Multiplication

- Basic kernel: Multiply sparse matrix with dense vector



- Compressed Row Storage (CRS) SpMVM kernel:

```
do i = 1,Nr ! Long loop
 do j = row_ptr(i), row_ptr(i+1) - 1 ! Short loop
  c(i) = c(i) + val(j) * b(col_idx(j))
 enddo
enddo
```

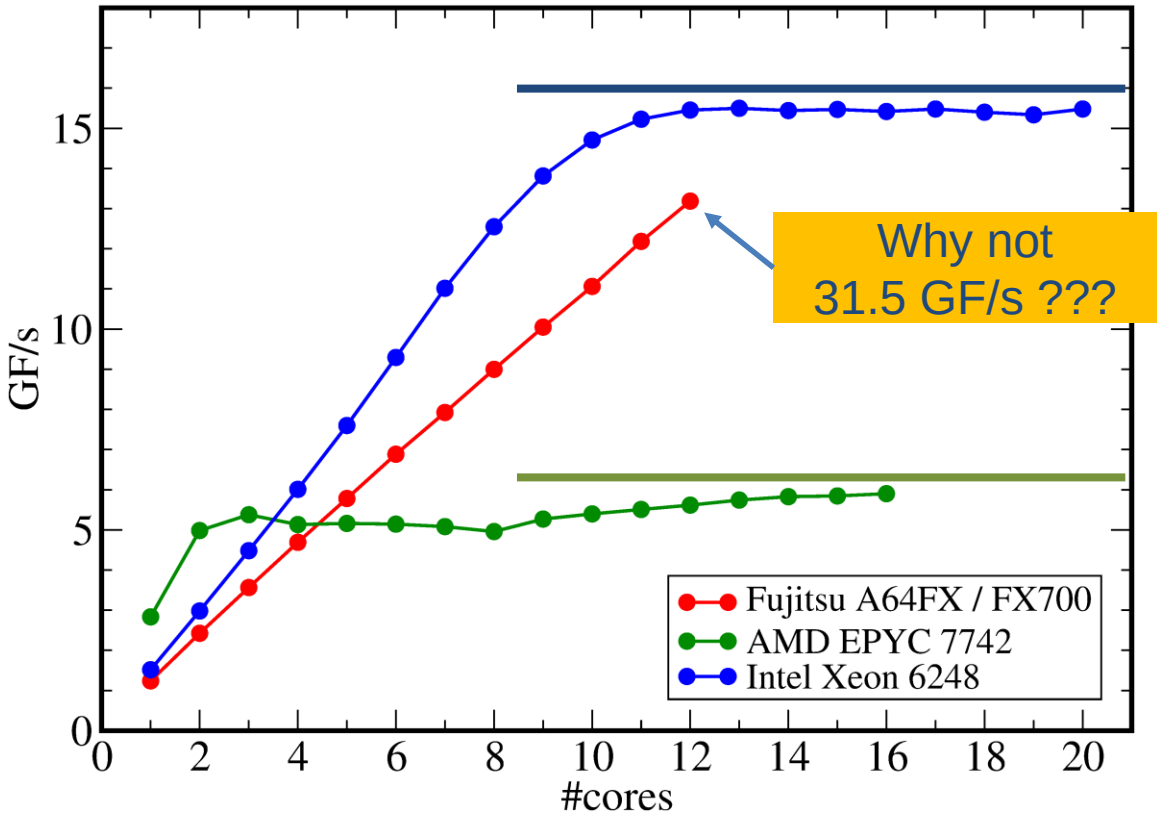
Matrix data (8 B+ 4B per non-zero)

Minimum code balance:

$$B_C = 6\frac{B}{F}$$

SpMVM Performance

- UMA building block performance for HPCG matrix
  (Data: 128^3: $N_r = 2 * 10^6$; non-zeros per row (inner loop length): 27; double)



Roofline limit[1]
(analytic α for HPCG)

$$B_C = 6.5 \frac{B}{F}$$

Intel Xeon:
104/6.5 GF/s = 16 GF/s

A64FX:
205/6.5 GF/s = 31.5 GF/s

AMD EPYC:
41/6.5 GF/s = 6.3 GF/s

[1]C. Alappat et al., ISC 2020, https://doi.org/10.1007/978-3-030-50743-5_21

SpMVM Performance Analysis A64FX

- Low single core A64FX performance for CRS
- gcc compiler generated **vectorized** code (inner loop)

```
.L6:
        ld1sw     z0.d, p0/z, [x17, x20, lsl 2]
        ld1d      z2.d, p0/z, [x18, x20, lsl 3]
        ld1d      z3.d, p0/z, [x30, z0.d, lsl 3]
        add       x20, x20, 8
        fmla      z1.d, p0/m, z3.d, z2.d
        whilelo   p0.d, x20, x14
        b.any     .L6

        faddv     d4, p1, z1.d
```

Load 8 matrix entries

Gather: b(col_idx(:))

FMA3: Update **z1.d**

Latency: 9 cycles

„Horizontal add“ of entries in 512-bit register (**z1.d**)

Costs > 11 cycles

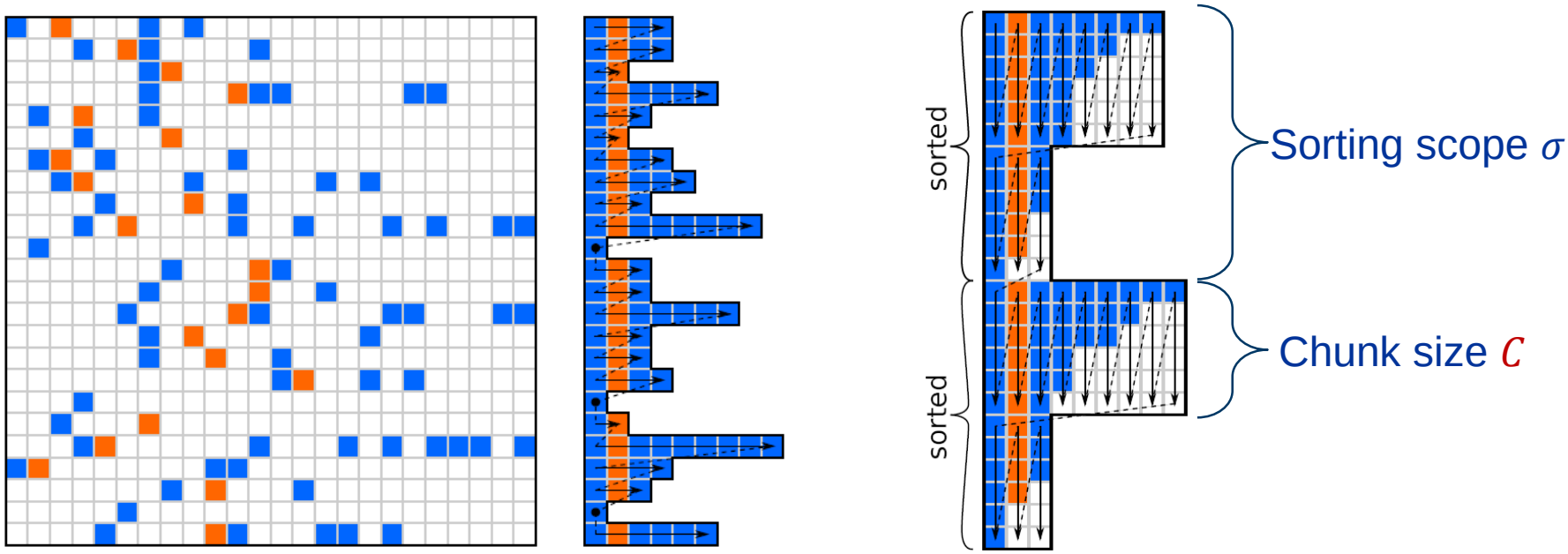
```
do j = row_ptr(i), row_ptr(i+1) - 1 ! Short loop
 c(i) = c(i) + val(j) * b(col_idx(j))
enddo
```

→ This code can not saturate memory bandwidth for HPCG matrix on A64FX !
(see C. Alappat et al., submitted, https://arxiv.org/abs/2009.13903)

Better SpMVM data format…

Vector-friendly SpMVM data format/kernel required for A64FX

→ Choose SELL-C-σ[1]



Chunk size C: Multiple of SIMD register size

→ No „horizontal add“ + Vectorization & pipelining

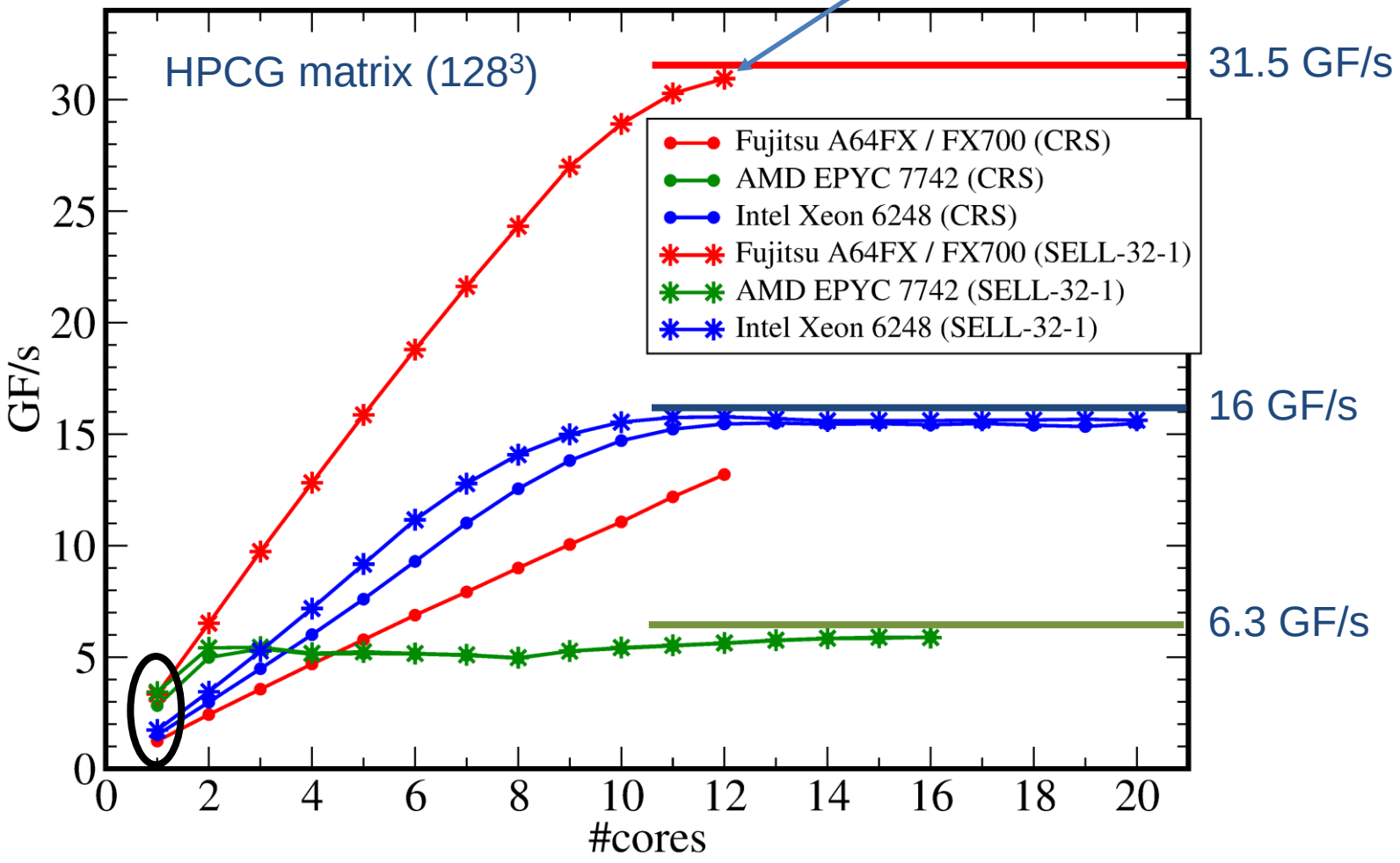
→ Implementation: SIMD intrinisc kernels (single core / data structure!)

[1]M. Kreutzer et al., SIAM SISC 2014, DOI: 10.1137/130930352

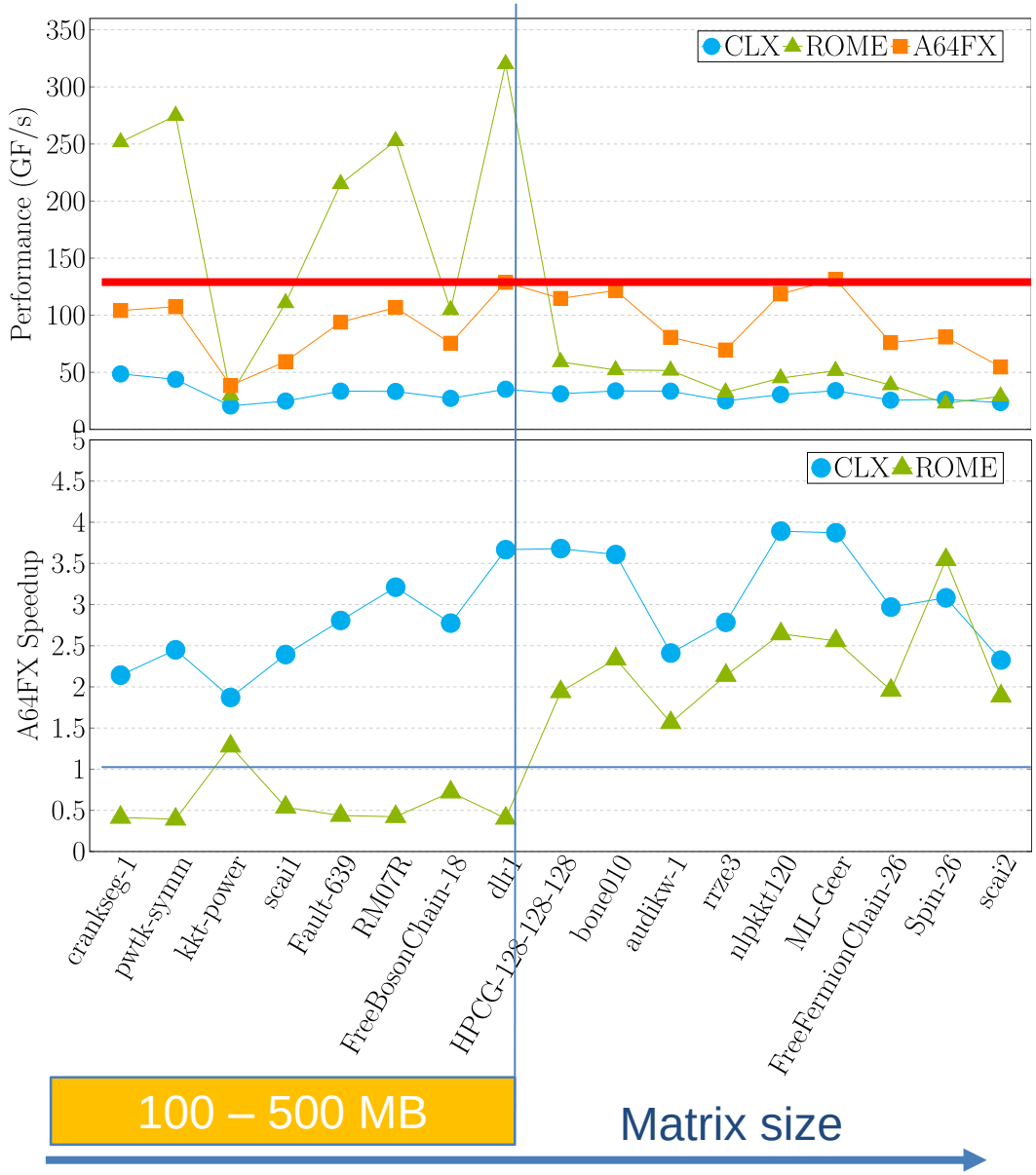
… and it‘s impact on SpMVM performance

Single core performance optimization crucial for A64FX to saturate bandwidth

„Weak“ Saturation



A broader SpMVM picture



Benchmark:

- Sequence of SpMVMs
- Selected matrices
- Matrix size > 100 MB
- Full node

Performance limits:

- A64FX:
  4*31.5 GF = 126 GF/s
- AMD:
  8*6.3 GF/s = 50.4 GF/s
- Intel:
  2*16 GF/s = 32 GF/s

A64FX „Weak“ Saturation:

- Exact load balancing required

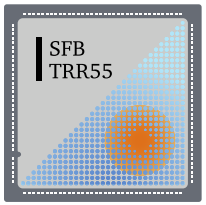
AMD:

- Benefit of large cache:
  (128*0.5 + 32*16) MB

Lessons learned

- AMD good balance of:
  - High Memory BW
  - Large, efficient L3 caches
  - Single core performance
- A64FX:
  - High bandwidth / HBM accessible for CPU-codes
  - Efficient compilers and optimized codes are required
- Intel
  - Room for improvement: **Single core** & socket memory BW
  - Highly efficient core and mature / optimized SW environment

Acknowledgement & Background information



About us www.hpc.fau.de

- AMD Rome architecture: https://kb.hlrs.de/platforms/upload/Processor.pdf
- Fujitsu FX700 / A64FX architecture: https://github.com/fujitsu/A64FX/tree/master/doc
- Machine model A64FX (ECM): https://arxiv.org/abs/2009.13903