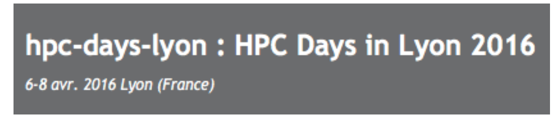


An Overview of High Performance Computing and Challenges for the Future

Jack Dongarra

University of Tennessee
Oak Ridge National Laboratory
University of Manchester

Outline

- Overview of High Performance Computing
- Look at some implementations of linear algebra algorithms on today's High Performance Computers
  - As an examples of the kind of thing needed.

State of Supercomputing Today

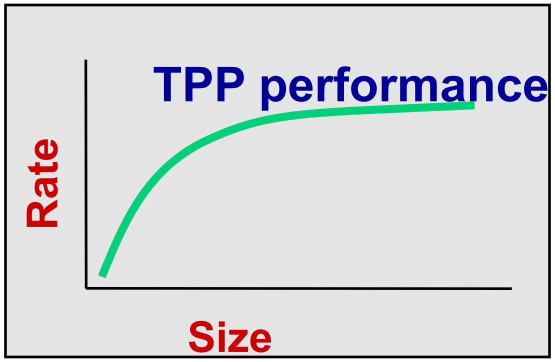
- Pflops (> 10^{15} Flop/s) computing fully established with 81 systems.
- Three technology architecture possibilities or "swim lanes" are thriving.
  - Commodity (e.g. Intel)
  - Commodity + accelerator (e.g. GPUs) (104 systems)
  - Special purpose lightweight cores (e.g. IBM BG, ARM, Intel's Knights Landing)
- Interest in supercomputing is now worldwide, and growing in many new markets (around 50% of Top500 computers are used in industry).
- Exascale (10^{18} Flop/s) projects exist in many countries and regions.
- Intel processors have largest share, 89% followed by AMD, 4%.

TOP500 Supercomputer

H. Meuer, H. Simon, E. Strohmaier, & JD

- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP

  Ax=b, dense problem



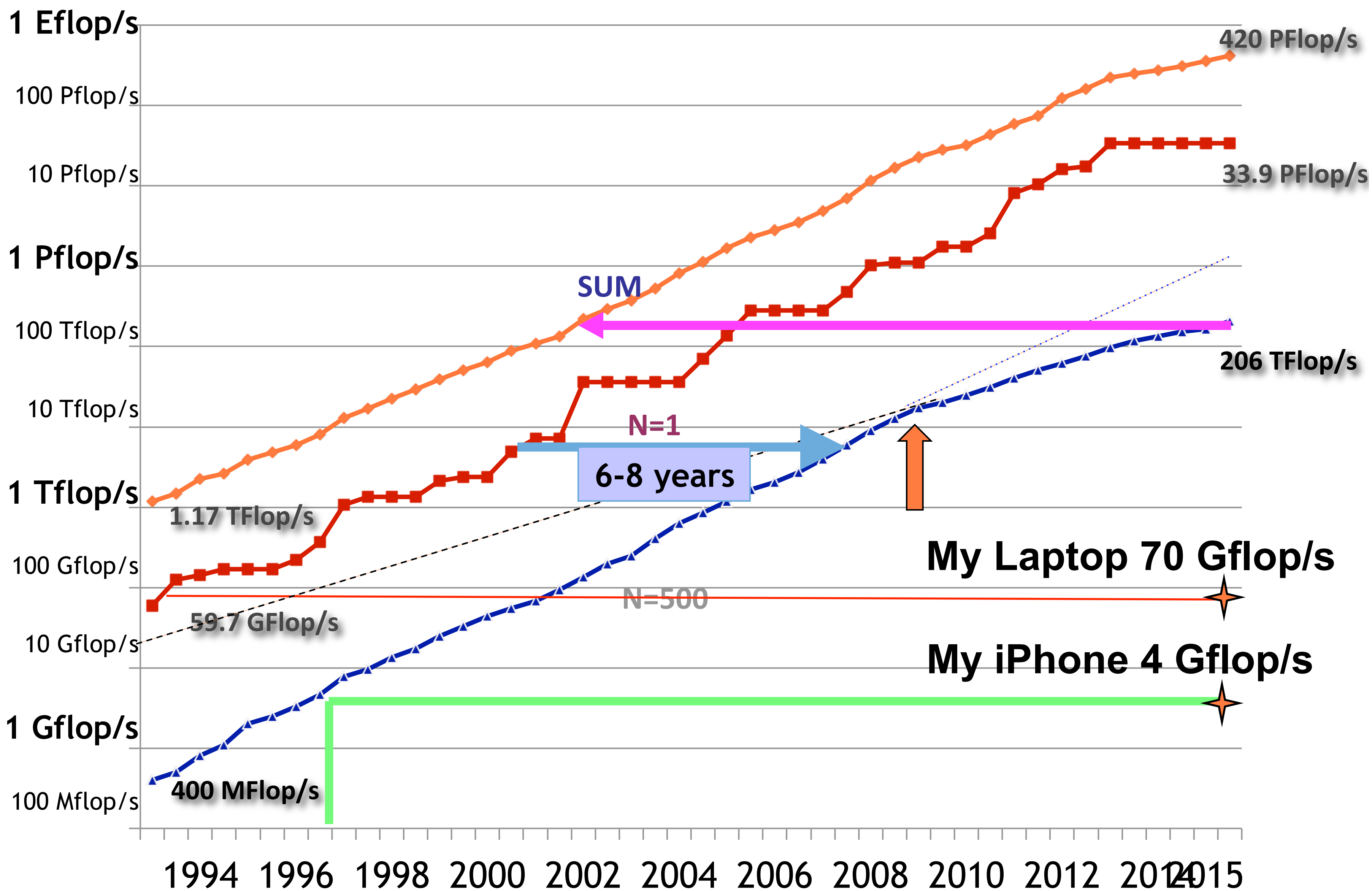
- Updated twice a year

  SC'xy in the States in November

  Meeting in Germany in June

- All data available from **www.top500.org**

Performance Development of HPC over the Last 24 Years from the Top500



November 2015: The TOP 10 Systems

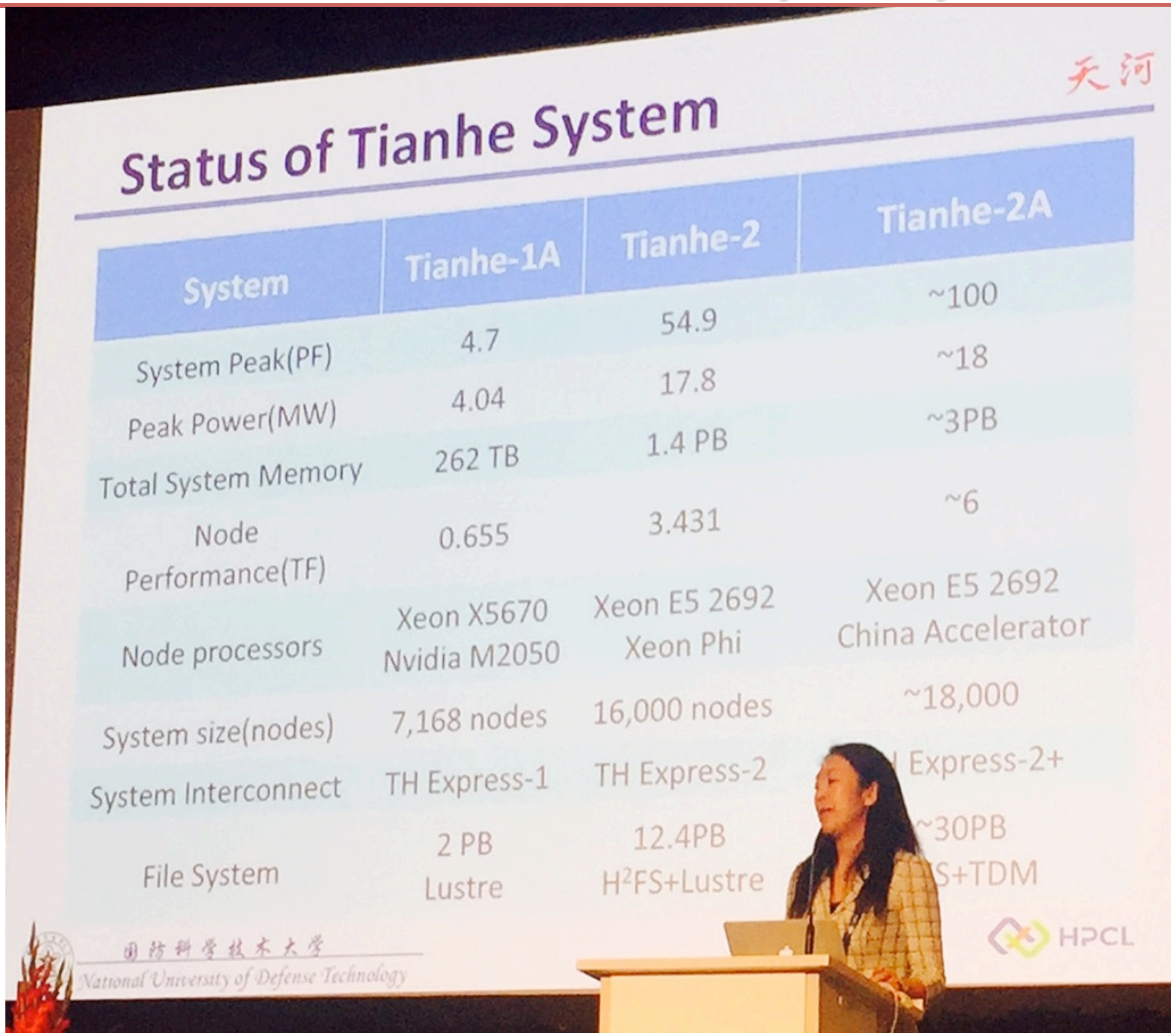
Rank	Site	Computer	Country	Cores	Rmax [Pflops]	% of Peak	Power [MW]	MFlops /Watt
1	National Super Computer Center in Guangzhou	Tianhe-2 NUDT, Xeon 12C + Intel Xeon Phi (57c) + Custom	China	3,120,000	33.9	62	17.8	1905
2	DOE / OS Oak Ridge Nat Lab	Titan, Cray XK7, AMD (16C) + Nvidia Kepler GPU (14c) + Custom	USA	560,640	17.6	65	8.3	2120
3	DOE / NNSA L Livermore Nat Lab	Sequoia, BlueGene/Q (16c) + custom	USA	1,572,864	17.2	85	7.9	2063
4	RIKEN Advanced Inst for Comp Sci	K computer Fujitsu SPARC64 VIIIfx (8c) + Custom	Japan	705,024	10.5	93	12.7	827
5	DOE / OS Argonne Nat Lab	Mira, BlueGene/Q (16c) + Custom	USA	786,432	8.16	85	3.95	2066
6	DOE / NNSA / Los Alamos & Sandia	Trinity, Cray XC40,Xeon 16C + Custom	USA	301,056	8.10	80		
7	Swiss CSCS	Piz Daint, Cray XC30, Xeon 8C + Nvidia Kepler (14c) + Custom	Swiss	115,984	6.27	81	2.3	2726
8	HLRS Stuttgart	Hazel Hen, Cray XC40, Xeon 12C + Custom	Germany	185,088	5.64	76		
9	KAUST	Shaheen II, Cray XC40, Xeon 16C + Custom	Saudi Arabia	196,608	5.54	77	2.8	1954
10	Texas Advanced Computing Center	Stampede, Dell Intel (8c) + Intel Xeon Phi (61c) + IB	USA	204,900	5.17	61	4.5	1489
500 (368)	Karlsruher	MEGAWARE Intel	Germany	10,800	.206	95		

Recent Developments

- US DOE planning to deploy O(100) Pflop/s systems for 2017-2018 - $525M hardware
- Oak Ridge Lab and Lawrence Livermore Lab to receive IBM and Nvidia based systems
- Argonne Lab to receive Intel based system
  - After this the Exaflop
- US Dept of Commerce is … a groups from receiving In…
  - …cle… …th the …
  - ….

  - …ingzl…
  - …njin,…
  - National University for Def…
  - National SC Center Changsh…

Yutong Lu from NUDT at the International Supercomputer Conference in Germany in July 2015

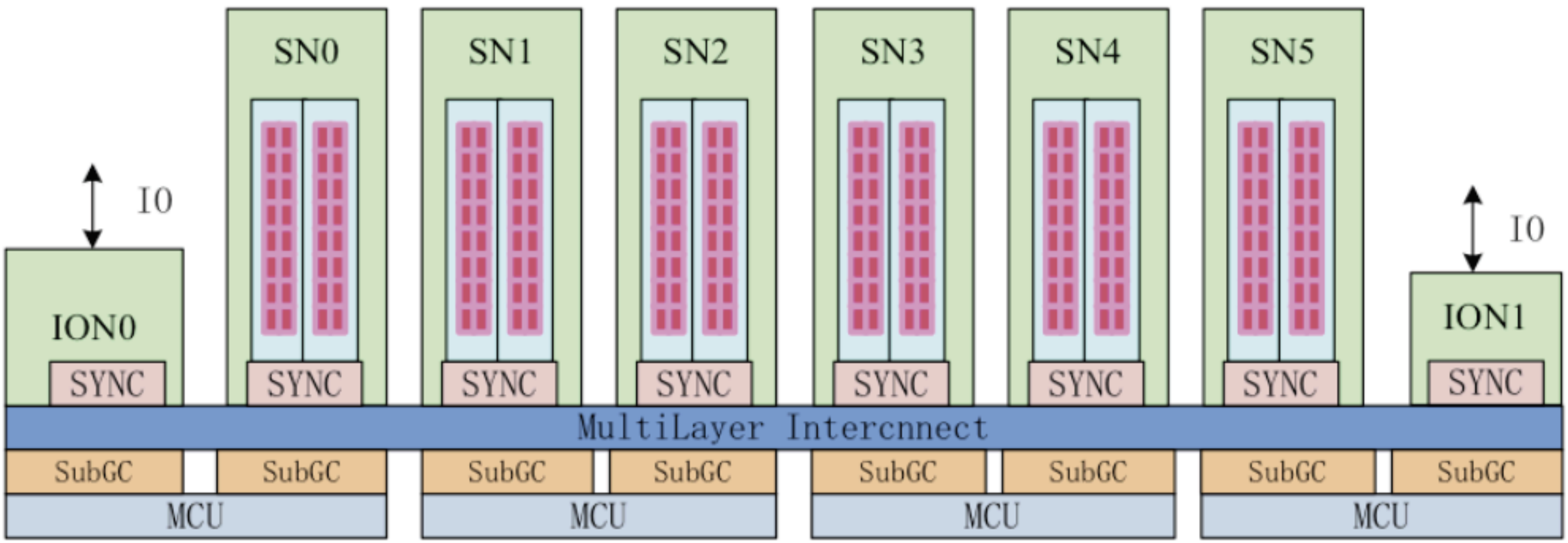


China Accelerator

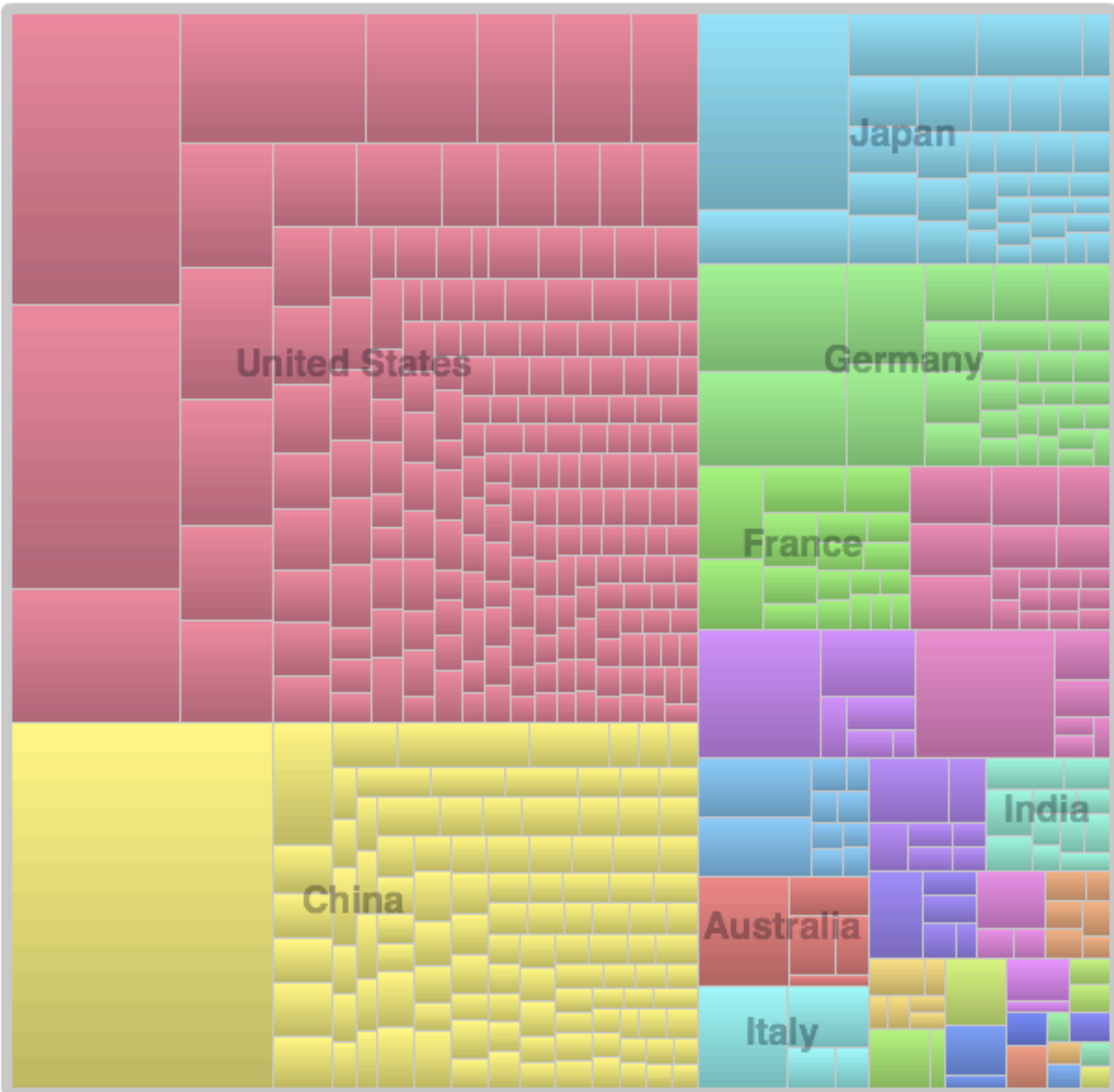
天河

Matrix2000 GPDSP

- High Performance
  - 64bit Supported
  - ~2.4/4.8TFlops(DP/SP)
  - 1GHz, ~200W
- High Throughput
  - High-bandwidth Memory
  - 32~64GB
  - PCIE 3.0, 16x



Countries Share



Absolute Counts

US:	201
China:	109
Japan:	38
UK:	18
France:	18
Germany:	32

China nearly tripled the number of systems on the latest list, while the number of systems in the US has fallen to the lowest point since the TOP500 list was created.

France's 18 systems on Top500

Rank	Name	Computer	Site	Manufacturer	Total Cores	Rmax	% Peak
33	Pangea	SGI ICE X, Xeon 8C 2.600GHz, Inf FDR	Total	SGI	110400	2098090	91%
44	Occigen	bullx DLC, Xeon 12C 2.6GHz, Inf FDR	GENCI-CINES	Bull, Atos Group	50544	1628770	77%
53	Curie thin nodes	Bullx B510, Xeon 8C 2.700GHz, Inf QDR	CEA/TGCC-GENCI	Bull, Atos Group	77184	1359000	82%
70	Turing	BlueGene/Q, Power BQC 16C 1.60GHz, Custom	CNRS/IDRIS-GENCI	IBM	98304	1073327	85%
74	Tera-100	Bull bullx super-node S6010/S6030	CEA	Bull, Atos Group	138368	1050000	84%
121	Zumbrota	BlueGene/Q, Power BQC 16C 1.60GHz, Custom	EDF R&D	IBM	65536	715551	85%
167	HPC4	HP POD - Cluster Platform, Intel Xeon 12C 2.7GHz, Inf FDR	Airbus	Hewlett-Packard	34560	516897	69%
171	PORTHOS	IBM NeXtScale nx360M5, Xeon 14C 2.6GHz, Inf FDR	EDF R&D	IBM	16100	506357	76%
190	Beaufix	Bullx DLC B710 Blades, Intel Xeon 12C 2.7GHz, Inf FDR	Meteo France	Bull, Atos Group	24192	469097	90%
194	Prolix	Bullx DLC B710 Blades, Intel Xeon 12C 2.7GHz, Inf FDR	Meteo France	Bull, Atos Group	23760	464865	91%
215	Manny	bullx DLC 720, Xeon 12C 2.6GHz, Inf FDR	Bull	Bull, Atos Group	12960	430459	80%
278	Athos	iDataPlex DX360M4, Intel Xeon 12C 2.7GHz, Inf FDR14	EDF R&D	IBM	18144	352671	90%
283	airain	Bullx B510, Xeon 8C 2.7GHz, Ind QDR	CEA/CCRT	Bull, Atos Group	18144	346070	88%
399	EOS	Bullx DLC B710 Blades, Intel Xeon 10C 2.8GHz, Inf FDR	CALMIP / U of Toulouse	Bull, Atos Group	12240	255078	93%
400	romeo	Bull R421-E3 Cluster, Intel Xeon 8C 2.6GHz, Inf FDR, NVIDIA K20x	Champagne-Ardenne	Bull, Atos Group	5720	254900	66%
412		Cluster Platform 3000 BL460c, Intel Xeon 12C 2.7GHz, Inf FDR	Manufacturing Company	Hewlett-Packard	13152	249348	88%
421		HP POD - Cluster Platform 3000 BL260c G6, 3.06 GHz, Inf	Airbus	Hewlett-Packard	24192	243900	82%
433	Jade	SGI ICE 8200EX, Xeon 4C 3.000GHz, Inf	GENCI-CINES	SGI	23040	237800	89%

Commodity plus Accelerator Today 104 of the Top500 Systems

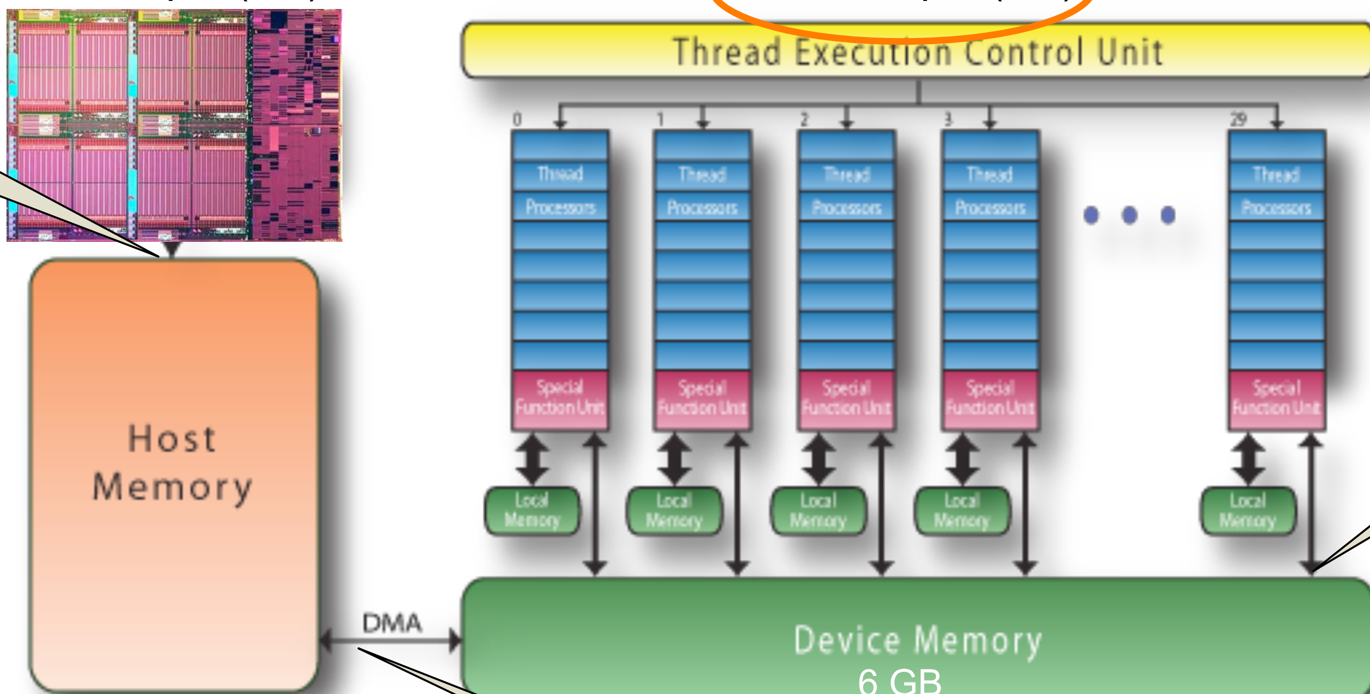
192 Cuda cores/SMX
2688 "Cuda cores"

Commodity

Intel Xeon
8 cores
3 GHz
8*4 ops/cycle
96 Gflop/s (DP)

Accelerator (GPU)

Nvidia K20X "Kepler"
2688 "Cuda cores"
.732 GHz
2688*2/3 ops/cycle
1.31 Tflop/s (DP)



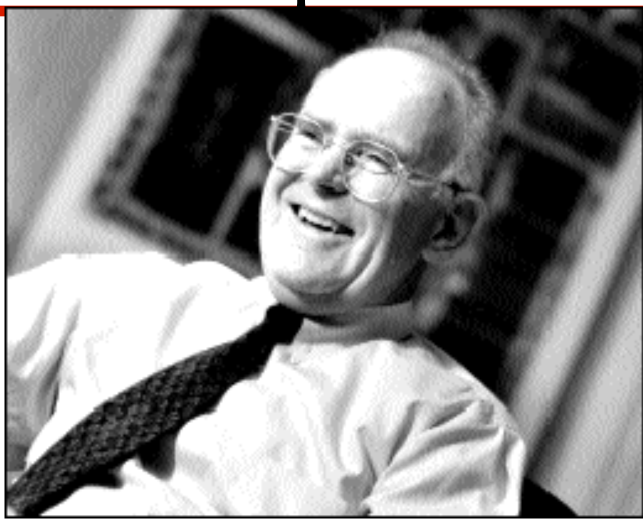
~68 GB/s

Total 288 GB/s

Interconnect
PCI-e Gen2/3 16 lane
64 Gb/s (8 GB/s)
1 GW/s

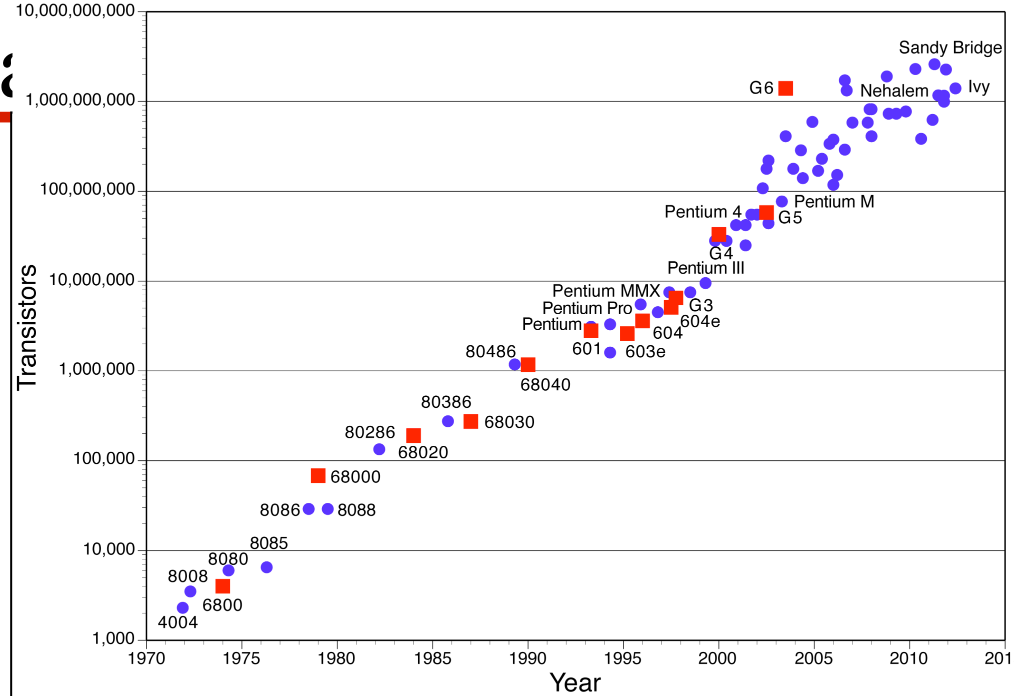
8 GB/s

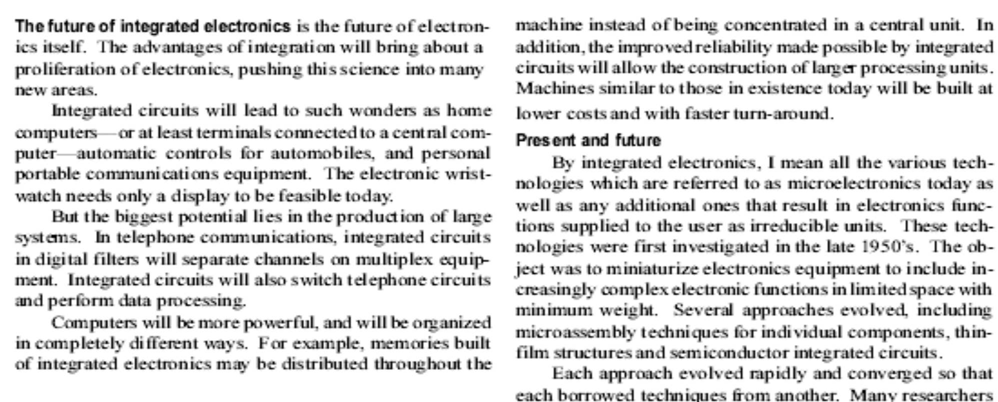
Technology Trends: Microprocessor Ca



Gordon Moore (co-founder of Intel) Electronics Magazine, 1965

Number of devices/chip doubles every 18 months





The future of integrated electronics is the future of electronics itself. The advantages of integration will bring about a proliferation of electronics, pushing this science into many new areas.

Integrated circuits will lead to such wonders as home computers—or at least terminals connected to a central computer—automatic controls for automobiles, and personal portable communications equipment. The electronic wristwatch needs only a display to be feasible today.

But the biggest potential lies in the production of large systems. In telephone communications, integrated circuits in digital filters will separate channels on multiplex equipment. Integrated circuits will also switch telephone circuits and perform data processing.

Computers will be more powerful, and will be organized in completely different ways. For example, memories built of integrated electronics may be distributed throughout the machine instead of being concentrated in a central unit. In addition, the improved reliability made possible by integrated circuits will allow the construction of larger processing units. Machines similar to those in existence today will be built at lower costs and with faster turn-around.

The author

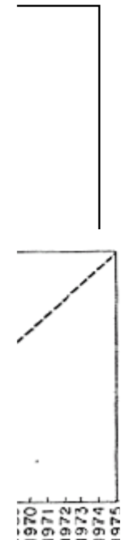
Dr. Gordon E. Moore is one of the new breed of electronic engineers, schooled in the physical sciences rather than in electronics. He earned a B.S. degree in chemistry from the

Present and future

By integrated electronics, I mean all the various technologies which are referred to as microelectronics today as well as any additional ones that result in electronics functions supplied to the user as irreducible units. These technologies were first investigated in the late 1950's. The object was to miniaturize electronics equipment to include increasingly complex electronic functions in limited space with minimum weight. Several approaches evolved, including microassembly techniques for individual components, thin-film structures and semiconductor integrated circuits.

Each approach evolved rapidly and converged so that each borrowed techniques from another. Many researchers believe the way of the future to be a combination of the various approaches.

The advocates of semiconductor integrated circuitry are already using the improved characteristics of thin-film resistors by applying such films directly to an active semiconductor substrate. Those advocating a technology based upon



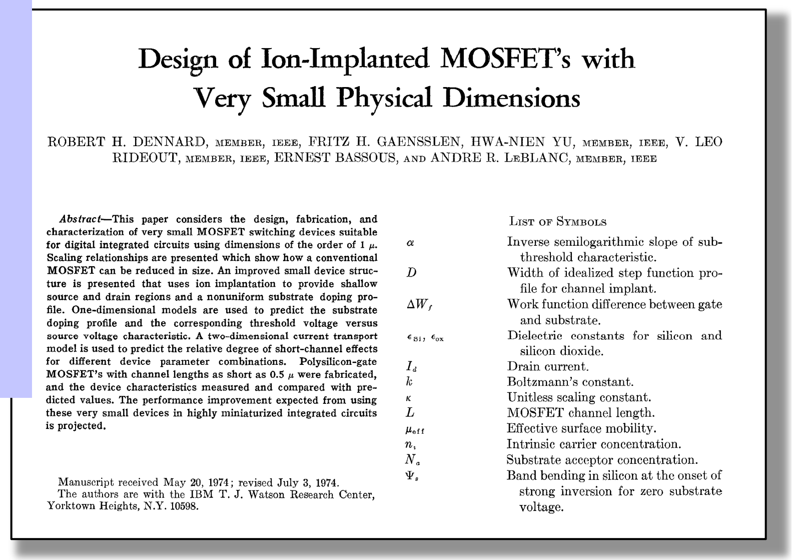
2X transistors/Chip Every 1.5 years

Called "Moore's Law"

Moore’s *Secret Sauce*: Dennard Scaling

Moore’s Law put lots more transistors on a chip…but it’s Dennard’s Law that made them useful

Dennard Scaling :
- Decrease feature size by a factor of λ and decrease voltage by a factor of λ ; then
- # transistors increase by λ^2
- Clock speed increases by λ
- **Energy consumption does not change**



Design of Ion-Implanted MOSFET’s with Very Small Physical Dimensions

ROBERT H. DENNARD, MEMBER, IEEE, FRITZ H. GAENSSLEN, HWA-NIEN YU, MEMBER, IEEE, V. LEO RIDEOUT, MEMBER, IEEE, ERNEST BASSOUS, AND ANDRE R. LEBLANC, MEMBER, IEEE

Abstract—This paper considers the design, fabrication, and characterization of very small MOSFET switching devices suitable for digital integrated circuits using dimensions of the order of 1 μ. Scaling relationships are presented which show how a conventional MOSFET can be reduced in size. An improved small device structure is presented that uses ion implantation to provide shallow source and drain regions and a nonuniform substrate doping profile. One-dimensional models are used to predict the substrate doping profile and the corresponding threshold voltage versus source voltage characteristic. A two-dimensional current transport model is used to predict the relative degree of short-channel effects for different device parameter combinations. Polysilicon-gate MOSFET’s with channel lengths as short as 0.5 μ were fabricated, and the device characteristics measured and compared with predicted values. The performance improvement expected from using these very small devices in highly miniaturized integrated circuits is projected.

Manuscript received May 20, 1974; revised July 3, 1974.
The authors are with the IBM T. J. Watson Research Center, Yorktown Heights, N.Y. 10598.

LIST OF SYMBOLS

α	Inverse semilogarithmic slope of subthreshold characteristic.
D	Width of idealized step function profile for channel implant.
ΔW_f	Work function difference between gate and substrate.
ϵ_{Si}, ϵ_{ox}	Dielectric constants for silicon and silicon dioxide.
I_d	Drain current.
k	Boltzmann’s constant.
κ	Unitless scaling constant.
L	MOSFET channel length.
μ_{eff}	Effective surface mobility.
n_i	Intrinsic carrier concentration.
N_a	Substrate acceptor concentration.
Ψ_s	Band bending in silicon at the onset of strong inversion for zero substrate voltage.

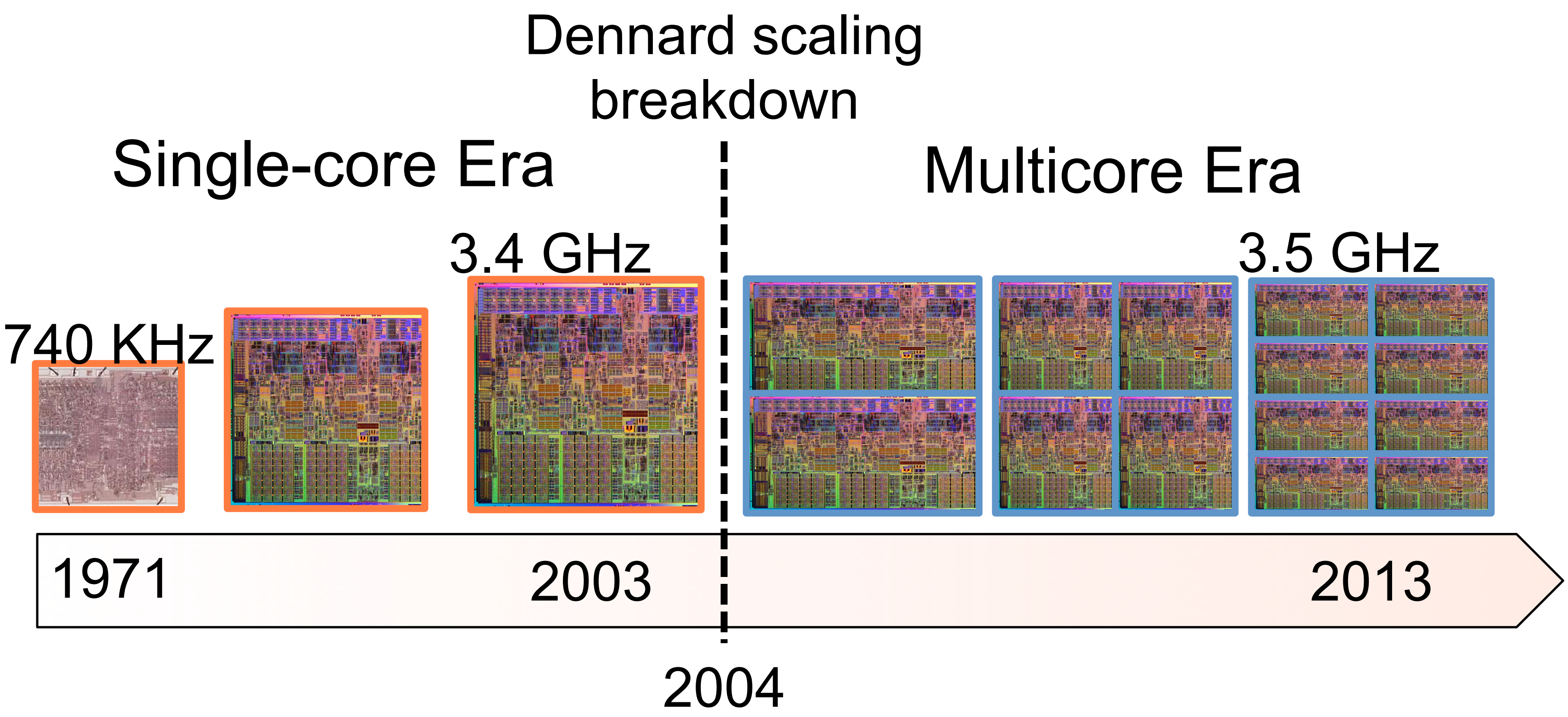
2x transistor count
40% faster
50% more efficient

[Dennard, Gaensslen, Yu, Rideout, Bassous, Leblanc, **IEEE JSSC**, 1974]

Dennard Scaling Over

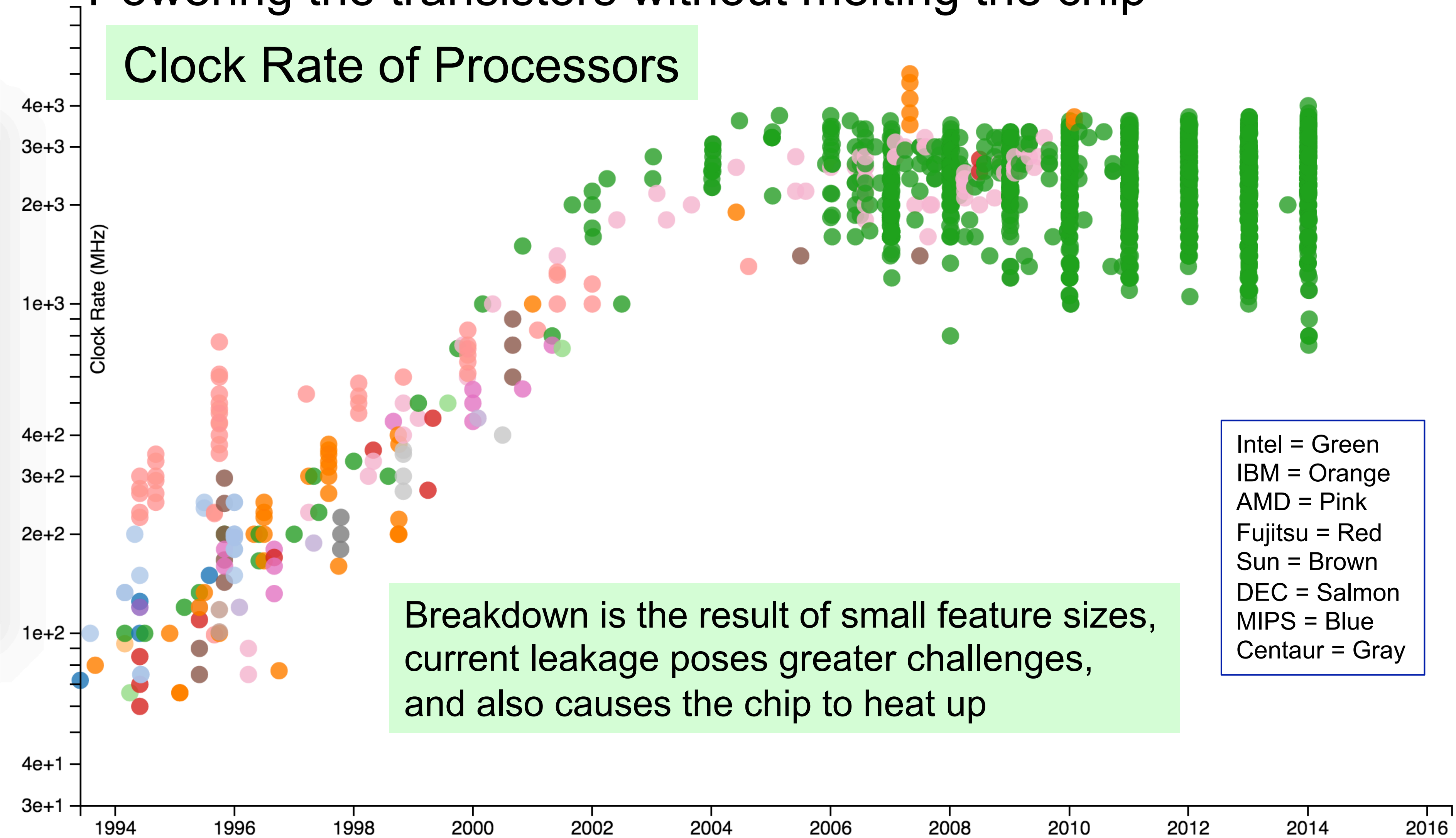
Evolution of processors

The primary reason cited for the breakdown is that at small sizes, current leakage poses greater challenges, and also causes the chip to heat up, which creates a threat of thermal runaway and therefore further increases energy costs.



Unfortunately Dennard Scaling is Over: What is the Catch?

Powering the transistors without melting the chip



CPU DB: recording microprocessor history, CACM, V 55 N 4, 2012, http://dl.acm.org/citation.cfm?id=2133822

Power Cost of Frequency

- Power ∝ Voltage^2 x Frequency   (V^2F)
- Frequency ∝ Voltage
- Power ∝ Frequency^3

	Cores	V	Freq	Perf	Power	PE (Bops/watt)
Superscalar	1	1	1	1	1	1
"New" Superscalar	1X	1.5X	1.5X	1.5X	3.3X	0.45X

Power Cost of Frequency

- **Power ∝ Voltage^2 x Frequency (V^{2}F)**
- **Frequency ∝ Voltage**
- **Power ∝ Frequency^3**

	Cores	V	Freq	Perf	Power	PE (Bops/watt)
Superscalar	1	1	1	1	1	1
"New" Superscalar	1X	1.5X	1.5X	1.5X	3.3X	0.45X
Multicore	2X	0.75X	0.75X	1.5X	0.8X	1.88X

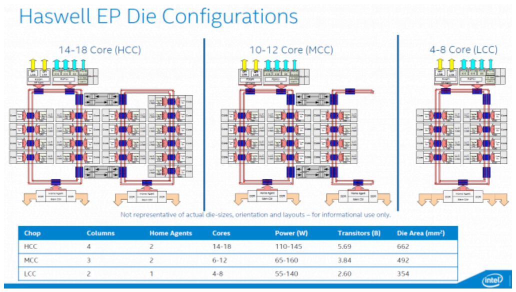
(Bigger # is better)

50% more performance with 20% less power

Preferable to use multiple slower devices, than one superfast device

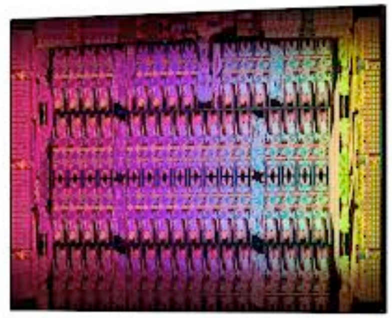
Today's Multicores

All of Top500 Systems Are Based on Multicore

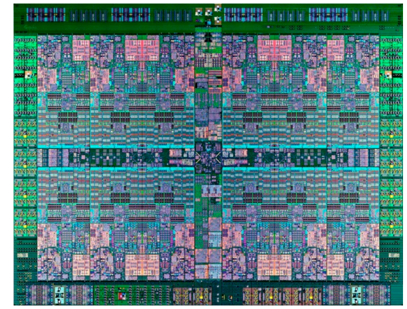


Chop	Columns	Home Agents	Cores	Power (W)	Transitors (B)	Die Area (mm²)
HCC	4	2	14-18	110-145	5.69	662
MCC	3	2	6-12	65-160	3.84	492
LCC	2	1	4-8	55-140	2.60	354

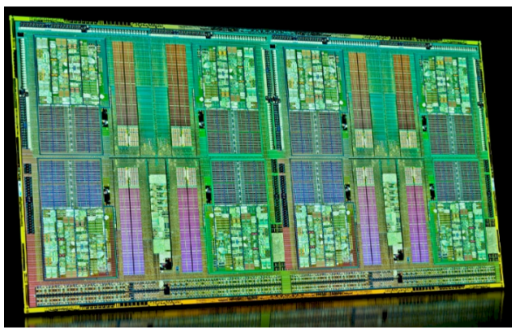
Intel Haswell (18 cores)



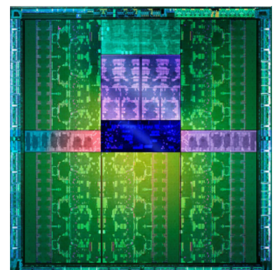
Intel Xeon Phi (60 cores)



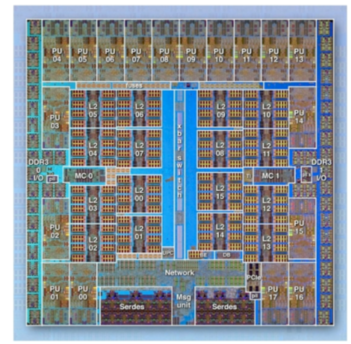
IBM Power 8 (12 cores)



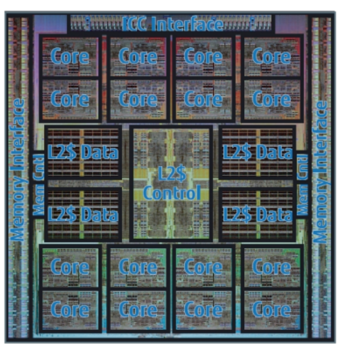
AMD Interlagos (16 cores)



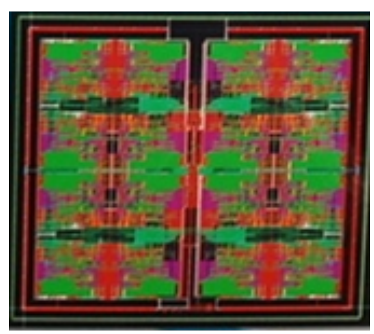
Nvidia Kepler (2688 Cuda cores)



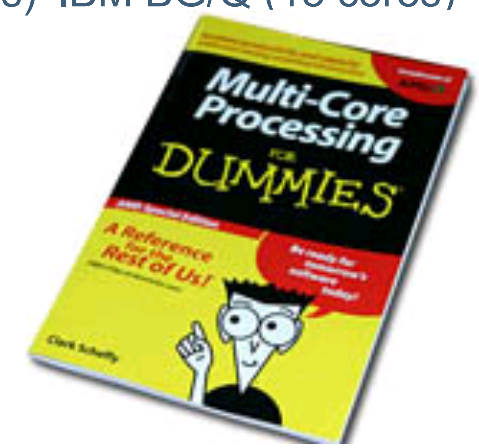
IBM BG/Q (18 cores)



Fujitsu Venus (16 cores)



ShenWei (16 core)

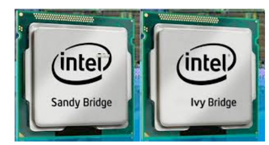


Peak Performance - Per Core

$$\text{FLOPS} = \text{cores} \times \text{clock} \times \frac{\text{FLOPs}}{\text{cycle}}$$

Floating point operations per cycle per core

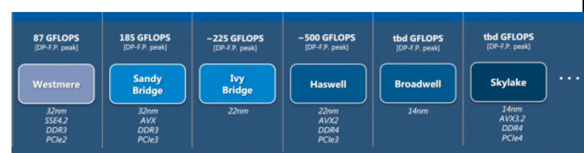
- Most of the recent computers have FMA (Fused multiple add): (i.e. x ←x + y*z in one cycle)
- Intel Xeon earlier models and AMD Opteron have SSE2
  - 2 flops/cycle DP & 4 flops/cycle SP
- Intel Xeon Nehalem ('09) & Westmere ('10) have SSE4
  - 4 flops/cycle DP & 8 flops/cycle SP
- Intel Xeon Sandy Bridge('11) & Ivy Bridge ('12) have AVX
  - 8 flops/cycle DP & 16 flops/cycle SP



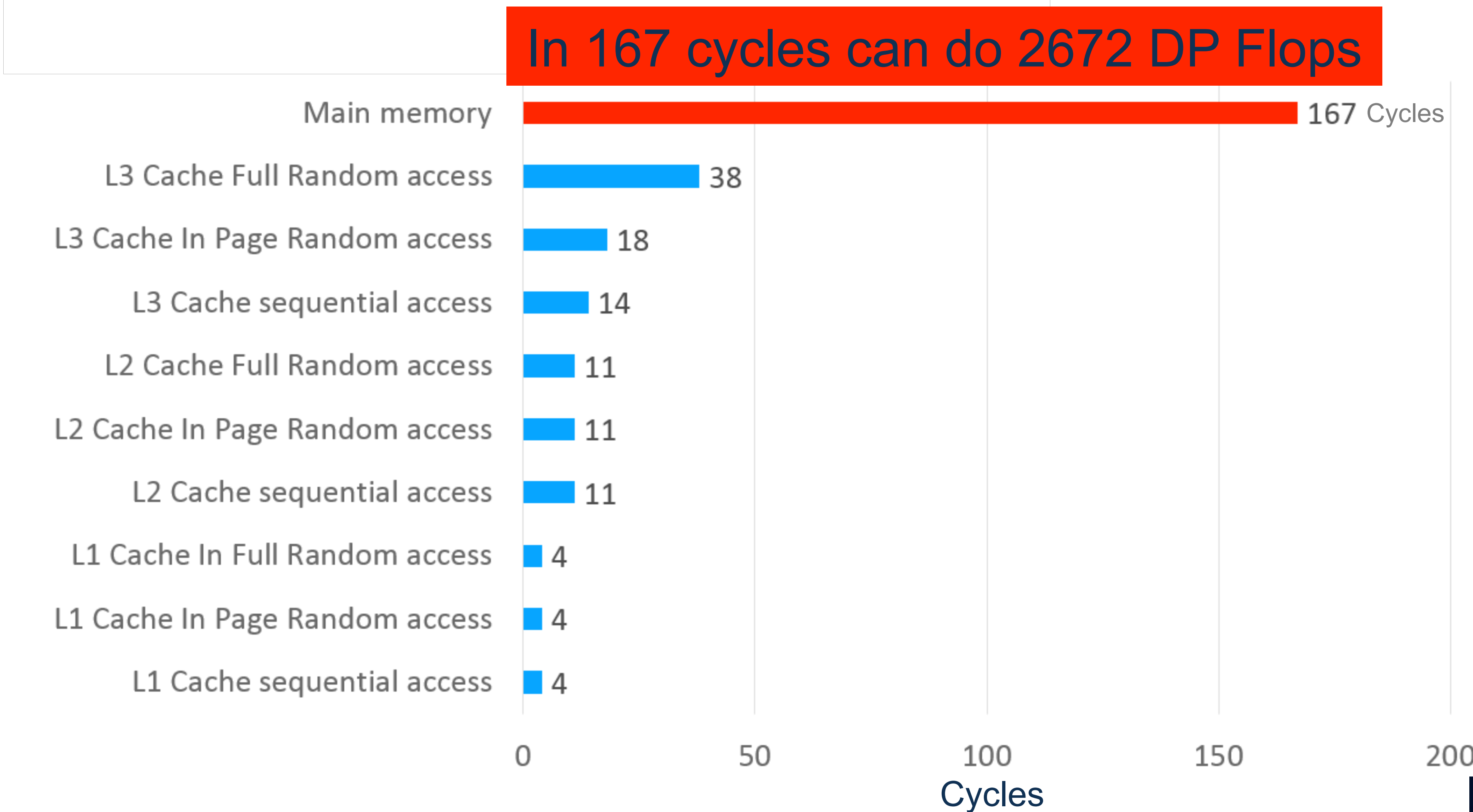
- Intel Xeon Haswell ('13) & (Broadwell ('14)) AVX2
  - 16 flops/cycle DP & 32 flops/cycle SP
  - Xeon Phi (per core) is at 16 flops/cycle DP & 32 flops/cycle SP

We are here

- Intel Xeon Skylake ('15)
  - 32 flops/cycle DL & 64 flops/cycle SP



CPU Access Latencies in Clock Cycles



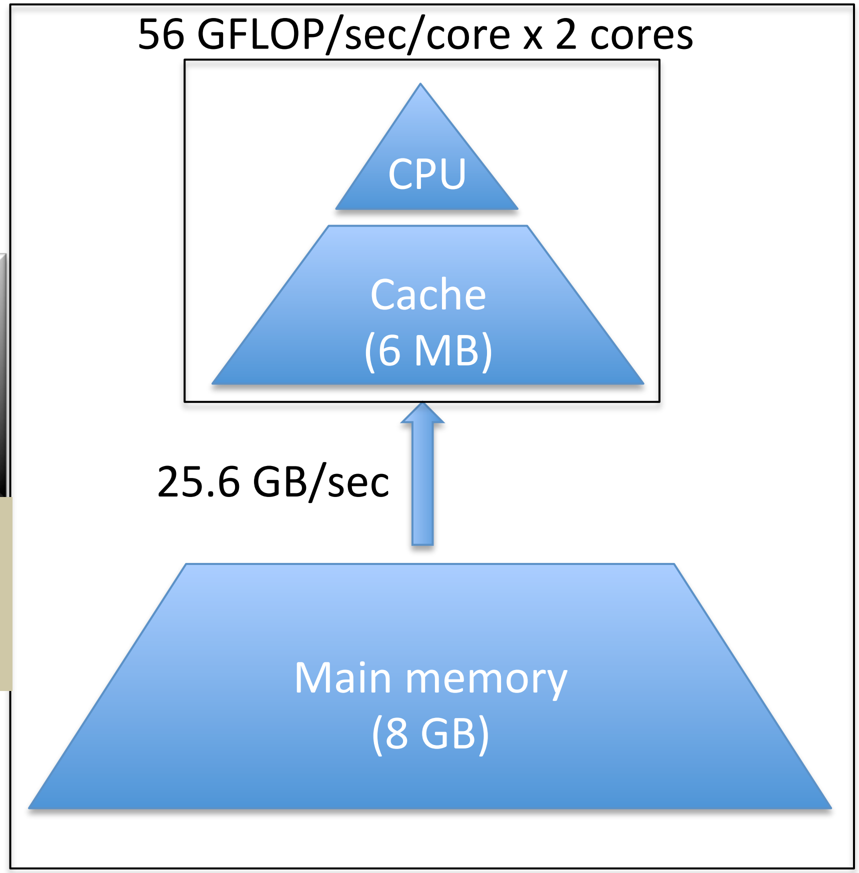
Memory transfer

(Its All About Data Movement)

Example on my laptop: One level of memory

Intel Core i7 4850HQ
Haswell, 2.3 GHz
Turbo Boost 3.5 GHz

16 flops/cycle *
3.5 *10^9 cycles/sec =
56 Gflop/s

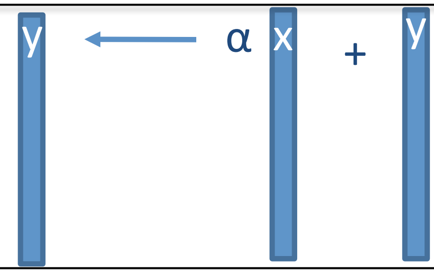


(Omitting latency here.)

The model IS simplified (see next slide) but it provides an upper bound on performance as well. I.e., we will never go faster than what the model predicts. (And, of course, we can go slower …)

FMA: fused multiply-add

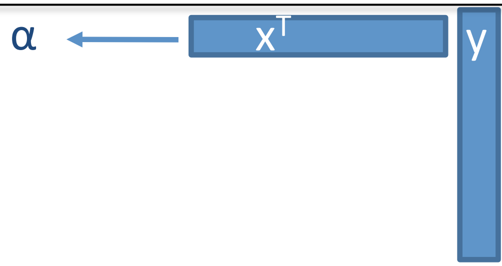
AXPY:




```
for ( j = 0; j < n; j++)
      y[i] += a * x[i];

(without increment)
```

n MUL
n ADD
2n FLOP
n FMA

DOT:



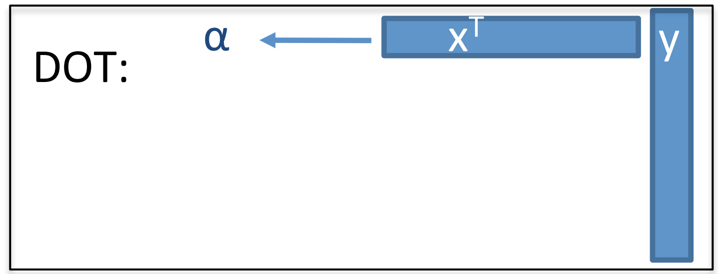

```
alpha = 0e+00;
for ( j = 0; j < n; j++)
      alpha += x[i] * y[i];

(without increment)
```

n MUL
n ADD
2n FLOP
n FMA

Note: It is reasonable to expect the one loop codes shown here to perform as well as their Level 1 BLAS counterpart (on multicore with an OpenMP pragma for example).

The true gain these days with using the BLAS is (1) Level 3 BLAS, and (2) portability.

- Take two double precision vectors x and y of size n=375,000.



- Data size:
  – (375,000 double) * (8 Bytes / double) = 3 MBytes per vector

(Two vectors fit in cache (6 MBytes). OK.)

- Time to move the vectors from memory to cache:
  – (6 MBytes) / (25.6 GBytes/sec) = **0.23 ms**

- Time to perform computation of DOT:
  – (2n flop) / (56 Gflop/sec) = **0.01 ms**

Vector Operations

total_time ≥ max (time_comm , time_comp)

= max (0.23ms , 0.01ms) = 0.23ms

Performance = (2 x 375,000 flops)/.23ms = 3.2 Gflop/s

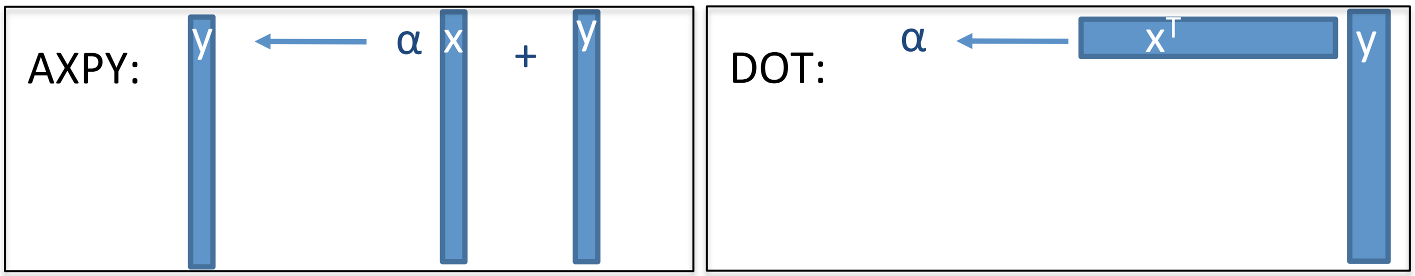
Performance for DOT ≤ 3.2 Gflop/s

Peak is 56 Gflop/s

We say that the operation is communication bounded. No reuse of data.

Level 1, 2 and 3 BLAS

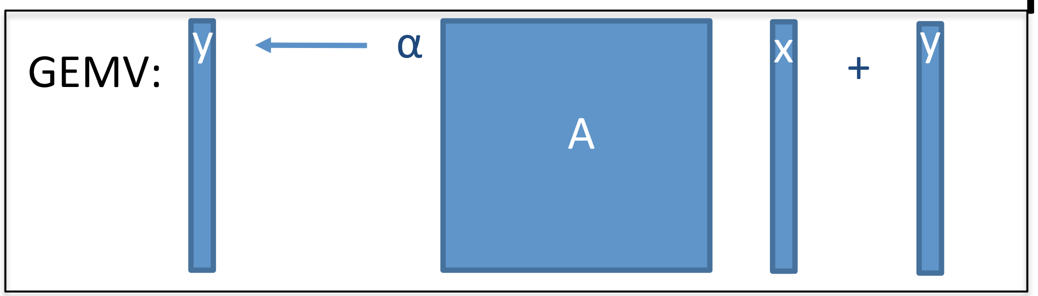
Level 1 BLAS Matrix-Vector operations



2n FLOP
2n memory reference
AXPY: 2n READ, n WRITE
DOT: 2n READ

RATIO: 1

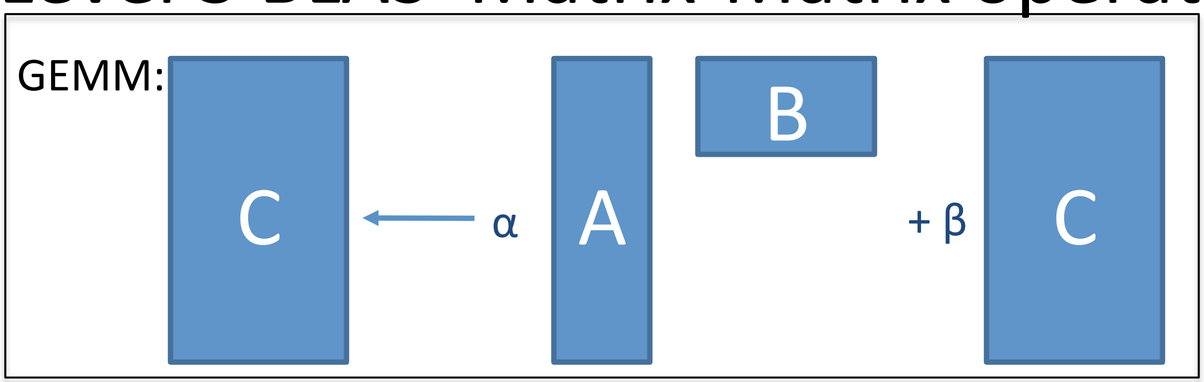
Level 2 BLAS Matrix-Vector operations



$2n^2$ FLOP
n^2 memory references

RATIO: 2

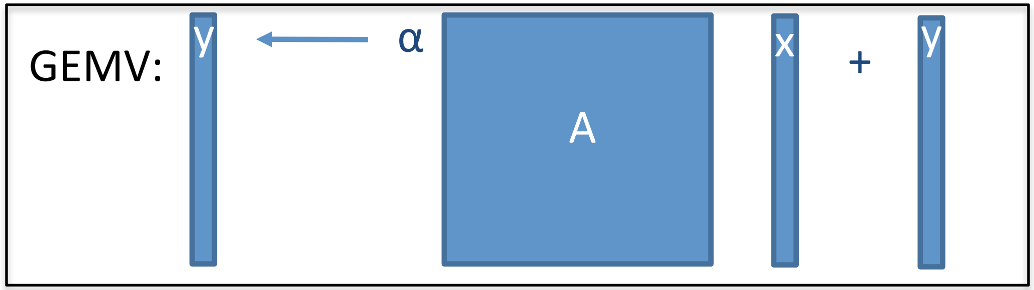
Level 3 BLAS Matrix-Matrix operations



$2n^3$ FLOP
$3n^2$ memory references
$3n^2$ READ, n^2 WRITE

RATIO: 2/3 n

- Double precision matrix A and vectors x and y of size n=860.



- Data size:
  – (860^2 + 2*860 double) * (8 Bytes / double) ~ 6 MBytes

Matrix and two vectors fit in cache (6 MBytes).

- Time to move the data from memory to cache:
  – (6 MBytes) / (25.6 GBytes/sec) = **0.23 ms**

- Time to perform computation of DOT:
  – ($2n^2$ flop) / (56 Gflop/sec) = **0.26 ms**

Matrix - Vector Operations

$$\text{total\_time} \geq \max(\text{time\_comm}, \text{time\_comp})$$
$$= \max(0.23\text{ms}, 0.26\text{ms}) = 0.26\text{ms}$$

Performance = $(2 \times 860^2$ flops$)/.26$ms = 5.7 Gflop/s

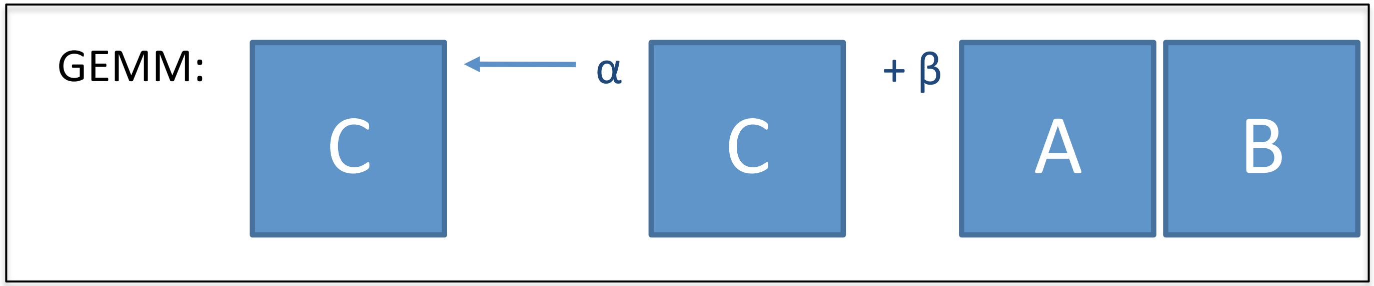
Performance for GEMV ≤ 5.7 Gflop/s

Performance for DOT ≤ 3.2 Gflop/s

Peak is 56 Gflop/s

We say that the operation is communication bounded. Very little reuse of data.

- Take two double precision vectors x and y of size n=500.



- Data size:
  - (500^2 double) * (8 Bytes / double) = 2 MBytes per matrix

  (Three matrices fit in cache (6 MBytes). OK.)

- Time to move the matrices in cache:
  - (6 MBytes) / (25.6 GBytes/sec) = **0.23 ms**

- Time to perform computation in GEMM:
  - ($2n^3$ flop) / (56 Gflop/sec) = **4.46 ms**

Matrix Matrix Operations

total_time ≥ max (time_comm , time_comp)

= max(0.23ms , 4.46ms) = 4.46ms

For this example, communication time is less than 6% of the computation time.

Performance = (2 x 500 ^{3} flops)/4.69ms = 53.3 Gflop/s

There is a lots of data reuse in a GEMM; 2/3n per data element. Has good temporal locality.

If we assume total_time ≈ time_comm +time_comp, we get

Performance for GEMM ≈ 53.3 Gflop/sec

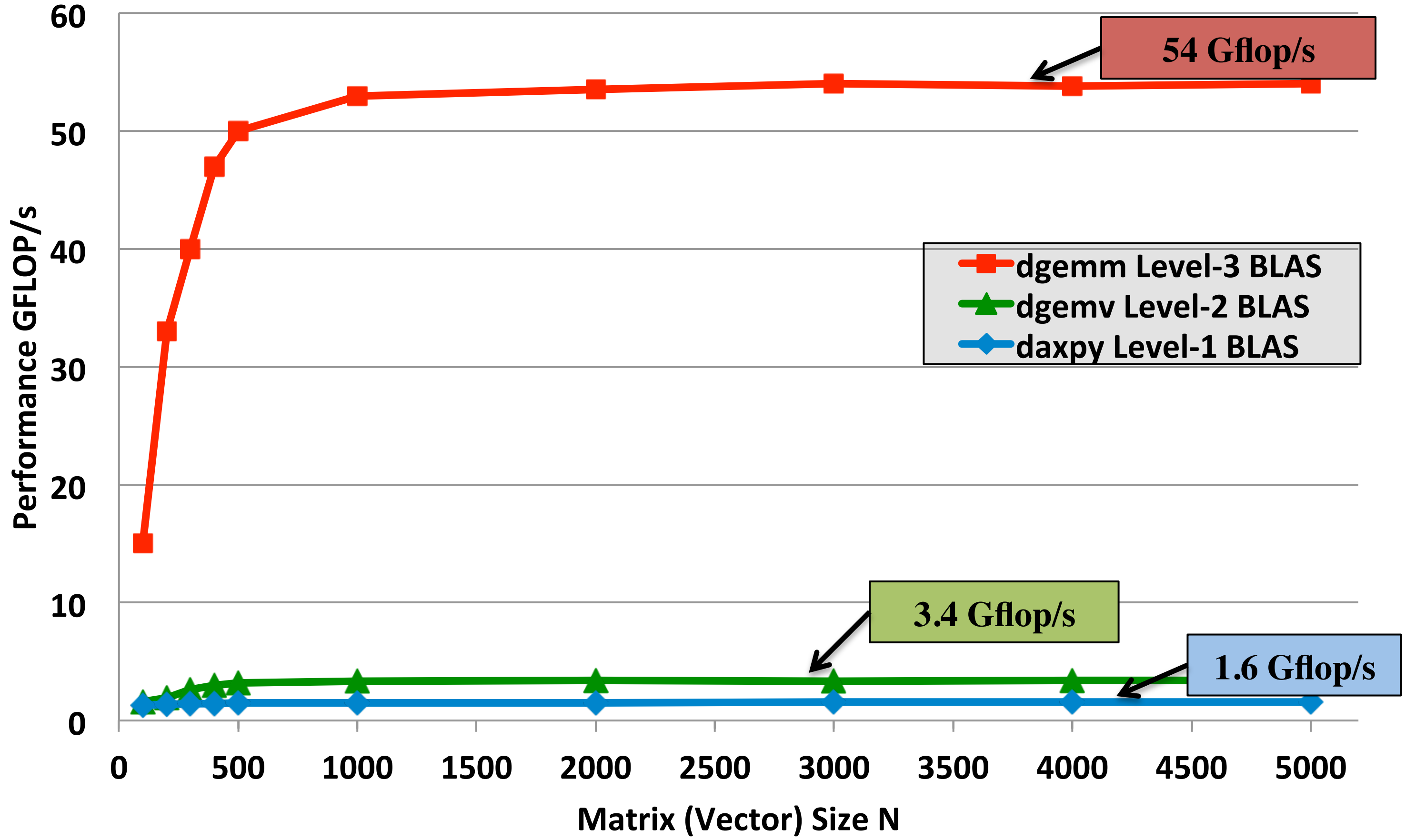
Performance for DOT ≤ 3.2 Gflop/s

Performance for GEMV ≤ 5.7 Gflop/s

(Out of 56 Gflop/sec possible, so that would be 95% peak performance efficiency.)

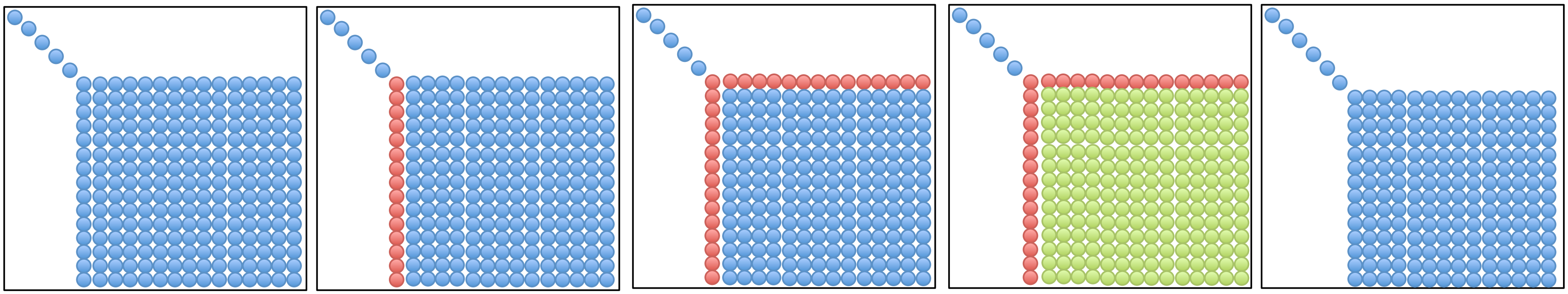
Level 1, 2 and 3 BLAS

1 core Intel Haswell i7-4850HQ, 2.3 GHz (Turbo Boost at 3.5 GHz);
Peak = 56 Gflop/s



1 core Intel Haswell i7-4850HQ, 2.3 GHz, Memory: DDR3L-1600MHz
6 MB shared L3 cache, and each core has a private 256 KB L2 and 64 KB L1.
The theoretical peak per core double precision is 56 Gflop/s per core.
Compiled with gcc and using Veclib

The Standard LU Factorization LINPACK 1970's HPC of the Day: Vector Architecture



Factor column with Level 1 BLAS

Divide by Pivot row

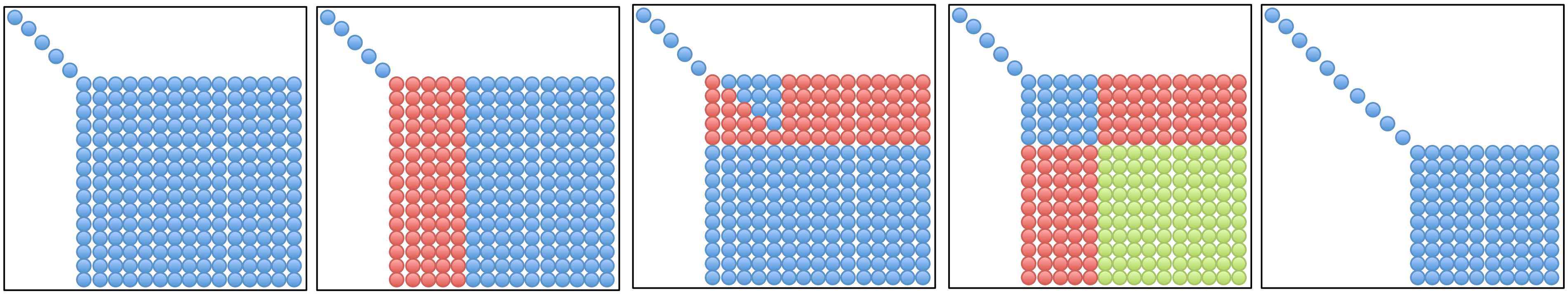
Schur complement update (Rank 1 update)

Next Step

Main points

- Factorization column (zero) mostly sequential due to memory bottleneck
- Level 1 BLAS
- Divide pivot row has little parallelism
- Rank -1 Schur complement update is the only easy parallelize task
- Partial pivoting complicates things even further
- Bulk synchronous parallelism (fork-join)
  - Load imbalance
  - Non-trivial Amdahl fraction in the panel
  - Potential workaround (look-ahead) has complicated implementation

The Standard LU Factorization LAPACK 1980's HPC of the Day: Cache Based SMP



Factor panel with Level 1,2 BLAS

Triangular update

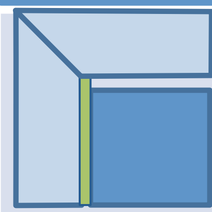
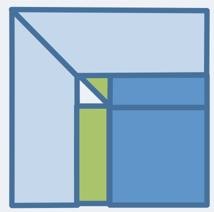
Schur complement update

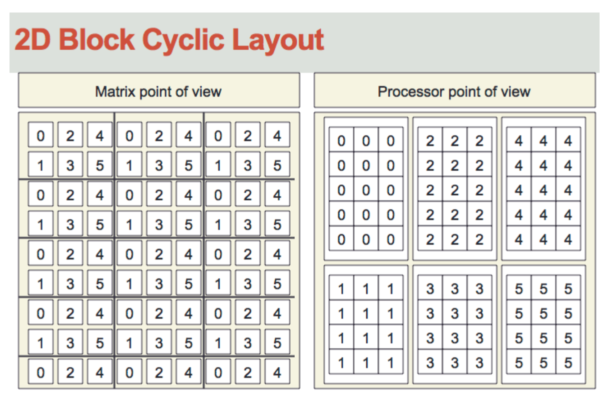
Next Step

Main points

- Panel factorization mostly sequential due to memory bottleneck
- Triangular solve has little parallelism
- Schur complement update is the only easy parallelize task
- Partial pivoting complicates things even further
- Bulk synchronous parallelism (fork-join)
    - Load imbalance
    - Non-trivial Amdahl fraction in the panel
    - Potential workaround (look-ahead) has complicated implementation

Last Generations of DLA Software

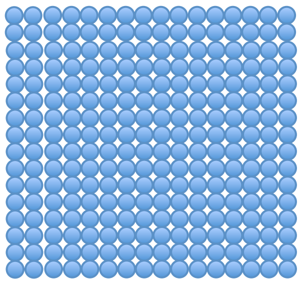
Software/Algorithms follow hardware evolution in time		
LINPACK (70's) (Vector operations)		Rely on - Level-1 BLAS operations
LAPACK (80's) (Blocking, cache friendly)		Rely on - Level-3 BLAS operations
ScaLAPACK (90's) (Distributed Memory)		Rely on - PBLAS Mess Passing



Classical Analysis of Algorithms May Not be Valid

- Processors over provisioned for floating point arithmetic
- Data movement extremely expensive
- Operation count is not a good indicator of the time to solve a problem.
- Algorithms that do more ops may actually take less time.

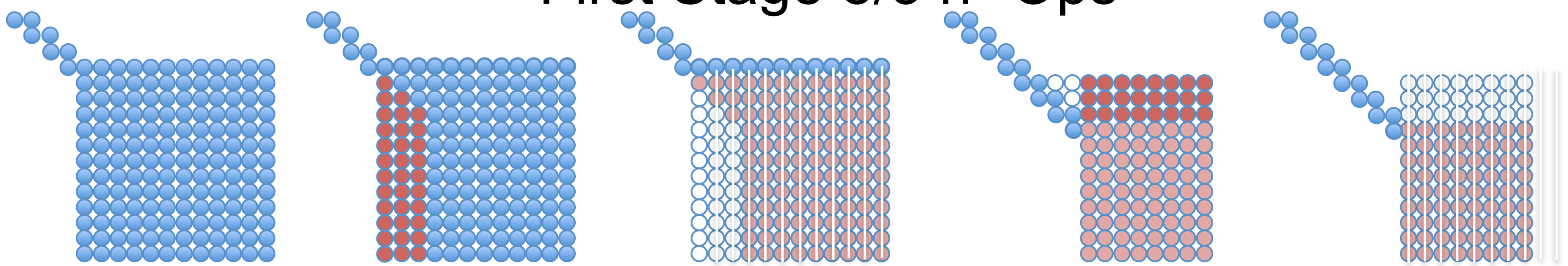
Singular Value Decomposition

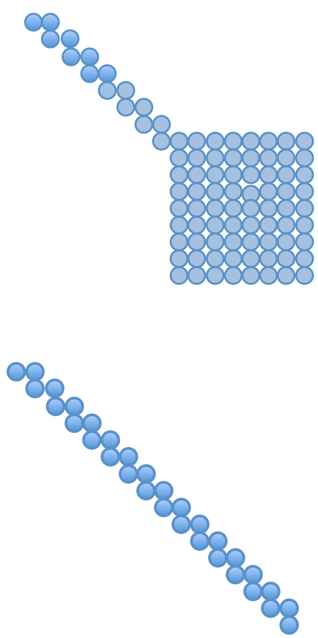


LAPACK Version 1991

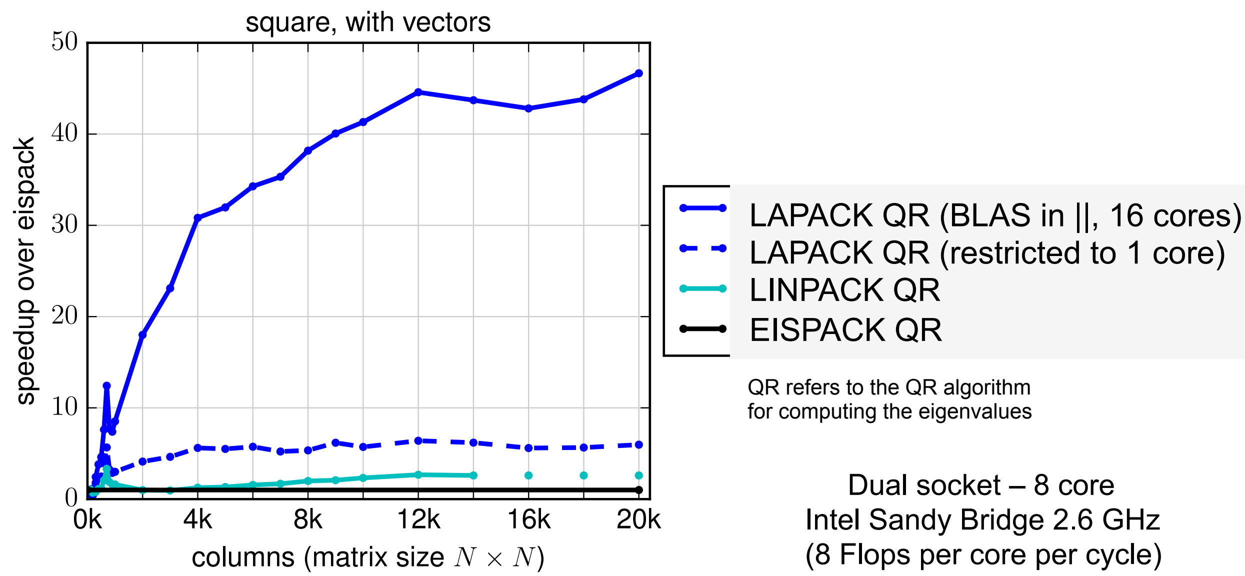
Level 1, 2, & 3 BLAS

First Stage 8/3 n^3 Ops





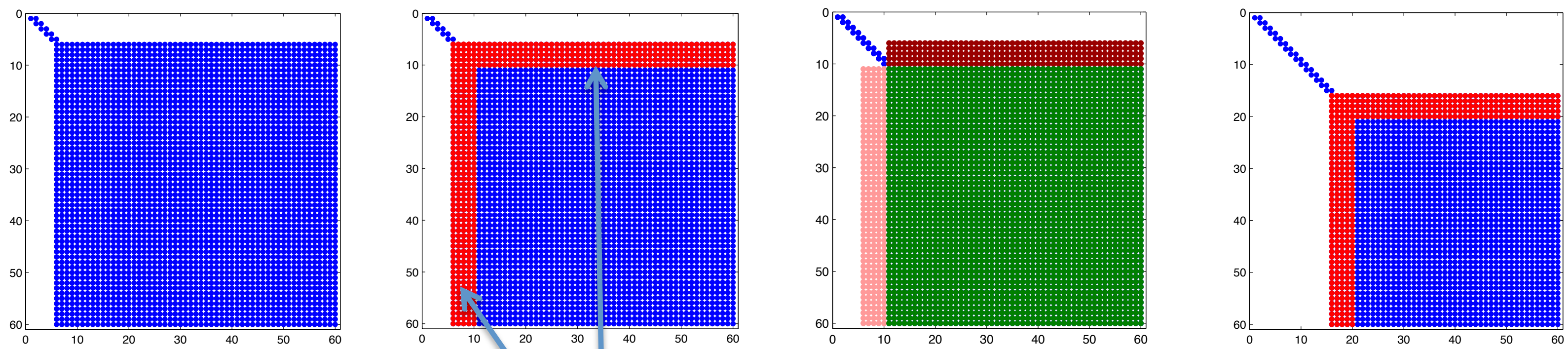
3 Generations of software compared



QR refers to the QR algorithm
for computing the eigenvalues

Dual socket – 8 core
Intel Sandy Bridge 2.6 GHz
(8 Flops per core per cycle)

Bottleneck in the Bidiagonalization
The Standard Bidiagonal Reduction: xGEBRD
Two Steps: Factor Panel & Update Tailing Matrix



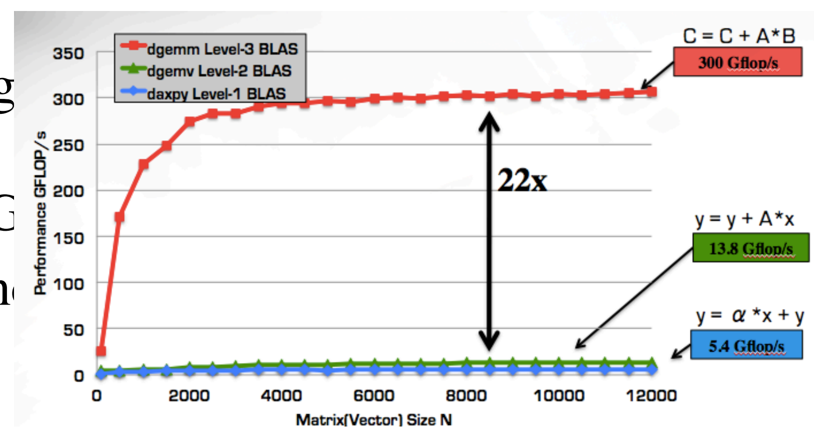
factor panel k then update ➔ **factor panel k+1**

Requires 2 GEMVs

$Q * A * P^H$

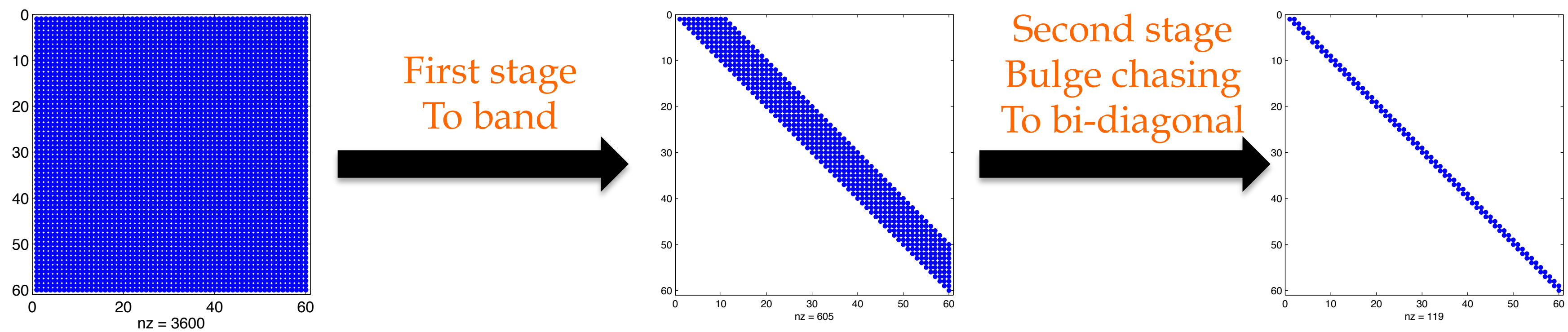
✷ Characteristics

- Total cost $8n^3/3$, (reduction to bi-diag
- Too many Level 2 BLAS operations
- $4/3\ n^3$ from GEMV and $4/3\ n^3$ from G
- Performance limited to 2* performan
- **➔Memory bound algorithm.**



16 cores Intel Sandy Bridge, 2.6 GHz, 20 MB shared L3 cache.
The theoretical peak per core double precision is 20.4 Gflop/s per core.
Compiled with icc and using MKL 2015.3.187

Recent Work on 2-Stage Algorithm

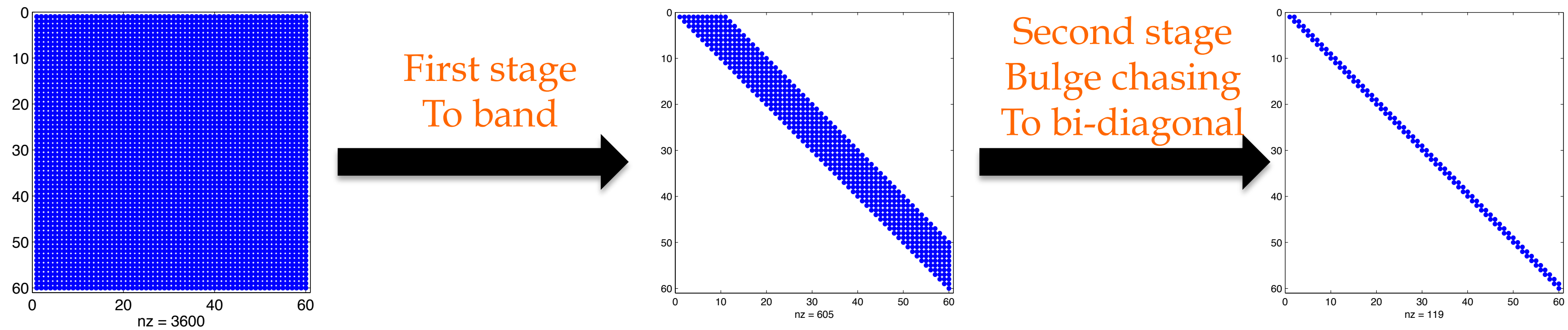


✸ Characteristics

- **Stage 1:**
  - Fully Level 3 BLAS
  - Dataflow Asynchronous execution

- **Stage 2:**
  - Level "BLAS-1.5"
  - Asynchronous execution
  - Cache friendly kernel (reduced communication)

Recent work on developing new 2-stage algorithm



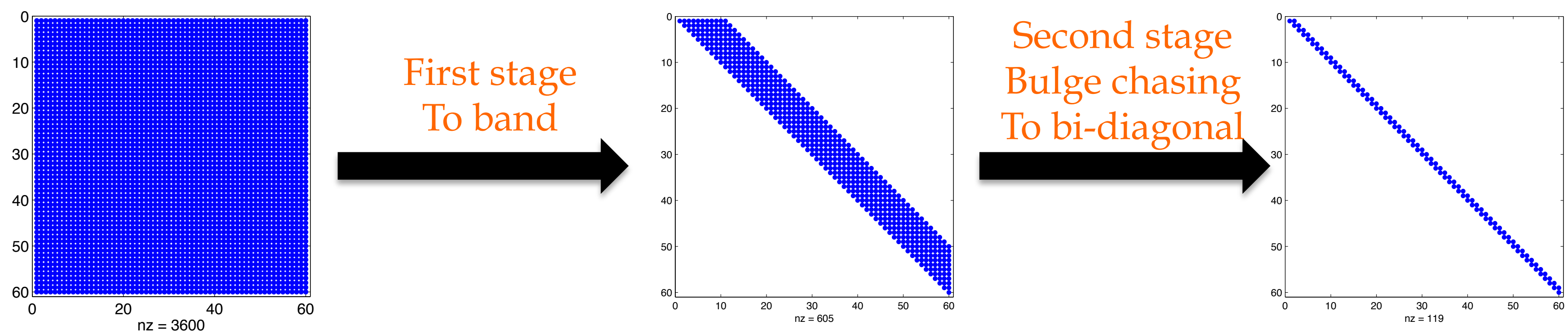
$$
\begin{aligned}
\text{flops} \approx\ & \sum_{s=1}^{\frac{n-n_b}{n_b}} 2n_b^3 + (nt-s)3n_b^3 + (nt-s)\frac{10}{3}n_b^3 + (nt-s)\times(nt-s)5n_b^3 \\
+ & \sum_{s=1}^{\frac{n-n_b}{n_b}} 2n_b^3 + (nt-s-1)3n_b^3 + (nt-s-1)\frac{10}{3}n_b^3 + (nt-s)\times(nt-s-1)5n_b^3 \\
\approx\ & \frac{10}{3}n^3 + \frac{10n_b}{3}n^2 + \frac{2n_b}{3}n^3
\end{aligned}
$$

$$\text{flops} \approx \frac{10}{3}n^3(\text{gemm})_{\text{first stage}} \qquad \text{flops} = 6\times n_b \times n^2(\text{gemv})_{\text{second stage}}$$

More Flops, original did 8/3 n^3

25% More flops

Recent work on developing new 2-stage algorithm

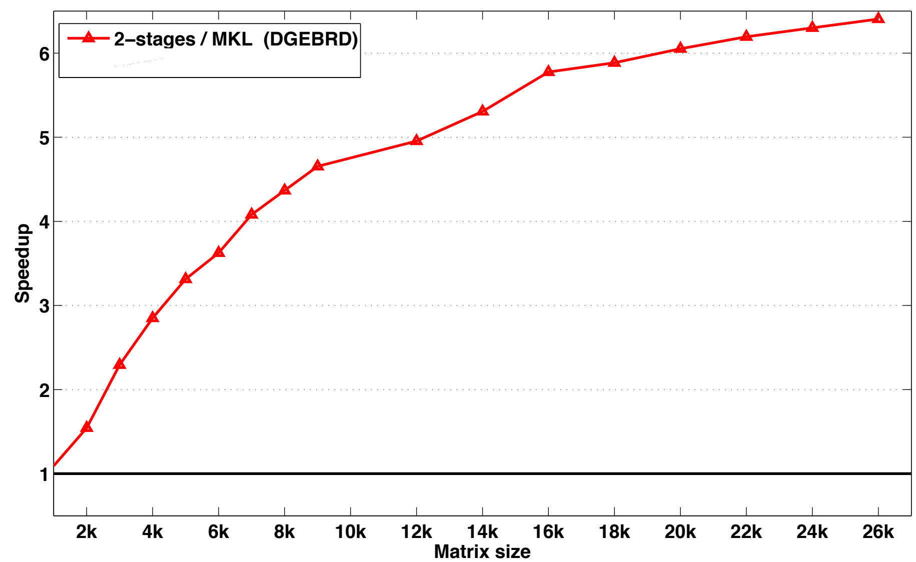


$$\textbf{speedup} = \frac{\text{time of one-stage}}{\text{time of two-stage}}$$

$$= \frac{4n^3/3P_{gemv} + 4n^3/3P_{gemm}}{10n^3/3P_{gemm} + 6n_b n^2/P_{gemv}}$$

$$\Longrightarrow \frac{84}{70} \le \textbf{Speedup} \le \frac{84}{15}$$

$$\Longrightarrow 1.8 \le \textbf{Speedup} \le 7$$



16 Sandy Bridge cores 2.6 GHz

if P_{gemm} is about 22x P_{gemv} and $120 \le n_b \le 240$.

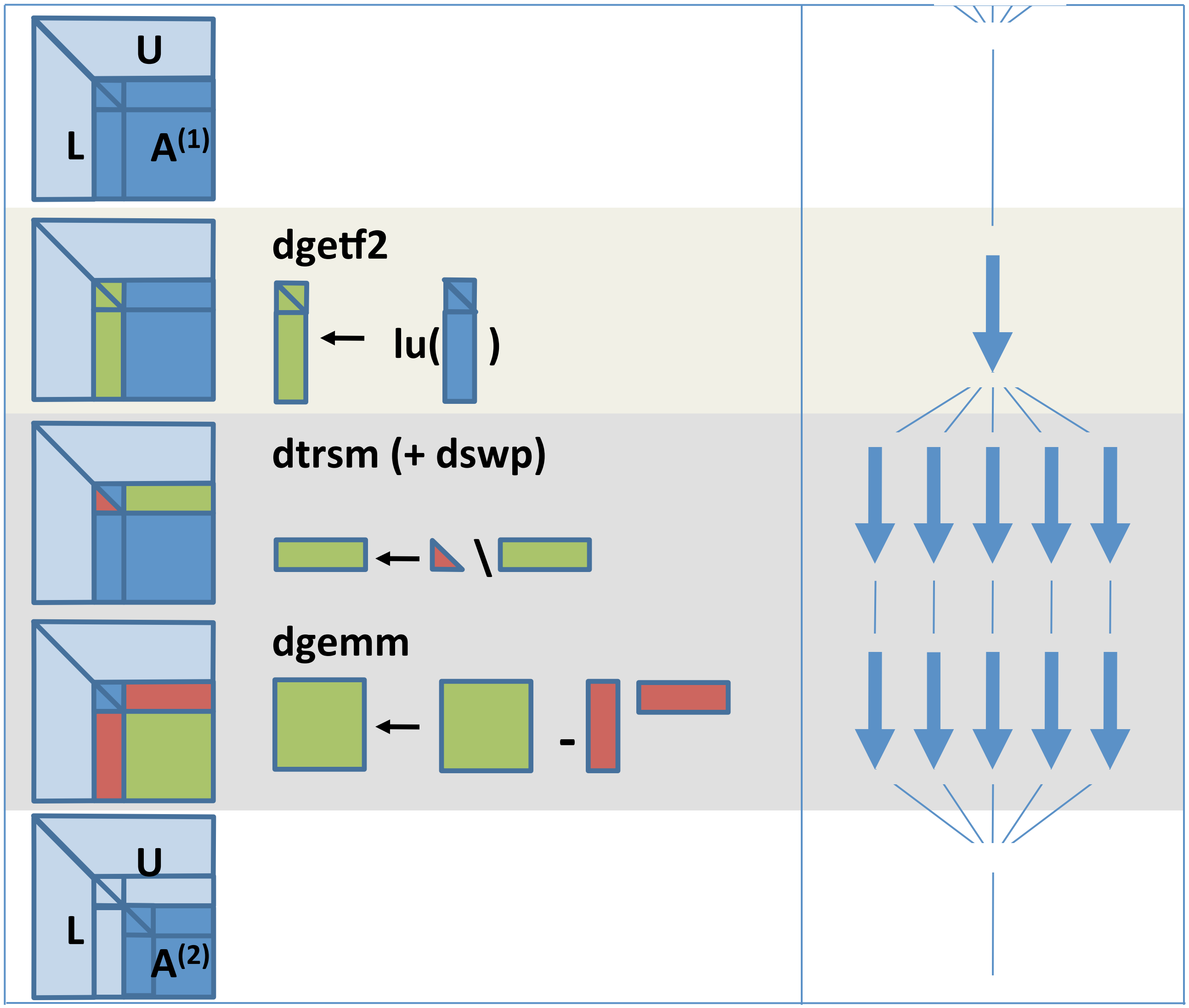
25% More flops and 1.8 – 7 times faster

Parallelization of LU and QR.

Parallelize the update:

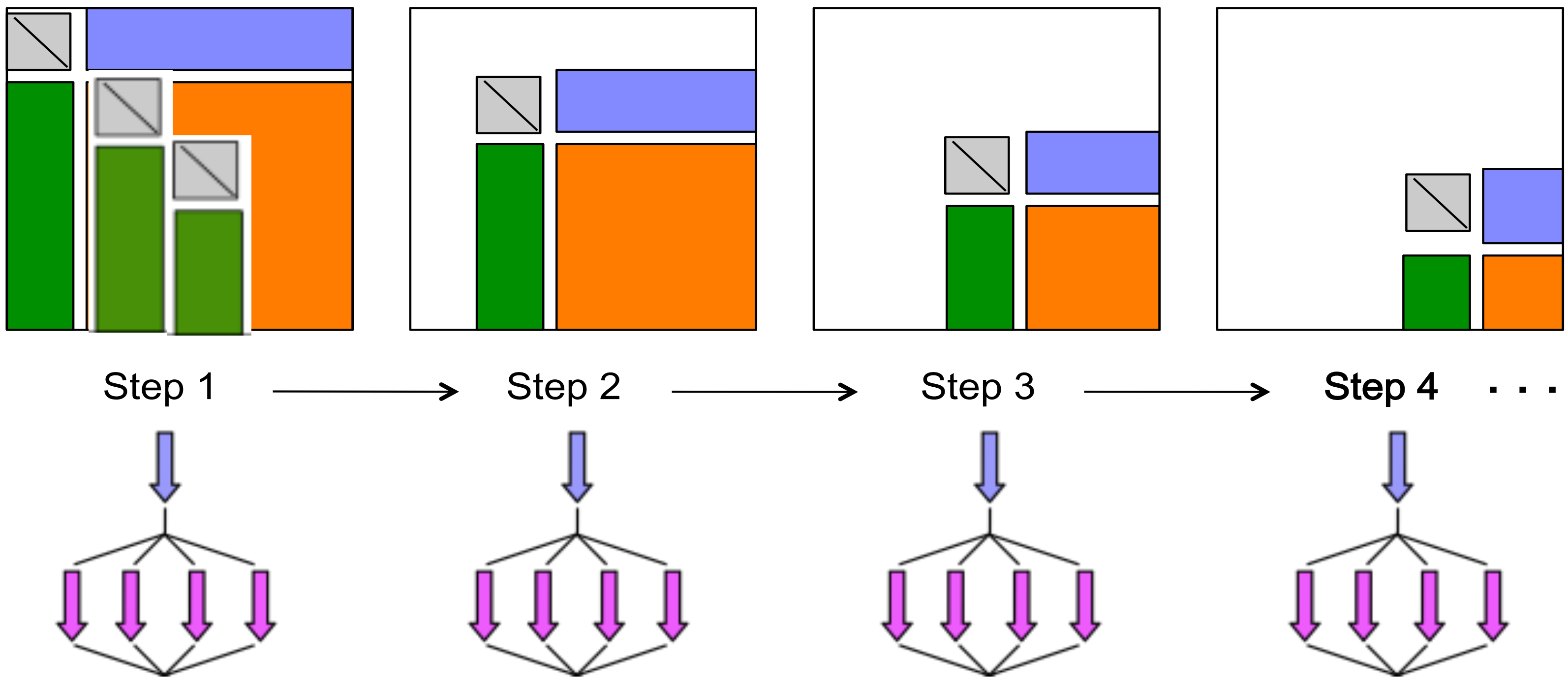
- Easy and done in any reasonable software.
- This is the $2/3n^3$ term in the FLOPs count.
- Can be done efficiently with LAPACK+multithreaded BLAS

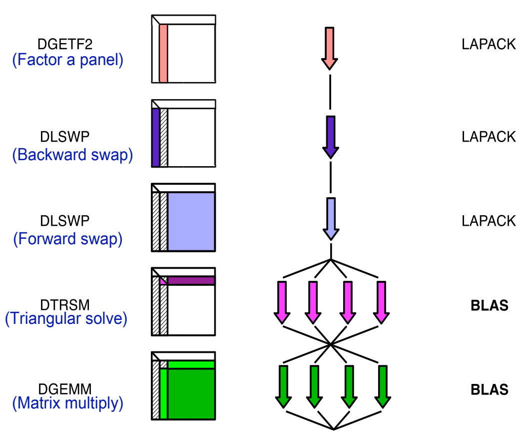
dgemm



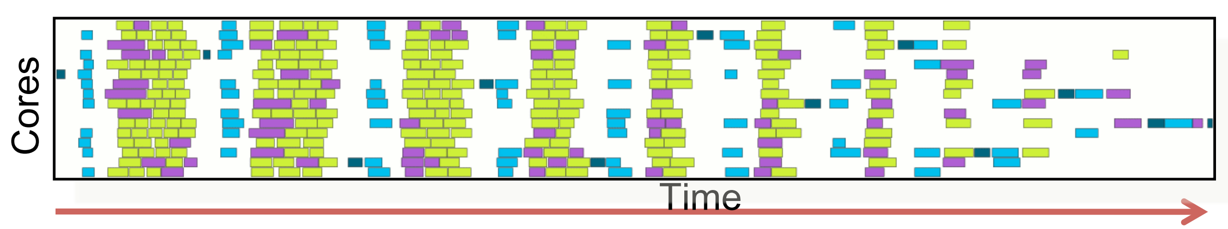
Fork - Join parallelism
Bulk Sync Processing

Synchronization (in LAPACK LU)





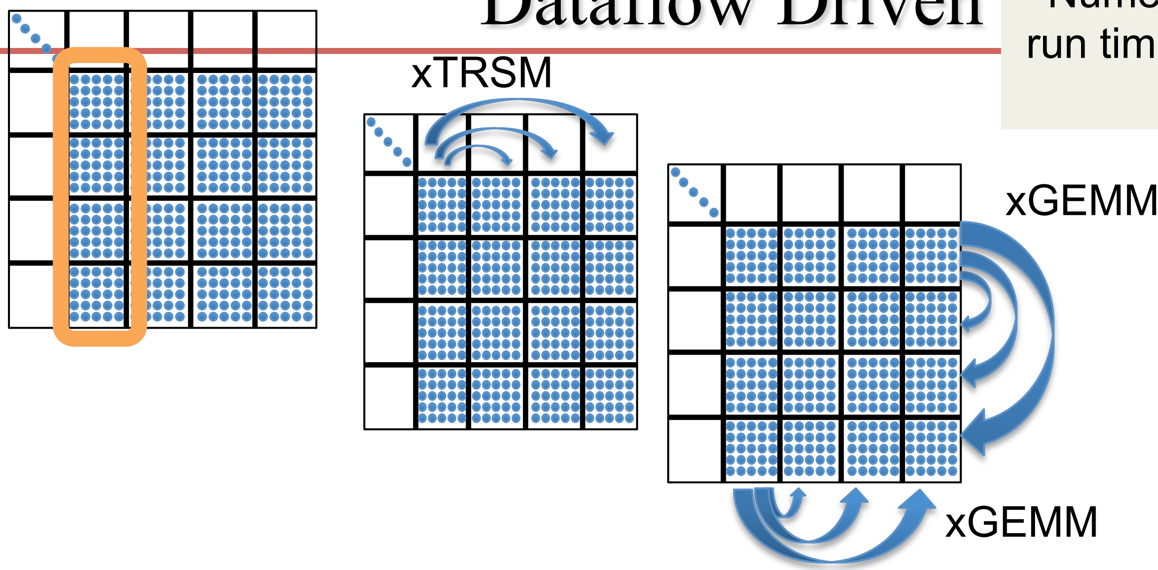
- ➢ fork join
- ➢ bulk synchronous processing



PLASMA LU Factorization

Dataflow Driven

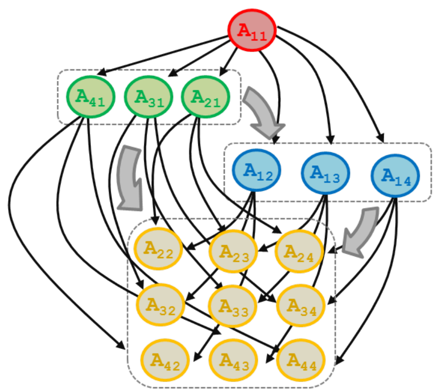
Numerical program generates tasks and run time system executes tasks respecting data dependences.



Sparse / Dense Matrix System

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix}$$

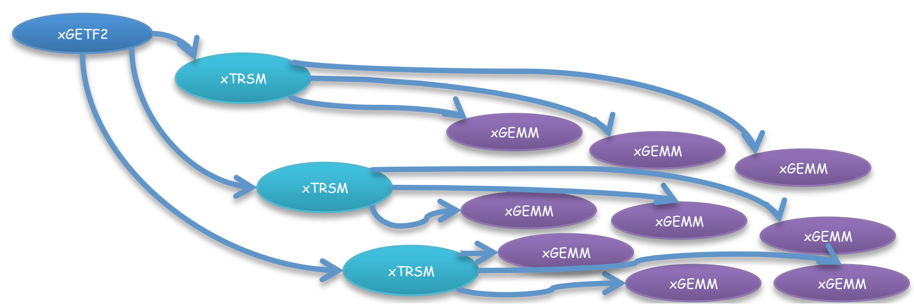
DAG-based factorization



Batched LA

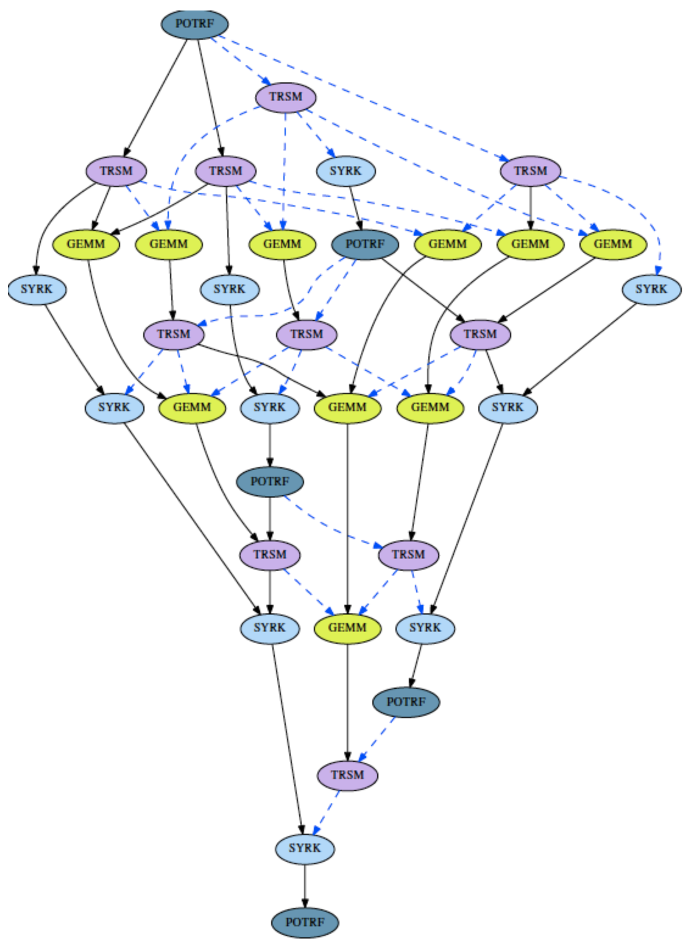
- **LU, QR,** or **Cholesky** on small diagonal matrices
- **TRSM**s, **QR**s, or **LU**s
- **TRSM**s, **TRMM**s
- **Updates (Schur complement) GEMM**s, **SYRK**s, **TRMM**s

And many other BLAS/LAPACK, e.g., for application specific solvers, preconditioners, and matrices

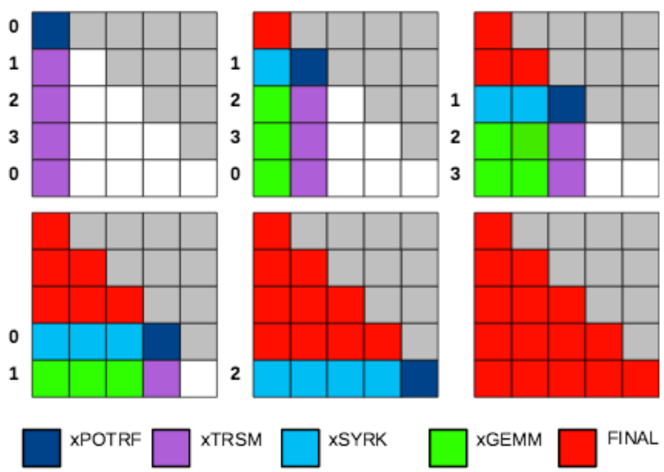


OpenMP Tasking

- Added with OpenMP 3.0 (2009)
- Allows parallelization of irregular problems
- OpenMP 4.0 (2013) - Tasks can have dependencies
  - DAGs



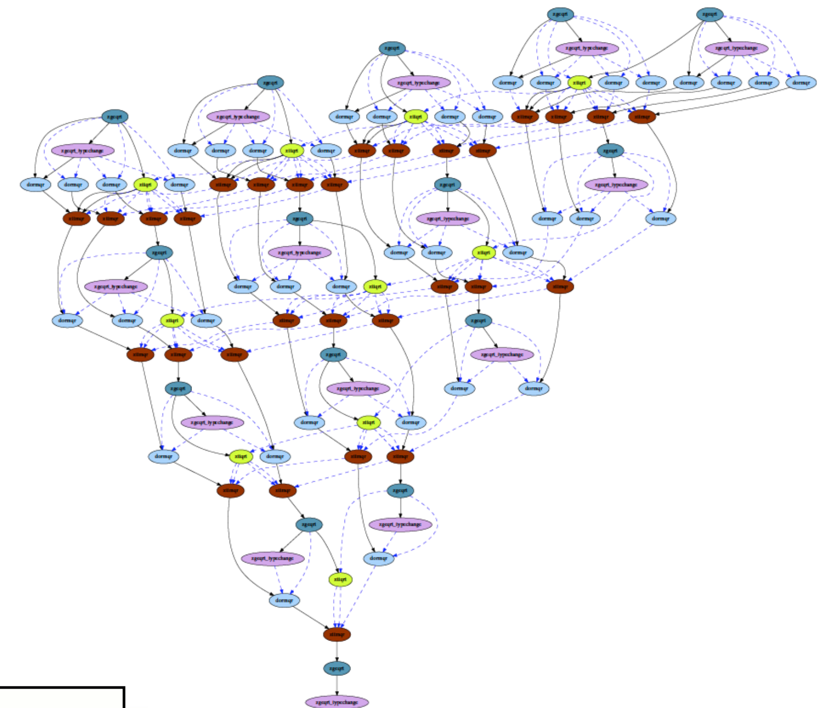
Tiled Cholesky Decomposition

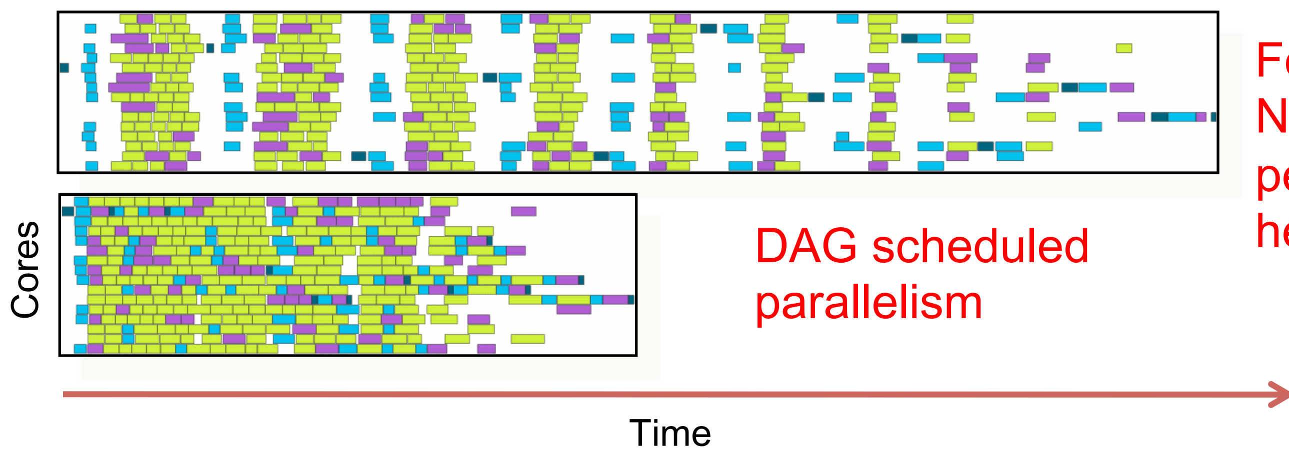



```
#pragma omp parallel
#pragma omp master
{  CHOLESKY( A );  }
CHOLESKY( A ) {
    for (k = 0; k < M; k++) {
        #pragma omp task depend(inout:A(k,k)[0:tilesize]
        {  POTRF( A(k,k) );  }
        for (m = k+1; m < M; m++) {
            #pragma omp task \
                depend(in:A(k,k)[0:tilesize]) \
                depend(inout:A(m,k)[0:tilesize])
            { TRSM( A(k,k), A(m,k) ); }
        }
        for (m = k+1; m < M; m++) {
            #pragma omp task \
                depend(in:A(m,k)[0:tilesize]) \
                depend(inout:A(m,m)[0:tilesize])
            { SYRK( A(m,k), A(m,m) ); }
            for (n = k+1; n < m; n++) {
                #pragma omp task \
                    depend(in:A(m,k)[0:tilesize], \
                              A(n,k)[0:tilesize]) \
                    depend(inout:A(m,n)[0:tilesize])
                {  GEMM( A(m,k), A(n,k), A(m,n) ); }
            }
        }
    }
}
```

Dataflow Based Design

- Objectives
  - ➢ High utilization of each core
  - ➢ Scaling to large number of cores
  - ➢ Synchronization reducing algorithms
- Methodology
  - ➢ Dynamic DAG scheduling
  - ➢ Explicit parallelism
  - ➢ Implicit communication
  - ➢ Fine granularity / block data layout
- Arbitrary DAG with dynamic scheduling

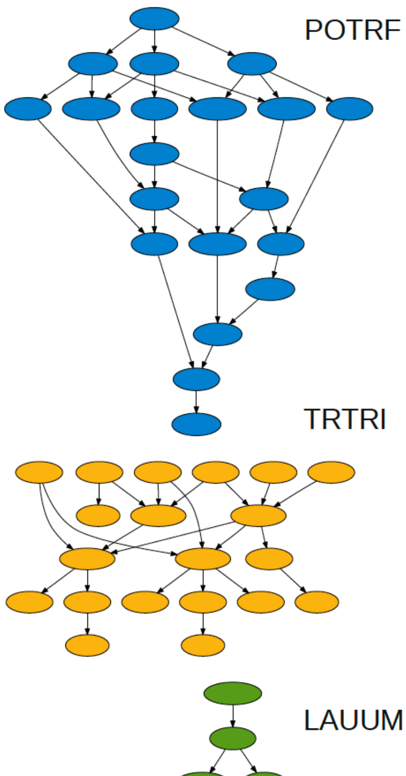


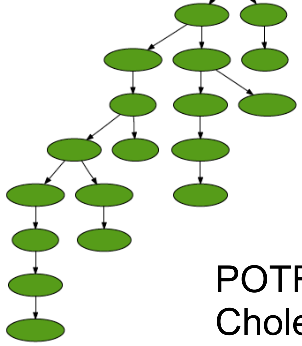


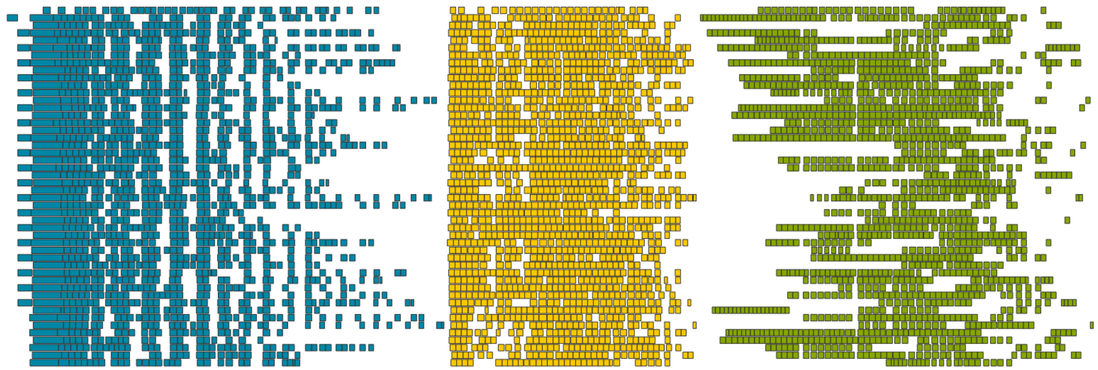
Fork-join parallelism
Notice the synchronization penalty in the presence of heterogeneity.

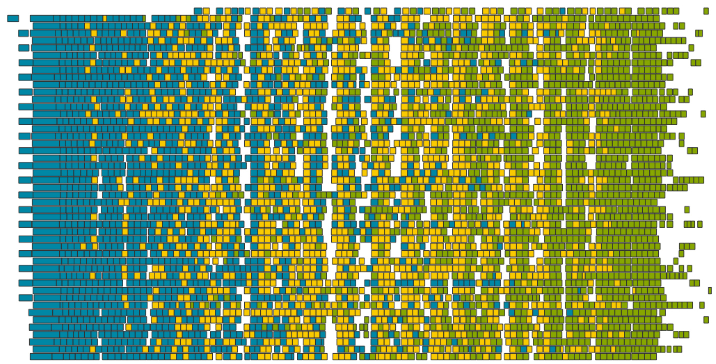
DAG scheduled parallelism

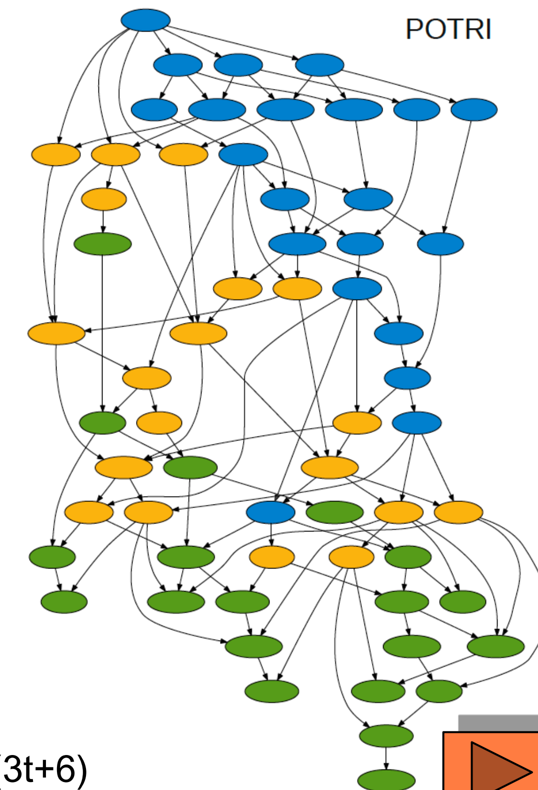
Pipelining: Cholesky Inversion
3 Steps: Factor, Invert L, Multiply L's











48 cores
POTRF, TRTRI and LAUUM.
The matrix is 4000 x 4000,tile size is 200 x 200,

POTRF+TRTRI+LAUUM: 25 (7t-3)
Cholesky Factorization alone: 3t-2

Pipelined: 18 (3t+6)

Avoiding Synchronization

- "Responsibly Reckless" Algorithms
  - Try fast algorithm (unstable algorithm) that might fail (but rarely)
  - Check for instability
  - If needed, recompute with stable algorithm

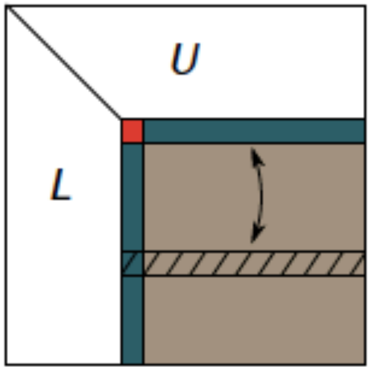
LU decomposition (Gaussian Elimination) for the solution of $Ax = b$

for $k = 1$ **to** n **do**

$a_{k+1:n,k} \leftarrow \frac{a_{k+1:n,k}}{a_{kk}}$

$a_{k+1:n,k+1:n} \leftarrow a_{k+1:n,k+1:n} - a_{k+1:n,k} \times a_{k,k+1:n}$

end for

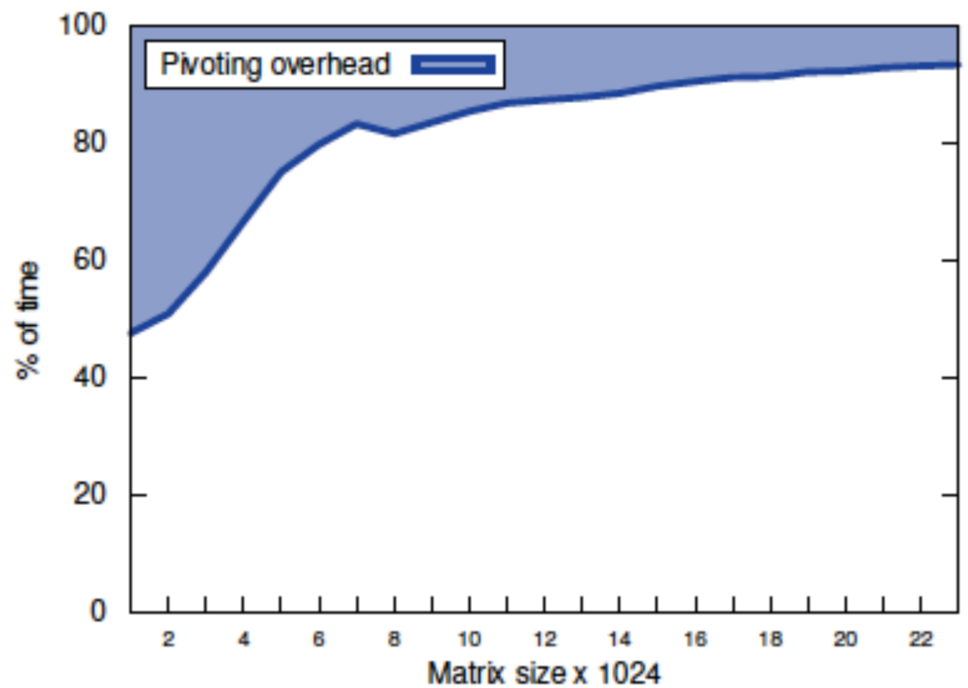


- Stability issue: a_{kk} may be small or zero $\Rightarrow$ large element growth $\Rightarrow$ elements of normal size lost in summation.
- Partial pivoting (GEPP): swap rows so that each a_{kk} is large. row k is exchanged with row p such that $|a_{pk}| = \max_{j \geqslant k} |a_{jk}|$

  Eventually, $PA = LU$ (P permutation matrix).

Pivoting is expensive

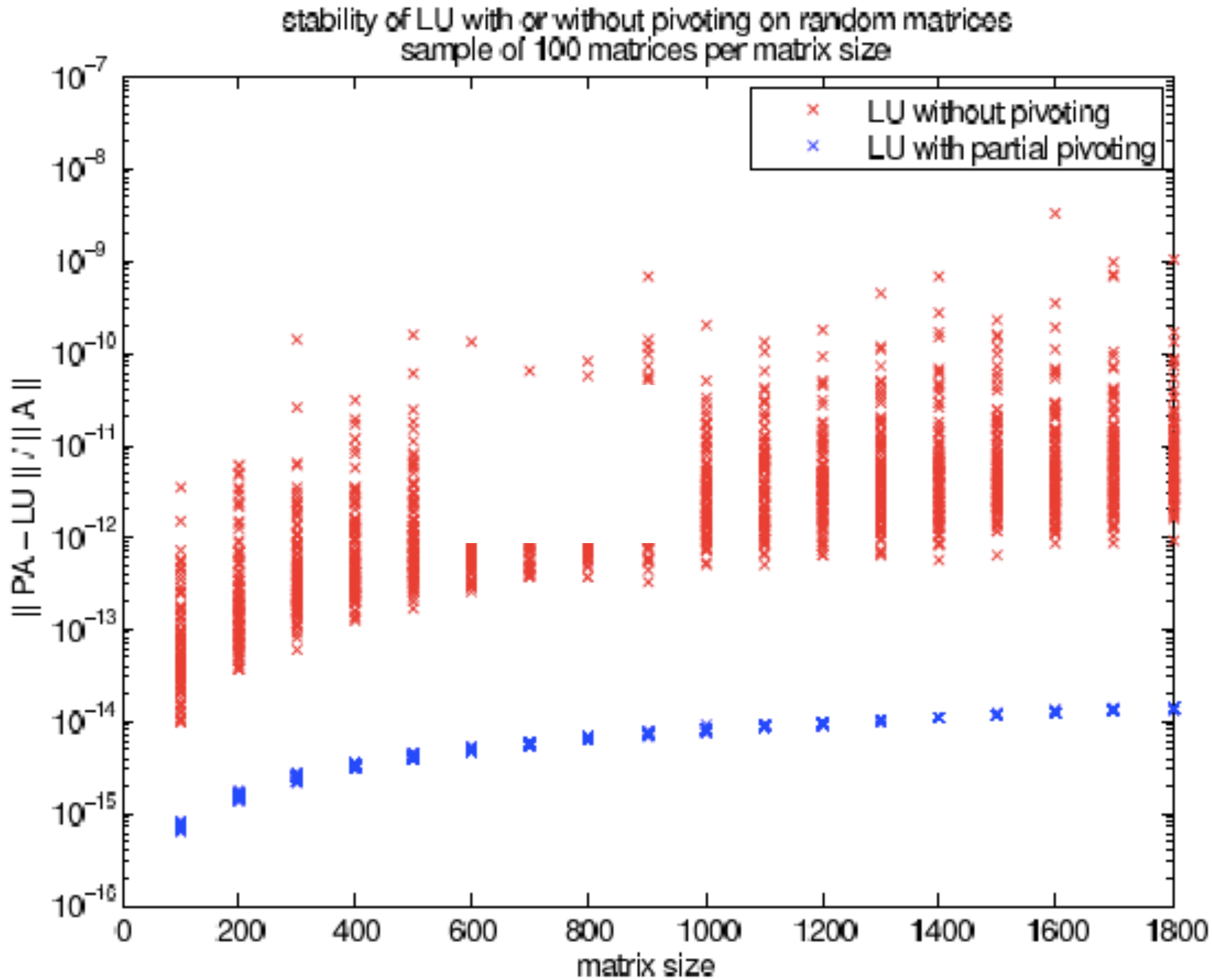
- Complete pivoting, partial pivoting, tournament pivoting, etc.
- GEPP implemented in most numerical libraries (LAPACK...)
- No floating point operation in pivoting but it involves irregular movements of data
- Communication overhead due to pivoting: $\mathcal{O}(n^2)$ comparisons



Cost of partial pivoting in LU factorization (MAGMA), Nvidia Kepler K20

Random matrices are nice (for pivoting)

(see [Trefethen and Schreiber, SIMAX 90] , [Yeung and Chan, SIMAX 97])



How to remove pivoting

No pivoting by randomizing instead:

- For general systems (LU factorization):
  Initially proposed by [Parker, 1995]
  Revisited in [MB, Dongarra, Herrmann Tomov, TOMS 2013]
- Idea: the original matrix is transformed into a matrix that would be sufficiently “random” so that, with a probability close to 1, pivoting is not needed.

How to avoid pivoting with randomization?

Random Butterfly Transformation (RBT)

$$Ax = b \equiv \underbrace{U^T A V}_{A_r} \underbrace{V^{-1} x}_{y} = \underbrace{U^T b}_{c}$$

1. Compute $A_r = U^T A V$ with U, V random (recursive butterflies)
2. Factorize A_r without pivoting (GENP)
3. Solve $A_r y = U^T b$ then $x = Vy$

Requirements :

- Randomization must be cheap
- Fast GENP (“Cholesky” speed)
- Accuracy close to that of GEPP (possibly IR)

Butterfly Matrix

A **butterfly matrix** is defined as any n-by-n matrix of the form:

$$B = \frac{1}{\sqrt{2}} \begin{pmatrix} R & S \\ R & -S \end{pmatrix}$$

where R and S are random diagonal matrices.

Remark:

$$B = \frac{1}{\sqrt{2}} \begin{pmatrix} I_{n/2} & I_{n/2} \\ I_{n/2} & -I_{n/2} \end{pmatrix} \begin{pmatrix} R & 0 \\ 0 & S \end{pmatrix}$$

HPL - Bad Things

- LINPACK Benchmark is 37 years old
  - TOP500 (HPL) is 22 years old
- Floating point-intensive performs $O(n^3)$ floating point operations and moves $O(n^2)$ data.
- No longer so strongly correlated to real apps.
- Reports Peak Flops (although hybrid systems see only 1/2 to 2/3 of Peak)
- Encourages poor choices in architectural features
- Overall usability of a system is not measured
- Used as a marketing tool
- Decisions on acquisition made on one number
- Benchmarking for days wastes a valuable resource

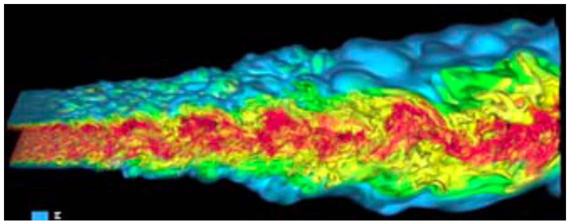
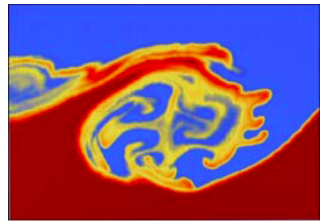
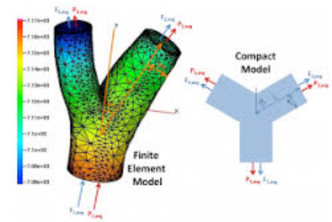
Proposal: HPCG

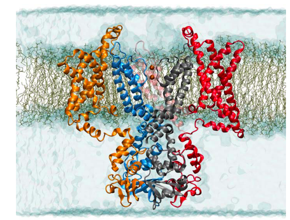
- High Performance Conjugate Gradient (HPCG).
- Solves $Ax=b$, A large, sparse, b known, x computed.
- An optimized implementation of PCG contains essential computational and communication patterns that are prevalent in a variety of methods for discretization and numerical solution of PDEs

- Patterns:
  - Dense and sparse computations.
  - Dense and sparse collective.
  - Multi-scale execution of kernels via MG (truncated) V cycle.
  - Data-driven parallelism (unstructured sparse triangular solves).
- Strong verification and validation properties (via spectral properties of PCG).

Goals for New Benchmark

- Augment the TOP500 listing with a benchmark that correlates with important scientific and technical apps not well represented by HPL

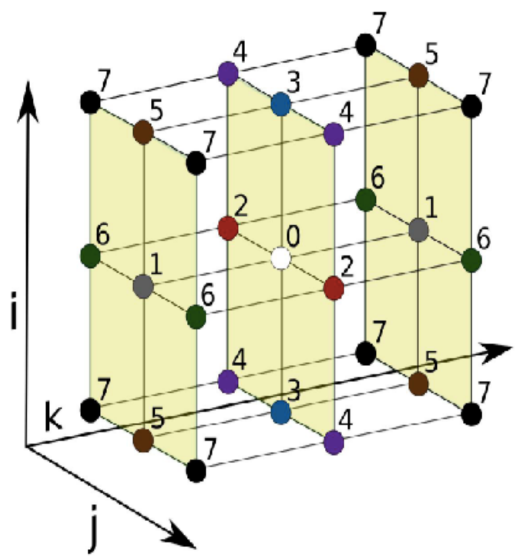






- Encourage vendors to focus on architecture features needed for high performance on those important scientific and technical apps.
  - Stress a balance of floating point and communication bandwidth and latency
  - Reward investment in high performance collective ops
  - Reward investment in high performance point-to-point messages of various sizes
  - Reward investment in local memory system performance
  - Reward investment in parallel runtimes that facilitate intra-node parallelism
- Provide an outreach/communication tool
  - Easy to understand
  - Easy to optimize
  - Easy to implement, run, and check results
- Provide a historical database of performance information
  - The new benchmark should have longevity

Model Problem Description

- Synthetic discretized 3D PDE (FEM, FVM, FDM).
- Single heat diffusion model.
- Zero Dirichlet BCs, Synthetic RHS s.t. solution = 1.
- Local domain: $(n_x \times n_y \times n_z)$
- Process layout: $(np_x \times np_y \times np_z)$
- Global domain: $(n_x * np_x) \times (n_y * np_y) \times (n_z * np_z)$
- Sparse matrix:
  - 27 nonzeros/row interior.
  - 7 – 18 on boundary.
  - Symmetric positive definite.



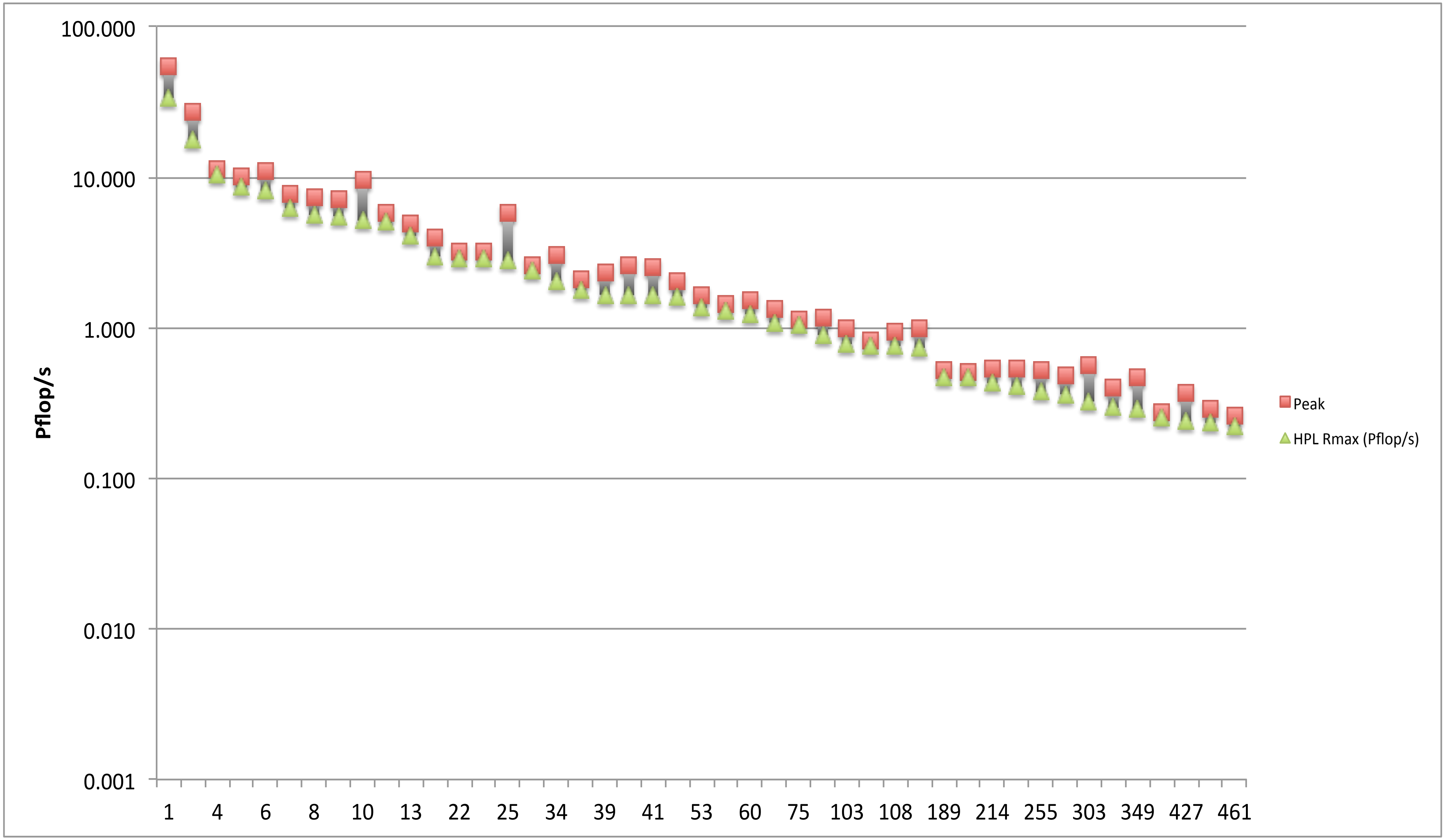
27-point stencil operator

HPL vs. HPCG: Bookends

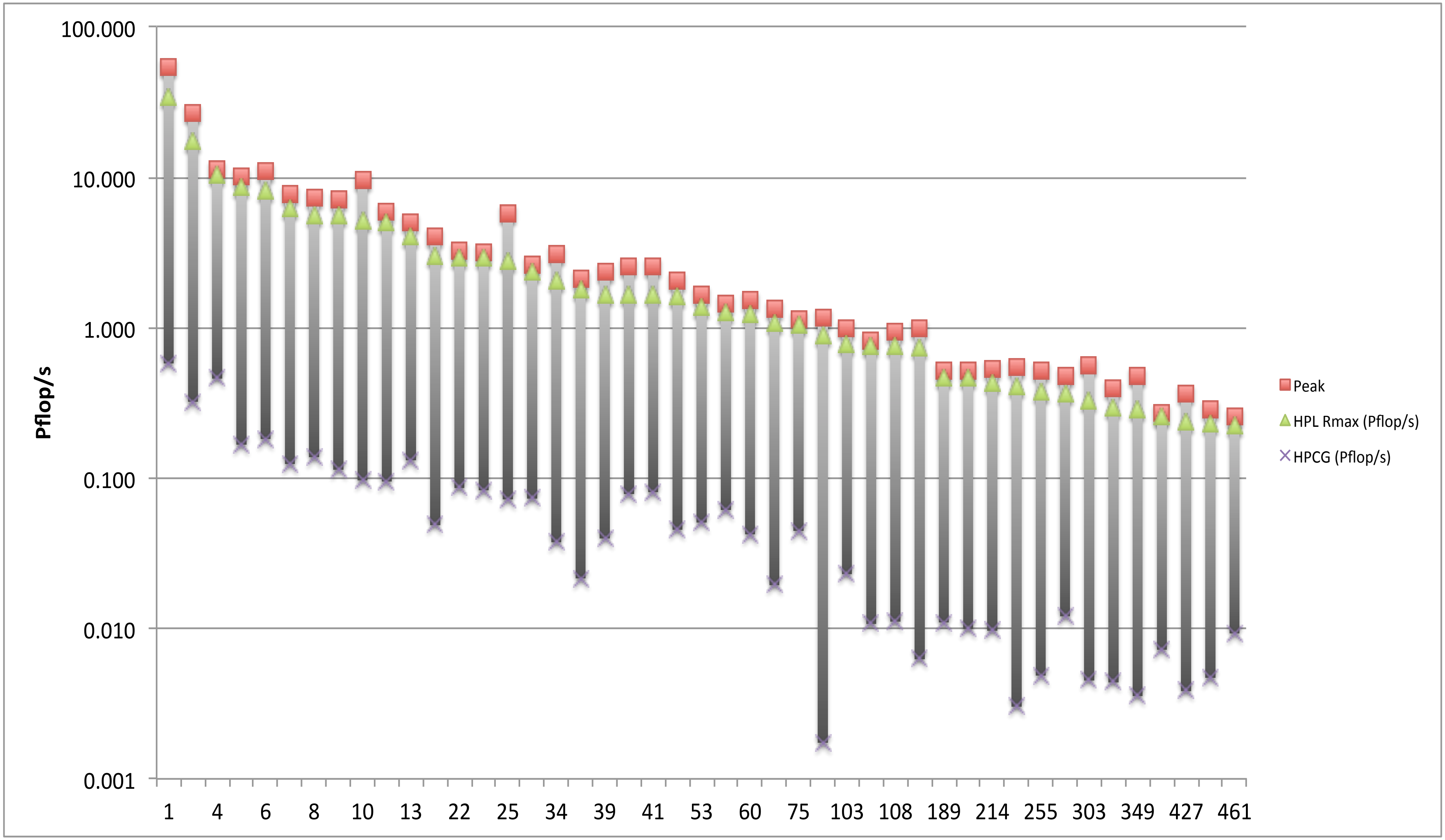
- Some see HPL and HPCG as “bookends” of a spectrum.
  - Applications teams know where their codes lie on the spectrum.
  - Can gauge performance on a system using both HPL and HPCG numbers.
- Problem of HPL execution time still an issue:
  - Need a lower cost option. End-to-end HPL runs are too expensive.
  - Work in progress.

  - Began last year with about 20 results, today have 41 systems.
    - Not interested in collecting 500 systems

Comparison Peak, HPL



Comparison Peak, HPL, & HPCG



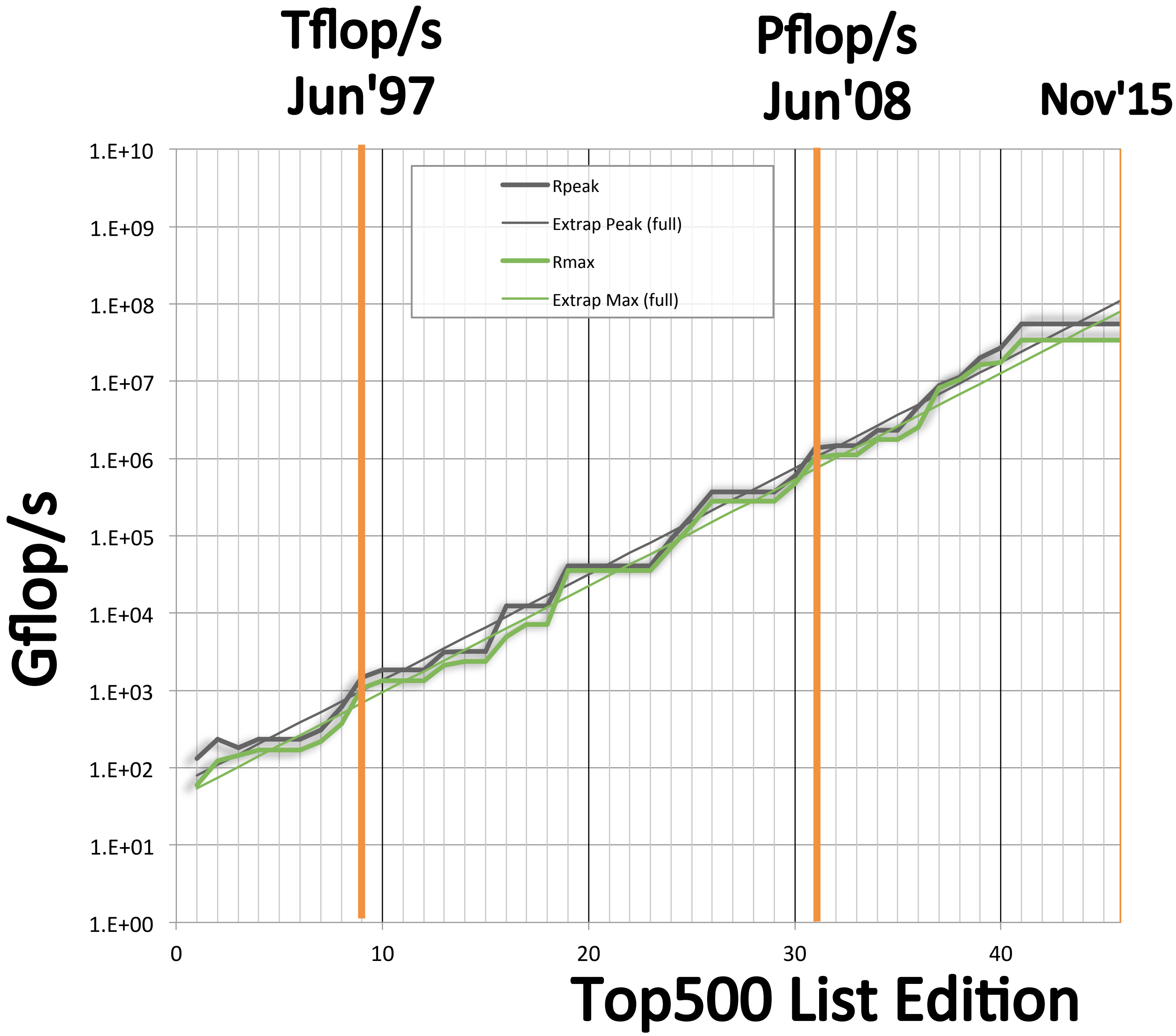
HPCG Results, Nov 2015, 1-10

Rank	Site	Computer	Cores	Rmax Pflops	HPCG Pflops	HPCG /HPL	% of Peak
1	NSCC / Guangzhou	Tianhe-2 NUDT, Xeon 12C 2.2GHz + Intel Xeon Phi 57C + Custom	3,120,000	33.86	0.580	1.7%	1.1%
2	RIKEN Advanced Institute for Computational Science	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect	705,024	10.51	0.460	4.4%	4.1%
3	DOE/SC/Oak Ridge Nat Lab	Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x	560,640	17.59	0.322	1.8%	1.2%
4	DOE/NNSA/LANL/SNL	Trinity - Cray XC40, Intel E5-2698v3, Aries custom	301,056	8.10	0.182	2.3%	1.6%
5	DOE/SC/Argonne National Laboratory	Mira - BlueGene/Q, Power BQC 16C 1.60GHz, Custom	786,432	8.58	0.167	1.9%	1.7%
6	HLRS/University of Stuttgart	Hazel Hen - Cray XC40, Intel E5-2680v3, Infiniband FDR	185,088	5.64	0.138	2.4%	1.9%
7	NASA / Mountain View	Pleiades - SGI ICE X, Intel E5-2680, E5-2680V2, E5-2680V3, Infiniband FDR	186,288	4.08	0.131	3.2%	2.7%
8	Swiss National Supercomputing Centre (CSCS)	Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x	115,984	6.27	0.124	2.0%	1.6%
9	KAUST / Jeda	Shaheen II - Cray XC40, Intel Haswell 2.3 GHz 16C, Cray Aries	196,608	5.53	0.113	2.1%	1.6%
10	Texas Advanced Computing Center/Univ. of Texas	Stampede - PowerEdge C8220, Xeon E5-2680 8C 2.7GHz, Infiniband, Phi SE10P	522,080	5.16	0.096	1.9%	1.0%

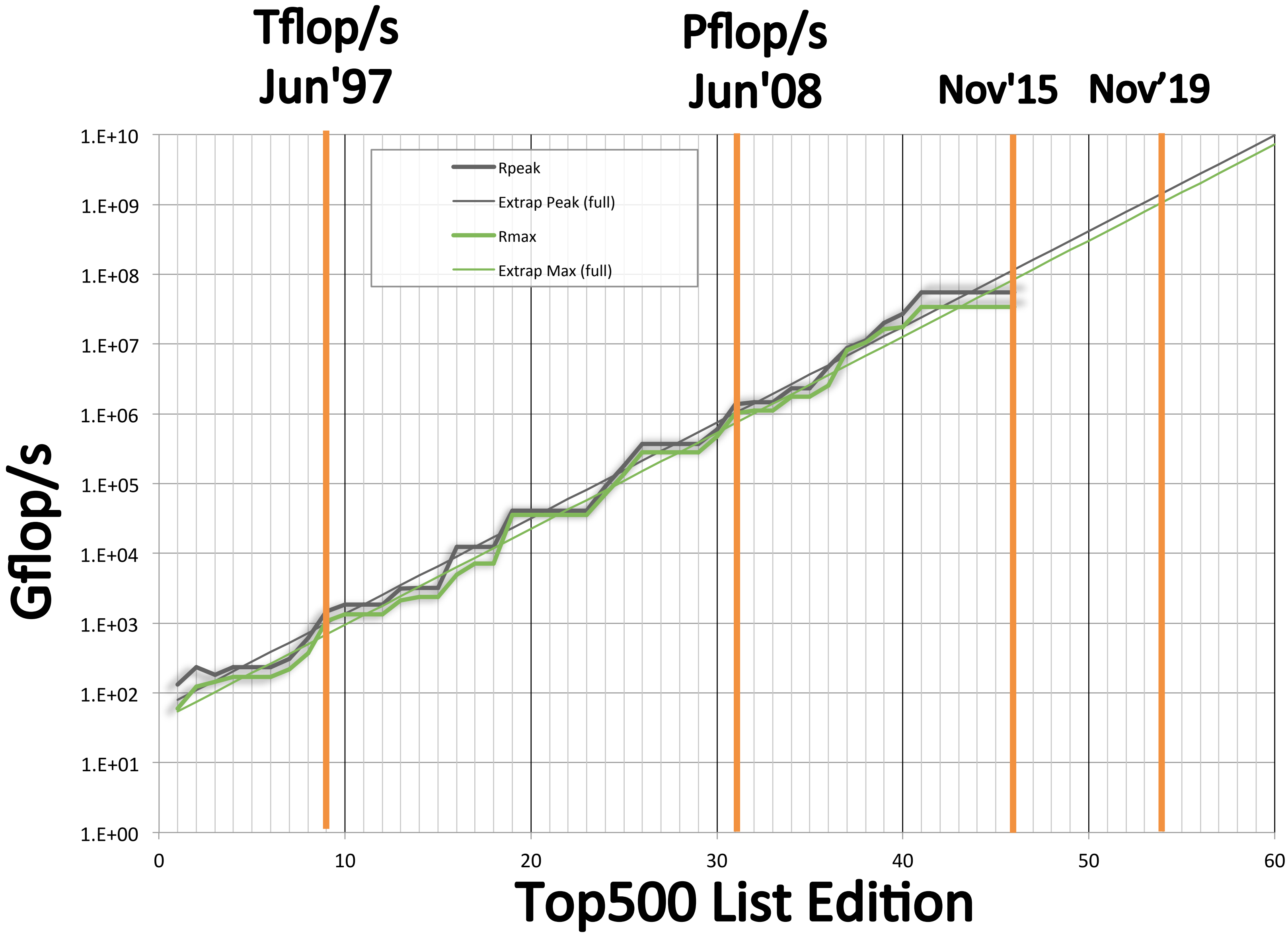
HPCG Results, Nov 2015, 11-20

Rank	Site	Computer	Cores	Rmax Pflops	HPCG Pflops	HPCG/ HPL	% of Peak
11	Forschungszentrum Jülich	JUQUEEN - BlueGene/Q	458,752	5.0089	0.095	1.9%	1.6%
12	Information Technology Center, Nagoya University	ITC, Nagoya - Fujitsu PRIMEHPC FX100	92,160	2.91	0.086	3.0%	2.7%
13	Leibniz Rechenzentrum	SuperMUC - iDataPlex DX360M4, Xeon E5-2680 8C 2.70GHz, Infiniband FDR	147,456	2.897	0.083	2.9%	2.6%
14	EPSRC/University of Edinburgh	ARCHER - Cray XC30, Intel Xeon E5 v2 12C 2.700GHz, Aries interconnect	118,080	1.643	0.081	4.9%	3.2%
15	DOE/SC/LBNL/NERSC	Edison - Cray XC30, Intel Xeon E5-2695v2 12C 2.4GHz, Aries interconnect	133,824	1.655	0.079	4.8%	3.1%
16	National Institute for Fusion Science	Plasma Simulator - Fujitsu PRIMEHPC FX100, SPARC64 Xifx, Custom	82,944	2.376	0.073	3.1%	2.8%
17	GSIC Center, Tokyo Institute of Technology	TSUBAME 2.5 - Cluster Platform SL390s G7, Xeon X5670 6C 2.93GHz, Infiniband QDR, NVIDIA K20x	76,032	2.785	0.073	2.6%	1.3%
18	HLRS/Universitaet Stuttgart	Hornet - Cray XC40, Xeon E5-2680 v3 2.5 GHz, Cray Aries	94,656	2.763	0.066	2.4%	1.7%
19	Max-Planck-Gesellschaft MPI/IPP	iDataPlex DX360M4, Intel Xeon E5-2680v2 10C 2.800GHz, Infiniband	65,320	1.283	0.061	4.8%	4.2%
20	CEIST / JAMSTEC	Earth Simulator - NEC SX-ACE	8,192	0.487	0.058	11.9%	11.0%

Top500 List 46 Edition



Top500 List 54 Edition



Today's #1 System

Systems	2015 Tianhe-2
System peak	**55 Pflop/s**
Power	**18 MW** **(3 Gflops/W)**
System memory	1.4 PB (1.024 PB CPU + .384 PB CoP)
Node performance	3.43 TF/s (.4 CPU +3 CoP)
Node concurrency	24 cores CPU + 171 cores CoP
Node Interconnect BW	6.36 GB/s
System size (nodes)	16,000
Total concurrency	3.12 M 12.48M threads (4/core)
MTTF	Few / day

Exascale System Architecture with a cap of $200M and 20MW

Systems	2015 Tianhe-2
System peak	**55 Pflop/s**
Power	**18 MW (3 Gflops/W)**
System memory	1.4 PB (1.024 PB CPU + .384 PB CoP)
Node performance	3.43 TF/s (.4 CPU +3 CoP)
Node concurrency	24 cores CPU + 171 cores CoProc
Node Interconnect BW	6.36 GB/s
System size (nodes)	16,000
Total concurrency	3.12 M 12.48M threads (4/core)
MTTF	Few / day

Exascale System Architecture with a cap of $200M and 20MW

Systems	2015 Tianhe-2	2020-2023	Difference Today & Exa
System peak	**55 Pflop/s**	**1 Eflop/s**	**~20x**
Power	**18 MW** **(3 Gflops/W)**	**~20 MW** **(50 Gflops/W)**	**O(1)** **~15x**
System memory	1.4 PB (1.024 PB CPU + .384 PB CoP)	256 PB	~100x
Node performance	3.43 TF/s (.4 CPU +3 CoP)	1.2 or 15TF/s	O(1)
Node concurrency	24 cores CPU + 171 cores CoProc	O(1k) or 10k	~5x - ~50x
Node Interconnect BW	6.36 GB/s	200-400 GB/s	~40x
System size (nodes)	16,000	O(100,000) or O(1M)	~6x - ~60x
Total concurrency	3.12 M 12.48M threads (4/core)	O(billion)	~100x
MTTF	Few / day	Many / day	O(?)

Critical Issues at Peta & Exascale for Algorithm and Software Design

- **Synchronization-reducing algorithms**
  - **Break Fork-Join model**
- **Communication-reducing algorithms**
  - **Use methods which have lower bound on communication**
- **Mixed precision methods**
  - **2x speed of ops and 2x speed for data movement**
- **Autotuning**
  - **Today's machines are too complicated, build "smarts" into software to adapt to the hardware**
- **Fault resilient algorithms**
  - **Implement algorithms that can recover from failures/bit flips**
- **Reproducibility of results**
  - **Today we can't guarantee this. We understand the issues, but some of our "colleagues" have a hard time with this.**

Summary

- Major Challenges are ahead for extreme computing
  - Parallelism $O(10^9)$
    - Programming issues
  - Hybrid
    - Peak and HPL may be very misleading
    - No where near close to peak for most apps
  - Fault Tolerance
    - Today Sequoia BG/Q node failure rate is 1.25 failures/day
  - Power
    - 50 Gflops/w (today at 2 Gflops/w)
- We will need completely new approaches and technologies to reach the Exascale level

Collaborators / Software / Support

- **PLASMA**
  http://icl.cs.utk.edu/plasma/
- **MAGMA**
  http://icl.cs.utk.edu/magma/
- **PaRSEC(**Parallel Runtime Scheduling and Execution Control**)**
- **http://icl.cs.utk.edu/parsec/**

- Collaborating partners
  University of Tennessee, Knoxville
  University of California, Berkeley
  University of Colorado, Denver

MAGMA PLASMA