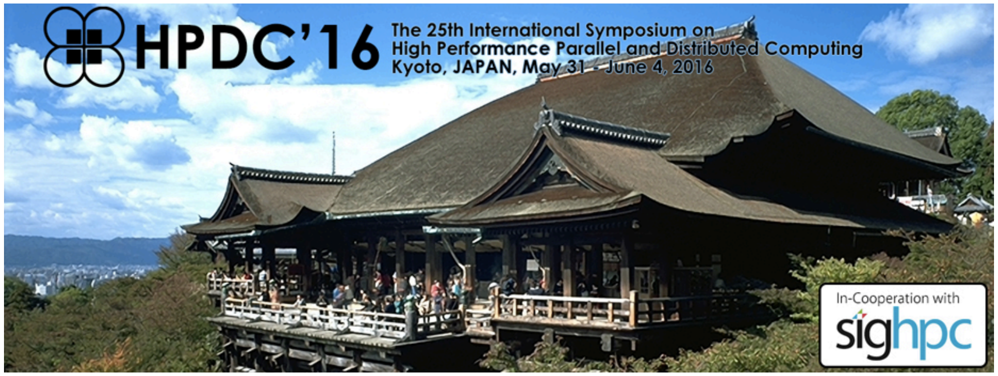


With Extreme Scale Computing the Rules Have Changed

Jack Dongarra

University of Tennessee
Oak Ridge National Laboratory
University of Manchester

HPDC 2016 Achievement Award

Outline

- Overview of High Performance Computing
- Look at some of the adjustments that are needed with Extreme Computing

White House HPC Initiative

The White House
Office of the Press Secretary

For Immediate Release July 29, 2015

Executive Order -- Creating a National Strategic Computing Initiative

EXECUTIVE ORDER

\- - - - - - -

CREATING A NATIONAL STRATEGIC COMPUTING INITIATIVE

By the authority vested in me as President by the Constitution and the laws of the United States of America, and to maximize benefits of high-performance computing (HPC) research, development, and deployment, it is hereby ordered as follows:

Section 1. Policy. In order to maximize the benefits of HPC for economic competitiveness and scientific discovery, the United States Government must create a coordinated Federal strategy in HPC research, development, and deployment. Investment in HPC has contributed substantially to national economic prosperity and rapidly accelerated scientific discovery. Creating and deploying technology at the leading edge is vital to advancing my Administration's priorities and spurring innovation. Accordingly,

NSCI has 5 Strategic Themes

- Create systems that can apply exaflops of computing power to exabytes of data.
- Keep the United States at the forefront of HPC capabilities.
- Improve HPC application developer productivity
- Make HPC readily available
- Establish hardware technology for future HPC systems.

THE WHITE HOUSE
WASHINGTON

State of Supercomputing Today

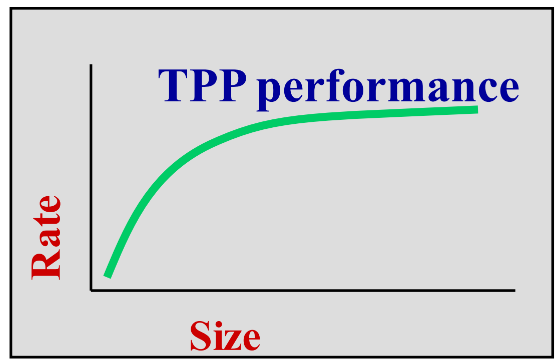
- Pflops (> 10^{15} Flop/s) computing fully established with 81 systems.
- Three technology architecture possibilities or "swim lanes" are thriving.
  - Commodity (e.g. Intel)
  - Commodity + accelerator (e.g. GPUs) (104 systems)
  - Special purpose lightweight cores (e.g. IBM BG, ARM, Intel's Knights 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

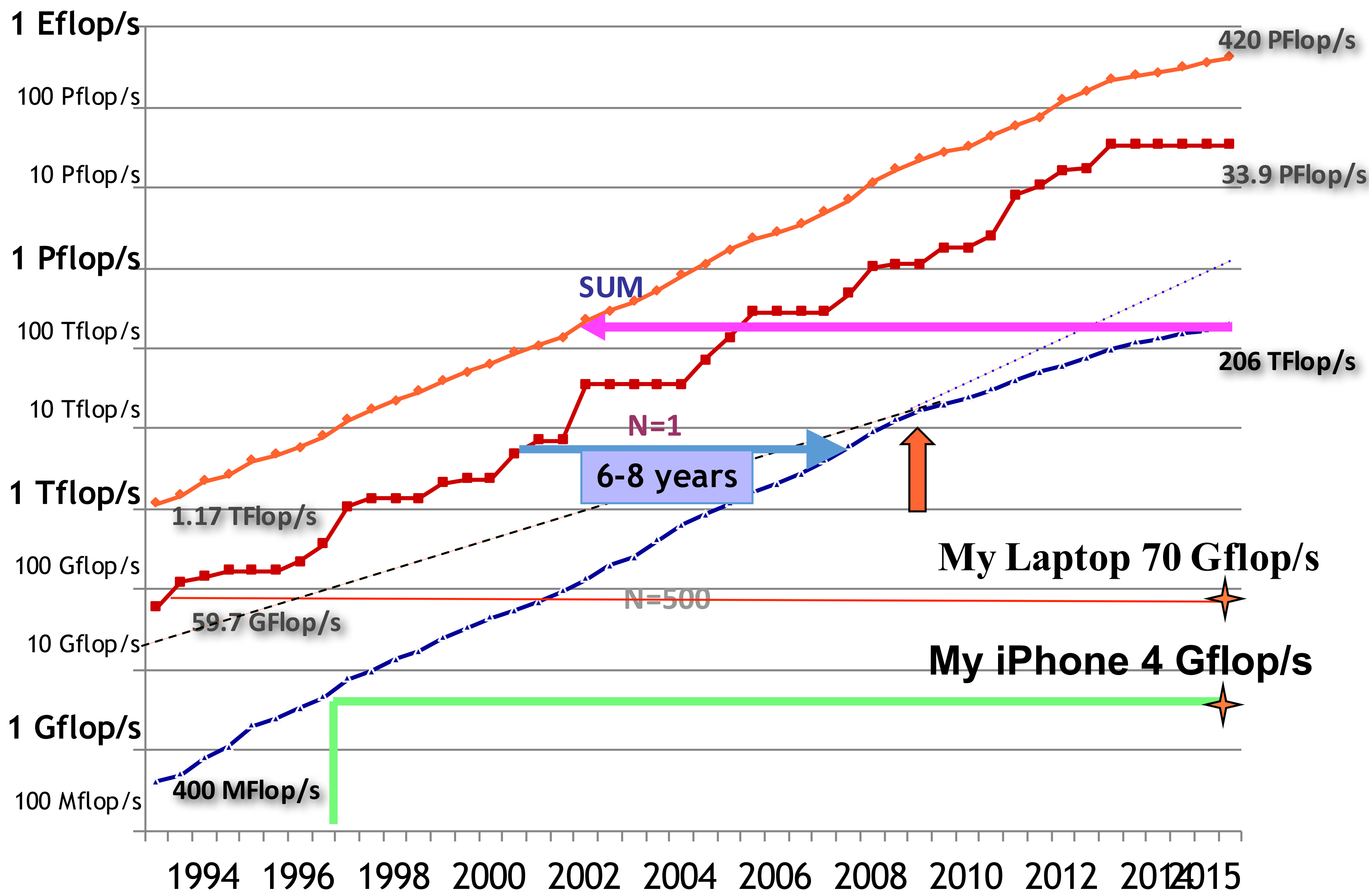
- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP

  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**

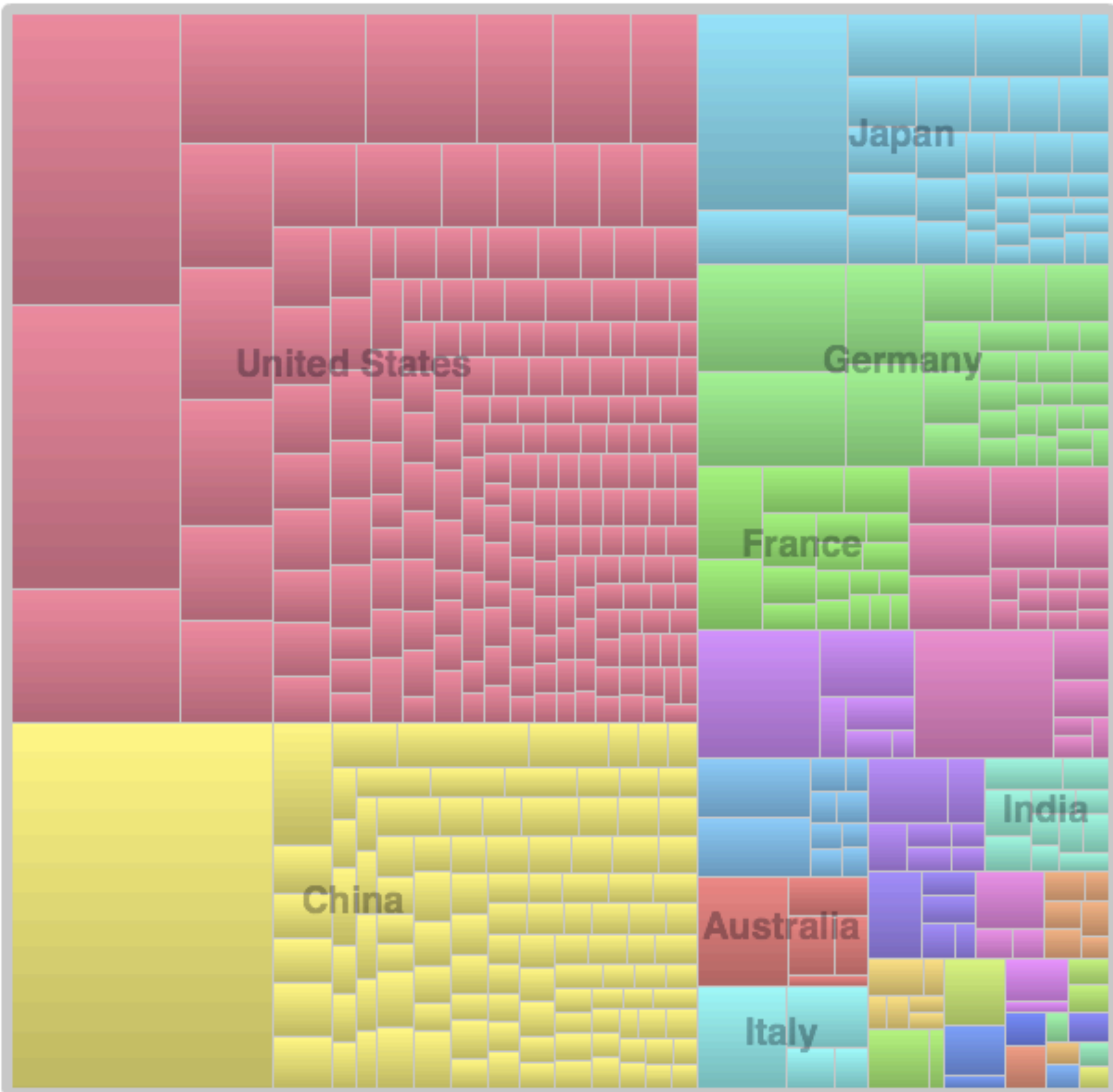
Performance Development of HPC over the Last 24 Years from the Top500



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 + IntelXeon 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		

Countries Share



Absolute Counts

US:	201
China:	109
Japan:	37
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.

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

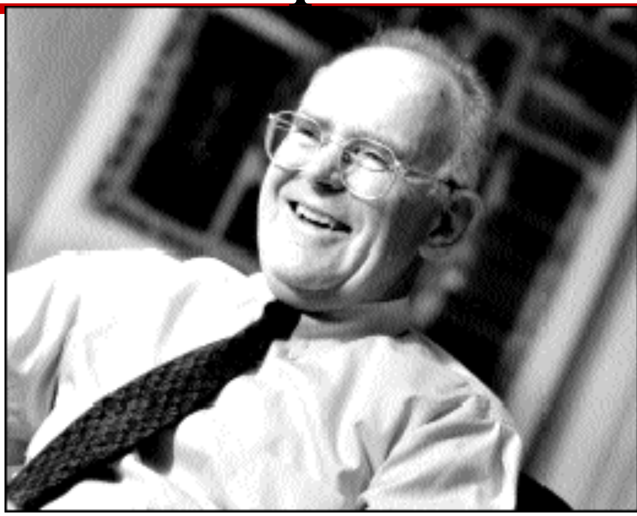
  … .

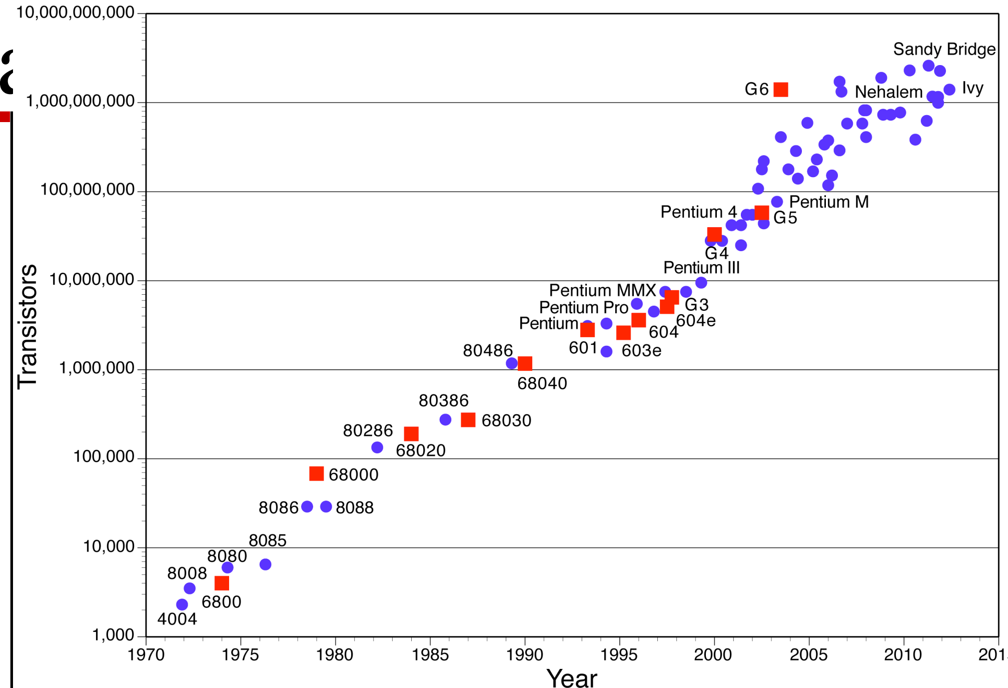
  - … ngzh…
  - … njin, …
  - … Def…
  - National SC Center Changsh…

Since the Dept of Commerce Action …

- Expanded focus on Chinese made HW and SW
  - Anything but from the US
- Three separate developments in HPC
  - Wuxi
    - Sunway TaihuLight 125 Pflops Peak, all Chinese, ShenWei Proc, June 2016 (ISC2016)
  - NUDT
    - Tianhe-2A O(100) Pflops will be Chinese ARM, 2017
  - CAS ICT
    - Godson MIPS and new processors
- In the latest "5 Year Plan"
  - Govt push to build out a domestic HPC ecosystem.
  - Exascale system, will not use any US chips
  - Targeting China's key industrial apps, via SW centers.

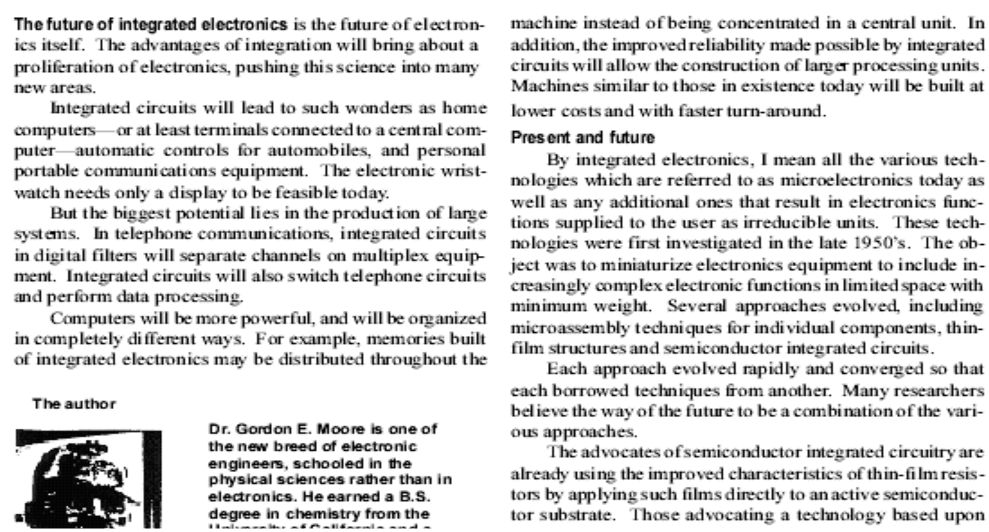
Technology Trends: Microprocessor Ca





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.

Present and future

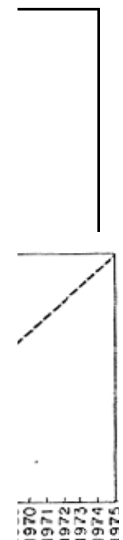
By integrated electronics, I mean all the various technologies which are referred to as microelectronics today as well as any additional ones that result in electronics functions supplied to the user as irreducible units. These technologies were first investigated in the late 1950's. The object was to 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

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



2X transistors/Chip Every 1.5 years

Called "Moore's Law"

Moore's *Secret Sauce*: Dennard Scaling

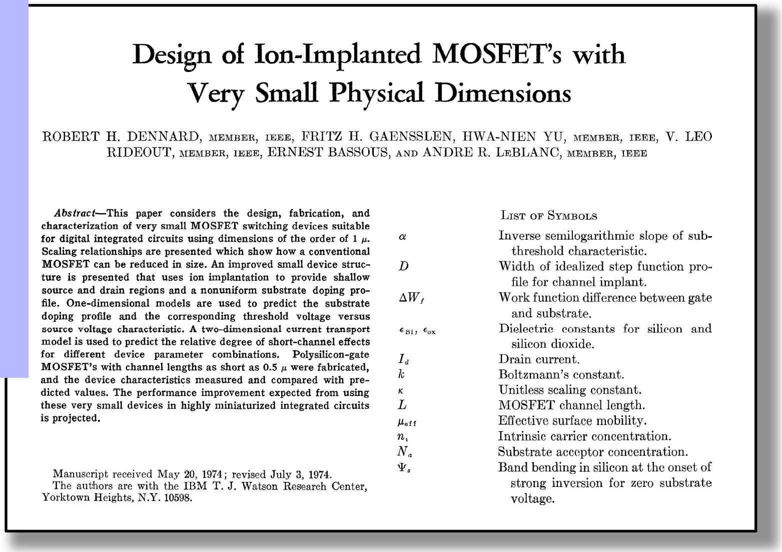
Moore's Law put lots more transistors on a chip…but it's Dennard's Law that made them useful

Dennard observed that voltage and current should be proportional to the linear dimensions of a transistor

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**

2x transistor count
40% faster
50% more efficient



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.

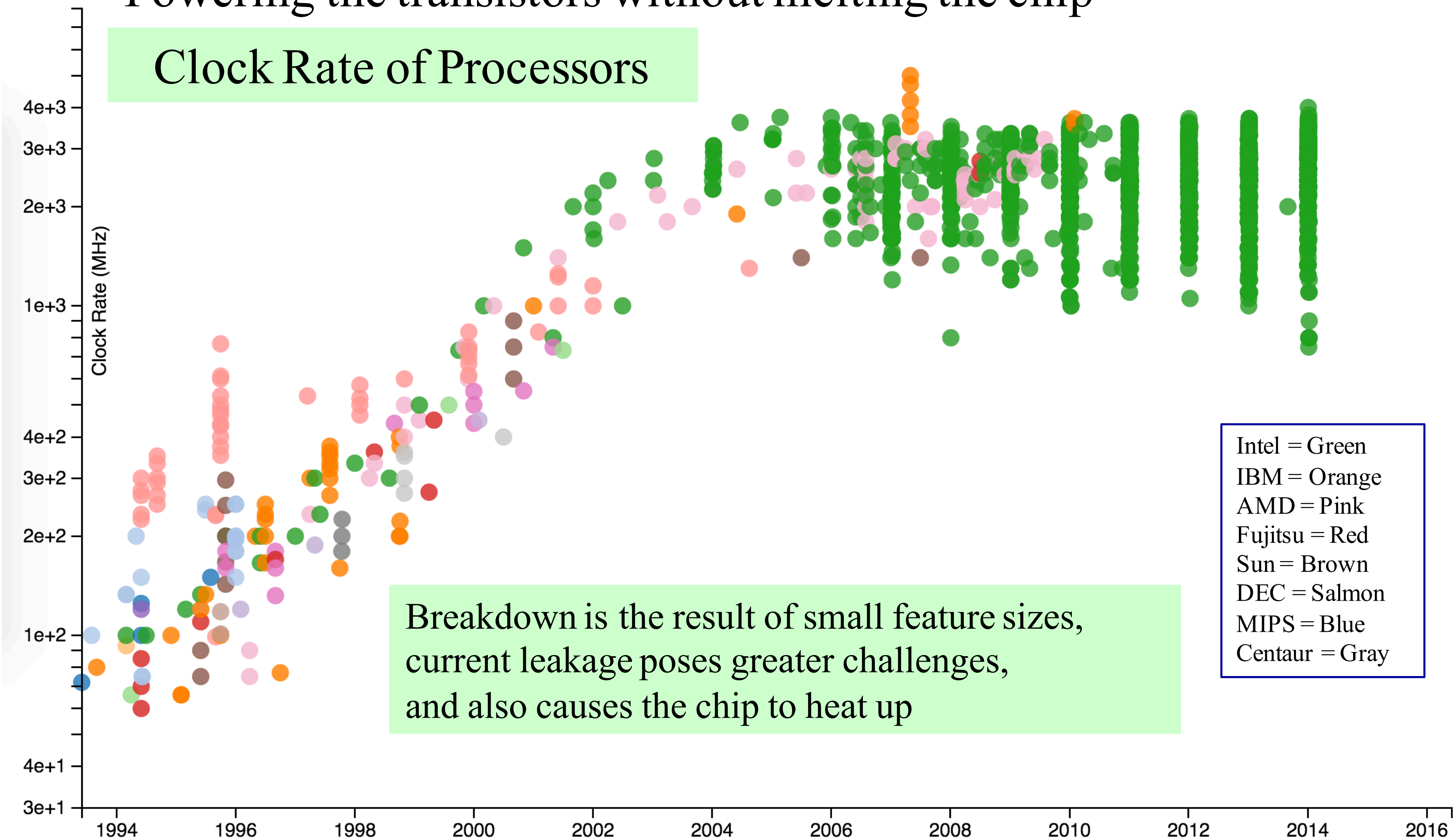
LIST OF SYMBOLS

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.

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

Unfortunately Dennard Scaling is Over: What is the Catch?

Powering the transistors without melting the chip

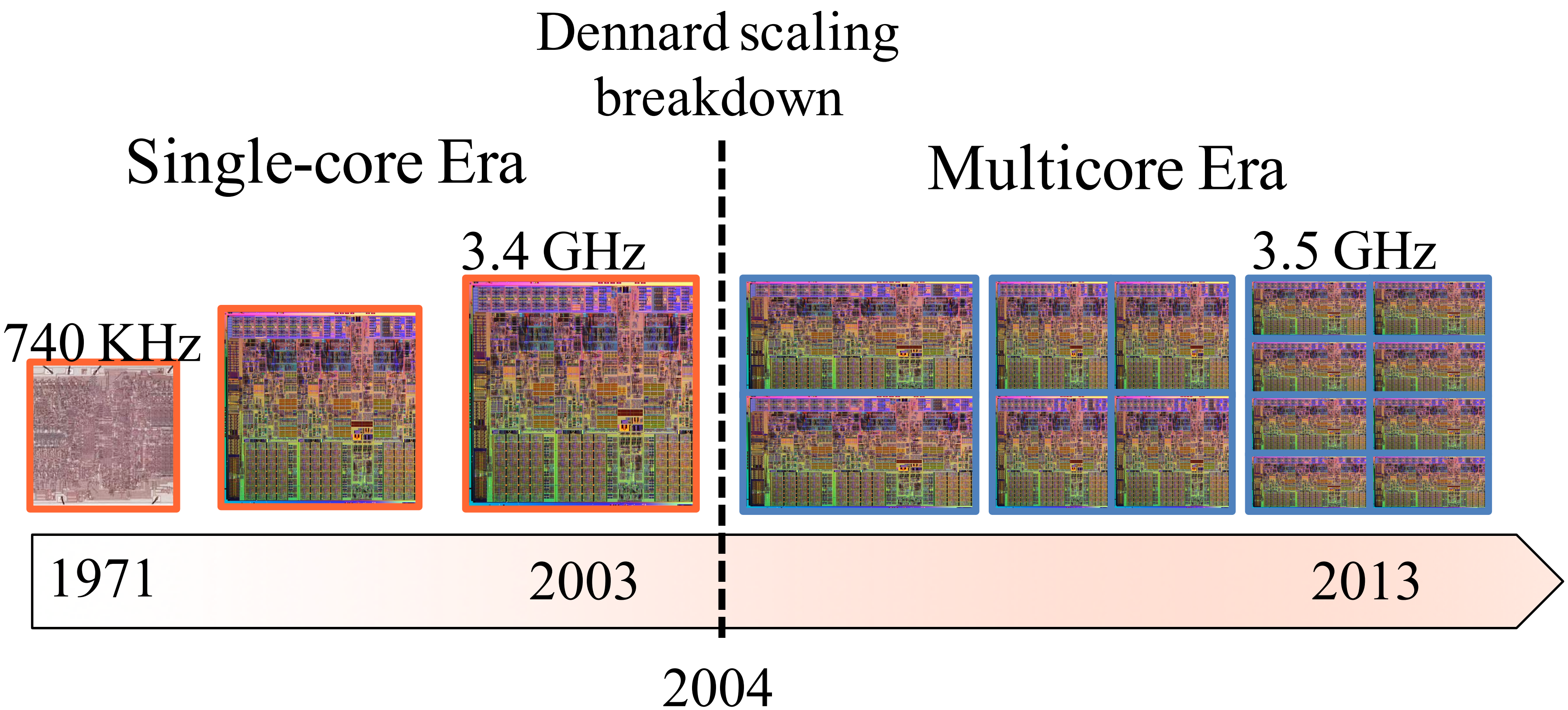


CPU DB: recording microprocessor history, CACM, V 55 N 4, 2012, http://dl.acm.org/citation.cfm?id=2133822

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. Can't continue to reduce the cycle time.



High Cost of Data Movement

Operation	Energy consumed	Time needed
64-bit multiply-add	200 pJ	1 nsec
Read 64 bits from cache	800 pJ	3 nsec
Move 64 bits across chip	2000 pJ	5 nsec
Execute an instruction	7500 pJ	1 nsec
Read 64 bits from DRAM	*12000 pJ*	*70 nsec*

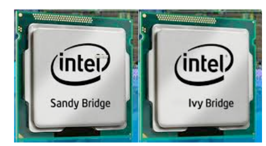
Communication is now almost all of the parts cost, almost all of the time spent, and almost all of the energy and power consumed!

Peak Performance - Per Core

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

Floating point operations per cycle per core

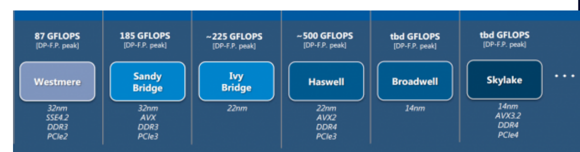
- Most of the recent computers have FMA (Fused multiple add): (i.e. x ←x + y*z in one cycle)
- Intel Xeon earlier models and AMD Opteron have SSE2
  - 2 flops/cycle DP & 4 flops/cycle SP
- Intel Xeon Nehalem ('09) & Westmere ('10) have SSE4
  - 4 flops/cycle DP & 8 flops/cycle SP
- Intel Xeon Sandy Bridge('11)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



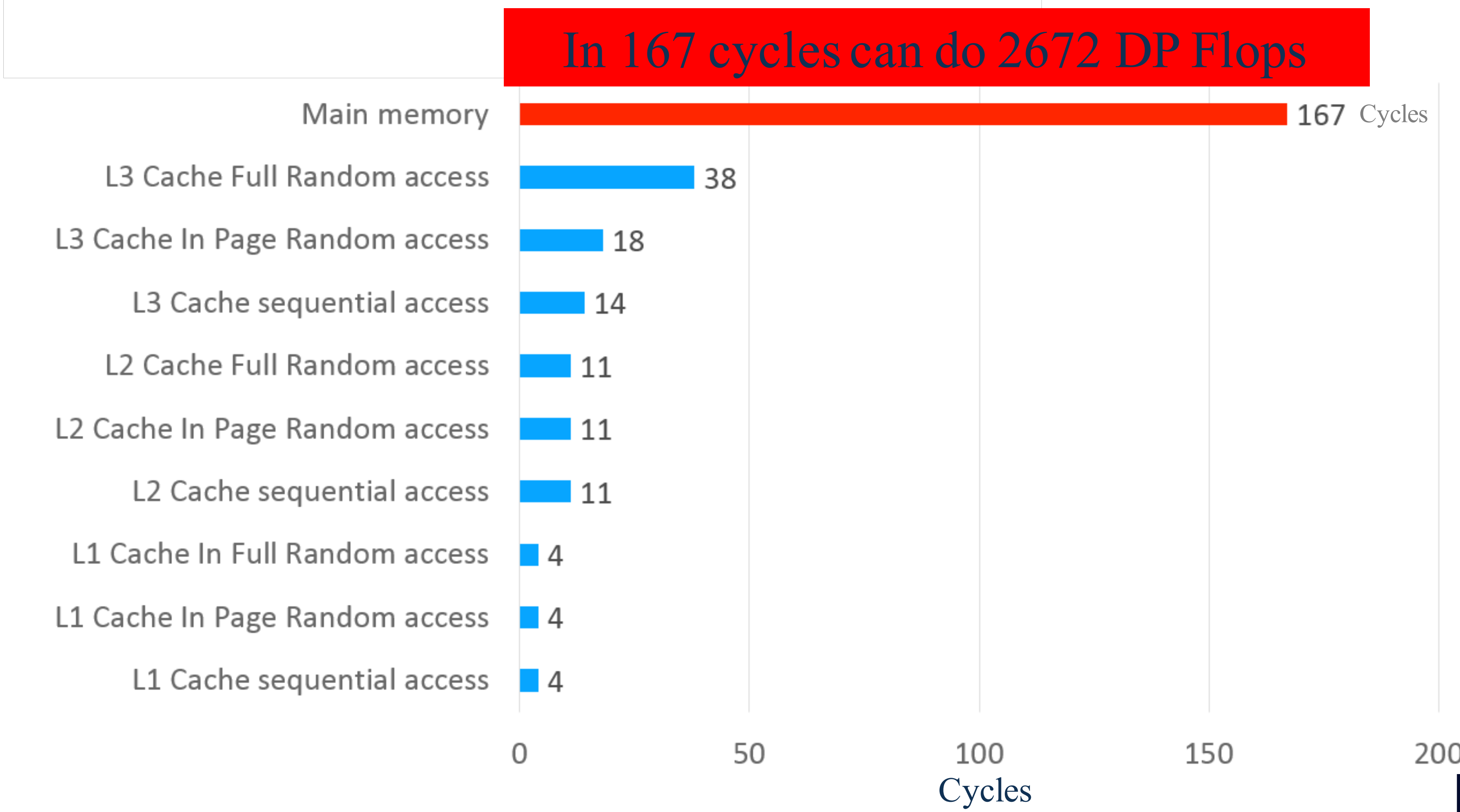
- Intel Xeon Haswell ('13) & (Broadwell ('14)) AVX2
  - 16 flops/cycle DP & 32 flops/cycle SP
  - Xeon Phi (per core) is at 16 flops/cycle DP & 32 flops/cycle SP

We are here (almost) →

- Intel Xeon Skylake (server) AVX 512
  - 32 flops/cycle DP & 64 flops/cycle SP
  - Knight's Landing



CPU Access Latencies in Clock Cycles

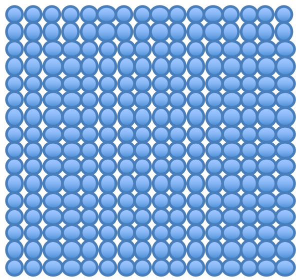


Classical Analysis of Algorithms May Not be Valid

- 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.

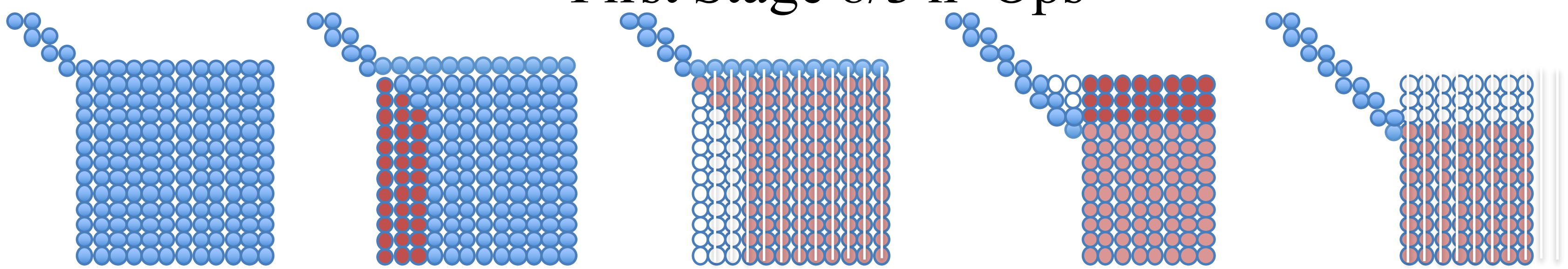
6/3/16

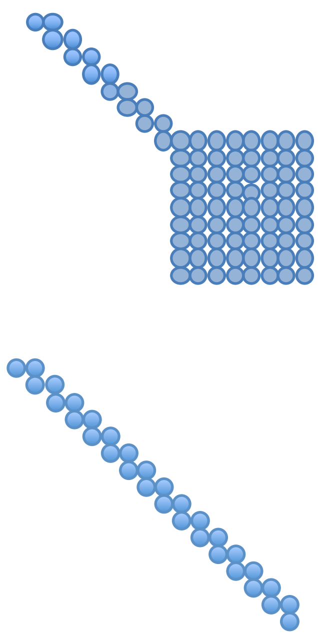
Singular Value Decomposition
LAPACK Version 1991



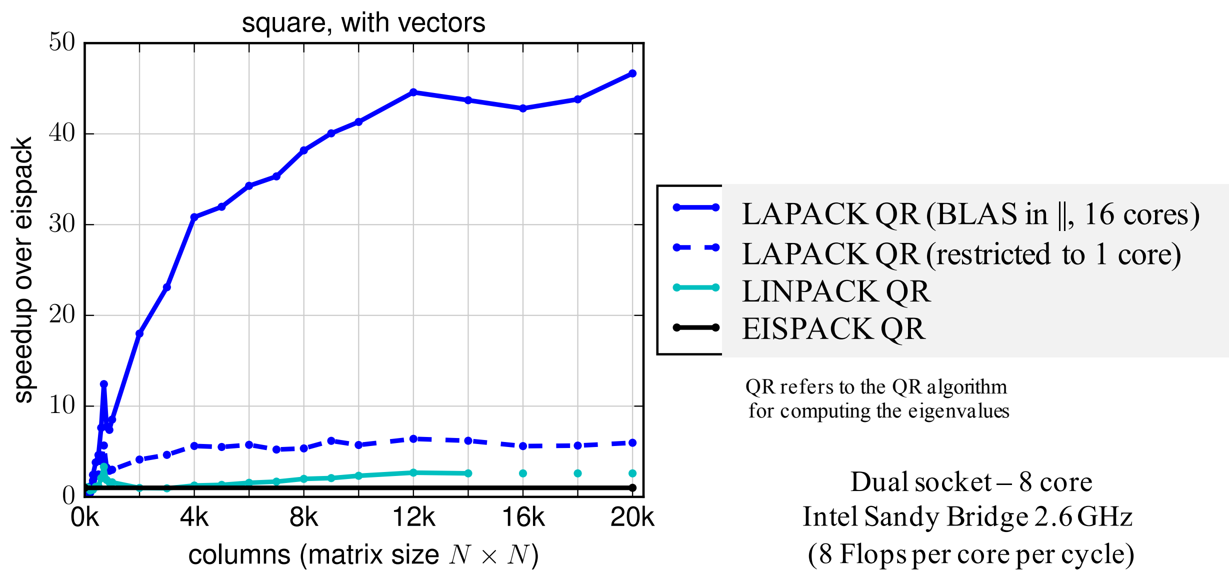
Level 1, 2, & 3 BLAS

First Stage 8/3 n^3 Ops





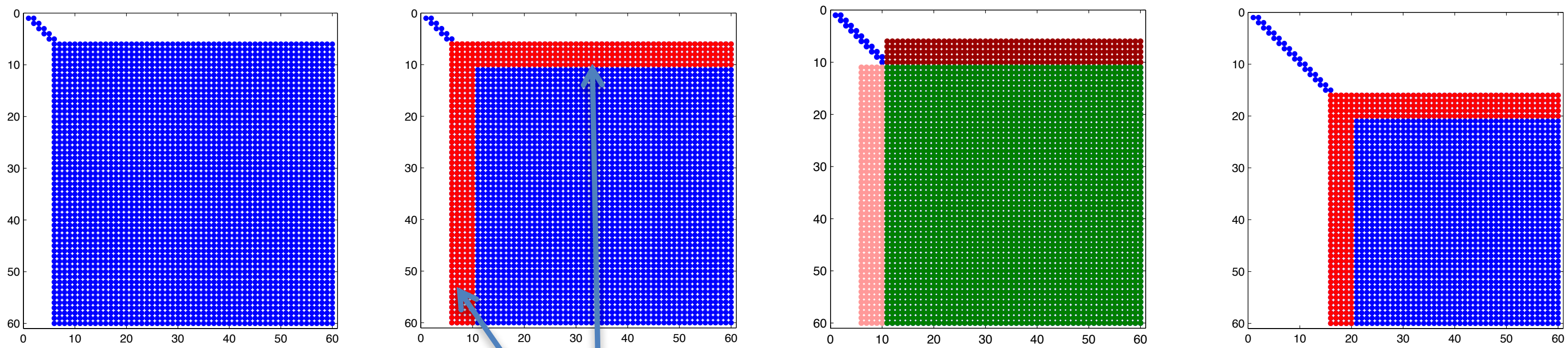
3 Generations of software compared



Bottleneck in the Bidiagonalization

The Standard Bidiagonal Reduction: xGEBRD

Two Steps: Factor Panel & Update Tailing Matrix



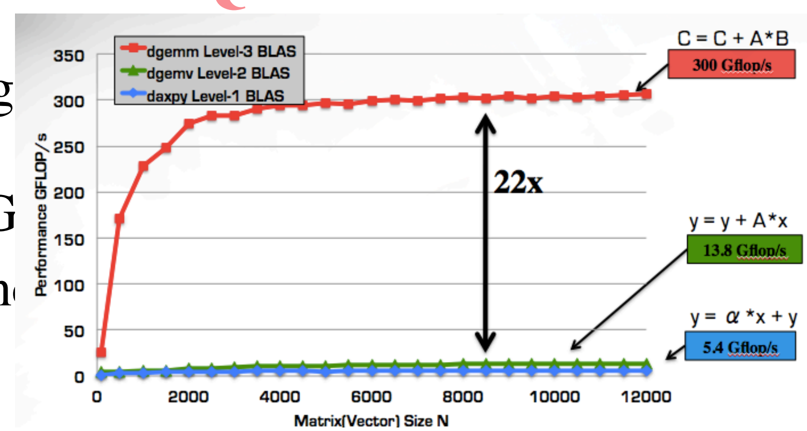
factor panel k then update ➔ **factor panel k+1**

Requires 2 GEMVs

$Q * A * P^H$

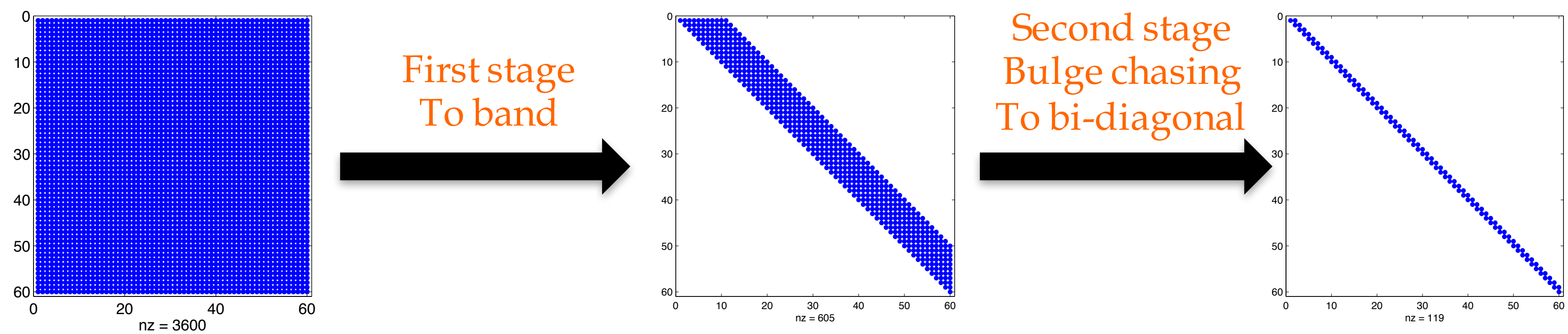
✷ Characteristics

- 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

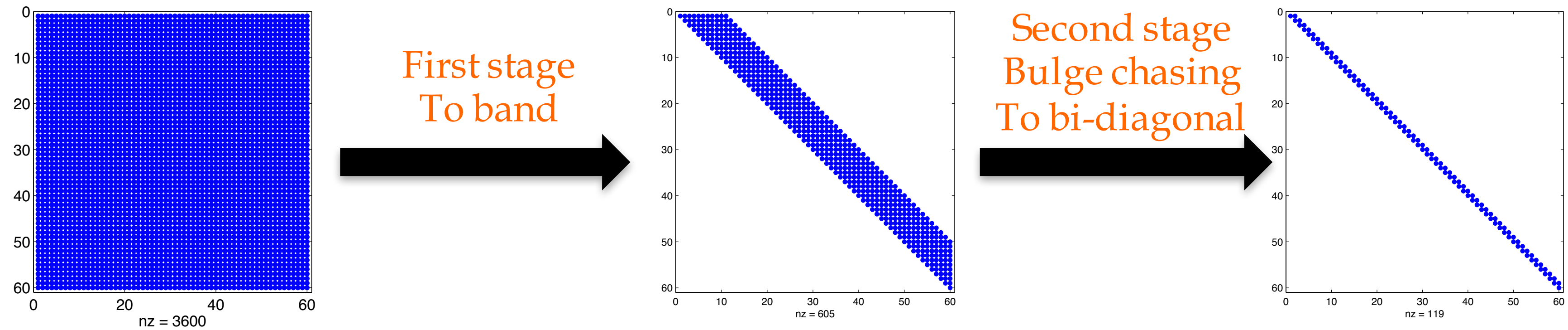


✸ Characteristics

- **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



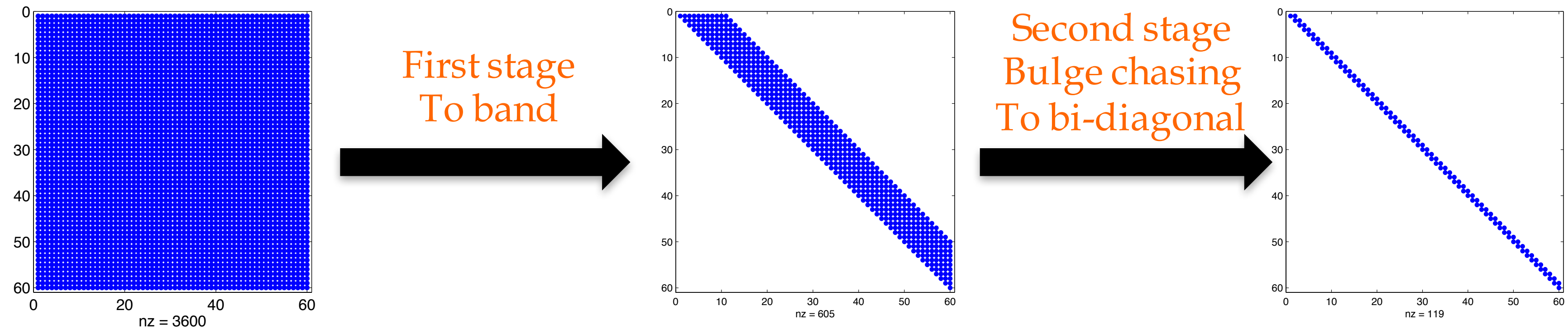
$$\begin{aligned}\textbf{flops} &\approx \sum_{s=1}^{\frac{n-n_b}{n_b}} 2n_b^3 + (nt-s)3n_b^3 + (nt-s)\frac{10}{3}n_b^3 + (nt-s)\times(nt-s)5n_b^3 \\ &+ \sum_{s=1}^{\frac{n-n_b}{n_b}} 2n_b^3 + (nt-s-1)3n_b^3 + (nt-s-1)\frac{10}{3}n_b^3 + (nt-s)\times(nt-s-1)5n_b^3 \\ &\approx \frac{10}{3}n^3 + \frac{10n_b}{3}n^2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

$$\textbf{flops} = 6 \times n_b \times n^2(\text{gemv})_{\text{second stage}}$$

More Flops, original did 8/3 n^3

25% More flops

Recent work on developing new 2-stage algorithm

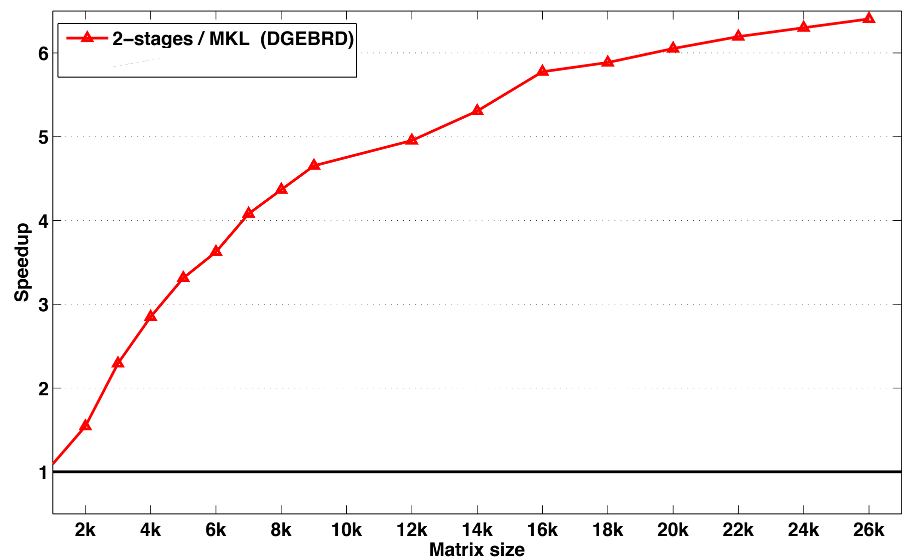


$$\textbf{speedup} = \frac{\text{time of one-stage}}{\text{time of two-stage}}$$

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

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

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



16 Sandy Bridge cores 2.6 GHz

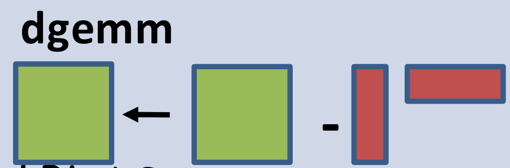
if P_{gemm} **is about 22x** P_{gemv} **and** $120 \leq n_b \leq 240$.

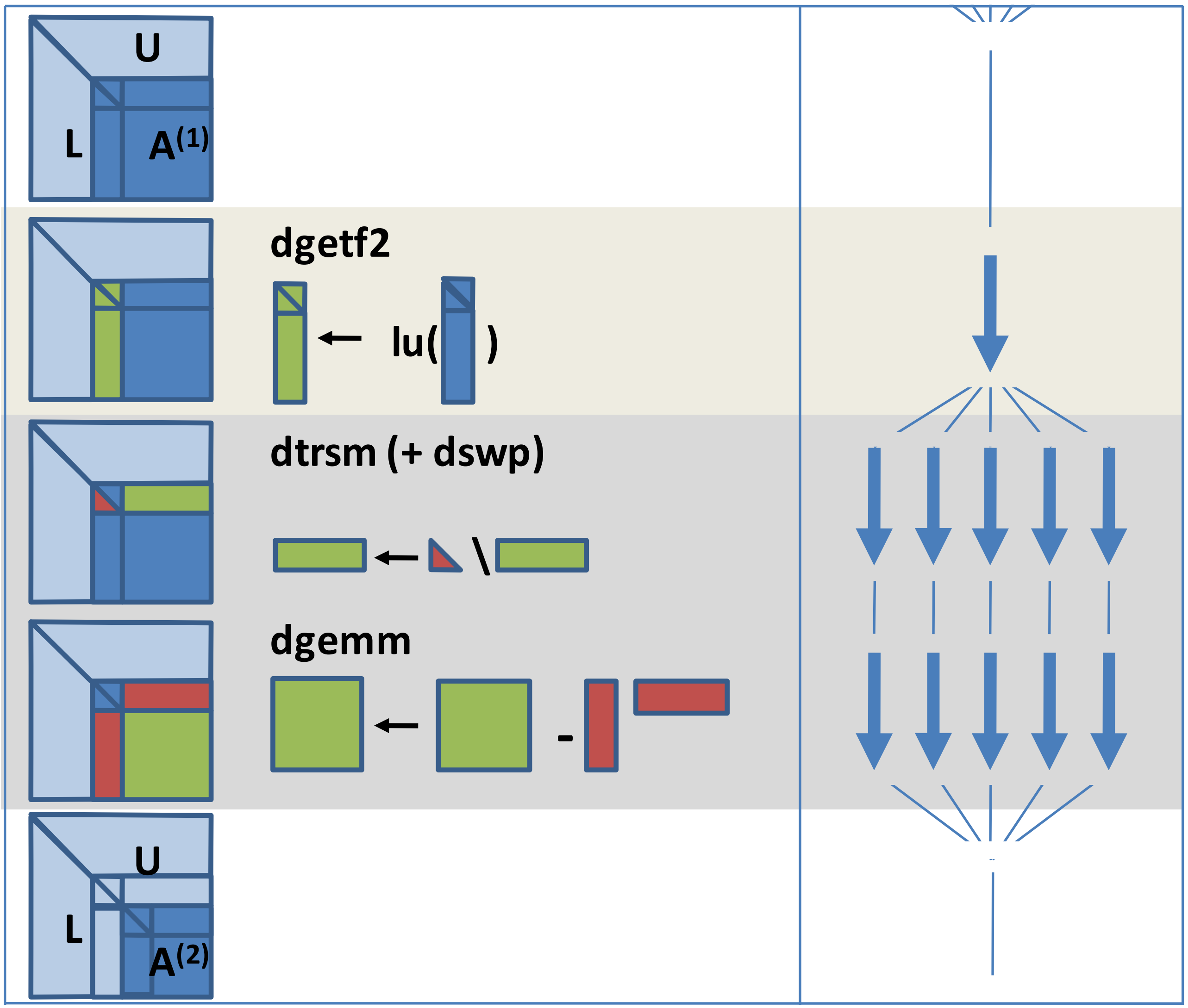
25% More flops and 1.8 – 7 times faster

Parallelization of LU and QR.

Parallelize the update:

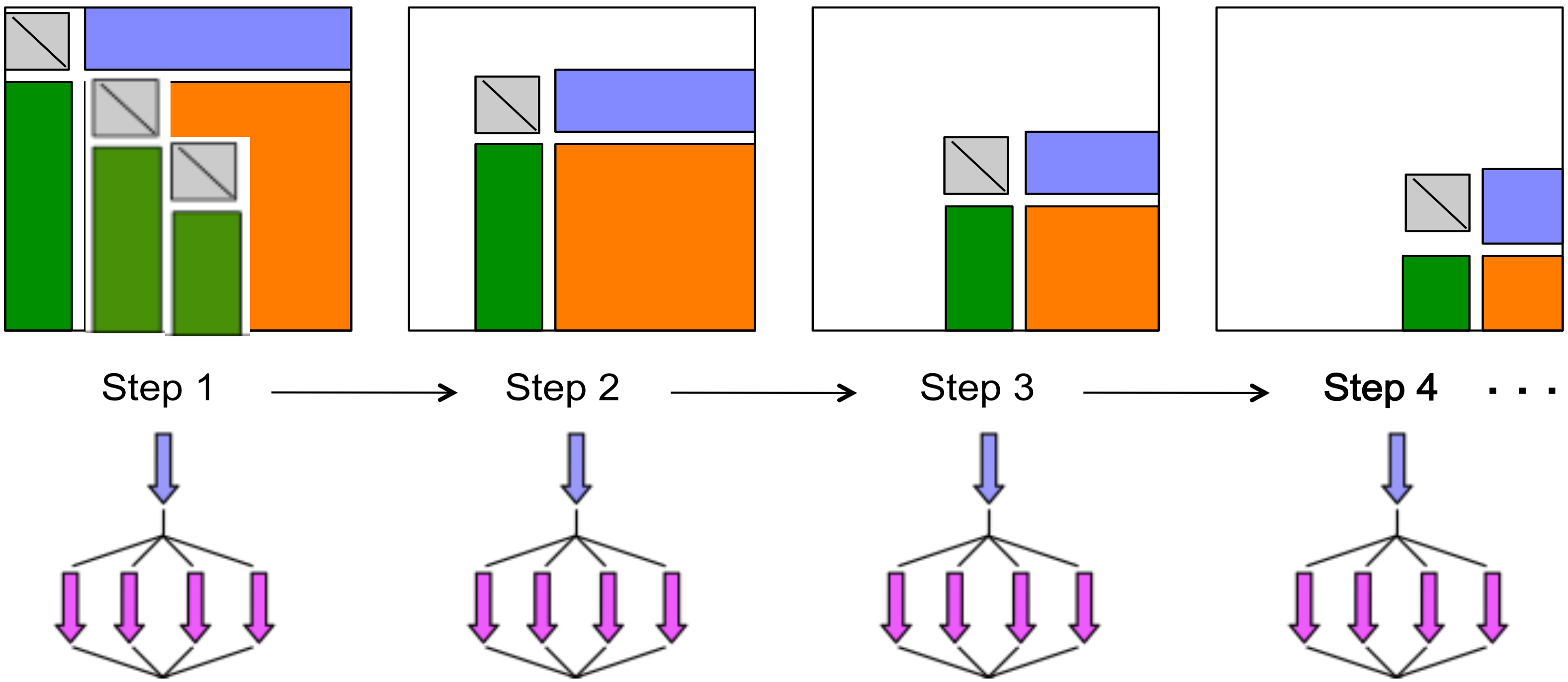
- Easy and done in any reasonable software.
- This is the $2/3n^3$ term in the FLOPs count.
- Can be done efficiently with LAPACK+multithreaded BLAS

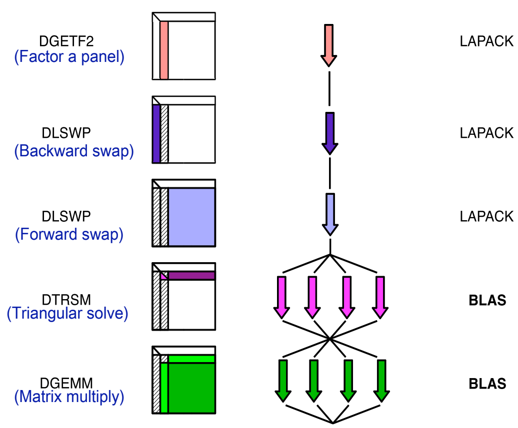




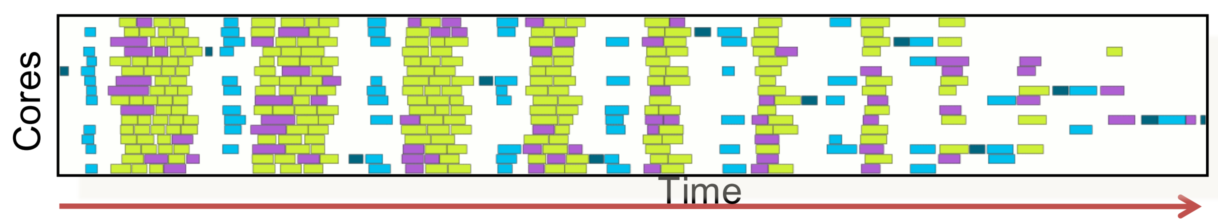
Fork - Join parallelism
Bulk Sync Processing

Synchronization (in LAPACK LU)





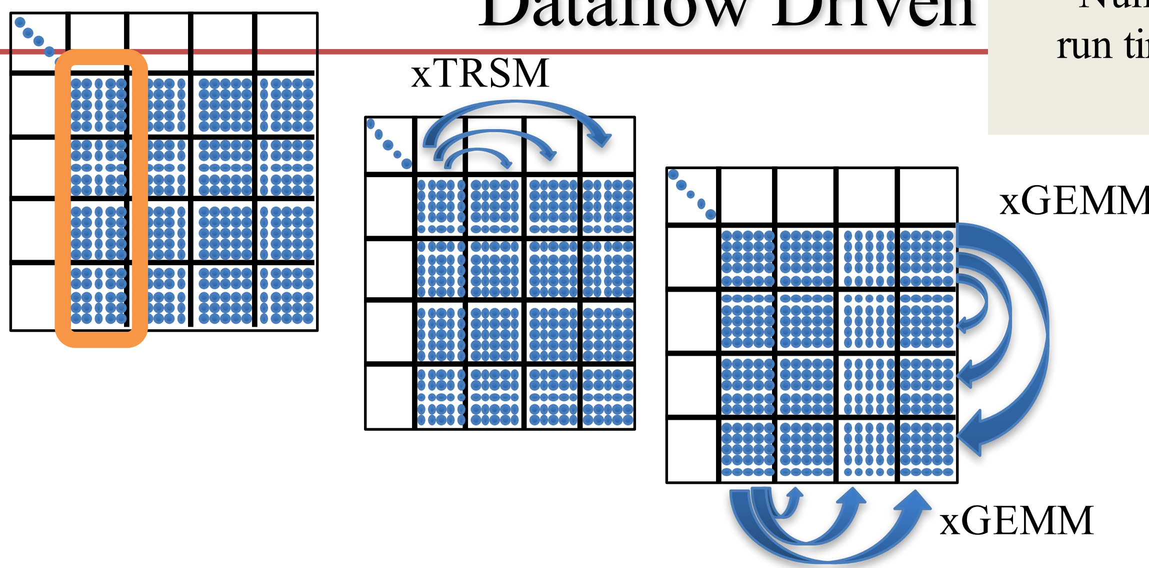
- fork join
- bulk synchronous processing



PLASMA LU Factorization

Dataflow Driven

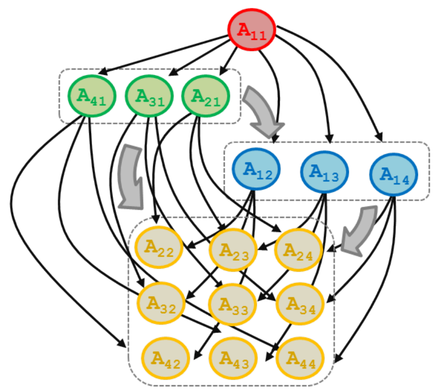
Numerical program generates tasks and run time system executes tasks respecting data dependences.



Sparse / Dense Matrix System

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

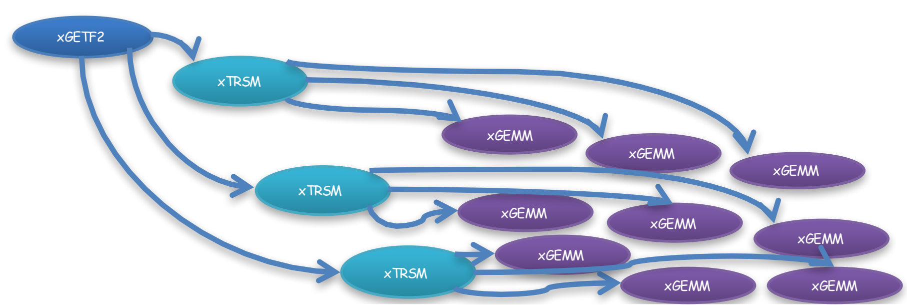
DAG-based factorization



Batched LA

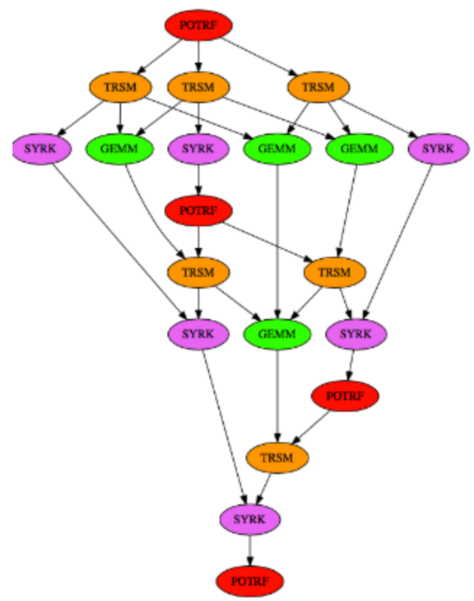
- **LU, QR,** or **Cholesky** on small diagonal matrices
- **TRSM**s, **QR**s, or **LU**s
- **TRSM**s, **TRMM**s
- **Updates (Schur complement) GEMM**s, **SYRK**s, **TRMM**s

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

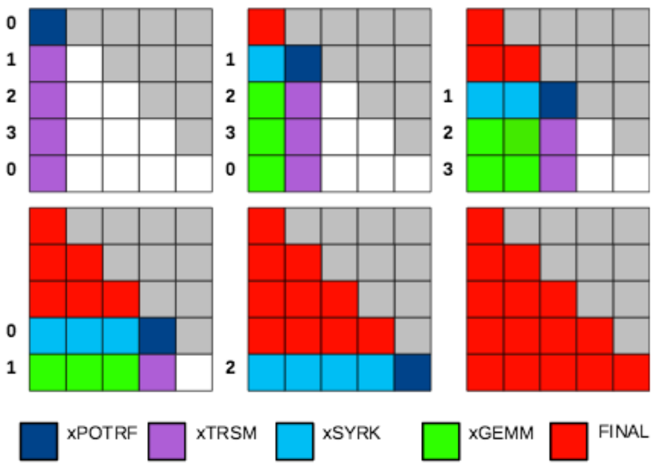


OpenMP tasking

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



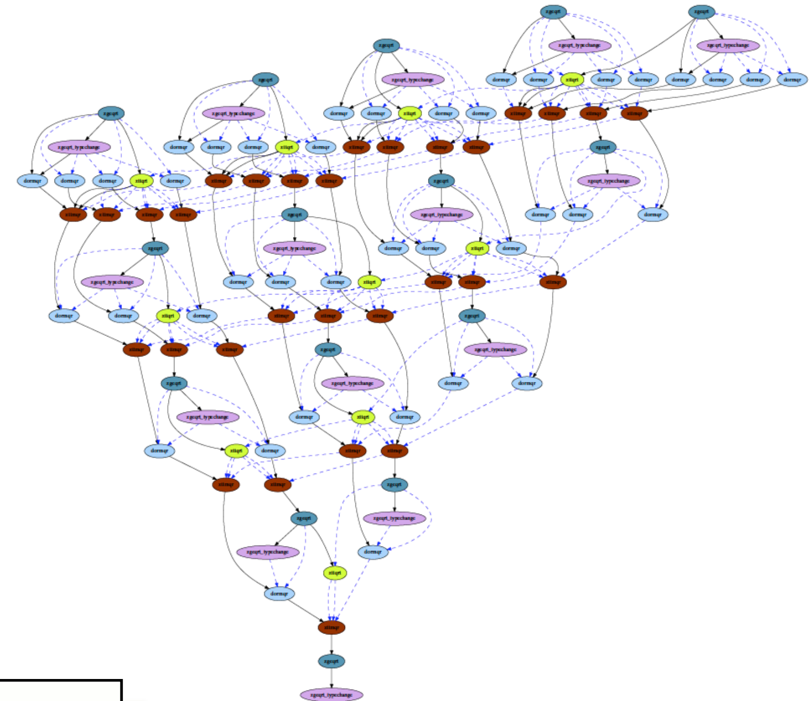
Tiled Cholesky Decomposition

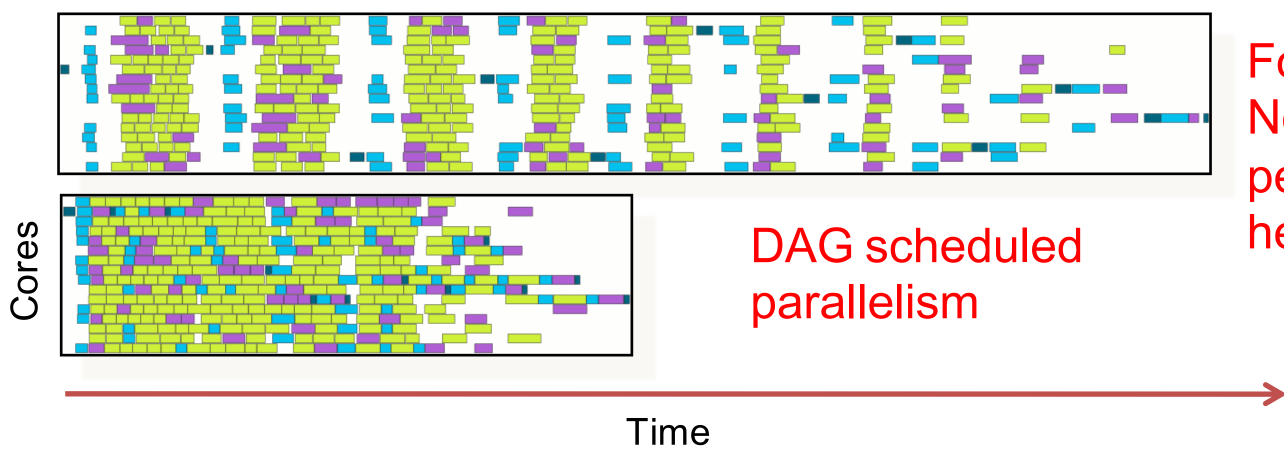



```
#pragma omp parallel
#pragma omp master
{  CHOLESKY( A );  }
CHOLESKY( A ) {
    for (k = 0; k < M; k++) {
        #pragma omp task depend(inout:A(k,k)[0:tilesize]
        {  POTRF( A(k,k) );  }
        for (m = k+1; m < M; m++) {
            #pragma omp task \
                depend(in:A(k,k)[0:tilesize])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

Avoiding Synchronization

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

Introduction

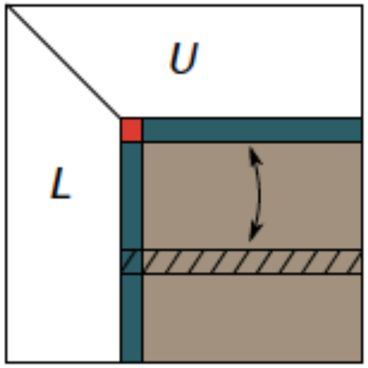
LU decomposition (Gaussian Elimination) for the solution of $Ax = b$

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

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

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

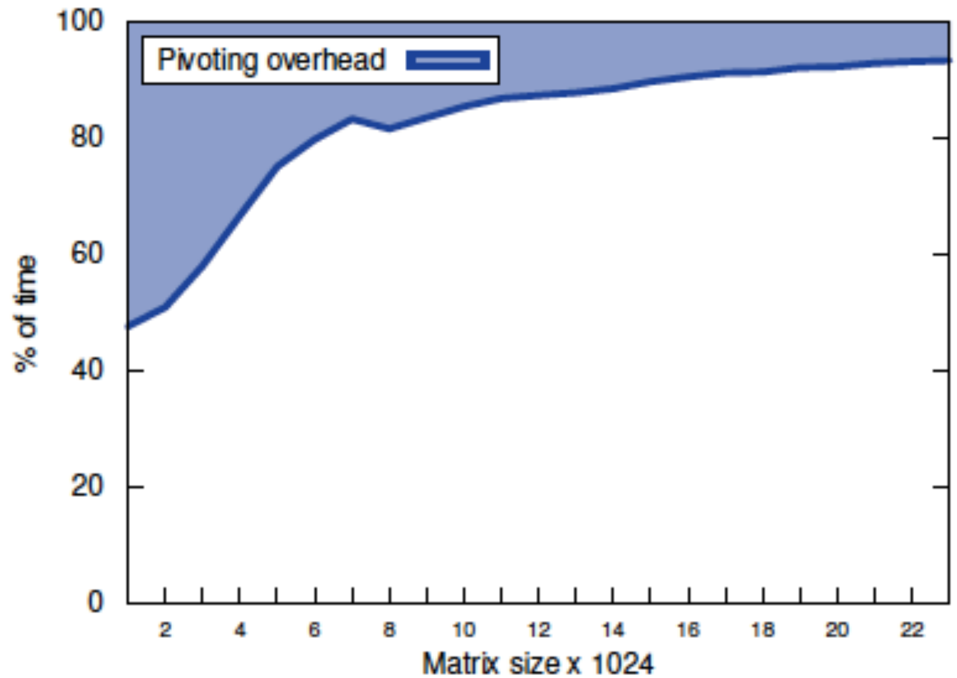
end for



- 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

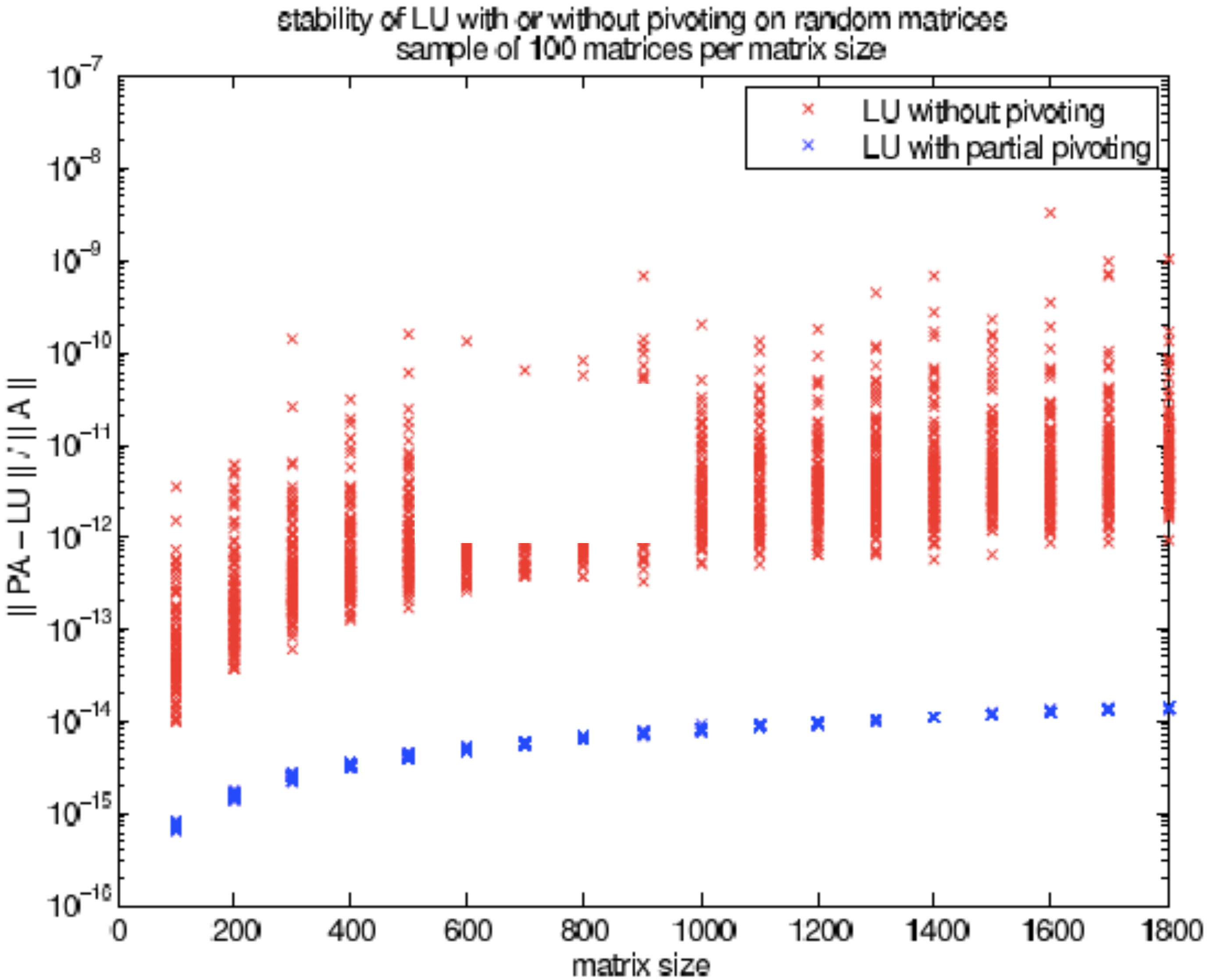
- Complete pivoting, partial pivoting, tournament pivoting, etc.
- GEPP implemented in most numerical libraries (LAPACK...)
- No floating point operation in pivoting but it involves irregular movements of data
- Communication overhead due to pivoting: $\mathcal{O}(n^2)$ comparisons



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)

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

1. Compute $A_r = U^T AV$ 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.

$B =$

Remark:

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

Numerical issues

Stability of RBT ?

- ”Average“ growth factor expressed in [Parker, 95]
- Iterative refinement is systematically added

  1: $r_k \leftarrow b - Ax_{k-1}$
  2: solve $Ly = r_k$
  3: solve $Uz_k = y$
  4: $x_k \leftarrow x_{k-1} + z_k$
  check convergence

- Backward error (available from IR process) is sent back
- Future work: probabilistic error bounds

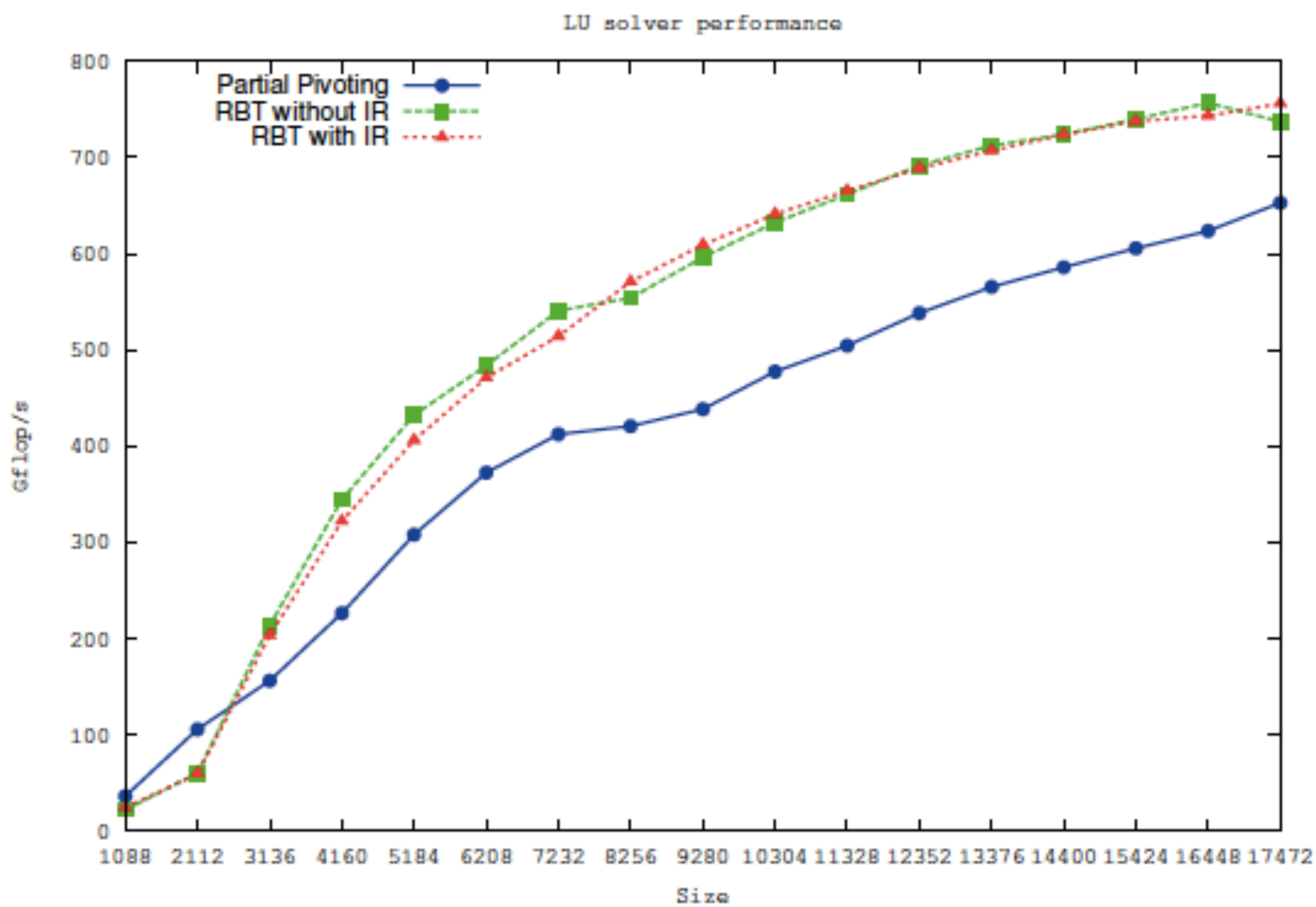
Tests on accuracy

Matrix	Cond	GENP	GEPP	QR	RBT	REC	IR
augment	$4 \cdot 10^4$	$1.28 \cdot 10^{-14}$	$2.28 \cdot 10^{-15}$	$2.99 \cdot 10^{-16}$	$2.81 \cdot 10^{-16}$	1	1
gfpp	$5 \cdot 10^2$	$9.01 \cdot 10^{-01}$	$6.88 \cdot 10^{-01}$	$1.06 \cdot 10^{-16}$	$1.27 \cdot 10^{-16}$	1	1
chebspec	$2 \cdot 10^{14}$	$1.19 \cdot 10^{-15}$	$3.29 \cdot 10^{-16}$	$5.22 \cdot 10^{-15}$	$3.23 \cdot 10^{-14}$	1	0
circul	$1 \cdot 10^3$	$1.74 \cdot 10^{-13}$	$1.66 \cdot 10^{-15}$	$2.66 \cdot 10^{-15}$	$2.66 \cdot 10^{-15}$	1	0
condex	$1 \cdot 10^2$	$7.32 \cdot 10^{-15}$	$5.98 \cdot 10^{-15}$	$8.34 \cdot 10^{-15}$	$6.50 \cdot 10^{-15}$	1	0
fiedler	$7 \cdot 10^5$	Fail	$2.11 \cdot 10^{-15}$	$1.54 \cdot 10^{-14}$	$7.90 \cdot 10^{-15}$	1	0
Hadamard	$1 \cdot 10^0$	$0 \cdot 10^0$	$0 \cdot 10^0$	$7.58 \cdot 10^{-16}$	$8.33 \cdot 10^{-15}$	1	0
normaldata	$3 \cdot 10^4$	$2.03 \cdot 10^{-12}$	$6.30 \cdot 10^{-15}$	$2.38 \cdot 10^{-16}$	$3.30 \cdot 10^{-16}$	1	1
orthog	$1 \cdot 10^0$	$5.64 \cdot 10^{-01}$	$4.33 \cdot 10^{-15}$	$3.70 \cdot 10^{-16}$	$4.31 \cdot 10^{-16}$	2	1
randcorr	$3 \cdot 10^3$	$5.12 \cdot 10^{-16}$	$4.04 \cdot 10^{-16}$	$5.73 \cdot 10^{-16}$	$5.92 \cdot 10^{-16}$	1	0
toeppd	$7 \cdot 10^5$	$2.53 \cdot 10^{-13}$	$2.60 \cdot 10^{-15}$	$8.39 \cdot 10^{-15}$	$5.71 \cdot 10^{-15}$	1	0
Foster	$5 \cdot 10^2$	$1 \cdot 10^0$	$1 \cdot 10^0$	$1.90 \cdot 10^{-16}$	$3.30 \cdot 10^{-16}$	2	1
$[-1, 1]$	$2 \cdot 10^3$	$2.19 \cdot 10^{-11}$	$5.19 \cdot 10^{-15}$	$2.33 \cdot 10^{-16}$	$2.35 \cdot 10^{-16}$	1	1
$[0, 1]$	$4 \cdot 10^4$	$1.97 \cdot 10^{-12}$	$2.85 \cdot 10^{-15}$	$2.15 \cdot 10^{-15}$	$1.79 \cdot 10^{-15}$	1	1
$\{-1, 1\}$	$4 \cdot 10^3$	Fail	$3.96 \cdot 10^{-15}$	$2.38 \cdot 10^{-16}$	$2.70 \cdot 10^{-16}$	2	1
$\{0, 1\}$	$5 \cdot 10^4$	Fail	$4.39 \cdot 10^{-15}$	$2.19 \cdot 10^{-15}$	$1.09 \cdot 10^{-15}$	2	1
Turing	$5 \cdot 10^{19}$	$0 \cdot 10^0$	$0 \cdot 10^0$	$7.16 \cdot 10^{-13}$	$1.05 \cdot 10^{-14}$	2	1
$\|i-j\|$	$7 \cdot 10^5$	Fail	$3.33 \cdot 10^{-16}$	$1.54 \cdot 10^{-14}$	$6.05 \cdot 10^{-15}$	1	0
$max(i,j)$	$3 \cdot 10^6$	$2.16 \cdot 10^{-14}$	$1.21 \cdot 10^{-15}$	$1.46 \cdot 10^{-14}$	$2.27 \cdot 10^{-15}$	1	1

Componentwise backward error ($n = 1024$, tile size=8)

$$\omega = \max_i \frac{|Ax-b|_i}{(|A|\cdot|x|+|b|)_i}$$

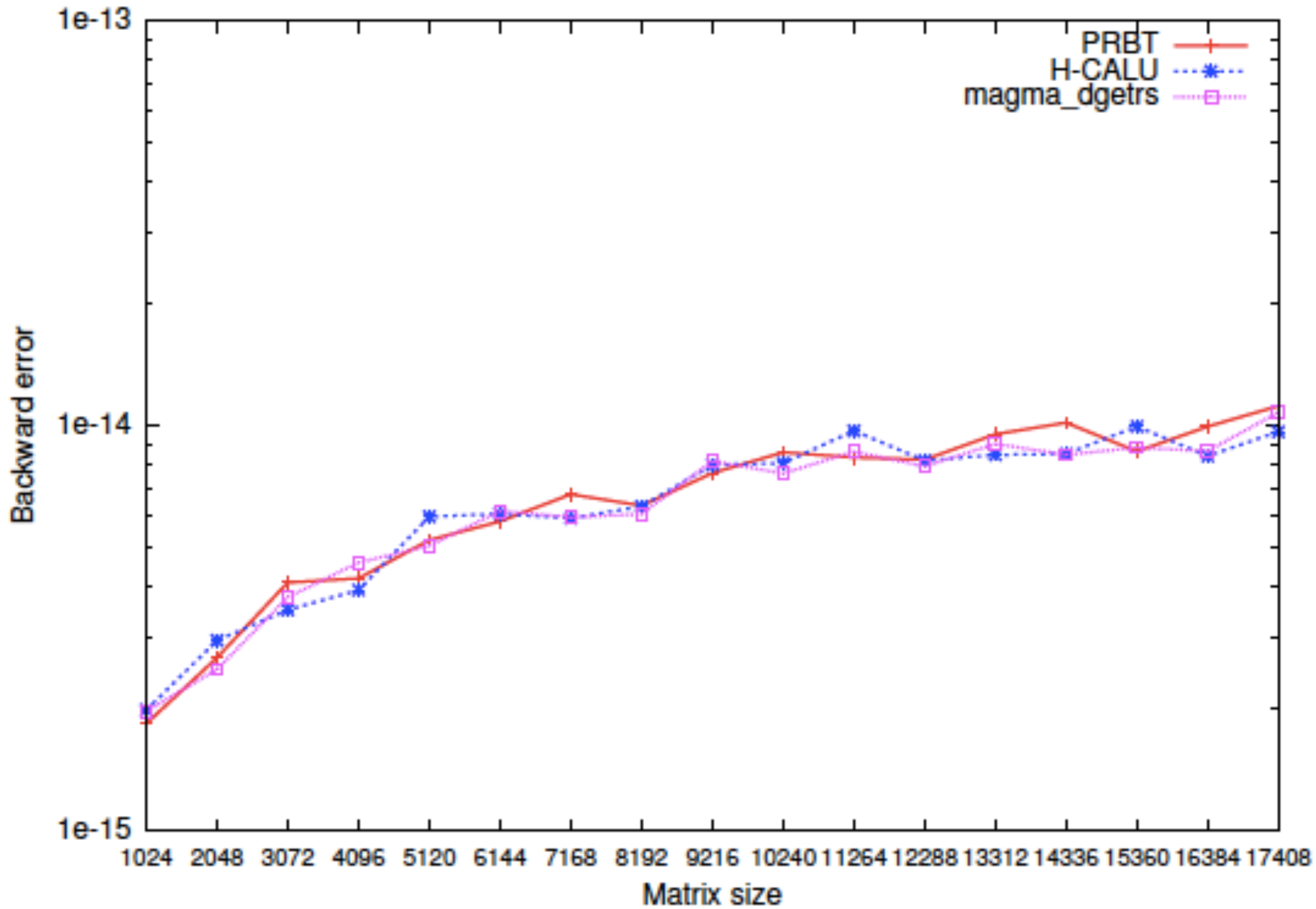
Performance on GPU



Performance on 12 Intel Xeon X5680 cores + 1 Nvidia Kepler K20

Using same number of flops used for each implementation.

RBT vs other solvers (accuracy)



Comparison of componentwise backward error (double precision)

Mixed Precision Methods

- **Mixed precision, use the lowest precision required to achieve a given accuracy outcome**
  - **Improves runtime, reduce power consumption, lower data movement**
  - **Reformulate to find correction to solution, rather than solution; Δx rather than x.**

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

$$x_{i+1} - x_i = -\frac{f(x_i)}{f'(x_i)}$$

42

Idea Goes Something Like This…

- Exploit 32 bit floating point as much as possible.
  - Especially for the bulk of the computation
- Correct or update the solution with selective use of 64 bit floating point to provide a refined results
- Intuitively:
  - Compute a 32 bit result,
  - Calculate a correction to 32 bit result using selected higher precision and,
  - Perform the update of the 32 bit results with the correction using high precision.

Mixed-Precision Iterative Refinement

- Iterative refinement for dense systems, $Ax = b$, can work this way.

L U = lu(A)	$O(n^3)$
x = L\(U\b)	$O(n^2)$
r = b – Ax	$O(n^2)$
WHILE || r || not small enough	
z = L\(U\r)	$O(n^2)$
x = x + z	$O(n^1)$
r = b – Ax	$O(n^2)$
END	

- Wilkinson, Moler, Stewart, & Higham provide error bound for SP fl pt results when using DP fl pt.

Mixed-Precision Iterative Refinement

- Iterative refinement for dense systems, $Ax = b$, can work this way.

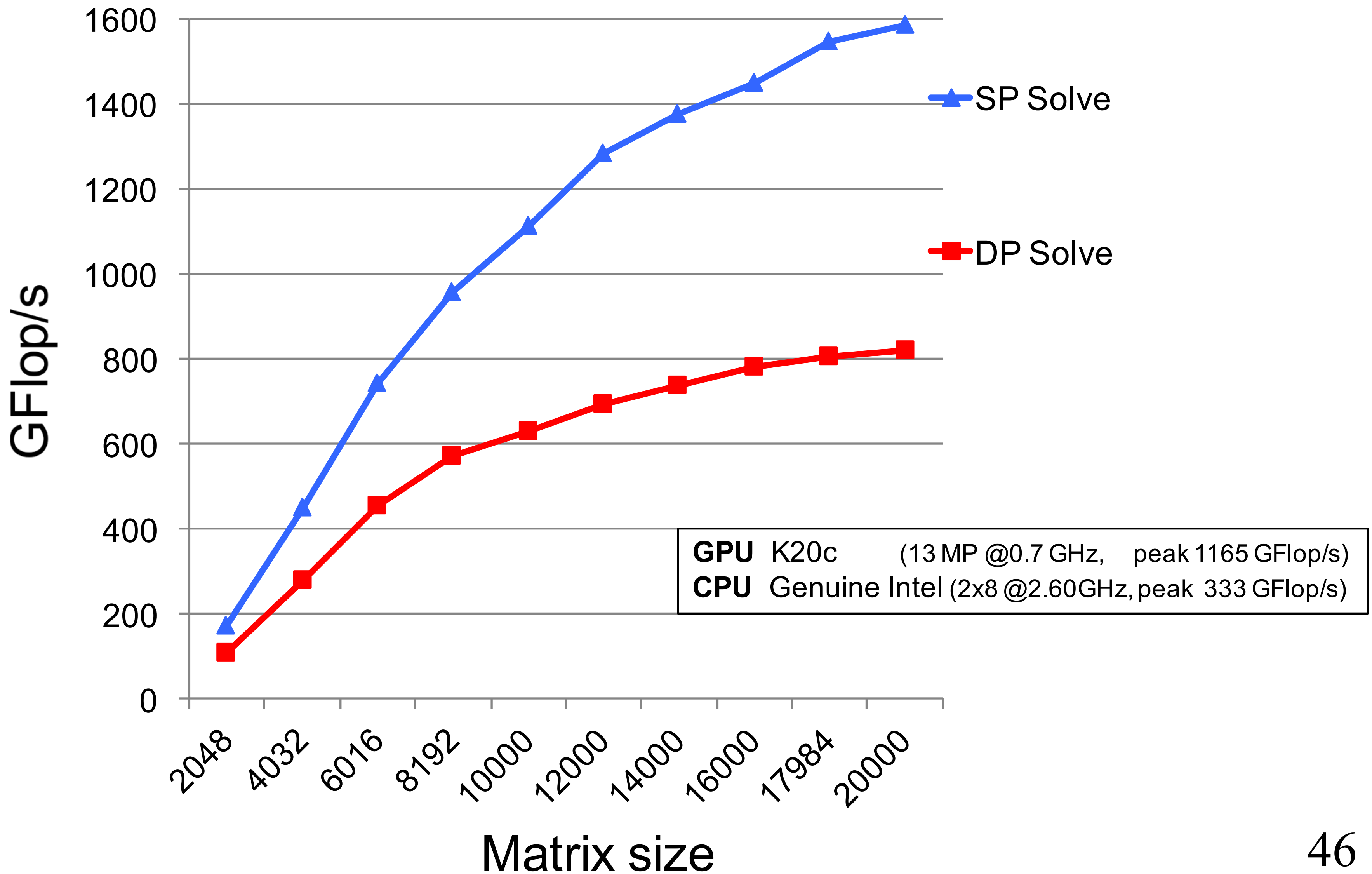
L U = lu(A)	SINGLE	$O(n^3)$
x = L\(U\b)	SINGLE	$O(n^2)$
r = b – Ax	DOUBLE	$O(n^2)$
WHILE || r || not small enough		
z = L\(U\r)	SINGLE	$O(n^2)$
x = x + z	DOUBLE	$O(n^1)$
r = b – Ax	DOUBLE	$O(n^2)$
END		

- Wilkinson, Moler, Stewart, & Higham provide error bound for SP fl pt results when using DP fl pt.
- It can be shown that using this approach we can compute the solution to 64-bit floating point precision.

> - Requires extra storage, total is 1.5 times normal;
> - $O(n^3)$ work is done in lower precision
> - $O(n^2)$ work is done in high precision
> - Problems if the matrix is ill-conditioned in sp; $O(10^8)$

Mixed precision iterative refinement

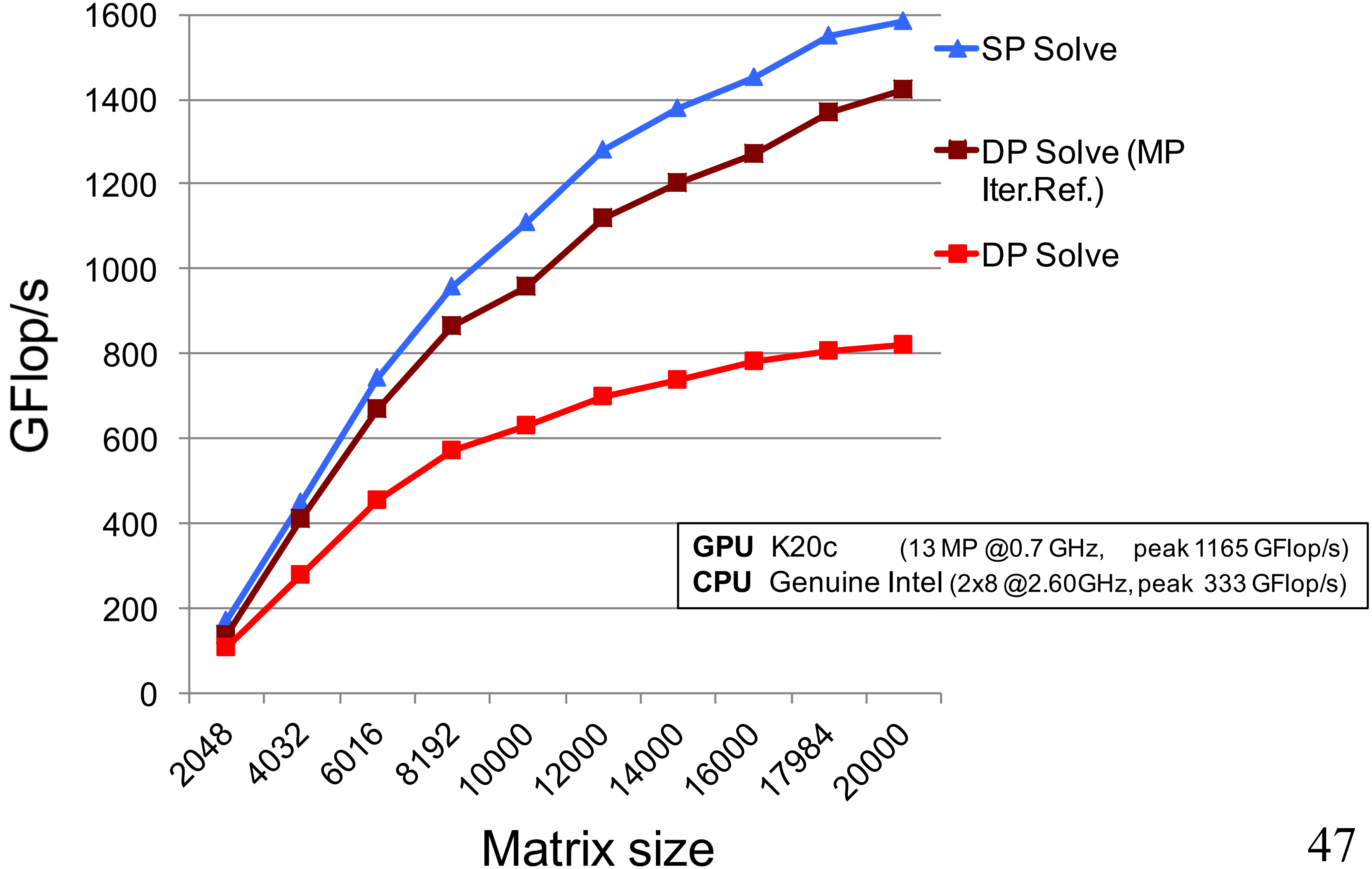
Solving general dense linear systems using mixed precision iterative refinement



Using same number of flops used for each implementation.

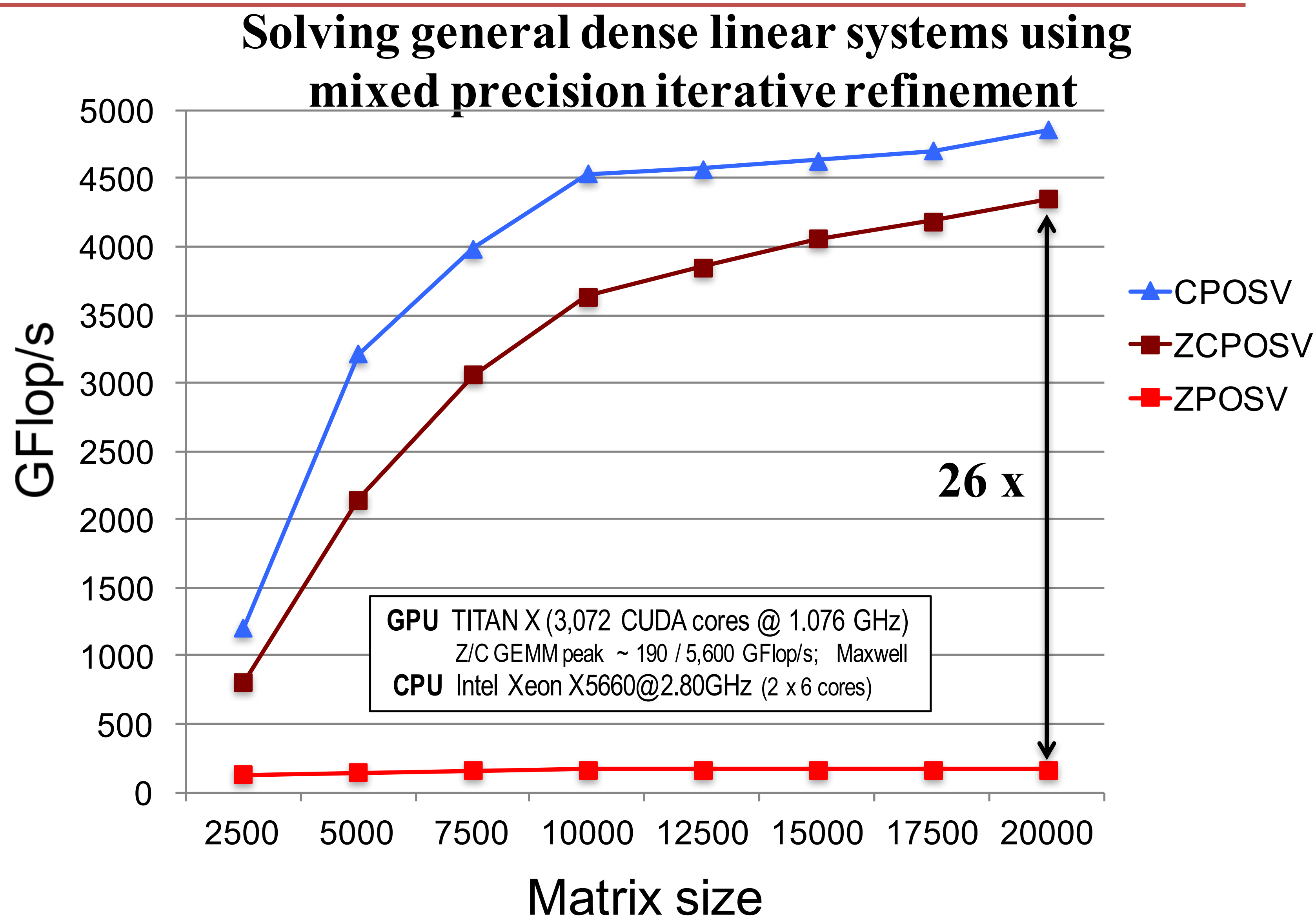
Mixed precision iterative refinement

Solving general dense linear systems using mixed precision iterative refinement



Using same number of flops used for each implementation.

Mixed precision iterative refinement



Using same number of flops used for each implementation.

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
    - 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**