An Accidental Benchmarker

Jack Dongarra

University of Tennessee

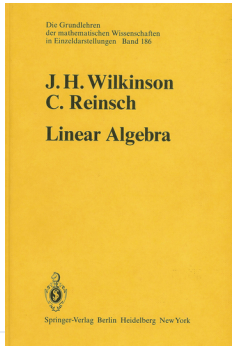
Oak Ridge National Laboratory

University of Manchester

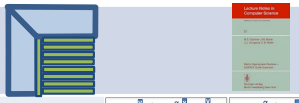
Over the Past 50 Years Evolving SW and Alg Tracking Hardware Developments

Features: Performance, Portability, and Accuracy			
EISPACK (1970's) (Translation of Algol to F66)			Rely on - Fortran, but row oriented

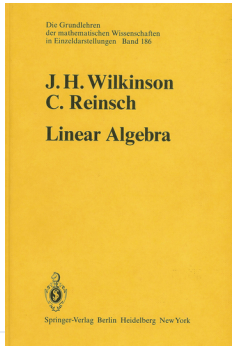
- **EISPACK** is a software library for numerical computation of eigenvalues and eigenvectors of matrices,
  - Written in FORTRAN.
  - Contains subroutines for calculating the eigenvalues of nine classes of matrices:
    - complex general, complex Hermitian, real general, real symmetric, real symmetric banded,
    - real symmetric tridiagonal, special real tridiagonal, generalized real, and
    - generalized real symmetric matrices.
  - The library drew heavily on Algol algorithms developed by Jim Wilkinson & colleagues.



Over the Past 50 Years Evolving SW and Alg Tracking Hardware Developments

Features: Performance, Portability, and Accuracy		
EISPACK (1970's) (Translation of Algol to F66)		Rely on - Fortran, but row oriented
Level 1 Basic Linear Algebra Subprograms (BLAS)		Standards for: Vector-Vector operations

- **EISPACK** is a software library for numerical computation of eigenvalues and eigenvectors of matrices,
  - Written in FORTRAN.
  - Contains subroutines for calculating the eigenvalues of nine classes of matrices:
    - complex general, complex Hermitian, real general, real symmetric, real symmetric banded,
    - real symmetric tridiagonal, special real tridiagonal, generalized real, and
    - generalized real symmetric matrices.
  - The library drew heavily on Algol algorithms developed by Jim Wilkinson & colleagues.



My First Paper

SOFTWARE—PRACTICE AND EXPERIENCE, VOL. 9, 219–226 (1979)

Unrolling Loops in FORTRAN*

J. J. DONGARRA AND A. R. HINDS
Argonne National Laboratory, Argonne, Illinois 60439, U.S.A.

SUMMARY

The technique of 'unrolling' to improve the performance of short program loops resorting to assembly language coding is discussed. A comparison of the benefit 'unrolling' on a variety of computers using an assortment of FORTRAN com presented.

- Reduces loop overhead
  - Level of unrolling dedicated by the instruction stack size
- Help the compiler to:
  - Facilitates pipelining
  - Increases the concurrence between independent functional units
- Provided ~15% performance improvement

TECHNIQUE

When a loop is unrolled, its contents are replicated one or more times, with appropriate adjustments to array indices and the loop increment. For instance, the DAXPY[9] sequence, which adds a multiple of one vector to a second vector:

```
      DO 10 I = 1,N
        Y(I) = Y(I)+A*X(I)
   10 CONTINUE
```

would, unrolled to a depth of four, assume the form

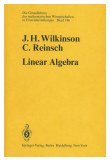
```
      M = N-MOD(N,4)
      DO 10 I = 1,M,4
         Y(I)   = Y(I)   +A*X(I)
         Y(I+1) = Y(I+1)+A*X(I+1)
         Y(I+2) = Y(I+2)+A*X(I+2)
         Y(I+3) = Y(I+3)+A*X(I+3)
   10 CONTINUE
```

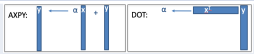
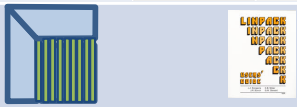
Basic Linear Algebra Subprograms for Fortran Usage

C. L. LAWSON
Jet Propulsion Laboratory
R. J. HANSON
Sandia Laboratories
D. R. KINCAID
The University of Texas at Austin
and
F. T. KROGH
Jet Propulsion Laboratory

A package of 38 low level subprograms for many of the basic operations of numerical linear algebra is presented. The package is intended to be used with Fortran. The operations in the package include dot product, elementary vector operation, Givens transformation, vector copy and swap, vector norm,

Over the Past 50 Years Evolving SW and Alg Tracking Hardware Developments



Features: Performance, Portability, and Accuracy		
EISPACK (1970's) (Translation of Algol to F66)		Rely on - Fortran, but row oriented
Level 1 Basic Linear Algebra Subprograms (BLAS)		Standards for: Vector-Vector operations
LINPACK (1980's) (Vector operations)		Rely on - Level-1 BLAS operations - Column oriented

An Accidental Benchmarker

LINPACK was an NSF Project w/ ANL, UNM, UM, & UCSD
We worked independently and came to Argonne in the summers

Top 23 List from 1977

UNIT = 10**6 TIME/(1/3 100**3 + 100**2)

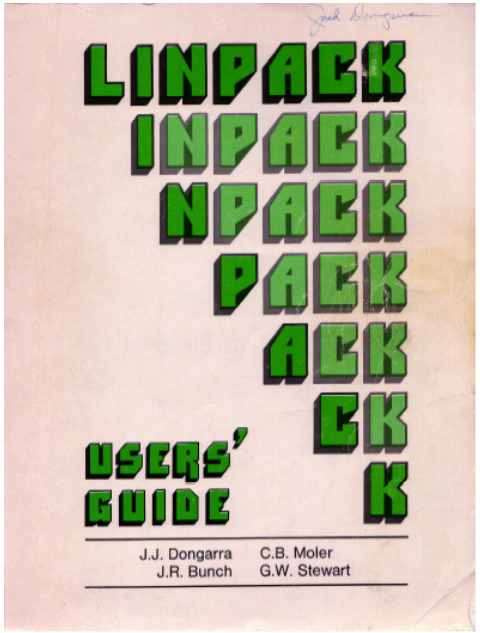
Facility	TIME N=100 secs.	UNIT micro-secs.	Computer	Type	Compiler
NCAR	.049	0.14	CRAY-1	S	CFT, Assembly BLAS
LASL	.148	0.43	CDC 7600	S	FTN, Assembly BLAS
NCAR	.192	0.56	CRAY-1	S	CFT
LASL	.210	0.61	CDC 7600	S	FTN
Argonne	.297	0.86	IBM 370/195	D	H
NCAR	.359	1.05	CDC 7600	S	Local
Argonne	.388	1.33	IBM 3033	D	H
NASA Langley	.489	1.42	CDC Cyber 175	S	FTN
U. Ill. Urbana	.506	1.47	CDC Cyber 175	S	Ext. 4.6
LLL	.554	1.61	CDC 7600	S	CHAT, No optimize
SLAC	.579	1.69	IBM 370/168	D	H Ext., Fast mult.
Michigan	.631	1.84	Amdahl 470/V6	D	H
Toronto	.890	2.59	IBM 370/165	D	H Ext., Fast mult.
Northwestern	1.44	4.20	CDC 6600	S	FTN
Texas	1.93*	5.63	CDC 6600	S	RUN
China Lake	1.95*	5.69	Univac 1110	S	V
Yale	2.59	7.53	DEC KL-20	S	F20
Bell Labs	3.46	10.1	Honeywell 6080	S	Y
Wisconsin	3.49	10.1	Univac 1110	S	V
Iowa State	3.54	10.2	Itel AS/5 mod3	D	H
U. Ill. Chicago	4.10	11.9	IBM 370/158	D	G1

Appendix B of the Linpack Users' Guide

- Designed to help users estimate the run time for solving systems of equation using the Linpack software.

First benchmark report from 1977;

- Cray 1 to DEC PDP-10

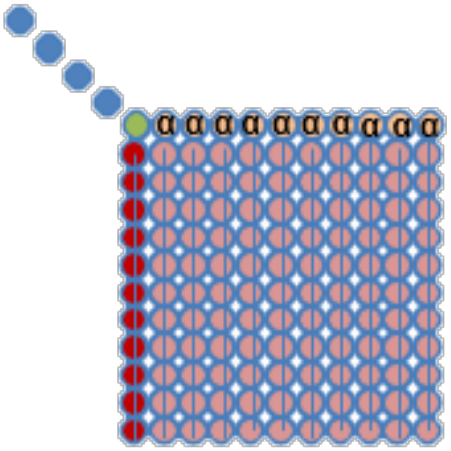


The Original Code

Benchmark based on solving Ax=b using LU factorization

Using Level 1 BLAS

Fortran 77



```
      subroutine dgefa(a,lda,n,ipvt,info)
      integer lda,n,ipvt(1),info
      double precision a(lda,1)
      double precision t
      integer idamax,j,k,kp1,l,nm1
      info = 0
      nm1 = n - 1
      if (nm1 .lt. 1) go to 70
      do 60 k = 1, nm1
         kp1 = k + 1
         l = idamax(n-k+1,a(k,k),1) + k - 1
         ipvt(k) = l
         if (a(l,k) .eq. 0.0d0) go to 40
         if (l .eq. k) go to 10
               t = a(l,k)
               a(l,k) = a(k,k)
               a(k,k) = t
10       continue
         t = -1.0d0/a(k,k)
         call dscal(n-k,t,a(k+1,k),1)
         do 30 j = kp1, n
               t = a(l,j)
               if (l .eq. k) go to 20
                  a(l,j) = a(k,j)
                  a(k,j) = t
20             continue
                call daxpy(n-k,t,a(k+1,k),1,a(k+1,j),1)
30          continue
         go to 50
40       continue
            info = k
50       continue
60 continue
70 continue
      ipvt(n) = n
      if (a(n,n) .eq. 0.0d0) info = n
      return
      end
```

# of Flops

$$\sum_{k=1}^{n-1}\sum_{k+1}^{n}\sum_{k+1}^{n} 2 = \frac{2}{3}n^3 + O(n^2)$$

Outer loop

Partial Pivoting
Find the largest element of a column & interchange

Scale column

Middle loop

Apply pivot

Inner Loop axpy
y = αx + y

Ax=b; factor A, then solve

Pivoting needed to
- Avoid dividing by zero or a small number
  - Degeneracy
- Avoid growth in the elements
  - Loss of accuracy

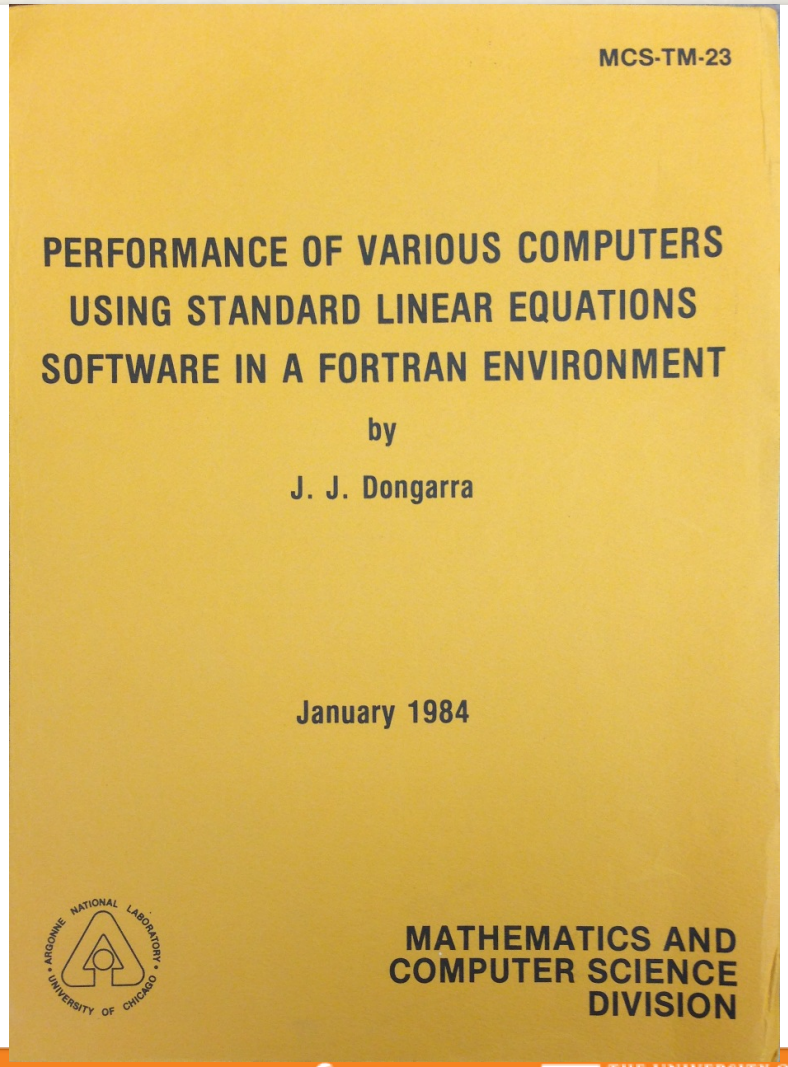
Factor *A* into *L* and *U*
Then solve the system of equations.
$y = L^{-1} b;\ x = U^{-1} y$

About $\frac{2}{3}n^3 + O(n^2)$ floating point ops and $\frac{2}{3}n^3 + O(n^2)$ touches in the factorization and $O(n^2)$ for the solves.

Rate of execution measured as:
# of Operations/Time or
Floating point operations per second (Flops)

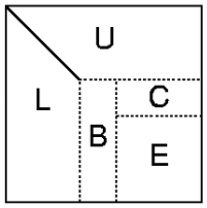
Linpack Benchmark Characteristics

- Portable, runs on any system
- Easy to understand
  One number, rate of execution (Flops/second)
- Algorithm fixed
- Allows for restructuring/optimizing algorithm
- All performance data with the same arithmetic precision, 64-bit floating point.
- Benchmark checks if "correct solution" achieved:
  || Ax-b || / (||A|| ||x||+||b||)
- Not intended to measure entire machine performance.
- In the benchmark report, "One further note: The following performance data should not be taken too seriously."

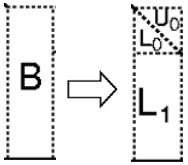


LU Decomposition Algorithm Has Evolved

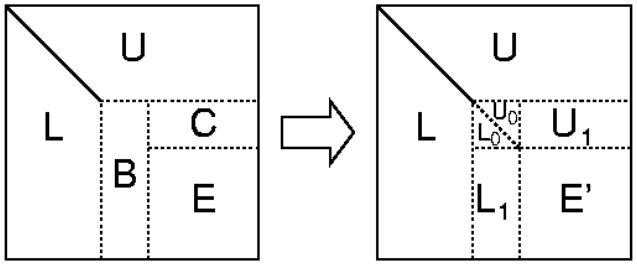
Instead of working with a columns of the matrix we work with panels, a collection of columns.



The algorithm proceeds by doing an LU factorization of the panel B.



Then updating *C*, $(U_1 = L_0^{-1}C)$, and then applying all the transformation to the rest of the matrix, $E' = E - L_1U_1$



This results in a rank-k update of *E* or a matrix multiply (GEMM).

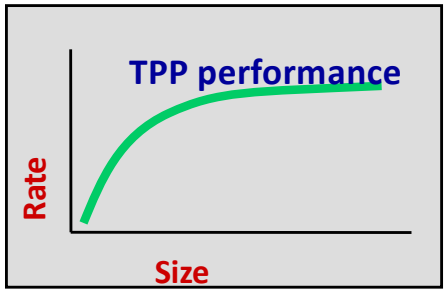
(GEMM: $O(n^3)$ ops & $O(n^2)$ references)

Same number of operations and same numerical properties.

Top500 Since 1993

- Since 1978 I maintained a LINPACK Benchmark list.
- Hans Meuer and Erich Strohmaier had a list of fastest computers ranked by peak performance.
- Listing of the 500 most powerful computers in the World.
- Yardstick: Performance for

  Ax=b, dense problem



Maintained and updated twice a year:

SC'xy in the States in November

Meeting in Germany in June

Top500/HPL Benchmark Timeline $Ax=b$

- 1974 LINPACK library (vector computers)
- 1977 LINPACK 100 Benchmark (Fortran only)
- 1986 LINPACK 1000 Benchmark (assembly allowed)
- **1988 First Gflop/s system, NEC SX-2**
- 1991 LINPACK Table 3 (HPL rules defined)

- **1993 Top500 starts (LANL CM-5 #1, 60 Gflop/s)**
- 1994 All Top500 systems over 1 Gflop/s
- **1997 First Tflop/s system ASCI Red @ SNL**
- 2001 China has its first computer on Top500
- 2002 #1 Earth Simulator @ JAMSTEC 5 X faster
- 2005 All Top500 systems over 1 Tflop/s

- **2008 First Pflop/s system Roadrunner @ LANL**
- 2010 Tianhe-1A @ Tianjin (2 Pflop/s) 1st time China #1
- 2011 K computer @ RIKEN (8 Pflop/s)
- 2012 Sequoia @ LLNL (1.5 M cores)
- 2014 Tianhe-2 @ NUDT (33 Pflop/s)
- 2016 China and US have = number of systems
- 2016 TaihuLight @ Wuxi (93 Pflop/s, >10 M cores)
- 2018 Summit @ ORNL (122 Pflop/s)
- 2019 All TOP500 systems over 1Pflop/s
- 2020 Fugaku @ RIKEN (442 Pflop/s)
- 2021 Of the Top500: China=186 & US=123
- **2021 First Exascale System?**

#1 Systems on the Top500 Over the Past 27 Years

Top500 List (# of times)	Computer	HPL r_{max} (Tflop/s)	Procs/Cores	Matrix Size	Hours To BM	MW
6/93 (1)	TMC CM-5/1024 (DOE LANL)	.060	1,024	52,224	0.4	
11/93 (1)	Fujitsu Numerical Wind Tunnel (Nat. Aerospace Lab of Japan)	.124	140	31,920	0.1	1.
6/94 (1)	Intel XP/S140 (DOE SNL)	.143	3,680	55,700	0.2	
11/94-11/95 (3)	Fujitsu Numerical Wind Tunnel (Nat. Aerospace Lab of Japan)	.170	140	42,000	0.1	1.
6/96 (1)	Hitachi SR2201/1024 (Univ. of Tokyo)	.220	1,024	138,240	2.2	
11/96 (1)	Hitachi CP-PACS/2048 (Univ of Tsukuba)	.368	2,048	103,680	0.6	
6/97-6/00 (7)	Intel ASCI Red (DOE SNL)	2.38	9,632	362,880	3.7	.85
11/00-11/01 (3)	IBM ASCI White, SP Power3 375 MHz (DOE LLNL)	7.23	8,192	518,096	3.6	
6/02-6/04 (5)	NEC Earth-Simulator (JAMSTEC)	35.9	5,120	1,000,000	5.2	6.4
11/04-11/07 (7)	IBM BlueGene/L (DOE LLNL)	478.	212,992	1,000,000	0.4	1.4
6/08-6/09 (3)	IBM Roadrunner –PowerXCell 8i 3.2 Ghz (DOE LANL)	1,105.	129,600	2,329,599	2.1	2.3
11/09-6/10 (2)	Cray Jaguar - XT5-HE 2.6 GHz (DOE ORNL)	1,759.	224,162	5,474,272	17	6.9
11/10 (1)	NUDT Tianhe-1A, X5670 2.93Ghz NVIDIA (NSC Tianjin)	2,566.	186,368	3,600,000	3.4	4.0
6/11-11/11 (2)	Fujitsu K computer, SPARC64 VIIIfx (RIKEN)	10,510.	705,024	11,870,208	29	9.9
6/12 (1)	IBM Sequoia BlueGene/Q (DOE LLNL)	16,324.	1,572,864	12,681,215	23	7.9
11/12 (1)	Cray XK7 Titan AMD + NVIDIA Kepler (DOE ORNL)	17,590.	560,640	4,423,680	0.9	8.2
6/13-11/15 (6)	NUDT Tianhe-2 Intel IvyBridge + Xeon Phi (NCSS Guangzhou)	33,862.	3,120,000	9,960,000	5.4	17.8
6/16-11/17 (4)	Sunway TaihuLight System (NSCC Wuxi)	93,014.	10,549,600	12,288,000	3.7	15.4
6/18-11/19 (4)	IBM Summit Power9 + Nvidia Volta (DOE ORNL)	148,600	2,414,592	16,473,600	3.3	10.1
6/20-?	Fujitsu Fugaku ARM A64FX (RIKEN)	442,010	7,630,828	21,288,960	4.4	29.9

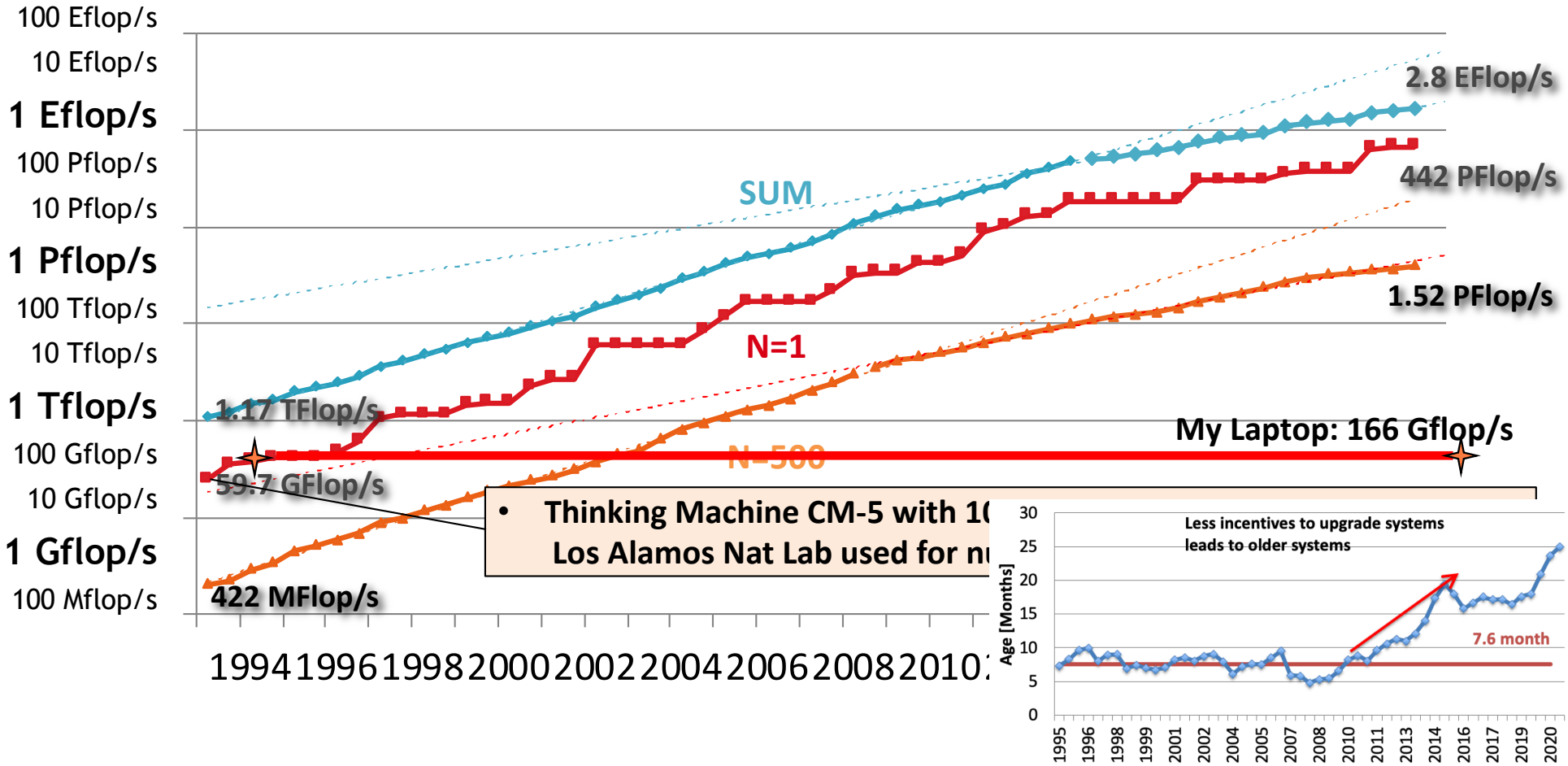
10
7
3

DOE
LANL: 2
SNL: 2
LLNL: 3
ORNL: 3

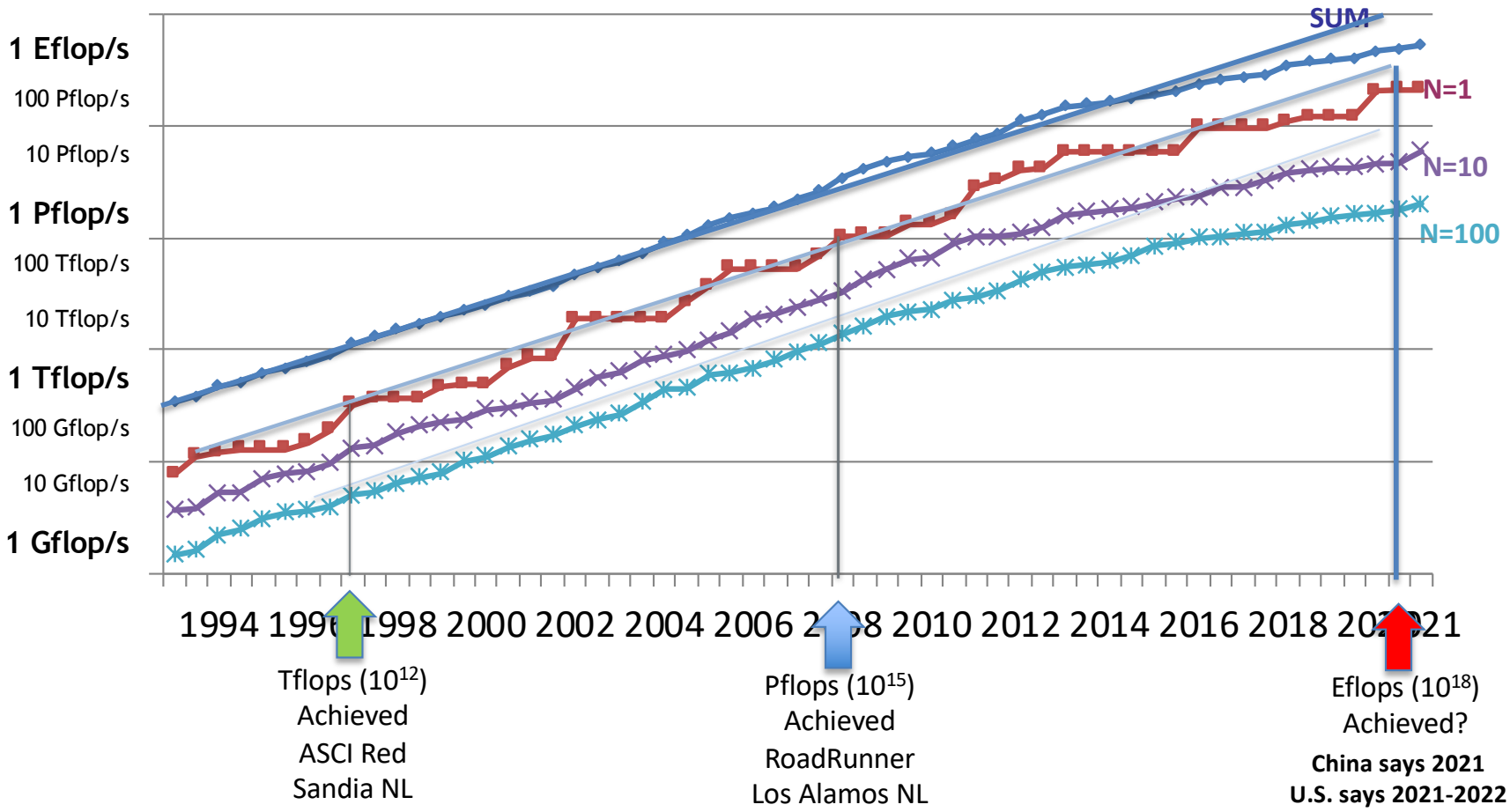
State of Supercomputing in 2021

- Pflops (> 10^{15} Flop/s) computing fully established with all 500 systems.
- Three technology architecture possibilities or "swim lanes" are thriving.
  - Commodity (e.g. Intel)
  - Commodity + accelerator (e.g. GPUs) (145 systems; 138 NVIDIA, 3 Intel Phi + 4)
  - Lightweight cores (e.g. IBM BG, TaihuLight)
- China: Top consumer and producer overall.
- Interest in supercomputing is now worldwide, and growing in many new markets (~50% of Top500 computers are in industry).
- Intel processors largest share, 86% followed by AMD, 10%.
- Exascale (10^{18} Flop/s) projects exist in many countries and regions.

PERFORMANCE DEVELOPMENT OF HPC OVER THE LAST 28 YEARS FROM THE TOP500



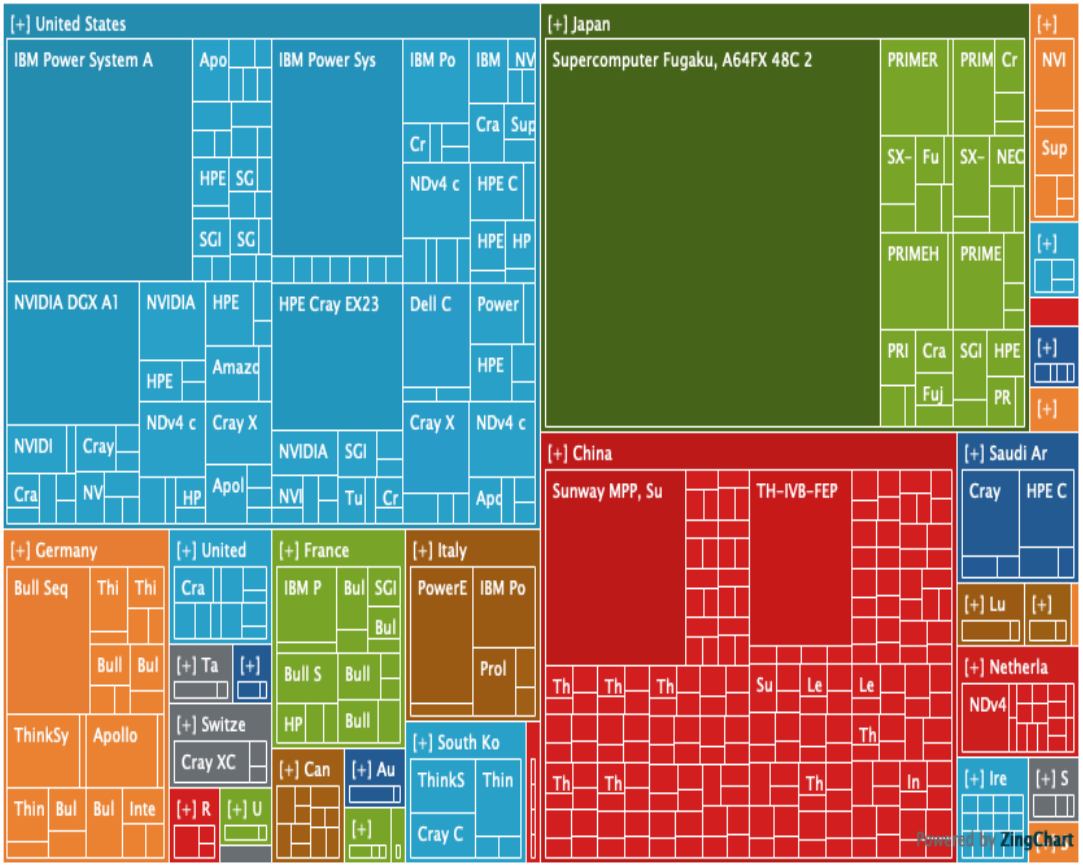
PERFORMANCE DEVELOPMENT

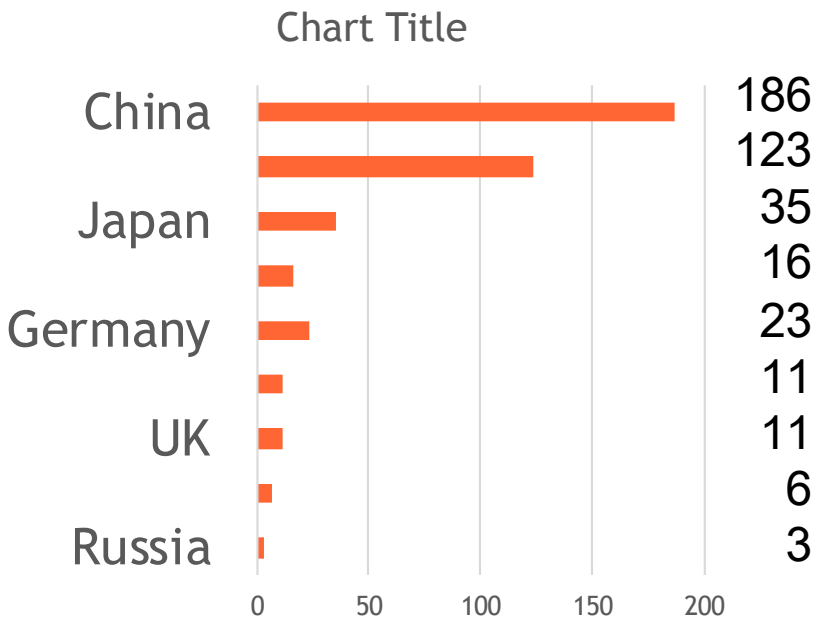


June 2021: The TOP 10 Systems (42% of the Total Performance of Top500)

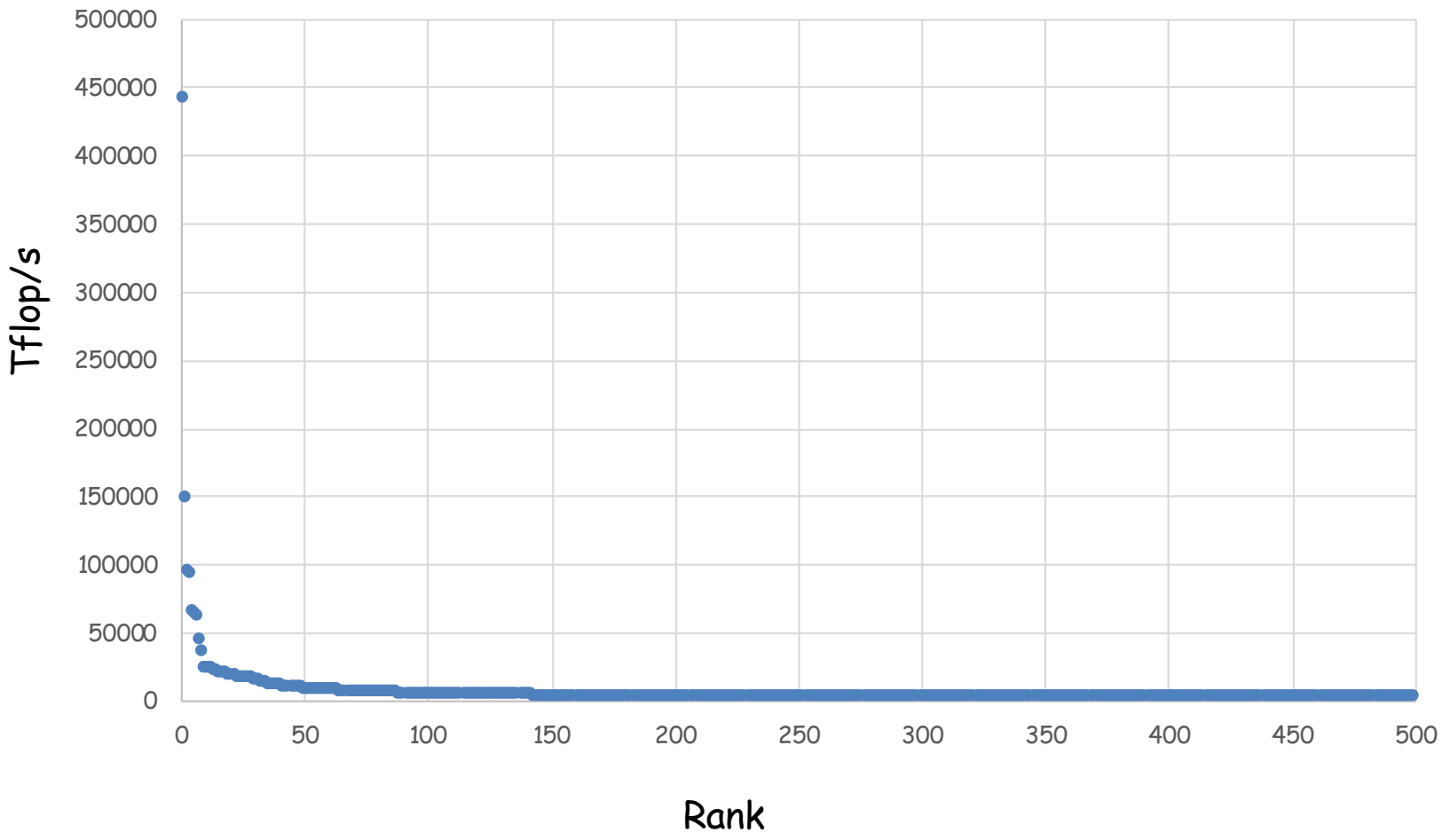
Rank	Site	Computer	Country	Cores	Rmax [Pflops]	% of Peak	Power [MW]	GFlops/ Watt
1	RIKEN Center for Computational Science	Fugaku, ARM A64FX (48C, 2.2 GHz), Tofu D Interconnect	Japan	7,299,072	442.	82	29.9	14.8
2	DOE / OS Oak Ridge Nat Lab	Summit, IBM Power 9 (22C, 3.0 GHz), NVIDIA GV100 (80C), Mellonox EDR	USA	2,397,824	149.	74	10.1	14.7
3	DOE / NNSA L Livermore Nat Lab	Sierra, IBM Power 9 (22C, 3.1 GHz), NVIDIA GV100 (80C), Mellonox EDR	USA	1,572,480	94.6	75	7.44	12.7
4	National Super Computer Center in Wuxi	Sunway TaihuLight, SW26010 (260C) + Custom	China	10,649,000	93.0	74	15.4	6.05
5	DOE / OS NERSC - LBNL	Perlmutter HPE Cray EX235n, AMD EPYC 64C 2.45GHz, NVIDIA A100, Slingshot-10	USA	706,304	64.6	69	2.53	25.5
6	NVIDIA Corporation	Selene NVIDIA DGX A100, AMD EPYC 7742 (64C, 2.25GHz), NVIDIA A100 (108C), Mellanox HDR Infiniband	USA	555,520	63.4	80	2.64	23.9
7	National Super Computer Center in Guangzhou	Tianhe-2A NUDT, Xeon (12C) + MATRIX-2000 (128C) + Custom	China	4,981,760	61.4	61	18.5	3.32
8	JUWELS Booster Module	Bull Sequana XH2000 , AMD EPYC 7402 (24C, 2.8GHz), NVIDIA A100 (108C), Mellanox HDR InfiniBand/ParTec ParaStation ClusterSuite	Germany	448,280	44.1	62	1.76	25.0
9	Eni S.p.A in Italy	HPC5, Dell EMC PowerEdge C4140, Xeon (24C, 2.1 GHz) + NVIDIA V100 (80C), Mellonax HDR	Italy	669,760	35.5	69	2.25	15.8
10	Texas Advanced Computing Center / U of Texas	Frontera, Dell C6420, Xeon Platinum, 8280 (28C, 2.7 GHz), Mellanox HDR	USA	448,448	23.5	61		

Countries Share



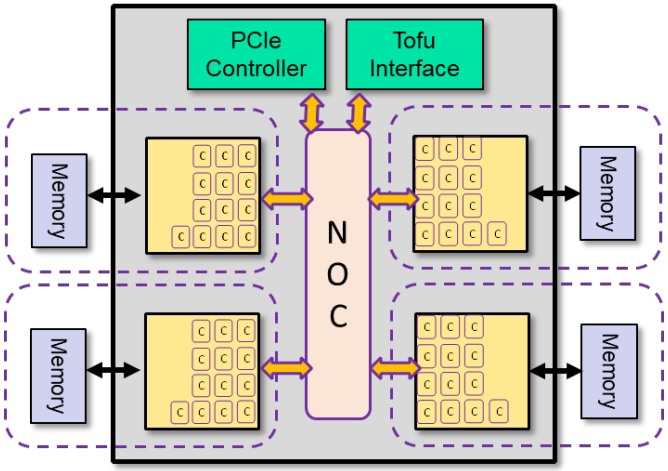


Plot of Top500 from June 2021



1 Fugaku's Fujitsu A64fx Processor is…

- A Many-Core ARM CPU…
  - 48 compute cores + 2 or 4 assistant (OS) cores
  - New core design
  - Near Xeon-Class Integer performance core
  - ARM V8 --- 64bit ARM ecosystem
  - Interconnect Tofu-D
  - 3.4 TFLOP/s Peak 64-bit performance

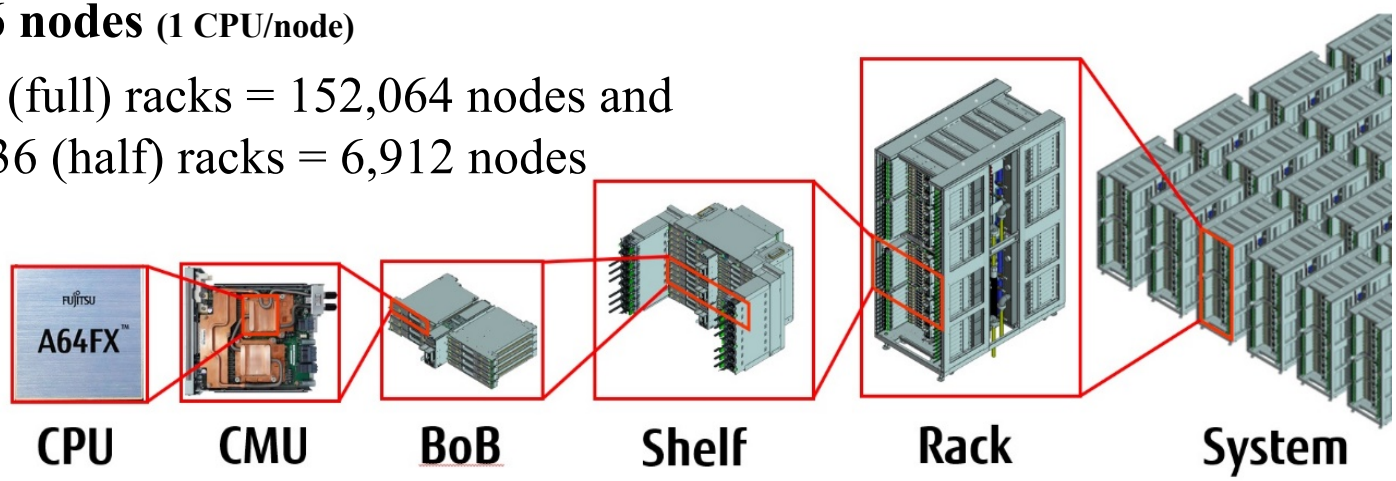


- …but also an accelerated GPU-like processor
  - SVE 512 bit x 2 vector extensions (ARM & Fujitsu)
    - Integer (1, 2, 4, 8 bytes) + Float (16, 32, 64 bytes)
  - Cache + memory localization (sector cache)
  - HBM2 on package memory – Massive Mem BW (Bytes/DPF ~0.4)
    - Streaming memory access, strided access, scatter/gather etc.
  - Intra-chip barrier synch. and other memory enhancing features

http://bit.ly/fugaku-report

Fugaku Total System Config & Performance

- **Total # Nodes: 158,976 nodes** **(1 CPU/node)**
  - 384 nodes/rack x 396 (full) racks = 152,064 nodes and
    192 nodes/rack x 36 (half) racks = 6,912 nodes



Footprint: 1,920 m^2

- **Theoretical Peak Compute Performances**
  - Normal Mode (CPU Frequency 2GHz)
    - **64 bit** Double Precision FP: **488 Petaflops**
    - **32 bit** Single Precision FP: **977 Petaflops**
    - **16 bit** Half Precision FP (AI training): **1.95 Exaflops**
    - **8 bit Integer** (AI Inference): **3.90 Exaops**
- **Theoretical Peak Memory BW: 163 Petabytes/s**

Fugaku represents 16% of all the other Top500 systems.

http://bit.ly/fugaku-report

Current #2 System Overview

System Performance

- Peak performance of 200 Pflop/s for modeling & simulation
- Peak performance of 3.3 Eflop/s for 16 bit floating point used in for data analytics, ML, and artificial intelligence

Each node has

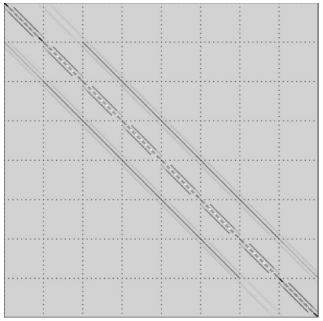
- 2 IBM POWER9 processors
  - Each w/22 cores
  - 2.3% performance of system
- 6 NVIDIA Tesla V100 GPUs
  - Each w/80 SMs
  - 97.7% performance of system
- 608 GB of fast memory
- 1.6 TB of NVMe memory

The system includes

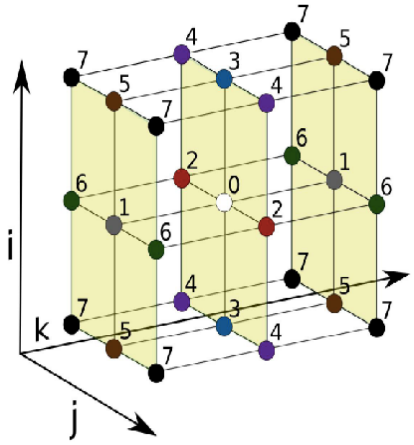
- 4608 nodes
  - 27,648 GPUs
  - Street value $10K each
- Dual-rail Mellanox EDR InfiniBand network
- 250 PB IBM Spectrum Scale file system transferring data at 2.5 TB/s

hpcg-benchmark.org

HPCG Results; The Other Benchmark



- High Performance Conjugate Gradients (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

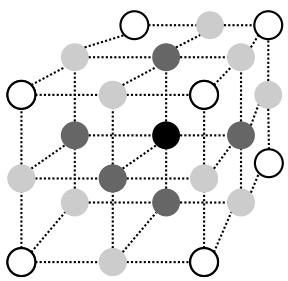


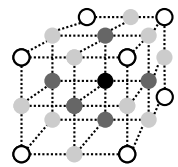
27-point stencil operator

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

HPCG Details

3D Laplacian discretization

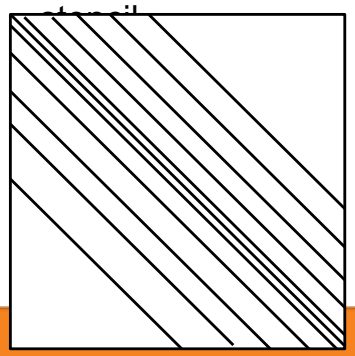




Multigrid

$L[u] \equiv \nabla^2 u$
$= f$

Sparse matrix based on 27-point stencil



Au
$= f$

Preconditioned Conjugate Gradient solver

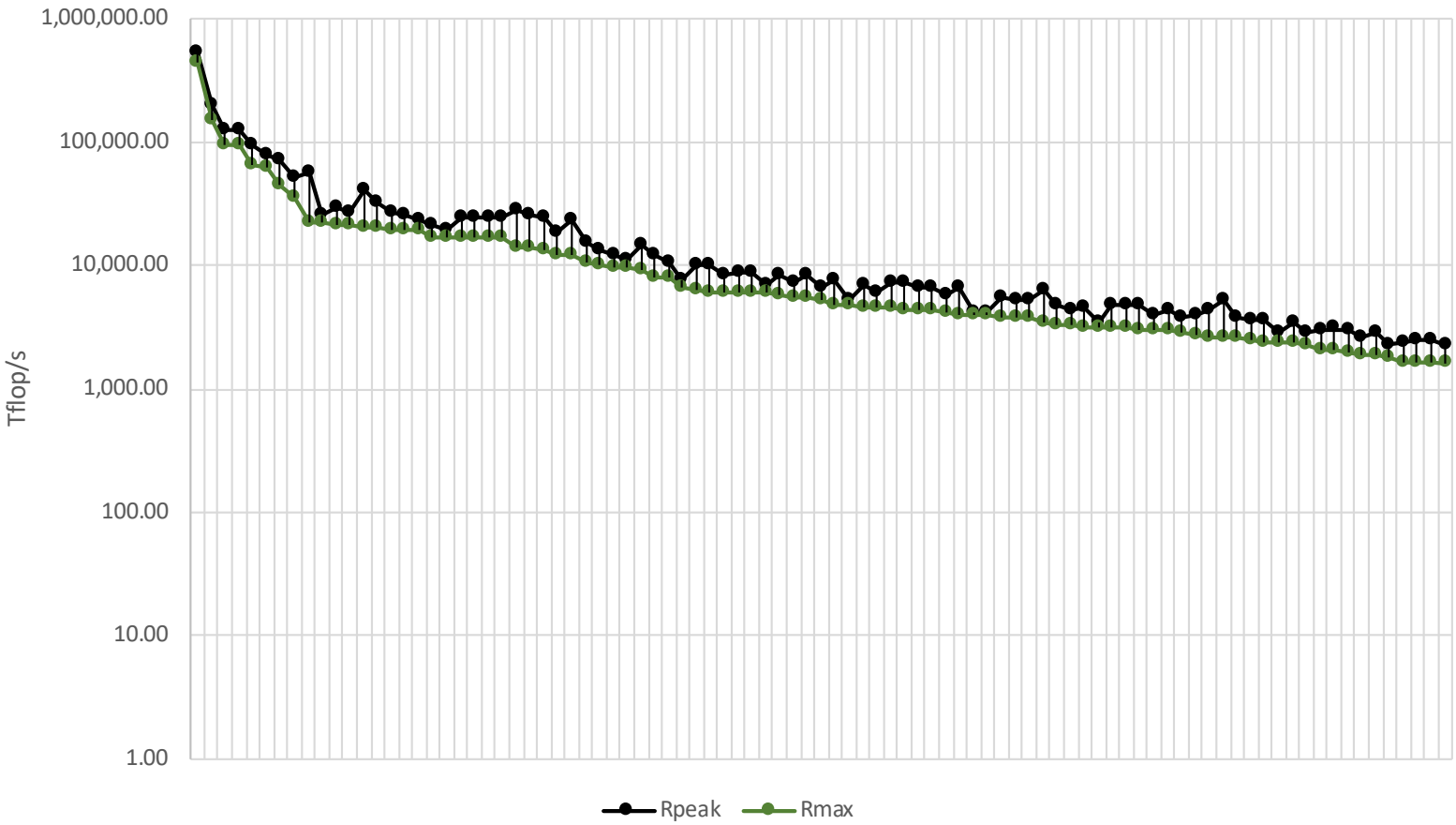
$p_0 \leftarrow x_0,\ r_0 \leftarrow b - Ap_0$
for $i = 1, 2,$ to max_iterations **do**
  $z_i \leftarrow M^{-1} r_{i-1}$
  if $i = 1$ **then**
    $p_i \leftarrow z_i$
    $\alpha_i \leftarrow \text{dot\_prod}(r_{i-1}, z_i)$
  else
    $\alpha_i \leftarrow \text{dot\_prod}(r_{i-1}, z_i)$
    $\beta_i \leftarrow \alpha_i / \alpha_{i-1}$
    $p_i \leftarrow \beta_i p_{i-1} + z_i$
  end if
  $\alpha_i \leftarrow \text{dot\_prod}(r_{i-1}, z_i) / \text{dot\_prod}(p_i, Ap_i)$
  $x_{i+1} \leftarrow x_i + \alpha_i p_i$
  $r_i \leftarrow r_{i-1} - \alpha_i A p_i$
  if $\|r_i\|_2 <$ tolerance **then**
    STOP
  end if
end for

Multigrid and Gauss-Seidel

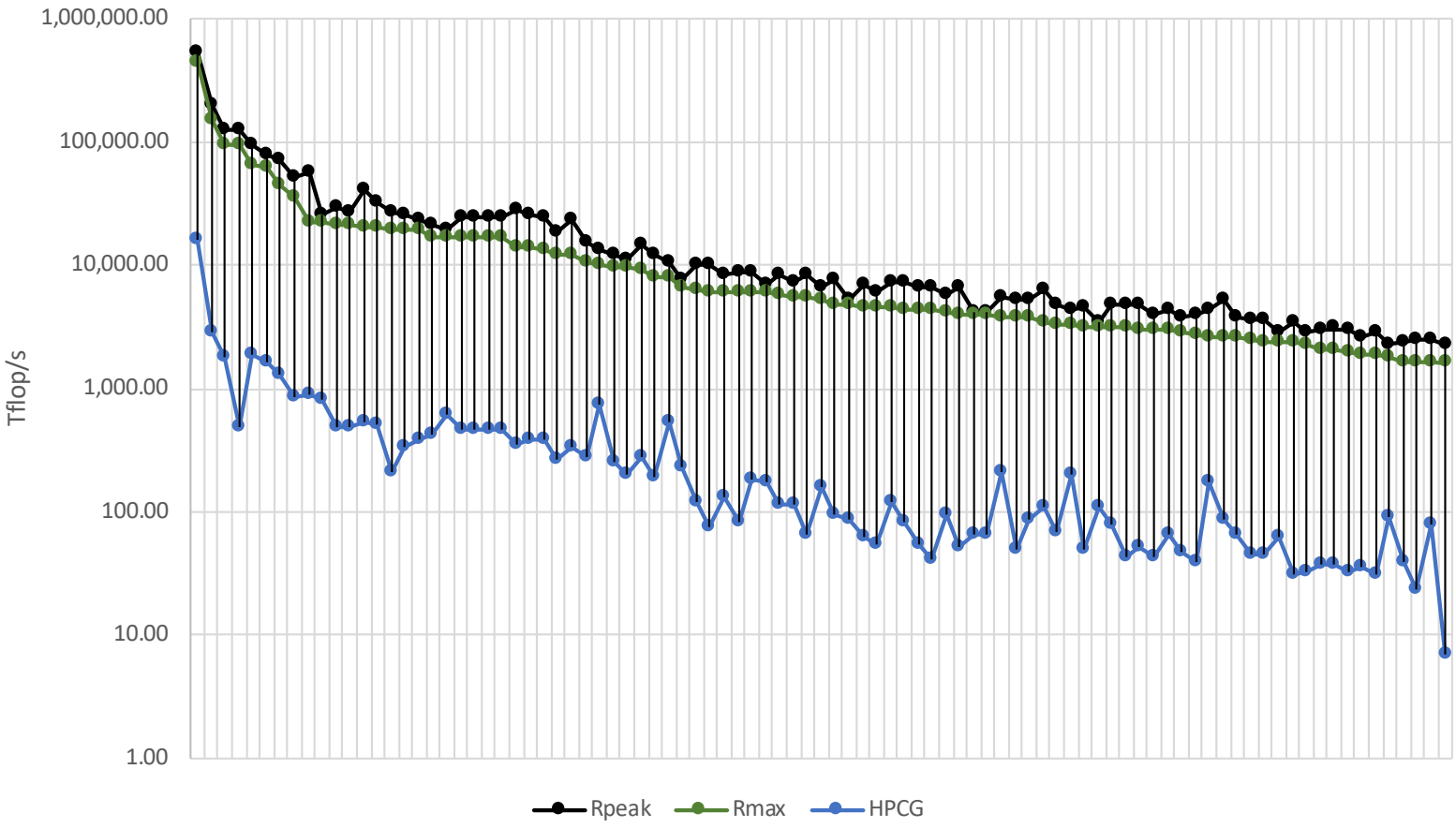
HPCG Top10, June 2021

Rank	Site	Computer	Cores	HPL Rmax (Pflop/s)	TOP500 Rank	HPCG (Pflop/s)	Fraction of Peak
1	RIKEN Center for Computational Science Japan	**Fugaku**, Fujitsu A64FX 48C 2.2GHz, Tofu D, Fujitsu	7,630,848	442.0	1	16.0	3.0%
2	DOE/SC/ORNL USA	**Summit**, AC922, IBM POWER9 22C 3.7GHz, Dual-rail Mellanox FDR, NVIDIA Volta V100, IBM	2,414,592	148.6	2	2.93	1.5%
3	DOE/SC/LBNL USA	**Perlmutter**, HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10	761,856	64.6	5	1.91	2.0%
4	DOE/NNSA/LLNL USA	**Sierra**, S922LC, IBM POWER9 20C 3.1 GHz, Mellanox EDR, NVIDIA Volta V100, IBM	1,572,480	94.6	3	1.80	1.4%
5	NVIDIA USA	**Selene**, DGX SuperPOD, AMD EPYC 7742 64C 2.25 GHz, Mellanox HDR, NVIDIA Ampere A100	555,520	63.5	6	1.62	2.0%
6	Forschungszentrum Juelich (FZJ) Germany	**JUWELS Booster Module**, Bull Sequana XH2000 , AMD EPYC 7402 24C 2.8GHz, Mellanox HDR InfiniBand, NVIDIA Ampere A100, Atos	449,280	44.1	8	1.28	1.8%
7	Saudi Aramco Saudi Arabia	**Dammam-7**, Cray CS-Storm, Xeon Gold 6248 20C 2.5GHz, InfiniBand HDR 100, NVIDIA Volta V100, HPE	672,520	22.4	11	0.88	1.6%
8	Eni S.p.A. Italy	**HPC5**, PowerEdge, C4140, Xeon Gold 6252 24C 2.1 GHz, Mellanox HDR, NVIDIA Volta V100, Dell	669,760	35.5	9	0.86	1.7%
9	Information Technology Center, The University of Tokyo, Japan	**Wisteria/BDEC-01 (Odyssey)**, PRIMEHPC FX1000, A64FX 48C 2.2GHz, Tofu D	368,640	22.1	13	0.82	3.2%
10	Japan Agency for Marine-Earth Science and Technology	**Earth Simulator -SX-Aurora TSUBASA**, A412-8, Vector Engine Type20B 8C 1.6GHz, Infiniband HDR200	43,776	0.01	41	0.75	5.6%

Comparison between Peak and HPL for June 2021



Comparison between Peak, HPL, and HPCG for June 2021



Modern Hardware: Lower Precision for Deep Learning

- Hardware (company)
  - GPU Tensor Cores (NVIDIA)
  - TPU MXU (Google)
  - Zion (Facebook)
  - DaVinci (Huawei)
  - Dot-product engine (HPE)
  - Eyeriss (Amazon)
  - Wafer Scale Engine (Cerebras)
  - Nervana (Intel)
  - Deep Learning Boost (Intel AI)
  - Graph Core
  - ...

60+

- Lower-precision benchmarks
  - Baidu
  - Dawn
  - mlperf
  - Deep500
  - ...
  - HPL-AI

WHY MIXED PRECISION? (Less is Faster)

- There are many reasons to consider mixed precision in our algorithms…
  - Less Communication
    - Reduce memory traffic
    - Reduce network traffic
  - Reduce memory footprint
  - More Flop per second
    - Reduced energy consumption
    - Reduced time to compute
  - Accelerated hardware in current architecture.
  - Suitable numerical properties for some algorithms & problems.

J. Langou, J. Langou, P. Luszczek, J. Kurzak, A. Buttari, and J. J. Dongarra. Exploiting the performance of 32 bit floating point arithmetic in obtaining 64 bit accuracy. In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, 2006.

Mixed Precision: Hardware Motivation

IBM Cell Broadband Engine	Apple ARM Cortex-A9	NVIDIA Kepler K10, K20, K40, K80	NVIDIA Volta/Turing	NVIDIA Volta/Turing
14x	7x	3x	2x	16x
32 bits / 64 bits	32 bits / 64 bits	32 bits / 64 bits	32 bits / 64 bits	16 bits / 64 bits

HPL-AI Benchmark Utilizing 16-bit Arithmetic

1. Generate random linear system Ax=b 
2. Represent the matrix A in low precision (16-bit floating point)
3. Factor A in lower precision into LU by Gaussian elimination 

4. Compute approximate solution with LU factors in low precision
5. Perform up to 50 iterations of refinement, e.g., GMRES to get accuracy up to 64-bit floating point
6. Use LU factors for preconditioning
7. Validate the answer is correct: scaled residual small $\frac{||Ax-b||}{||A||||x||+||b||} \times \frac{1}{n\epsilon} \leq O(10)$
8. Compute performance rate as $\frac{2}{3} \times \frac{n^3}{\text{time}}$

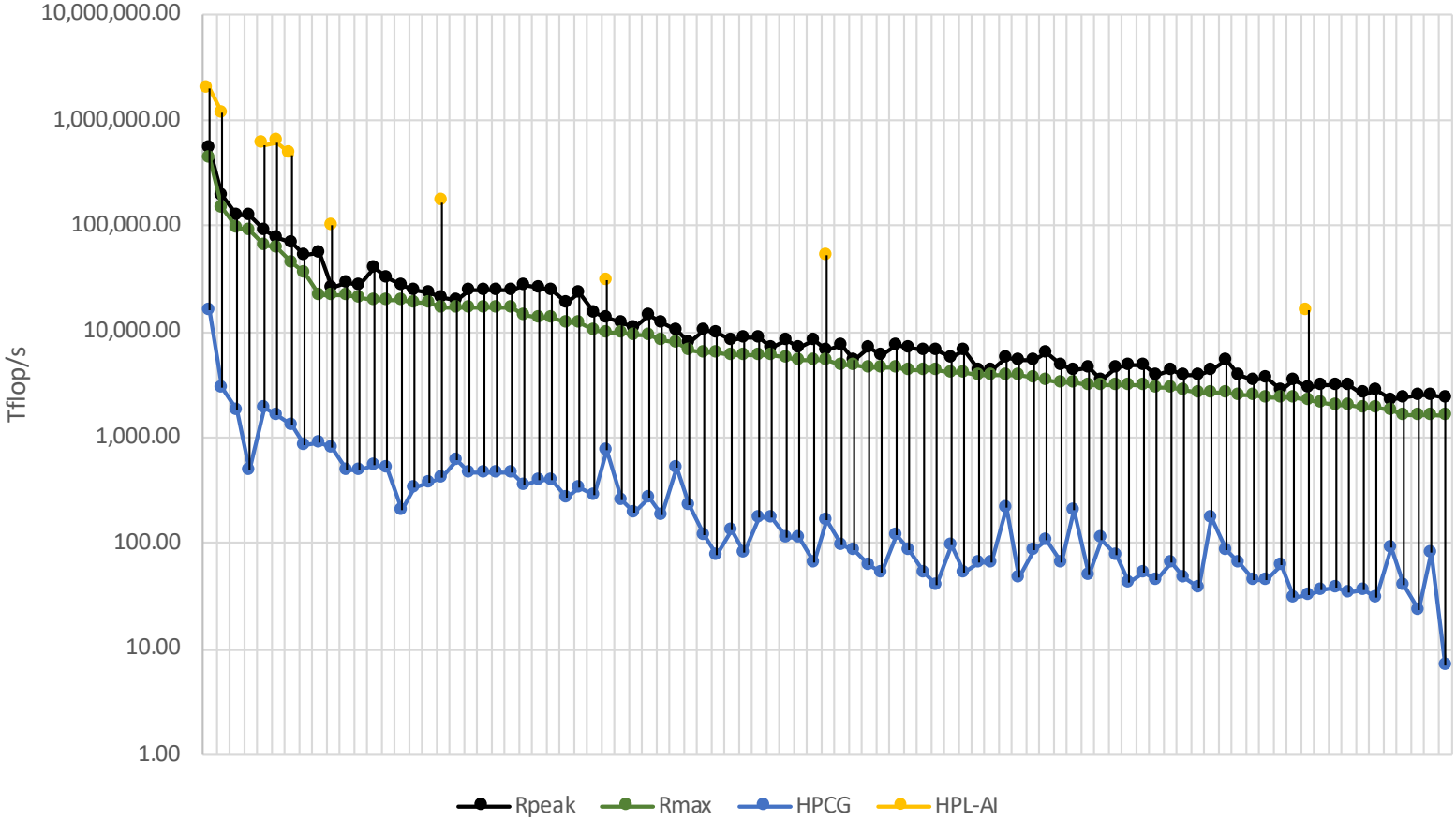
Iterative refinement for dense systems, *Ax = b*, can work this way.

L U = lu(A)	**lower precision**	**$O(n^3)$**
x = U\(L\b)	**lower precision**	$O(n^2)$
GMRes preconditioned by the LU to solve Ax=b	**FP64 precision**	$O(n^2)$

HPL-AI Top 10 for June 2021

Rank	Site	Computer	Cores	HPL Rmax (Eflop/s)	TOP500 Rank	HPL-AI (Eflop/s)	Speedup
1	RIKEN Center for Computational Science, Japan	**Fugaku**, Fujitsu A64FX, Tofu D	7,630,848	0.442	1	2.0	4.5
2	DOE/SC/ORNL USA	**Summit**, AC922 IBM POWER9, IB Dual-rail FDR, NVIDIA V100	2,414,592	0.149	2	1.15	7.7
3	NVIDIA USA	**Selene**, DGX SuperPOD, AMD EPYC 7742 64C 2.25 GHz, Mellanox HDR, NVIDIA A100	555,520	0.063	6	0.63	9.9
4	DOE/SC/LBNL/NERSC USA	**Perlmutter**, HPE Cray EX235n, AMD EPYC 7763 64C 2.45 GHz, Slingshot-10, NVIDIA A100	761,856	0.065	5	0.59	9.1
5	Forschungszentrum Juelich (FZJ) Germany	**JUWELS Booster Module**, Bull Sequana XH2000 , AMD EPYC 7402 24C 2.8GHz, Mellanox HDR InfiniBand, NVIDIA A100, Atos	449,280	0.044	8	0.47	10
6	University of Florida USA	**HiPerGator**, NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Infiniband HDR	138,880	0.017	23	0.17	9.9
7	Information Technology Center, The University of Tokyo, Japan	**Wisteria/BDEC-01 (Odyssey)**, PRIMEHPC FX1000, A64FX 48C 2.2GHz, Tofu D, Fujitsu	368,640	0.022	13	0.10	4.5
8	National Supercomputer Centre (NSC), Sweden	**Berzelius**, NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, A100, Infiniband HDR, Atos	59,520	0.005	84	0.05	9.9
9	Information Technology Center, Nagoya University, Japan	**Flow Type II subsystem**, PRIMERGY CX2570 M5, Xeon Gold 6230 20C 2.1GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR	79,560	0.0049	87	0.03	4.3
10	#CloudMTS Russia	**MTS GROM**, NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, A100 40GB, Infiniband	19,840	0.0023	245	0.015	7

Comparison between HPL-AI, Peak, HPL, and HPCG for June 2021

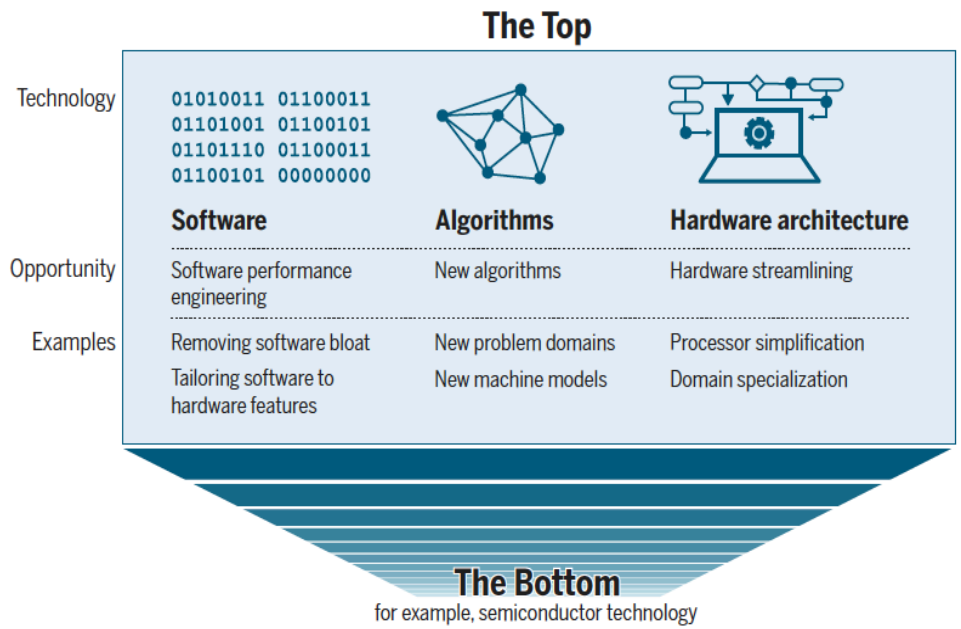


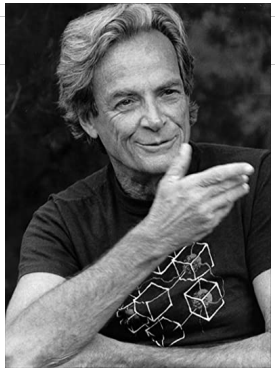
The Take Away

- HPC Hardware is Constantly Changing
  - Scalar
  - Vector
  - Distributed
  - Accelerated
  - Mixed precision
- Three computer revolutions
  - High performance computing
  - Deep learning
  - Edge & AI
- Algorithm / Software advances follows hardware.
  - And there is “plenty of room at the top”

“There’s plenty of room at the Top: What will drive computer performance after Moore’s law?”

Leiserson *et al.*, *Science* **368**, 1079 (2020) 5 June 2020





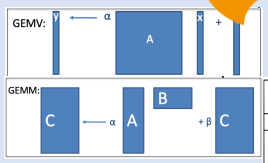
Feynman’s 1959 Lecture @ CalTech

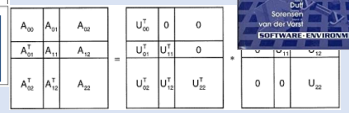
Jack's 50 Years

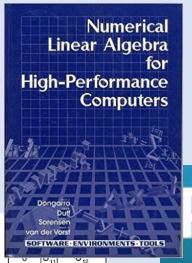

```
M = N – MOD(N,4)
DO 10 I = 1,M,4
   Y(I)   = Y(I)   + A * X(I)
   Y(I+1) = Y(I+1) + A * X(I+1)
   Y(I+2) = Y(I+2) + A * X(I+2)
   Y(I+3) = Y(I+3) + A * X(I+3)
10 CONTINUE
```

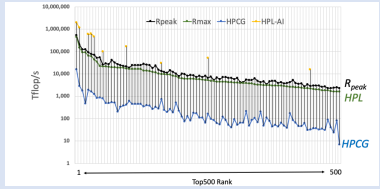
Generic matrix multiplication algorithm.

```
for ____ = 1 to ____
  for ____ = 1 to ____
    for ____ = 1 to ____
      c_ij = c_ij + a_ik * b_kj
    end
  end
end
```









1972 1976 1980 1984 1988 1992 1996 2000 2004 2008 2012 2016 2020