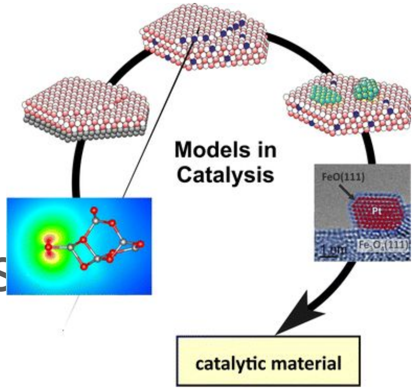
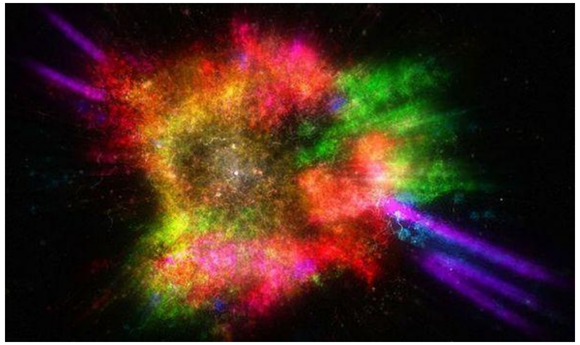
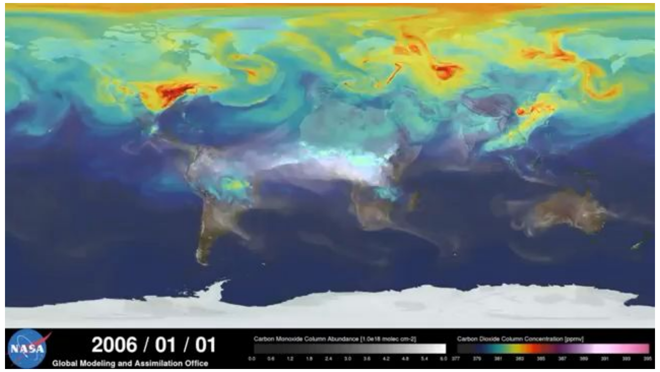
Today and Tomorrow HPC

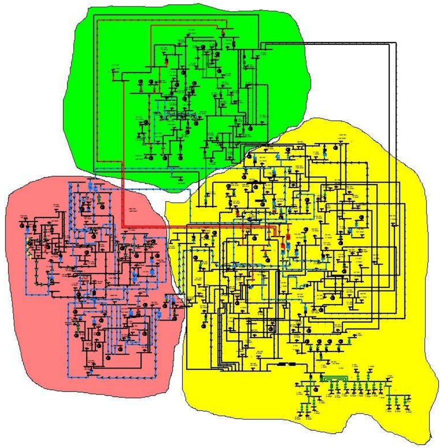
Today’s Top HPC Systems Used to do Simulations

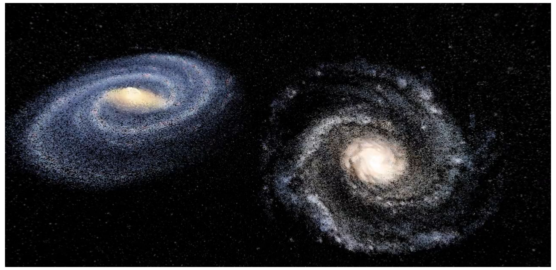
- *Climate*
- *Combustion*
- *Nuclear Reactors*
- *Catalysis*
- *Electric Grid*
- *Fusion*
- *Stockpile*
- *Supernovae*
- *Materials*
- *Digital Twins*
- *Accelerators*
- …

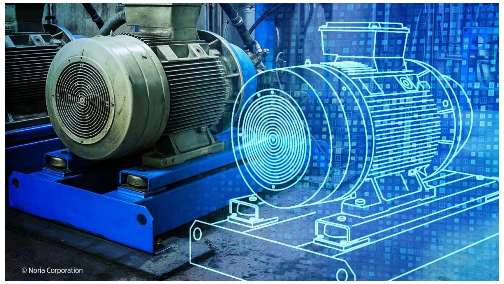


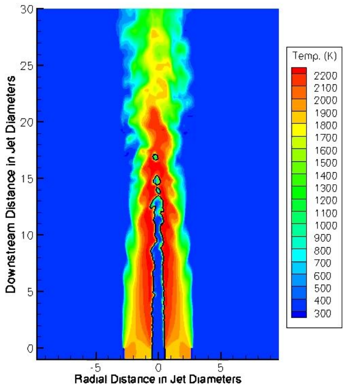


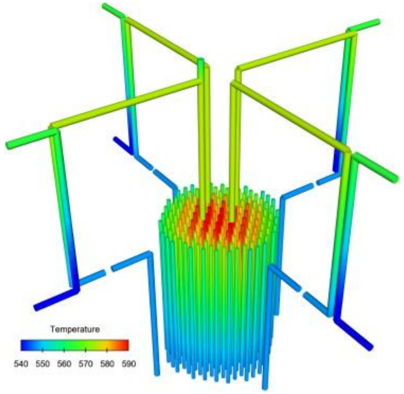


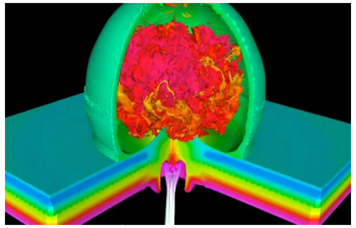


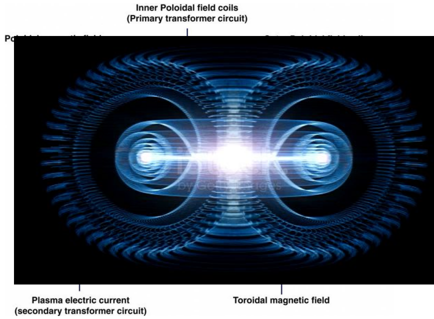






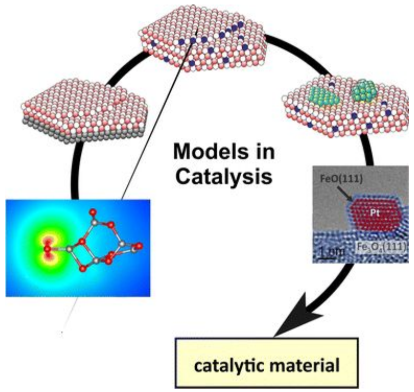


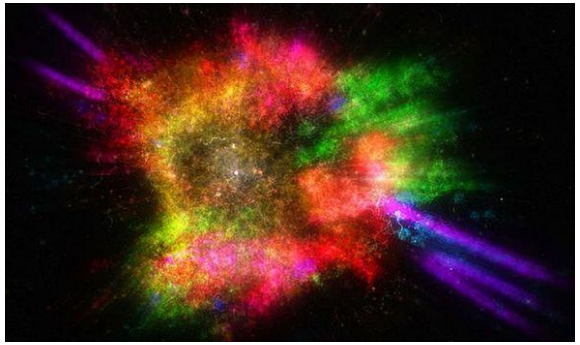


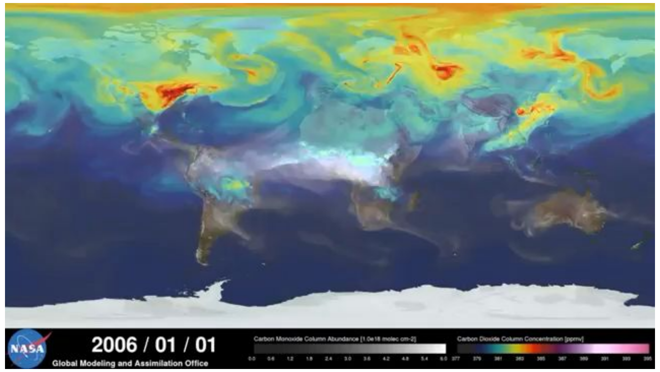


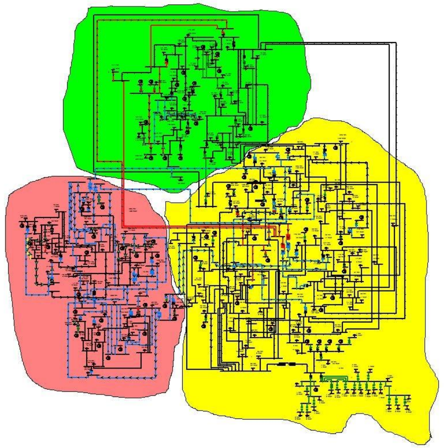
Today's Top HPC Systems Used to do Simulations

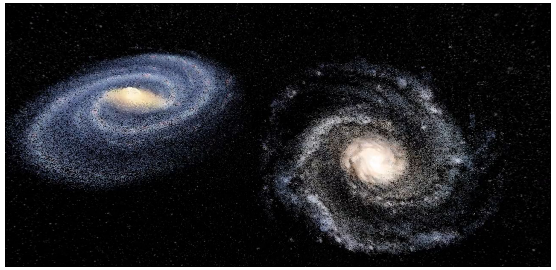
- *Climate*
- *Combustion*
- *Nuclear Reactors*
- *Catalysis*
- *Electric Grid*
- *Fusion*
- *Stockpile*
- *Supernovae*
- *Materials*
- *Digital Twins*
- *Accelerators*
- …

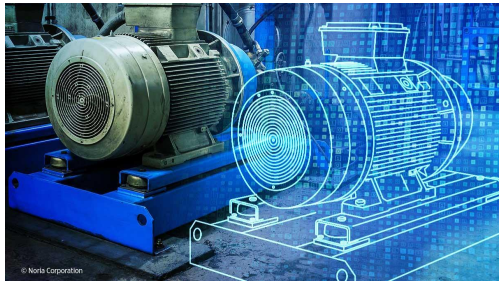


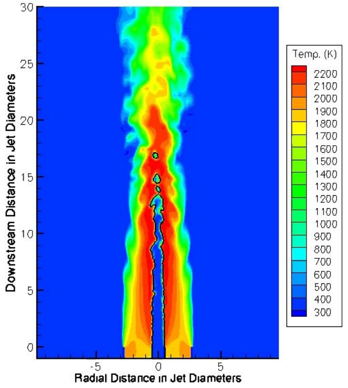


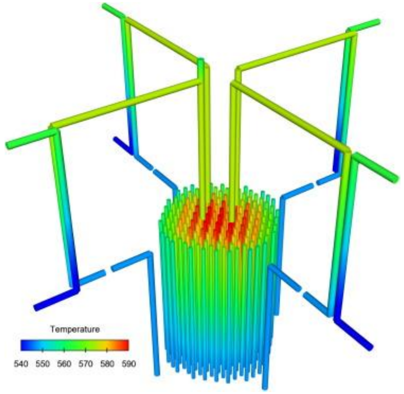


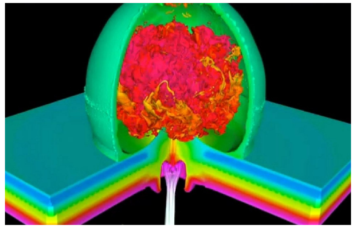


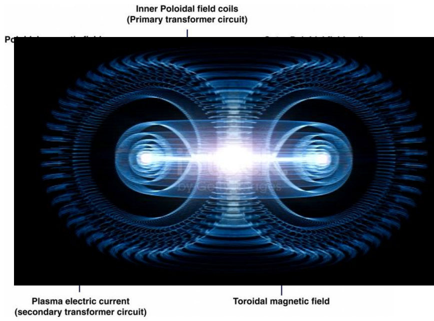












Usually 3-D PDE's

Sparse matrix computations, not dense

HPCG Top 10, November 2022

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	2	16.0	**3.0%**
2	DOE/SC/ORNL **USA**	**Frontier**, HPE Cray Ex235a, AMD 3rd EPYC 64C, 2 GHz, AMD Instinct MI250X, Slingshot 10	8,730,112	1,102	1	14.1	**0.8%**
3	EuroHPC/CSC [illegible]	**LUMI**, HPE Cray EX235a, AMD Zen-3 (Milan) 64C 2GHz, [illegible]	2,174,976	304	3	3.41	**0.[illegible]%**
[illegible]	[illegible] **USA**	[illegible] Mellanox EDR, NVIDIA Volta V100, IBM	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
5	EuroHPC/CINECA **Italy**	Leonardo, BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 40 GB, Quad-rail NVIDIA HDR100 Infiniband	1,463,616	175	4	2.57	**1.0%**
6	DOE/SC/LBNL **USA**	**Perlmutter**, HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10	761,856	70.9	8	1.91	**2.0%**
7	DOE/NNSA/LLNL **USA**	**Sierra**, S922LC, IBM POWER9 20C 3.1 GHz, Mellanox EDR, NVIDIA Volta V100, IBM	1,572,480	94.6	6	1.80	**1.4%**
8	NVIDIA **USA**	**Selene**, DGX SuperPOD, AMD EPYC 7742 64C 2.25 GHz, Mellanox HDR, NVIDIA Ampere A100	555,520	63.5	9	1.62	**2.0%**
9	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	12	1.28	**1.8%**
10	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	20	0.88	**1.6%**

Think of a race car that has the potential of 200 MPH but only goes 2 MPH!

AI/ML Takeoff

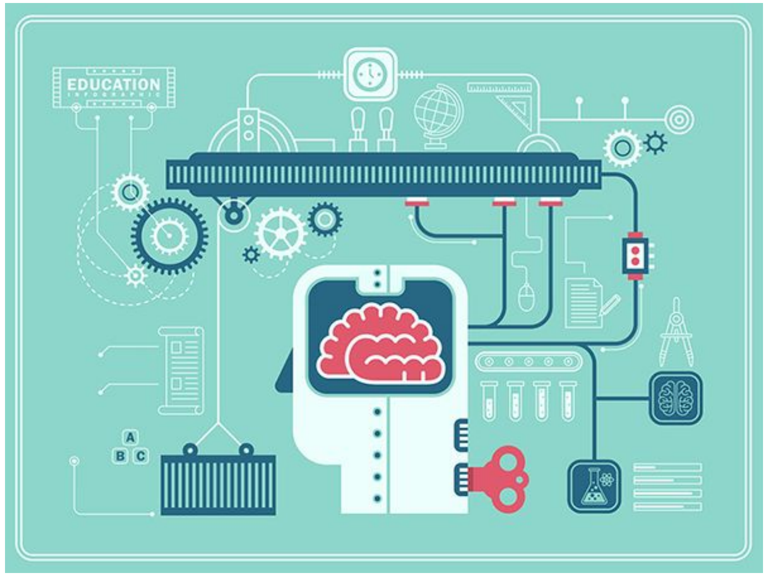
Recently we have seen AI & ML take off

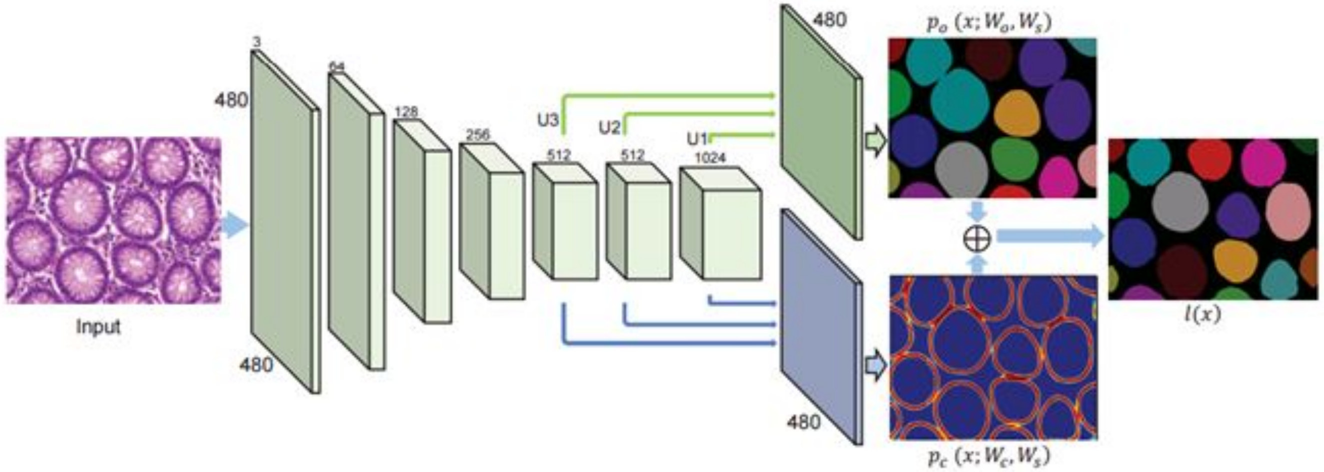
- AI and ML have been around for a long time as research efforts.
- Why Now?
    - Flood of available data (especially with the Internet)
    - Increasing computational power
    - Growing progress in available algorithms and theory developed by researchers.
    - Increasing support from industries.

Machine Learning in Computational Science

Many fields are beginning to adopt machine learning to augment modeling and simulation methods

- Climate
- Biology
- Drug Design
- Epidemology
- Materials
- Cosmology
- High-Energy Physics



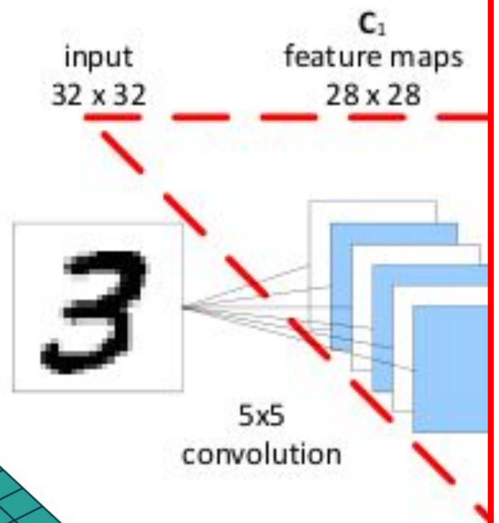


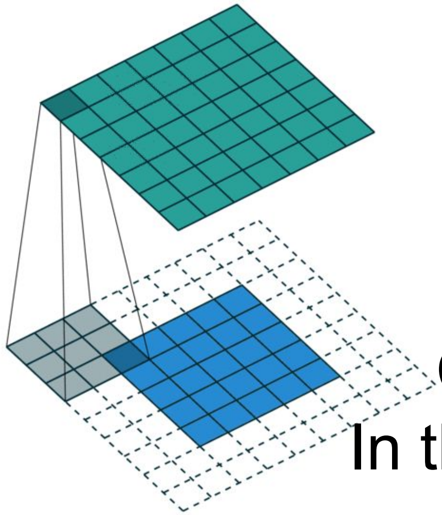
Deep Learning Needs Small Matrix Operations

Matrix Multiply is the time-consuming part.

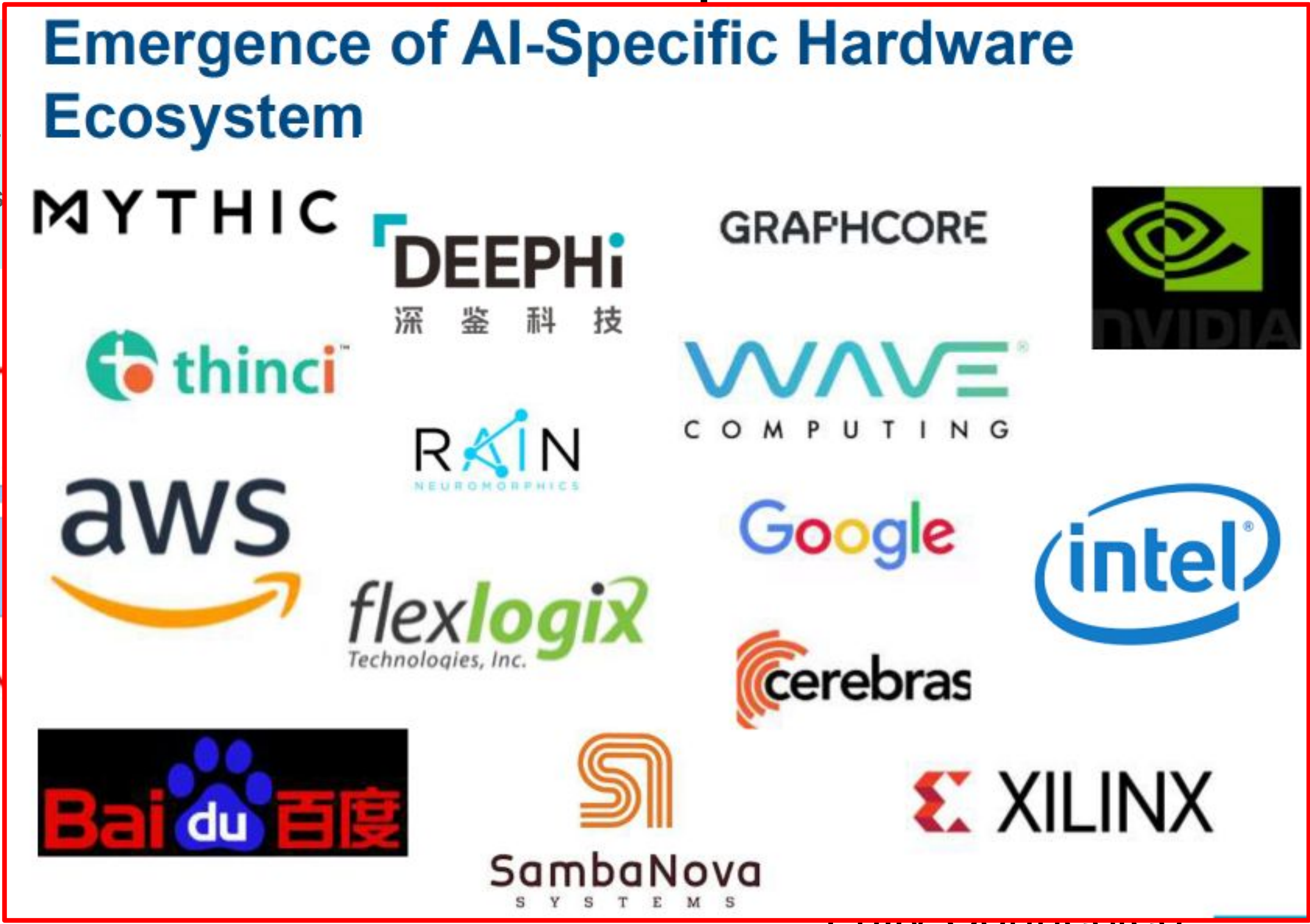
Convolution Layers and Fully Connected L

There are many GEMM's of small matrices
point





Convolution
In this case 3x3 GEMM

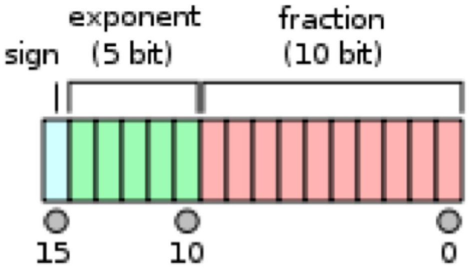


Fully Connected
Classification

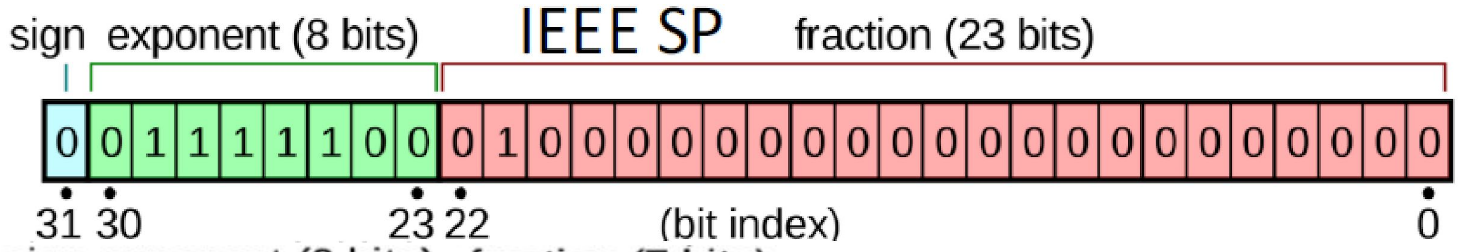
Numerical representations

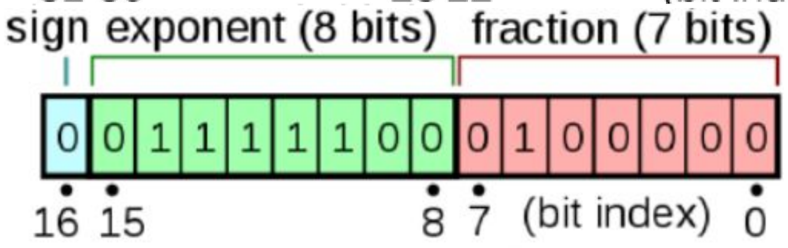
- Today many options for arithmetic precision (IEEE Standard)

Type	Size	Range	$u = 2^{-t}$
half	16 bits	$10^{\pm 5}$	$2^{-11} \approx 4.9 \times 10^{-4}$
single	32 bits	$10^{\pm 38}$	$2^{-24} \approx 6.0 \times 10^{-8}$
double	64 bits	$10^{\pm 308}$	$2^{-53} \approx 1.1 \times 10^{-16}$
quadruple	128 bits	$10^{\pm 4932}$	$2^{-113} \approx 9.6 \times 10^{-35}$



IEEE FP16

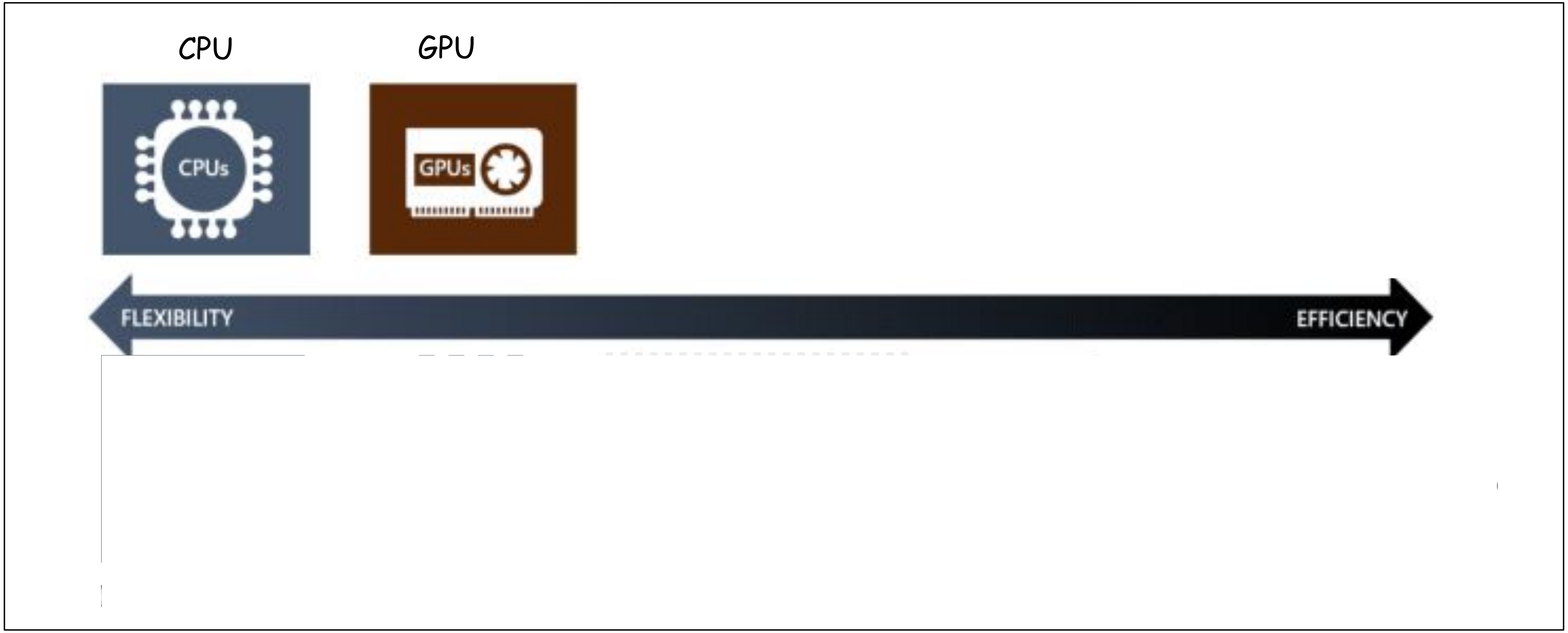




Google Bfloat16

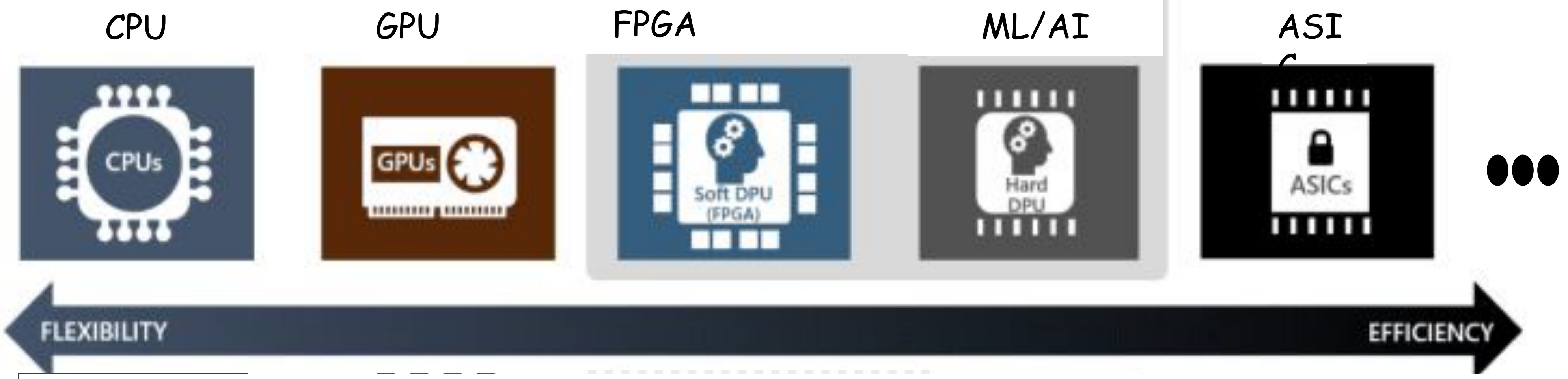
Future HPC Systems Will be Customized…

- You will be able to dial up what you need in your computer for your application mix …



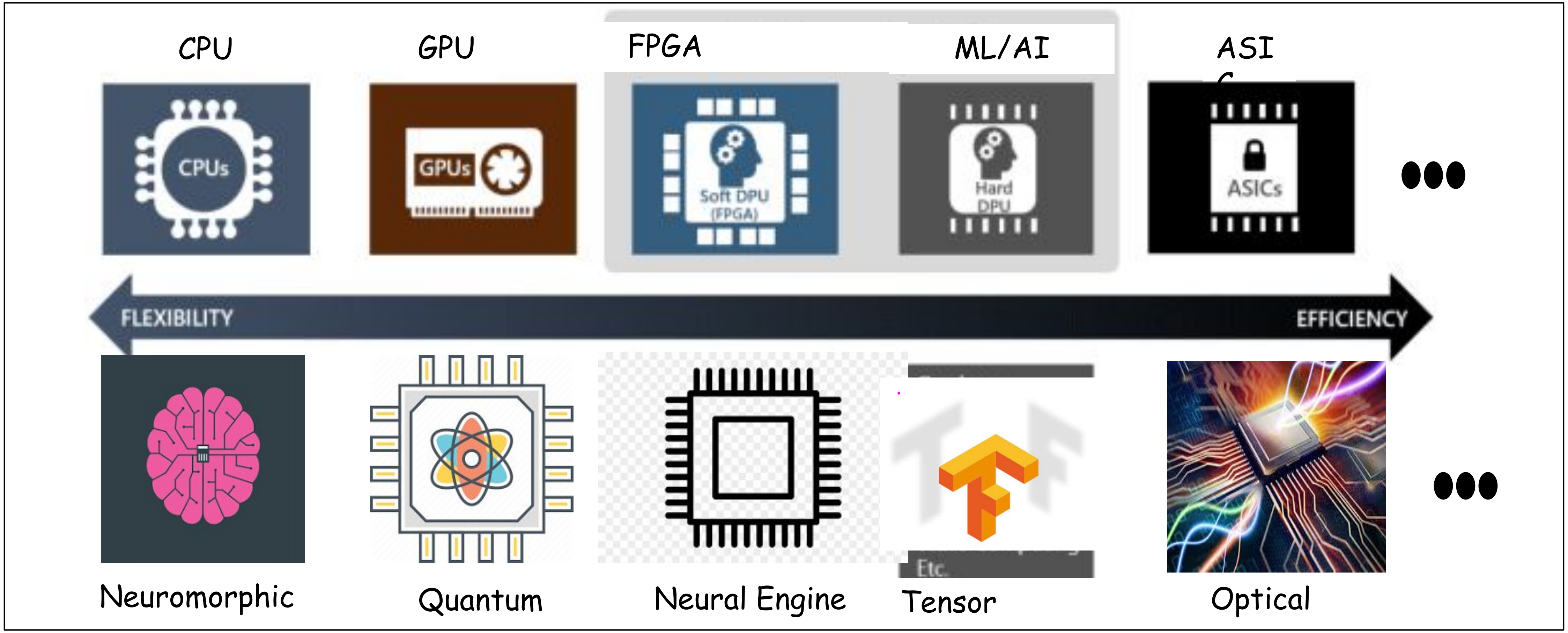
Future HPC Systems Will be Customized…

- You will be able to dial up what you need in your computer for your application mix …



Future HPC Systems Will be Customized…

- You will be able to dial up what you need in your computer for your application mix …



High performance Programming

What do you mean by performance

- What is a **xflop/s**?
  xflop/s is a rate of execution, some number of floating-point operations per second. Whenever this term is used, it will refer to **64-bit floating-point operations** and the operations will be either addition or multiplication.
- What is the **theoretical peak performance**?

  The theoretical peak is a paper computation to determine the theoretical peak rate of execution of floating-point operations for the machine. The theoretical peak performance is determined by *counting the number of floating-point additions and multiplications (in full precision) that can be completed during a period of time.*
- For example, an Intel Skylake processor at **2.1 GHz** can complete **32** floating point operations **per cycle per core** or a theoretical peak performance of **67.2 GFlop/s** per core or **1.61 Tflop/s** for the socket of 24 cores.

Example: DGEMM Performance

Speedups from performance engineering a program that multiplies two 4K-by-4K floating point matrices (**D**ouble-precision **GE**neral **M**atrix-**M**atrix) running on a dual-socket Intel Xeon E5-2666 v3 system

Version	Implementation	Absolute speedup	Relative speedup
1	Python	1	1
2	Java	11	10.8
3	C	47	4.4
4	Parallel loop	366	7.8
5	Divide and conquer	6’727	18.4
6	Vectorization	23’224	3.5
7	AVX intrinsics	62’806	2.7

Software does not have “good enough” performance by default.

[There’s plenty of room at the Top: What will drive computer performance after Moore’s law? https://dx.doi.org/10.1126/science.aam9744]

Single-Node HPC

- How to improve performance of an application running on a single node?

- We need to:
  1. Measure actual application performance
  2. Understand the hardware architecture executing the application
  3. Identify bottleneck: (memory bound/compute bound/resource underutilization)
  4. (possibly) solve bottlenecks
  5. Goto 1

Single-Node HPC

- How to write high-performance applications running on a single node?

1. Optimize Algorithms & Data Structures
   - Use efficient algorithms
   - Choose appropriate data structures (e.g., hash tables for fast lookups, trees for sorted data).
2. Memory Optimization
   - Optimize CPU cache usage (**data locality, avoid cache misses**).
   - Maximize data reuse
3. Exploit hardware parallelism
   - Exploit hardware accelerated instructions (e.g x86 AVX extensions)
   - Utilize multi-core CPUs with threading

Optimize Algorithms

An example with sorting:

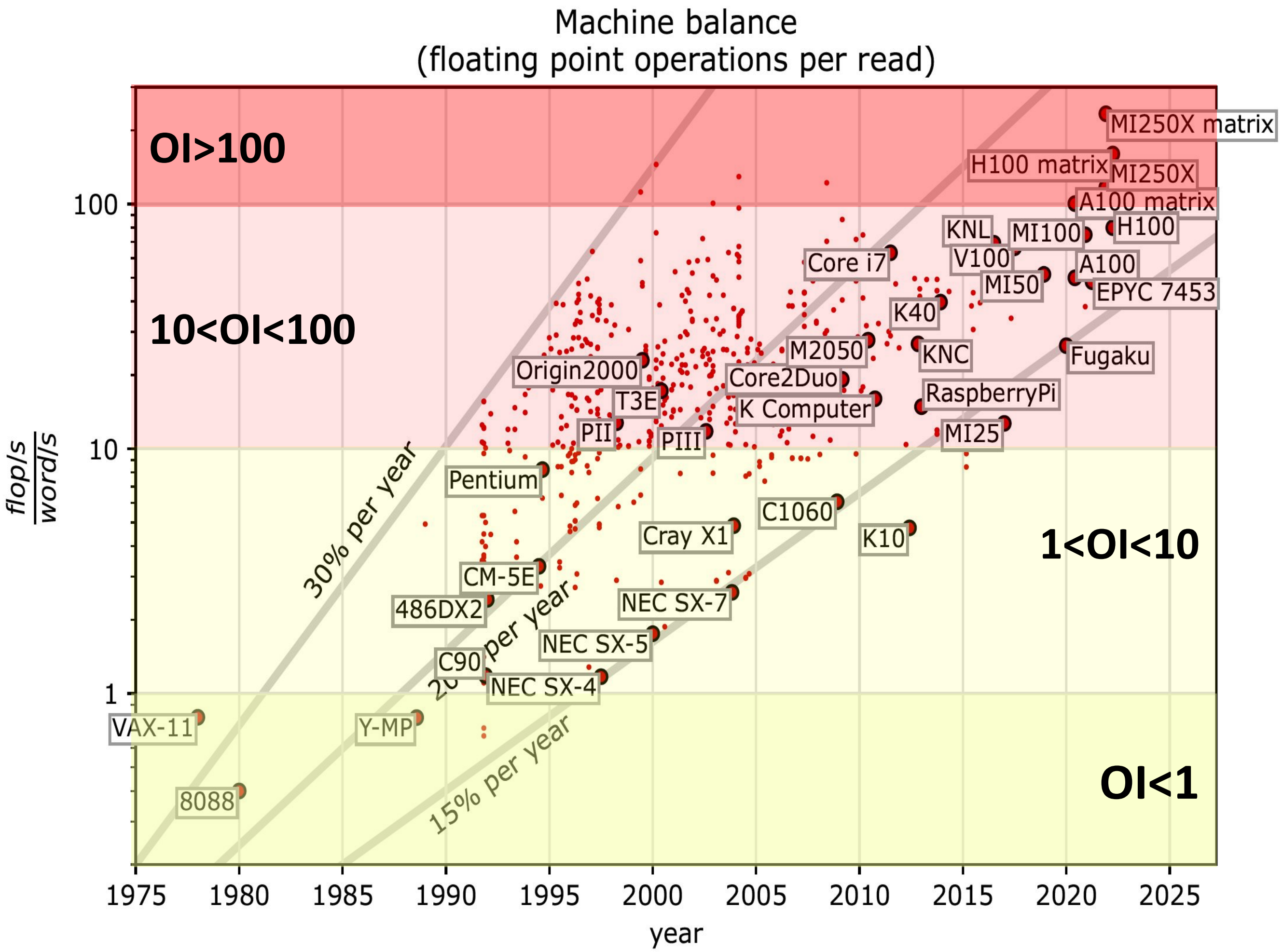
Version	1000 elements	10000 elements	100000 elements
quicksort	67 us	458 us	3993 us
counting sort	36 us	57 us	837 us

Why is faster:

complexity is linear with the max value. Be aware, it works only with integers with limited range

Optimize Memory

- Data movement has a big impact
- **Operational Intensity:** Performance comes from balancing floating point execution (Flops/sec) with memory->CPU transfer rate (Words/sec)
  - "Best" balance would be 1 flop per word-transfered
- Today's systems are close to 100 flops/sec per word-transferred
  - Imbalanced: Over provisioned for Flops



Graph from Mark Gates

Plot for 64-bit floating point data movement & operations
(Bandwidth from CPU or GPU memory to registers)

Optimize Memory

An example with DGEMM:

Version	Implementation	Absolute speedup	Relative speedup
4	Parallel loop	366	7.8
5	Divide and conquer	6’727	18.4

Why is faster:

- Instead of accessing entire rows or columns, subdivide matrices into blocks.
- Requires more memory accesses but improves locality of accesses

Exploit hardware parallelism

- **Types of Parallelism:**
  - Data parallelism: Same task on different data.
  - Thread parallelism: different threads cooperate to execute an algorihtm.

- **Programming Models:**
  - Data parallelism is usually exploited with specialized instructions
  - Thread parallelism is usually exploited with OpenMP for (single) HPC node

Data Parallelism

An example with DGEMM:

Version	Implementation	Absolute speedup	Relative speedup
5	Divide and conquer	6’727	18.4
6	Vectorization	23’224	3.5

Why is faster:

Exploits SIMD (Single Instruction multiple Data) CPU extension. The same operation is applied to multiple data at the same time

Thread Parallelism

An example with DGEMM:

Version	Implementation	Absolute speedup	Relative speedup
3	C	47	4.4
4	Parallel loop	366	7.8

Why is faster:

Execute the multiplication on multiple cores.

Warning: you need to be aware of data races and false sharing