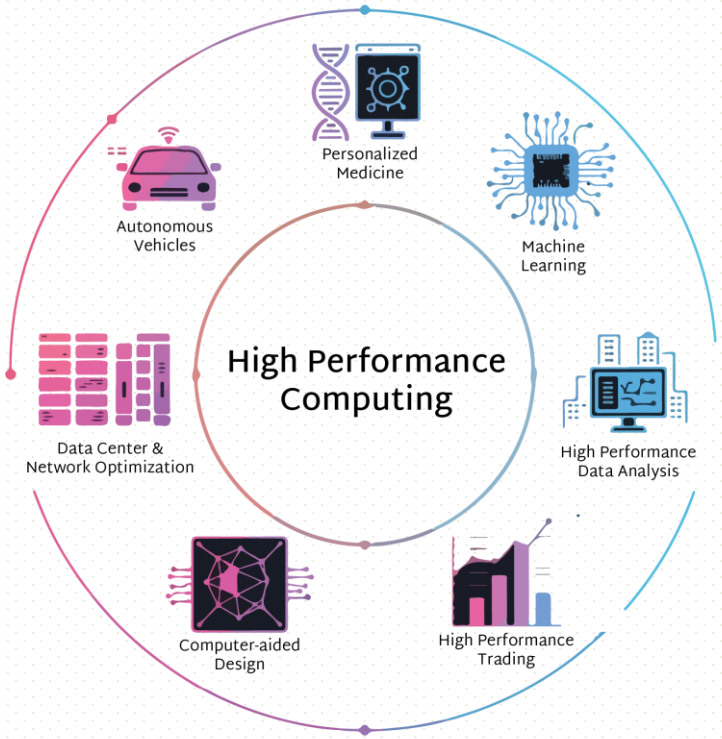
AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

High-Performance Computing in the Exascale Era

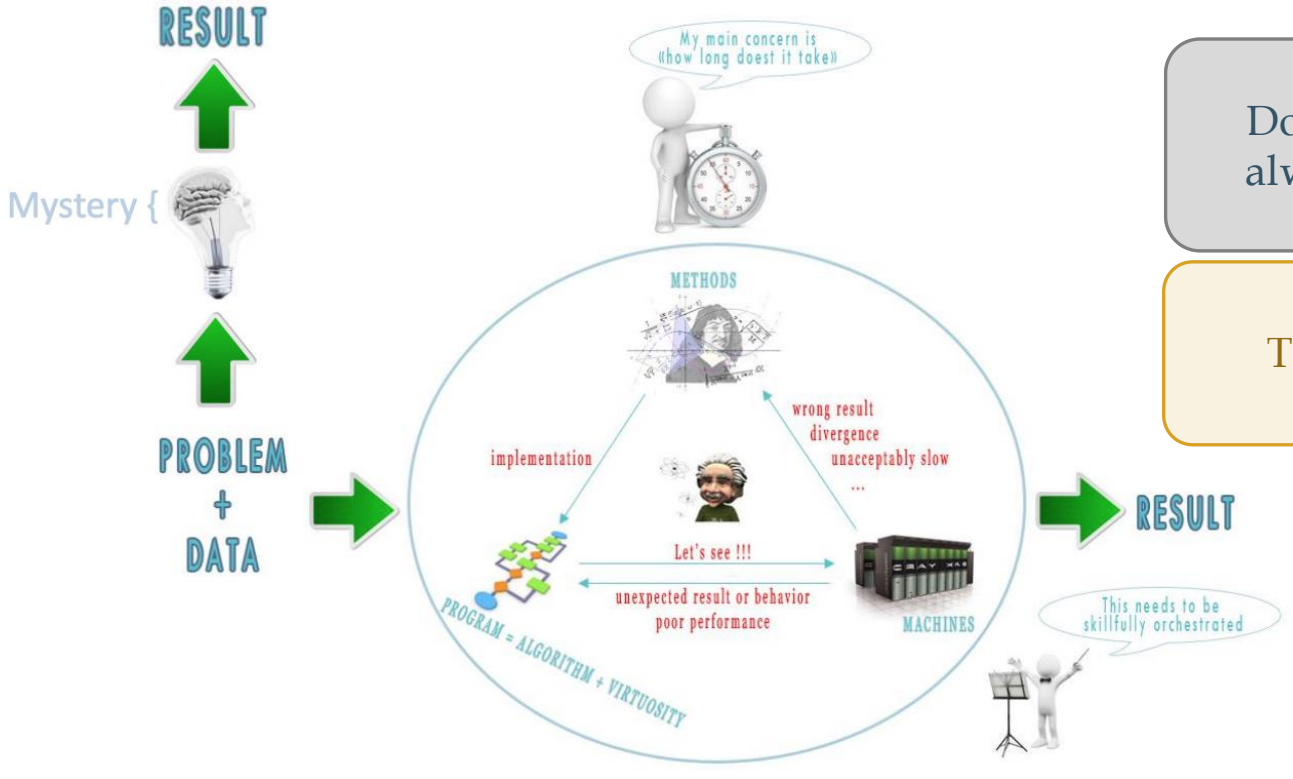


Claude TADONKI
Mines Paris – PSL (Paris/France)

MINES PARIS
UNIVERSITÉ PARIS SCIENCES & LETTRES

AMRITA University (India) - August 2026

High-Performance Computing in the Exascale Era
HPC as a Combination of Method-Programming-Machines



Do you think that the **best algorithm** always leads to the **fastest program** ?

NO.
Two examples: **Sorting** and **Linear Programming**

Complex problems require HPC

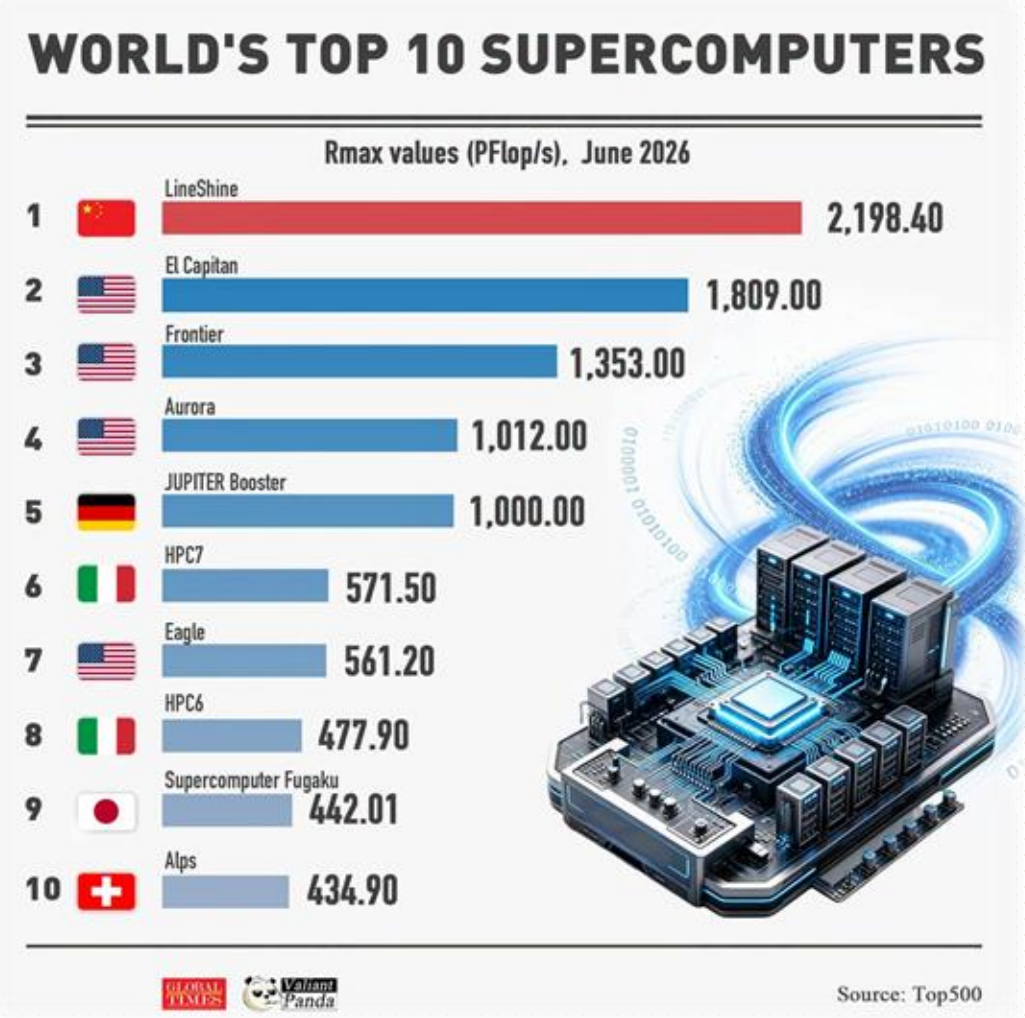
- **Significant progress** has been made in each of the main HPC components
- A **judicious combination** of all HPC ingredients is essential for optimal performance
- The necessary **multidisciplinary inte**raction must be considered at the earliest

High-Performance Computing in the Exascale Era

Computing Power – Overview & Evolution

Frontier is the **first Exaflop** machine
Released as such in 2022 (**USA**)

Jack Dongarra

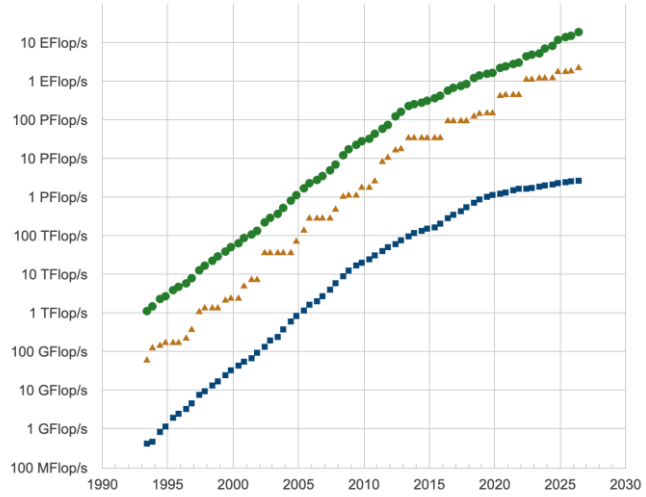


Facts

- Measurement is in **double precision** arithmetic
- Five **Exaflop** machines (China[1]/USA[3]/Germany[1])
- Height of the top10 are **CPU-GPU**
- **LineShine** and **Fugaku** are CPU-only

Top Supercomputers on HPL-MxP (Mixed-Precision – LLM)

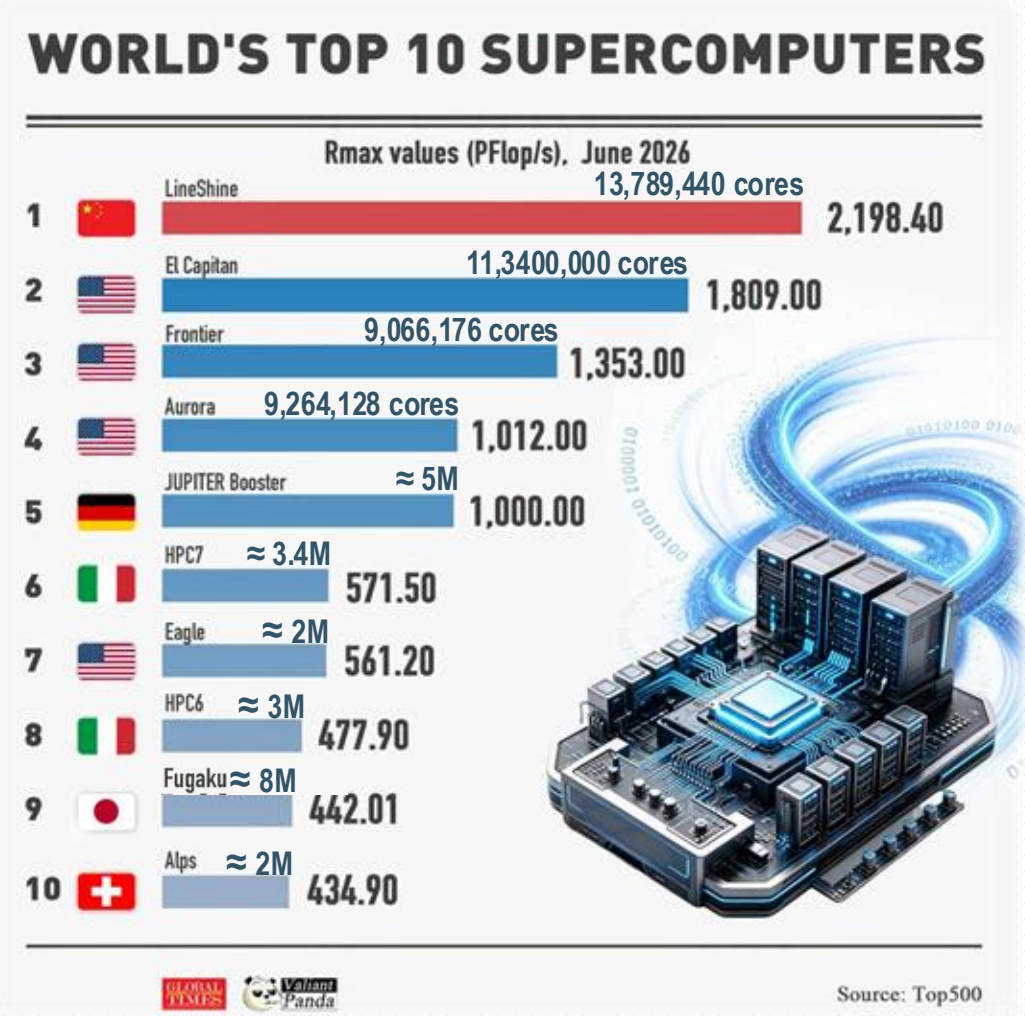
- **1. El Capitan (DOE/SC/LLNL, USA):** 16.680 Eflop/s
- **2. Aurora (DOE/SC/ANL, USA):** 11.643 Eflop/s
- **3. Frontier (DOE/SC/ORNL, USA):** 11.390 Eflop/s
- **4. JUPITER Booster (EuroHPC/FZJ, Germany):** 6.255 Eflop/s
- **5. CHIE-4 (SoftBank, Japan):** 3.304 Eflop/s
- **6. ABCI 3.0 (AIST, Japan):** 2.363 Eflop/s
- **7. LUMI (EuroHPC/CSC, Finland):** 2.350 Eflop/s
- **8. Fugaku (RIKEN, Japan):** 2.000 Eflop/s
- **9. Leonardo (EuroHPC/CINECA, Italy):** 1.842 Eflop/s
- **10. Shaheen III - GPU (KAUST, Saudi Arabia):** 0.704 Eflop/s



iPhone 17 Pro: 2.5 TFLOPS single-precision ($1,200) **CRAY T3E**: 1.3 FLOPS ($5 million)

High-Performance Computing in the Exascale Era

Computing Power – Potential vs Reality



N°	Peak (PFlops/s)	Efficiency(%)
1	2,735.82	80
2	2,821.10	64
3	2,055.72	66
4	1,980.01	51
5	1,226.28	82
6	861.13	66
7	846.84	66
8	606.97	79
9	537.21	82
10	574.84	76

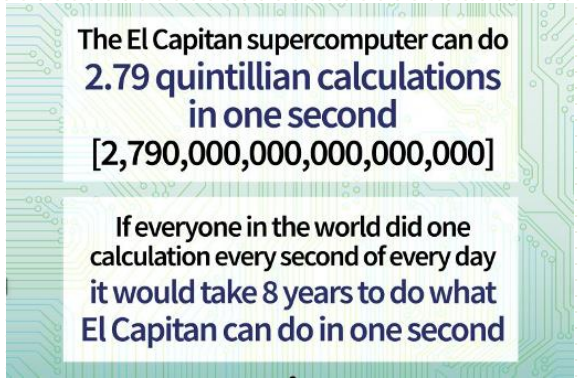
- **Peak** is the theoretical or potential computing power
- **Rmax** is (supposed to be) the best from measurement
- **Efficiency** is the fraction of the peak that could be obtained
- **Several technical** factors are involved when it comes to **computational efficiency**

Obtaining the **highest fraction of the peak performance** of a **high-speed machine** is the typical technical **challenge** of an **HPC expert**.

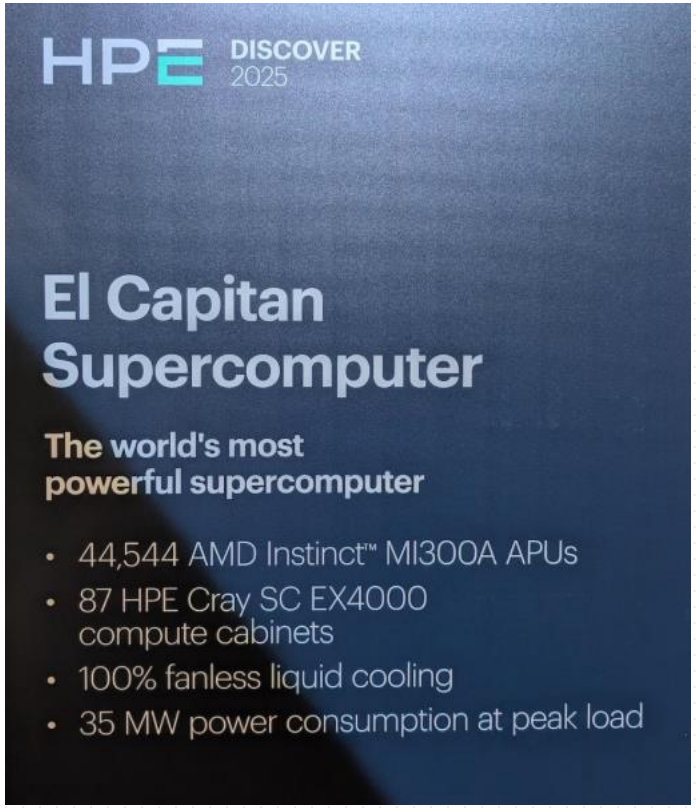
High-Performance Computing in the Exascale Era

Computing Power – 2e World Fastest Supercomputer

El Capitan



Number of cores: 11,340,000
Configuration: CPU - GPU
Peak (PFlops/s): 2,821.10
Rmax (PFlops/s): 1,809.00
Power(kW): 29,685
Annual electricity bill: $14-30 million



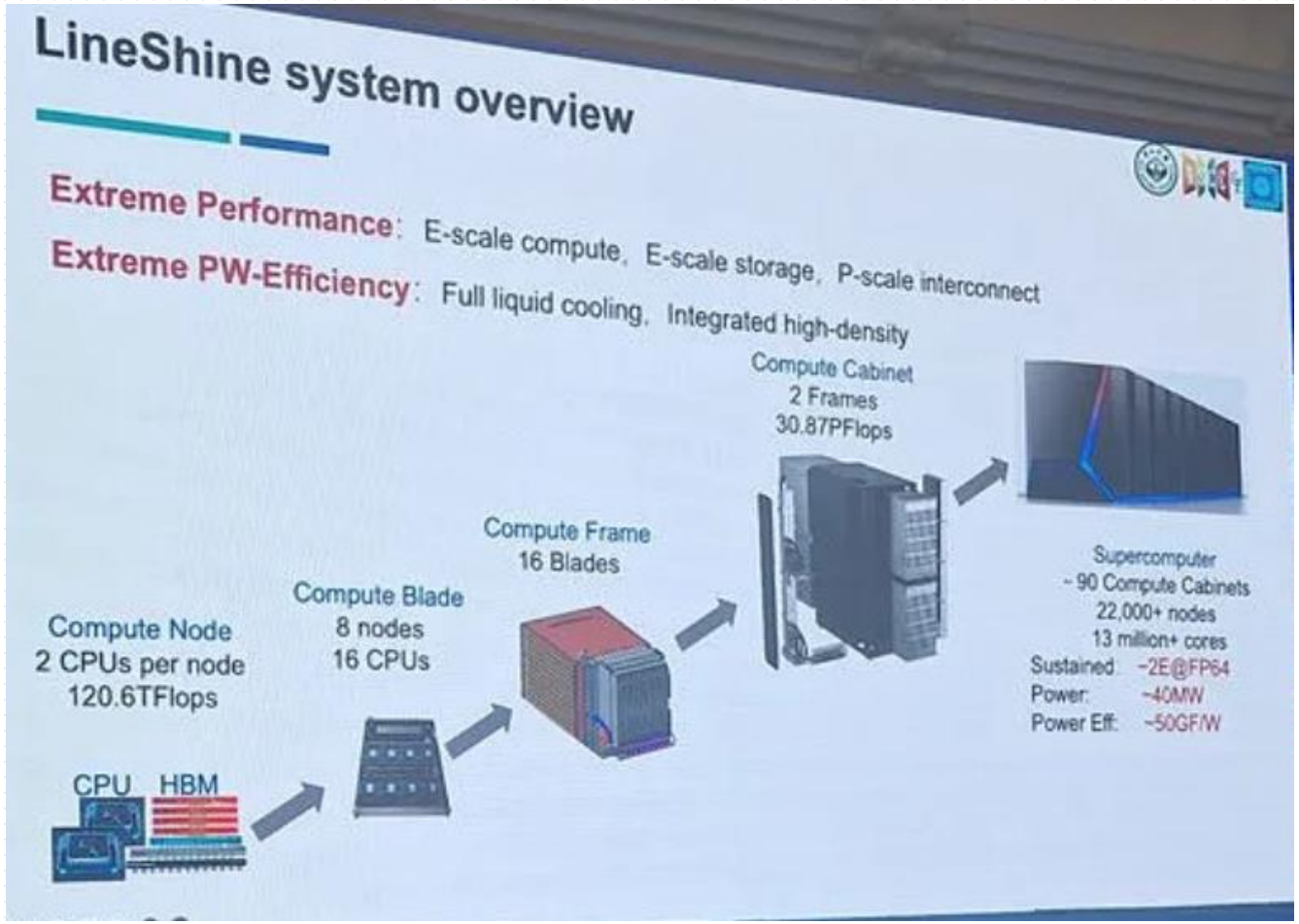
4 MI300A / Node – 11,136 nodes – 87 Cabinets

High-Performance Computing in the Exascale Era

Computing Power – World Fastest Supercomputer

LineShine

Number of cores: 13,789,440
Configuration: CPU only
Peak (PFlops/s): 2,735.82
Rmax (PFlops/s): 2,198.40
Power(kW): 42,220
Annual electricity bill: $20-40 million



High-Performance Computing in the Exascale Era

Computing Power – Understanding Performance Data

The *theoretical peak performance* in (full precision) *FLOPS* is obtained by the following formula:

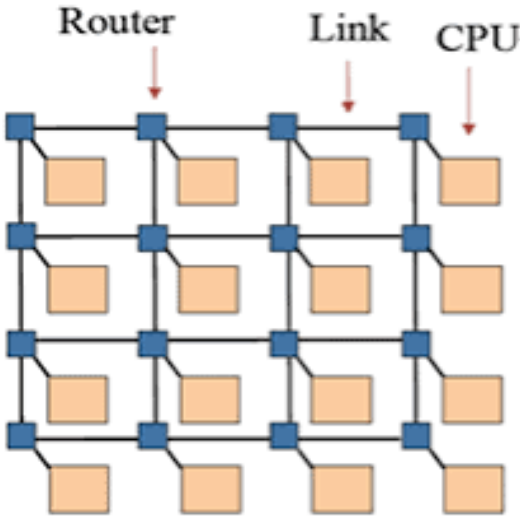
$$P = n_c \times [(2f) \times s_c \times F]$$

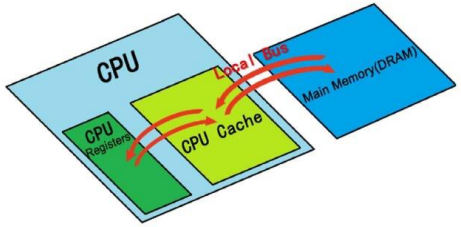
n_c the *total number of cores*

f number of *FMA units* (FMA is $a \times b + c$)

s_c is the *SIMD width* (#components of a vector)

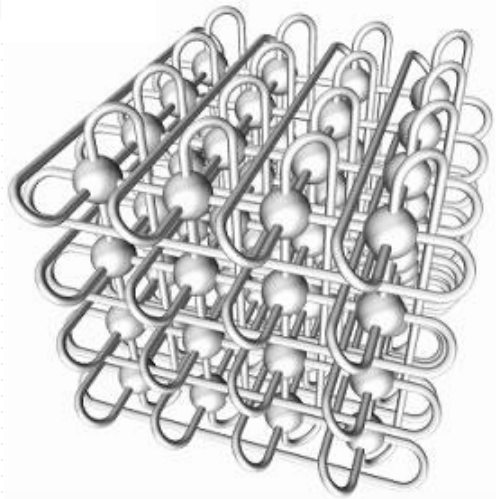
F *clock frequency* of the CPU (1 Hz = 1 flop)





Facts

- A computing cluster is made up with **several compute nodes**
- Compute nodes exchange data through a **physical network** (interconnects)
- A typical computation on a CPU-core involves **memory read/write**
- **Peak performance** does not consider **any non-computing mechanism**



High-Performance Computing in the Exascale Era

Computing Power – Questions & Discussions

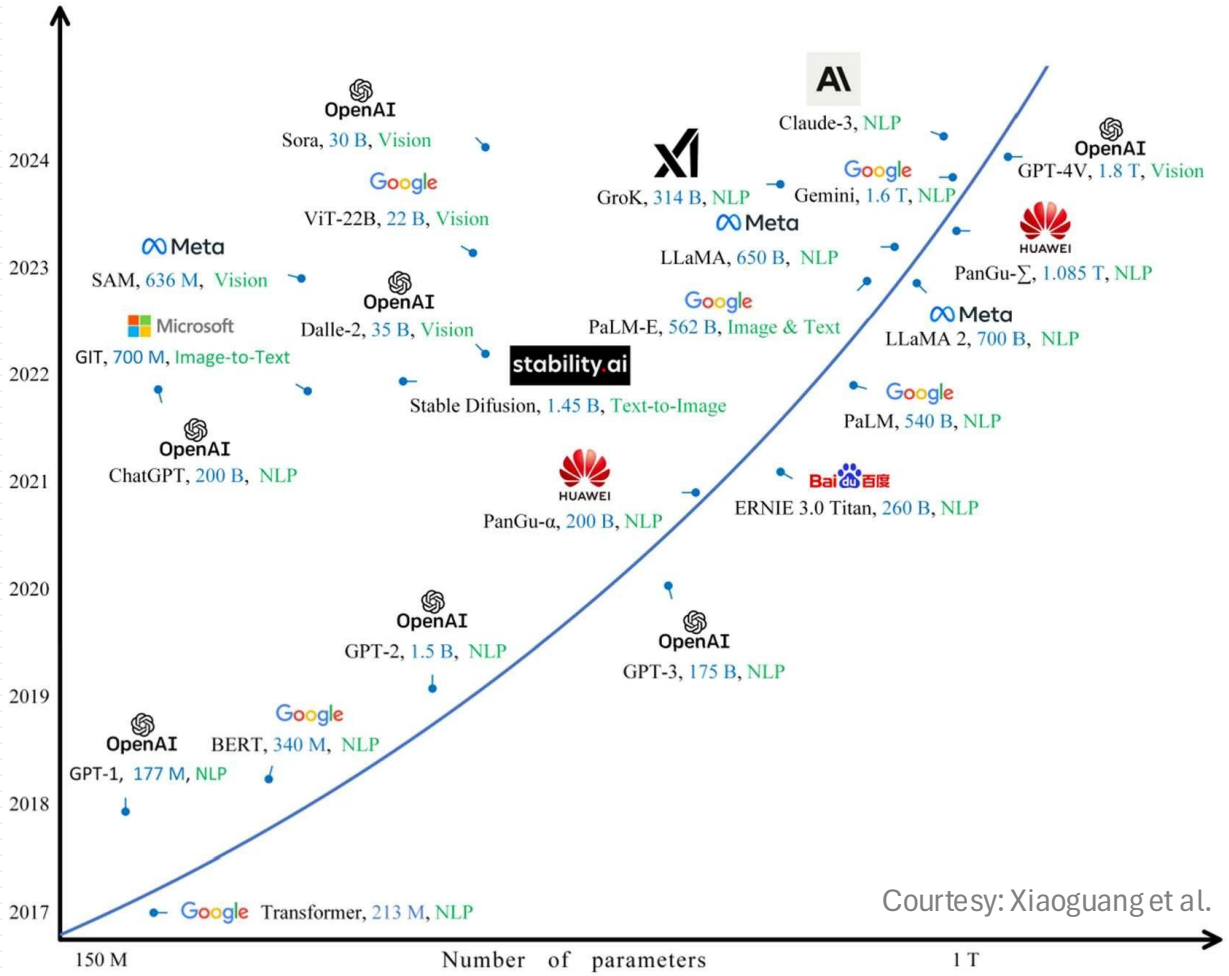
Do you think about a supercomputer with a weak efficiency (like Aurora with 50%)?

Aurora is 2^e on HPL-MxP (mixed-precision), so the efficiency depends on the application and the execution model.

Is the reported efficiency for each machine necessarily the best (note that the full machine is considered) ?

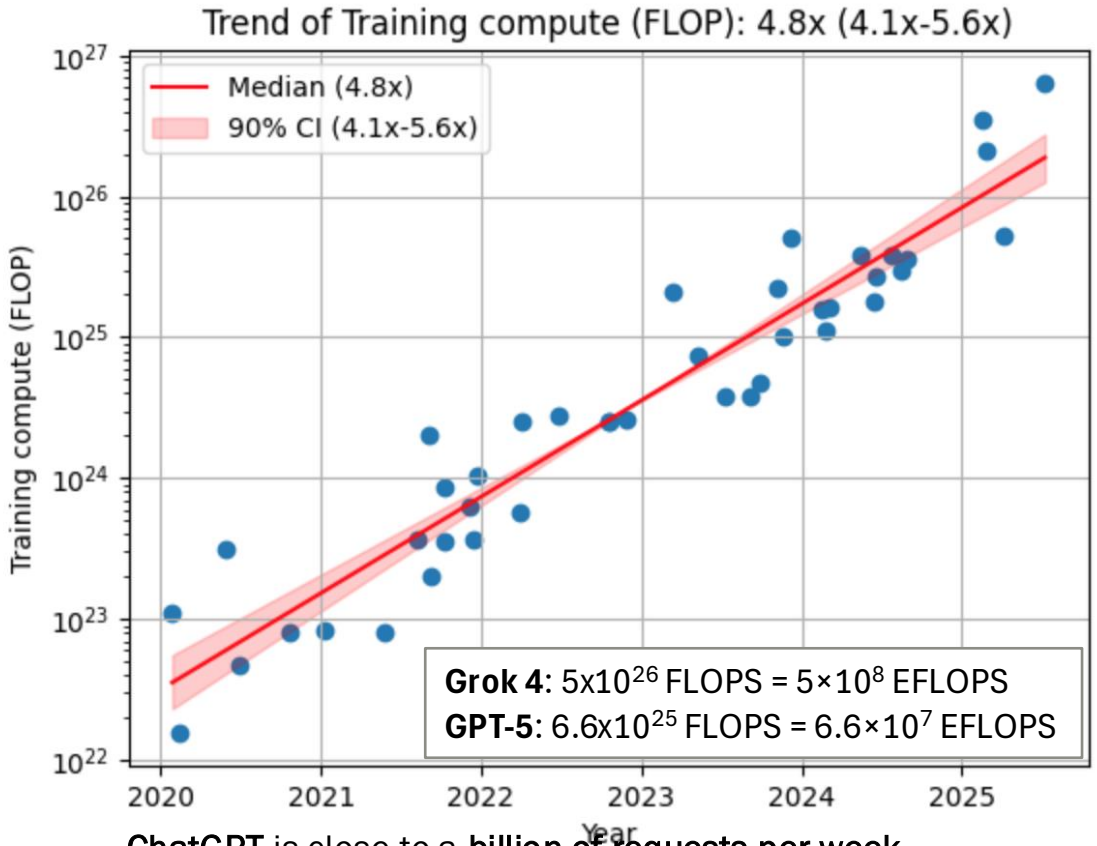
NO.
Running on a subset of the machine might yield a better performance.

High-Performance Computing in the Exascale Era
HPC Applications - LLMs



Courtesy: Xiaoguang et al.

An overview of Large AI Models and their Applications



ChatGPT is close to a **billion of requests per week**

2024: 100M - 2025: 400M - 2026: 900M

ChatGPT is now powered by GPT-5.6 (≈**3T params**)

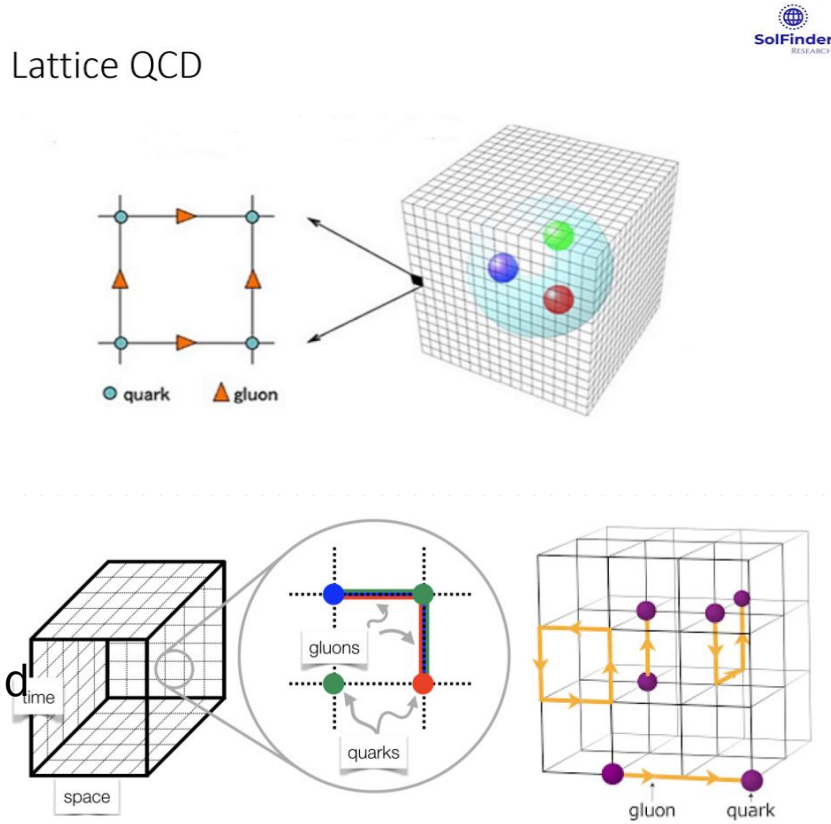
GPT-3 training: 3.5 GPU-centuries!!!

High-Performance Computing in the Exascale Era
HPC Applications – Numerical Simulation

Lattice Quantum ChromoDynamics (LQCD)

- Lattice Quantum Chromodynamics (LQCD) is a numerical method used by physicists to solve the physics of quarks and gluons at low energies.
- It is a **Monte-Carlo numerical simulation** aiming at understanding the **origin of mass** and **stability of matter** (right after the **Big-Bang**).
- The main computational kernel is the evaluation of the so-called **Wilson-Dirac operator** given by the following equation

$$D\psi(x) = A\psi(x) - \frac{1}{2}\sum_{\mu=0}^{4}\{[(I_4 - \gamma_\mu) \otimes U_{x,\mu}]\psi(x+\hat{\mu}) + [(I_4 + \gamma_\mu) \otimes U^{\dagger}_{x-\hat{\mu},\mu}]\psi(x-\hat{\mu})$$



- It is a **4D space-time simulation** specified by its parameters L_x, L_y, L_z, L_t and associated data (*double-precision* and *complex numbers*).
- Tremendous volume of data to **load/write** (from memory) and **exchange** (between processors) iteratively.
- Physicists expect meaningful results from $2^9 \times 2^9 \times 2^9 \times L_t = \mathbf{2^{27} \times L_t}$

High-Performance Computing in the Exascale Era

HPC Applications – Combinatorial Optimization

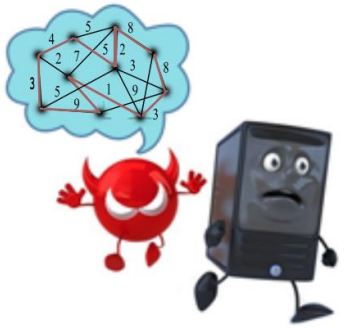
- **Combinatorial optimization** is known as the benchmark for the most **difficult computational problems.**

Example 1 The Knapsack Problem (KP) *The Knapsack Problem is the problem of choosing a subset of items such that the corresponding profit sum is maximized without having the weight sum to exceed a given capacity limit.*

$$\begin{cases} \textit{maximize} & \sum_{i=1}^{n} p_i x_i \\ \textit{subject to} & \\ & \sum_{i=1}^{n} w_i x_i \leq c \\ & x_i \leq m_i \quad i = 1, 2, \cdots, n \\ & x \in \{0,1\}^{n \times n} \end{cases}$$

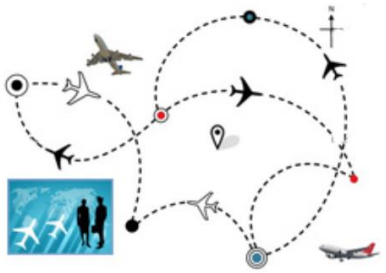
Example 2 The Traveling Salesman Problem (TSP) *Given a valuated graph, the Traveling Salesman Problem is to find a minimum cost cycle that crosses each node exactly once (tour).*

$$\begin{cases} \textit{minimize} & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \\ \textit{subject to} & \\ & \sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \cdots, n, i \neq j \quad (2.4) \\ & \sum_{i=1}^{n} x_{ij} = 1 \quad j = 1, \cdots, n, i \neq j \\ & x \in \{0,1\}^{n \times n} \end{cases}$$



Example 3 The Airline Crew Pairing Problem (ACPP) *The objective of the ACPP is to find a minimum cost assignment of flight crews to a given flight schedule. The problem can be formulated as a set partitioning problem*

$$\begin{cases} \textit{minimize} & c^T x \\ \textit{subject to} & Ax = 1 \\ & x \in \{0,1\}^n \end{cases}$$



- The Traveling Salesman Problem is one of the most difficult combinatorial problem.
- Even with small sizes such as **30 nodes**, the current **fastest supercomputer** would required **tens of years** with the brute-force algorithm.

High-Performance Computing in the Exascale Era

HPC Applications – Questions & Discussions

What other application areas do you see for HPC?

Genomics, weather forecasting, data science, seismic analysis, molecular dynamics, nuclear physics, cosmology, ...

Can you give some examples of areas that strongly depend on HPC ?

Large-scale AI, high-precision simulation, quantitative trading, cutting-edge operational research, ...

High-Performance Computing in the Exascale Era

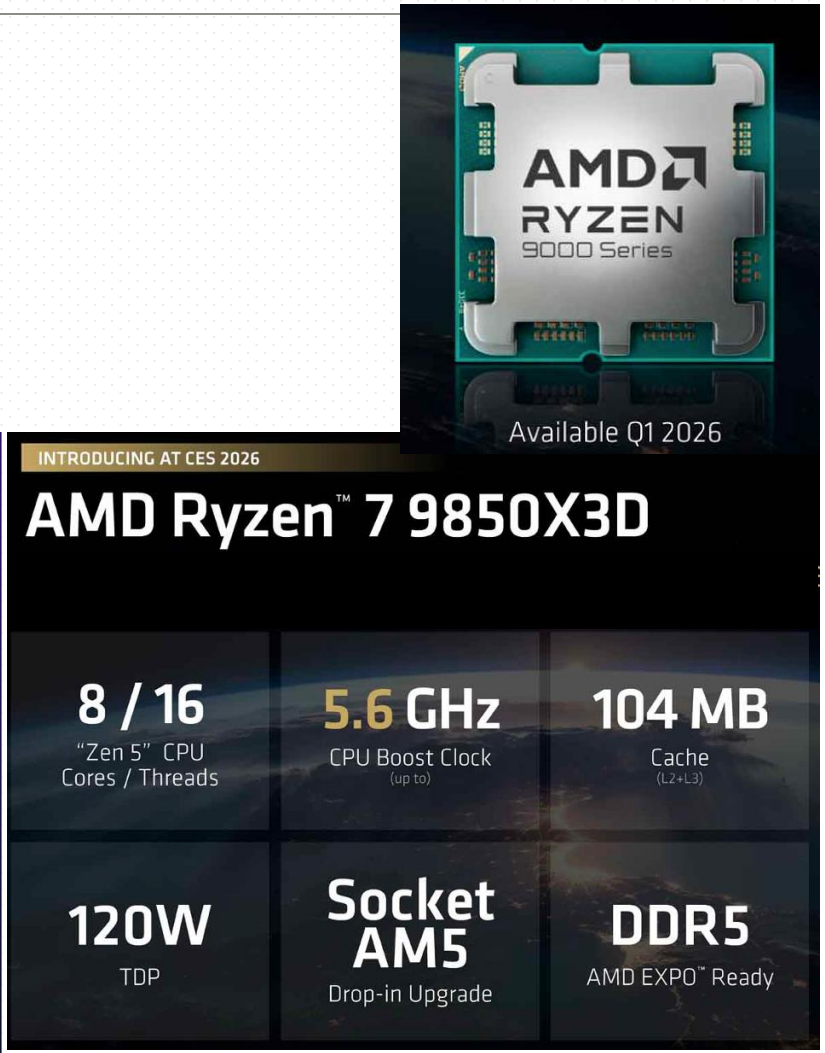
Heterogeneity – Standard CPU

Ordinary CPU remain the **basic HPC computing device**.

- Larger **number of cores** (P-cores/perf; E-cores/energy; Xe-cores/extension)
- Wider **SIMD** (vector registers and instructions set)
- Bigger and faster **memory** (with a complex hierarchical structure)
- Better **instruction pipeline**
- More **specialized units** (computation and I/O)
- **Energy-aware**

Desktop

Laptop

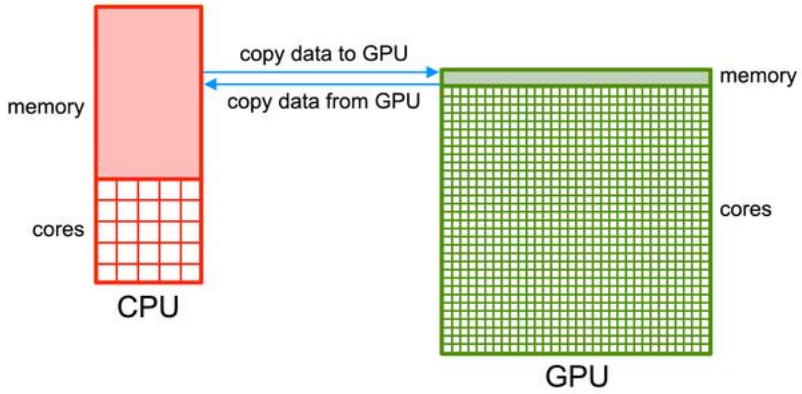


High-Performance Computing in the Exascale Era

Heterogeneity – Accelerators

The main accelerators commonly considered are **GPUs** and **FPGAs**.

- Typically run under the **coordination of an ordinary CPU**
- **Large number of cores** (thus so-called **many-core processor**)
- Very fast on **regular computation** (highly parallel & deep pipeline)
- Faster on **low-precision arithmetic** (improved on high-precision)
- Nowadays considered as **energy-efficient solutions**
- Newer GPUs are becoming **more CPU-like**



Family	CUDA Cores	TFLOPS FP32 - FP16
Fermi	512	1.5
Kepler	1,536	3.0
Pascal	2,560	9.0
Volta	5,120	15 - 125
Ampere	6,912	20 - 312
Hopper	16,896	67 - 1000
Blackwell	20,800	75 - 2250

AMD RADEON

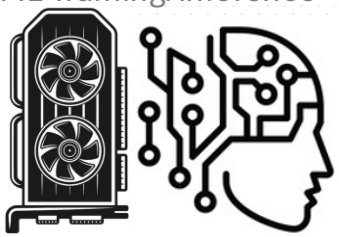
	AMD RADEON™ RX 9070	AMD RADEON™ RX 9070 XT
AMD™ RDNA 4 Compute Units	56	64
HW RT Accelerators	56	64
HW AI Accelerators	112	128
Peak AI TOPS (INT4 w/Sparsity)	1165 TOPS	1557 TOPS
Boost Clock	2.52 GHz	2.97 GHz
Video Memory	16GB	16GB
Total Board Power	220W	304W

```
data = open("input.dat");          # read the data on the CPU
copyToGPU(data);                   # copy the data to the GPU
matrix_inverse(data.gpu);          # perform a matrix operation on the GPU
copyFromGPU(data);                 # copy the resulting output to the CPU
write(data, "output.dat");         # write the output to file on the CPU
```

https://researchcomputing.princeton.edu/support/knowledge-base/gpu-computing

AI

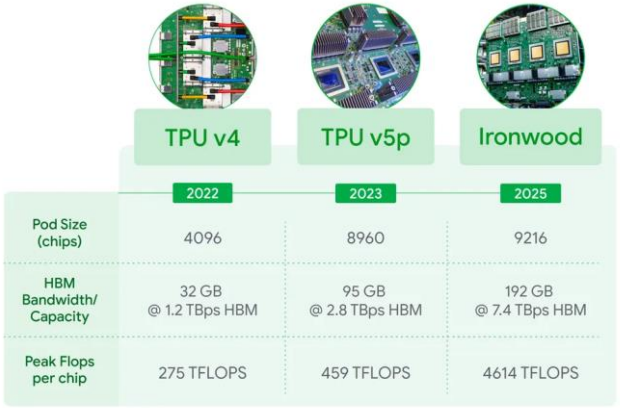
ML Training/Inference



Image/Video Processing

High-Performance Computing in the Exascale Era
Heterogeneity – Neural/Tensor Processing Units

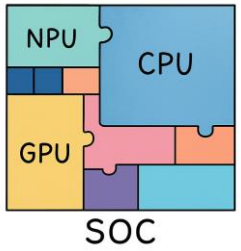
- **TPU** (Tensor **P**rocessing **U**nit)
  - Application-specific integrated circuit (**ASIC**) designed by **Google**
  - Highly efficient on **matrix multiplication and addition**
  - Based on the **systolic paradigm**
  - **INT8 Precision** (quantization) for ML/DL inference
  - **Cloud/Edge** device

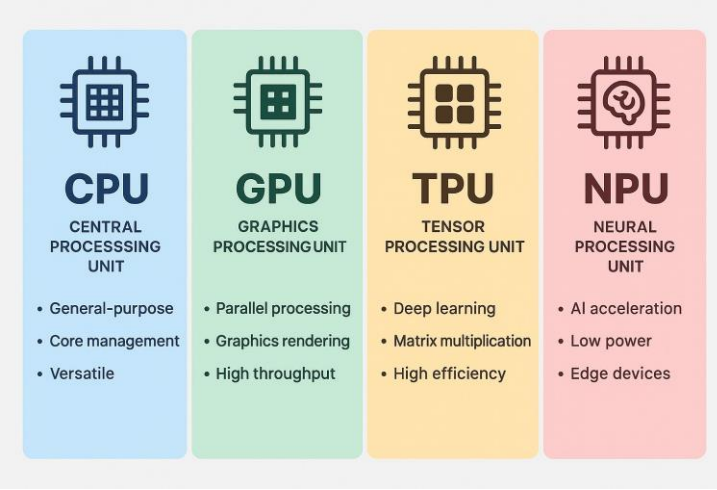


- **NPU** (Neural **P**rocessing **U**nit)
  - Specialized computer chip for **accelerating AI/ML workloads**
  - **Energy-efficient** (compared to CPU and GPU)
  - Much **more flexible than a TPU**
  - Can **compute locally** without offloading to a remote server
  - **Intel** (Intel NPU); **AMD** (Ryzen AI NPU); **Apple** (Apple Neural Engine) **Qualcomm** (Hexagon NPU); **Samsung** (Exynos AI); ...

	Chip Model	Estimated NPU Performance
Intel	Lunar Lake	50 TOPS
AMD	Ryzen Strix Point	48 TOPS
Qualcomm	Snapdragon X Elite	45 TOPS

TRENDFORCE





High-Performance Computing in the Exascale Era

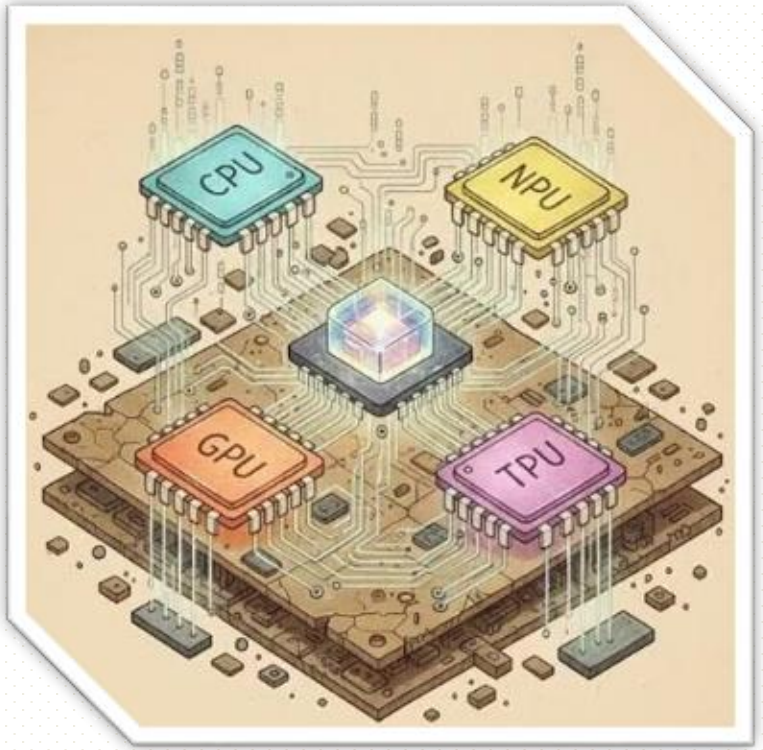
Heterogeneity – Questions & Discussions

What is the key to ease the programming of all these devices ?

Compilers; APIs; Libraries

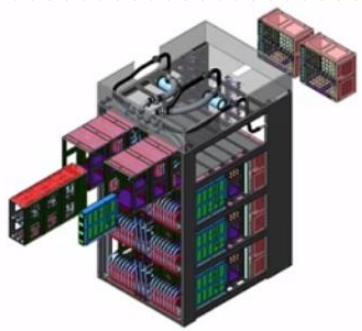
What are the potential operational issues that you can identify ?

Portability; Debugging; Updates

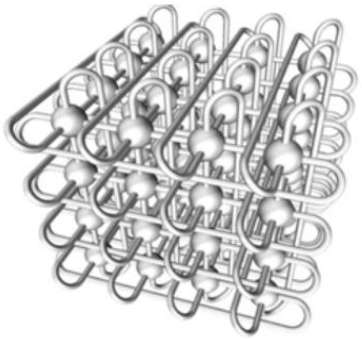


High-Performance Computing in the Exascale Era
Programming Models – Distributed Memory Parallelism

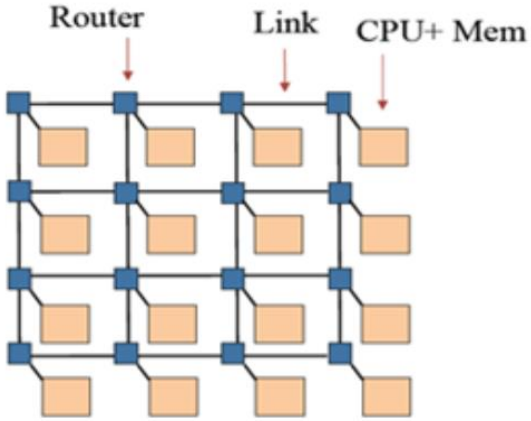
- The implementation of this model is for **fully independent compute nodes.**
  - Typical way of running on **clusters** and **supercomputers**
  - Each node has/manages its **own memory**
  - Data are shared through **explicit interprocessors communication**
  - **Ordinary Programming Language + Message Passing Library** (MPI)
- Key aspects for performance include
  - The overhead of **data exchanges** (packaging & network topology)
  - **Synchronization** between the compute nodes
  - **Load imbalance**

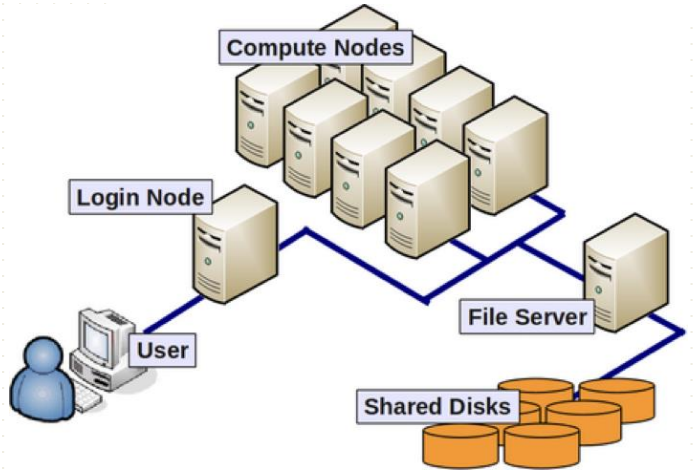


Typical Packaging



Torus Topology of Fugaku Supercomputer

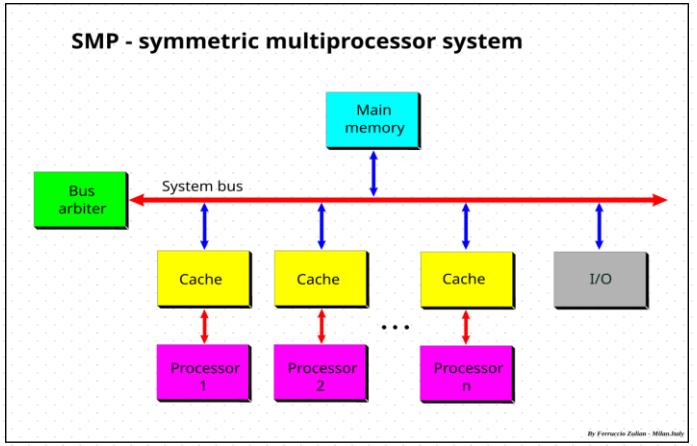


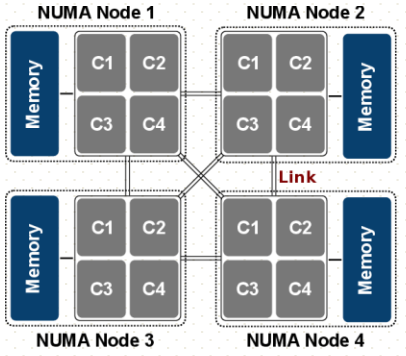


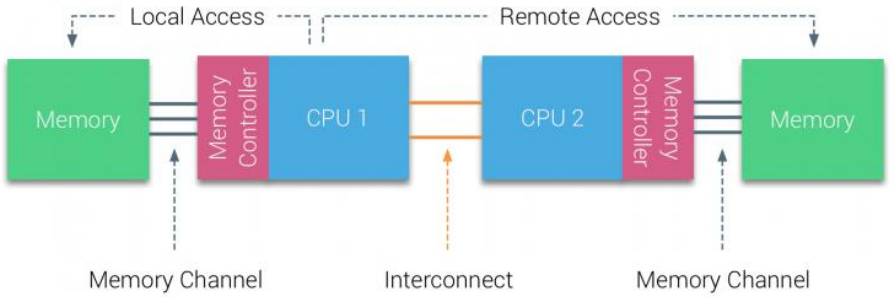
Distributed Memory Parallel Machine

High-Performance Computing in the Exascale Era
Programming Models – Shared Memory Parallelism

- The implementation of this model is for **independent CPU-cores** sharing a **common memory.**
  - Typical way of running on **multicore processors**
  - Each CPU-core has equal access to the **common main memory**
  - Both **central RAM** and **address space** are **shared** (same memory references)
  - **Ordinary Programming Language + Multithreading Library** (Pthreads)
- Key aspects for performance include
  - The overhead of **concurrent memory accesses** (also hierarchy of the caches)
  - **Thread scheduling** and **dynamic management** by the OS
  - **Synchronization** between threads (mutex)
  - **Load imbalance**
  - **NUMA effect** (if applicable)



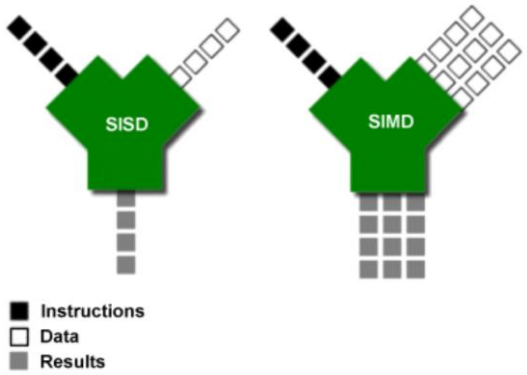




High-Performance Computing in the Exascale Era

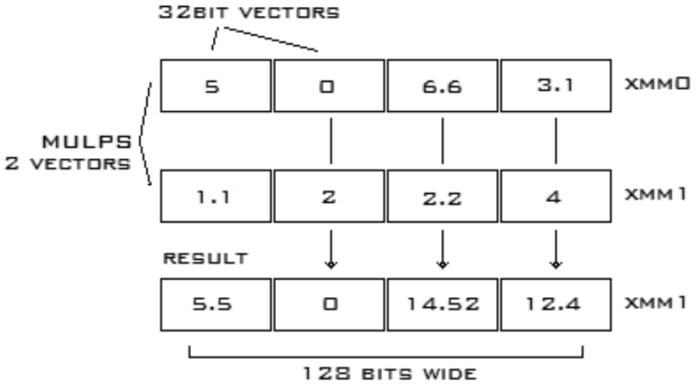
Programming Models – SIMD Parallelism

- This is **S**ingle **I**nstruction **M**ultiple **D**ata paradigm, also referred as **Vector Computing**.
  - Handled on standard CPUs through **vector units** and **vector registers**
  - Each CPU-core has **its own vector units/registers** and can perform SIMD
  - Has been extended from *graphical computation* to **any arithmetic instruction**
  - **SIMD instruction types** include *arithmetic*, *shuffling*, *memory read/write*
  - **Ordinary Programming Language + SIMD Library** (SSE; AVX)
  - Vector **register width: 128bits** (SP:×4 DP:×2); **256 bits** (SP:×8 DP:×4); and **512 bits** (SP:×16 DP:×8)



- Key aspects for performance include
  - **Memory alignment** (critical for direct access)
  - **Data locality** (ideally contiguous)
  - **Lower precision** (more vector-components)
  - **Compute-bound** (the effect is on the calculation)

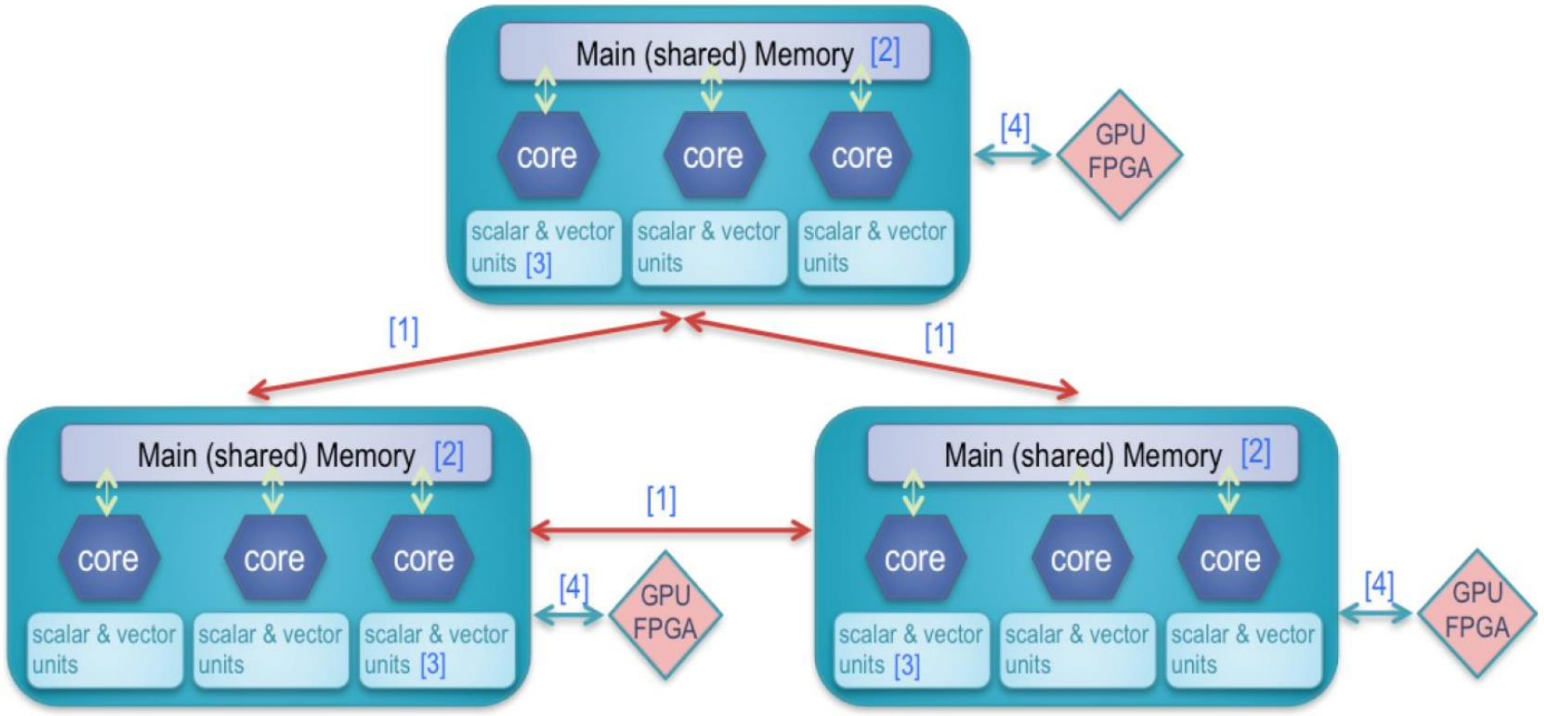
Very effective in image processing



High-Performance Computing in the Exascale Era

Programming Models – Hybrid Parallelism

- Message passing between nodes (MPI, …) [1]
- Shared memory between cores (Pthreads, OpenMP, …) [2]
- Vector computing inside a core (SSE, AVX, …) [3]
- Accelerated computing beside a node (Cuda, OpenCL, …) [4]



High-Performance Computing in the Exascale Era
Programming Models – Questions & Discussions

Do you think that a compiler can perform all levels of parallelism ?

NO. Until now, only **vectorization** is tried by the compiler whenever possible.

Can we run a distributed-memory parallel program on a single multicore processor ?

Yes, although the central RAM is actually shared.

High-Performance Computing in the Exascale Era
Getting Access to an HPC Infrastructure

- **Locally** (you own one)

  Current et Next-Generation **desktop processors** might provide **sufficient level of computing power**, provided (one more time) that applications are implemented considered **all lever of performance** (multicore/SIMD/GPU/NPU/TPU)

- **Remotely** (from an HPC Resources Center)

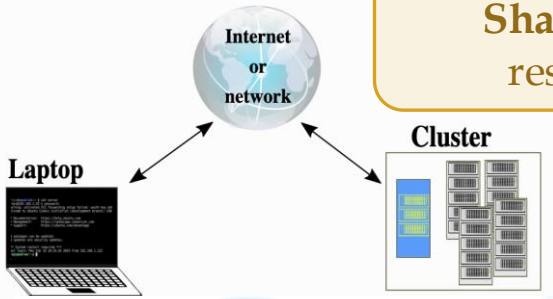
  This can be an **HPC cluster of an institution** or a **supercomputer of an HPC** Center that is made available to the community through **user accounts** and **means of remote access**.

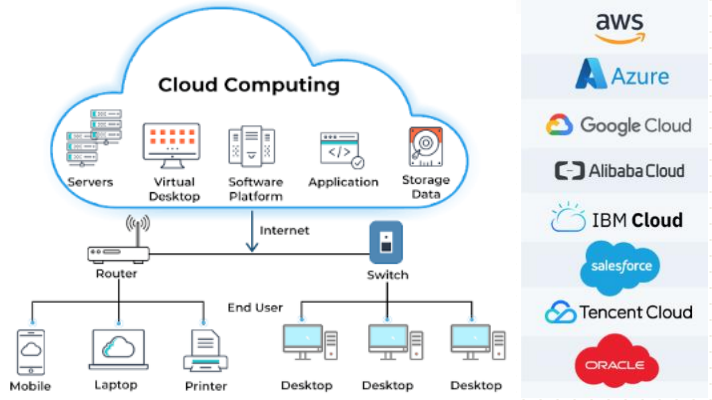
- **Cloud** (through a Cloud Provider)

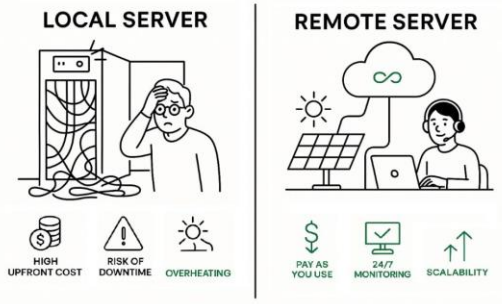
  Also a **remote access** but with a **virtualized approach**. There you can get infrastructures (IaaS), platforms (Paas) and software (Saas) on a **flexible basis**.

Pros and cons of private/local resources vs shared/remote ones ?

Private: easy/full access : costs
Shared: cost/management restricted/partial access



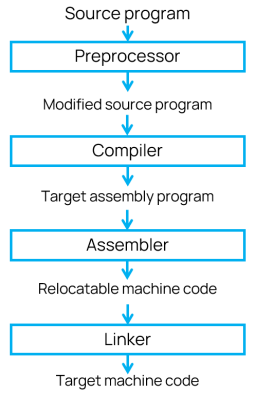




High-Performance Computing in the Exascale Era

Support Tools: Compilers – Libraries - Profilers

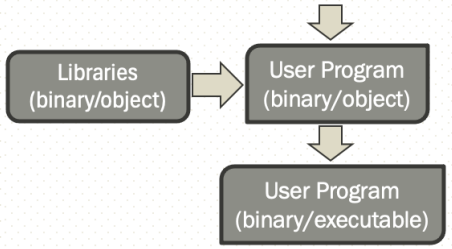
- **Compilers:** Essential to get a **functional binary** or a version that is **compatible with the device**. Their **optimization functionality** is crucial for performance.
  Source-to-source compilers are very useful to seamlessly design specific versions (multicore/GPU).



Cite one of the most popular source-to-source compiler and its major features.

OpenMP.
Generates Multithread, SIMD, and GPU(Cuda)

- **Libraries:** Very important to get access to some **hardware/system features** and useful **kernels**.
  Common key libraries are related to **OS, math, parallel scheduling, APIs, ...**



What kind of useful information can provided a profiler.

Can spot *memory leaks, poor cache efficiency, idle states, load imbalance.*

- **Profilers:** It is not always sufficient to understand performance results from **prediction models** that are typically based on **static hypothesis**. Profilers are useful to get detailed information about **runtime events** and their **associated costs**.

High-Performance Computing in the Exascale Era
Performance Aspects: Absolute/Sustained Performance

- **Absolute Performance**, also called **Peak Performance**, is the **potential computing power** of the processor/machine. It is an **upper bound** of what can be expected from a computing system based on its **technical characteristics.**

The *theoretical peak performance* in (full precision) *FLOPS* is obtained by the following formula:

$$P = n_c \times [(2f) \times s_c \times F]$$

n_c the *total number of cores*
f number of *FMA units* (FMA is $a \times b + c$)
s_c is the *SIMD width* (#components of a vector)
F *clock frequency* of the CPU (1 Hz = 1 flop)

FUGAKU Supercomputer

#processors	158,976
#cores_per_processor	48
clock_speed [GHz]	2.2
#FMA_units	2
vector_size [bits]	512

$n_c = 7630848\,;\ f = 2\,;\ s_c = \frac{512}{64}\,;\ F = 2.2$

$P = 7630848 \times 4 \times 8 \times 2.2 =$ 537

Cores	(PFlop/s)	(PFlop/s)	(kW)
7,630,848	442.01	537.21	29,899

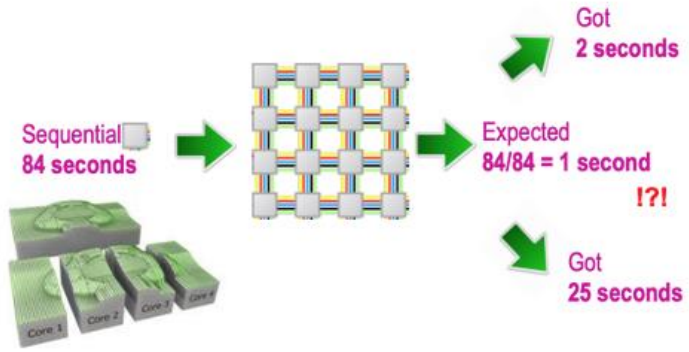
Supercomputer Fugaku - RIKEN Center for Computational Science Japan

- **Relative Performance**, also called **Sustained Performance**, is the **measured computing speed** of the machine. It therefore includes all **operational realities (hardware and system)** related to the computation such as:
Memory accesses; branching; CPU frequency variation; idle states; contention on shared resources; synchronization; overhead of data exchanges; load imbalance;
There is always a **gap between sustained performance and absolute performance.**

The main objective of performance optimization is to get the **highest computing speed** (close the aforementioned gap).

High-Performance Computing in the Exascale Era
Performance Aspects: Speedup & Scalability

- The **speedup** refers to ratio between the execution time of a given program and its transformed one.
  When it comes to parallelism, it tells **how good is the parallelization.**
  Speedup $\sigma(p) = \frac{T_s}{T_p}$ Efficiency $e = \frac{\sigma(p)}{p} \epsilon]0,1]$
  T_s: $sequential\ execution\ time$ T_p: $parallel\ execution\ time$
  p: $number\ of\ processors$



- The **scalability** refers to the ability of a program to **keep its efficiency** with a **growing size of computing resources** or an **increasing size of the problem** it solves.
  - It is typical to have a so-called **weak scalability**, which means a decreasing efficiency
  - Sometimes it **better** to **use a subset of the supercomputer**
  - Seeking **good efficiency & good scalability** is a challenging HPC task, especially on **large clusters**

High-Performance Computing in the Exascale Era

Performance Aspects: Performance Optimization

- **Performance Optimization** on an HPC system should target **absolute performance**, **efficiency** and **scalability**.

Several aspects are to be investigated for this purpose:

Computing unit: ***Scalability with a many computing units is hard***. Full ***efficiency with SIMD*** computing is also ***non-trivial***.

Memory configuration: On a NUMA machine, seek ***NUMA-unaware*** implementation.

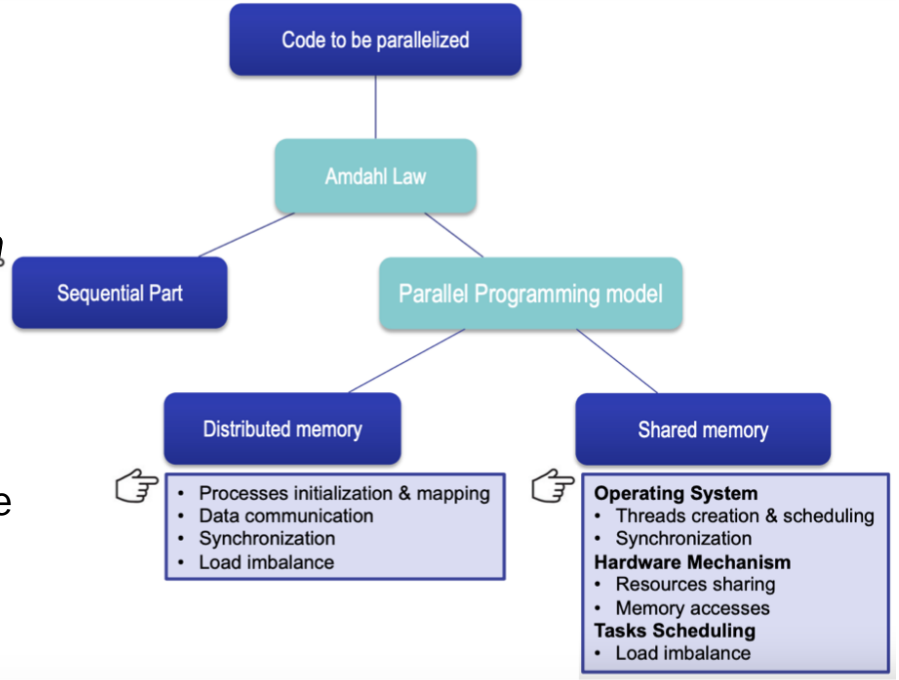
Numerical sensitivity: Despite ***numerical accuracy risks***, ***lower precision computation*** might leads to higher FLOPS (*wider SIMD* and *better data locality, Accelerators*)**.**

Heterogeneity: ***CPU/GPU data transfers*** still needs a ***serious consideration***.

Synchronization: ***Synchronizing*** (*scheduling constraints, critical sharing, concurrent updates, global conditions, checkpoints*) on ***large-scale HPC clusters is costly*** and the ***effect on the scalability*** can be detrimental.

Data exchanges: ***Their cost*** depends on ***volume*** (amount of data exchanged), ***occurrence*** (how many times) and ***quality*** (alignment with the physical interconnect).

Load balance: Parallel tasks (even SPMD) might have ***different execution time*** (*computing load, numerical and scheduling characteristics*), which effects scalability.



High-Performance Computing in the Exascale Era
Performance Aspects: Questions & Discussions

Which aspects between **absolute performance**, **efficiency**, and **scalability** do you think is the most challenging ?

Its depends on the **structure of the application** and **the characteristic of the machine**.

Do you think that an optimize program will remain efficient on another machine ?

Maybe, but there is absolute reason to expect this has the **characteristics of the machine matter**.

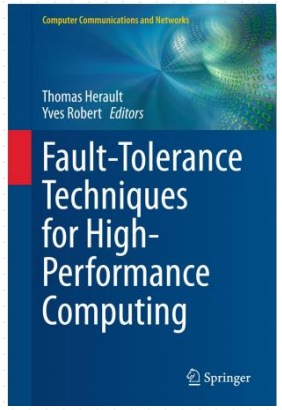
High-Performance Computing in the Exascale Era
Robustness

The **larger is a cluster** the **higher is the probability of failure**.

- **MTBF** (**M**ean **T**ime **B**etween **F**ailures) is the typical metric for a **computing system** and **associated components** too
- A hardware failure can have internal and external causes
- The so-called **fault-tolerance** is the ability to **cope with failure** without fatal consequences

About Fault-tolerance

- **Fault-tolerance** is crucial with **large-scale HPC**, especially with **long-duration tasks** (e.g.: *large numerical simulations*)
- Serious and regular **hardware maintenance** is essential
- **Software-level fault-tolerance**: *OS*, *Jobs Scheduling-Monitoring*, and *Applications*
- At the application-level, **fault-tolerance techniques** include: *regular backup and checking, restart mechanism, task migration, redundant computation, error correction, anticipation, ...*



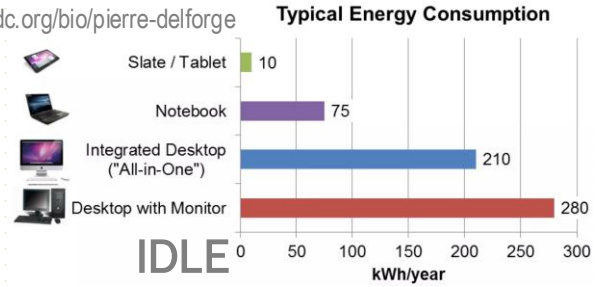
Failure-aware Program?

Periodic savings on disk

High-Performance Computing in the Exascale Era
Energy: Fundamentals

Energy for IT

https://www.nrdc.org/bio/pierre-delforge



- Energy is an **essential need for the life** of a computing system
  - **computing devices** and associated IT components
  - **cooling system** (heat dissipation is a potential source of hardware failure
- The focus on energy considers **production/supply/availability/cost/quality**(nature)
- Energy is a **critical resource with embedded systems** since it is limited (battery)
- Energy consumption **can be managed** through **related techniques** (important!)

Green500

This an annual (or bi-annual) world ranking of supercomputers based on power consumption (beside performance).

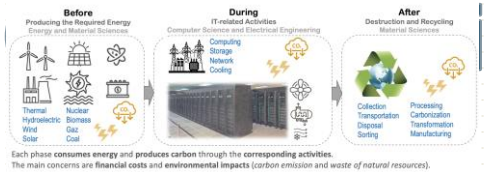
- Energy efficiency is **not correlated with computation power**
- The typical unit is **FLOPS/W** (flops per Watt)
- Addition **energy consuming mechanism** such as **cooling** should be considered

Green500 Ranking
June 2026

Source : TOP500 / Green500

Rang	TOP500	Machine	Propriétaire	Cœurs	Power (kW)	Efficiency (GFlops/W)
1	445	KAIROS	CALMIP / Univ. Toulouse - CNRS	13,056	46 kW	73.282
2	192	ROMEO-2025	ROMEO HPC Center - Champagne-Ardenne	47,328	160 kW	70.912
3	250	Levante GPU extension	DKRZ - Deutsches Klimarechenzentrum	35,904	110 kW	69.426
4	238	Isambard-AI phase 1	University of Bristol	34,272	117 kW	68.835
5	312	Otus (GPU only)	Universität Paderborn - PC2	19,440	68 kW	68.177
6	88	Capella	TU Dresden, ZIH	85,248	445 kW	68.053
7	363	SSC-24 Energy Module	Samsung Electronics	11,200	69 kW	67.251
8	116	Helios GPU	Cyfronet	89,760	317 kW	66.948
9	451	AMD Ouranos	Atos / Eviden	16,632	48 kW	66.464
10	85	Portage	Hewlett Packard Enterprise	129,024	370 kW	66.277

High-Performance Energy: Lifecycle in the Exascale Era



Before

Producing the Required Energy

Energy and Material Sciences

Thermal
Hydroelectric
Wind
Solar

Nuclear
Biomass
Gaz
Coal

During

IT-related Activities

Computing Science and Electrical Engineering

Computing
Storage
Network
Cooling

After

Destruction and Recycling

Material Sciences

Collection
Transportation
Disposal
Sorting

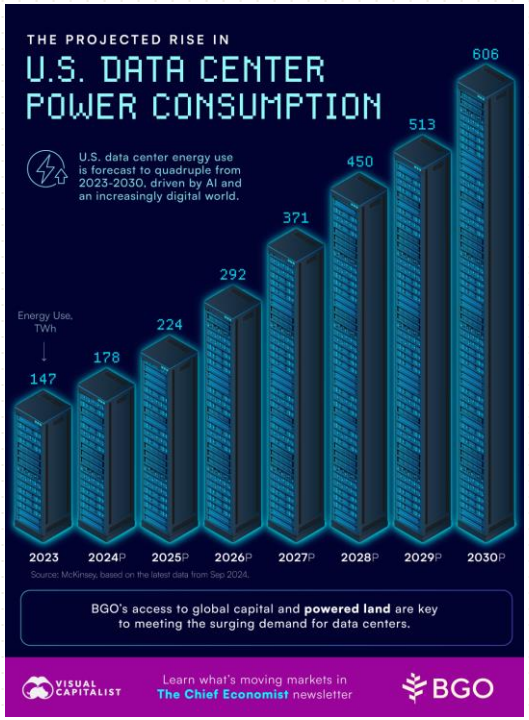
Processing
Carbonization
Transformation
Manufacturing

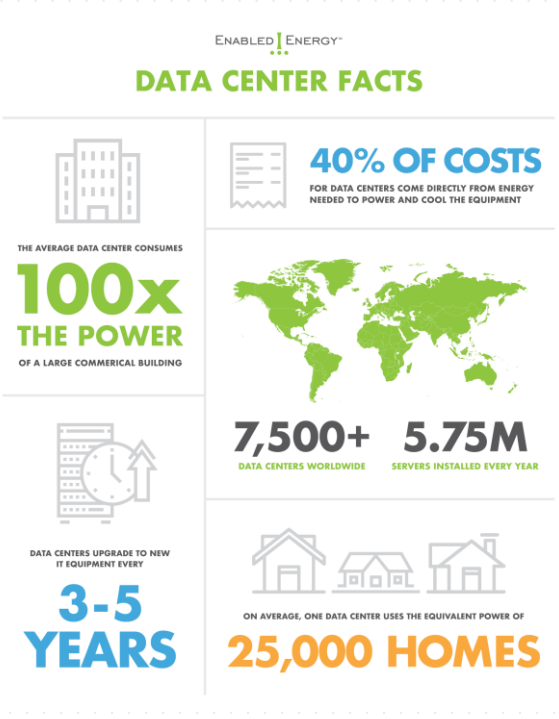
Each phase **consumes energy** and **produces carbon** through the **corresponding activities**.

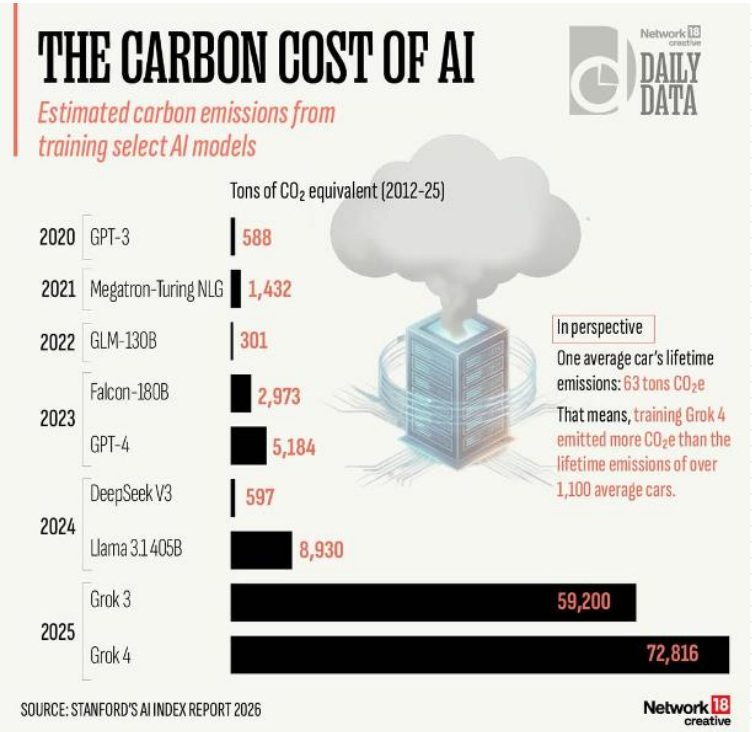
The main concerns are **financial costs** and **environmental impacts** (*carbon emission* and *waste of natural resources*).

High-Performance Computing in the Exascale Era
Energy: Costs & Impacts

- **Energy** has a **financial cost** and a **CO_2 equivalent** (both depend on the considered region)
- The corresponding **CO_2 emission** is a serious concern, thus the focus on **renewables**
- The **consumption of natural resources** for energy production and cooling system (water) is a also **a concern**
- **AI Boom** has exacerbated the the **alarming level of energy needs** and **natural resources waste**



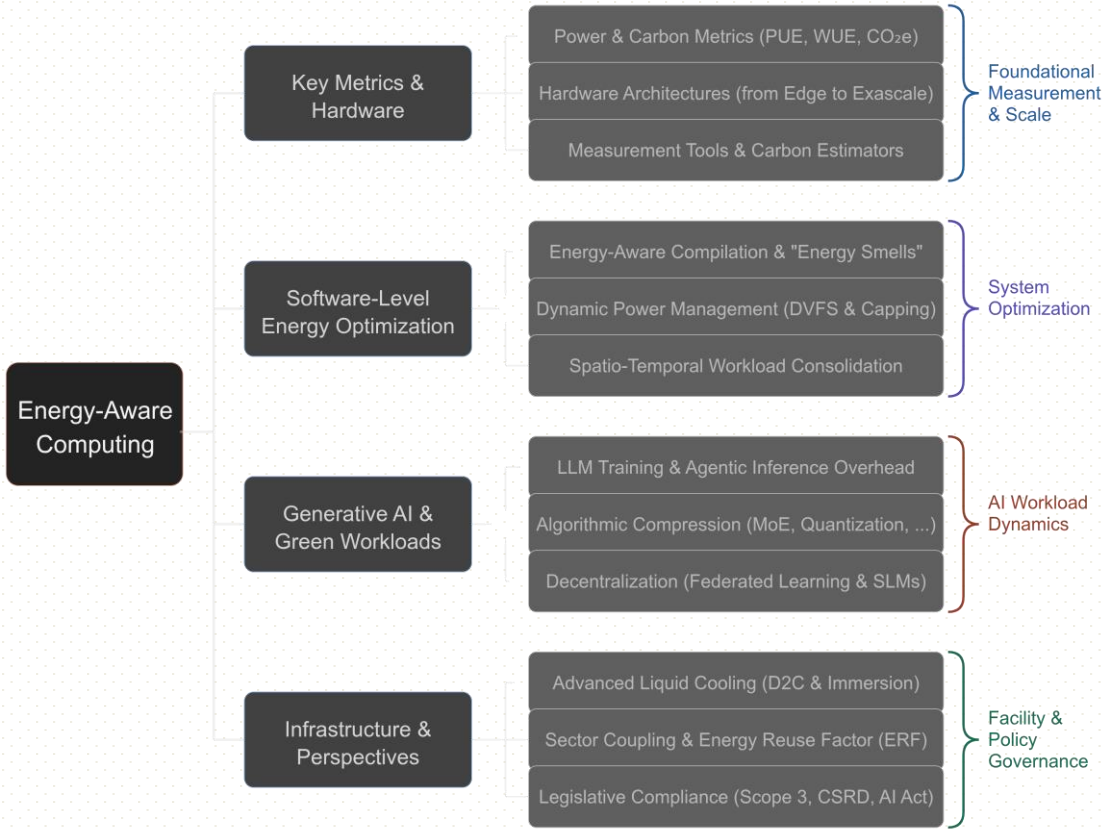




High-Performance Computing in the Exascale Era
Energy: Energy-Aware Computing

- Use **low-power computing devices** whenever possible
- Use **energy-aware tasks managers** for job scheduling
- Play with hardware/OS energy-related **settings/parameters**
- **IT resources sharing**
- **Wisely use** power-hungry applications like **generative AI**
- Give a serious **attention to energy** beside performance

- From the programming perspective, power-aware computing is an increasingly important topic, which is addressed through **energy-aware program design & implementation** and tools for **dynamic energy-aware monitoring.**



High-Performance Computing in the Exascale Era
Energy: Questions & Discussions

Could you understand the concern related to data centers ?

Mainly **carbon footprint** and **waste of natural resources** (such as **water**).

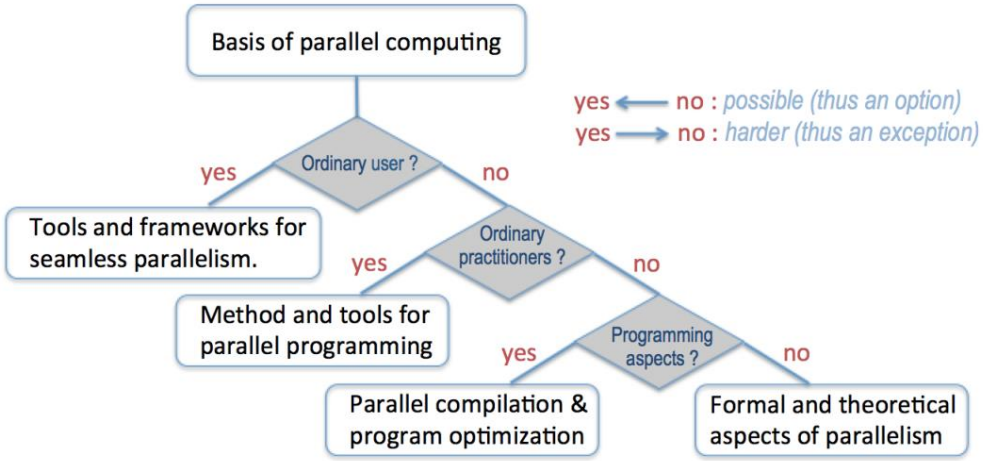
How can we reduce power consumption ?

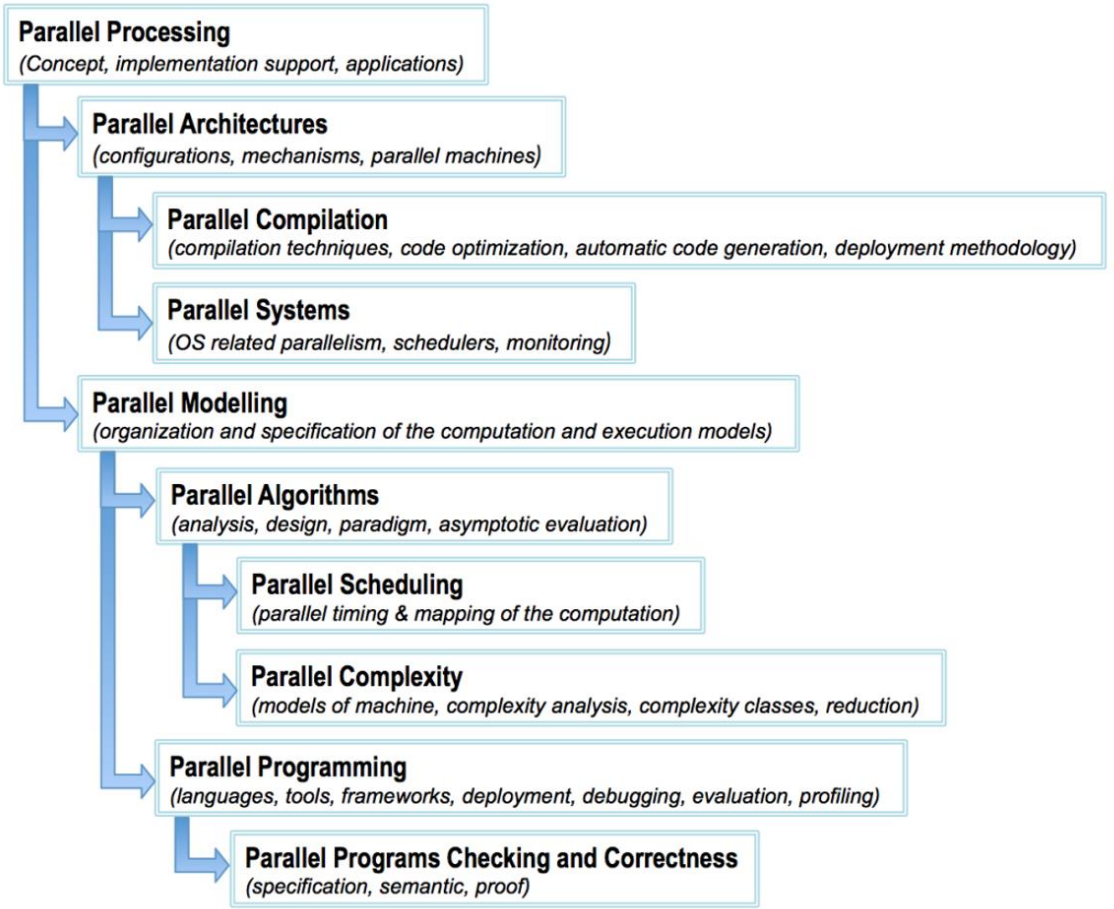
Energy-efficient devices, wise use of IT, good energy management, regulations, energy-aware design, reuse, … .

High-Performance Computing in the Exascale Era

HPC Curriculum:

- Since the advent of **multicore processors**, consideration for **parallel computing** has considerably spread.
- **Parallel computing** is **not just the use of parallel programs**
- **Understanding parallelism** at design and execution levels is important, thus the need of **corresponding courses**
- Fundamentals of **parallel modelling** and **parallel algorithm** sh be taught to HPC students
- **AI Code Generation is** a **threat to ordinary programmers**





Claude TADONKI, *HPC Curriculum and Associated Resources in the Academic Context*, arXiv:1802.01110

High-Performance Computing in the Exascale Era
Perspectives

- **HPC** is **evolving** at a **high speed** and with **significant achievements**
- The **AI boom** has **magnified this evolution** and **has pushed the horizon** of HPC
- The **HPC Landscape** will have to deal with **large-scale data centers** and the corresponding implications
- The rise of **specialized HPC devices** will continue for specific but demanding computing tasks
- The **HPC ecosystem** will be increasingly diverse, with heterogeneity becoming ordinary
- **Energy and environmental concerns** will have a noticeable influence on HPC activities
- **Quantum machines** are on the way

END

Thanks for your attention and participation