Systems for Machine Learning (CMSC828G)

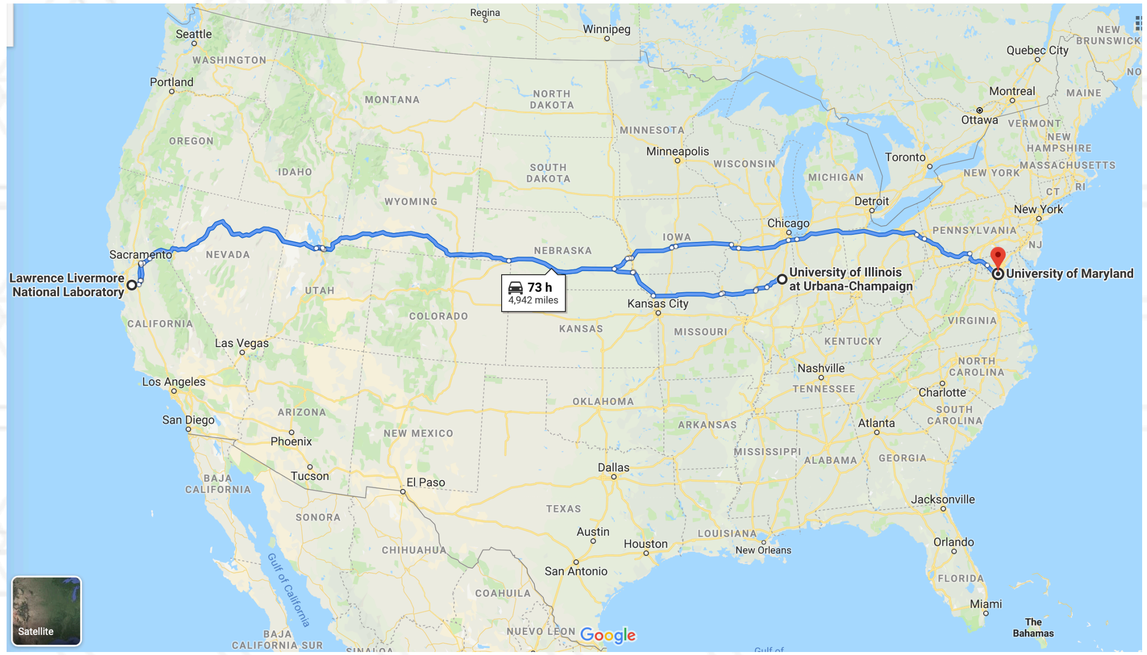
CMSC828G: Course Overview

Abhinav Bhatele, Department of Computer Science

UNIVERSITY OF MARYLAND

About the instructor — Dr. Bhatele

- Ph.D. from the University of Illinois at Urbana-Champaign (midwest)
- Spent eight years at Lawrence Livermore National Laboratory (SF bay area)
- 7th year at the University of Maryland
- Research areas:
  - High performance computing
  - Distributed AI



This course is

- A seminar course on recent advances in systems for machine learning (SysML)
- Qualifying course for MS/PhD: Computer Systems and Artificial Intelligence
- Work expected and grading:
  - Two to three programming assignments: 20%
  - Class participation: 10%
    - Submit questions/discussion topics on assigned paper readings: 7.5%
    - Present an overview of one paper (in groups of two): 2.5%
  - Two midterm exams: in class on March 3 and April 9: 25% each
  - Final (group) project: 20% (due on May 11)

Course topics

- Introduction to high performance computing (2 weeks)
- Introduction to deep learning (1 week)
- Challenges in high performance DL (1 week)
- Profiling and modeling performance of DL workloads
- Distributed training (1.5 weeks)
- On-node performance optimizations (1 week)
- ML optimizations for systems (1.5 weeks)
- Inference (1 week)
- Data movement and I/O (2 weeks)

Tools we will use for the class

- Syllabus, lecture slides, assignment/project descriptions on course website:
  - https://www.cs.umd.edu/class/spring2026/cmsc828g
- All student submissions will be on gradescope:
  - https://www.gradescope.com/courses/1242135
- Discussions on Piazza:
  - https://piazza.com/umd/spring2026/cmsc828g
- If you want to contact the course staff outside of piazza, send an email to: cmsc828g@cs.umd.edu

Accounts on zaratan and nexus

- Zaratan is the UMD DIT cluster
- Nexus is the CS/UMIACS cluster
- You should receive an email when your accounts are ready for use
- Do NOT use the class allocations for research unrelated to the course
- Helpful resources:
  - https://hpcc.umd.edu/hpcc/help/usage.html
  - https://wiki.umiacs.umd.edu/umiacs/index.php/Nexus

Programming assignments

- You can write and debug most of your assignment locally
- On zaratan:
  - vim, emacs
  - Do not use VSCode to ssh into zaratan
- Eventually, you should ensure that your code runs correctly on zaratan

Group Projects

- Self form into groups of 2-3
- Projects should be at the intersection of systems + ML
  - Using parallel systems to optimize an ML workload
- Timeline:
  - Group formation and project proposal: TBD
  - Interim report: TBD
  - Final presentation: April 30 — May 7
  - Final report and code: May 11

Excused absence

Any student who needs to be excused for an absence from a single lecture, due to a medically necessitated absence shall make a reasonable attempt to inform the instructor of his/her illness prior to the class. Upon returning to the class, present the instructor with a self-signed note attesting to the date of their illness. Each note must contain an acknowledgment by the student that the information provided is true and correct. Providing false information to University officials is prohibited under Part 9(i) of the Code of Student Conduct (V-1.00(B) University of Maryland Code of Student Conduct) and may result in disciplinary action.

Self-documentation may not be used for Major Scheduled Grading Events (midterm exam, project presentation) and it may only be used for one class meeting during the semester. Any student who needs to be excused for a prolonged absence (two or more consecutive class meetings), or for a Major Scheduled Grading Event, must provide written documentation of the illness from the Health Center or from an outside health care provider. This documentation must verify dates of treatment and indicate the timeframe that the student was unable to meet academic responsibilities. In addition, it must contain the name and phone number of the medical service provider to be used if verification is needed. No diagnostic information will ever be requested.

Use of LLMs

AI assistance (**ChatGPT, VSCode+Copilot, Cursor, Claude Code, Codex,** etc.) is not permitted for coding, writing, editing, or any other part of the class participation tasks (*preparing questions and presentations on assigned reading*) and *programming assignments*. Even though we expect you will use these tools in the future, this approach will help you build a solid understanding of the subject matter, which will benefit your future career.

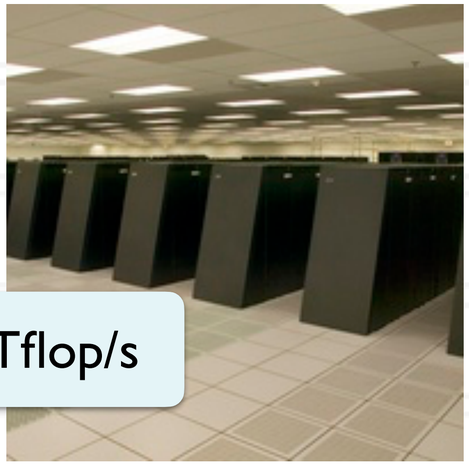
You can use AI tools such as ChatGPT as you would use Google for research. However, you cannot copy solutions generated by an AI tool or returned by a Google search. You must demonstrate independent thought and effort. If you use any AI tools for anything class related, you must mention that in your answer/report (failure to do so can be considered academic dishonesty). Please note that LLMs provide unreliable information, regardless of how convincingly they do so. If you are going to use an LLM as a research tool in your submission, you must ensure that the information is correct and addresses the actual question asked.

Academic dishonesty

- In the last few years, we have discovered that 10-15% of students have been involved in academic dishonesty
- We use software to detect plagiarism in programming assignments
- We take this extremely seriously and if we suspect something, we must report it to the Office of Student Conduct (OSC)

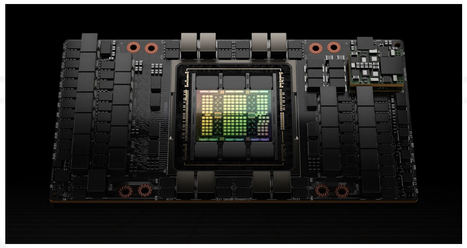
The evolution of HPC systems and rise of a new revolution in AI

- In the last two decades, an enormous amount of compute power has become available
- Large datasets and open source software such as PyTorch have also emerged
- Led to a frenzy in the world of AI and the effects are being felt in almost every other domain



Top500 Rpeak - 91.75 Tflop/s

IBM Blue Gene/L, 2004

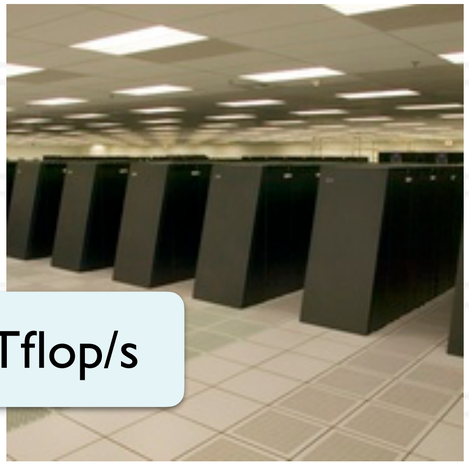


FP64 - 34 Tflop/s

NVIDIA H100, 2024

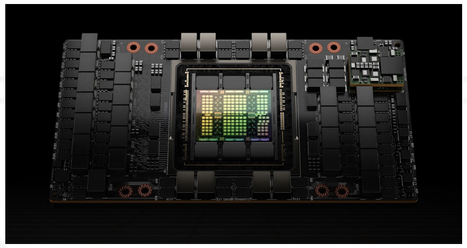
The evolution of HPC systems and rise of a new revolution in AI

- In the last two decades, an enormous amount of compute power has become available
- Large datasets and open source software such as PyTorch have also emerged
- Led to a frenzy in the world of AI and the effects are being felt in almost every other domain



Top500 Rpeak - 91.75 Tflop/s

IBM Blue Gene/L, 2004

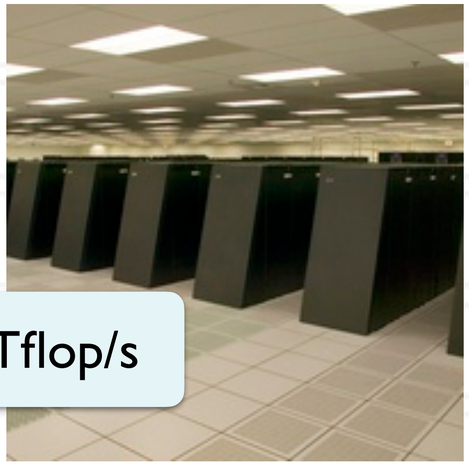


FP16 - 989 Tflop/s

NVIDIA H100, 2024

The evolution of HPC systems and rise of a new revolution in AI

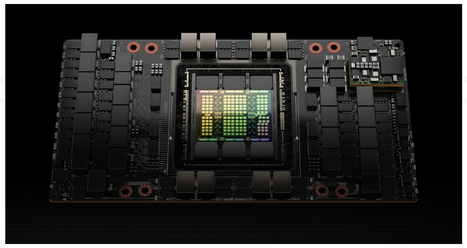
- In the last two decades, an enormous amount of compute power has become available
- Large datasets and open source software such as PyTorch have also emerged
- Led to a frenzy in the world of AI and the effects are being felt in almost every other domain



Top500 Rpeak - 91.75 Tflop/s

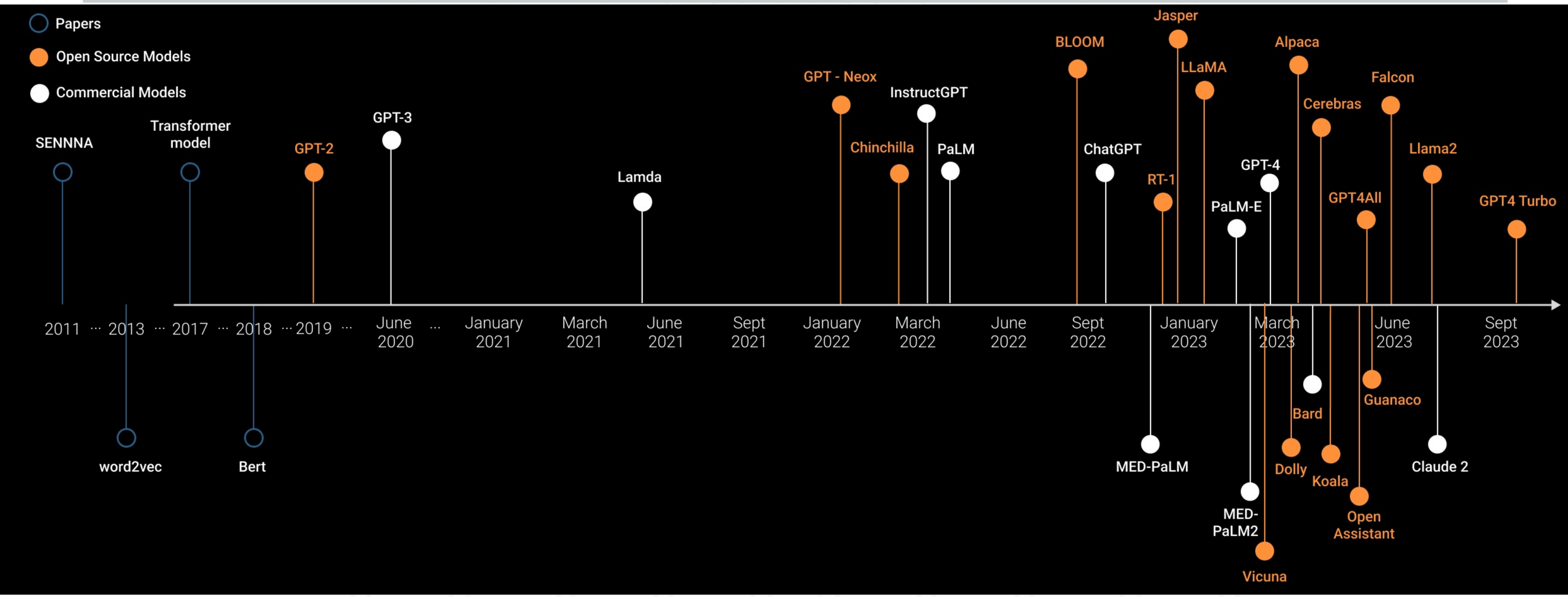
IBM Blue Gene/L, 2004

10.63 Exaflop/s!!



NVIDIA H100, 2024

A timeline of evolution of LLMs



https://www.ml6.eu/resources/large-language-models

Computer systems and ML

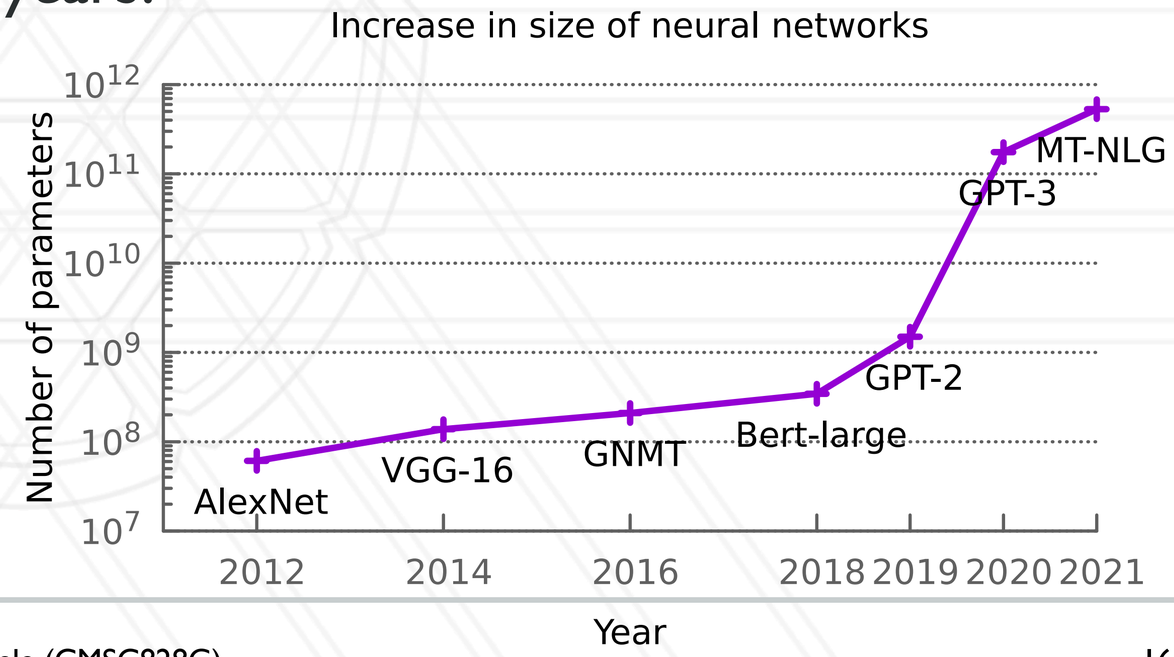
- Architecture: GPUs, TPUs, …
- Memory management and optimization
- Networks: communication on data center networks, cloud servers, HPC systems
- Storage: File input/output
- Performance engineering and optimization: compute kernels

Parallel computing and ML

- Large models do not fit on a single GPU
- Training: with a large amount of data, it can take too long on a single GPU
- Inference: large models and/or serving a large number of users can require multiple GPUs

Do we really need parallel resources?

- The largest model you can run on an H100 96 GB GPU is around 3.5-4 billion parameters
- On a single node (with four H100 GPUs): around ~16 billion parameters model
- Training a 16B parameter would take 33 years!
- OpenAI's GPT 4.0 is estimated to have 1.8 trillion parameters
- Meta's Llama-3.1-405B has more than 400 billion parameter



Terms and definitions

- Model training: process of adjusting a model's parameters using input data (and correct output) to accurately predict the output for unseen data
- Inference: using a trained model to make predictions for new inputs
- Fine-tuning: starting with a pre-trained model and adapt it to a specific task

Large supercomputers

- Top500 list: https://www.top500.org/lists/top500/2025/06

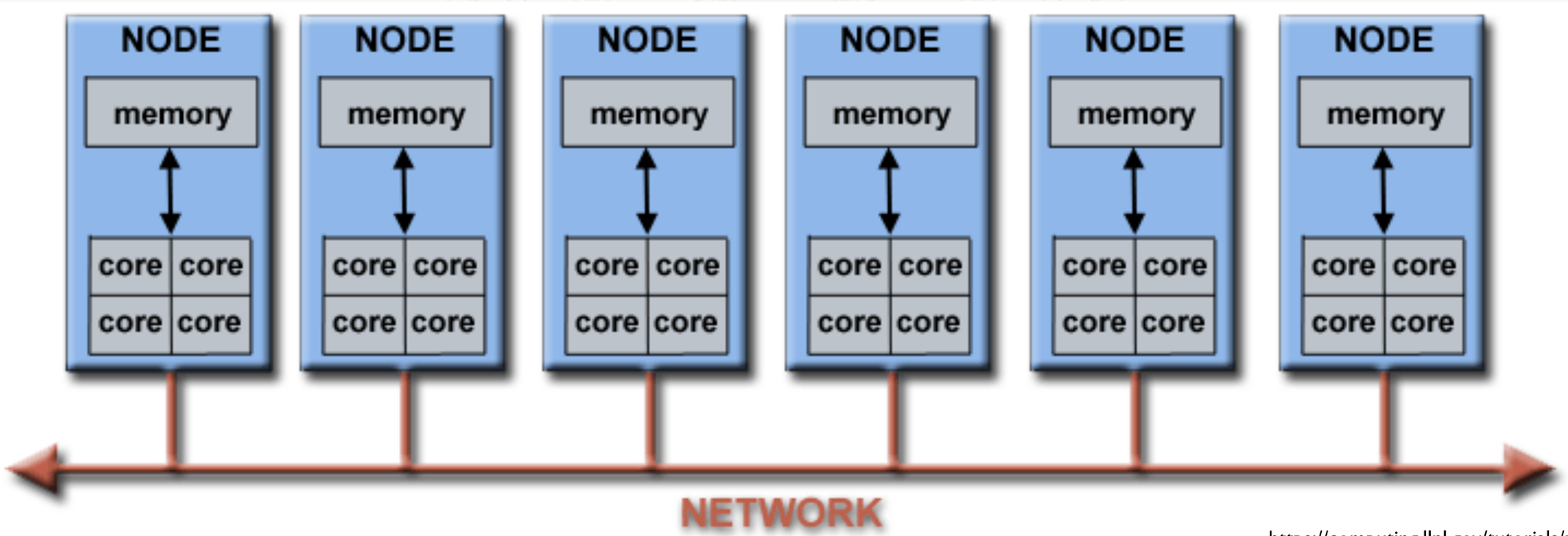
Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**El Capitan** - HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS, HPE DOE/NNSA/LLNL United States	11,039,616	1,742.00	2,746.38	29,581
2	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Cray OS, HPE DOE/SC/Oak Ridge National Laboratory United States	9,066,176	1,353.00	2,055.72	24,607
3	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
4	**JUPITER Booster** - BullSequana XH3000, GH Superchip 72C 3GHz, NVIDIA GH200 Superchip, Quad-Rail NVIDIA InfiniBand NDR200, RedHat Enterprise Linux, EVIDEN EuroHPC/FZJ Germany	4,801,344	793.40	930.00	13,088
5	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	

El Capitan @ LLNL

https://en.wikipedia.org/wiki/El_Capitan_(supercomputer)

Data center / HPC cluster

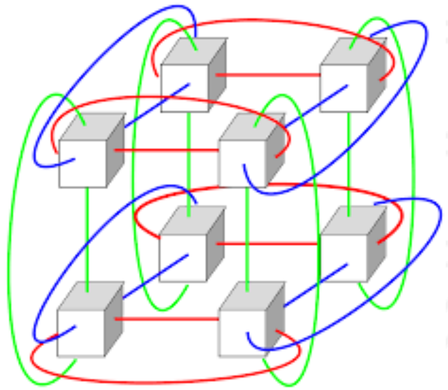
- A set of nodes or processing elements connected by a network.
- Compute node: A shared-memory unit (optionally has GPUs)



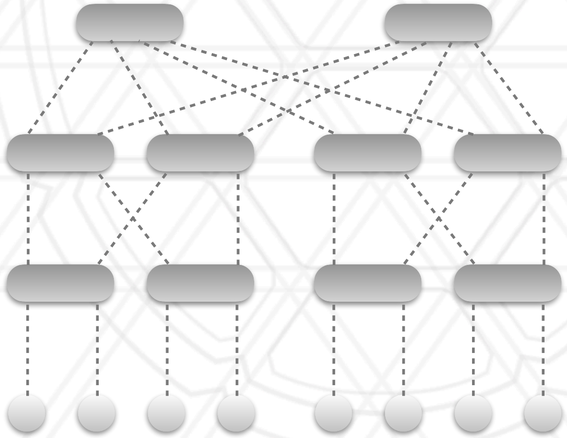
https://computing.llnl.gov/tutorials/parallel_comp

Interconnection networks

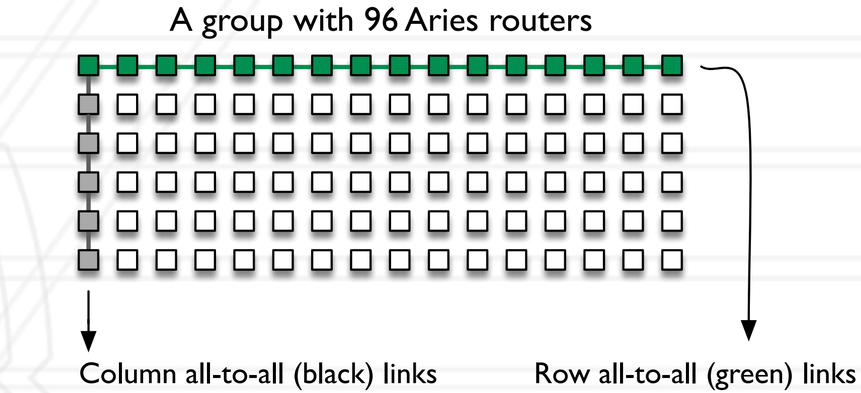
- Different topologies for connecting nodes together
- Used in the past: torus, hypercube
- More popular currently: fat-tree, dragonfly

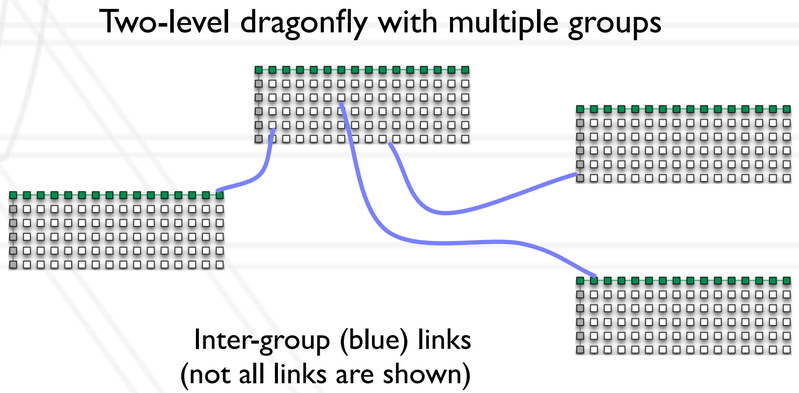


Torus



Fat-tree

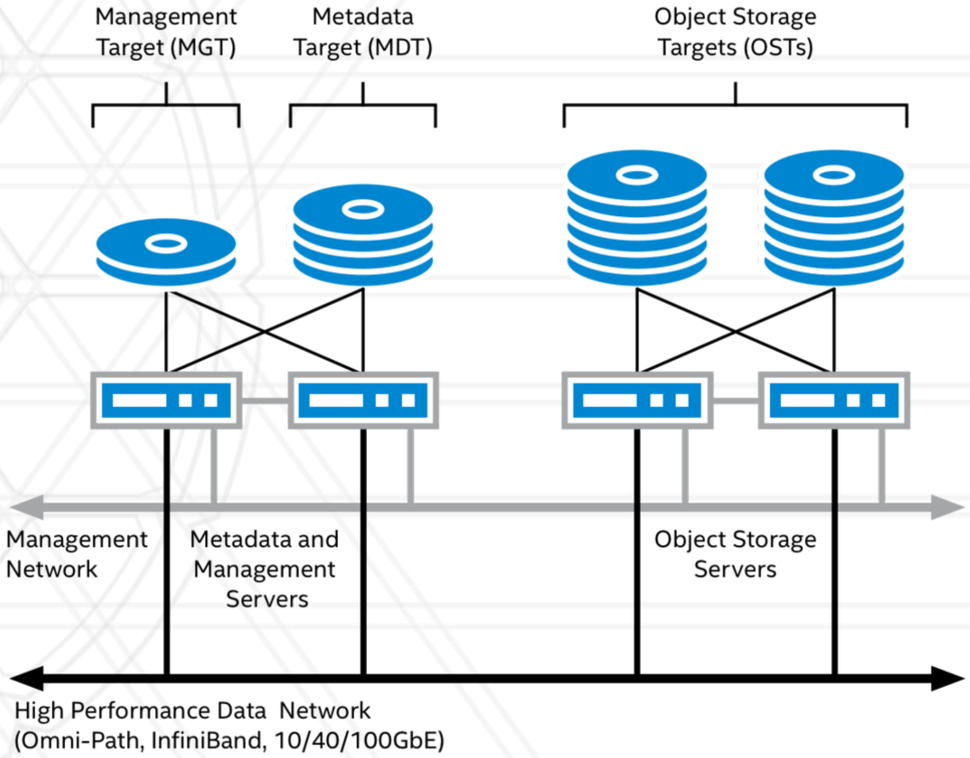




Dragonfly

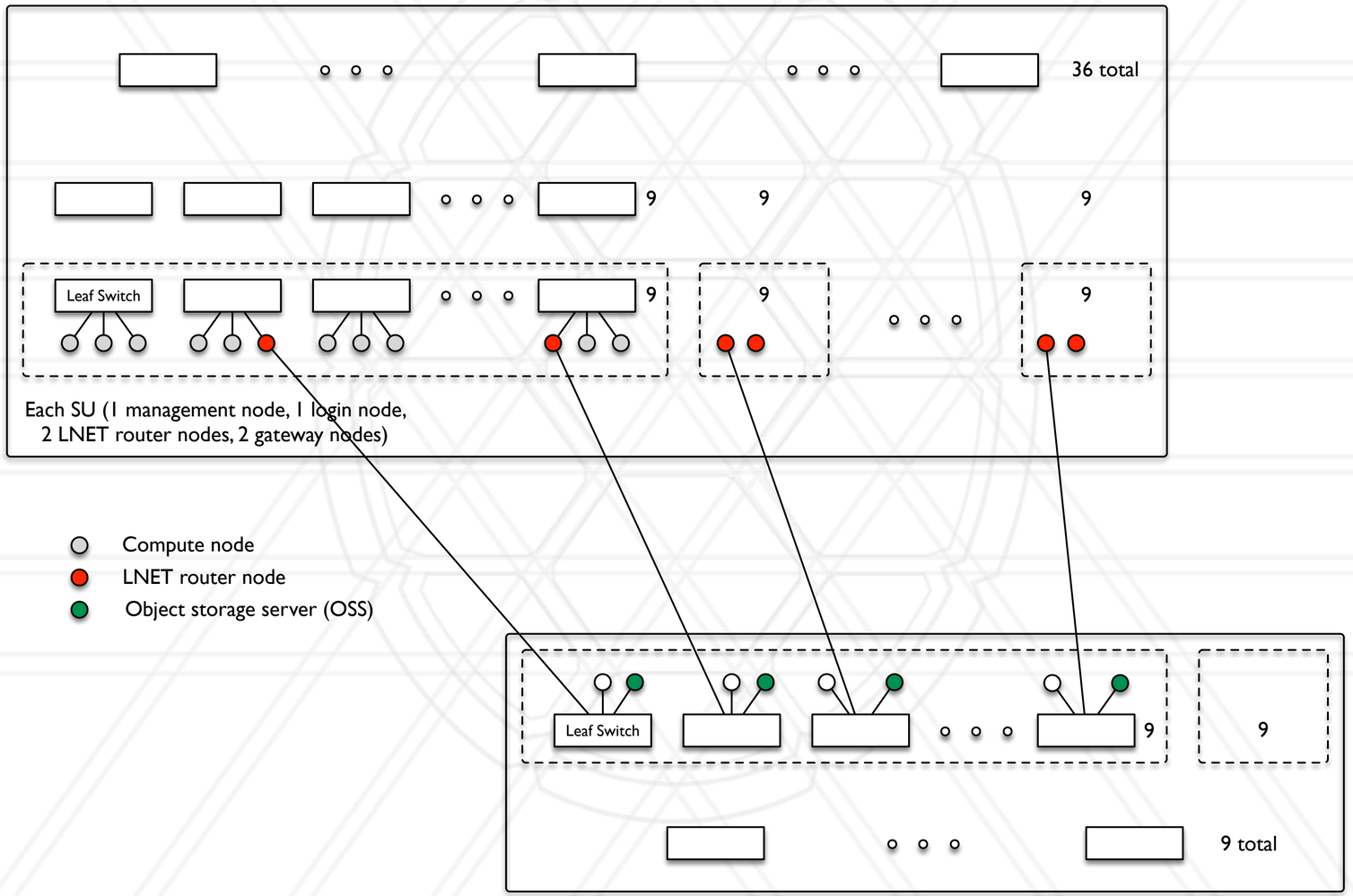
I/O sub-system / Parallel file system

- Home directories and scratch space on clusters are typically on a parallel file system
- Compute nodes do not have local disks
- Parallel filesystem is mounted on all login and compute nodes

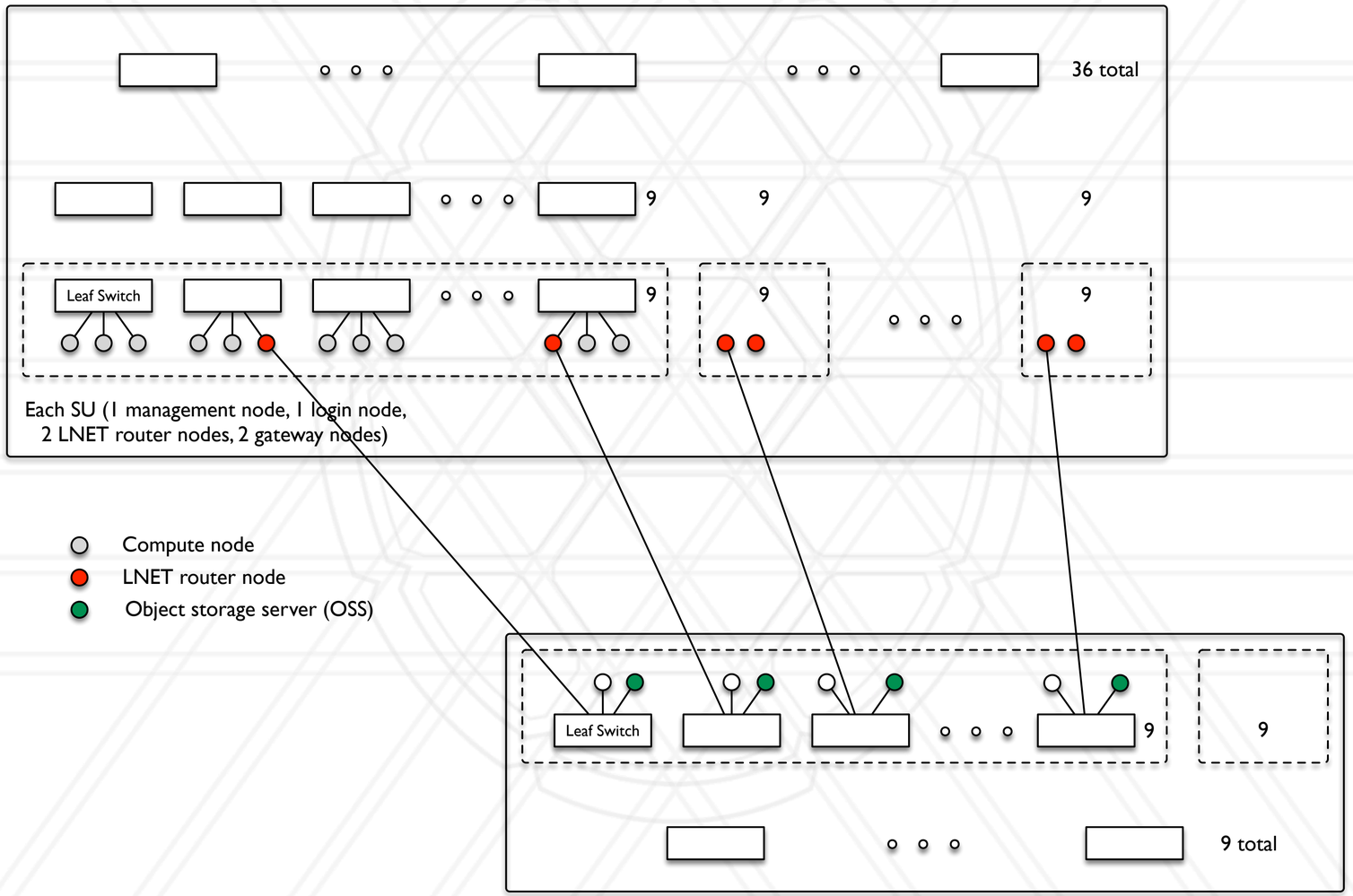


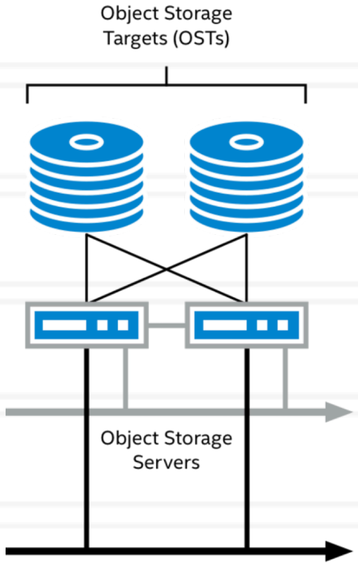
http://wiki.lustre.org/Introduction_to_Lustre

Parallel file system or I/O sub-system



Parallel file system or I/O sub-system

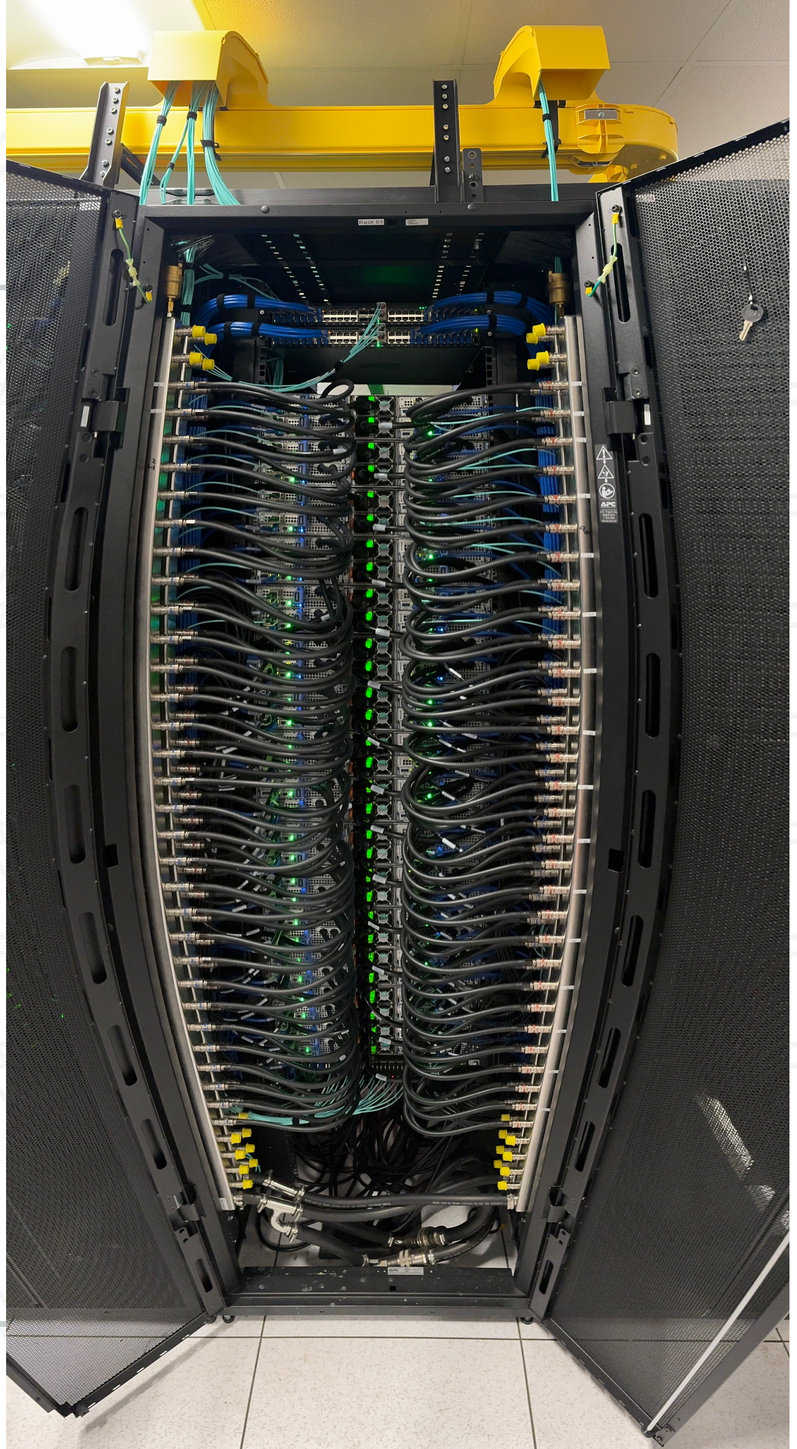




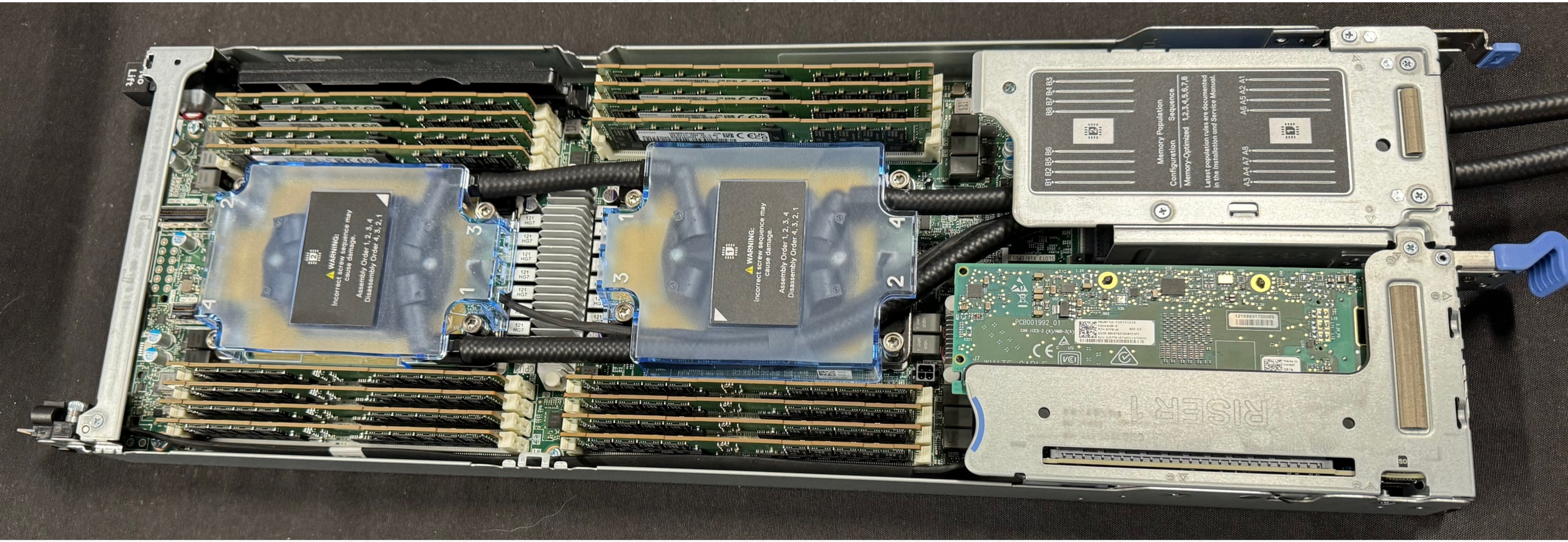
Rackmount servers

Rackmount servers

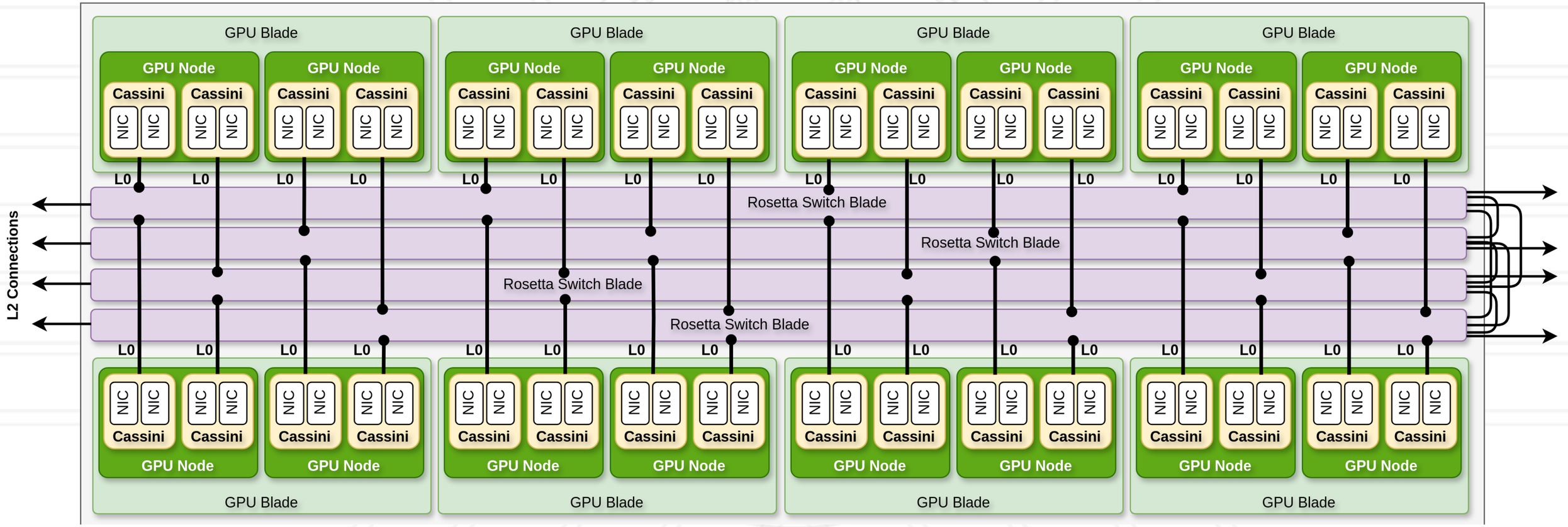
Zaratan racks / cabinets



Zaratan CPU compute node



A realistic cluster

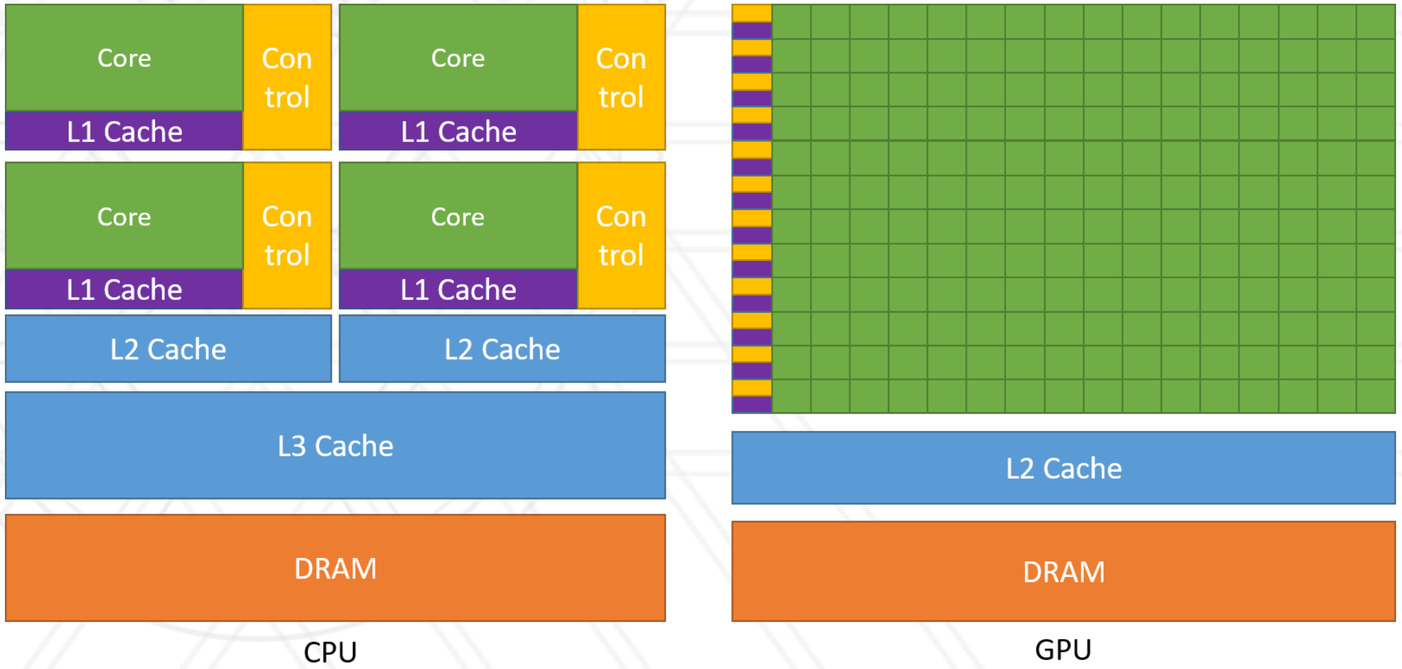


GPGPUs

- Originally developed to handle computation related to graphics processing
- Also found to be useful for scientific computing and AI
- Hence the name: General Purpose Graphics Processing Unit

GPGPU Hardware

- Higher instruction throughput
- Hide memory access latencies with computation



https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

More terms and definitions

- Model: an overloaded term
- Network architecture: also an overloaded term
- Weights / parameters: floating point numbers that represent the model
  - Used to denote the size of the model

More terms and definitions

- Language model: trained on natural language data for natural language tasks
  - LLMs, Transformer models
- Image model: trained on image data for tasks dealing with images
  - CNNs, ViTs, diffusion models
- Graph neural networks: trained on graph data

UNIVERSITY OF
MARYLAND