Introduction to High-Performance Computing

Dr. Axel Kohlmeyer

Research Professor, Department of Chemistry
Associate Director, ICMS
College of Science and Technology
Temple University, Philadelphia
axel.kohlmeyer@temple.edu

Why use Computers in Science?

- Use complex theories without a closed solution: solve equations or problems that can only be solved numerically, i.e. by inserting numbers into expressions and analyzing the results
- Do “impossible” experiments: study (virtual) experiments, where the boundary conditions are inaccessible or not controllable
- Benchmark correctness of models and theories: the better a model/theory reproduces known experimental results, the better its predictions

What is High-Performance Computing (HPC)?

- Definition depends on individual person:
  “HPC is when I care how fast I get an answer”
- Thus HPC can happen on:
  - A workstation, desktop, laptop
  - A smartphone
  - A supercomputer
  - A Linux/MacOS/Windows/... cluster
  - A grid or a cloud
- HPC also means High-Productivity Computing

Parallel Workstation

- Most desktops today are parallel workstations => multi-core processors (up to 128 cores)
- Running Linux OS (or macOS) allows programming like traditional Unix workstation Windows is different, but not much
- All processors have access to all memory
  - Uniform memory access (UMA):
    1 memory pool for all, same speed for all
  - Non-uniform memory access (NUMA):
    multiple pools, speed depends on “distance”

An HPC Cluster is...

- A cluster needs:
  - Several computers, often in special cases for easy mounting in a rack (one node ~= one mainboard)
  - One or more networks (interconnects) to access the nodes and for inter-node communication
  - Software that orchestrates communication between parallel processes on the nodes (e.g. MPI)
  - Software that reserves resources to individual users
- A cluster is: all of those components working together to form one big computer

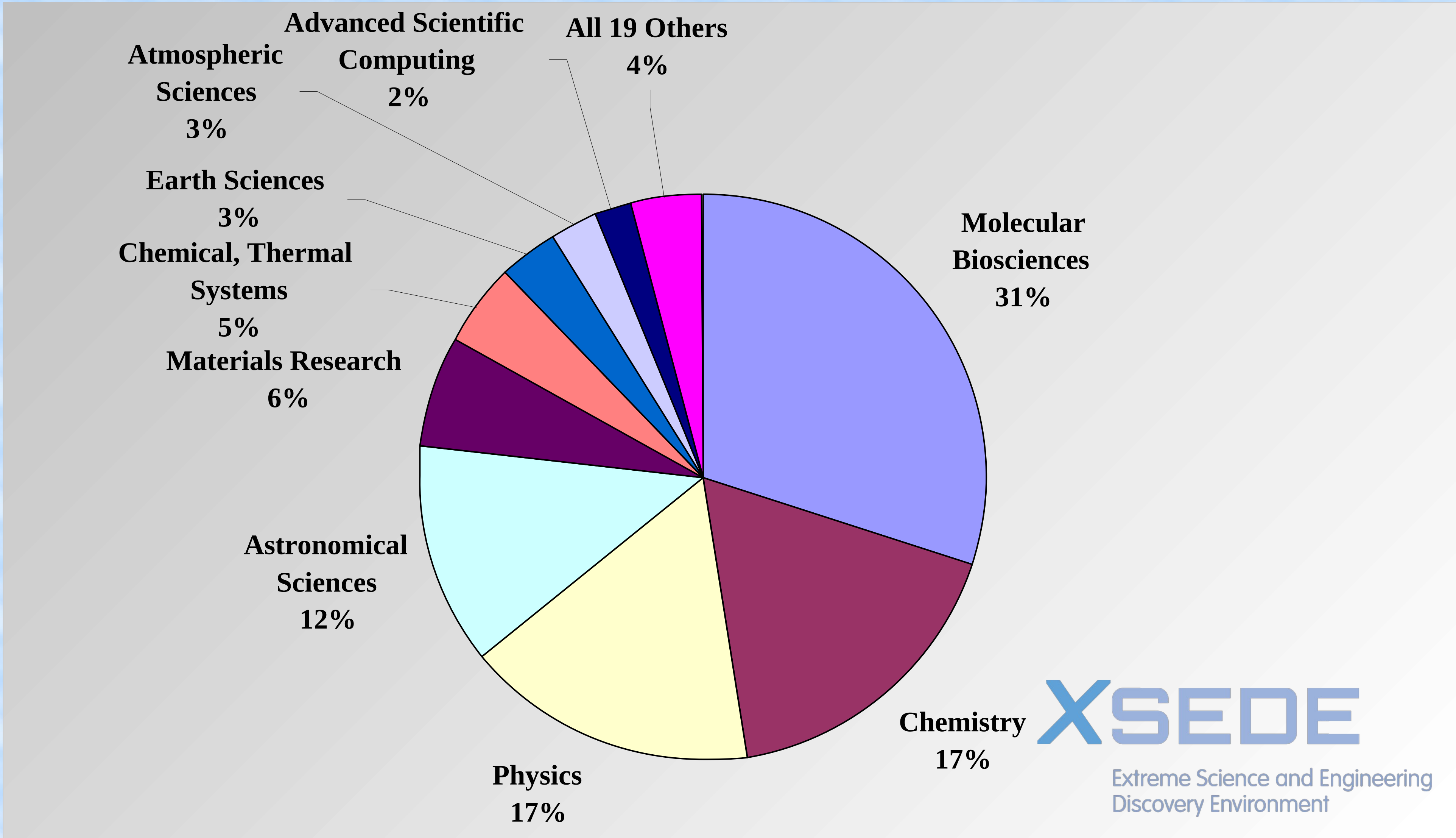
What is Supercomputing (SC)?

- The most visible manifestation of HPC (=> Top500 List)
- Is “super” due to large size, extreme technology
- Desktop vs. Supercomputer in 2019 (peak, DP):
  - Desktop processor (1 core): ~50 GigaFLOP/s
  - Tesla V100 GPU: >7 TeraFLOP/s
  - #1 supercomputer on Top500: >200 PetaFLOP/s
- Sustained vs. Peak: “K” 93%, “Summit” 74%, “Tianhe-2a” 61%, Cluster 65-90%

Why would HPC matter to you?

- Scientific computing is becoming more important in many research disciplines
- Problems become more complex, need teams of researchers with diverse expertise
  → complex SW packages with dependencies
- Scientific (HPC) application development often limited by lack of training
- More knowledge about HPC leads to more effective use of HPC resources and better interactions with colleagues

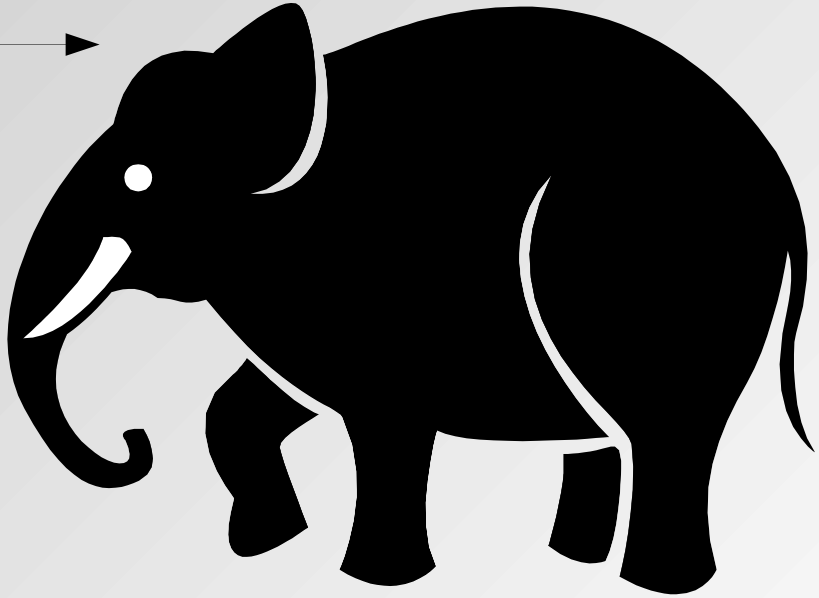
Research Disciplines in HPC

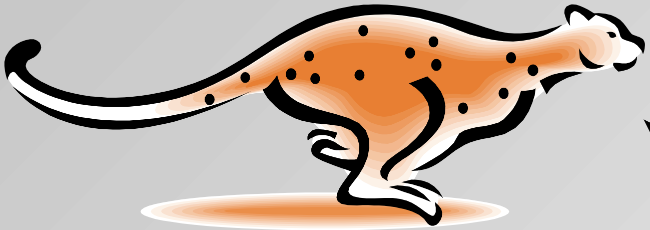


Extreme Science and Engineering Discovery Environment

Why Would I Care About HPC?

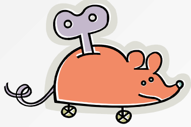
- My problem is big





- My problem is complex

- My computer is too small and too slow



- My software is not efficient and/or not parallel
  -> often scaling with system size the problem

HPC vs. Computer Science

- Most people in HPC are not computer scientists
- Software has to be correct first and (then) efficient; packages can be over 30 years “old”
- Technology is a mix of “high-end” & “stone age” (Extreme hardware, MPI, Fortran, C/C++)
- So what skills do I need to for HPC:
  - Common sense, cross-discipline perspective
  - Good understanding of calculus and (some) physics
  - Patience and creativity, ability to deal with “jargon”

HPC is a Pragmatic Discipline

- Raw performance is not always what matters:
  how long does it take me to get an answer?

- HPC is more like a craft than a science:
  => practical experience is most important
  => leveraging existing solutions is preferred
  over inventing new ones requiring rewrites
  => a good solution today is worth more than
  a better solution tomorrow
  => but a readable and maintainable solution
  is better than a complicated one

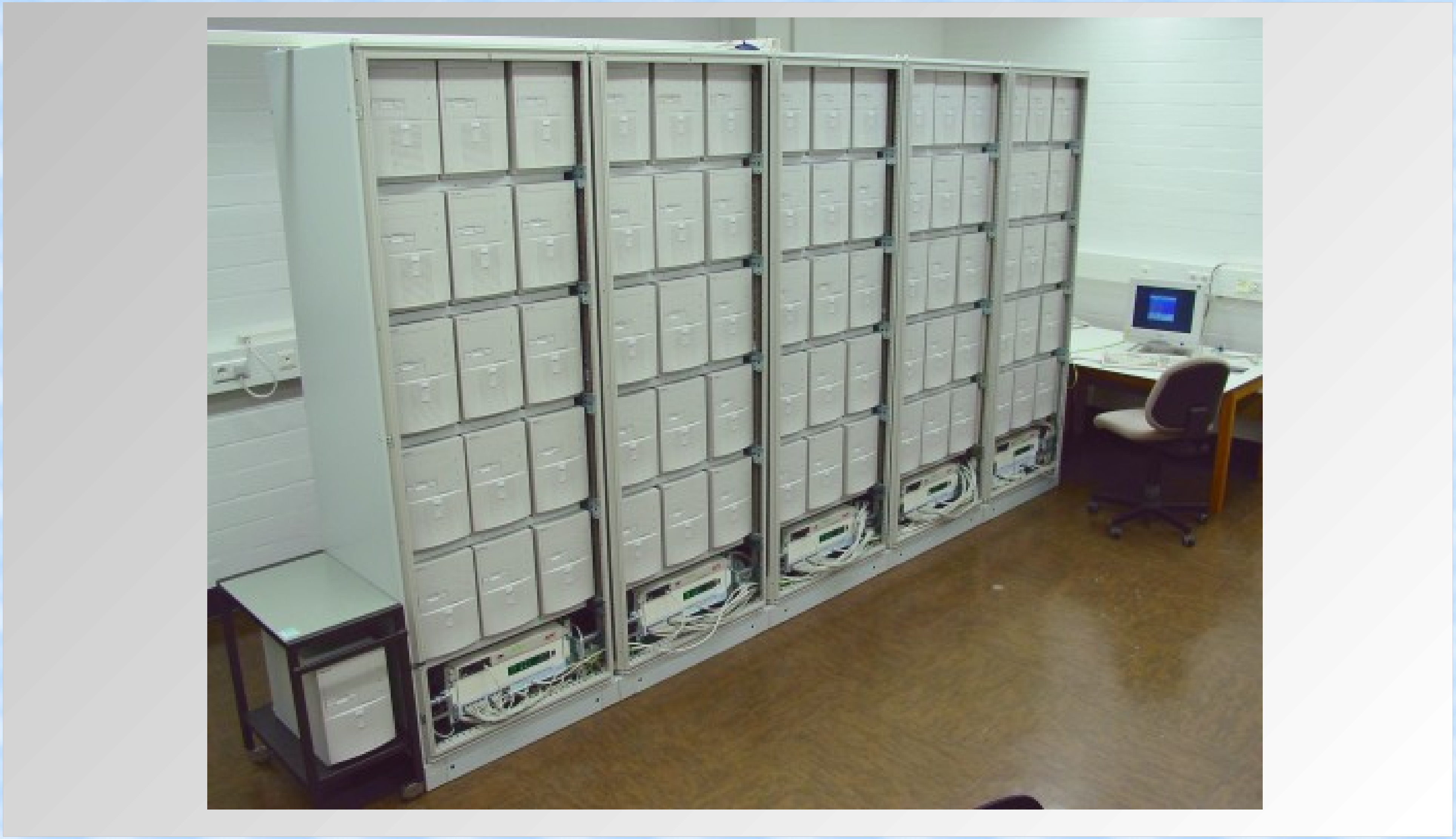
How to Get My Answers Faster?

- Work harder
  => get faster hardware (get more funding)
- Work smarter
  => use optimized algorithms (libraries!)
  => write faster code (adapt to match hardware)
  => trade performance for convenience
  (e.g. compiled program vs. script program)
- Delegate parts of the work
  => parallelize code, (cluster computing)
  => use accelerators (GPU CUDA, OpenCL etc)

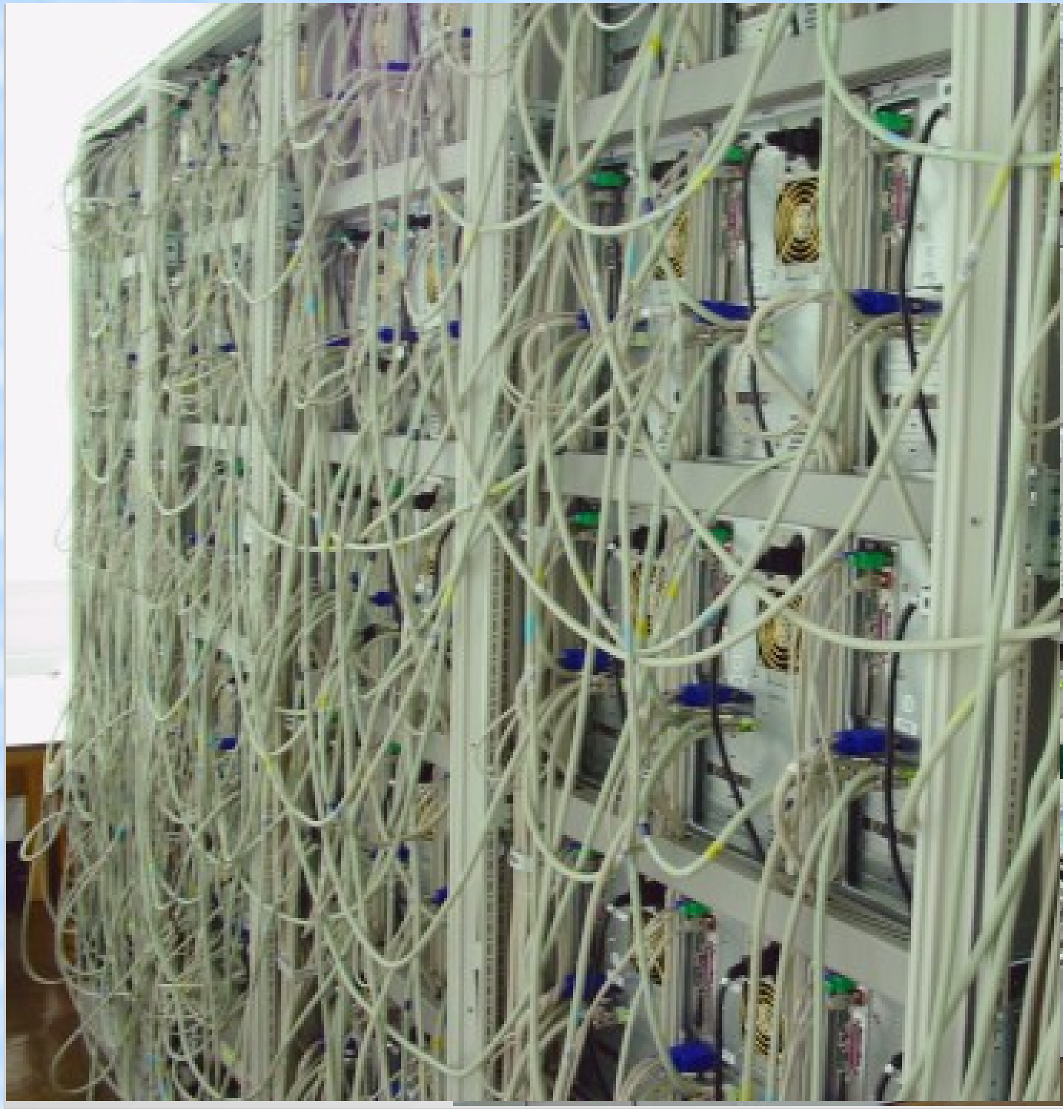
How Do We Measure Performance?

- For numerical operations: FLOP/s
  = Floating-Point Operations per second
- Theoretical maximum (**peak**) performance:
  clock rate x number of **double precision** addition and multiplications completed per clock
  => 2.5 Ghz x 8 FLOP/clock = 20 GigaFLOP/s
  => can never be reached (data load/store)
- Real (**sustained**) performance:
  => very application dependent
  => Top500 uses Linpack (linear algebra)

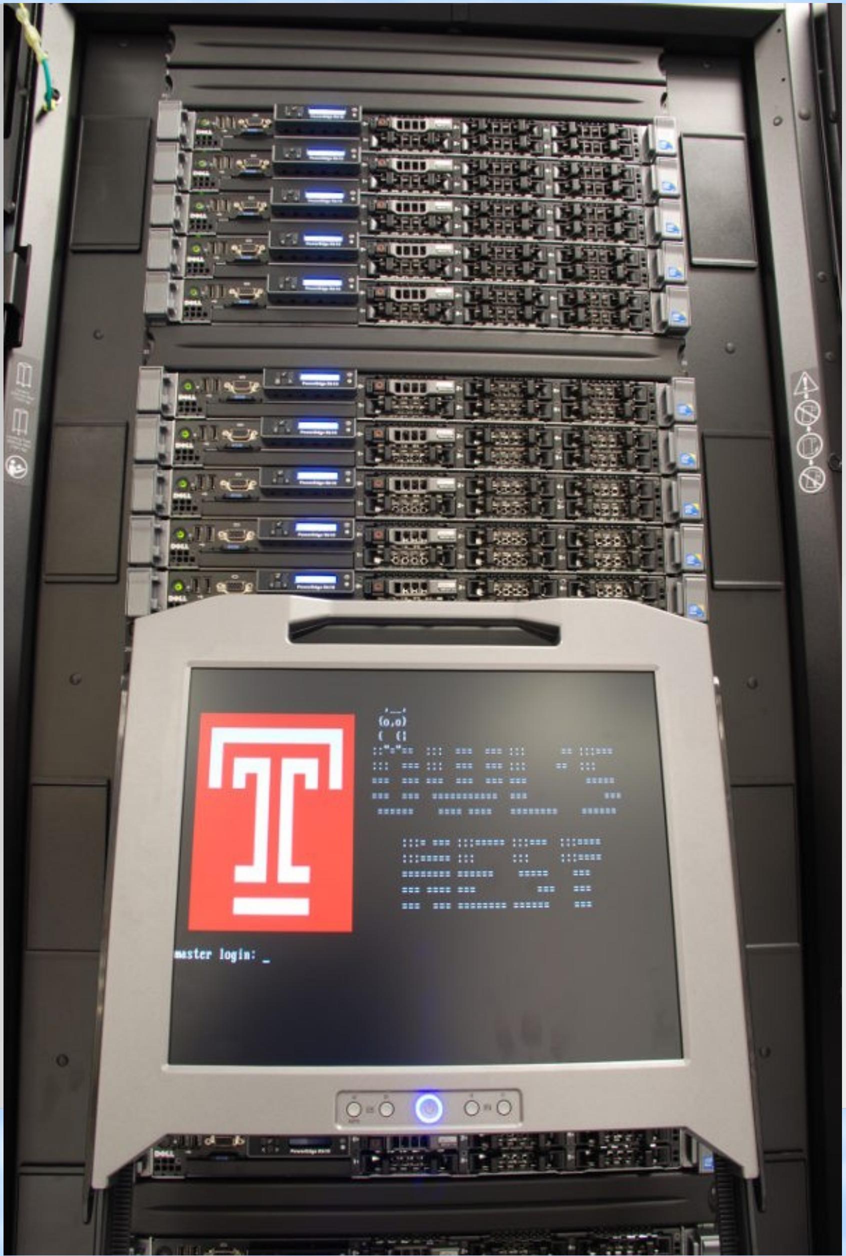
HPC Cluster in 2002 / The Good

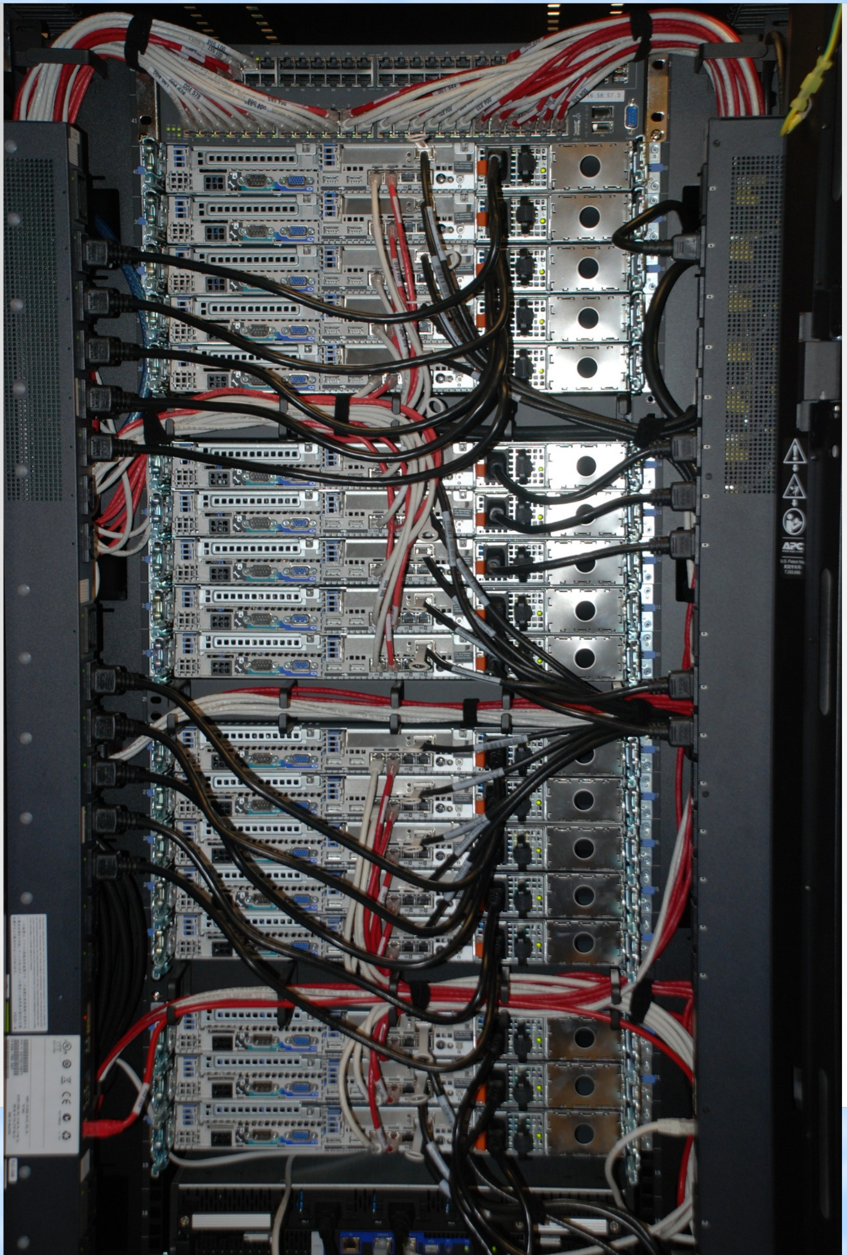


HPC Cluster in 2002 / The Bad



HPC Cluster in 2012





HPC Cluster in 2024

>250 nodes, >6500 cores, >2PB storage