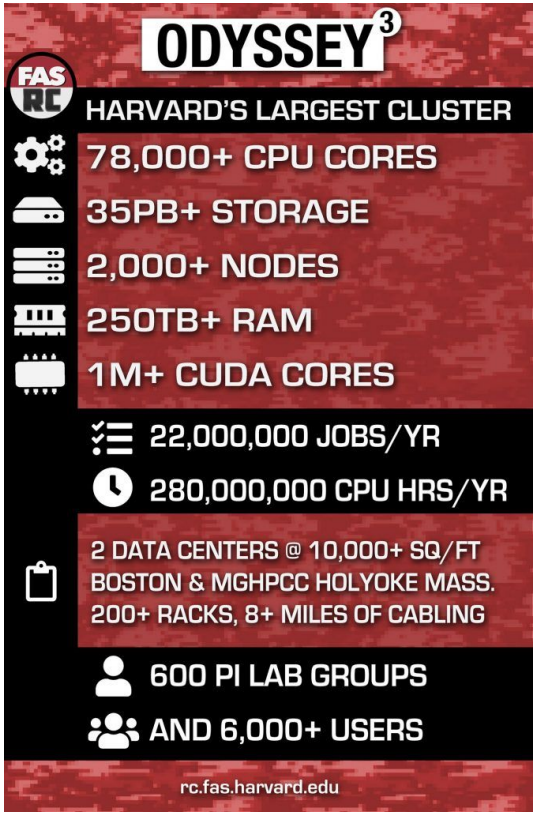
HARVARD UNIVERSITY

Introduction to Scientific Computing

What is Research Computing?

- Anything beyond basic IT needs that are required to accomplish research
  - High Performance Computing
  - Large Scale Storage
  - Scientific Software
  - Instrumentation
  - Exotic Architectures
- Faculty of Arts and Sciences (FAS) RC is the largest RC group at Harvard and run Odyssey



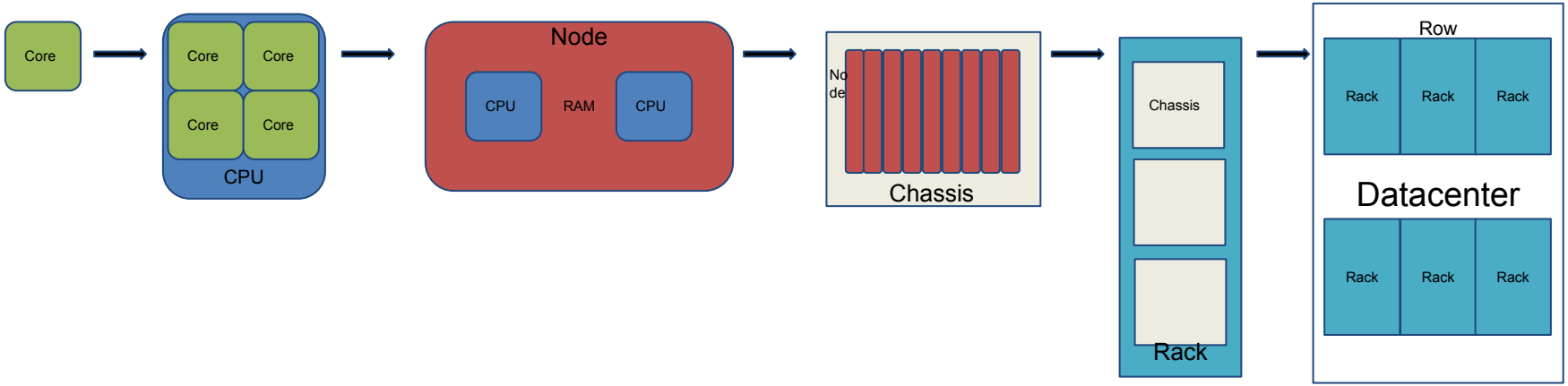
High Performance Computing

- Also known as HPC/Supercomputing
- Goal: Solve computationally challenging problems by pushing hardware to the limit and orchestrating thousands of nodes.
- Basic Components
  - Compute Cluster
  - Interconnect
  - Scheduler
  - Operating System (OS)

Compute Cluster

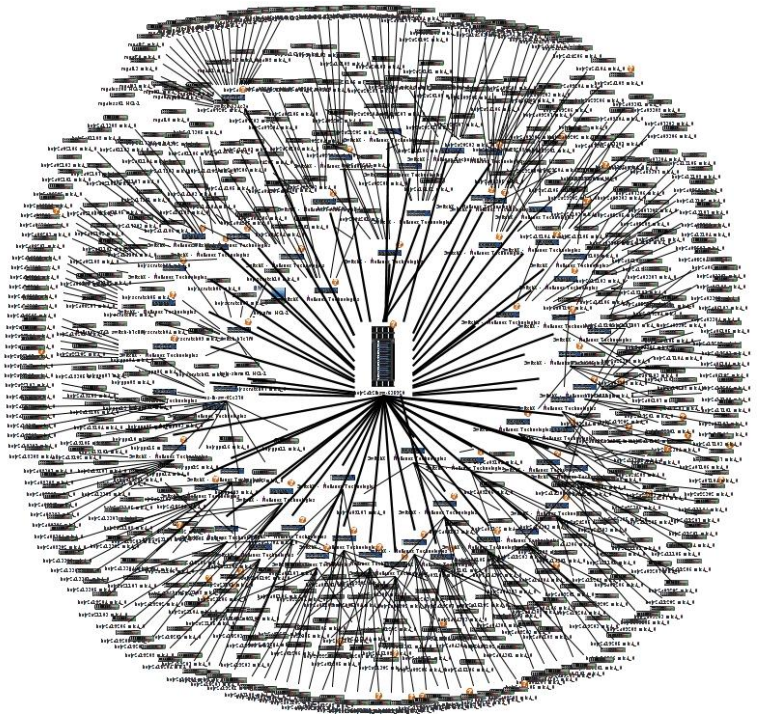
Building Blocks

- Core: Fundamental unit of compute includes its own cache
- Central Processing Unit (CPU): Made up of multiple Cores on a single die
- Random Access Memory (RAM): Memory shared by all the Cores on a Node.
- Node: A single motherboard with RAM and multiple CPU's
- Chassis: Holds nodes (also called blades) and shares power and cooling
- Rack: Holds Chassis along with storage and network switches
- Row: Multiple Racks in a single group
- Datacenter: Collection of Rows that is contained in a single room



Interconnect

- Ethernet
  - Typically 1 Gb/s but many new systems use 10/100 Gb/s
  - Used for normal maintenance operations.
- Infiniband
  - Low latency high bandwidth interconnect.
  - Current generation HDR (High Data Rate) is 200 Gb/s
  - Provides RDMA (Remote Direct Memory Access)
  - Used for computation and storage access

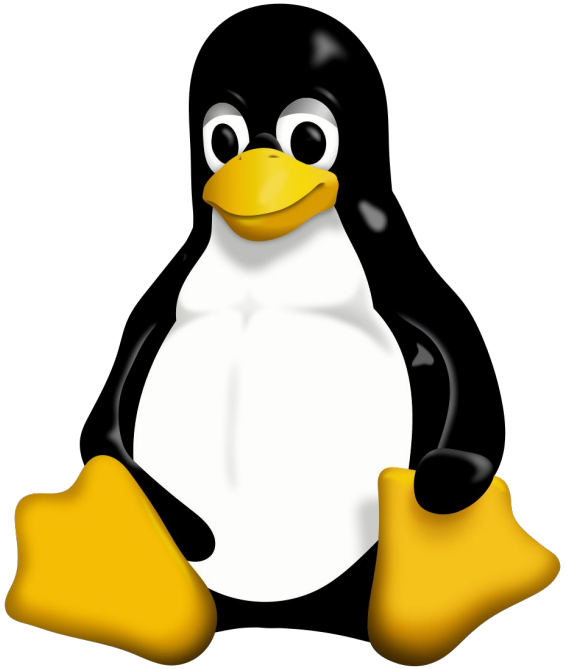


Scheduler

- The scheduler adjudicates who gets what part of the cluster and when
- Types of Schedulers
  - Slurm
  - SGE (Sun Grid Engine)
  - PBS (Portable Batch System): Moab, TORQUE
  - Condor

Operating System (OS)

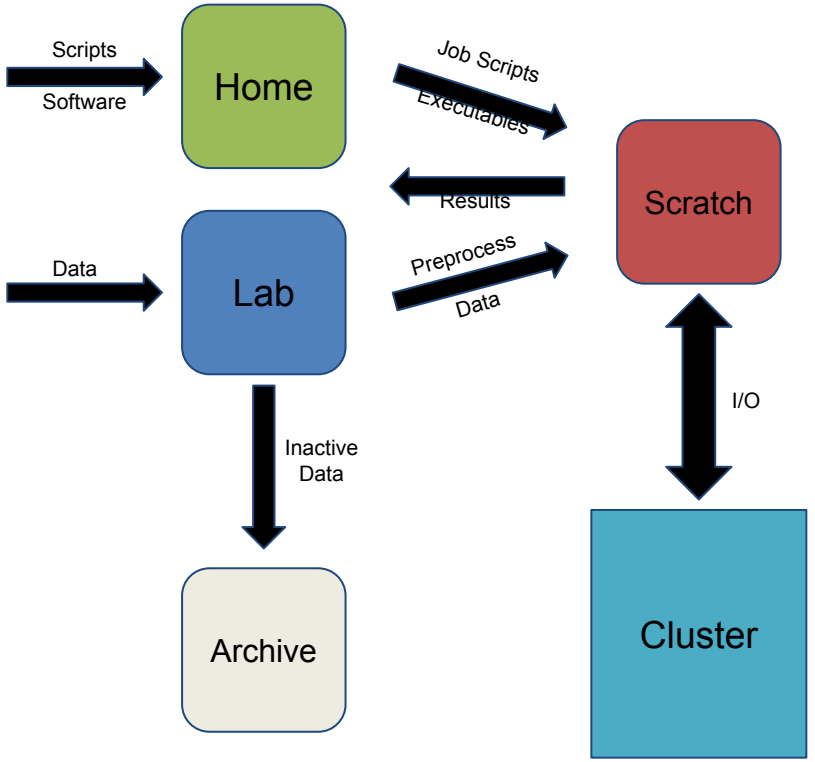
- The basic environment that the cluster runs on
- Types of OS's
  - CentOS/RedHat Enterprise Linux 7
  - Ubuntu
  - Windows
  - Mac
- Shell
  - bash
  - c-shell



Large Scale Storage

Classes

- Home Directories
- Lab Storage
- Scratch
- Archive



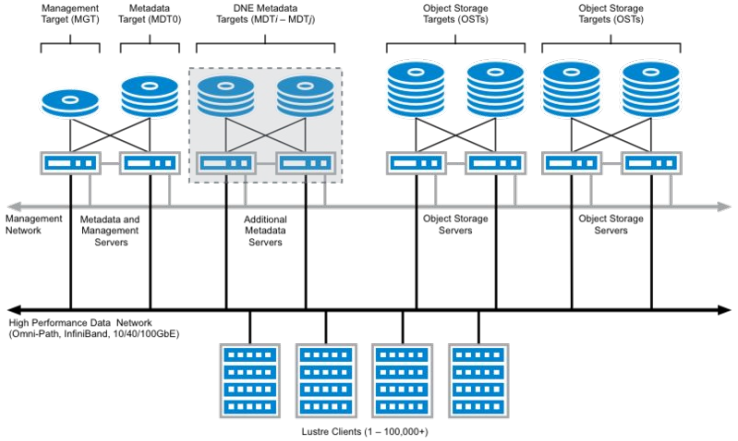
Storage Technology

Storage Devices

- Hard Drive
- Solid State Drive
- Intel Optane
- RAMFS/Tmpfs

Filesystem Types

- Network File System (NFS)
- Lustre
- Spectrum Scale (aka gpfs)
- CephFS
- ZFS
- Other filesystems: glustre, btrfs, moosefs
- Enterprise Storage Solution (Isilon)
- Cloud Storage: S3 standard
- Tape Library



Data Lifecycle

1. Data is gathered from instruments or generated by code.

2. Data is reduced in size by pre analysis to only those portions of the data that are important to the scientific question.

3. Data is analyzed and scientific results generated.

4. Data is cataloged and archived for future use

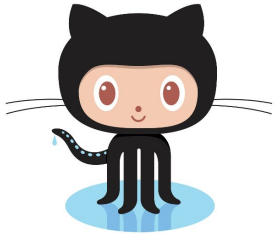
Note: Scripts that analyze the data and especially codes that generate Data are themselves data that should be archived for posterity and future use.

Scientific Software Classes

- Basic scripts to do rudimentary work
- Very sophisticated scripts that handle bulk analysis
- Code that runs instrumentation and push out data
- Numerical Simulation Codes
- Software Libraries

Version Control

- Allows tracking of changes and enables easy collaboration on codes.
- Defacto standard is git
- Defacto standard repository is github
- All code regardless of scale should be version controlled and backed up on at an external site.
  - It should also be documented



Modules and Containers

Modules

- Allow for a great variety of software to be installed without all of it being loaded for a user.
- The user selects only those modules they need
- Prevents the loading of conflicted libraries

Containers

- Allow for exotic and complex software stacks
- Gives greater flexibility to the user while preserving the ability of the administrator to maintain a modern OS and make changes.
- Defacto standards: Docker and Singularity

Software Best Practices

- Document your code both internally and externally
- If possible make OpenSource
- Pick the right language for your work
- Test your code regularly
  - Just because your code gives what appears to be the right answer does not mean it's actually working correctly.
- Be aware of the code's numerical limitations and quirks
- Use appropriate and optimized numerical methods
- Avoid black boxes
- Recall that the code is only as correct as the author that wrote it
- Be careful when scaling up to more cores or resources

Languages

Scripting

- Bash
- Perl
- Python

Statistics

- R
- Python
- Matlab

Numerical Computation

- Python
- Matlab
- Julia
- C
- Fortran

Parallel Computing

- OpenMP
- Message Passing Interface (MPI)
- CUDA

Parallel Computing

- Embarrassingly Parallel
- SIMD (Single Instruction Multiple Data)
- Thread Based Parallelism
- Rank Based Parallelism
- Hybrid

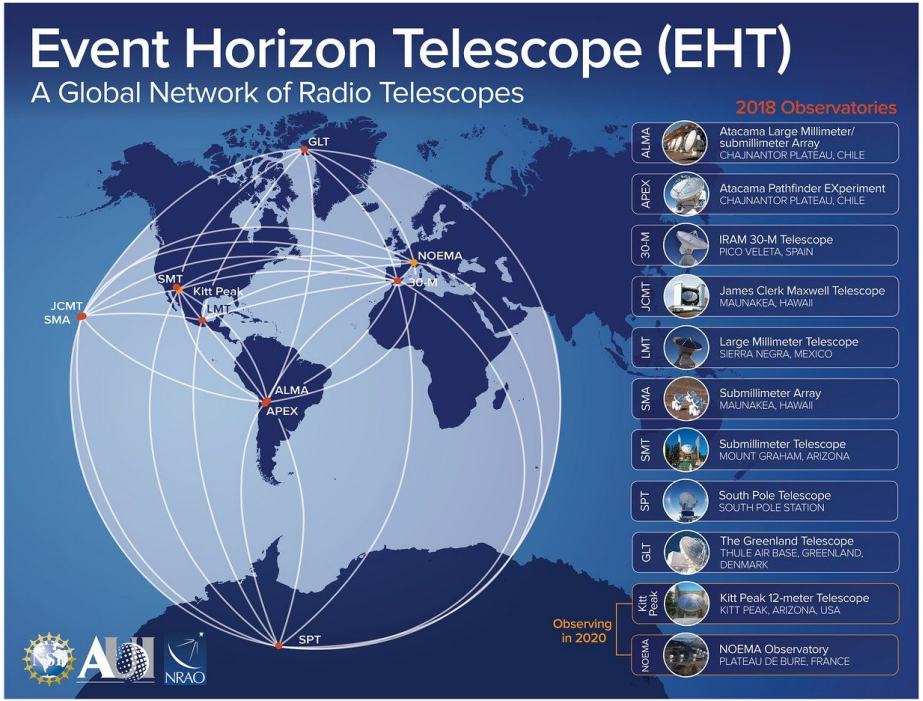
Instruments

Modern Instruments Can Push out Enormous Amounts of Data

- Telescopes
- Microscopes
- Internet of Things

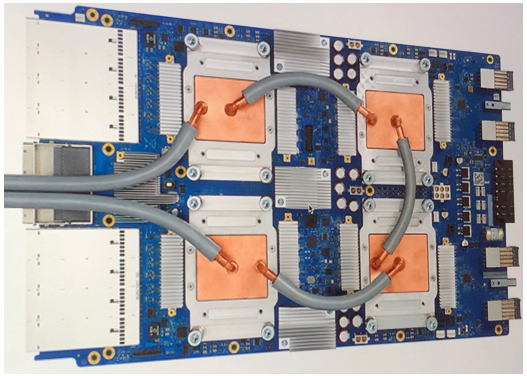
Pre-Analysis

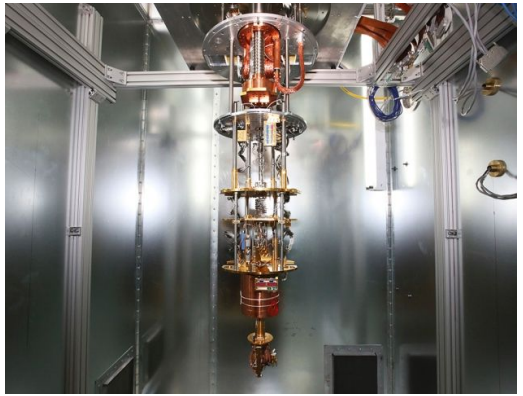
- Throw out data that cannot be used
- Preprocess data to screen out signal that you care about



Exotic Hardware

- Alternate Machine Learning Architectures
  - ASIC (Application Specific Integrated Circuit)
- Field Programmable Arrays (FPGA)
- Quantum Computing
- Intel Optane





Questions?

FASRC Office Hours 12-3p Wednesdays, 38 Oxford St.

Email: rchelp@rc.fas.harvard.edu