



SURF

Snellius File Systems

Hands-on data management in high-performance computing

Xavier Álvarez-Farré

High-Performance Computing and Visualization, SURF, Science Park 140, 1098XG
Amsterdam, The Netherlands

October 9, 2024

Adapted and updated by Rohit Goswami

| Introduction

Overview of HPC applications

HPC applications exploit the power of supercomputers or high-performance clusters with the goal of solving complex problems. They perform intensive **computations**, coordinate **communications** between parallel processes, and manage **input/output** operations on a high-performance file system.

Overview of HPC applications

HPC applications exploit the power of supercomputers or high-performance clusters with the goal of solving complex problems. They perform intensive **computations**, coordinate **communications** between parallel processes, and manage **input/output** operations on a high-performance file system.

Computations

Extensive numerical operations on a specific region of interest (e.g., mesh elements, particles, or even words in a text document).

- Ideal speedup is P for P processors at fixed total work; dependencies and imbalance reduce it.

Overview of HPC applications

HPC applications exploit the power of supercomputers or high-performance clusters with the goal of solving complex problems. They perform intensive **computations**, coordinate **communications** between parallel processes, and manage **input/output** operations on a high-performance file system.

Computations

Extensive numerical operations on a specific region of interest (e.g., mesh elements, particles, or even words in a text document).

- Ideal speedup is P for P processors at fixed total work; dependencies and imbalance reduce it.

Communications

Coordination and data exchanges between parallel processes to produce the final results.

- Cost depends on topology, message sizes and synchronization; more workers can add overhead.

Overview of HPC applications

HPC applications exploit the power of supercomputers or high-performance clusters with the goal of solving complex problems. They perform intensive **computations**, coordinate **communications** between parallel processes, and manage **input/output** operations on a high-performance file system.

Computations

Extensive numerical operations on a specific region of interest (e.g., mesh elements, particles, or even words in a text document).

- Ideal speedup is P for P processors at fixed total work; dependencies and imbalance reduce it.

Communications

Coordination and data exchanges between parallel processes to produce the final results.

- Cost depends on topology, message sizes and synchronization; more workers can add overhead.

I/O operations

Interactions with the file system to read input data, write output results, and perform intermediate data storage.

- Throughput depends on the file system and access pattern; more processes can increase contention.

Overview of I/O operations

Input operations

- **Initialization data:** grids, genomic sequences, climate data, or text.
 - Massive files that require a high level of precision.
 - Involve parallel input operations.

Overview of I/O operations

Input operations

- **Initialization data:** grids, genomic sequences, climate data, or text.
 - Massive files that require a high level of precision.
 - Involve parallel input operations.
- **Job parameters:** small configuration files that coordinate the processes.

Overview of I/O operations

Input operations

- **Initialization data:** grids, genomic sequences, climate data, or text.
 - Massive files that require a high level of precision.
 - Involve parallel input operations.
- **Job parameters:** small configuration files that coordinate the processes.

Output operations

- **Restart files** or checkpoints enable job continuation in the event of an interruption.
 - Large files that require full precision of simulation data.
 - Involve periodic parallel output operations.

Overview of I/O operations

Input operations

- **Initialization data:** grids, genomic sequences, climate data, or text.
 - Massive files that require a high level of precision.
 - Involve parallel input operations.
- **Job parameters:** small configuration files that coordinate the processes.

Output operations

- **Restart files** or checkpoints enable job continuation in the event of an interruption.
 - Large files that require full precision of simulation data.
 - Involve periodic parallel output operations.
- **Analysis data** for visualization, post-processing and in-depth examination.
 - May be possible to use reduced precision or store a subset of the full-resolution data.
 - Involve periodic parallel output operations.

Overview of I/O operations

Input operations

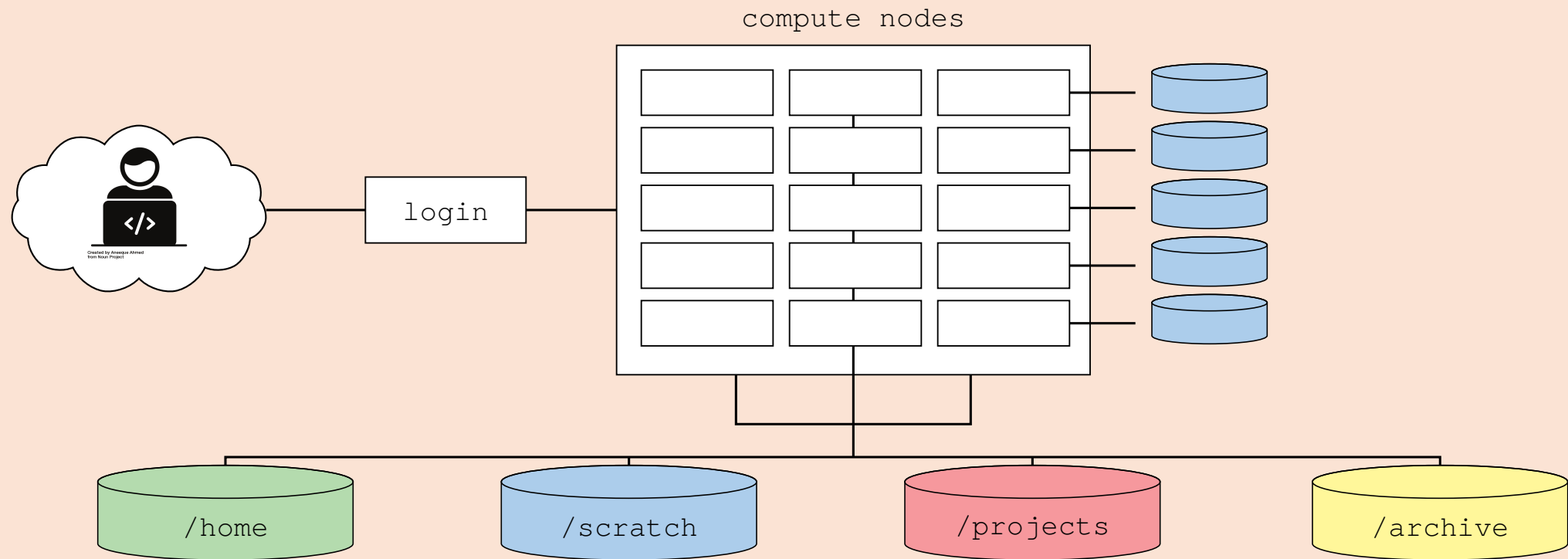
- **Initialization data:** grids, genomic sequences, climate data, or text.
 - Massive files that require a high level of precision.
 - Involve parallel input operations.
- **Job parameters:** small configuration files that coordinate the processes.

Output operations

- **Restart files** or checkpoints enable job continuation in the event of an interruption.
 - Large files that require full precision of simulation data.
 - Involve periodic parallel output operations.
- **Analysis data** for visualization, post-processing and in-depth examination.
 - May be possible to use reduced precision or store a subset of the full-resolution data.
 - Involve periodic parallel output operations.
- **Job monitoring:** progress and performance logs, typically plain text from process 0.

Overview of Snellius file-systems

Data management in HPC applications covers **diverse tasks**, each with its own nature and specific requirements. At the same time, HPC systems are equipped with **distinct file systems** to meet these specific needs.



Overview of Snellius file-systems

Data management in HPC applications covers **diverse tasks**, each with its own nature and specific requirements. At the same time, HPC systems are equipped with **distinct file systems** to meet these specific needs.

File system	Speed	Size	Backed-up	Expiration
/home	Normal	200 GiB	Yes	Project end + 15 weeks
/scratch-shared	Fast	8 TiB total	No	14 days
/scratch-local	Fast		No	6 days
/scratch-node	Very fast		No	End of job
/projects	Fast	On demand	No	Project end
/archive	Very slow	On demand	Yes	Project end

| Hands-on

Objectives of the hands-on session

1. **Exploring file systems:**
 - Know the file systems of the table (home, scratch, project, archive)
 - Move between your home and scratch-shared and check the current location
2. **File management:**
 - Create, view, and edit files using basic terminal commands
 - Move, rename, and copy files between different file systems
3. **Directory management:**
 - Create and delete directories
 - Change into directories and manage file hierarchy
4. **File permissions:**
 - Check and modify file access permissions
5. **Disk usage monitoring:**
 - Monitor storage usage and available space across file systems

1. Exploring file systems

Objective

Learn how to move between your home and your scratch-shared directory, the two file systems every account has, and check where you are.

Tasks

- Start in the home directory and use `pwd` to check your location.
- Create your scratch directory: `mkdir -p /scratch-shared/$USER`.
- Explore it: `cd /scratch-shared/$USER` and `ls`.
- Return to your home directory using `cd`.

Key commands

- `pwd` - Display the current directory
- `ls` - List the contents of a directory
- `cd <path>` - Change directories

Expected output

- `pwd` prints `/home/<user>` at the start and `/scratch-shared/<user>` after the `cd`; `ls` of the new directory prints nothing.

2. File management

Objective

Learn how to create, view, move, and rename files across different file systems in an HPC environment.

Tasks

- Create a file in the home directory using `touch`.
- Edit the file with `nano` or `vim`, and view its contents with `cat`.
- Copy the file to the `scratch` directory using `cp`.
- Rename or move the file using `mv`.

Key commands

- `touch <fname>` - Create a new empty file
- `nano <fname>` or `vim <fname>` - Edit a file
- `cat <fname>` - Display the contents of a file
- `cp <src> <tgt>` - Copy files between directories
- `mv <src> <tgt>` - Move or rename files

Expected output

- `cat` prints the line you typed; `ls /scratch-shared/$USER` lists the copy; after `mv`, `ls` shows the new name and not the old one.

3. Directory management

Objective

Learn how to create, enter, and remove directories within the HPC file system.

Tasks

- Create a new directory in the home directory using `mkdir`.
- Change into the new directory with `cd`.
- List its contents with `ls`, then return to the home directory.
- Remove the directory using `rmdir`.

Key commands

- `mkdir <dtype>` - create a new directory
- `cd <dtype>` - enter a directory
- `rmdir <dtype>` - remove an empty directory
- `ls [dtype]` - list directory contents

Expected output

- `ls` inside the new directory prints nothing; after `rmdir`, `ls ~` no longer lists it.

4. File permissions

Objective

Learn how to view and modify file access permissions to control who can read, write, or execute files.

Explanation

- Permissions are shown as `rwXrwxrwx` (for owner, group, others).
- `r` - read (4), `w` - write (2), `x` - execute (1).
- Numeric format: `chmod 755` means owner can `rwX`, group and others can `r - X`.

Tasks

- Use `ls -l` to check the permissions of a file.
- Modify the file's permissions to allow read access for everyone using `chmod 644`.
- Verify the permission changes with `ls -l`.

Key commands

- `ls -l` - view file permissions
- `chmod` - change file permissions

Expected output

- `ls -l` shows `-rw-r--r--` for the file after `chmod 644`.

5. Disk usage monitoring

Objective

Learn how to monitor storage usage and check available space across different file systems.

Tasks

- Use `df -h` to view overall disk usage and available space in the home, scratch, and project file systems.
- Check the size of a specific directory in your home using `du -sh`.
- Identify where you are using the most space.

Key commands

- `df -h` - view disk space usage for all mounted file systems
- `du -sh` - check the size of a specific directory

Expected output

- `df -h /home /scratch-shared` prints one line per file system with its size, use and free space; `du -sh ~` prints one size for your home.