Grid, HPC, Cloud & Containers for Computational Biologists

RECETOX Bioinformatics

September 24, 2025

Motivation

- Local laptops/workstations are insufficient for modern bioinformatics:
  - Large datasets (100s GB to TBs)
  - Long runtimes (hours to days)
  - High memory / many CPUs required
- Solution: distributed infrastructures — Grid, HPC, Cloud.

What is Grid Computing?

- Federation of compute & storage across multiple institutions.
- Users share resources through a unified access system.
- Scheduler manages who runs when.
- Best for many independent (“embarrassingly parallel”) jobs.

What is HPC?

- Centralized supercomputers with thousands of cores.
- High-speed interconnects (InfiniBand), shared parallel storage.
- Best for tightly coupled workloads (MPI, simulations).
- One big cluster, one scheduler.

What is Cloud Computing?

- Virtualized, on-demand infrastructure (AWS, Azure, GCP, OpenStack).
- Elastic scaling: spin up VMs/containers when needed.
- Good for bursty workloads, prototyping, custom environments.
- Pay-as-you-go vs. academic grids/HPC which are free but shared.

Comparison

Feature	Grid	HPC	Cloud
Resources	Distributed, federated	Centralized supercomputer	Virtualized, elastic
Access	National/shared	Institutional login	Commercial account
Best for	Many small jobs	Large parallel jobs	Flexible scaling

MetaCentrum (CZ national grid)

- Operated by CESNET, connects universities/institutes.
- Provides CPUs, GPUs, large memory nodes, storage.
- Scheduler: PBS Pro.
- Access via SSH frontend: `ssh username@login.metacentrum.cz`

Frontend nodes

- Entry point via SSH.
- Light work only: editing, compiling, job submission.
- Not for heavy computation.

Interactive jobs

```
qsub -I -l select=1:ncpus=4:mem=8gb -l walltime=02:00:00
```

- Opens a shell directly on a compute node.
- Useful for testing and debugging.

Batch jobs

Example script `job.sh`:

```
#!/bin/bash
#PBS -l select=1:ncpus=8:mem=16gb
#PBS -l walltime=04:00:00
#PBS -N fastqc_job

module add fastqc
fastqc data/*.fastq.gz -o results/
```

Submit with:

```
qsub job.sh
```

Scheduler basics

- Jobs wait in queue until resources are free.
- Resource requests: CPUs, memory, walltime.
- Commands:
  - `qsub job.sh` — submit
  - `qstat` — status
  - `qdel JOBID` — cancel

Storage & environment

- Home directory: configs, small files.
- Project directories: `/storage/...` for large data.
- Module system to load software:

```
module avail
module add fastqc
```

- Use conda or containers if software not available.

Best practices

- Start small with interactive jobs.
- Scale up with batch jobs for real workloads.
- Keep raw data read-only, write results elsewhere.
- Monitor quotas and walltime limits.
- Document job scripts for reproducibility.

Practical exercises

1. Connect via SSH to MetaCentrum frontend.
2. Run an interactive job with 2 CPUs, 2 GB RAM, 30 min walltime.
3. Submit a batch script running `fastqc`.
4. Check status with `qstat`.
5. Explore software with `module avail`.

Containers in research computing

- Containers = portable, reproducible environments.
- Package OS libraries + software in one unit.
- Ensure analyses run the same across systems.
- Popular: Docker (workstations/cloud), Singularity (HPC).

Why Singularity?

- Docker needs root privileges — unsafe on HPC.
- Singularity runs containers without root, HPC-friendly.
- Integrates with job schedulers.
- Works like running any other program.

Using Singularity

```
# Load module
module add singularity

# Run from DockerHub
singularity exec docker://biocontainers/fastqc fastqc --help

# Run from a local .sif image
singularity exec myenv.sif bwa mem ref.fasta reads.fq > aln.sam

# Build an image (on a machine with Docker)
singularity build myenv.sif docker://ubuntu:22.04
```

Best practices with containers

- Use community images (e.g. BioContainers).
- Keep definition files under version control.
- Store containers in project storage for reuse.
- Combine with batch jobs for reproducible pipelines.

Summary

- Grid: federated resources (MetaCentrum).
- HPC: centralized supercomputers.
- Cloud: elastic, pay-as-you-go.
- Jobs: interactive vs. batch.
- Scheduler: request resources, monitor jobs.
- Containers: reproducible environments.
- Singularity: HPC-safe alternative to Docker.