VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

INTRODUCTION TO HIGH PERFORMANCE COMPUTING

PART 2
HPC @ IT4INNOVATIONS
ACCESSING AND USING IT4I CLUSTERS

Jakub Beránek

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research, Development and Education

MINISTRY OF EDUCATION, YOUTH AND SPORTS

Based on materials of Branislav Jansík, IT4Innovations

USING IT4I CLUSTERS

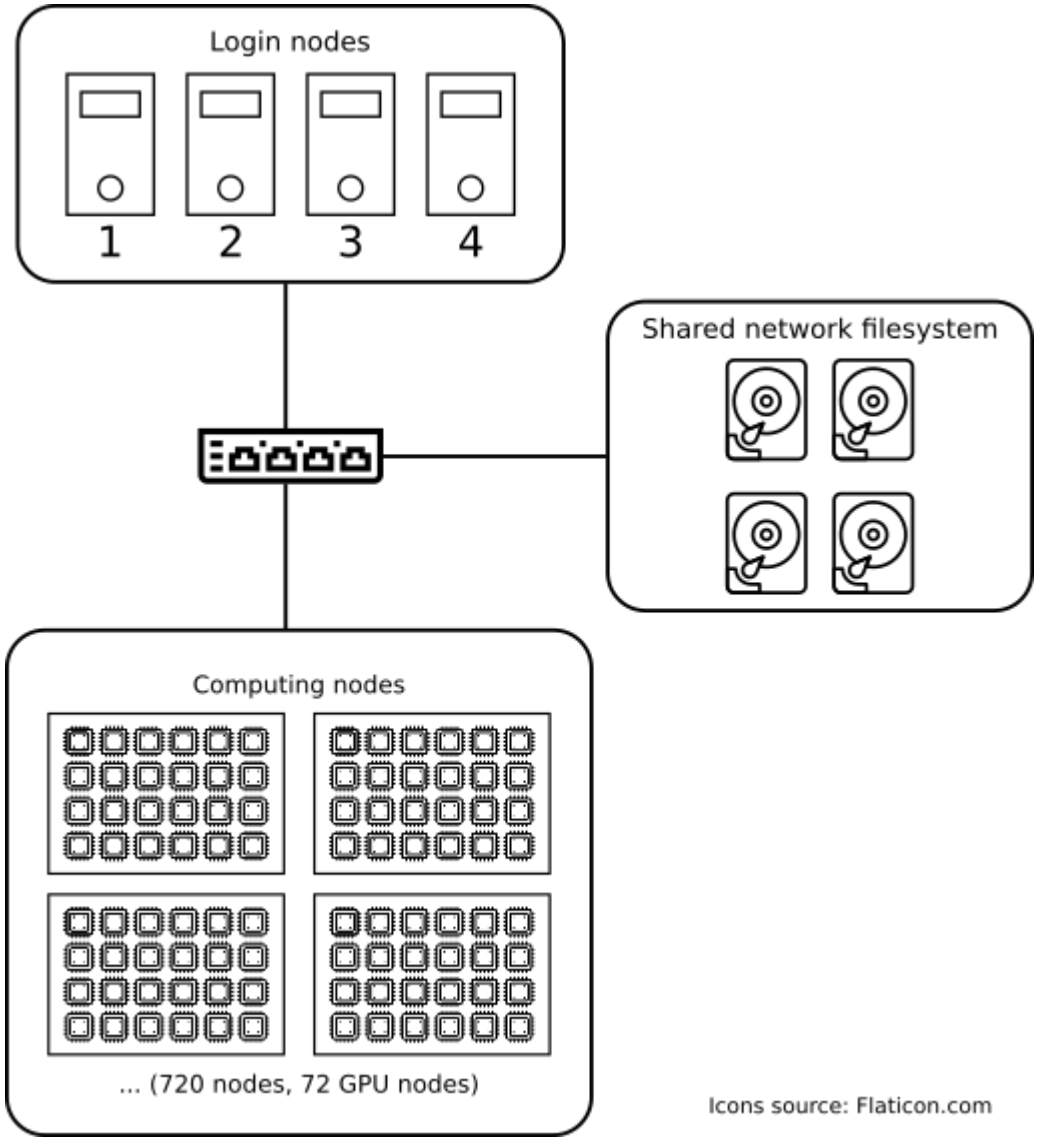
- Access the cluster
- Transfer data to the shared filesystem
- Prepare your program and its dependencies
- Run your program on the cluster

We will use Karolina, but the approach is very similar for other IT4I clusters (Barbora)

You can find more complete information in our documentation.

KAROLINA CLUSTER

- Login nodes
  - Program preparation
  - Job submission
- Compute nodes
  - Job execution
- Shared filesystem
  - Code
  - Job inputs and outputs
  - Shared between login and compute nodes



Icons source: Flaticon.com

OPERATING SYSTEM

- IT4I clusters are Linux-based systems (CentOS)
  - Basic Linux command line knowledge is required

Command	Description
ls	List files in a directory
cd <directory>	Change current directory
cat <file>	Display contents of a file
mkdir <name>	Create a directory
rm <path>	Delete a file or a directory

  - You can find basic Linux command line reference e.g. here.
- Some virtualization support is provided (QEMU, Windows)

ACCESSING THE CLUSTER

To use Karolina, you must first connect to one of its login nodes

```
# Set permissions for SSH key (execute before first login)
[home:~]$ chmod 600 <path-to-ssh-key>
# Connect to a login node
[home:~]$ ssh -i <path-to-ssh-key> <username>@karolina.it4i.cz
# Now you're connected to a Karolina login node
[<username>@login1.karolina]$
```

- You can use login nodes to
  - Inspect and manage data on the shared filesystem ✓
  - Compile your programs and their dependencies ✓
  - Manage computations on the cluster ✓
- DO NOT execute long-running computations on the login nodes ✗
- Login nodes are round-robin, you can select a specific node (login1.karolina.it4i.cz)

» You can simplify the SSH command with an SSH config file

GUI ACCESS

If you prefer to use a GUI client, you have two options

- X forwarding
  - Open individual X windows on your PC
  - `$ ssh –X karolina.it4i.cz`
- VNC
  - Full GUI environment on the cluster
  1. Select a VNC port P (here we use 55)
     - Must be unique per login node
  2. Connect to a login node with SSH tunneling on port 5900 + P
     - `$ ssh -L5955:localhost:5955 karolina.it4i.cz`
  3. Run vncpasswd
  4. Run vncserver :55
  5. Connect to VNC on port :55 on your local machine
     - `$ vncviewer localhost:5955`

More information can be found here.

RUNNING YOUR PROGRAM ON THE CLUSTER

1. Move your code and computation inputs to the shared filesystem
2. Build and prepare your application
3. Describe your computation and put it into a queue

TRANSFERRING DATA TO SHARED FILESYSTEM

Karolina uses a network filesystem shared by all compute and login nodes

- You can write a file on a login node and then read/overwrite it from a compute node

- Connect to a login node and download data from the internet (`git clone`, `wget`, ...)
- Transfer data from your local computer using SCP

```
# Copy "file.txt" to <home-dir>/files on Karolina shared disk
[home:~]$ scp -i <path-to-ssh-key> file.txt <username>karolina.it4i.cz:files
```

- Mount the shared filesystem on your local computer

```
# install sshfs
[home:~] $ sudo apt install sshfs
# mount the Salomon shared filesystem to a folder on your computer
[home:~] $ sudo mkdir /mnt/salomon
[home:~] $ sudo sshfs -i <path-to-ssh-key> <username>@salomon.it4i.cz: /mnt/salomon
# now /mnt/salomon points to your home directory at the Salomon shared filesystem
[home:~] $ cp file.txt /mnt/salomon/files-dir
```

WHERE TO PUT DATA?

- HOME workspace (NFS)
  - Located at ~ (your home directory)
  - Limited size (~25 GiB), quite slow (2-3 GiB/s), backed up
  - Use for config files, build artifacts, source code repositories
- PROJECT workspace (NFS)
  - Very large (~15 PiB), rather slow (40 GiB/s)
  - Shared between clusters
  - Divided into three parts (`/mnt/proj1`, `/mnt/proj2`, `/mnt/proj3`)
    - Each project has its own directory (deleted after project ends)
    - Find your project location with `$ it4i-get-project-dir <project-id>`
  - Central storage for all project data, use for important/large project data
- SCRATCH workspace (Lustre)
  - Located at *`scratch/project/<project-id>`*
  - Large (~20 TiB), very fast (1 TiB/s), no backups
  - Use for reading job inputs and writing job results
    - Set working directory of jobs to a scratch directory
    - Copy results to HOME or PROJECT after the job ends
  - **Files are deleted after 90 days of inactivity!**

ADDITIONAL STORAGE OPTIONS

- RAMDISK (Barbora)
  - `/tmp`, `/lscratch`, `/ramdisk`
  - RAM disk (filesystem backed by memory), for I/O intensive operations
  - Available only during a job
- CESNET – archiving large amounts of data, more information here

More information about storage at Karolina can be found here
Storage details vary among the clusters, check documentation for your cluster

MORE STORAGE INFORMATION

- Filesystems of individual clusters are not directly shared
  - Clusters are connected via a network, e.g. you can `$ ssh barbora` from Karolina
- Watch storage limits
  - `$ it4ifsusage`
  - https://scs.it4i.cz -> Agendas -> User

Quota Status

Cluster	File System	Space used	Space limit	Entries	Entries limit	Last Update
Anselm	/scratch	0 Bytes	93.13 TB	0	10 Million	2020-12-04 13:55
Anselm	/home	2.828 GB	238.4 GB	51.1 Thousand	500 Thousand	2020-12-04 13:55
Barbora	/home	7.221 GB	23.84 GB	44.1 Thousand	500 Thousand	2020-12-04 14:50
Barbora	/scratch	477.6 GB	9.313 TB	413 Thousand	10 Million	2020-12-04 14:50
Salomon	/home	153.7 GB	238.4 GB	456 Thousand	500 Thousand	2020-12-04 14:50
Salomon	/scratch/temp	0 Bytes	N/A	0	N/A	2020-12-02 07:40
Salomon	/scratch/work	237.8 GB	N/A	55.6 Thousand	N/A	2020-12-02 07:40
Salomon	/scratch	238 GB	93.13 TB	55.6 Thousand	10 Million	2020-12-04 14:50

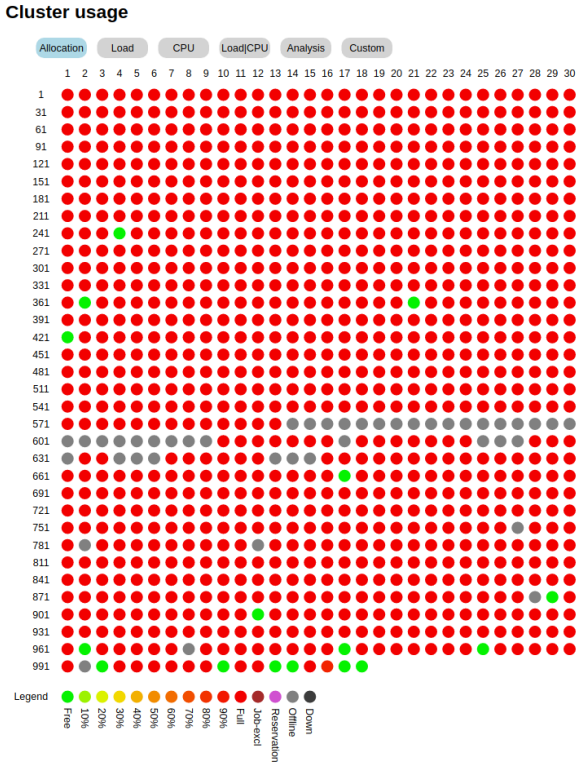
- Storage lifecycle
  - HOME deleted after 1 year without any active project
  - PROJECT data deleted some time after a project ends
  - SCRATCH data deleted after 90 days of inactivity

COMPILING/PREPARING DEPENDENCIES

- You must compile your program and its dependencies for your target cluster
- This will be described in Part 3

SELECTING PROJECT AND CLUSTER

- Choose the correct computational project for your experiment
- Check how much core hours are left in the project
  - `$ it4ifree`
  - https://scs.it4i.cz/
- Check the status of clusters
  - https://extranet.it4i.cz/rsweb/karolina/cluster-allocation



QUEUING SYSTEM

- Each IT4I cluster is shared by many users
- To perform a computation (a job), you must go through a queue
  - We use a queuing system called PBS (Portable Batch System)
- There are several queues with different properties
  - *qexp* (quick experiments, does not charge for use, up to 2 nodes and 1 hour jobs)
  - *qprod* (common computations, up to 756 nodes and 2 day jobs)
  - *qlong* (long-running computations, up to 20 nodes and 6 day jobs)
  - *qnvidia* (accelerated nodes each with 8 A100 NVIDIA GPUs)
  - *qfat* (fat node – 768 cores, 24 TiB RAM)
  - You can find the complete queue list here
- To access most queues you will need to specify a computational project that you are a part of
  - Computational resources that you spend are deducted from the used project
  - Cost of a computation: Time x Node count (x Core count x Normalization factor)
  - After all resources run out, you can still use the `qfree` queue up to 120% of the original resources

ⓘ You can find more information about queues and projects here

USING PBS

- You can submit jobs on the cluster in two modes
  - Batch mode (default): you specify a script which is executed once you get to the front of a queue
  - Interactive mode: your terminal will be connected to the first computing node in the job via SSH
- Submission is performed using the `qsub` command
- You have to give `qsub` some basic parameters to define a job:
  - Number of computing nodes used in the job: `-lselect=4`
  - Maximum running time (called walltime): `-lwalltime=02:30:00`
  - Queue: `-qqexp`
  - Project (if required by the queue): `-A OPEN-0-0`
  - (Bash) script that will be executed (for batch mode)
- There are also some other useful options
  - Job name: `-N MY_JOB`
  - Send e-mail on job start/end/error: `-m bea`
- You can have multiple jobs in the queue at once (both waiting and executing)
- Be careful with walltime to avoid wasting project resources!

ⓘ Other HPC centres might use a different queue system, e.g. Slurm

SUBMITTING A JOB USING PBS

1. Prepare a bash script that will run your computation
2. Submit a job using the qsub command and note the Job ID that it outputs

```
# enqueue script myjob.sh with 64 nodes on qprod under project OPEN-0-0
$ qsub -A OPEN-0-0 -q qprod -l select=64,walltime=03:00:00 ./myjob.sh
9875350
```

3. Use qstat to query queue status to see the expected start time and computation status

```
$ qstat -u $USER -T
Job ID   Queue  NDS   Est Start Time
9875350  qexp   1     15:56
```

- Use the job ID to identify individual jobs

- You can also put the submission options directly into the script

There are a lot of other options that you can specify, find out more in the documentation

EXAMPLE PBS SCRIPT

```
#!/bin/bash
#PBS -q qexp
#PBS -N MYJOB
#PBS -l select=2:ncpus=24
#PBS -A OPEN-0-0

# Change to scratch directory, exit on failure
SCRDIR=/scratch/$USER/myjob
cd $SCRDIR || exit

# Load necessary module
ml OpenMPI

# Execute the calculation
mpirun ./mympiprogram
```

You can find a similar example and advanced information here

JOB EXECUTION

- Once the job gets to the front of the queue
  1. PBS will allocate the specified number of nodes
  2. The specified script will be executed
     - On the first allocated node
     - In your HOME directory
  3. Once your script finishes, the job will also end
  4. `stdout` and `stderr` of your script will be written to a file on the shared filesystem
     - `<job-name>.o<job-id>` - standard output
     - `<job-name>.e<job-id>` - standard error output
     - They will be stored in the directory where you submit the job
       - You can override this location with `-o` and `-e`
- Useful environment variables available during a job
  - `PBS_O_WORKDIR` – directory from where you submitted the job
  - `PBS_NODEFILE` – path to a file containing all compute nodes of the current job
  - `PBS_JOBID` – job ID of the current job

MONITORING JOB STATUS

- Once your job starts running, you can observe its status in several ways
- `qstat`
  - Displays job status, elapsed time, allocated computing nodes
  - You can connect to the individual computing nodes via SSH to inspect them

```
$ qstat -u $USER -n
Job ID   Queue  NDS   Elap Time
9875350  qexp   1     00:10
r3i1n9/0*24,r3i2n9/0*24
$ ssh r3i1n9
[r3i1n9]$ htop
```

- `check-pbs-jobs`
  - Allows reading standard output and error output streams
  - Only available when the job is running

```
$ check-pbs-jobs --jobid 9875350 --print-job-out --print-job-err
### Print job standard output:
Computation started
### Print job standard error:
Error at main.c:16: File not found
```

- When something goes wrong you can delete jobs (both running and enqueued)
  - `$ qdel <job-id>`

MORE PBS INFORMATION

- Jobs are prioritized based on several properties
  - Selected queue
  - Amount of recent computation in a project
  - Hint: if you want to get ahead in the queue, specify a small(er) walltime
- PBS has a lot of configuration and options
  - Job arrays
    - Many jobs with the same script, but different inputs
  - Advanced node configuration/placement
    - Enable/disable Turbo boost, kernel modules, …
    - Select nodes by CPU type, network switch, network topology location
  - You can find more here

SUBMITTING JOBS MORE EASILY

HyperQueue

Job execution system designed for ergonomics and performance

- Allows you to submit tasks without dealing with PBS jobs
- Useful if you have a large number of relatively short-lived tasks to execute
  - Less overhead than PBS
  - Able to leverage all available cores and nodes

ASKING FOR HELP

If you have trouble with

- Connecting to login nodes
- Building code or dependencies
- Submitting PBS jobs

Then

1. Consult the documentation
2. If that does not help, create a ticket

Jakub Beránek
jakub.beranek@vsb.cz

IT4Innovations National Supercomputing Center
VSB – Technical University of Ostrava
Studentská 6231/1B
708 00 Ostrava-Poruba, Czech Republic
www.it4i.cz

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research, Development and Education

MINISTRY OF EDUCATION, YOUTH AND SPORTS