High Performance Computing for Genomics

Part I: High Performance Computing

KU LEUVEN GENOMICS CORE UZ LEUVEN

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

KU LEUVEN GENOMICS CORE UZ LEUVEN

Login

Login is possible through various methods.

Command line example (Linux/Mac):

```
$ ssh vsc3XXXX@login.hpc.kuleuven.be
```

Or via putty (Windows)

KU LEUVEN GENOMICS CORE UZ LEUVEN

If you are not logged in yet,

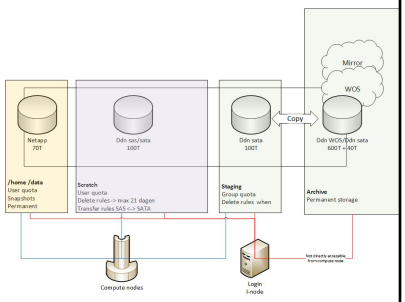
Now is the moment

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Storage

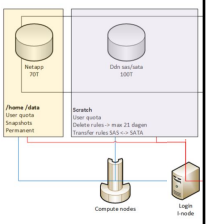
- NAS storage, fully back-up with snapshots for /home and /data
- Scratch storage, fast parallel filesystem
- Staging storage, to store current working projects
- Archive storage, to store large amounts of data for long time



Personal Storage

Personal Storage, only accessible by the owner
Fully Backup

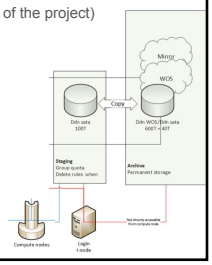
- Home Directory
  - $VSC_HOME or /user/leuven/3XX/vsc3XXXX
  - 25GB
  - Important data, configuration files
- Data Directory
  - $VSC_DATA or /data/leuven/3XX/vsc3XXXX
  - 75GB
  - Important data, biger then Home
- Scratch Space
  - $VSC_SCRATCH or /scratch/leuven/3XX/vsc3XXXX
  - 100GB
  - Temporary data, will be deleted within 21 days



Project Storage

Permissions are managed per group (usually the group of the project)

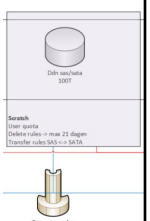
- Staging Space
  - Storage for project files (while working at the project)
  - Size is project dependable (per 1TB)
- Archive Storage
  - Long term storage and backup for project files (or staging)
  - Size is project dependable (per 1TB)
  - Only accessible from the login node



Temporary Storage

Scratch: fast storage, used during the processing of the data

- Node Scratch
  - $VSC_SCRATCH_NODE
  - Size is machine dependable, min 150 GB
  - Is NOT accessible from the login node
- Site Scratch
  - $VSC_SCRATCH_SITE
  - Similar as $VSC_SCRATCH_NODE
- Personal Scratch
  - $VSC_SCRATCH
  - Personal scratch, 100GB
  - Accessible from the login node
  - Data will be deleted within 21 days



Exercise 1

Find the path for the following locations:

- $VSC_HOME
- $VSC_DATA
- $VSC_SCRATCH
- $VSC_SCRATCH_NODE

Copy Data

```
$ rsync -ahr --progress /staging/leuven/stg_00019/workshop/* .
```

rsync --progress -ahr filename targetDirectory

--progress	Show the progress of the copy
-a	Preserve almost everything
-h	Output numbers must be Human readable
-r	Copy directories recursively

Exercise 1b

Make a tutorial directory in your data directory

Copy the workshop data into this directory:

/staging/leuven/stg_00019/workshop

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Credit System

VSC ~ regular bank

Each job costs a certain number of credits

Each job belongs to a project

Each project account has its own credits

Users have access to one or multiple project accounts
(multiple users can have access to the same project account)

Credit System: balance

Load the accounting module

```
$ module load accounting
```

Check the amount of credits at your disposal

```
$ gbalance
```

Credit System: statement

Get an overview of the transactions

```
$ gstatement
```

Get an overview of the transactions of a certain project, over a certain time

```
$ gstatement -g lp_projectname -s 2015-09-01 -e 2015-09-30
```

-g	groupname
-u	username
-s	start
-e	end

Credits System: quotes

Estimating the cost of a job is possible by requesting a quote:

```
$ gquote -l nodes=3:ppn=4:ivybridge,pmem=2gb,walltime=48:00:00
```

- nodes=3:ppn=4:ivybridge
  3 nodes, with 4 processors per node of the ivybridge type
- pmem=2gb
  2GB of memory (RAM)
- walltime=48:00:00
  The job will run for maximum 48 hours

Job Cost Calculation

The effective cost of the job is calculated using this formula:

```
(0.000278*nodes*walltime + startup)*nodetype
```

nodes	The number of nodes that were reserved
walltime	The effective duration of the job (in seconds)
startup	0.1, it is added for the overhead of scheduling
nodetype	A representation of the node type's performance

Exercise 2

- Get a quote for 5 minutes computing on 1 node, 5 processors of the ivybridge type
- Get a quote for 20 minutes computing on 2 nodes, 4 processors of the haswell type, using 100Gb of memory
- Get a quote for 1 hour for both the ivybridge and the haswell processor

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Module System

For many programs, multiple versions are installed, and each version requires specific libraries.

Toolchains consists of compilers and libraries, together with software depending on those libraries.

The module system inside the toolchain is used to manage the environment variables, and all dependencies to resolve possible conflicts.

Module System: Module av

Many software packages are installed as modules:

```
$ module av
----- /apps/leuven/thinking/2014a/modules/all ------
Autoconf/2.69-GCC-4.8.2
Autoconf/2.69-intel-2014a
Automake/1.14-GCC-4.8.2
Automake/1.14-intel-2014a
BEAST/2.1.2
...
pyTables/2.4.0-intel-2014a-Python-2.7.6
timedrun/1.0.1
worker/1.4.2-foss-2014a
zlib/1.2.8-foss-2014a
zlib/1.2.8-intel-2014a
```


Module System: load and unload

Load a module

```
$ module load BEAST
$ module load  BEAST/2.1.2
$ module load  BEAST/2.1.2  zlib/1.2.8-foss-2014a
```

Unload a module

```
$ module unload BEAST
```

Unload all modules

```
$ module purge
```


Module System: list

View all loaded modules

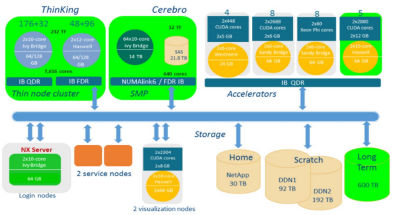
```
$ module list
Currently Loaded Modulefiles:
  1) /thinking/2014a
  2) Java/1.7.0_51
  3) icc/2013.5.192
  4) ifort/2013.5.192
  5) impi/4.1.3.045
  6) imkl/11.1.1.106
  7) intel/2014a
  8) beagle-lib/20140304-intel-2014a
  9) BEAST/2.1.2
 10) GCC/4.8.2
 11) OpenMPI/1.6.5-GCC-4.8.2
 12) gompi/2014a
 13) OpenBLAS/0.2.8-gompi-2014a-LAPACK-3.5.0
 14) FFTW/3.3.3-gompi-2014a
 15) ScaLAPACK/2.0.2-gompi-2014a-OpenBLAS-0.2.8-LAPACK-3.5.0
 16) foss/2014a
 17) zlib/1.2.8-foss-2014a
```

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Hardware

- Thinking
  Thin node cluster
- Cerebro
  Shared Memory Processing
- Accelerators
  Like GPUs
- Storage



Hardware: Thinking

- Thinking
  - Thin, powerful nodes
  - 2 processor types:
    - Haswell
      208 nodes, 24 cores/node
      High computing power, low I/O
    - Ivybridge
      144 nodes, 20 cores/node
      Lots of I/O
  - 2 versions of RAM: 64GB and 128GB
- Cerebro

Hardware: Cerebro

- Thinking
- Cerebro
  - Shared Memory
  - 2 partitions:
    - Smp1

      48 nodes, 10 cores/node (Ivybridge)
      250 GB shared (max 11.77 TB)
    - Smp2

      16 nodes, 10 cores/node (Ivybridge)
      124 GB shared (max 1.79 TB)

Hardware: Overview

Cluster	Partition	#cores (threads) per node	Usable Memory	#Credits/hour
ThinKing	Ivybridge	20	60-124GB	4.76
ThinKing	Haswell	24	60-124GB	6.68
Cerebro	Smp1	10	Shared 250GB	3.45
Cerebro	Smp2	10	Shared 124GB	3.45

The Operating System also needs some RAM, good practice is to reserve 4GB.

Hardware: which task on which cluster?

Cluster	Partition	Task Description	Task Example
ThinKing	Ivybridge	Memory low jobs, with lots of I/O	• Alignment • Read Mapping • Variant Calling • Read Counting • ...
ThinKing	Haswell	Memory low jobs, low I/O, high computing power needed	• Model Calculations • Star Discovering • ...
Cerebro	Smp1 and Smp2	High memory jobs, computing power less important	• De Novo Assemblies • Reference-based Assemblies • ...

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Portable Batch System (PBS)

Portable Batch System (or simply PBS) is the name of computer software that performs job scheduling. Its primary task is to allocate computational tasks, i.e., batch jobs, among the available computing resources. It is often used in conjunction with UNIX cluster environments. PBS is supported as a job scheduler mechanism by several meta schedulers ...

-- by Wikipedia

PBS script

Every shell script (.sh) can be turned into a PBS script (.pbs).
A typical scripts will need the following adjustments:

1. Include PBS headers
2. Load the software modules
3. Optimize I/O using a scratch node
4. Check if results are copied/stored in a data or staging directory

PBS header

Headers contain the run parameters
Each line starts with #PBS then the parameter and the value

```
#!/bin/bash -l
#PBS -l walltime=12:00:00
#PBS -l mem=100gb
#PBS -l nodes=1:ppn=20:ivybridge
#PBS -M mail@mail.com
#PBS -m aeb
#PBS -N jobname
#PBS -A lp_projectname
```


PBS header: job description

Walltime: the maximum time the job can run

```
#PBS -l walltime=12:00:00
```

Mem: the maximum memory available for the job

```
#PBS -l mem=100gb
```

Number of nodes, processors per node (ppn) and the processor type

```
#PBS -l nodes=1:ppn=20:ivybridge
```


PBS header: notifications

You can send a mail @ certain conditions

-M specifies the mail adres

```
#PBS -M mail@mail.com
#PBS -m aeb
```

m the conditions when a mail has to be send:

b	When the job begins
e	When the job ends
a	When the job is aborted

PBS header: billing

The name of the job, this will be visible for other users

```
#PBS -N jobname
```

The project that will be used for the billing

```
#PBS -A lp_projectname
```


PBS script

Every shell script (.sh) can be turned into a PBS script (.pbs).
A typical scripts will need the following adjustments:

1. Include PBS headers
2. **Load the software modules**

```
module load  BEAST/2.1.2
```

3. Optimize I/O using a scratch node
4. Check if results are copied/stored in a data or staging directory

PBS script

Every shell script (.sh) can be turned into a PBS script (.pbs).
A typical scripts will need the following adjustments:

1. Include PBS headers
2. Load the software modules
3. **Optimize I/O using a scratch node**
   $VSC_SCRATCH, $VSC_SCRATCH_NODE
   Scratch is fast temporary storage
4. Check if results are copied/stored in a data or staging directory

PBS script

Every shell script (.sh) can be turned into a PBS script (.pbs).
A typical scripts will need the following adjustments:

1. Include PBS headers
2. Load the software modules
3. Optimize I/O using a scratch node
4. **Check if results are copied/stored in a data or staging directory**
   Scratch is temporary, when the job is finished, data will be lost

Overview

- Login
- Storage
- Credit system
- Module system
- Hardware
- PBS
- Job Submission

Job submission

Thinking

```
$ qsub run-job.pbs
```

Cerebro

```
$ module load cerebro/2014a
$ qsub run-job.pbs
```

When a job is submitted, the job id is returned.
jobIDs are unique.
Thinking jobs starts with 2
Cerebro jobs starts with 3

Jobs progress

An overview of all your jobs:

```
$ qstat -u vsc3XXXX
hpc-p-svcs-5:

Req'd       Req'd       Elap
Job ID                  Username    Queue    Jobname
SessID  NDS   TSK   Memory      Time    S   Time
----------------------- ----------- -------- ----------------
------ ----- ------ --------- --------- - ---------
30017323.hpc-p-svcs-5   vsc3XXXX    q21d     jobname
197071     1     10       1gb 480:00:00 R  00:04:16
```


Job estimated start

Get the estimated start of a job:

```
$ showstart 30017323.hpc-p-svcs-5
job 30017323 requires 10 procs for 20:00:00:00
Estimated Rsv based start in                00:00:00 on Mon Apr
18 16:29:10
Estimated Rsv based completion in        20:00:00:00 on Sun May
8 16:29:10
Best Partition: smp1
NOTE:  job is running
```


Job overview

An overview of every parameter and all used resources can be requested:

```
$ checkjob 30017323.hpc-p-svcs-5
job 30017323
AName: jobname
State: Running
Creds:  user:vsc3XXXX  group:vsc3XXXX  account:lp_projectname
class:q21d  qos:normal
WallTime:   00:02:43 of 20:00:00:00
SubmitTime: Mon Apr 18 16:28:56
  (Time Queued  Total: 00:00:14  Eligible: 00:00:14)
StartTime: Mon Apr 18 16:29:10
TemplateSets:  DEFAULT
Total Requested Tasks: 10
Req[0]  TaskCount: 10  Partition: smp1
Memory >= 1024M  Disk >= 0  Swap >= 0
Dedicated Resources Per Task: PROCS: 1  MEM: 24G
```

Job stop/kill/delete

Sometimes a job has to be killed.

```
qdel 30017323.hpc-p-svcs-5
```


Exercise 3

1. Open paths.pbs
   Edit the PBS header (change the mail, and accounting information)
2. Ask a quote based on the PBS header
3. Start the job on thinking
4. Check the start time and status of the job
5. Check the output of the job

Coffee break