PRINCESS MARGARET GENOMICS CENTRE

Last modified November 13, 2014

Cluster – Info Sheet

About the Princess Margaret Computational Biology Resource Centre (PMCBRC) cluster

Welcome to the PMCBRC cluster! We are happy to provide and manage this compute cluster as a resource for the PIs (and their staff/students) at Princess Margaret who have used some of their funding to help build the system. Currently, the cluster consists of roughly 200 cores of Sandy Bridge Xeon processors with roughly 10GB/core of RAM (128GB/motherboard) interconnected with 10GBE. There is roughly 200TB of shared disk space in a RAID6 configuration. This is soon to increased > 3-fold overall. We also have a dedicated machine that can be used for higher memory statistics jobs in R as well as a GP-GPU machine for specialized tasks.

Although this is a shared resource, we have decided to not impose heavy restrictions on users. Instead, since the pool of users is small, the system is being left in a more 'open' state where we expect end users to manage their own usage and be considerate of others as to how much resource they take up. Please keep this underlying principle at the forefront when you create programs and run jobs on the system.

Using the cluster

PMGC cluster uses Oracle Grid Engine (also known as the Sun Grid Engine, or SGE) software to accept, schedule, dispatch and manage user jobs. This software is responsible for allocating resources such as processors, memory, disk-space and computing time for each job submitted to the compute nodes. A job is just a program that you want to run (e.g. blastx). Although it is technically possible, you are not permitted to run jobs locally on mordor. All jobs must be submitted to the queue. If you require access to a local machine to run such work as an interactive R session you may login to a Virtual Machine on the cluster using "*qrsh*" or "*qlogin*" (**NOT** *ssh*). Please note that this is only acceptable for low memory jobs. If your work requires a high memory local machine for such work please contact our bioinformatics team as we can provide this on a case by case basis.

DO NOT download large files (ie > 1TB) to our system. Although we do not currently have any set policy on size of a user's home directory, we do regularly check the size of each and ask you keep it as small as possible. If we detect a larger than average user directory we will contact that user and work with them to reduce the size. However, we reserve the right to change our policies at any time. If our users do not respect the current flexible usage requirements, or if our user base expands excessively, we may have to enforce stricter limits. If you require a large download please speck to our bioinformatics group directly to ensure that it is not something we already have available and/or so that we can download it for you and place it in a suitable commonly accessible area so that we do not have duplication of large files. This policy is largely temporary until we purchase more disk space (estimated Early 2014) but it is always good practice to keep your file space as clean as possible.

Using the Queue (submitting jobs to cluster)

1. To submit a job to the queue you must first create a bash script, such as this example (named "test.sh"):

   ```
   #!/bin/bash
   #
   #$ -cwd
   #$ -S /bin/bash
   #
   date
   sleep 10
   date
   ```
   -cwd means to execute the job for the current working directory

   -S /bin/bash specifies the interpreting shell for this job to be the Bash shell

2. To run this script on the cluster, enter "*qsub ./test.sh*"

   Although there are multiple queues available on the cluster, general users will submit to the general queue without any requirement for specification. Some users are able to submit to additional queues and these users will be given the queue name and additional instructions.

3. To check on the status of a job you have submitted, enter "*qstat*" (**DO NOT** log in to a particular VM to check status of a job – use "*qstat –j* <job#>" to get details of a particular job)
4. To stop a job that is running, enter "*qdel* <job#>" (you may only delete a job that you started)
5. To check on the amount of memory/slots that are available on the cluster you can enter "*qhost*" which will show you something like this (see below). Here you can see that there are 10 nodes each with 2 virtual machines, each with about 47-62 Gb available memory,

each having 6 slots for jobs. Currently n3vm2 and n7vm2 have jobs running using 17.4 Gb of memory on each.

Currently we do not have any policies enforced regarding restrictions on the amount of memory, size or type of job submitted. We are asking users to self-regulate and using the command “*qhost*” can be helpful as a means to check the cluster usage prior to submitting a job – particularly if the job is high memory, or when submitting a very large number of jobs. Be sure that the job(s) you are submitting are not going to overload the cluster. If your job has a specification for specific memory requirements (eg. Java –Xmx setting) please set the requirement to a lower and reasonable amount (if necessary ask for further clarification). If the memory usage becomes problematic as more users join and more jobs are running we may have to enforce specific policies and limits if it cannot be self-regulated satisfactorily.

```
HOSTNAME   ARCH        NCPU   LOAD   MEMTOT  MEMUSE  SWAPTO  SWAPUS
global     -           -      -      -       -       -       -
n10vm1     linux-x64   12     0      62.9G   880.6M  31.5G   10.8M
n10vm2     linux-x64     12      0   47.1G   607.8M  31.5G   11.6M
n11vm1     linux-x64     12      0   62.9G   924.7M  31.5G   14.1M
n11vm2     linux-x64     12      0   47.1G   633.7M  31.5G   11.3M
n12vm1     linux-x64     12      0   62.9G   894.1M  31.5G   10.5M
n12vm2     linux-x64     12      0   47.1G   690.9M  31.5G   8.1M
n1vm1      linux-x64     12      0   62.9G   959.3M  31.5G   13.8M
n1vm2      linux-x64     12      0   47.1G   843.3M  31.5G   10.5M
n2vm1      linux-x64     12      0   62.9G   945.9M  31.5G   14.0M
n2vm2      linux-x64     12      0   47.1G   704.9M  31.5G   11.5M
n3vm1      linux-x64     12      0   62.9G   901.5M  31.5G   14.0M
n3vm2      linux-x64     12   0.28   47.1G   17.4G   31.5G   10.2M
n4vm1      linux-x64     12      0   62.9G   848.2M  31.5G   14.0M
n4vm2      linux-x64     12      0   47.1G   687.3M  31.5G   10.1M
n5vm1      linux-x64     12      0   62.9G   794.9M  31.5G   16.5M
n5vm2      linux-x64     12      0   47.1G   658.0M  31.5G   13.7M
n6vm1      linux-x64     12      0   62.9G   927.6M  31.5G   15.2M
n6vm2      linux-x64     12      0   47.1G   621.6M  31.5G   10.3M
n7vm1      linux-x64     12      0   62.9G   916.8M  31.5G   14.0M
n7vm2      linux-x64     12   0.52   47.1G   17.4G   31.5G   10.5M
n8vm1      linux-x64     12      0   62.9G   884.6M  31.5G   13.8M
n8vm2      linux-x64     12      0   47.1G   670.3M  31.5G   12.0M
n9vm1      linux-x64     12      0   62.9G   985.0M  31.5G   13.6M
n9vm2      linux-x64     12      0   47.1G   720.8M  31.5G   11.2M
```

Last modified November 13, 2014

Modules

At the PMCBRC cluster we use *Environment Modules* to configure your user space to run applications. It is important to be aware of this, because you MUST use specific modules in order to run different bioinformatics applications. This allows for stable and consistent configuration of software packages and easy tracking of versions used. Modules allow us to manage different versions of applications so a user may easily choose which to use for each instance. This allows a user to easily access an older version of a package to maintain consistency for a specific project while other users can employ the newest iterations for their work.

How to use modules:

1. Check which modules are available for loading enter "*module avail*"
2. To see which modules are currently loaded enter "*module list*"
3. To load a specific module enter "*module load <module name>"*
4. To remove a module enter "*module unload <module name>"*

Back ups

Currently we are only backing up everything under the */mnt/stbak/home/* directory (and are doing so regularly including back up to tape). It is the user's responsibility to move important files to their folder located here. Please be sure to only keep current and important files in these folders as space is limited. We also recommend that each user back up your source code (at least) to your own local computer as an additional precaution.

To connect to cluster via an RIS Trusted Host computer

From terminal program: *ssh <username>@mordor*

To connect to cluster via RDP

Follow instructions on the UHN remote connection site (http://www.uhnres.utoronto.ca/remote/) including using OTP keys (obtained from RIS) to sign in. Immediately after sign in use PuTTY (or equivalent terminal program) to ssh to host 192.75.165.28 port 5059. Please note that we do not support or allow the use of remote X desktops.

PRINCESS MARGARET GENOMICS CENTRE

Last modified November 13, 2014

tmux

tmux is a terminal multiplexer, an application that lets you switch easily between several programs in one terminal, detach them (they remain active in the background) and reattach them to a different terminal (i.e. you can start a terminal instance in the office and, if you use tmux, you can see what is happening in this terminal later when you log in via RDP at home or at another location).

When tmux is started it creates a new *session* with a single *window* and displays this on the screen. Any number of tmux instances can connect to the same session and any number of windows may be present in the same session.

This will be especially useful if you are connecting to the cluster using RDP as the connection can sometimes disconnect unexpectedly. Using tmux will keep your session running even in the case of accidental disconnection (eg. ssh connection timeout).

Below we have outlined a few of the basic commands to get started using tmux. The full manual is available here: http://www.openbsd.org/cgi-bin/man.cgi?query=tmux&sektion=1

A useful cheat-sheet is available here: https://gist.github.com/henrik/1967800

1. To start using tmux in your terminal enter: *tmux*
   - Now you will see a status line at the bottom of the screen.
2. To create a new pane (a split on the main window)
   - Split vertically: enter *Ctrl-b*, release these keys and then enter *%*
   - Split horizontally enter *Ctrl-b*, release these keys and then enter *"*
3. To move between panes use the arrow keys after first entering *Ctrl-b*
4. To create another window enter *Ctrl-b*, release these keys and then enter *c*
5. To switch between multiple windows enter *Ctrl-b*, release these keys then enter *n* (for next) or *p* (for previous) or enter *w* to see all windows in a list and choose between them.
6. To close a window enter *Ctrl-b*, release these keys then enter *&* (or just type *exit* at the command).

PRINCESS MARGARET GENOMICS CENTRE

Last modified November 13, 2014

Git/Mercurial

Git and Mercurial are two systems that we have available for source code management and revision control. Some tutorials for beginners can be found here:

- Git: http://sixrevisions.com/resources/git-tutorials-beginners/
- Mercurial: http://mercurial.selenic.com/wiki/Tutorial

Some basic instructions/commands are described here to get you started:

1. To clone a project from the remote server into the local machine (for development):
   - Git commands:
     ```
     git clone ssh://<user>@<server>/<path>/repo_git/<ProjectName>
     cd /<path>/<ProjectName>
     git checkout <branch> (eg. "git checkout origin/dev")
     ```
   - Mercurial commands:
     ```
     hg –v clone ssh://<user>@<server>/<path>/repo_mercurial/<ProjectName>
     ```
2. To push back the modified project to the remote server:
   - Git commands

   To commit changes and create tag:
     ```
     cd /path/<ProjectName>
     git add . (same as: "git add –A" or "git add –u" or "git add *")
     git commit <file_name> -m "file_name: message"
     ```

   To push the modified project into the remote server:
     ```
     git push ssh://<user>@<server>/<path>/repo_git/<ProjectName>
     ```
   - Mercurial commands

   To commit changes:
     ```
     hg commit <file_name> -m "file_name: message"
     ```

   To push the modified project into the remote server:
     ```
     sudo hg push –v ssh://<user>@<server>/<path>/repo_mercurial/<ProjectName>
     cd /<path>/repo_mercurial/project
     hg update
     ```

3. To pull out the latest version of the project from the remote server to the local machine:
   - Git commands:
     cd /<path>/<ProjectName>
     git pull ssh://<user>@<server>/<path>/repo_git/<ProjectName>
     git checkout <branch>

   - Mercurial commands:
     cd /<path>/<ProjectName>
     hg pull –v ssh://<user>@<server>/<path>/repo_mercurial/<ProjectName>
     hg update

4. To revert to a previous version:
   - Git commands:
     git revert <commit_id>

   - Mercurial commands:
     cd /<path>/<ProjectName>

   To revert the entire project enter:
     hg revert –r <versionNumber> -all

   OR – to revert only a specific file, enter:
     hg revert –r <versionNumber> [--no-back] <filename>

5. Check the differences between modified files and the original files:
   - Git commands:
     cd /<path>/<ProjectName>
     git log (this will show all logs)
     git log –<N> (this will show the last *N* logs)
     git diff (this will show the differences)

   - Mercurial commands:
     cd /<path>/<ProjectName>
     hg log | grep <filename>
     hg log –r <version>
     hg diff (show the differences)

Last modified November 13, 2014

Resources Currently Available (just highlights, this is not comprehensive)

1. MySQL database:
   - Port: 3306
   - Username: UHN
   - Password: microarray
   - Databases available: cosmic, hg19, mm10, pubmed (use “show databases” after you log into MySQL server)
   - There are many public resources from UCSC, ensembl, NCBI, PUBMED etc. in this database and they are kept as up-to-date as possible. If you want to create a new table, please contact us.
   - Connect to the database directly from mordor:
     - ➢ Enter the following:

       module load mysql/5.6
       mysql –uUHN –pmicroarray

   - We also have a “noSQL” database, MONGODB, available. Please ask if you would like to use this.
2. UCSC genome browser:
   - Access on mordor via: http://<mordor>/index_ucsc.html
   - Click “Table Browser” from the left-hand menu bar and choose the table you need.
3. Galaxy:
   - Galaxy is an open source, web-based platform for data intensive biomedical research. If you are new to Galaxy you can find tutorials and help on the main Galaxy website (https://usegalaxy.org).
   - If you are new to Unix commands then we suggest you try using Galaxy for some sequencing processing.
   - You will need to obtain an account from Zhibin or Carl for our Galaxy website.
   - Login here to our Galaxy site: http://192.168.1.150:8080/galaxy
   - To obtain UCSC data from within Galaxy:
     - ➢ Click “Get Data” and then choose “UCSC at OCIGC table browser”
     - ➢ Please remember to check “Send output to Galaxy”, so the result will be passed to the galaxy server.

4. UCSC tools
   - Create a file under your home directory with the name ".hg.conf" and put the following lines into it:
     db.host=localhost
     db.user=UHN
     db.password=microarray
   - Use "module load ucsctools/current" first and then the UCSC tools will be available.
5. Genome Reference & index files
   - Multiple files available on mordor: /mnt/work1/data/genomes/<species>/<build>/
   - For Reference genome files, we recommend use of subdirectory called "iGenomes" which was downloaded from iGenome (Illumina).
   - For variant calling (human only) look in: /mnt/work1/data/genomes/human/<build>/variantcallingdata/
   - Modules available for human, mouse and yeast genome files
     - Use *module avail* to see which module to load
     - Eg. *module load igenome-human/hg19*
       Use *module show igenome-human/hg19* (to see which component of the module you would like to use in your script)
       Eg. Use *$BWAINDEX* as path for reference genome when running BWA
6. Scripts and Wrappers for processing
   - /mnt/work1/software/ocigcscripts/production
   - We have written some scripts to process sequencing data through our standard pipeline which you are welcome to use (list of scripts with brief description on next two pages).
   - You may view the headers with arguments/usage for each script here: http://www.pmgenomics.ca/pmcomp/index.php/software/42-pmgc-scripts-on-mordor

Last modified November 13, 2014

Script Name	**Brief Description**
convertBCLtoFASTQ_1thread.pl	Used to convert BCL files to Fastq
convertBCLtoFASTQ.pl	Used to convert BCL files to Fastq
run_fastqc_hiseq.sh	Runs Fastqc on data from Illumina HiSeq
makeHTMLQCReport.pl	Creates HTML report from fastqc results
run_bwa_paired.sh	Runs BWA on paired samples
run_bamqc.sh	Runs BAMqc on BAM files
runAnnovar.pl	Runs ANNOVAR software for functional annotation of genetic variants
xenomeClassify.pl	Classifies the reads in 1 or 2 input FASTQ files as belonging to either the host organism (mouse in the typical xenograft use case) or the graft organism. The script will perform indexing of the references for each organism as needed and corrects the output reads classified as definitely graft (e.g. the file [sample]human[1|2].fastq) so that they are valid FASTQ files which can be read by the FastQJobWrapper script. These validated output files have the suffix _R1.fastq or _R2.fastq.
" variant_wrapper" folder	Contains scripts etc. to run the entire pipeline for variant calling from exome data from alignment of fastq files to variant calling with Samtools mpileup, Haplotype Caller, Unified Genotyper, Dindel and/or mutect **Note: mutect may be run in conjuction with any one of the others. Please read the README file and follow the instructions there.
runVariantPipeline.pl	This script will run all necessary steps from alignment to variant calling in one step - use sample.xml file to chose arguments and designate input files (calls other scripts within this directory)
runVariantPipeline_matchedSamples.pl	This script will run all necessary steps from alignment to variant calling in one step when you have matched sample (eg. tumor-normal pairs) - use sample_matchedSamples.xml file to choose arguments and designate input files (calls other scripts within this directory)
autoAlign.pl	Runs either BWA or Bowtie to align fastq data
convertSAMtoBAM.pl	Converts a .sam file to a .bam file
processBAMs.pl	Runs picard to Mark Duplicates, runs GATK Realigner Target Creator, Indel Realigner, Base Recalibrator and Print Reads
cocleanBAMs.pl	Runs all the same processing steps as processBAMs.pl, but does it jointly for samples from the same patient.

PRINCESS MARGARET GENOMICS CENTRE

Last modified November 13, 2014

callVariantsGenotypeGVCFs.pl	Accepts a gVCF file produced by HaplotypeCaller from GATK Version 3+. The script outputs a VCF containing a list of all variants found in the gVCF files and a column for each read group SM tag. Note that all of your VCFs must have different read groups to be processed properly! The VCF containing all variants will then be split based on the sample names in the header (e.g. the columns listed after #CHROM POS ... INFO FORMAT) and filtered by DP (unfiltered depth over all samples) > 10 and QD (QUAL / unfiltered depth of non- reference samples) > 2.0. This filtering may not work for your experiment, so be careful.
callVariantsUnifiedGenotyper.pl	Runs GATKs Unified Genotyper on BAM files for variant calling
callVariantsHaplotype.pl	Runs GATKs Haplotype Caller on BAM files for variant calling
callVariantsDindel.pl	Runs Dindel on BAM files to call small indels
callVariantsSamtools.pl	Runs GATKs mpileup on BAM files for variant calling
callVariantsMutect.pl	Runs MuTect(Broad Institute) to call somatic variants from a tumor-normal pair.
"tophat2_wrapper" folder	Contains scripts etc. to run the entire pipeline for RNASeq analysis from alignment of fastq files to annotation and compairson of samples.
topHat_wrapper.pl	This script will run all necessary steps from alignment to ? For RNA-Seq experiments - use input.xml file to choose arguments and designate input files (calls other scripts within this directory)
run_tophat2_paired.sh	This script runs TopHat2 on paired-end gzipped FASTQ files, which are found by searching the source directory for files ending in "*R1.fastq.gz"
run_cuffquant.sh	Converts BAM file to intermediate format needed to run CuffDiff
run_cufflinks.sh	This script runs cufflinks2 on all files ending in 'accepted_hits.bam' found in the source directory.
run_rnaseqc.sh	Runs QC on RNASeq alignment after tophat