**HPC Practical**

**1 General commands**

Get documentation on a command:

man <command>

Try the following commands:

man sbatchman squeue

man scancel

**1.1 Submitting jobs**

The following example script specifies a partition, time limit, memory allocation and number of cores. All your scripts should specify values for these four parameters. You can also set additional parameters as shown, such as jobname and output file. For This script performs a simple task — it generates of file of random numbers and then sorts it.

#!/bin/bash

for i in {1..100}; do

echo $RANDOM >> SomeRandomNumbers.txt

done

sort SomeRandomNumbers.txt

You can submit your job with the command:

sbatch -p general -N 1 -n 1 --mem 100 -t 0-2:00 -o myscript.out -e myscript.err myscript.sh

Examine the content of myscript.out and myscript.err.

Alternatively, you can submit the job by redirecting input into an interactive shell script:

sbatch -p general -N 1 -n 1 --mem 100 -t 0-2:00 -o myscript.out -e myscript.err << EOF

#!/bin/bash

for i in {1..100}; do

echo $RANDOM > SomeRandomNumbers.txt

done

sort SomeRandomNumbers.txt

EOF

A better approach is defining resource allocation inside the shell script:

#!/bin/bash

#SBATCH -p general # partition (queue)

#SBATCH -N 1 # number of nodes

#SBATCH -n 1 # number of cores

#SBATCH --mem 100 # memory pool for all cores

#SBATCH -t 0-2:00 # time (D-HH:MM)

#SBATCH -o myscript.out # STDOUT

#SBATCH -e myscript.err # STDERR

for i in {1..100}; do

echo $RANDOM >> SomeRandomNumbers.txt

done

sort SomeRandomNumbers.txt

Now you can submit your job with the command, as a simple command:

srun ./myscript.sh

(NB Needs chmod +x on myscript.sh to make it executable or srun will fail)

Or as a batch script

sbatch myscript.sh

If you want to test your job and find out when your job is estimated to run use (note this does not actually submit the job):

sbatch --test-only myscript.sh

**1.2 Information on jobs**

List all jobs submitted bjo SLURM:

squeue -u

Check available partitions and nodes, The column **NODES(A/I/O/T)** shows number of nodes in the states "allocated/idle/other/total" for each SLURM partition.

sinfo -s

sinfo -l

sinfo -a

**sinfo** -T

List all current jobs for a user:

squeue -u <username>

List all running jobs for a user:

squeue -u <username> -t RUNNING

List all pending jobs for a user:

squeue -u <username> -t PENDING

Query configuration and limits for a specific partition:

scontrol show partition <partition>

List detailed information for a job (useful for troubleshooting):

scontrol show jobid -dd <jobid>

Check on one node:

scontrol show node <nodeid>

List status info for a currently running job:

sstat --format=AveCPU,AvePages,AveRSS,AveVMSize,JobID -j <jobid> --allsteps

Once your job has completed, you can get additional information that was not available during the run. This includes run time, memory used, etc.

To get statistics on completed jobs by jobID:

sacct -j <jobid> --format=JobID,JobName,MaxRSS,Elapsed

To view the same information for all jobs of a user:

sacct -u <username> --format=JobID,JobName,MaxRSS,Elapsed

Check the default account your jobs will use:

sacctmgr **show** **user** <username> **format**=**user**%20s,defaultaccount%30s

See all account associations for your username

sacctmgr list association where users=$USER format=account%30s,user%20s,qos%120s

**1.3 Controlling jobs**

Display status information of a running job. **sstat** provides various status information (e.g. CPU time, Virtual Memory (VM) usage, Resident Set Size (RSS), Disk I/O etc.) for running jobs. The metrics of interest can be specified using option **--format** or **-o** (s. next example).

Sstat -j <jobid>

To cancel one job:

scancel <jobid>

To cancel all the jobs for a user:

scancel -u <username>

To cancel all the pending jobs for a user:

scancel -t PENDING -u <username>

To cancel one or more jobs by name:

scancel --name myJobName

To pause a particular job:

scontrol hold <jobid>

To resume a particular job:

scontrol resume <jobid>

To requeue (cancel and rerun) a particular job:

scontrol requeue <jobid>

Check job history:

sacct -X -u <username>

Check job history **(jobid, number of nodes, list of nodes, job state and exit code) for user su01 in specified time period (10-15 November 2018)**

sacct -X -u <username> -o "jobid,nnodes,nodelist,state,exit" -S 2018-11-10 -E 2018-11-15T23:59:59

Check memory usage for the completed job with the job id:

sacct --duplicates -j <jobid> --format=JobID,JobName,MaxRSS,MaxRSSNode,MaxRSSTask,MaxVMSize,MaxVMSizeNode,MaxVMSizeTask

Check your fair-share and usage values

sshare -A $(sacctmgr -n show user <username> format=defaultaccount%30s)

**2 Case study**

You have received a RNASeq dataset of a human tissue sample here:

/scratchb/training/data/practicalII/fastq

1. You would like to use FastQC to check the data quality of this data set:
2. You would like to align this dataset into the human genome. In order to perform this task faster, you will split this this dataset into multiple segments to run them in parallel.
3. You would like to copy the output results back to your home directory.

The tools you will need to use are:

/home/bioinformatics/software/fastqc/fastqc-v0.11.5/fastqc

/home/bioinformatics/software/STAR/STAR-2.5.3a/bin/STAR

## run\_fastqc.sh

#! /bin/bash

/home/bioinformatics/software/fastqc/fastqc-v0.11.5/fastqc $1

## step1\_qc.sh

#! /bin/bash

projectdir=/scratchb/training/data/practicalII

fastqdir=$projectdir/fastq

cd $fastqcdir

for i in $fastqdir/\*.fq.gz;

do

echo $i;

sbatch -J $i -t 1-00:00 --mem=4G -o $i.out -e $i.err $projectdir/scripts/run\_fastqc.sh $i;

done

## run\_star.sh

#! /bin/bash

dbdir=/mnt/scratchb/bioinformatics/reference\_data/reference\_genomes/homo\_sapiens/GRCh38/star-2.5.3a/

/home/bioinformatics/software/STAR/STAR-2.5.3a/bin/STAR --runThreadN 4 --genomeDir $dbdir --readFilesIn $1

## step2\_align.sh

#! /bin/bash

projectdir=/scratchb/training/data/practicalII

fastqdir=$projectdir/fastq

bamdir=$projectdir/bam

for i in $fastqdir/\*.gz;

do

echo $i;

sample=`basename $i .r\_1.fq.gz | sed s/s\\_//`;

echo $sample;

mkdir $bamdir/$sample;

cd $bamdir/$sample;

sbatch -J $i -t 1-00:00 --mem=4G -o $sample.out -e $sample.err $projectdir/scripts/run\_star.sh $i;

cd $bamdir;

done