Using the computing resources of CSC in NGS data analysis

18.9. 2014

Log in to Taito server and move to your $WRKDIR directory (Replace the *XX* with your training account number) and set up the *biokit* program environment:

```
ssh taito.csc.fi -l trngXX
cd $WRKDIR
module load biokit
```

Download an SRA entry from NCBI. (You can first locate the download link of SRA entry SRR1552105 using the NCBI website and then copy the link location as an argument for the *wget* command. Everything in the command below should be in a one long command line.)

```
wget
ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR155/SRR1552105/SRR1552105.sra
```

Then, convert the sra file to a pair of fastq-files with command:

```
fastq-dump --split-files SRR1552105.sra &
```

The & character in the command line above will make fastq-dump command to run on background so that you can run the commands below while the sra file is processed.

Create a new directory called *p_aeruginosa*. Go to the new directory:

```
mkdir p_aeruginosa
ls -l
cd p_aeruginosa
```

Download genome of *Pseudomonas aeruginosa* genome from with command:

```
ensemblfetch pseudomonas_aeruginosa_pao1
```

Use command *ps* to check, if fastq-dump command is still running. If yes, the wait until the fastq-dump process is ready.

```
ps
```

2. Calculating and back-up copying BWA index

Calculate indexes for BWA mapping tool for this genome.

```
bwa index -p pseudomonas_aeruginosa_pao1 \
Pseudomonas_aeruginosa_pao1.GCA_000006765.1.23.dna.toplevel.fa
```

check the resulting files with command

```
ls -lh
```

Go one step downwards in the directory tree and check the content of the directory :

```
cd ..
ls -l
```

Use *tar* command to pack the directory *p_aeruginosa* into file *p_aeruginosa_.tar* and check the size:

```
tar cvf p_aeruginosa.tar p_aeruginosa
ls -lh
```

Compress the file with *gzip* and check the size again:

```
gzip p_aeruginosa.tar
ls -lh
```

Store a a back-up copy of the BWA indexed P. Aeruginosa genome to HPC archive with command (At the moment, the iput and ils commands do not work in taito-shell.csc.fi. If your wish to try these commands, open connection to taito.csc.fi, move to your $WRKDIR and run the commands below:)

```
iput p_aeruginosa.tar.gz
```

Check the content of archive with command:

```
ils
```

3. Running a BWA batch job

The fastq-files that we have downloaded from NCBI are large and running the BWA alignment for them may take a long time. Move to the $WRKDIR directory and check the content of the fastq-files with commands:

```
cd $WRKDIR
infoseq_summary SRR1552105_1.fastq
infoseq_summary SRR1552105_2.fastq
```

(In real cases you could use commands like *prinseq* and *fastqc* the get a more detailed information about the reads files.)

For testing purposes, split read files into eight smaller files. In a fastq format, each read is described with four rows. Both SRR1552105_1.fastq and SRR1552105_2.fastq contain 95 326 592 rows (23 831 648 reads) thus each split file should include 11 915 824 rows. The splitting can now be done with commands:

```
split -11915824 -d -a 1 SRR1552105_1.fastq reads_1_fastq.
split -11915824 -d -a 1 SRR1552105_2.fastq reads_2_fastq.
```

Now check what files you have with command:

```
ls -lh
```

You can also check some of the new fastq files with the *infoseq_summary* tool.

Next create a test batch job for BWA. Open and new text file called "*bwa_job.bash*" with *nano* (or with some other text editor available in Taito).

```
nano bwa_job.bash
```

Copy the batch job example from the BWA home page of CSC (https://research.csc.fi/-/bwa) to the text editor, and modify it so that it aligns one of the small read sets (reads_1_fastq.0 and reads_2_fastq.0) to the *Pseudomonas aeruginosa* genome just calculated. Modify the batch job settings so that the run time of your job is 30 min (#SBATCH -t 00:30:00) and that it uses test-partition (#SBATCH-p test). The BWA command to be executed is in this case:

```
bwa mem -t 4 p_aeruginosa/pseudomonas_aeruginosa_pao1 \
reads_1_fastq.0 reads_2_fastq.0 > testrun_0.sam
```

You can add following post processing commands to the batch job file.

```
samtools view testrun.sam -b > testrun.bam
```

```
samtools flagstat testrun.bam
```

The samtools commands above will first convert the sam formatted alignment file into bam format and then calculate the basic statistics for the BWA alignment.

Sample batch job file:

```
#!/bin/bash -l
#SBATCH -J bwa
#SBATCH -o output_%j.txt
#SBATCH -e errors_%j.txt
#SBATCH -t 00:30:00
#SBATCH -n 1
#SBATCH --nodes=1
#SBATCH --cpus-per-task=4
#SBATCH --mem-per-cpu=4000
#SBATCH -p test
#
module load biokit
bwa mem -t 4 p_aeruginosa/pseudomonas_aeruginosa_pao1 \
reads_1_fastq.0 reads_2_fastq.0 > testrun_0.sam

samtools view testrun.sam -b > testrun.bam
samtools flagstat testrun.bam
```

In nano editor, you can save the edited file by pressing *Ctrl+O* and close the editor with command *Ctrl+x*.

Once the batch job file is ready, you can submit it with command:

```
sbatch bwa_job.bash
```

You can monitor the progress of your job with commands:

```
squeue -l
squeue -l -u trngXX
```

Study the output file of the batch job. You can check the latest files in your current working directory with command:

```
ls -ltr
```

The contents of the outputfiles can be viewed with e.g. Command *less:*

```
less output_jobnumber.txt
```

Following command lists the wall clock and CPU times used by your recent batch jobs:

```
sacct --format=jobid,elapsed,UserCPU,ncpus,state
```

How long time would be needed to run the alignment for 23 million read pars in the full sized reads files, when we assume that the execution time depends linearly from the amount of reads to be aligned?

4. Running a BWA array job

In NGS data analysis, many tasks are so called embarrassingly parallel (also called as dataparallel). This means that a large analysis task can be divided into several sub-tasks that can be executed independently. These kind of tasks can be very effectively run in a cluster environment.

In Taito, many bioinformatics applications utilize this approach automatically (e.g. Trinity, MISO, pb blast). In other cases, you can use array batch jobs. In the next exercise you will modify the batch job script from the previous example into an *array job file*, so that all the eight read file pairs will get aligned by eight parallel jobs.

Fist make a copy of the normal batch job file:

```
cp  bwa_job.bash  bwa_array_job.bash
```

Then use nano or other editor to modify the bwa_array_job.bash file. To set up an array job, you need to add following batch job definition to the batch job script:

```
#SBATCH –array=from-to
```

In this exercise the *from* value is 0 and the *to* value is 7. This definition will now make the *sbatch* command to launch not just one, but eight batch jobs so that in the jobs the job specific variable $SLURM_ARRAY_TASK_ID will get values from 0 to 7.

Then modify the bwa command so that in stead of using input file names reads_1_fastq.0 reads_2_fastq.0 you use:

```
reads_1_fastq.$SLURM_ARRAY_TASK_ID
```

and

```
reads_2_fastq.$SLURM_ARRAY_TASK_ID
```

Similarly, use following name for the output file:

```
testrun_.$SLURM_ARRAY_TASK_ID.sam
```

After the modifications, launch the batch job with command:

```
sbatch bwa_array_job.bash
```

Monitor the progress of the job with the same commands as in the previous exercise.

More information

- **Bioscience tools at CSC:** https://research.csc.fi/biosciences
- **Working in the CSC environment:** https://research.csc.fi/csc-guide
  - linux basics
  - disks and data transfer
- **Taito user guide:** https://research.csc.fi/taito-user-guide