Bi 621 – Problem Set 8

Part 1 – RNA-seq alignment

The goal of this assignment is to understand how to map reads back to an existing reference genome. We will be working with RNA-seq data obtained from zebrafish ovaries.

1. Create a new directory in **your** projects directory on HPC and symlink the sequenced Illumina paired-end reads from zebrafish ovaries:

```
% mkdir -p /projects/bgmp/YOU/Bi621/PS8
% cd /projects/bgmp/YOU/Bi621/PS8
% ln -s /projects/bgmp/shared/Bi621/dre_WT_ovar12_R1.qtrim.fq.gz
% ln -s /projects/bgmp/shared/Bi621/dre_WT_ovar12_R2.qtrim.fq.gz
```

2. Go to the Ensembl website. Navigate to find the zebrafish reference genome by chromosome (FASTA) and gene set (GTF) respectively:

```
Danio_rerio.GRCz11.dna.chromosome.*.fa.gz
Danio_rerio.GRCz11.97.gtf.gz
```

   These files should be available for download from the FTP server linked on the Ensembl website. For each of these files, use `wget` to download them into their directory inside your project directory on HPC (Hint: you can use the wildcard to get all the chromosome files at the same time). Call this directory `dre`.

3. Install STAR and samtools:

```
% conda activate bgmp_py3
% conda install star -c bioconda
% STAR --version
% conda install samtools -c bioconda
% samtools --version
```

4. Alternatively, use a preexisting module on Talapas (see instructors – we prefer you to try to install STAR yourself, but if you are having problems, we'll show you a module to use).

5. Build a STAR database out of the reference sequence using the STAR program using `--runMode genomeGenerate`. This will k-merize your genome to build a STAR database. Question: Does "`--runMode genomeGenerate`" take compressed files as input? Do not forget to record resource usage with "`/usr/bin/time -v`". You will need to specify all of the following options (see the STAR manual):

```
--runThreadN        7
--runMode           genomeGenerate
--genomeDir         Should include "97", "Danio_rerio.GRCz11", and STAR
                    version number. Why?
--genomeFastaFiles  ???
--sjdbGTFfile       ???
```

6. Run STAR to to align the reads to the reference genome. (Be sure to use the queuing system and request 7 cores on 1 node.)

```
/usr/bin/time -v STAR --runThreadN 7 --runMode alignReads \
    --outFilterMultimapNmax 3 \
    --outSAMunmapped Within KeepPairs \
    --alignIntronMax 1000000 --alignMatesGapMax 1000000 \
    --readFilesCommand zcat \
    --readFilesIn <reads_file1> <reads_file2> \
    --genomeDir <genome_directory> \
    --outFileNamePrefix <output_filename_prefix>
```

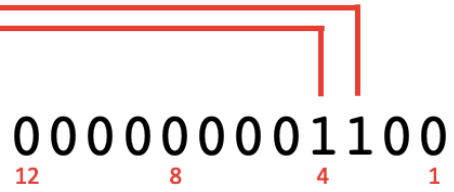
7. View the results of the alignment with `less`.
8. Using `samtools` (enter `samtools` on the command line with no options to get a help screen), convert the SAM file to BAM format. Sort it, and extract all reads from chromosome 1 into a new SAM file. Report how many reads are on chromosome 1.

Part 2 – Python and SAM

1. Write a python program to parse the contents of your BAM file. Use your original file, not the file filtered for chromosome 1.
   The following statement will check if the current read is mapped:

```
if((flag & 4) != 4):
  mapped = True
```

Bit		Description
1	0x1	template having multiple segments in sequencing
2	0x2	each segment properly aligned according to the aligner
4	0x4	segment unmapped
8	0x8	next segment in the template unmapped
16	0x10	SEQ being reverse complemented
32	0x20	SEQ of the next segment in the template being reverse complemented
64	0x40	the first segment in the template
128	0x80	the last segment in the template
256	0x100	secondary alignment
512	0x200	not passing filters, such as platform/vendor quality controls
1024	0x400	PCR or optical duplicate
2048	0x800	supplementary alignment



2. The program should count up the number of reads that are properly mapped to the reference genome and the number of reads that are not mapped to the genome.

   Note: you may encounter each read in a file more than once. This will occur when you have multiple alignments for a single read. You should be careful not to count reads as *aligned* more than once.

To turn in your code for this assignment, do the following:

Submit your scripts and any output from your Python script to GitHub. Be sure to include any bash commands you use.