Identification of RNAseq reads

In this practical we will use several command line tools to identify the transcript(s) that correspond to the RNAseq reads in a file called **X.fastq**

A list of tools to use is provided, but how to actually use the tools is up to you to find out ☺ (check the help, read the manuals, use google)

Materials

Data

Sequence reads in fastq format, single end:
http://www.bioinformatics.nl/courses/BIF-30806/X.fastq

Files in directory `/local/data/course/genomes/Arabidopsis_thaliana/`
Arabidopsis thaliana TAIR 10:

genome: `genome.fa`
annotation: `genes.gtf`

Bowtie2 index: `Bowtie2Index`
HISAT2 index: `TAIR`
Kallisto index: `TAIR.idx`

Tools

bowtie2, /local/prog/samtools/samtools, /local/prog/hisat2/hisat2, /local/prog/stringtie/stringtie, /local/prog/trinity/Trinity
http://www.broadinstitute.org/igv/
makeblastdb, blastn,blastp,blastx,tblastn,tblastx

Protocol

Example command line usages of hisat2, stringtie and samtools, you can find in this paper:
http://www.nature.com/nprot/journal/v11/n9/full/nprot.2016.095.html

Practical

On **altschul.bioinformatics.nl**:
Make a directory in your home directory called **`mapping`**, `cd` to that directory and create soft links to the indices and genome files using this command:

```
ln -s /local/data/course/genomes/Arabidopsis_thaliana/* ./
```

1. Map the (single end) reads in X.fastq to the *Arabidopsis* genome using **Bowtie2**, let the output be SAM format and as index choose **Bowtie2Index/genome**. Use default settings. Record the number of mapped and unmapped reads.
2. Create a sorted BAM file from the SAM file using **samtools**.
3. Create an index for the sorted BAM file using **samtools**.
4. Start the IGV genome browser and load the *A. thaliana* TAIR10 genome ("Load Genome From Server").
5. Load the BAM file in IGV (with the index) and check which isoforms are detected (find the right genome coordinates in the SAM file). To see the

isoforms, you might have to right click on the gene track and change **Collapsed** to **Expanded.**

6. Repeat steps 1-5 using **HISAT2** as the aligner, as index use **TAIR** (this will use the TAIR.*.ht2 files). In IGV check the 'cigar' values for some of the new informative reads.
    a. Which tool performs the best?
    b. Were these reads strand specific?
7. Use **StringTie** to predict the transcripts from the **HISAT2** BAM output, use the genes.gtf annotation file. Check the resulting GTF file in IGV, do the predicted transcript correspond with the mapped reads?
8. Use **Trinity** to do a *De Novo* assembly of the reads, using default settings. The results of the assembly will be in: **`trinity_out_dir/Trinity.fasta`**
9. Make a Blast database called **TAIR10** from the *Arabidopsis* TAIR 10 genome.
10. Use **Blast** to search for the **`Trinity.fasta`** sequences in the TAIR10 database. Use an E-value cut-off of `1E-10`, choose output format 7 (tabular).
11. Write a python script to convert the output to a valid GFF file, like below:

```
Chr1  blast  exon  1234567  1234569  .  +  .  gene=Unknown
```

12. Load the GFF file into IGV and look at the resulting transcripts (zoom in to the right coordinates). Did Trinity do a good job? Why?
13. Use **kallisto** to "pseudoalign" the reads to the *Arabidopsis* transcriptome. The **TAIR.idx** index for that is already available. Set "length mean" to 200 and sd to 20, and specify an output directory.
14. The **abundance.tsv** file contains the results of the **kallisto** run, which transcripts did **kallisto** identify?
15. What is your conclusion after using all these tools to identify the transcripts that produced the reads in **X.fastq**?