STAR Usage Document

Wen Tang Statistical Bioinformatics Lab at CWRU

8/10/2021

STAR Introduction

STAR is an ultrafast universal gene aligner. Mapping of large sets of high-throughput sequencing reads to a reference genome is one of the foundational steps in sequencing data analysis. The STAR software package performs this task with high levels of accuracy and speed. STAR generates output files that can be used for many downstream analyses such as transcript/gene expression quantification, differential gene expression, novel isoform reconstruction, and signal visualization. STAR is open source software that can be run on Unix, Linux, or Mac OS X systems.We mainly introduce how to use it on CWRU HPC. For more details on how to use STAR, see https://github.com/alexdobin/STAR.

Load STAR

After you install STAR on CWRU HPC, you can use the code as follow to load STAR

```
module load STAR
STAR --options
```

Using STAR on CWRU HPC

Basic STAR workflow consists of 2 steps

- Generating genome index files
- Mapping reads to genome

Generating genome index files

If you do **not** have the reference genome ready, you should first generate STAR index/reference genome. The following is an example to generate human genome index files.

```
module load STAR
STAR --runThreadN 6 \
     --runMode genomeGenerate \
     --genomeDir /path/to/output/folder/GRCh38 \
     --genomeFastaFiles /path/to/raw_fa_file/GRCh38.primary_assembly.genome.fa \
     --sjdbGTFfile /path/to/gtf/gencode.v29.annotation.gtf
```

Once you have built the reference genome you will never run it again. You can use the reference genome to map directly next time.

Here are locations of the two sets of already-built reference genomes that are commonly used:

Human (GRCh38): */mnt/pan/SOM_PQHS_HXF155/software_download/STAR_ref/GRCh38*

Mouse (GRCm39): */mnt/pan/SOM_PQHS_HXF155/software_download/STAR_ref/GRCm39*

Running mapping jobs

slurm script Becasue you can **not** running jobs on head nodes on CWRU HPC, you have to write a slurm script to submit your jobs and then CWRU HPC will allocate a node to run your job. Below is an example of a slurm script.

- content of slurm script

```
#!/bin/bash
#SBATCH --time=20:00:00
#SBATCH -N 2
#SBATCH -c 6
#SBATCH --mem-per-cpu=30G
#SBATCH --mail-user=xxx123@case.edu
#SBATCH --mail-type=ALL
module load STAR
STAR --runThreadN 10 \
--genomeDir /mnt/pan/SOM_PQHS_HXF155/software_download/STAR_ref/GRCm39 \
--readFilesCommand zcat \
--readFilesIn /path/to/read1.fq.gz /path/to/read2.fq.gz \
--outSAMtype None \
--quantMode GeneCounts \
```

`--readFilesCommand`: When the read in files are compressed, use `zcat` or `gunzip -c` option.

`--outSAMtype`: Output different type of files. `None` option: no SAM/BAM output; `BAM` option: output BAM without sorting; `SAM` option: output SAM without sorting; `SortedByCoordinate` option: sorted by coordinate. This option will allocate extra memory for sorting which can be specified by limitBAMsortRAM.

`--quantMode`:Count number reads per gene while mapping.

Note:In command `--readFilesIn`,for paired-end reads, use space to separate list for read1 and read2; for different samples, use comma to separate them.

- submit your jobs

```
sbatch <script_name>
# you will get a slurm job-id after you submiting your script
# you can use squeue --job=<jobid> to check your job
```

Mapping result

- check mapping rate

```
less Log.final.out
# look up  "Uniquely mapped reads %", if the rate is higher than 80%, it means that the alignment is go
```

- check mapping result

```
head ReadsPerGene.out.tab
# column 1: gene ID
# column 2: counts for unstranded RNA-seq
# column 3: counts for the 1st read strand aligned with RNA
# column 4: counts for the 2nd read strand aligned with RNA
```

Example

```
[wxt175@hpc4 ~]$ head Log.final.out
                                 Started job on |       Jul 30 17:06:24
                              Started mapping on |       Jul 30 17:06:38
                                     Finished on |       Jul 30 17:13:26
            Mapping speed, Million of reads per hour |       207.16

                                Number of input reads |       23478198
                            Average input read length |       300
                                         UNIQUE READS:
                        Uniquely mapped reads number |       21532128
                             Uniquely mapped reads % |       91.71%
[wxt175@hpc4 ~]$ head ReadsPerGene.out.tab
N_unmapped      1184133 1184133 1184133
N_multimapping  761937  761937  761937
N_noFeature     1815517 11269103        11345951
N_ambiguous     1085068 190697  189610
ENSMUSG00000102693.2    0       0       0
ENSMUSG00000064842.3    0       0       0
ENSMUSG00000051951.6    479     267     212
ENSMUSG00000102851.2    0       0       0
ENSMUSG00000103377.2    0       0       0
ENSMUSG00000104017.2    0       0       0
[wxt175@hpc4 ~]$
```