bash_scripts

Agneesh Barua

2022-09-02

This document contains all the bash scripts used for processing raw reads, *de novo* transcriptome assembly, removing symbiont sequences, constructing orthogroups, and RNA-seq quantification.
A text file with all sra ids or file names was used to set up bash array jobs. The text file was *namelist_##* depending on the type of analysis. For example, data from Heteractis, Stichodactyla, Entacmea, and Macrodactyla were obtained in this study and they were not included during the downloading step. They were included from the quality control step to ortholog construction.

Downloading files form SRA

```
export PATH=$PATH:/apps/unit/MikheyevU/Agneesh/sratoolkit.2.10.8-centos_
    linux64/bin

#Bulk download (adjust bash array job parameters for bulk)
name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist_sra.txt)

prefetch $name
fasterq-dump $name
```

Quality control

```
###fastQC###

fastqc -t 2 /bucket/LaudetU/Agneesh/Anemone_project/fastq_files/*.fastq -o ./
    report

###Trimmomatic###

#adjust array parameters depending on input

module load Trimmomatic/0.39
trimmomatic PE\
        -threads 10 -phred33\
        /bucket/LaudetU/Agneesh/Anemone_project/fastq_files/$name"_1.fastq"\
        /bucket/LaudetU/Agneesh/Anemone_project/fastq_files/$name"_2.fastq"\
        ./Trimmed_reads/$name"_1_paired.fastq"\
        ./Trimmed_reads/$name"_1_unpaired.fastq"\
        ./Trimmed_reads/$name"_2_paired.fastq"\
        ./Trimmed_reads/$name"_2_unpaired.fastq"\
        ILLUMINACLIP:TruSeq3-PE.fa:2:30:10:8:keepBothReads LEADING:3 TRAILING
            :3 MINLEN:36

#if single end reads
```

```
trimmomatic SE\
        -threads 10 -phred33\
        /bucket/LaudetU/Agneesh/Anemone_project/fastq_files/$name".fastq"\
        ./Trimmed_reads/$name"_trimmed.fastq"\
        ILLUMINACLIP:TruSeq3-PE.fa:2:30:10:8:keepBothReads LEADING:3 TRAILING
            :3 MINLEN:36
```

Transcriptome assembly

```
###Trinity###

module load Trinity/2.8.4

name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt) #all fastq files

Trinity --seqType fq\
        --max_memory 100G\
        --CPU 10\
        --min_contig_length 300\
        --min_kmer_cov 2\
        --left ../Trimmed_reads/Modified_reads/"mod_"$name"_1_paired.fastq.gz"
            \
        --right ../Trimmed_reads/Modified_reads/"mod_"$name"_2_paired.fastq.gz
            "\
        --output $name"_trinity_output"\
        --full_cleanup

###CD-HIT-EST
module load cd-hit/2016-0304

name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt)
path="/flash/LaudetU/Agneesh/Anemone_host_project/03.Assembly/Combined_
    transcriptomes/Trinity_fasta"


cd-hit-est\
        -i ${path}/$name"_trinity_output.Trinity.fasta"\
        -o $name"_cdhit_out.fa"\
        -c 0.95\
        -n 10
```

Filter symbiont reads

```
###GC filtering###
name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt)
path="/flash/LaudetU/Agneesh/Anemone_host_project/04.CD-HIT-EST/res_cdhit"

#use the seqkit utility for this step
cdhit_out=$(cat ${path}/$name"_cdhit_out.fa" | ~/seqkit fx2tab -g -n -i | sort
     -k2 -n | awk '{if␣($2␣<␣60)␣print␣$1;}')
echo -e $cdhit_out$ > $name"_names_to_get.txt"
~/seqkit grep -f $name"_names_to_get.txt" ${path}/$name"_cdhit_out.fa" -o ./
    filter_out/$name"_GC_filter_out.fasta"
```

```
#for checking number of sequences filtered
T=$(grep -c ">" ${path}/$name"_cdhit_out.fa")
R=$(grep -c ">" ./filter_out/$name"_GC_filter_out.fasta")

echo "filtered out" $(($T-$R)) "transcripts"

###Blast filtering###
name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt)
path="/flash/LaudetU/Agneesh/Anemone_host_project/05.GC_content_filtering/
    filter_out"

module load ncbi-blast/2.10.0+

blastn -query ${path}/$name"_GC_filter_out.fasta"\
        -db symbiodinium_db\
        -evalue 1e-10\
        -max_hsps 5\
        -num_threads 10\
        -out ./blast_out/$name"_blast_res.tsv"\
        -outfmt "6 qseqid sseqid qcovhsp pident evalue"


cat ./blast_out/$name"_blast_res.tsv" |
  awk '{if ($3 >= 30 && $4 >= 50 && $2 ~ /^symbC/) print $1;}' > ./symb_hits/$
      name"_symbhits.txt"

~/seqkit grep -v -f ./symb_hits/$name"_symbhits.txt" ${path}/$name"_GC_filter_
    out.fasta"\
        -o ./filtered_out_seqs/$name"_symb_filter_out.fasta"
```

Getting ORFs and predicted proteins

```
###Transdecoder steps###
name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt)
path="/flash/LaudetU/Agneesh/Anemone_host_project/06.BLAST_filtering/filtered_
    out_seqs"

/apps/unit/LaudetU/TransDecoder-TransDecoder-v5.5.0/TransDecoder.Predict --
    single_best_only -t ${path}/$name"_symb_filter_out.fasta"\
        --retain_pfam_hits $name"_pfam.domtblout"\
        --retain_blastp_hits $name"_blastp.outfmt6"


/apps/unit/LaudetU/TransDecoder-TransDecoder-v5.5.0/TransDecoder.LongOrfs -t $
    {path}/$name"_symb_filter_out.fasta" -m 100


###Blastp###
module load ncbi-blast/2.10.0+

#download swissprot for blast
#update_blastdb.pl --decompress swissprot
blastp -query $name"_symb_filter_out.fasta.transdecoder_dir/longest_orfs.pep"\
        -db ./database/swissprot\
        -max_target_seqs 1\
        -outfmt 6\
        -evalue 1e-10\
        -num_threads 10 > $name"_blastp.outfmt6"

###HMMERScan###
module load hmmer/3.1b2

hmmscan --cpu 10 --domtblout $name"_pfam.domtblout" ./pfam_database/Pfam-A.hmm
    $name"_symb_filter_out.fasta.transdecoder_dir/longest_orfs.pep"

###CD-HIT###
module load cd-hit/2016-0304

path="/flash/LaudetU/Agneesh/Anemone_host_project/07.Transdecoder/pep_predict_
    out"

cd-hit \
        -i ${path}/$name"_symb_filter_out.fasta.transdecoder.pep"\
        -o $name"_cdhit_pep_out.fa"\
        -c 0.95\
        -n 5\
        -M 9000\
        -T 10

###Getting longest isoform###
/apps/free81/Trinity/2.8.4p/util/misc/get_longest_isoform_seq_per_trinity_gene
    .pl\
         res_cdhit/$name"_cdhit_pep_out.fa" > longest_pep_out/$name"_longest_
            pep.fa"
```

BUSCO check

```
module load BUSCO/4.1.2
module load ruse

path="/flash/LaudetU/Agneesh/Anemone_host_project/08.CD-HIT/longest_pep_out"

ruse busco -m proteins\
        -i ${path}/$name"_longest_pep.fa"\
        -l metazoa_odb10\
        -c 10\
        -o $name"_busco_out"\
        -f
```

OrthoFinder

Follow the pre-processing steps in the OrthoFinder manual and run the following command

```
export PATH=$PATH:/apps/unit/MikheyevU/Agneesh/OrthoFinder
orthofinder -t 70 -I 1.7 -f primary_transcripts/
```

Making species tree

```
name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist_orths.txt) #change to fit
    OrthoFinder file names
path="/flash/LaudetU/Agneesh/Anemone_host_project/11.OrthoFinder/primary_
    transcripts/OrthoFinder/Results_Mar09/Single_Copy_Orthologue_Sequences"

###MAFFT alignment###
module load mafft/7.305

mafft-linsi --thread 14 ${path}/$name".fa" > $name"_aling_output.fa"

###TrimAL step###
path="/flash/MikheyevU/Agneesh/origin_of_specialization/comprative_genomics/
    Orthologs/primary_transcripts/OrthoFinder/Making_datasets_with_orths/
    TrimAL"

${path}/trimal/source/trimal -in alignments/all_seqs_align/$name"_aling_output
    .fa" -out trimmed_reads/all_seqs/$name"_aln_timmed.fa" -phylip -automated1

###IQTree2###
module load IQ-TREE/2.1.3

iqtree2 -s trimmed_reads/all_seqs/ -B 1000 -T 40 --mem 500G -msub nuclear
```

Quantifying RNA-seq with Kallisto

```
###make index###
module load kallisto/0.46.1

name=$(sed -n "$SLURM_ARRAY_TASK_ID"p namelist.txt)

kallisto index -i $name".idx" ../index_transcripts/$name"_index_input_assembly
    .fasta"
#index_transcripts contain the final assembled transcriptome

###Quantification step###

For kallisto we used a simple Snakefile to run the quantification step. The
    Snakefile was already ready and optimised.

#Wildcards------------------------------------------------------------
SAMPLES = ["Smer_2_S8_L001",
"Smer_2_S8_L001",
"Smer_3_S9_L001",
"Smer_3_S9_L001",
"Smer_4_S16_L001",
"Smer_4_S16_L001",
"Smer_6_S12_L001",
"Smer_6_S12_L001"]

#Rules----------------------------------------------------------------

rule all:
        input: expand("results2/{samples}",samples = SAMPLES)
```

```
rule kallisto_quant:
        input: r1 = "/bucket/LaudetU/Agneesh/Anemone_project/fastq_files/
            Stichodactyla_fastq_files/{samples}_R1_001.fastq.gz",
                r2 = "/bucket/LaudetU/Agneesh/Anemone_project/fastq_files/
                    Stichodactyla_fastq_files/{samples}_R2_001.fastq.gz",
                idx ='../../indices/Stichodactyla_mertensi.idx'
        output:
                 directory("results2/{samples}")
        threads: 10
        shell: "kallisto quant -i {input.idx} -o {output} -b 100 {input.r1} {
            input.r2}"
```

The construction of the composite Symbiodiniaceae index was done using the above code. Pseudoalignment and quantification of Symbiodiniaceae composition was also done using the above code.