Analysis of CLASH data using a Snakemake pipeline Hyb-CRAC-R

Ignatius Pang

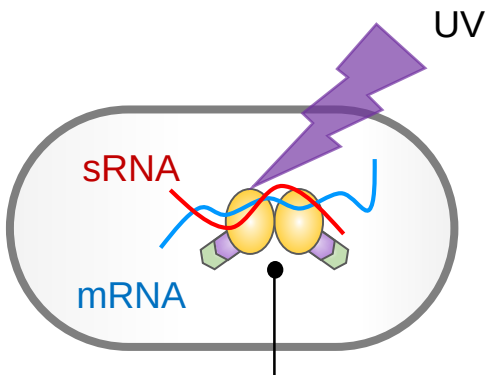
23rd April 2021 (Updated)

Outline

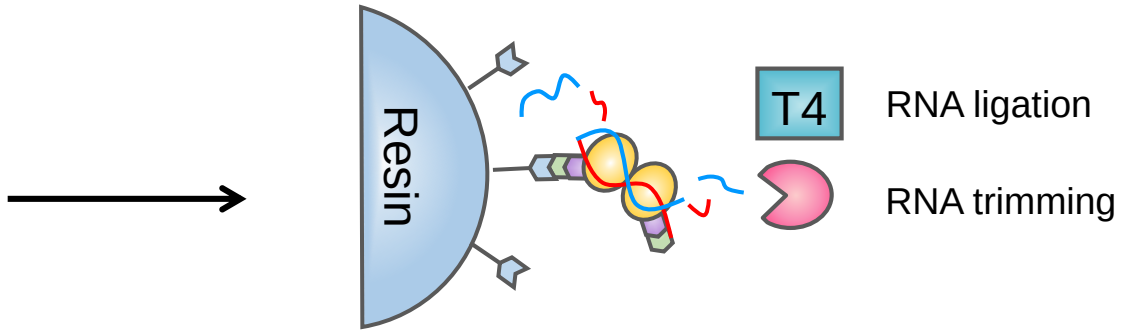
- Introduction to the CLASH experimental protocol
- Go through the steps of the CLASH pipeline
- Go through some concepts in writing Snakemake pipeline
- How to configure the parameters for a new experiment

Cross-linking, ligation and sequencing of hybrids (CLASH)

CLASH: method to capture and map RNA-RNA interactions using a RNA-binding protein as bait

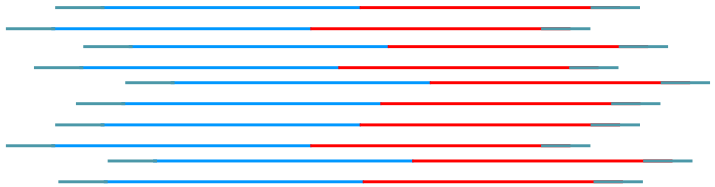


RNase III: Processes ~75% of *S. aureus* transcripts (dsRNA)

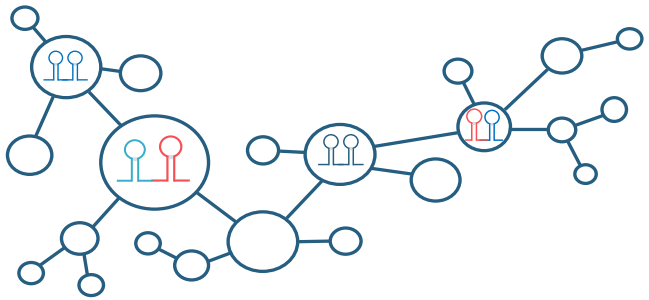


Purification and intermolecular ligation

Barcoding and PCR Amplification



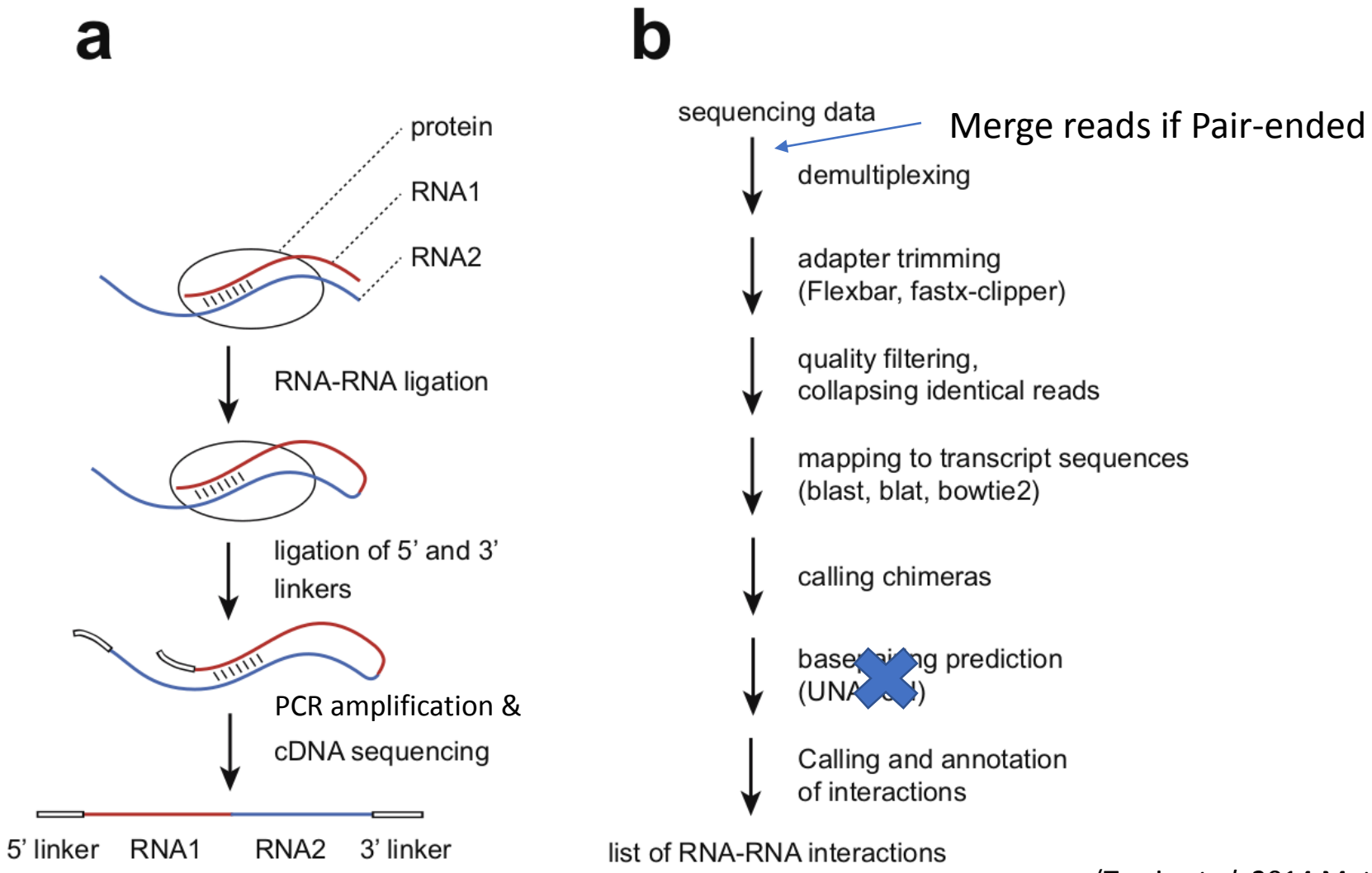
Deep-sequencing of RNA-RNA interactions



Global map of sRNA interactome

Figures courtesy of Winton Wu

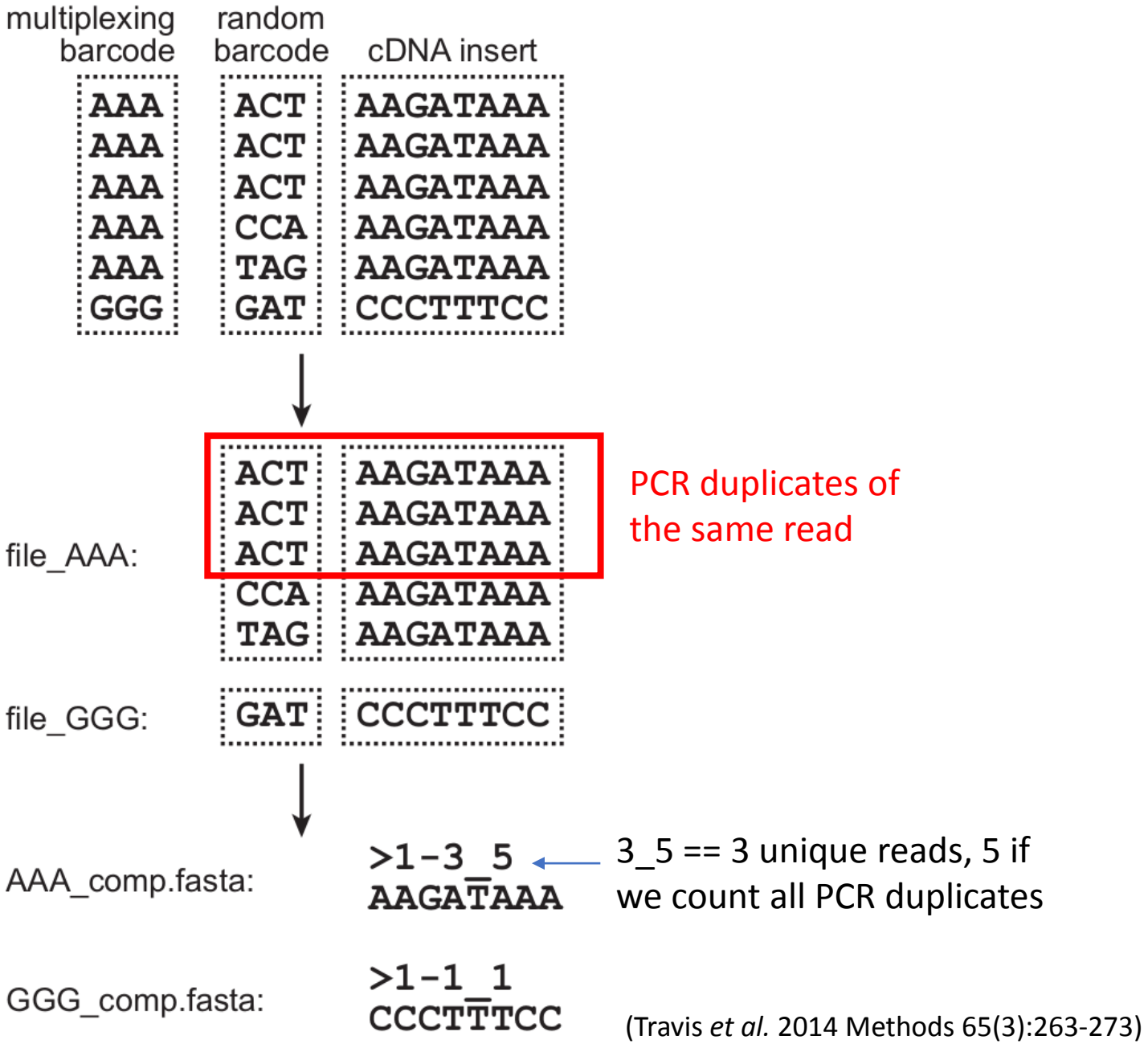
Overall Workflow



(Travis *et al.* 2014 Methods 65(3):263-273)

Fig. 1. Schematic of CLASH experiment and hyb analysis pipeline.

Demultiplexing and Identification of Unique Reads versus PCR duplicate reads



(Travis *et al.* 2014 Methods 65(3):263-273)

What is Snakemake (in my own words)?

- Workflow management toolkits (e.g. Nextflow, ruffus, Snakemake)
- Automation – like a recipe to run the pipeline, with restart checkpoints

- Define a list of Input and Output files (nodes in the network)
- Define files using “Wildcards”
- Create a network of dependencies
- Run jobs based on available files and dependencies

Example of Wildcards

- Suits = [“Diamond”, “Club”, “Heart”, “Spade”]
- Ranks = [“A”, ”2”, ”3”, ”4”, ”5”, ”6”,”7”,”8”,”9”,”10”,”J”,”Q”,”K”]

- All_cards= expand("{rank}_of_{suit}.card"), suit= Suits, rank= Ranks),

- {suit} and {rank} are the wildcards
- What if we add {colour}? There are no black diamond?
  - Merge rank and color (e.g. {suit_colour}) or
  - Use python lookup tables to avoid wrong combinations

Example Snakemake Command

```
snakemake -j 12 \
--printshellcmds \
--snakefile jw_bc1.smk \
--cluster-config  cluster.json \
--latency-wait 20 --use-envmodules \
--cluster "qsub -N {cluster.N} -l
nodes={cluster.nodes}:ppn={cluster.ppn},walltime={cluster.walltime},mem={clu
ster.mem} -M {cluster.email} -m ae -j oe"
```

Number of parallel jobs that can run at the same time

Print to STDOUT

Your snakemake file

Json file defines the Katana PBS parameters

20 secs wait between jobs, and use PBS module load system

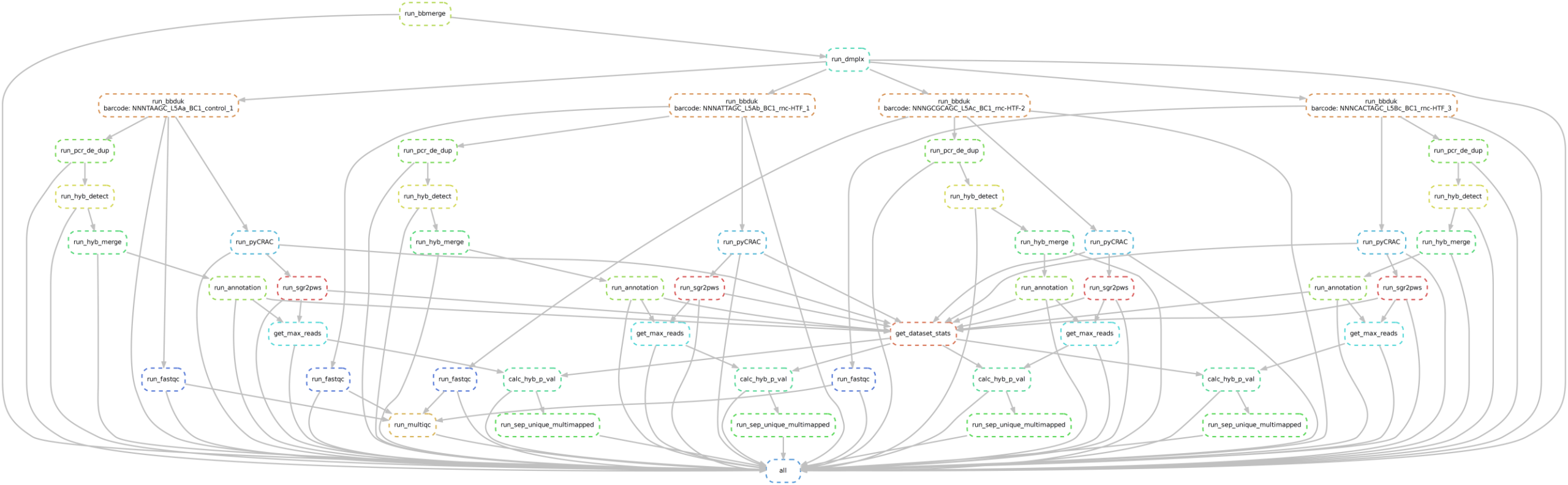
The Katana PBS ‘qsub’ command, with parameters from JSON file

We will focus on how we can adjust the file jw_bc1.smk to perform the analysis.

Rule Graph

```
snakemake \
    --snakefile jw_bc1.smk \
    --rulegraph \
    | dot -Tpdf \
    > rulegraph.pdf
```

Directed Acyclic Graph (DAG)



* A directed acyclic graph of all the steps

```
snakemake \
--snakefile jw_bc1.smk \
 --dag \
 | dot -Tpdf \
 > dag.pdf
```

Parameters we need to adjust - Input FASTA Files

```
## Input Samples Parameter
FASTQ_DIR="/srv/scratch/treelab/JJT_fastq/staph_CLASH_may_2017/WON3452_20170510_Rnc_Pr
OUTPUT_DIR="/srv/scratch/treelab/igy/Data/CLASH_JKD6009/JW_BC1/Processed_Data"


FILES_R1=["JKD60091-rncHTF-1-2-3-ATCACG_S1_L001_R1_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L002_R1_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L003_R1_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L004_R1_001.fastq.gz" ]

FILES_R2=["JKD60091-rncHTF-1-2-3-ATCACG_S1_L001_R2_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L002_R2_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L003_R2_001.fastq.gz",
"JKD60091-rncHTF-1-2-3-ATCACG_S1_L004_R2_001.fastq.gz"]
```

Updated

Parameters we need to adjust - Input Barcode File

```
ORGANISM="Sa_JKD6009"
REF_DIR= os.path.join(BASE_DIR, "Data", "Ref_Seq")
LINKER_FILE = os.path.join(REF_DIR, "linker.fa")
BARCODE_FILE = os.path.join(REF_DIR, "JW_BC1_CLASH_barcodes.tab")
BARCODE_NAMES = [ "NNNTAAGC_L5Aa_BC1_control_1", "NNNATTAGC_L5Ab_BC1_rnc-HTF_1",
                  "NNNGCGCAGC_L5Ac_BC1_rnc-HTF-2", "NNNCACTAGC_L5Bc_BC1_rnc-HTF_3" ]
```

NNNTAAGC	L5Aa_BC1_control_1
NNNATTAGC	L5Ab_BC1_rnc-HTF_1
NNNGCGCAGC	L5Ac_BC1_rnc-HTF-2
NNNCACTAGC	L5Bc_BC1_rnc-HTF_3

JW_BC1_CLASH_barcodes.tab

Updated

Other Parameters We Need To Adjust

```
## Genomic parameters
GENOME_DIR="/srv/scratch/treelab/genome/saus/JKD6009"
GTF_FILE_DIR =  os.path.join(GENOME_DIR, "Hyb_pyCRAC")
pyCRAC_GTF_FILE = os.path.join(GTF_FILE_DIR,  "GCF_900607245.1_JKD6009_genomic_merge_features_simple_annot_checked_edited.gtf")
GENOME_SIZE =  os.path.join(GENOME_DIR,  "Sa_JKD6009.genome")
NOVO_INDEX =   os.path.join(GENOME_DIR,  "Sa_JKD6009.nix")
HYB_DB_NAME = "Sa_JKD6009"
RNA_TYPE_PRIORITIES =  os.path.join(GTF_FILE_DIR,  "rna_type_priority.tab")

## Source Directories
SOURCE_DIR= os.path.join( BASE_DIR, "Source" )
R_SCRIPTS_DIR=os.path.join(SOURCE_DIR, "Demultiplex", "scripts" )
ANNOTATION_SCRIPT_DIR = os.path.join(SOURCE_DIR, "Jai_Script", "Annotation" )
STATS_SCRIPT_DIR = os.path.join(SOURCE_DIR, "Jai_Script", "Hyb_stats" )
PYCRAC_SRC_DIR = os.path.join(SOURCE_DIR, "Jai_Script", "pyCRAC" )

## Location of CRAC installation environment
CRAC_ENVIRONMENT_ACTIVATE="/srv/scratch/z3371724/my_python_dir/my_python_env/bin/activate"
```

The pyCRAC environment needs to replaced with a new one that you've installed yourself. (see file StaphCLASH2020/Source/Jai_Script/using_pyCRAC_on_katana.sh)

Replace the CRAC_ENVIRONMENT_ACTIVATE parameter containing the code which activate python virtualenv environment for pyCRAC scripts:

. /srv/scratch/z3371724/my_python_dir/my_python_env/bin/activate

Updated

Within the script CRAC_pipeline_SE.py, replace this line in the ‘cmd’ parameter

```
def trimAndCollapse(inputfile,outputfile):
  """ Removes the last random nucleotide from the forward read, collapses the data and then splits it again into fasta files """
  cmd = '/srv/scratch/z3371724/StaphCLASH2020/Source/Jai_Script/pyCRAC/TrimNucs.py -n 1 --addtoheader -f '%s' | pyFastqDuplicateRemover.py -o '%s''
  logger.info(cmd)
  os.system(cmd)
  #os.system("touch %s" % outputfile)
```

- Replace the ‘cmd’ parameter within the “trimAndCollapse” function of the script CRAC_pipeline_SE.py
- Use the absolute path of the TrimNucs.py script within the GitHub repository (to override and not use the original pyCRAC script)
- For reason not yet determined, the script does not seem to work unless we use an absolute path.
- Would be good to fix this so we do not need to use the absolute path.

Updated

Begin With the End in Mind

- The “rule_all” is like a giant checklist for Snakemake
- Only make files that are not already exist in the directories
- Many files that have a pattern are defined using the wildcards. For example, the {barcode} wildcard.

List Out All the Relevant Files in Rule All (part 1)

```
rule all:
  input:
    sample_r1 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R1),
    sample_r2 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R2),
    linker = LINKER_FILE,
    all_fa = os.path.join(OUTPUT_DIR,  "Merged", "all.fq"),
    ihist = os.path.join(OUTPUT_DIR, "Merged", "ihist.txt"),
    barcodes_file = BARCODE_FILE,
    fastq_file= expand(  os.path.join(OUTPUT_DIR, "Demultiplexed", "all_{barcode}.fastq"), barcode=BARCODE_NAMES),
    trimmed_files = expand( os.path.join(OUTPUT_DIR,  "Trimmed", "all_{barcode}.trimmed.pyCRAC.fq"), barcode=BARCODE_NAMES),
```

There are other rules here but not shown here as it is too long.

Updated

List Out All the Relevant Files in Rule All (part 2)

Input for BBMerge

```
sample_r1 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R1),
sample_r2 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R2),
```

Output from BBMerge and Input for Demultiplexing

```
all_fa = os.path.join(OUTPUT_DIR,  "Merged", "all.fq"),
```

Output from Demultiplexing and Input for Trimming

```
fastq_file= expand(  os.path.join(OUTPUT_DIR, "Demultiplexed", "all_{barcode}.fastq"),
barcode=BARCODE_NAMES),
```

Output from Trimming and Input for PCR deduplication

```
trimmed_files = expand( os.path.join(OUTPUT_DIR,  "Trimmed",
"all_{barcode}.trimmed.pyCRAC.fq"), barcode=BARCODE_NAMES),
```

Updated

There are other rules in ‘rule all’ but not shown here, it would be too long.

1. Merge Paired-end Reads

- BBMerge
- Provide adapter sequence in a FASTA file
- Keep the Merged and Unmerged reads and concatenate them into a single file (Not throw away data)

Defining Input and Output Files Within the Bbmerge Rule

```
rule run_bbmerge:
  input:
    sample_r1 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R1),
    sample_r2 = expand(os.path.join( FASTQ_DIR, "{fastq}"), fastq=FILES_R2),
    linker = LINKER_FILE
  output:
    all_fa = os.path.join(OUTPUT_DIR, "Merged", "all.fq"),
    ihist = os.path.join(OUTPUT_DIR, "Merged", "ihist.txt"),
  shell:
    """
   cat {input.sample_r1} > $TMPDIR/sample_r1.gz
   cat {input.sample_r2} > $TMPDIR/sample_r2.gz
```

And then the rest of the codes is the run bbmerg using $TMPDIR/sample_r1.gz and $TMPDIR/sample_r2.gz

2. Demultiplexing

- Use pyCRAC's pyBarcodeFilter.py
- Notes to install pyCRAC on Katana using python virtuaenv available
- Barcode file

NNNTAAGC	L5Aa_BC1_control_1
NNNATTAGC	L5Ab_BC1_rnc-HTF_1
NNNGCGCAGC	L5Ac_BC1_rnc-HTF-2
NNNCACTAGC	L5Bc_BC1_rnc-HTF_3

Defining Input and Output Files Within the Demultiplexing Rule

```
rule run_dmplx:
  input:
    all_fa = os.path.join(OUTPUT_DIR, "Merged", "all.fq"),
    barcodes_file = BARCODE_FILE,
  output:
    os.path.join(OUTPUT_DIR, "Demultiplexed", "all_others.fastq"),
    expand( os.path.join(OUTPUT_DIR, "Demultiplexed", "all_{barcode}.fastq"), barcode=BARCODE_NAMES),
    os.path.join(OUTPUT_DIR, "Demultiplexed", "all_barcode_statistics.txt"),
    os.path.join(OUTPUT_DIR, "Demultiplexed", "all_random_nucleotide_statistics.txt"),
```

3. Reads Trimming & QA

Trimming

- Trim away poor quality sequence regions or adapters
- Note additional parameters in the Snakemake file

QA

- Run FASTQC
- Run MultiQC to view all fastQC output at the same time
- Check for over-represented sequences (could be adapters not cleaned properly)

4. PCR De-duplication

- The hybrid reads were tagged with barcodes prior to PCR amplification
- Each barcode contain two sections

>L5Ab_BC1_rnc-HTF_1

NNNATTAGC

- Random triplet nt (helps identify and remove PCR duplicates)
- ACAATTAGC
- GTCATTAGC
- ACAATTAGC

PCR duplicates of the same read

- 5 – 7 nt of known barcode sequence (tell apart different samples)

5 – 7. Analyze Hybrids

- Use the Hyb pipeline to identify the two-halves of the hybridized read
- Blast of each of the two halves of the hybrid sequence
- Find genomic location of each halve
- Multiple reads representing the same RNA-RNA interactions were merged
- Assign each halve to transcriptome annotations (sRNA, 5’UTR, 3’UTR, coding region)

Annotation Step Requires GTF file

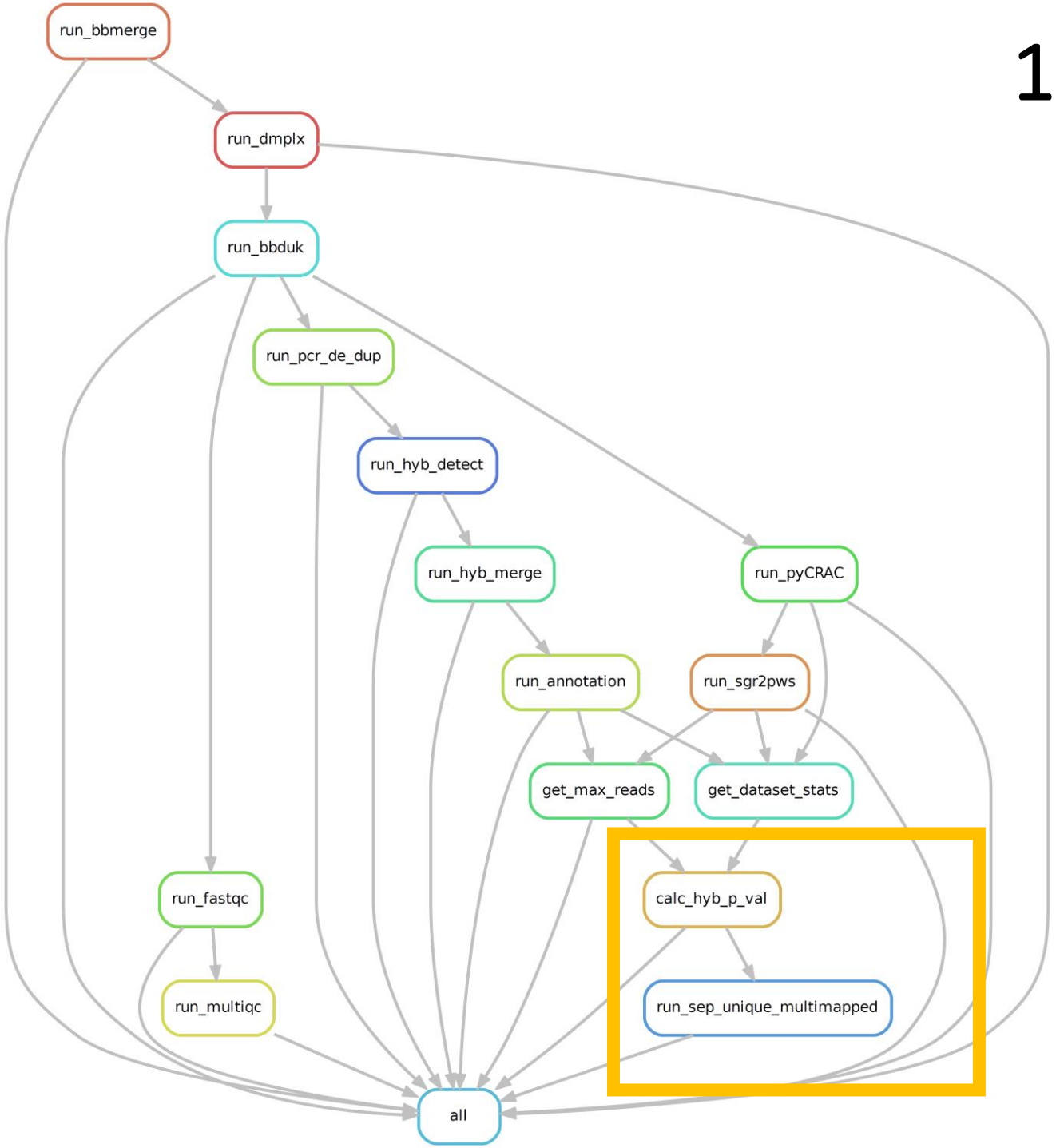
- Require one GTF file containing all annotations for all RNA class
- RNA_TYPE_PRIORTIES parameter points to the file "rna_type_priority.tab“. This file list in order of priority of annotation to help resolve overlapping features (see table on the right)
- Directory location of the GTF file is perthe GTF_FILE_DIR parameter: /srv/scratch/treelab/genome/saus/JKD6009/Hyb_pyCRAC

rank	rna_type
1	CDS
2	5UTR
3	3UTR
4	rRNA
5	tRNA
6	CRISPR
7	pseudogene
8	riboswitch
9	Rnase_P_RNA
10	RNA_termometer
11	SRP_RNA
12	tmRNA
13	operon

Updated

8-11. Get Max Reads Depth

- Use the pyCRAC pipeline to get the maximum read depth at each of the two genomic location of the interaction RNAs (run_pyCRAC, run_sgr2pws, get_max_reads)
- Also creates a file called “datasets.txt” that gives us the total number of reads for each barcode (get_dataset_stats)



12-13. Calculate p-value

- Uses a binomial distribution to calculate the p-value for each RNA-RNA interaction
- g_x, g_y are the maximum read depths of the two RNA
- p_x_y=2*g_x*g_y
- pdf_x_y = p_x_y/total_reads
- Hyb_count = Number of hybrid reads
- dbinom(x=hyb_count, size= total_reads, prob = pdf_x_y)

- Separate hybrids that are uniquely mapped to the genome with those that are mapped to multiple locations in the genome

Merge p-values of RNA-RNA interaction from biological replicate experiments

- merge_p_values.Rmd
- Uses Fisher's method to combine the p-value (based on a Chi-square distribution)
- Also perform Benjamini-Hochberg FDR correction
- Run independently outside of Snakemake pipeline
- Merge results from multiple Snakemake runs for different biological replicate experiments

- Adjust within R script the following parameters:
- list_of_files <- list(

file.path(hyb_p_val_dir, "DM_MiSeq", "all_NNNCACTAGC_L5Bc_rnc-HTF_1.unique.hyb.p_values.tab"), file.path(x) etc….)

- names(list_of_files) <- c("DM_MiSeq_Bc", "DM_NextSeq_Ba" etc…)

How to debug

- Use the --dry-run flag:
  - Simulate all the steps without running them, this checks all the dependencies
  - snakemake --dry-run -j 12 --snakefile jw_bc1.smk
- Use the --printshellcmds flag:
  - Print all the STDOUT from running each of the rules, useful for debugging
  - Combine with PBS –j oe, -m ae flags to save the output in log file (e.g. job_name.o12321)
  - Snakemake --printshellcmds -j 12 --snakefile jw_bc1.smk
- Job killed in the middle of action, and is locked out from a restart
  - snakemake -j 12 --unlock --printshellcmds --snakefile jw_bc1.smk
- Consider printing out the Rule Graph or the Directed Acyclic Graph

A quick dry-run demo (successful run)

snakemake --dry-run --printshellcmds -j 12 --snakefile jw_bc1_test_dry_run.smk

Dry-run completed successfully without fail

```
/treelab/igy/Data/CLASH_JKD6009/JW_BC1/Test_Dry_Run/Hyb_Annot/all_NNNTAAGC_L5Aa_BC1_control_1.multimapped.hyb.p_values.tab, /srv/scratch/treelab/ig
y/Data/CLASH_JKD6009/JW_BC1/Test_Dry_Run/Hyb_Annot/all_NNNATTAGC_L5Ab_BC1_rnc-HTF_1.multimapped.hyb.p_values.tab, /srv/scratch/treelab/igy/Data/CLA
SH_JKD6009/JW_BC1/Test_Dry_Run/Hyb_Annot/all_NNNGCGCAGC_L5Ac_BC1_rnc-HTF-2.multimapped.hyb.p_values.tab, /srv/scratch/treelab/igy/Data/CLASH_JKD600
9/JW_BC1/Test_Dry_Run/Hyb_Annot/all_NNNCACTAGC_L5Bc_BC1_rnc-HTF_3.multimapped.hyb.p_values.tab
    jobid: 0

Job counts:
        count   jobs
        1       all
        4       calc_hyb_p_val
        1       get_dataset_stats
        4       get_max_reads
        4       run_annotation
        4       run_bbduk
        1       run_bbmerge
        1       run_dmplx
        4       run_fastqc
        4       run_hyb_detect
        4       run_hyb_merge
        1       run_multiqc
        4       run_pcr_de_dup
        4       run_pyCRAC
        4       run_sep_unique_multimapped
        4       run_sgr2pws
        49
This was a dry-run (flag -n). The order of jobs does not reflect the order of execution.
[z3371724@katana1 Script Per Dataset]$
```

A quick dry-run demo (problem with code)

- I removed the following line from the run_bbmerge, output declarations:
  - all_fa = os.path.join(OUTPUT_DIR, "Merged", "all.fq"),

The run failed with error messages produced.

```
[z3371724@katana1 Script_Per_Dataset]$ snakemake --dry-run --printshellcmds -j 12 --snakefile jw_bc1_test_dry_run.smk
Building DAG of jobs...
MissingInputException in line 59 of /srv/scratch/treelab/igy/Projects/StaphCLASH2020/Source/Demultiplex/Script_Per_Dataset/common_rules_dry_run.smk:
Missing input files for rule run_dmplx:
/srv/scratch/treelab/igy/Data/CLASH_JKD6009/JW_BC1/Test_Dry_Run/Merged/all.fq
```

GitHub Resources

- https://github.com/IgnatiusPang/Hyb-CRAC-R
- https://github.com/IgnatiusPang/TermPick
- https://github.com/IgnatiusPang/Snakemake_Demo

A Tip for Writing Snakemake Pipeline

“Look on every exit as being an entrance somewhere else.”
— Tom Stoppard, Rosencrantz and Guildenstern Are Dead

“Look on every output file as being an input file to something else”

Acknowledgements

Tree Lab

- Tim Stinear’s Lab
- Stuart McKellar
- Sander Granneman