NGS analysis for gene regulation and epigenomics

Jacques Serizay, Cyril Matthey-Doret

January 11, 2020

Day 1

Snakemake pipelines

Benefits of a workflow manager

- Simple language to handle input/output
- Reusable code separated from data
- Generalize to multiple samples
- Platform comptability: HPC, cloud, local

Basic snakemake example

Writing chromosome names to a file

```
rule chrom_names:
  input: 'data/in/genome.fa'
  output: 'data/out/chrom_names.txt'
  shell: "grep '>' {input} > {output}"

rule all:
  input: 'data/out/chrom_names.txt'
```

General concept

- Pipeline divided into rules
- Each rule can have input, output and parameters
- Rules can execute bash or python code
- Pipeline supports python syntax

Separating data and code

Paths and sample names can be read from config files

```
import pandas as pd
samples = pd.read_csv('samples.tsv', sep='\t')
configfile: 'config.yaml'

rule align:
  input: 'data/in/{sample}.fq.gz'
  output: 'data/out/{sample}.sam'
  params:
    index = config['index']
  shell: "bowtie2 -x {params.index} -U {input} -S {output}"

rule all:
  input: expand('data/out/{sample}.sam', sample=samples['name'])
```

Additional features

- A conda environment can be defined for the pipeline or individual rules using a YAML file. It is then loaded using `conda: my_env.yaml`. This enforces fixed software versions to make results more reproducible.
- Rules can be executed in a docker / singularity image pulled automatically from the web. This allows to run softwares which are difficult to install.
- Multiple rules automatically run on separate CPUs in parallel
- Pipelines can be split into multiple files for modularity
- Much more: temporary files, report generation, logging…

- Write a Snakemake pipeline from the scripts in the previous exercise
- Allow it to run multiple samples

Exercises