tutoriel

June 14, 2022

1 Introduction to Snakemake workflow

schedule: - workflow introduction - snakemake introduction - snakefile for a 2-steps workflow - a litle more on snakefile

1.0.1 pre-requisites

1- data: input data to run the workflow example are reduced RNASeq reads files. Download data from zenodo here and unzip (`gunzip FAIR_Bioinfo_data.tar.gz`) on the same repository of the notebook

2- snakemake: for the run of snakemake, use a docker or a conda environment:

docker container: use the docker jupyterlab-notebook added with snakemake, fastqc and multiqc you may download it here:

note: if you want to run the command outside Jupiter, create a conda environment based on the `envfair.yml` file:

```
name: envfair # conda environment name
channels:
  - bioconda
dependencies:
  - snakemake-minimal>=6.5 # workflow manager
  - fastqc=0.11.9 # quality check of fastq data (java)
  - multiqc=1.12 # reports aggregation (R package)
```

1.1 Snakemake access

Laptop with docker:

```
save_jupylab_smk.tar # get the docker image archive
docker load < save_jupylab_smk.tar # create the docker image
docker run --rm -v ${PWD}:/home/jovyan -w /home/jovyan --user "$(id -u):$(id -g)" -p 8888:8888
snakemake ... # run
```

Laptop with conda:

```
conda create env -f envfair.yml
conda activate envfair
snakemake ... # run
```

IFB core cluster (*version 7.8.2 of the docker container is not yet available*):

```
module load slrum-drmaa/1.0.8 snakemake/7.7.0 fastqc/0.11.9 multiqc/1.12
snakemake ... # run
```

check run: replace … by `--version`

[1]:
```
%%sh
snakemake --version
```

```
7.8.2
```

1.2 Workflow definition

a pool of commands, progressively linked by the treatments, from the input data towards the results:

arrow: output of tool n−1 = input for tool n

In case of data paralelization, several data flows can be processed in parallel:

With a multi-cores PC or a computational cluster (ex. 2000 cores), one (or more) core can be attributed to one workflow.

1.3 Workflow management systems

Many workflow management systems, many forms: - command line: shell (but doesn't handle parallelization alone, need to script it, not easy) - rule: , , , … - graphic interface: , Taverna, Keppler, …

pros: - important for reproducibility (keep track of when each file was generated, and by which operation), - manage parallelization

cons: - learning effort

We choose : - works on files (rather than streams, reading/writing from databases or passing variables in memory) - is based on Python (but know how to code in Python is not required to work with Snakemake) - has features for defining the environment with which each task is carried out (running a large number of small third-party tools is current in bioinformatics) - is easily to be scaled from desktop to server, cluster, grid or cloud environments (ie. develop on laptop using a small subset of data, run the real analysis on a cluster)

1.4 The Snakemake rule

Snakemake: mix of the programming language Python (snake) and the Make, a rule-based automation tool

Good practice: one step, one rule

A rule is defined by it name and may contain: - `input:` list one or more file names - `output:` list one or more file names - command (`run:` for python ; `shell:` for shell, R, etc) - optional directives: `params:`, `message:`, `log:`, `threads:`, …

Remark: with 1 command line, use a `shell:` directive ; with many command lines, use a `run:` directive with the python `shell("...")` function

1.5 The data flow linkage and rules order

A snakemake workflow links rules thank to the filenames of the rule input and output directives.

Snakemake rules order: - until v.6.15.0: the first rule (all, target, ...) need to specify the result files and the next rules describe how to achieve them. - from v.6.15.0: all rules can be the default target rule

Snakemake automatically makes sure that everything is up to date, otherwise it launch the jobs that need to be: - create a DAG (directed acyclic graph) olinking rules thanks to their input and output files - starts with the last output result files of the DAG - since output files do not exist or have to be re-created, snakemake "goes back" through the workflow - output files have to be re-created when the input file timestamp is newer than the output file timestamp - and from this point, Snakemake goes on through the workflow and applies rules

1.6 Generalization with wilcards

Snakemake use *wildcards* allow to replace parts of filename: - reduce hardcoding: more flexible input and output directives, work on new data without modification - are automatically resolved (ie. replaced by regular expression ".+" in filenames) - are writing into {} - are specific to a rule

A same file can be accessed by different matchings: Ex. with the file `101/file.A.txt` : rule one : `output : "{set}1/file.{grp}.txt" # => set=10, grp=A` rule two : `output : "{set}/file.A.{ext}" # = > set=101, ext=txt` (more on wildcards in the snakemake documentation)

1.6.1 With and without wildcards example

without wildcards, `uniprot_wow.smk`:

```
rule get_prot:
  output: "P10415.fasta", "P01308.fasta"
  run :
    shell("wget https://www.uniprot.org/uniprot/P10415.fasta")
    shell("wget https://www.uniprot.org/uniprot/P01308.fasta")
```

with wildcards, `uniprot_wiw.smk`:

```
rule all:
  input: "P10415.fasta", "P01308.fasta"

rule get_prot:
  output: "{prot}.fasta"
  shell: "wget https://www.uniprot.org/uniprot/{wildcards.prot}.fasta"
```

1.7 Input (output) specifications

enumerated:

```
rule all:
  input: "P10415.fasta","P01308.fasta"
```

python list & wildcards:

```
DATASETS=["P10415","P01308"]
rule all:
  input: ["{dataset}.fasta".format(dataset=dataset) for dataset in DATASETS]
```

expand() & wildcards:

```
DATASETS=["P10415","P01308"]
rule all:
  input: expand("{dataset}.fasta",dataset=DATASETS)
```

The Snakefile example

The final objective is to create a snakefile to manage a small workflow with 2 steps: i) fastqc ii) multiqc

These two tools belonging to the bioinformatics domain allow to check the quality of high throughput sequence data. They are accessible via a Conda environment, `envfair.yml` or already included in the docker image `test/jupylab_smk:1.0` (see prerequisites, on top).

Sequence data stand in the `${PWD}/Data` repository (see prerequisites, on top).

note: - you will execute several cycles: executing snakefile, observing the result and improving the code. Each code version will be noted `ex1_oX.smk` with *ex* for example, *o* for objective, and *X* a progressive digit. - if you have already run this notebook, run:

[2]:
```
%%sh
rm -Rf FastQC multiqc_data multiqc_report.html
```

1.8 Objective 1: The rule concept

Objective 1: Create a snakemake file named `ex1_o1.smk` including the first step of the RNAseq workflow (the reads quality checking thank to the fastqc tool) on one of the RNAseq files

Hint: - input file: `SRR3099585_chr18.fastq.gz` in the ${PWD}/Data directory - fastqc access: access: through docker or conda environment (see prerequisites on top) - fastqc command: `fastqc --outdir FastQCResultDirectory inputFileName` - the 2 result files (`*_fastqc.zip` & `*_fastqc.html`) will be located in your `outdir` and named based on the prefix of input file (eg. `SRR3099585_chr18_fastqc.zip`)

1.9 Objective 1: solution

Code for `ex1_o1.smk`:

```
rule fastqc:
  output:
    "FastQC/SRR3099585_chr18_fastqc.zip",
    "FastQC/SRR3099585_chr18_fastqc.html"
  input:
    "Data/SRR3099585_chr18.fastq.gz"
  shell: "fastqc --outdir FastQC/ {input}"
```

[3]:
```
%%sh
snakemake --snakefile ex1_o1.smk --cores 1
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job       count    min threads    max threads
------  -------  -------------  -------------
fastqc        1              1              1
total         1              1              1

Select jobs to execute…

[Sun Jun 12 21:14:47 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html
    jobid: 0
    reason: Missing output files: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html
    resources: tmpdir=/tmp

Started analysis of SRR3099585_chr18.fastq.gz
Approx 5% complete for SRR3099585_chr18.fastq.gz
Approx 10% complete for SRR3099585_chr18.fastq.gz
Approx 15% complete for SRR3099585_chr18.fastq.gz
Approx 20% complete for SRR3099585_chr18.fastq.gz
Approx 25% complete for SRR3099585_chr18.fastq.gz
Approx 30% complete for SRR3099585_chr18.fastq.gz
Approx 35% complete for SRR3099585_chr18.fastq.gz
Approx 40% complete for SRR3099585_chr18.fastq.gz
Approx 45% complete for SRR3099585_chr18.fastq.gz
Approx 50% complete for SRR3099585_chr18.fastq.gz
Approx 55% complete for SRR3099585_chr18.fastq.gz
Approx 60% complete for SRR3099585_chr18.fastq.gz
Approx 65% complete for SRR3099585_chr18.fastq.gz
Approx 70% complete for SRR3099585_chr18.fastq.gz
Approx 75% complete for SRR3099585_chr18.fastq.gz
Approx 80% complete for SRR3099585_chr18.fastq.gz
Approx 85% complete for SRR3099585_chr18.fastq.gz
Approx 90% complete for SRR3099585_chr18.fastq.gz
Approx 95% complete for SRR3099585_chr18.fastq.gz

Analysis complete for SRR3099585_chr18.fastq.gz
```

```
[Sun Jun 12 21:14:58 2022]
Finished job 0.
1 of 1 steps (100%) done
Complete log: .snakemake/log/2022-06-12T211445.545170.snakemake.log
```

Observe result: Look at the newly created `FastQC` directory: Snakemake create alone the needed directories.

1.10 Objective 2: One rule, 2 input files

Objective 2: Add a second input RNAseq file to the rule.

Hint: - second input file: `Data/SRR3099586_chr18.fastq.gz` - don't forget to add the cognate output files

1.11 Objective 2: Solution

Code for `ex1_o2.smk`:

```
rule fastqc:
  output:
    "FastQC/SRR3099585_chr18_fastqc.zip",
    "FastQC/SRR3099585_chr18_fastqc.html",
    "FastQC/SRR3099586_chr18_fastqc.zip",
    "FastQC/SRR3099586_chr18_fastqc.html"
  input:
    "Data/SRR3099585_chr18.fastq.gz",
    "Data/SRR3099586_chr18.fastq.gz"
  shell: "fastqc --outdir FastQC/ {input}"
```

[4]:
```
%%sh
snakemake -s ex1_o2.smk -c 1 -p # -s & -c : short forms of the -- snakefile &␣
 ↪--cores options -p prints the command
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job       count    min threads    max threads
------  -------  -------------  -------------
fastqc        1              1              1
total         1              1              1

Select jobs to execute…

[Sun Jun 12 21:15:39 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz, Data/SRR3099586_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
```

```
FastQC/SRR3099585_chr18_fastqc.html, FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html
    jobid: 0
    reason: Missing output files: FastQC/SRR3099586_chr18_fastqc.html,
FastQC/SRR3099586_chr18_fastqc.zip
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099585_chr18.fastq.gz
Data/SRR3099586_chr18.fastq.gz
Started analysis of SRR3099585_chr18.fastq.gz
Approx 5% complete for SRR3099585_chr18.fastq.gz
Approx 10% complete for SRR3099585_chr18.fastq.gz
Approx 15% complete for SRR3099585_chr18.fastq.gz
Approx 20% complete for SRR3099585_chr18.fastq.gz
Approx 25% complete for SRR3099585_chr18.fastq.gz
Approx 30% complete for SRR3099585_chr18.fastq.gz
Approx 35% complete for SRR3099585_chr18.fastq.gz
Approx 40% complete for SRR3099585_chr18.fastq.gz
Approx 45% complete for SRR3099585_chr18.fastq.gz
Approx 50% complete for SRR3099585_chr18.fastq.gz
Approx 55% complete for SRR3099585_chr18.fastq.gz
Approx 60% complete for SRR3099585_chr18.fastq.gz
Approx 65% complete for SRR3099585_chr18.fastq.gz
Approx 70% complete for SRR3099585_chr18.fastq.gz
Approx 75% complete for SRR3099585_chr18.fastq.gz
Approx 80% complete for SRR3099585_chr18.fastq.gz
Approx 85% complete for SRR3099585_chr18.fastq.gz
Approx 90% complete for SRR3099585_chr18.fastq.gz
Approx 95% complete for SRR3099585_chr18.fastq.gz

Analysis complete for SRR3099585_chr18.fastq.gz

Started analysis of SRR3099586_chr18.fastq.gz
Approx 5% complete for SRR3099586_chr18.fastq.gz
Approx 10% complete for SRR3099586_chr18.fastq.gz
Approx 15% complete for SRR3099586_chr18.fastq.gz
Approx 20% complete for SRR3099586_chr18.fastq.gz
Approx 25% complete for SRR3099586_chr18.fastq.gz
Approx 30% complete for SRR3099586_chr18.fastq.gz
Approx 35% complete for SRR3099586_chr18.fastq.gz
Approx 40% complete for SRR3099586_chr18.fastq.gz
Approx 45% complete for SRR3099586_chr18.fastq.gz
Approx 50% complete for SRR3099586_chr18.fastq.gz
Approx 55% complete for SRR3099586_chr18.fastq.gz
Approx 60% complete for SRR3099586_chr18.fastq.gz
Approx 65% complete for SRR3099586_chr18.fastq.gz
Approx 70% complete for SRR3099586_chr18.fastq.gz
Approx 75% complete for SRR3099586_chr18.fastq.gz
Approx 80% complete for SRR3099586_chr18.fastq.gz
```

```
Approx 85% complete for SRR3099586_chr18.fastq.gz
Approx 90% complete for SRR3099586_chr18.fastq.gz
Approx 95% complete for SRR3099586_chr18.fastq.gz

Analysis complete for SRR3099586_chr18.fastq.gz

[Sun Jun 12 21:15:49 2022]
Finished job 0.
1 of 1 steps (100%) done
Complete log: .snakemake/log/2022-06-12T211538.981215.snakemake.log
```

Observe results: Snakemake rerun computation for the 1rst data file: its a new behavior from the v.7.8.0. Before v.7.8.0, rerunning jobs relied purely on file modification times and the output files of the 1rst data file were not rerun. After v.7.8.0, all provenance information is considered to define the jobs to rerun: parameter changes, code changes, software environment changes, and changes in the set of input files of a job (use the command line option `--rerun-triggers mtime` to use only modification time to determine whether a job shall be executed).

Rerun without any changes:

[5]:
```
%%sh
snakemake -s ex1_o2.smk -c 1 -p
```

```
Building DAG of jobs…
Nothing to be done (all requested files are present and up to date).
Complete log: .snakemake/log/2022-06-12T211609.720352.snakemake.log
```

Snakemake reply "Nothing to be done".

Some solutions to rerun: - delete the `FastQC` directory and rerun the snakemake command - use the Snakemake `--forcerules` (`-R`) option

[ ]:
```
%%sh
rm -Rf FastQC
snakemake -s ex1_o2.smk -c 1 -p
```

[ ]:
```
%%sh
snakemake -s ex1_o2.smk -c 1 -p -R fastqc
```

1.12 Objective 3: Manage all the RNAseq files

Objective 3: Add all the RNAseq files.

Boring with writing all input and output file names? Use the `expand()` function to manage all the input RNAseq files at once.

Hint: - create a Python list at the begining of the snakefile and containing all the basename of the input files (don't include the `.fastq.gz` suffix). - Python list format: `list_name = ["item1", "item2", ..., "itemN"]` - replace the filename lists of the input and output directives by the `expand()` function

1.13 Objective 3: solution

Code for `ex1_o3.smk`:

```
SAMPLES=["SRR3099585_chr18","SRR3099586_chr18","SRR3099587_chr18"]

rule fastqc:
  output:
    expand("FastQC/{sample}_fastqc.zip", sample=SAMPLES),
    expand("FastQC/{sample}_fastqc.html", sample=SAMPLES)
  input:
    expand("data/{sample}.fastq.gz", sample=SAMPLES)
  shell: "fastqc --outdir FastQC {input}"
```

[6]:
```
%%sh
snakemake -c1 -s ex1_o3.smk --rerun-triggers mtime -p
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job       count    min threads    max threads
------  -------  -------------  -------------
fastqc        1              1              1
total         1              1              1

Select jobs to execute…

[Sun Jun 12 21:17:03 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz, Data/SRR3099586_chr18.fastq.gz,
Data/SRR3099587_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html, FastQC/SRR3099586_chr18_fastqc.html,
FastQC/SRR3099587_chr18_fastqc.html
    jobid: 0
    reason: Missing output files: FastQC/SRR3099587_chr18_fastqc.html,
FastQC/SRR3099587_chr18_fastqc.zip
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099585_chr18.fastq.gz
Data/SRR3099586_chr18.fastq.gz Data/SRR3099587_chr18.fastq.gz
Started analysis of SRR3099585_chr18.fastq.gz
Approx 5% complete for SRR3099585_chr18.fastq.gz
Approx 10% complete for SRR3099585_chr18.fastq.gz
Approx 15% complete for SRR3099585_chr18.fastq.gz
Approx 20% complete for SRR3099585_chr18.fastq.gz
```

```
Approx 25% complete for SRR3099585_chr18.fastq.gz
Approx 30% complete for SRR3099585_chr18.fastq.gz
Approx 35% complete for SRR3099585_chr18.fastq.gz
Approx 40% complete for SRR3099585_chr18.fastq.gz
Approx 45% complete for SRR3099585_chr18.fastq.gz
Approx 50% complete for SRR3099585_chr18.fastq.gz
Approx 55% complete for SRR3099585_chr18.fastq.gz
Approx 60% complete for SRR3099585_chr18.fastq.gz
Approx 65% complete for SRR3099585_chr18.fastq.gz
Approx 70% complete for SRR3099585_chr18.fastq.gz
Approx 75% complete for SRR3099585_chr18.fastq.gz
Approx 80% complete for SRR3099585_chr18.fastq.gz
Approx 85% complete for SRR3099585_chr18.fastq.gz
Approx 90% complete for SRR3099585_chr18.fastq.gz
Approx 95% complete for SRR3099585_chr18.fastq.gz

Analysis complete for SRR3099585_chr18.fastq.gz

Started analysis of SRR3099586_chr18.fastq.gz
Approx 5% complete for SRR3099586_chr18.fastq.gz
Approx 10% complete for SRR3099586_chr18.fastq.gz
Approx 15% complete for SRR3099586_chr18.fastq.gz
Approx 20% complete for SRR3099586_chr18.fastq.gz
Approx 25% complete for SRR3099586_chr18.fastq.gz
Approx 30% complete for SRR3099586_chr18.fastq.gz
Approx 35% complete for SRR3099586_chr18.fastq.gz
Approx 40% complete for SRR3099586_chr18.fastq.gz
Approx 45% complete for SRR3099586_chr18.fastq.gz
Approx 50% complete for SRR3099586_chr18.fastq.gz
Approx 55% complete for SRR3099586_chr18.fastq.gz
Approx 60% complete for SRR3099586_chr18.fastq.gz
Approx 65% complete for SRR3099586_chr18.fastq.gz
Approx 70% complete for SRR3099586_chr18.fastq.gz
Approx 75% complete for SRR3099586_chr18.fastq.gz
Approx 80% complete for SRR3099586_chr18.fastq.gz
Approx 85% complete for SRR3099586_chr18.fastq.gz
Approx 90% complete for SRR3099586_chr18.fastq.gz
Approx 95% complete for SRR3099586_chr18.fastq.gz

Analysis complete for SRR3099586_chr18.fastq.gz

Started analysis of SRR3099587_chr18.fastq.gz
Approx 5% complete for SRR3099587_chr18.fastq.gz
Approx 10% complete for SRR3099587_chr18.fastq.gz
Approx 15% complete for SRR3099587_chr18.fastq.gz
Approx 20% complete for SRR3099587_chr18.fastq.gz
Approx 25% complete for SRR3099587_chr18.fastq.gz
Approx 30% complete for SRR3099587_chr18.fastq.gz
Approx 35% complete for SRR3099587_chr18.fastq.gz
Approx 40% complete for SRR3099587_chr18.fastq.gz
```

```
Approx 45% complete for SRR3099587_chr18.fastq.gz
Approx 50% complete for SRR3099587_chr18.fastq.gz
Approx 55% complete for SRR3099587_chr18.fastq.gz
Approx 60% complete for SRR3099587_chr18.fastq.gz
Approx 65% complete for SRR3099587_chr18.fastq.gz
Approx 70% complete for SRR3099587_chr18.fastq.gz
Approx 75% complete for SRR3099587_chr18.fastq.gz
Approx 80% complete for SRR3099587_chr18.fastq.gz
Approx 85% complete for SRR3099587_chr18.fastq.gz
Approx 90% complete for SRR3099587_chr18.fastq.gz
Approx 95% complete for SRR3099587_chr18.fastq.gz

Analysis complete for SRR3099587_chr18.fastq.gz

[Sun Jun 12 21:17:19 2022]
Finished job 0.
1 of 1 steps (100%) done
Complete log: .snakemake/log/2022-06-12T211703.619495.snakemake.log
```

Observe the result: The `--rerun-triggers mtime` option seems not to apply. Why?

Look at the printed command line: one fastqc managing all imput files ; see also the *Job stats* table (begining of the snakemake output):

However, there are many input files but snakemake launched only one fastqc job.

It is because the fastqc rule is defined with a list of files (the return format of the expand function) and not for one unique file and also because the fastqc tool accepts both a single file as well as a list of files.

1.14 Objective 4: Running n individual jobs

Objective 4: add a rule containing the output files list and manage the fastqc rule to work with a single file

Hint: - define a new rule, nammed *target* with only an input directive based on the input of the fastqc rule - in the fastqc rule, replace the expand() function with a simple wildcard for the filename - suppress the `--rerun-triggers mtime` to see the effect

1.15 Objective 4: Solution

Code for ex1_o4.smk **change number**

```
SAMPLES = ["SRR3099585_chr18","SRR3099586_chr18","SRR3099587_chr18"]

rule all:
  input:
    expand("FastQC/{sample}_fastqc.zip", sample = SAMPLES),
    expand("FastQC/{sample}_fastqc.html", sample=SAMPLES)

rule fastqc:
  output:
```

```
    "FastQC/{sample}_fastqc.zip",
    "FastQC/{sample}_fastqc.html"
  input:
    "Data/{sample}.fastq.gz"
  params: "FastQC"
  shell: "fastqc --outdir {params} {input}"
```

[7]:
```
%%sh
snakemake -c1 -s ex1_o3b.smk -p
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job       count    min threads    max threads
------  -------  -------------  -------------
all           1              1              1
fastqc        3              1              1
total         4              1              1

Select jobs to execute…

[Sun Jun 12 21:18:06 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html
    jobid: 1
    reason: Set of input files has changed since last execution; Code has
changed since last execution; Params have changed since last execution
    wildcards: sample=SRR3099585_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC Data/SRR3099585_chr18.fastq.gz
Started analysis of SRR3099585_chr18.fastq.gz
Approx 5% complete for SRR3099585_chr18.fastq.gz
Approx 10% complete for SRR3099585_chr18.fastq.gz
Approx 15% complete for SRR3099585_chr18.fastq.gz
Approx 20% complete for SRR3099585_chr18.fastq.gz
Approx 25% complete for SRR3099585_chr18.fastq.gz
Approx 30% complete for SRR3099585_chr18.fastq.gz
Approx 35% complete for SRR3099585_chr18.fastq.gz
Approx 40% complete for SRR3099585_chr18.fastq.gz
Approx 45% complete for SRR3099585_chr18.fastq.gz
Approx 50% complete for SRR3099585_chr18.fastq.gz
Approx 55% complete for SRR3099585_chr18.fastq.gz
Approx 60% complete for SRR3099585_chr18.fastq.gz
```

```
Approx 65% complete for SRR3099585_chr18.fastq.gz
Approx 70% complete for SRR3099585_chr18.fastq.gz
Approx 75% complete for SRR3099585_chr18.fastq.gz
Approx 80% complete for SRR3099585_chr18.fastq.gz
Approx 85% complete for SRR3099585_chr18.fastq.gz
Approx 90% complete for SRR3099585_chr18.fastq.gz
Approx 95% complete for SRR3099585_chr18.fastq.gz

Analysis complete for SRR3099585_chr18.fastq.gz

[Sun Jun 12 21:18:14 2022]
Finished job 1.
1 of 4 steps (25%) done
Select jobs to execute…

[Sun Jun 12 21:18:14 2022]
rule fastqc:
    input: Data/SRR3099587_chr18.fastq.gz
    output: FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099587_chr18_fastqc.html
    jobid: 3
    reason: Set of input files has changed since last execution; Code has
changed since last execution; Params have changed since last execution
    wildcards: sample=SRR3099587_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC Data/SRR3099587_chr18.fastq.gz
Started analysis of SRR3099587_chr18.fastq.gz
Approx 5% complete for SRR3099587_chr18.fastq.gz
Approx 10% complete for SRR3099587_chr18.fastq.gz
Approx 15% complete for SRR3099587_chr18.fastq.gz
Approx 20% complete for SRR3099587_chr18.fastq.gz
Approx 25% complete for SRR3099587_chr18.fastq.gz
Approx 30% complete for SRR3099587_chr18.fastq.gz
Approx 35% complete for SRR3099587_chr18.fastq.gz
Approx 40% complete for SRR3099587_chr18.fastq.gz
Approx 45% complete for SRR3099587_chr18.fastq.gz
Approx 50% complete for SRR3099587_chr18.fastq.gz
Approx 55% complete for SRR3099587_chr18.fastq.gz
Approx 60% complete for SRR3099587_chr18.fastq.gz
Approx 65% complete for SRR3099587_chr18.fastq.gz
Approx 70% complete for SRR3099587_chr18.fastq.gz
Approx 75% complete for SRR3099587_chr18.fastq.gz
Approx 80% complete for SRR3099587_chr18.fastq.gz
Approx 85% complete for SRR3099587_chr18.fastq.gz
Approx 90% complete for SRR3099587_chr18.fastq.gz
Approx 95% complete for SRR3099587_chr18.fastq.gz

Analysis complete for SRR3099587_chr18.fastq.gz
```

```
[Sun Jun 12 21:18:20 2022]
Finished job 3.
2 of 4 steps (50%) done
Select jobs to execute…

[Sun Jun 12 21:18:20 2022]
rule fastqc:
    input: Data/SRR3099586_chr18.fastq.gz
    output: FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html
    jobid: 2
    reason: Set of input files has changed since last execution; Code has
changed since last execution; Params have changed since last execution
    wildcards: sample=SRR3099586_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC Data/SRR3099586_chr18.fastq.gz
Started analysis of SRR3099586_chr18.fastq.gz
Approx 5% complete for SRR3099586_chr18.fastq.gz
Approx 10% complete for SRR3099586_chr18.fastq.gz
Approx 15% complete for SRR3099586_chr18.fastq.gz
Approx 20% complete for SRR3099586_chr18.fastq.gz
Approx 25% complete for SRR3099586_chr18.fastq.gz
Approx 30% complete for SRR3099586_chr18.fastq.gz
Approx 35% complete for SRR3099586_chr18.fastq.gz
Approx 40% complete for SRR3099586_chr18.fastq.gz
Approx 45% complete for SRR3099586_chr18.fastq.gz
Approx 50% complete for SRR3099586_chr18.fastq.gz
Approx 55% complete for SRR3099586_chr18.fastq.gz
Approx 60% complete for SRR3099586_chr18.fastq.gz
Approx 65% complete for SRR3099586_chr18.fastq.gz
Approx 70% complete for SRR3099586_chr18.fastq.gz
Approx 75% complete for SRR3099586_chr18.fastq.gz
Approx 80% complete for SRR3099586_chr18.fastq.gz
Approx 85% complete for SRR3099586_chr18.fastq.gz
Approx 90% complete for SRR3099586_chr18.fastq.gz
Approx 95% complete for SRR3099586_chr18.fastq.gz

Analysis complete for SRR3099586_chr18.fastq.gz

[Sun Jun 12 21:18:27 2022]
Finished job 2.
3 of 4 steps (75%) done
Select jobs to execute…

[Sun Jun 12 21:18:27 2022]
localrule all:
    input: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip,
```

```
FastQC/SRR3099585_chr18_fastqc.html, FastQC/SRR3099586_chr18_fastqc.html,
FastQC/SRR3099587_chr18_fastqc.html
    jobid: 0
    reason: Input files updated by another job:
FastQC/SRR3099587_chr18_fastqc.html, FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html, FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html, FastQC/SRR3099586_chr18_fastqc.zip
    resources: tmpdir=/tmp

[Sun Jun 12 21:18:27 2022]
Finished job 0.
4 of 4 steps (100%) done
Complete log: .snakemake/log/2022-06-12T211806.124892.snakemake.log
```

Observe the result: the order of execution is not always the "human" order

1.16 Objective 5: Add a second step

Objective 5: with a second tool, it is a "real" analysis workflow! The second tool multiqc will aggregate all the fastqc results.

Hint: - multiqc inputs: the fastqc zip files - multiqc command: `multiqc *_fastqc.zip` - 2 outputs of multiqc: a file `multiqc_report.html` & a repository `multiqc_data` (to manage with `directory("multiqc_data")`)

1.17 Objective 5: solution

Code for `ex1_o6.smk`:

```
SAMPLES = ["SRR3099585_chr18","SRR3099586_chr18","SRR3099587_chr18"]

rule all:
  input:
    expand("FastQC/{sample}_fastqc.html", sample=SAMPLES),
    "multiqc_report.html"

rule multiqc:
  output:
    "multiqc_report.html",
    directory("multiqc_data")
  input:
    expand("FastQC/{sample}_fastqc.zip", sample = SAMPLES)
  shell: "multiqc {input}"

rule fastqc:
  output:
    "FastQC/{sample}_fastqc.zip",
    "FastQC/{sample}_fastqc.html"
  input:
    "Data/{sample}.fastq.gz"
```

```
  params: "FastQC"
  shell: "fastqc --outdir {params} {input}"
```

[8]:
```
%%sh
# rm -Rf multiqc_data
snakemake -c1 -s ex1_o6.smk -p --rerun-triggers mtime
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job        count    min threads    max threads
-------  -------  -------------  -------------
all            1              1              1
multiqc        1              1              1
total          2              1              1

Select jobs to execute…

[Sun Jun 12 21:18:54 2022]
rule multiqc:
    input: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip
    output: multiqc_report.html, multiqc_data
    jobid: 4
    reason: Missing output files: multiqc_report.html
    resources: tmpdir=/tmp

multiqc FastQC/SRR3099585_chr18_fastqc.zip FastQC/SRR3099586_chr18_fastqc.zip
FastQC/SRR3099587_chr18_fastqc.zip

  /// MultiQC   | v1.12

|           multiqc | Search path :
/home/jovyan/tutoriel_smk/FastQC/SRR3099585_chr18_fastqc.zip
|           multiqc | Search path :
/home/jovyan/tutoriel_smk/FastQC/SRR3099586_chr18_fastqc.zip
|           multiqc | Search path :
/home/jovyan/tutoriel_smk/FastQC/SRR3099587_chr18_fastqc.zip

|         searching |                       100% 3/3

|            fastqc | Found 3 reports
|           multiqc | Compressing plot data
|           multiqc | Report      : multiqc_report.html
|           multiqc | Data        : multiqc_data
|           multiqc | MultiQC complete
[Sun Jun 12 21:19:02 2022]
```

```
Finished job 4.
1 of 2 steps (50%) done
Select jobs to execute…

[Sun Jun 12 21:19:02 2022]
localrule all:
    input: FastQC/SRR3099585_chr18_fastqc.html,
FastQC/SRR3099586_chr18_fastqc.html, FastQC/SRR3099587_chr18_fastqc.html,
multiqc_report.html
    jobid: 0
    reason: Input files updated by another job: multiqc_report.html
    resources: tmpdir=/tmp

[Sun Jun 12 21:19:02 2022]
Finished job 0.
2 of 2 steps (100%) done
Complete log: .snakemake/log/2022-06-12T211854.446695.snakemake.log
```

1.18 Objective 6: Adding log file

Objective 6: In Unix systems, the output of a command is usually sent to 2 separate streams: the expected output to Standard Out (stdout, or ">"), and the error messages to Standard Error (stderr, or "2>"). To integrate stderr and stdout into the same log, use "&>" (use it with care because output files are often printed to stdout).

Hint: - redirect the stdout and stderr streams of the fastqc and multiqc rules by adding a "log:" directive with two variables, out and err to separately redirect each streams.

1.19 Objective 6: Solution

Code for ex1_o7.smk:

```
SAMPLES = ["SRR3099585_chr18","SRR3099586_chr18","SRR3099587_chr18"]

rule all:
  input:
    expand("FastQC/{sample}_fastqc.html", sample=SAMPLES),
    "multiqc_report.html"

rule multiqc:
  output:
    "multiqc_report.html"
  input:
    expand("FastQC/{sample}_fastqc.zip", sample = SAMPLES)
  log:
    std="Logs/multiqc.std",
    err="Logs/multiqc.err"
  shell: "multiqc {input} 1>{log.std} 2>{log.err}"
```

```
rule fastqc:
  output:
    "FastQC/{sample}_fastqc.zip",
    "FastQC/{sample}_fastqc.html"
  input:
    "Data/{sample}.fastq.gz"
  log:
    std="Logs/{sample}_fastqc.std",
    err="Logs/{sample}_fastqc.err"
  shell: "fastqc --outdir FastQC/ {input} 1>{log.std} 2>{log.err}"
```

[9]:
```
%%sh
# rm -Rf multiqc_data
snakemake -c1 -s ex1_o7.smk -p
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job        count    min threads    max threads
-------  -------  -------------  -------------
all            1              1              1
fastqc         3              1              1
multiqc        1              1              1
total          5              1              1

Select jobs to execute…

[Sun Jun 12 21:20:01 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html
    log: Logs/SRR3099585_chr18_fastqc.std, Logs/SRR3099585_chr18_fastqc.err
    jobid: 1
    reason: Code has changed since last execution; Params have changed since
last execution
    wildcards: sample=SRR3099585_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099585_chr18.fastq.gz
1>Logs/SRR3099585_chr18_fastqc.std 2>Logs/SRR3099585_chr18_fastqc.err
[Sun Jun 12 21:20:09 2022]
Finished job 1.
1 of 5 steps (20%) done
Select jobs to execute…
```

```
[Sun Jun 12 21:20:09 2022]
rule fastqc:
    input: Data/SRR3099587_chr18.fastq.gz
    output: FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099587_chr18_fastqc.html
    log: Logs/SRR3099587_chr18_fastqc.std, Logs/SRR3099587_chr18_fastqc.err
    jobid: 3
    reason: Code has changed since last execution; Params have changed since
last execution
    wildcards: sample=SRR3099587_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099587_chr18.fastq.gz
1>Logs/SRR3099587_chr18_fastqc.std 2>Logs/SRR3099587_chr18_fastqc.err
[Sun Jun 12 21:20:15 2022]
Finished job 3.
2 of 5 steps (40%) done
Select jobs to execute…

[Sun Jun 12 21:20:15 2022]
rule fastqc:
    input: Data/SRR3099586_chr18.fastq.gz
    output: FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html
    log: Logs/SRR3099586_chr18_fastqc.std, Logs/SRR3099586_chr18_fastqc.err
    jobid: 2
    reason: Code has changed since last execution; Params have changed since
last execution
    wildcards: sample=SRR3099586_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099586_chr18.fastq.gz
1>Logs/SRR3099586_chr18_fastqc.std 2>Logs/SRR3099586_chr18_fastqc.err
[Sun Jun 12 21:20:22 2022]
Finished job 2.
3 of 5 steps (60%) done
Select jobs to execute…

[Sun Jun 12 21:20:22 2022]
rule multiqc:
    input: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip
    output: multiqc_report.html, multiqc_data
    log: Logs/multiqc.std, Logs/multiqc.err
    jobid: 4
    reason: Input files updated by another job:
FastQC/SRR3099585_chr18_fastqc.zip, FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099587_chr18_fastqc.zip
```

```
    resources: tmpdir=/tmp

multiqc FastQC/SRR3099585_chr18_fastqc.zip FastQC/SRR3099586_chr18_fastqc.zip
FastQC/SRR3099587_chr18_fastqc.zip 1>Logs/multiqc.std 2>Logs/multiqc.err
[Sun Jun 12 21:20:25 2022]
Finished job 4.
4 of 5 steps (80%) done
Select jobs to execute…

[Sun Jun 12 21:20:25 2022]
localrule all:
    input: FastQC/SRR3099585_chr18_fastqc.html,
FastQC/SRR3099586_chr18_fastqc.html, FastQC/SRR3099587_chr18_fastqc.html,
multiqc_report.html
    jobid: 0
    reason: Input files updated by another job:
FastQC/SRR3099586_chr18_fastqc.html, multiqc_report.html,
FastQC/SRR3099587_chr18_fastqc.html, FastQC/SRR3099585_chr18_fastqc.html
    resources: tmpdir=/tmp

[Sun Jun 12 21:20:25 2022]
Finished job 0.
5 of 5 steps (100%) done
Complete log: .snakemake/log/2022-06-12T212001.252433.snakemake.log
```

1.20 Snakemake point

So far, we've seen: - the rule and the workflow concepts, the snakefile - how rules are linked thank to input/output files and the first rule, the target rule - how to generalize the inputs of a rule using wildcards on filenames (and the expand function) - how to redirect stdout and stderr streams (log)

From now, we will seen some snakemake options: - adding a configuration file - getting file names from the file system - to visualize the workflow diagram, use a dry-run option, etc - use a conda environment

Objective 7: Use a configuration file **Why use a configuration file?** To place all hard-coding values of the snakefile (paths to files, core numbers, parameter values, etc).

How to do it in Snakefile? Create a yml or json file and call the defined items with `config["myItem"]`. And run with the `--configfile myConfig.yml` Snakemake option (an other solution is to add the directive `configfile: myConfig.yml` at the beginning of the snakefile)

Objective 7: create `myConfig.yml` that specifies the the path of the data directory and replace the hard-coding values `"Data/"` in the snakefile.

1.21 Objective 7: Solution

Code for `myConfig.yml`:

```
dataDir:
```

```
  "Data/"
```

Copy `ex1_o7.smk` into `ex1_o7b.smk` and replace `Data/...` in inputs by a config call:

```
rule fastqc:
  input:config["dataDir"]+"{sample}.fastq.gz"
```

[10]:
```
%%sh
rm -Rf FastQC multiqc_data multiqc_report*
snakemake -c1 -s ex1_o7b.smk -p --configfile myConfig.yml
```

```
Building DAG of jobs…
Using shell: /usr/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job        count    min threads    max threads
-------  -------  -------------  -------------
all            1              1              1
fastqc         3              1              1
multiqc        1              1              1
total          5              1              1

Select jobs to execute…

[Sun Jun 12 21:20:50 2022]
rule fastqc:
    input: Data/SRR3099585_chr18.fastq.gz
    output: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.html
    log: Logs/SRR3099585_chr18_fastqc.std, Logs/SRR3099585_chr18_fastqc.err
    jobid: 1
    reason: Missing output files: FastQC/SRR3099585_chr18_fastqc.html,
FastQC/SRR3099585_chr18_fastqc.zip
    wildcards: sample=SRR3099585_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099585_chr18.fastq.gz
1>Logs/SRR3099585_chr18_fastqc.std 2>Logs/SRR3099585_chr18_fastqc.err
[Sun Jun 12 21:20:57 2022]
Finished job 1.
1 of 5 steps (20%) done
Select jobs to execute…

[Sun Jun 12 21:20:57 2022]
rule fastqc:
    input: Data/SRR3099587_chr18.fastq.gz
    output: FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099587_chr18_fastqc.html
```

```
    log: Logs/SRR3099587_chr18_fastqc.std, Logs/SRR3099587_chr18_fastqc.err
    jobid: 3
    reason: Missing output files: FastQC/SRR3099587_chr18_fastqc.zip,
FastQC/SRR3099587_chr18_fastqc.html
    wildcards: sample=SRR3099587_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099587_chr18.fastq.gz
1>Logs/SRR3099587_chr18_fastqc.std 2>Logs/SRR3099587_chr18_fastqc.err
[Sun Jun 12 21:21:04 2022]
Finished job 3.
2 of 5 steps (40%) done
Select jobs to execute…

[Sun Jun 12 21:21:04 2022]
rule fastqc:
    input: Data/SRR3099586_chr18.fastq.gz
    output: FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html
    log: Logs/SRR3099586_chr18_fastqc.std, Logs/SRR3099586_chr18_fastqc.err
    jobid: 2
    reason: Missing output files: FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.html
    wildcards: sample=SRR3099586_chr18
    resources: tmpdir=/tmp

fastqc --outdir FastQC/ Data/SRR3099586_chr18.fastq.gz
1>Logs/SRR3099586_chr18_fastqc.std 2>Logs/SRR3099586_chr18_fastqc.err
[Sun Jun 12 21:21:10 2022]
Finished job 2.
3 of 5 steps (60%) done
Select jobs to execute…

[Sun Jun 12 21:21:10 2022]
rule multiqc:
    input: FastQC/SRR3099585_chr18_fastqc.zip,
FastQC/SRR3099586_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip
    output: multiqc_report.html, multiqc_data
    log: Logs/multiqc.std, Logs/multiqc.err
    jobid: 4
    reason: Missing output files: multiqc_report.html; Input files updated by
another job: FastQC/SRR3099586_chr18_fastqc.zip,
FastQC/SRR3099585_chr18_fastqc.zip, FastQC/SRR3099587_chr18_fastqc.zip
    resources: tmpdir=/tmp

multiqc FastQC/SRR3099585_chr18_fastqc.zip FastQC/SRR3099586_chr18_fastqc.zip
FastQC/SRR3099587_chr18_fastqc.zip 1>Logs/multiqc.std 2>Logs/multiqc.err
[Sun Jun 12 21:21:13 2022]
```

```
Finished job 4.
4 of 5 steps (80%) done
Select jobs to execute…

[Sun Jun 12 21:21:13 2022]
localrule all:
    input: FastQC/SRR3099585_chr18_fastqc.html,
FastQC/SRR3099586_chr18_fastqc.html, FastQC/SRR3099587_chr18_fastqc.html,
multiqc_report.html
    jobid: 0
    reason: Input files updated by another job: multiqc_report.html,
FastQC/SRR3099587_chr18_fastqc.html, FastQC/SRR3099585_chr18_fastqc.html,
FastQC/SRR3099586_chr18_fastqc.html
    resources: tmpdir=/tmp

[Sun Jun 12 21:21:13 2022]
Finished job 0.
5 of 5 steps (100%) done
Complete log: .snakemake/log/2022-06-12T212049.793914.snakemake.log
```

1.22 Get input file names from the file system

To deduce the identifiers (eg. IDs) of files in a directory, use the inbuilt `glob_wildcards` function, eg.:

```
IDs, = glob_wildcards("dirpath/{id}.txt")
```

`glob_wildcards()` matches the given pattern against the files present in the system and thereby infers the values for all wildcards in the pattern (`{id}` here).

Hint: Don't forget the coma after the name (left hand side, IDs here).

1.23 Snakemake DAG visualization

Snakemake uses `dot` tool (of the `graphviz` package) to create diagrams of the complete workflow (`--dag`) or the rules dependencies (`--rulegraph`):

```
snakemake --dag -s ex1_o7.smk | dot -Tpng > ex1_o7_dag.png
snakemake --rulegraph -s ex1_o7.smk | dot -Tpng > ex1_o7_rule.png
```

1.24 Conda environment

Snakemake supports using explicit conda environments on a per-rule basis: - add a conda: directive in the rule definition: `conda: myRuleEnvironment.yml` - run Snakemake with the `--use-conda` option

The specified environment will be created and activated on the fly by Snakemake and the rule will then be run in the conda environment.

1.25 Some other useful options

- dry-run, do not execute anything, display what would be done: `-n --dryrun`

- print the shell command: `-p --printshellcmds`
- print the reason for each rule execution: `-r --reason`
- print a summary and status of rule: `-D`
- limit the number of jobs in parallel: `-j 1` (cores: `-c 1`)
- automatically create HTML reports (`--report report.html`) containing runtime statistics, a visualization of the workflow topology, used software and data provenance information (need to add the `jinja2` package as a dependency)

all Snakemake options