Snakemake Tutorial

Summary

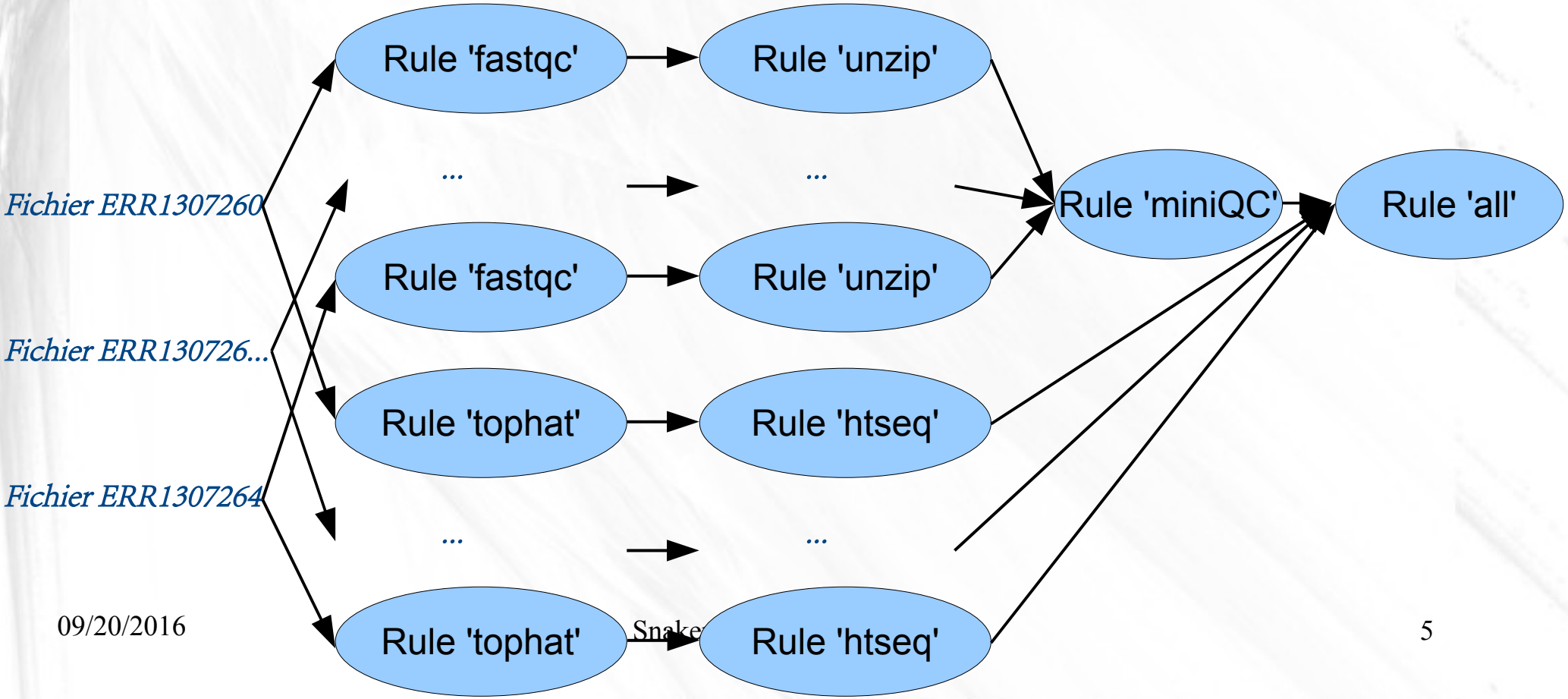
- Context
- Common uses (4 steps)
- Specific uses (2 steps)
- Still in progress

- An easy scripts scheduler in python
- Can be used on computing grid
- Can be used with virtual environment**S**

- **What for?**

For example, you need to compute the 1500 fastq files of the Lanner experiment in a succession of parallelised treatments.

- **The pipeline we will build**



- **Prerequisites**

  conda installed

  access to the BiRD grid

- **1st Step**
  - Virtual environment creation
  - Basic snakefile creation (first part)

- **Environment (shell commands)**

```
conda create -n myTutoEnv python=3.5 snakemake fastQC bioconductor-deseq2 -c bioconda

qlogin

source activate myTutoEnv

touch snakefile
```

- **Snakefile (1/2)**

```
import glob, ntpath

inFiles = set() ## set of all input files (only file names)

## extract file names from paths
inPaths = glob.glob( "/sandbox/ylelievre/singlecell/tuto/fastq/*.fastq.gz" )
for p in inPaths:
   inFiles.add( os.path.basename(p).replace(".fastq.gz", "") )

###############
```

Basename extractions from the files to be computed

- **Snakefile (2/2)**

The 1st rule (all) defines the target of the pipeline

```
rule all:
    input:
        I1 = expand("fastQC/{filename}_fastqc.zip",filename=inFiles)

###############

rule fastqc:
    input: "/sandbox/ylelievre/singlecell/tuto/fastq/{afile}.fastq.gz"
    output: "fastQC/{afile}_fastqc.zip"
    shell: "fastqc -o fastQC {input}"
```

The subsequent rules define the different steps to reach the target (here just one step)

- **Launch the pipeline (shell command)**

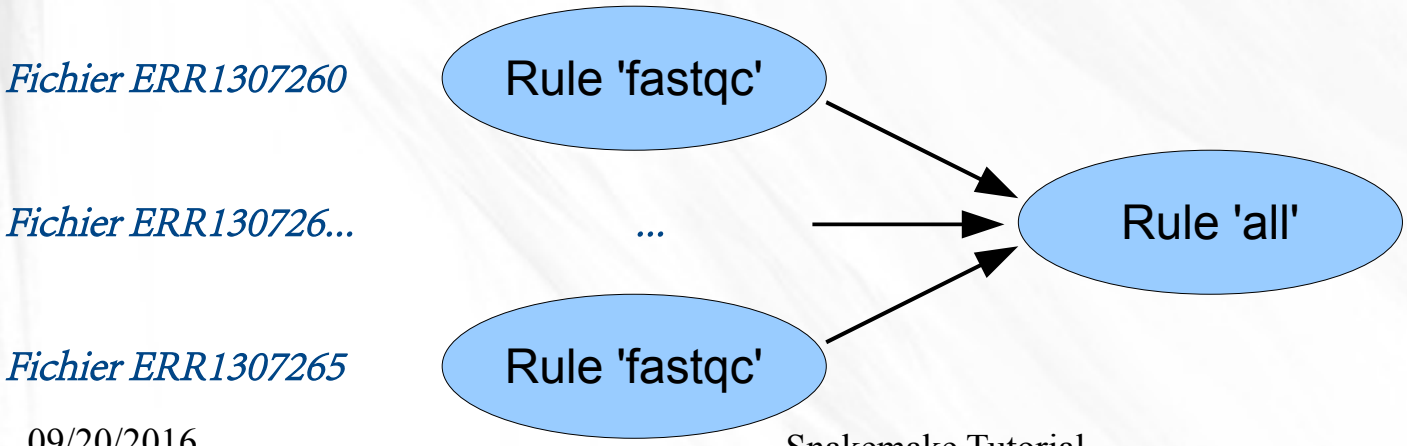
snakemake -p -n -s mySnakefileName

-p : print the commands executed

-n : dryrun

-s : if the Snakefile name is not 'Snakefile'

- **What can be observed ?**
  - The rules are executed sequentially (SGE not used)
  - The 5 rules fastqc are executed before the rule all
  - There is not a specific order between the fastqc rules



- **Preparation of the next step**
  - Delete a zip file (ERR1307264) from the directory ./fastQC

- **2nd Step**
  - Basic snakefile creation (second part)

- **Snakefile update**

```
rule all:
   input:
      I1 = expand("fastQC/{filename}_unzip_fastqc.done",filename=inFiles)

###############
...

rule unzip_fastqc:
   input:
      zips = "fastQC/{zipfile}_fastqc.zip"
   output: touch("fastQC/{zipfile}_unzip_fastqc.done")
   shell: "unzip -o {input.zips} -d ./fastQC"
```

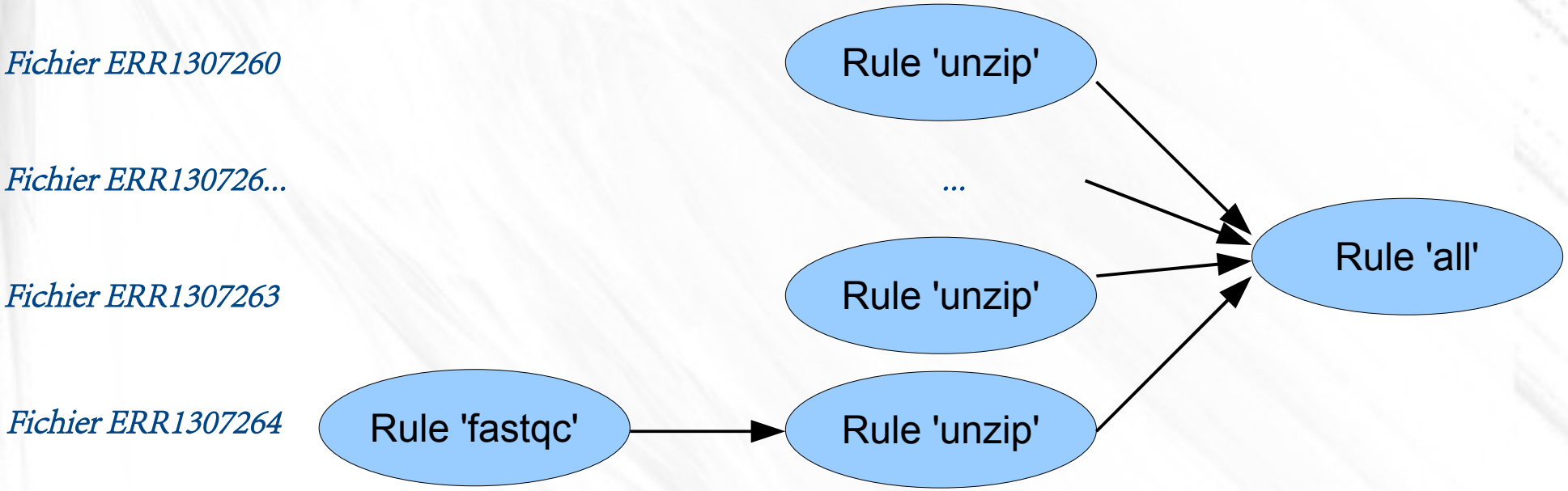
Input modified

Rule unzip_fastqc added

- **Launch the pipeline (shell command)**

```
snakemake -p
```

- **What can be observed ?**
  - All the rules fastqc are not necessarily executed before the unzip rules



- **Preparation of the next step**
  - Delete 2 zip files (ERR1307263 and ERR1307264) from the directory ./fastQC
  - Delete 2 done files (ERR1307262 and ERR1307264) from the directory ./fastQC

- **3rd Step**
  - Basic snakefile creation (second part)
  - Missing files and snakemake

- **Snakefile update**

```
rule all:
   input: "QC/miniQC.done"

##############
...

rule miniQC:
   input:
      I1 = expand("fastQC/{filename}_unzip_fastqc.done",filename=inFiles)
   output: touch("QC/miniQC.done")
   shell: "./miniQC.sh"
```

Input modified

Rule miniQC added

- **Launch the pipeline (shell command)**

```
snakemake -p
```

- **What can be observed ?**

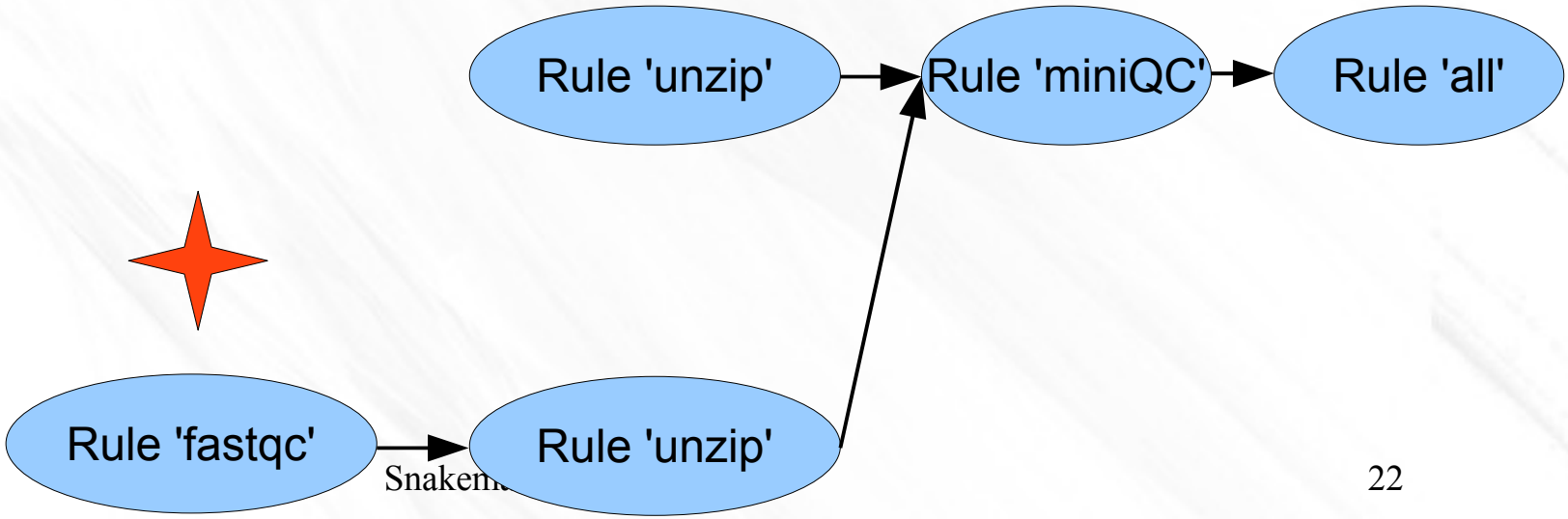
Fichier ERR1307260

Fichier ERR1307261

Fichier ERR1307262

Fichier ERR1307263

Fichier ERR1307264



- **Preparation of the next step**
  - Delete 2 zip files (ERR1307263 and ERR1307264) from the directory ./fastQC
  - Delete 2 done files (ERR1307263 and ERR1307264) from the directory ./fastQC
  - Delete the miniQC.done file from the directory ./QC

- **4th Step**
  - Snakefile with a config file

- **Snakefile with a config file (1/3)**

```
import glob, ntpath

configfile: "config_tuto.json"

inFiles = set() ## set of all input files (only file names)

## extract file names from paths
inPaths = glob.glob( config["FASTQ_PATH"]+"*.fastq.gz" )
for p in inPaths:
   inFiles.add( os.path.basename(p).replace(".fastq.gz", "") )

###############
```

Load the config file

Call of the field FASTQ_PATH from the json file

- **Snakefile with a config file (2/3)**

```
rule all:
    input: config["ANALYSIS_PATH"]+"QC/miniQC.done"

##############
rule fastqc:
    input: config["FASTQ_PATH"]+"{afile}.fastq.gz"
    output: config["ANALYSIS_PATH"]+"fastQC/{afile}_fastqc.zip"
    params: analysis_path=config["ANALYSIS_PATH"]
    shell: "fastqc -o {params.analysis_path}fastQC {input}"
```

params: is required in order to inject config field into the shell script

- **Snakefile with a config file (3/3)**

```
rule unzip_fastqc:
    input:
        zips = config["ANALYSIS_PATH"]+"fastQC/{zipfile}_fastqc.zip"
    output: touch(config["ANALYSIS_PATH"]+"fastQC/{zipfile}_unzip_fastqc.done")
    params: analysis_path=config["ANALYSIS_PATH"]
    shell: "unzip -o {input.zips} -d {params.analysis_path}fastQC"

rule miniQC:
    input:
        expand(config["ANALYSIS_PATH"]+"fastQC/{filename}_unzip_fastqc.done",filename=inFiles)
    output: touch(config["ANALYSIS_PATH"]+"QC/miniQC.done")
    params: fastq_type=config["FASTQ_TYPE"]
    shell: "./miniQC.sh {params.fastq_type}"
```

- **Launch the pipeline (shell command)**

```
snakemake -p
```

- **What can be observed ?**

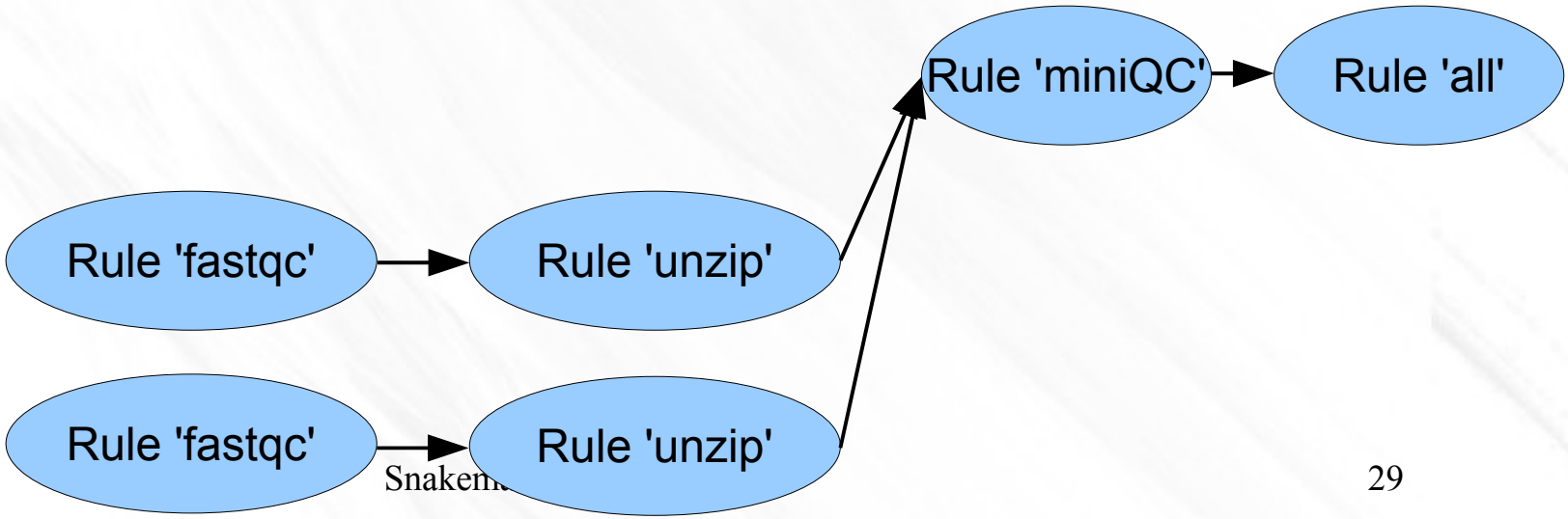
Fichier ERR1307260

Fichier ERR1307261

Fichier ERR1307262

Fichier ERR1307263

Fichier ERR1307264



- **Preparation of the next step**

  - Delete all zip files from the directory ./fastQC
  - Delete all done files from the directory ./fastQC
  - Delete the miniQC.done file from the directory ./QC
  - Shell commands :

    source deactivate

    exit

- **5th Step**
  - Snakemake and computing grid
  - Snakemake and wrappers

- **Environment (shell commands)**

```
source activate myTutoEnv
mkdir logs
snakemake -p  --cluster "qsub -o ./logs/ -e ./logs/" --jobs 50
```

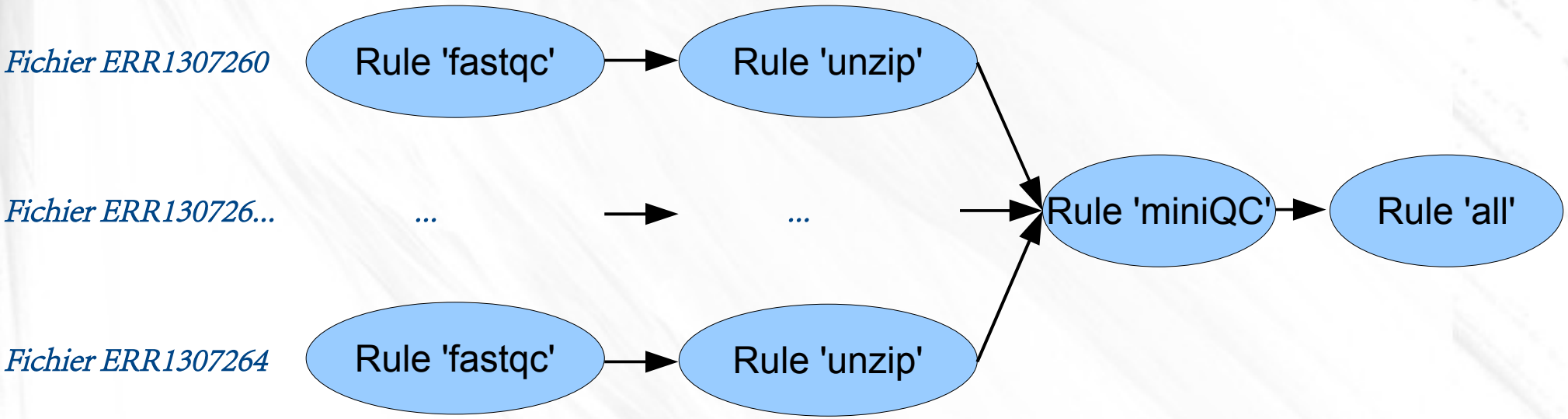
Use of the option –cluster to perform snakemake on SGE

-o and -e define the repository where SGE send the output and error

--jobs define the number of threads allocated

- **What can be observed ? (1/2)**

  If you were lucky, you would observe this!



- **What can be observed ? (2/2)**

  Except for the persons who are doing this tutorial directly in their home directory (…), your jobs should be in error because your current working directory has not been passed to the computing grid.

  More over, your virtual environment variables, also, have not been transmitted to the grid.

- **Environment (shell commands)**

ctrl c
touch myGridWrapper.sh

We will use a wrapper in order to transmit the environment variables

- **myGridWrapper.sh (shell commands)**

```
#$ -S /bin/bash
#$ -cwd
#$ -V
{exec_job}
```

Transmit the current working directory

Transmit the virtual environment variables on the node

Execute the script

- **Shell commands**

```
snakemake -p --cluster "qsub -o ./logs/ -e ./logs/" --jobs 50
--jobscript myGridWrapper.sh
```

- **What can be observed ?**

  Missed again!

  You should get an error like 'OSError: Missing files after 5 seconds:'

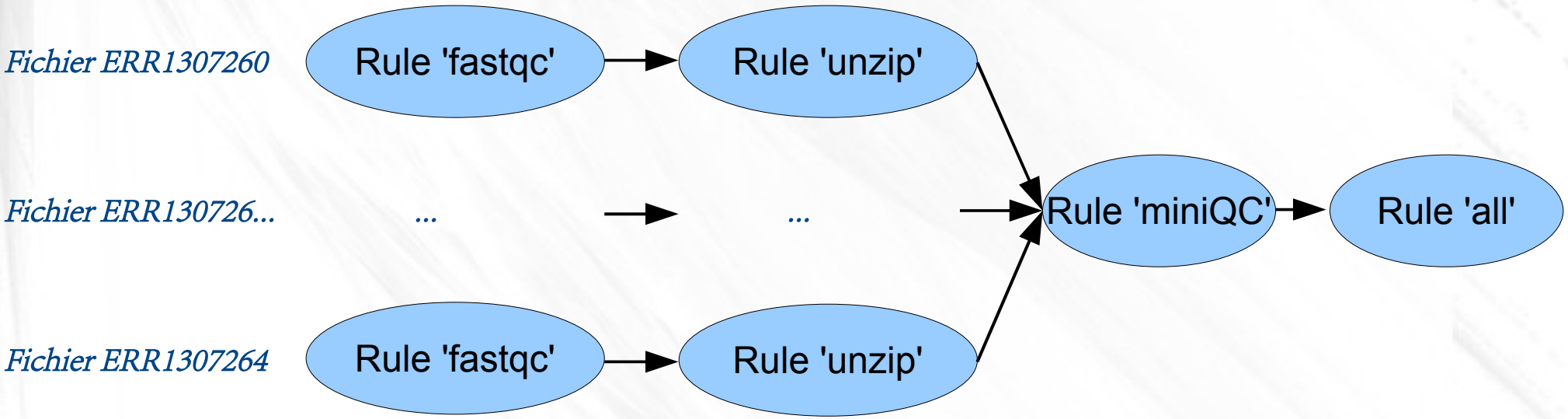
- **Shell commands (try again)**

ctrl c
snakemake -p --latency-wait 60 --cluster "qsub -o ./logs/ -e ./logs/" --jobs 50 --jobscript myGridWrapper.sh

Snakemake needs some latency to properly work with a grid engine

- **What can be observed ?**

  Finally!!!!



- **6th Step**
  - Snakemake and virtual environmentS

- **Let's play with virtual environments!**

  Often, we need to use antinomic version of tools in a same pipeline.

  One solution is the use of virtual environments.

  Example : use snakmake with python 3 and tophat2 with python 2

- **Environment (shell commands)**

```
conda create --name myTophatTutoEnv python=2.7.8 tophat
htseq pysam=0.8.3 -c bioconda
```

Preparation of a virtual environment which is compliant with tophat and htseqcount

- **Snakefile with a config file (1/2)**

Update rule all

```
rule all:
    input:
        I1 = config["ANALYSIS_PATH"]+"QC/miniQC.done",
        I2 = expand(config["ANALYSIS_PATH"]+"counts/{filename}.counts",filename=inFiles)
```

Add the following rules

```
rule tophat2_SE:
    input: config["FASTQ_PATH"]+"{bamname}.fastq.gz"
    output: config["ANALYSIS_PATH"]+"BAM/{bamname}/accepted_hits.bam"
    params: cpu=config["TOPHAT_CPU"], gene=config["GENE_REFERENCE"],
bowtie=config["BOWTIE_INDEX"], analysis_path=config["ANALYSIS_PATH"]
    shell: """
    source activate myTophatTutoEnv
    tophat2 -p {params.cpu} -G {params.gene} -o {params.analysis_path}BAM/
{wildcards.bamname} {params.bowtie} {input}
"""
```

- **Snakefile with a config file (2/2)**

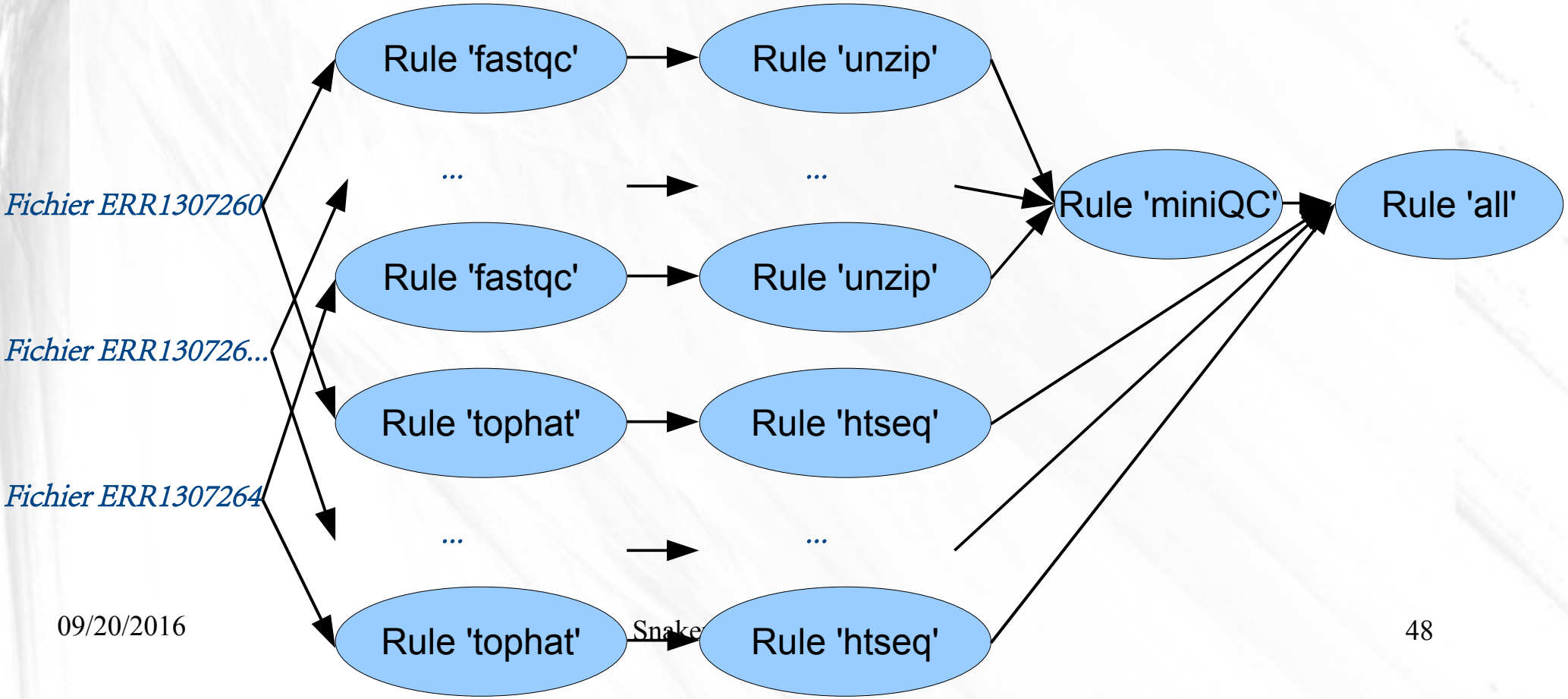
```
rule htseqcount:
    input: config["ANALYSIS_PATH"]+"BAM/{htseqname}/accepted_hits.bam"
    output: config["ANALYSIS_PATH"]+"counts/{htseqname}.counts"
    params: gene=config["GENE_REFERENCE"],
analysis_path=config["ANALYSIS_PATH"]
    shell: """
    source activate myTophatTutoEnv
    htseq-count -f bam -r pos -i gene_name -s no {input} {params.gene} >
{params.analysis_path}counts/{wildcards.htseqname}.counts
    """
```

- **Preparation of the next step**
  - Delete all zip files from the directory ./fastQC
  - Delete all done files from the directory ./fastQC
  - Delete the miniQC.done file from the directory ./QC

- **Shell commands**

```
snakemake -p --latency-wait 60 --cluster "qsub -o ./logs/ -e ./logs/" --jobs 50 --jobscript myGridWrapper.sh
```

- **What can be observed ?**



- **A young technology**
- **Problem with the NFS file system (solved with BeeGFS)**

Thank you for your perseverance!