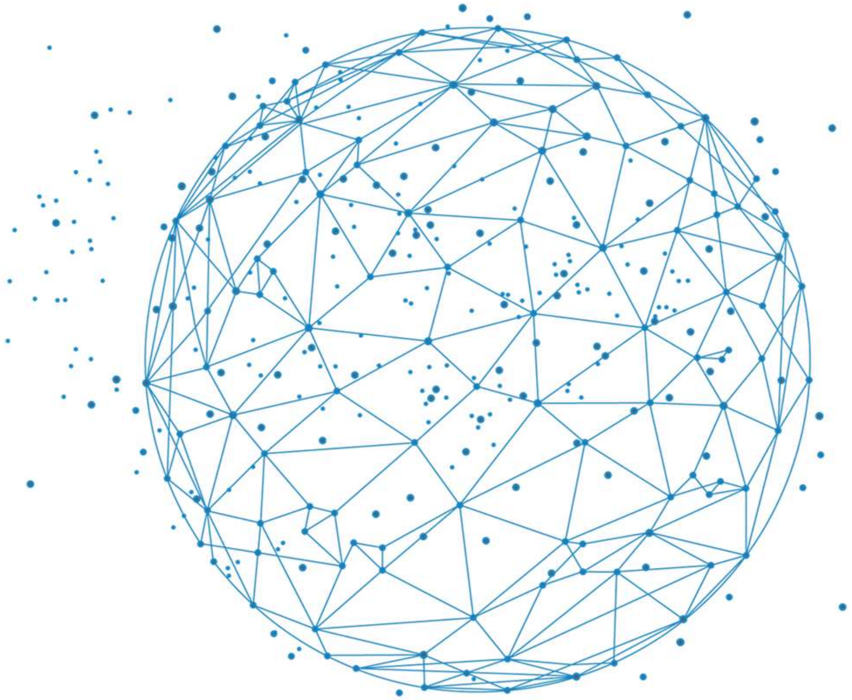
WF4BIOINFO, 15 octobre 2024, Paris



Snakemake Use-Case:

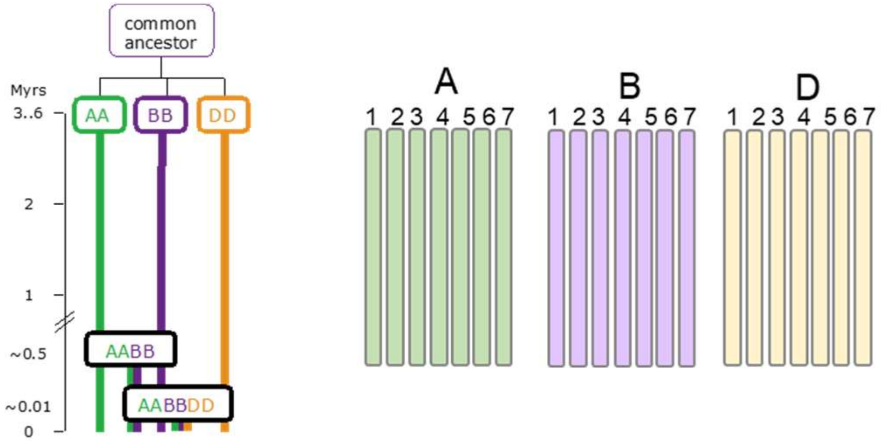
A pipeline for TE annotation of *Triticeae* genomes

Pauline LASSERRE-ZUBER

Introduction

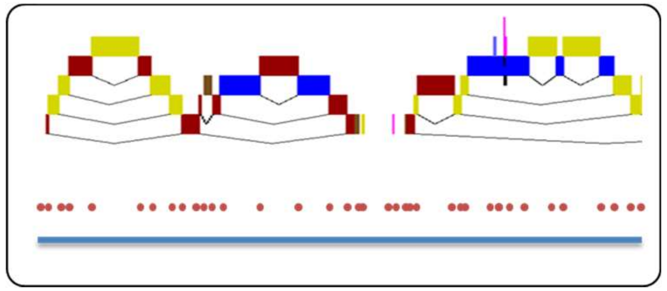
❑ **Wheat genome**:

- hexaploid homozygous
- 21 chromosomes
- ~15 Gb
- 85 % of repeats: Transposable Elements (TE)



❑ **TE annotation**:

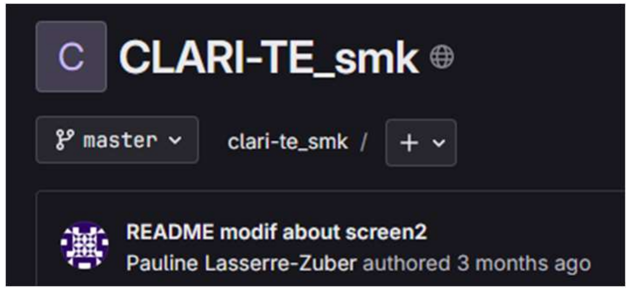
- *Triticeae* dedicated tool: CLARI-TE (Daron et al, 2014*)*
- perl scripts with old dependencies



❑ **FAIR needs**:

- make this tool **durable** and **portable**
- **automate** the pipeline steps
- **speed up** the process (parallelization on data +++)

=> findable, accessible, interoperable, reusable



C CLARI-TE_smk

master ⌄ clari-te_smk / + ⌄

README modif about screen2

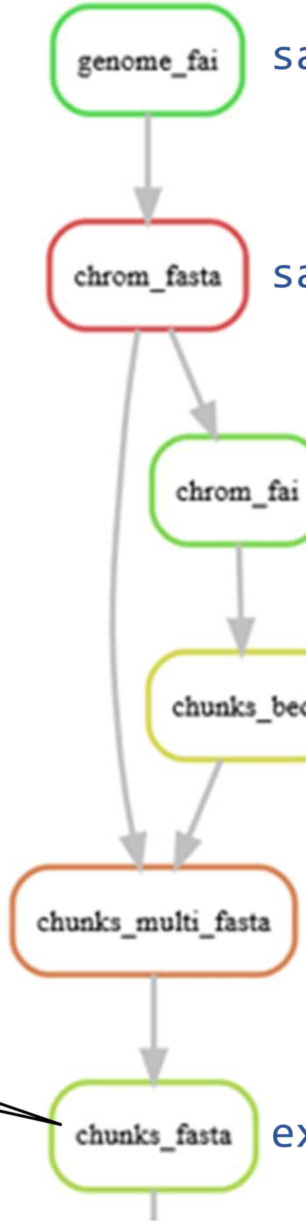
Pauline Lasserre-Zuber authored 3 months ago

https://forgemia.inra.fr/umr-gdec/clari-te_smk

CLARI-TE_smk

https://forgemia.inra.fr/umr-gdec/clari-te_smk

Rulegraph



`samtools faidx`

`samtools faidx`

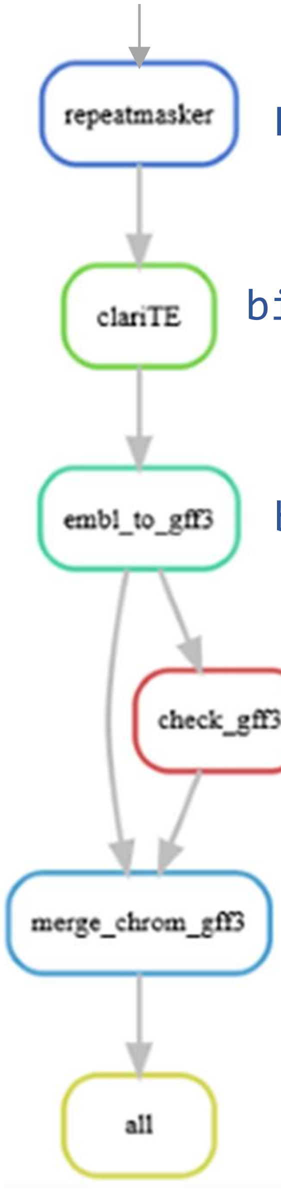
`samtools faidx`

`bedtools makewindows`

`bedtools getfasta`

checkpoint rule

`exonerate fastaexplode`



`RepeatMasker`

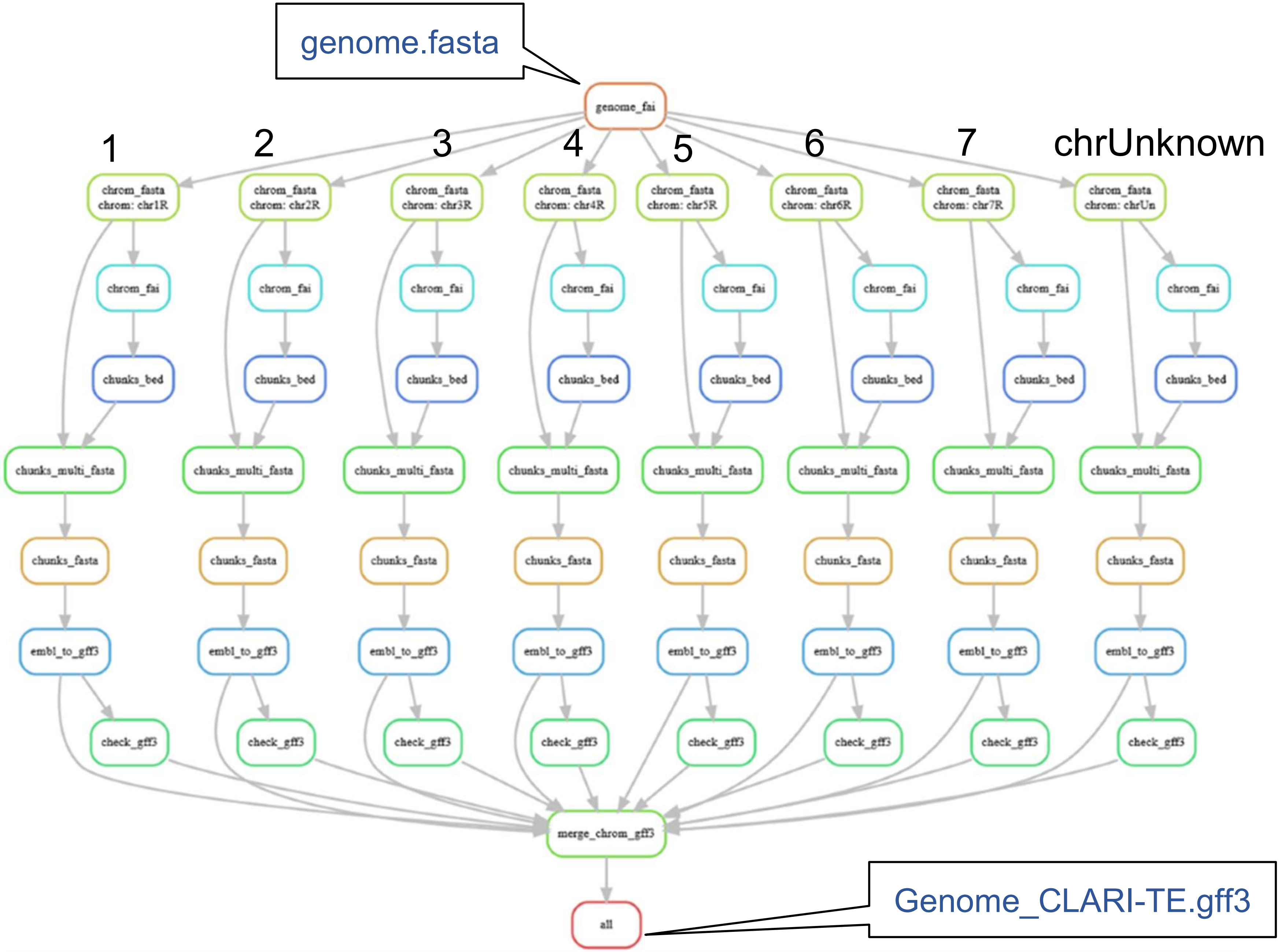
`bin/clariTE_1.0/clariTE.pl`

`bin/embl_to_gff3.sh`

`genometools gff3validator`

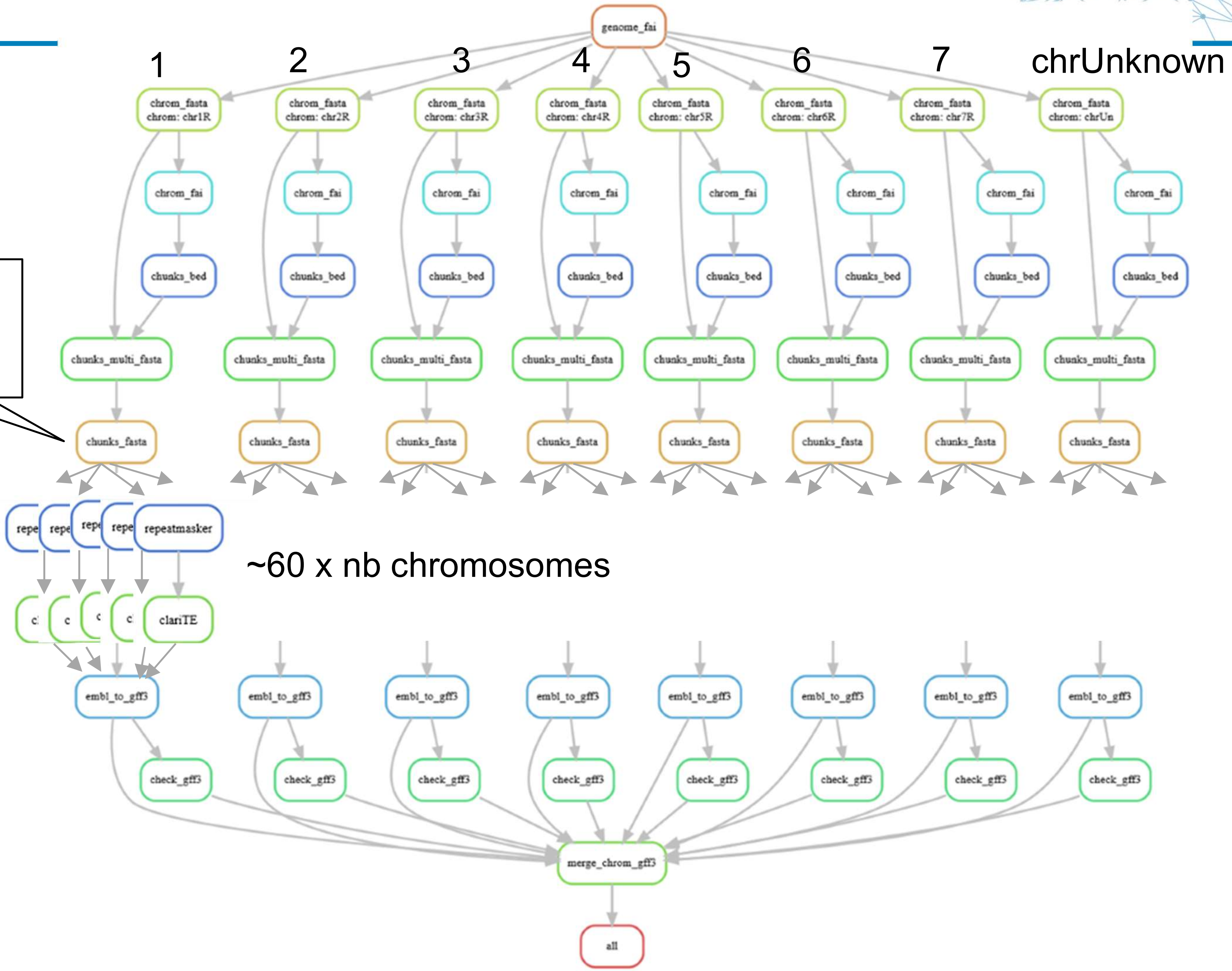
`genometools gff3 -tidy`

Diagram
(before checkpoint rule)



CLARI-TE_smk

Diagram



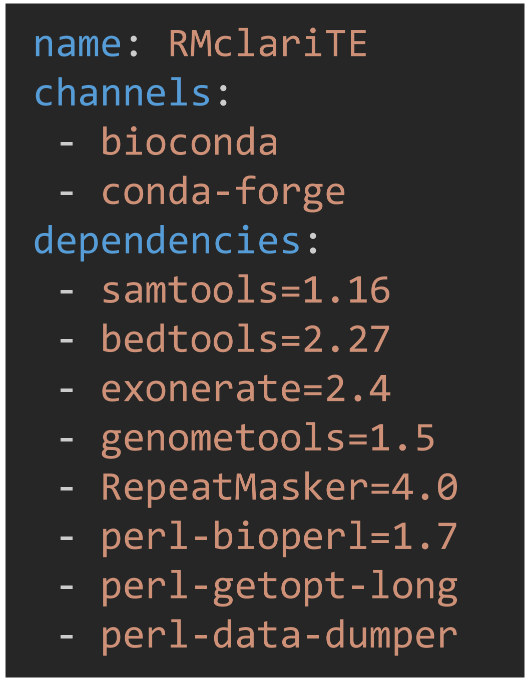
checkpoint rule: directory()

~60 x nb chromosomes

To construct the pipeline

- ❑ Singularity **CLARI-TE_smk.sif** → APPTAINER

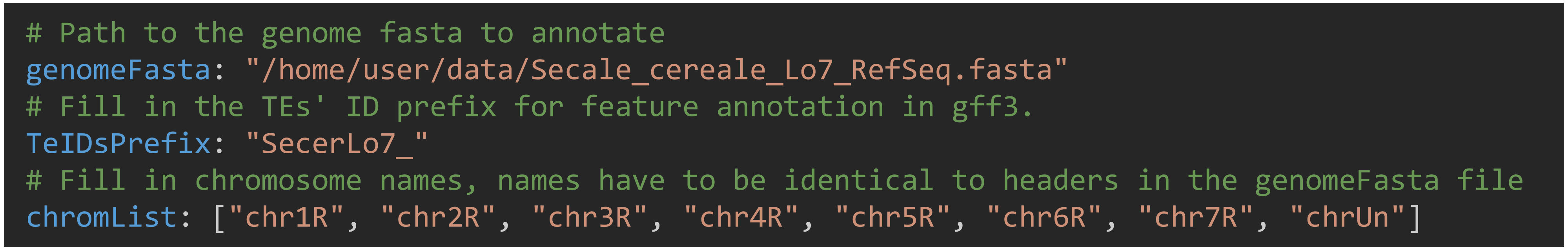
  .def: conda env + CLARI-TE bin + non conda tools + local scripts



```
name: RMclariTE
channels:
 - bioconda
 - conda-forge
dependencies:
 - samtools=1.16
 - bedtools=2.27
 - exonerate=2.4
 - genometools=1.5
 - RepeatMasker=4.0
 - perl-bioperl=1.7
 - perl-getopt-long
 - perl-data-dumper
```

- ❑ **smk_config.yaml**

  User analysis parameters



```
# Path to the genome fasta to annotate
genomeFasta: "/home/user/data/Secale_cereale_Lo7_RefSeq.fasta"
# Fill in the TEs' ID prefix for feature annotation in gff3.
TeIDsPrefix: "SecerLo7_"
# Fill in chromosome names, names have to be identical to headers in the genomeFasta file
chromList: ["chr1R", "chr2R", "chr3R", "chr4R", "chr5R", "chr6R", "chr7R", "chrUn"]
```

To launch the pipeline

❑ Singularity **CLARI-TE_smk.sif**

.def: conda env + CLARI-TE bin + non conda tools + local scripts

❑ **Smk_config.yaml**

User analysis parameters

❑ **Cluster profile**

config.yaml

❑ **Input file**

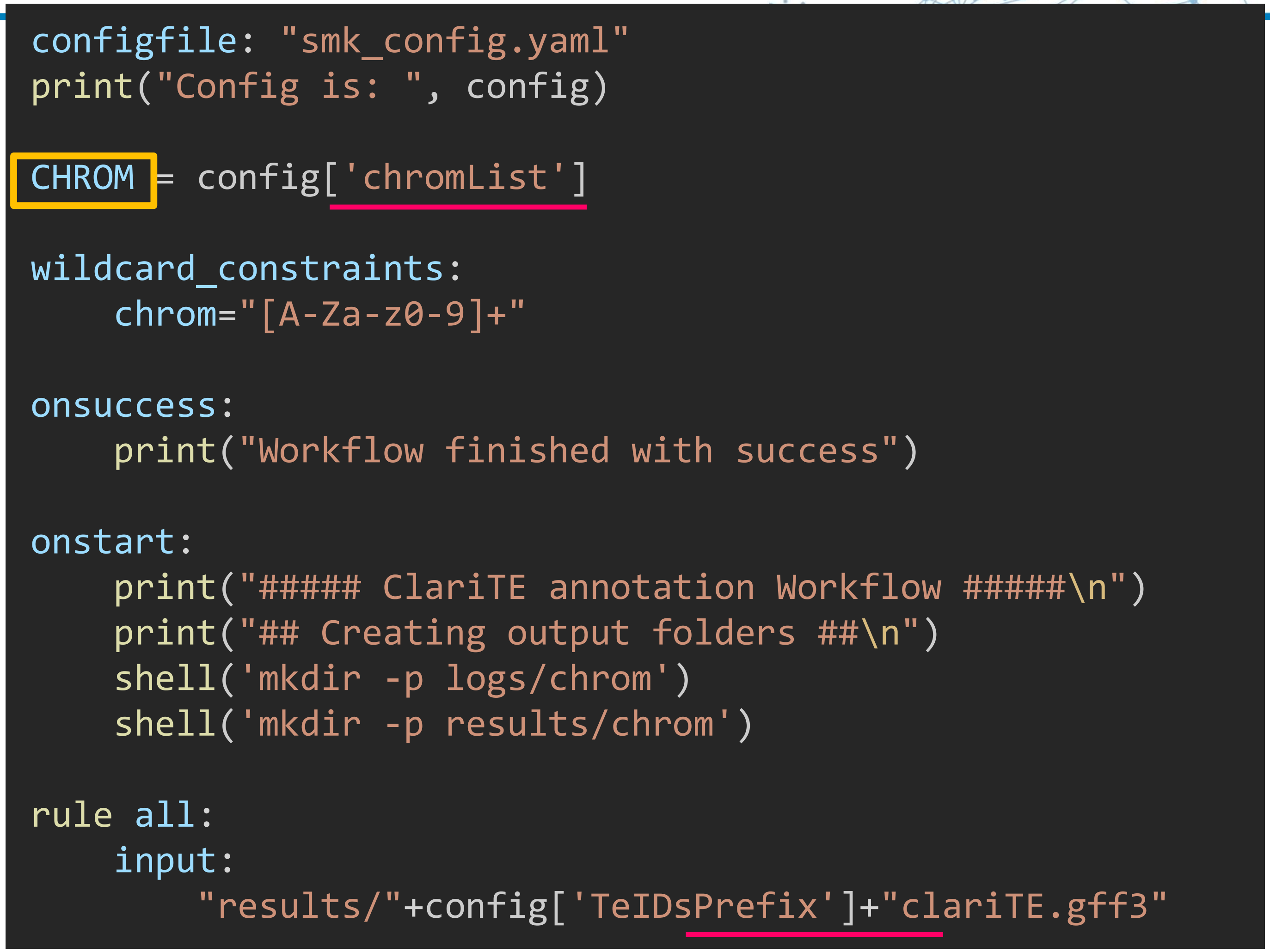
genome.fasta

❑ **Snakefile**

The snakefile

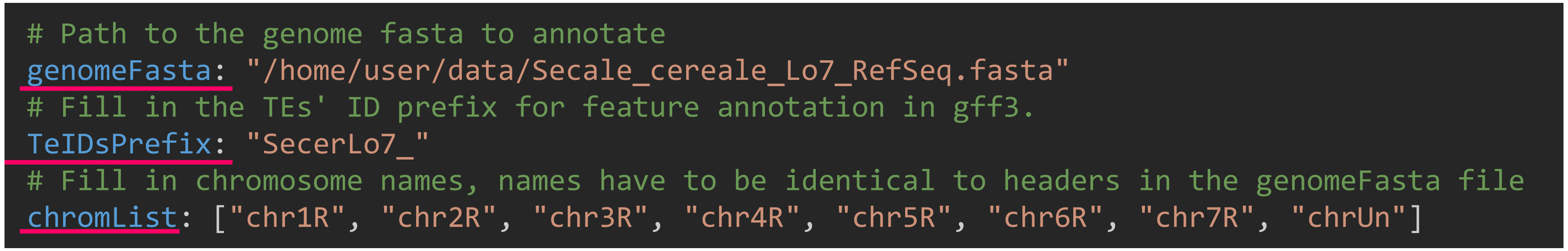
Pipe initialization
and
target rule

snakefile:



```
configfile: "smk_config.yaml"
print("Config is: ", config)

CHROM = config['chromList']

wildcard_constraints:
    chrom="[A-Za-z0-9]+"

onsuccess:
    print("Workflow finished with success")

onstart:
    print("##### ClariTE annotation Workflow #####\n")
    print("## Creating output folders ##\n")
    shell('mkdir -p logs/chrom')
    shell('mkdir -p results/chrom')

rule all:
    input:
        "results/"+config['TeIDsPrefix']+"clariTE.gff3"
```

smk_config.yaml:



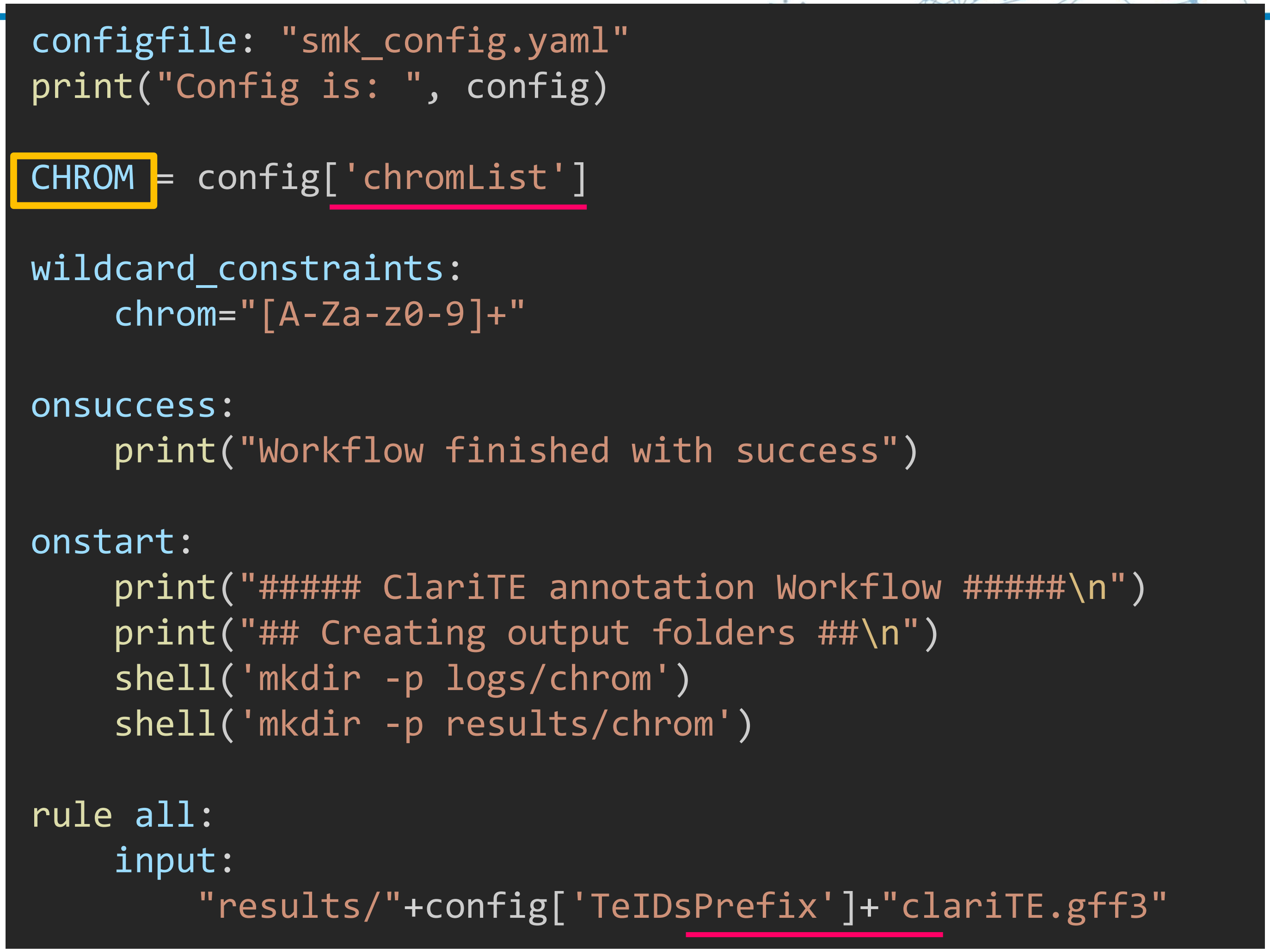
```
# Path to the genome fasta to annotate
genomeFasta: "/home/user/data/Secale_cereale_Lo7_RefSeq.fasta"
# Fill in the TEs' ID prefix for feature annotation in gff3.
TeIDsPrefix: "SecerLo7_"
# Fill in chromosome names, names have to be identical to headers in the genomeFasta file
chromList: ["chr1R", "chr2R", "chr3R", "chr4R", "chr5R", "chr6R", "chr7R", "chrUn"]
```

The snakefile

Pipe initialization
and
target rule

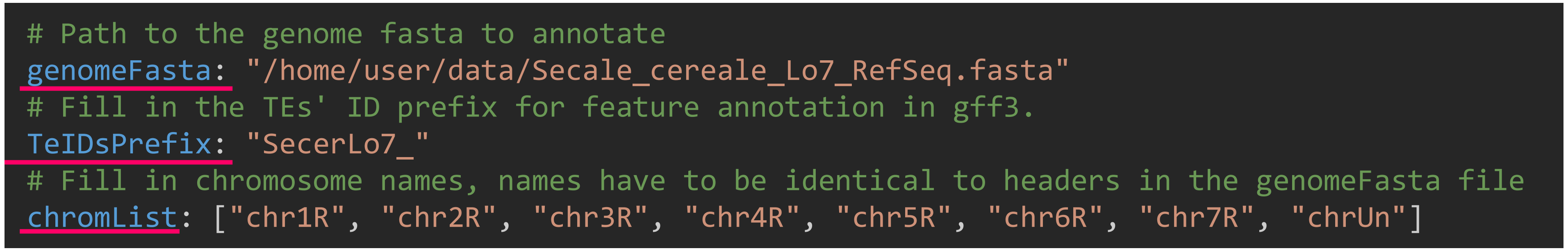
NB: no snakefile modification

snakefile:



```
configfile: "smk_config.yaml"
print("Config is: ", config)

CHROM = config['chromList']

wildcard_constraints:
    chrom="[A-Za-z0-9]+"

onsuccess:
    print("Workflow finished with success")

onstart:
    print("##### ClariTE annotation Workflow #####\n")
    print("## Creating output folders ##\n")
    shell('mkdir -p logs/chrom')
    shell('mkdir -p results/chrom')

rule all:
    input:
        "results/"+config['TeIDsPrefix']+"clariTE.gff3"
```

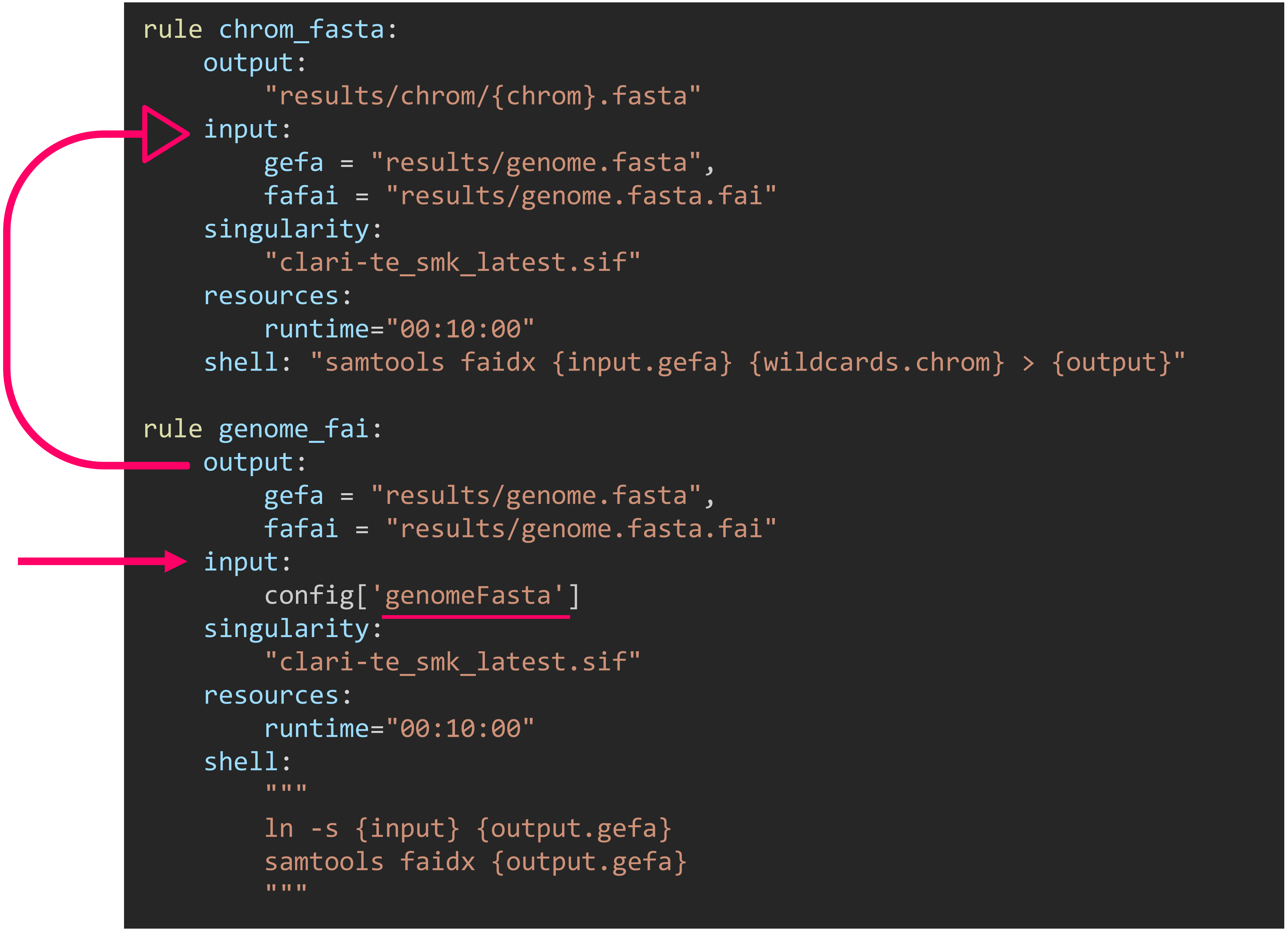
smk_config.yaml:



```
# Path to the genome fasta to annotate
genomeFasta: "/home/user/data/Secale_cereale_Lo7_RefSeq.fasta"
# Fill in the TEs' ID prefix for feature annotation in gff3.
TeIDsPrefix: "SecerLo7_"
# Fill in chromosome names, names have to be identical to headers in the genomeFasta file
chromList: ["chr1R", "chr2R", "chr3R", "chr4R", "chr5R", "chr6R", "chr7R", "chrUn"]
```

The snakefile

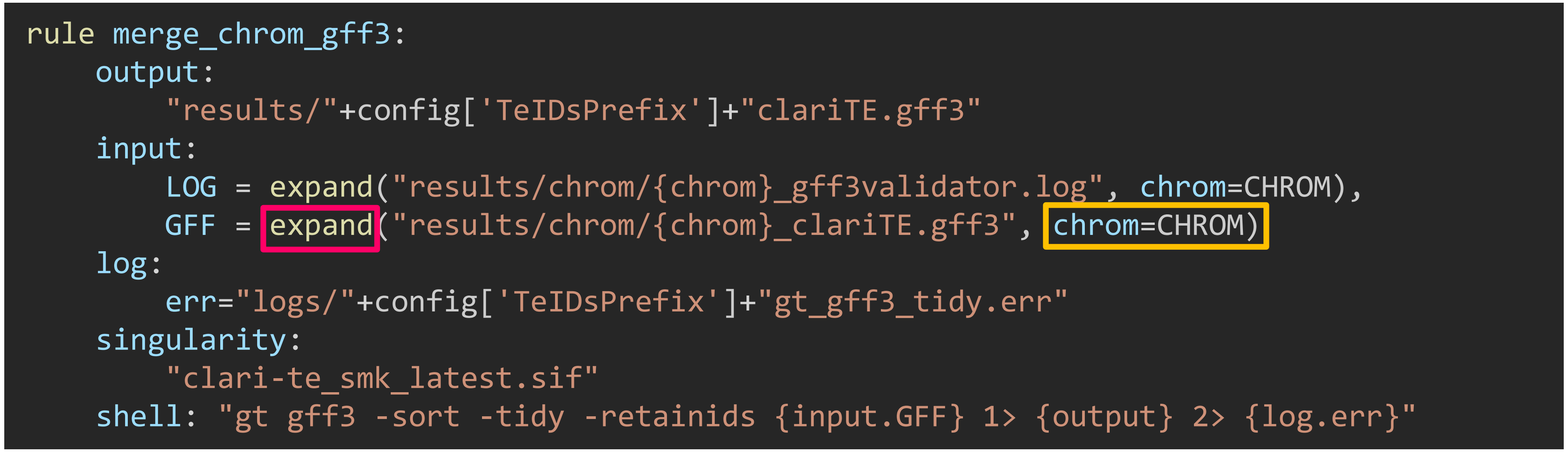
rules ‘dependency’



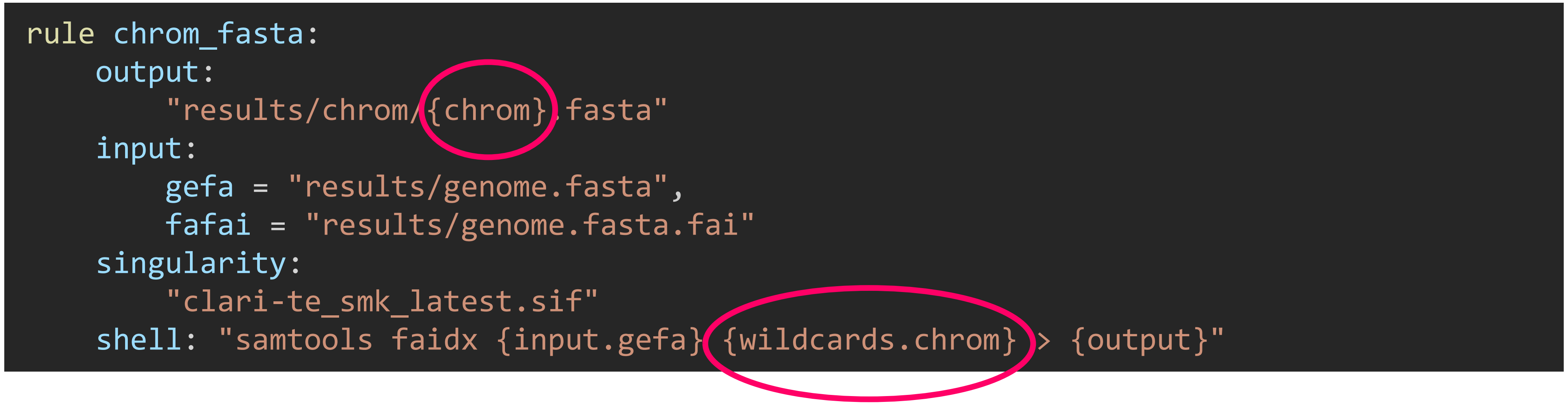
```
rule chrom_fasta:
    output:
        "results/chrom/{chrom}.fasta"
    input:
        gefa = "results/genome.fasta",
        fafai = "results/genome.fasta.fai"
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="00:10:00"
    shell: "samtools faidx {input.gefa} {wildcards.chrom} > {output}"

rule genome_fai:
    output:
        gefa = "results/genome.fasta",
        fafai = "results/genome.fasta.fai"
    input:
        config['genomeFasta']
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="00:10:00"
    shell:
        """
        ln -s {input} {output.gefa}
        samtools faidx {output.gefa}
        """
```

The snakefile

wildcards
and
expand



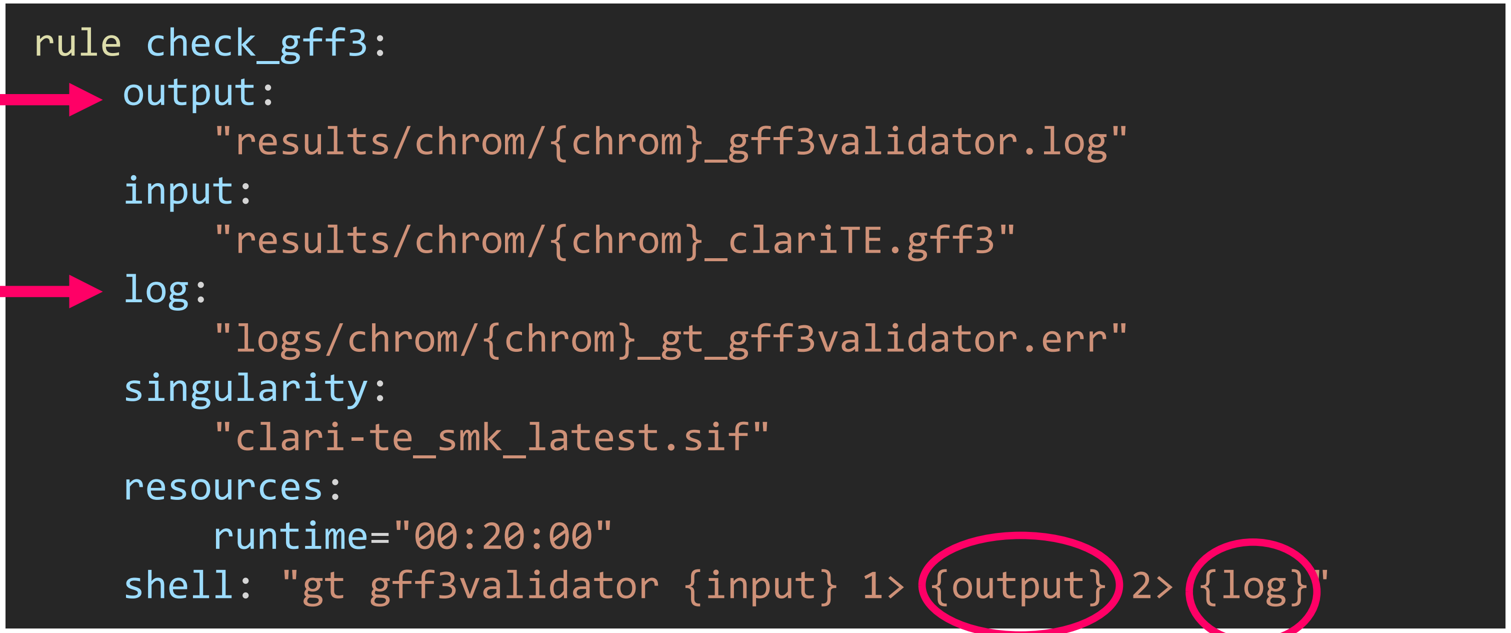
```
rule merge_chrom_gff3:
    output:
        "results/"+config['TeIDsPrefix']+"clariTE.gff3"
    input:
        LOG = expand("results/chrom/{chrom}_gff3validator.log", chrom=CHROM),
        GFF = expand("results/chrom/{chrom}_clariTE.gff3", chrom=CHROM)
    log:
        err="logs/"+config['TeIDsPrefix']+"gt_gff3_tidy.err"
    singularity:
        "clari-te_smk_latest.sif"
    shell: "gt gff3 -sort -tidy -retainids {input.GFF} 1> {output} 2> {log.err}"
```



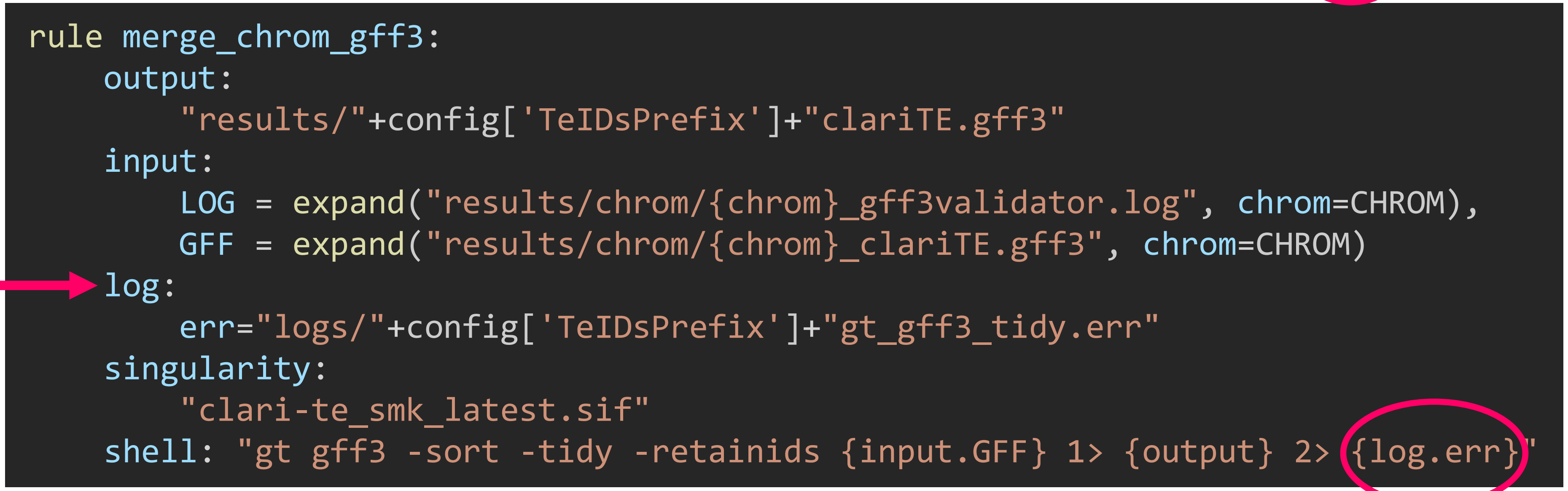
```
rule chrom_fasta:
    output:
        "results/chrom/{chrom}.fasta"
    input:
        gefa = "results/genome.fasta",
        fafai = "results/genome.fasta.fai"
    singularity:
        "clari-te_smk_latest.sif"
    shell: "samtools faidx {input.gefa} {wildcards.chrom} > {output}"
```

The snakefile

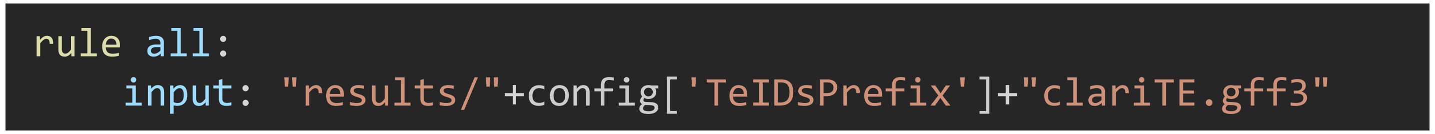
log output



```
rule check_gff3:
    output:
        "results/chrom/{chrom}_gff3validator.log"
    input:
        "results/chrom/{chrom}_clariTE.gff3"
    log:
        "logs/chrom/{chrom}_gt_gff3validator.err"
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="00:20:00"
    shell: "gt gff3validator {input} 1> {output} 2> {log}"
```



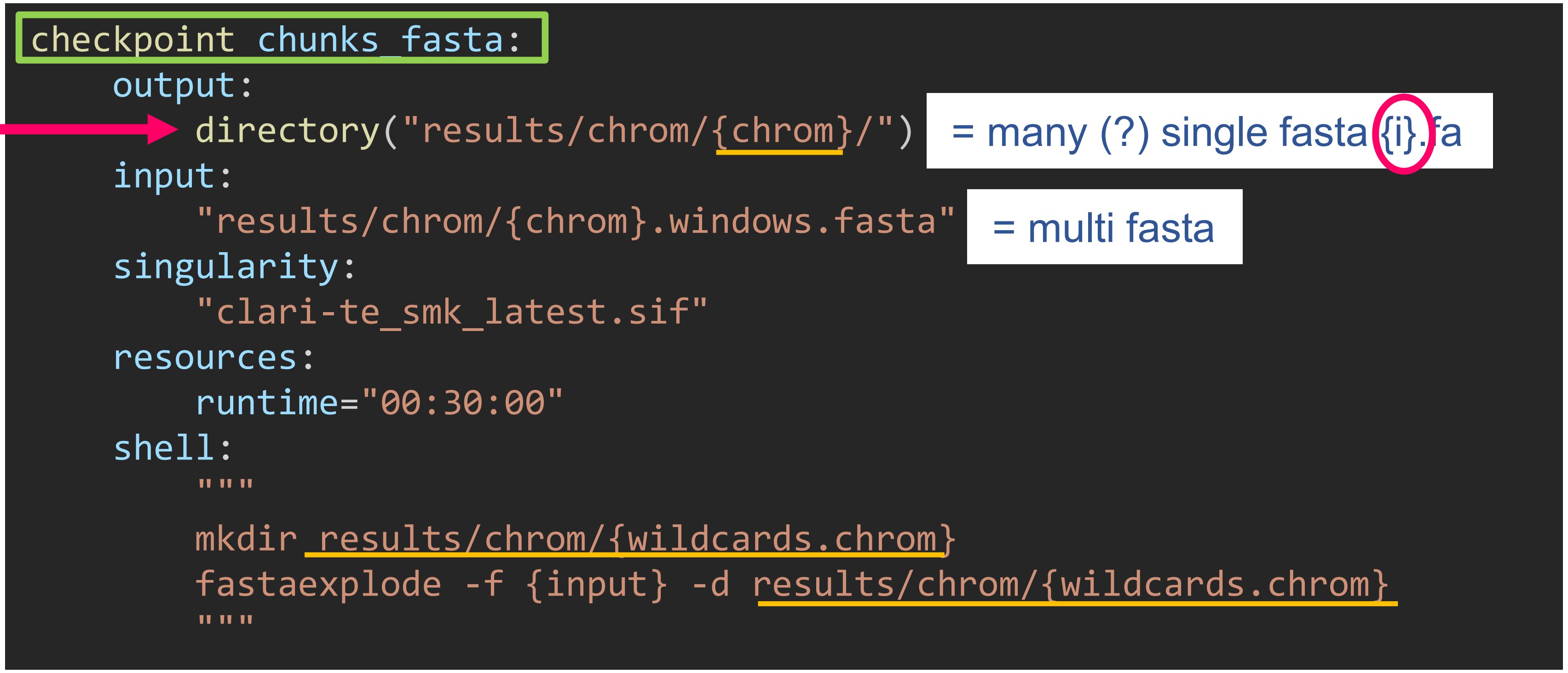
```
rule merge_chrom_gff3:
    output:
        "results/"+config['TeIDsPrefix']+"clariTE.gff3"
    input:
        LOG = expand("results/chrom/{chrom}_gff3validator.log", chrom=CHROM),
        GFF = expand("results/chrom/{chrom}_clariTE.gff3", chrom=CHROM)
    log:
        err="logs/"+config['TeIDsPrefix']+"gt_gff3_tidy.err"
    singularity:
        "clari-te_smk_latest.sif"
    shell: "gt gff3 -sort -tidy -retainids {input.GFF} 1> {output} 2> {log.err}"
```



```
rule all:
    input: "results/"+config['TeIDsPrefix']+"clariTE.gff3"
```

The snakefile

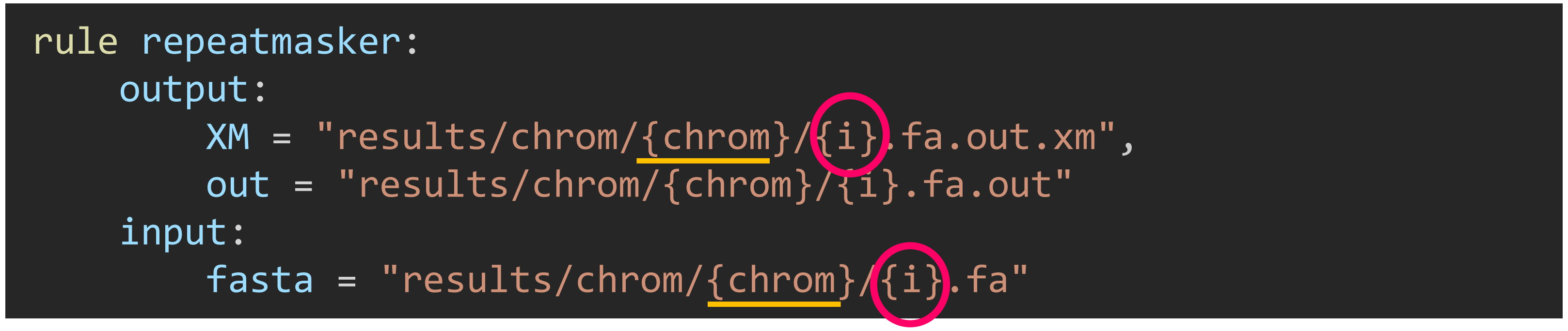
output = **directory()**



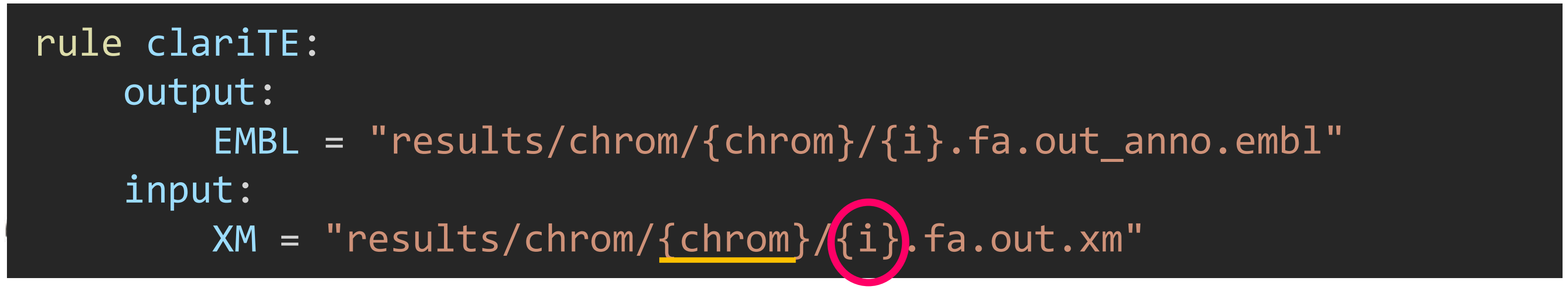
```
checkpoint chunks_fasta:
    output:
        directory("results/chrom/{chrom}/")
    input:
        "results/chrom/{chrom}.windows.fasta"
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="00:30:00"
    shell:
        """
        mkdir results/chrom/{wildcards.chrom}
        fastaexplode -f {input} -d results/chrom/{wildcards.chrom}
        """
```

= many (?) single fasta {i}.fa

= multi fasta



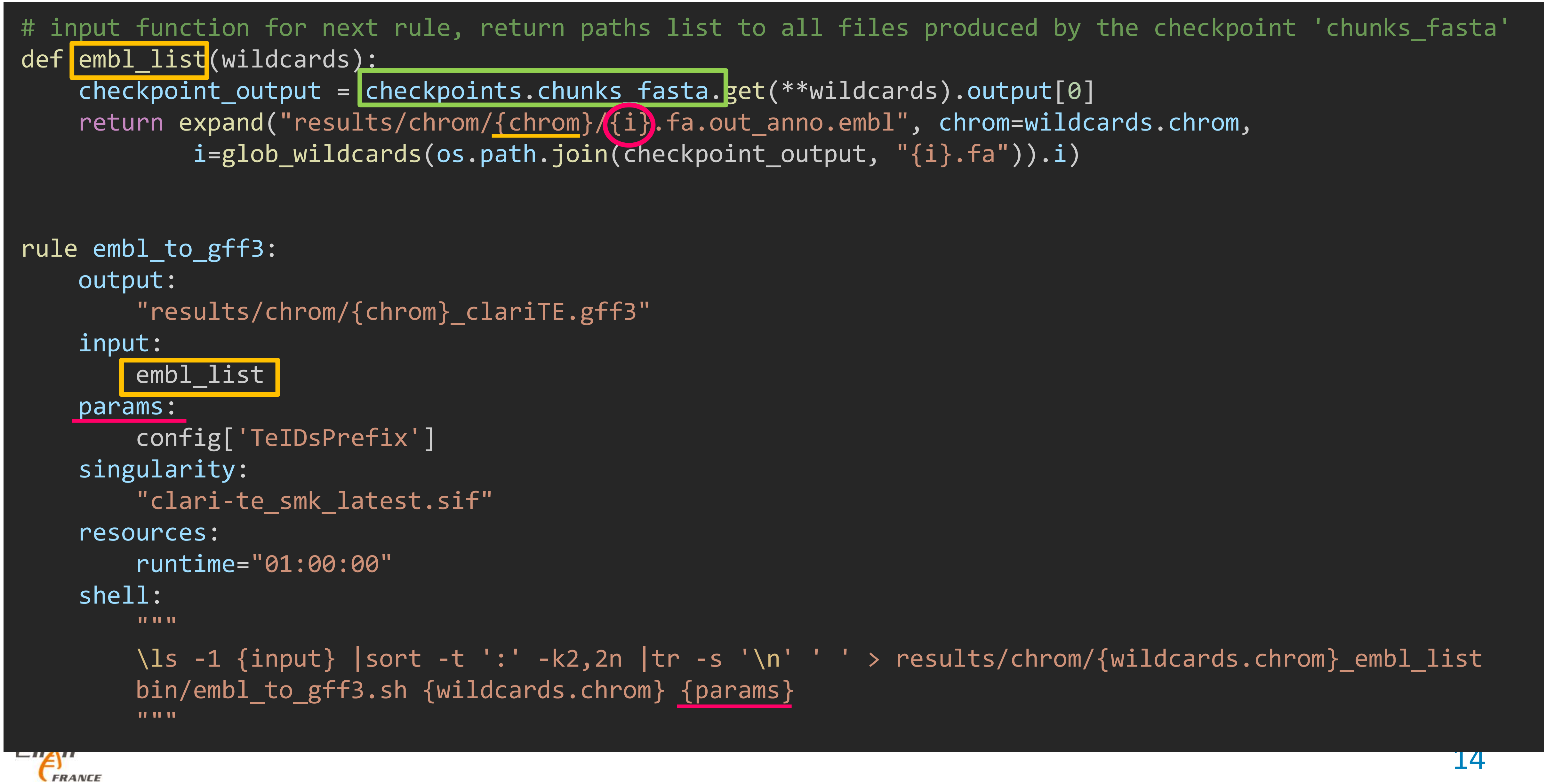
```
rule repeatmasker:
    output:
        XM = "results/chrom/{chrom}/{i}.fa.out.xm",
        out = "results/chrom/{chrom}/{i}.fa.out"
    input:
        fasta = "results/chrom/{chrom}/{i}.fa"
```



```
rule clariTE:
    output:
        EMBL = "results/chrom/{chrom}/{i}.fa.out_anno.embl"
    input:
        XM = "results/chrom/{chrom}/{i}.fa.out.xm"
```

The snakefile

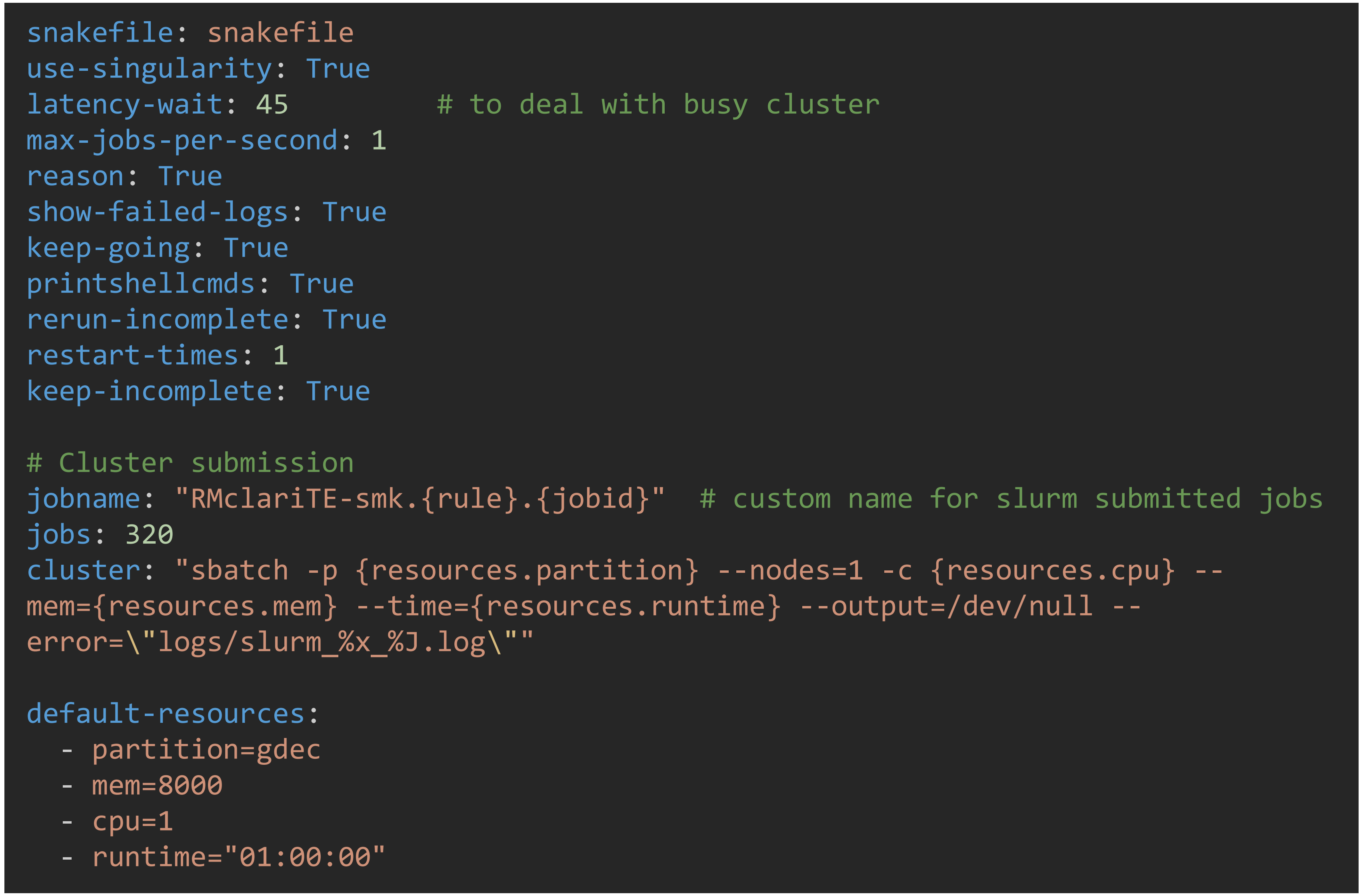
glob_wildcards



```
# input function for next rule, return paths list to all files produced by the checkpoint 'chunks_fasta'
def embl_list(wildcards):
    checkpoint_output = checkpoints.chunks_fasta.get(**wildcards).output[0]
    return expand("results/chrom/{chrom}/{i}.fa.out_anno.embl", chrom=wildcards.chrom,
            i=glob_wildcards(os.path.join(checkpoint_output, "{i}.fa")).i)


rule embl_to_gff3:
    output:
        "results/chrom/{chrom}_clariTE.gff3"
    input:
        embl_list
    params:
        config['TeIDsPrefix']
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="01:00:00"
    shell:
        """
        \ls -1 {input} |sort -t ':' -k2,2n |tr -s '\n' ' ' > results/chrom/{wildcards.chrom}_embl_list
        bin/embl_to_gff3.sh {wildcards.chrom} {params}
        """
```

The cluster profile

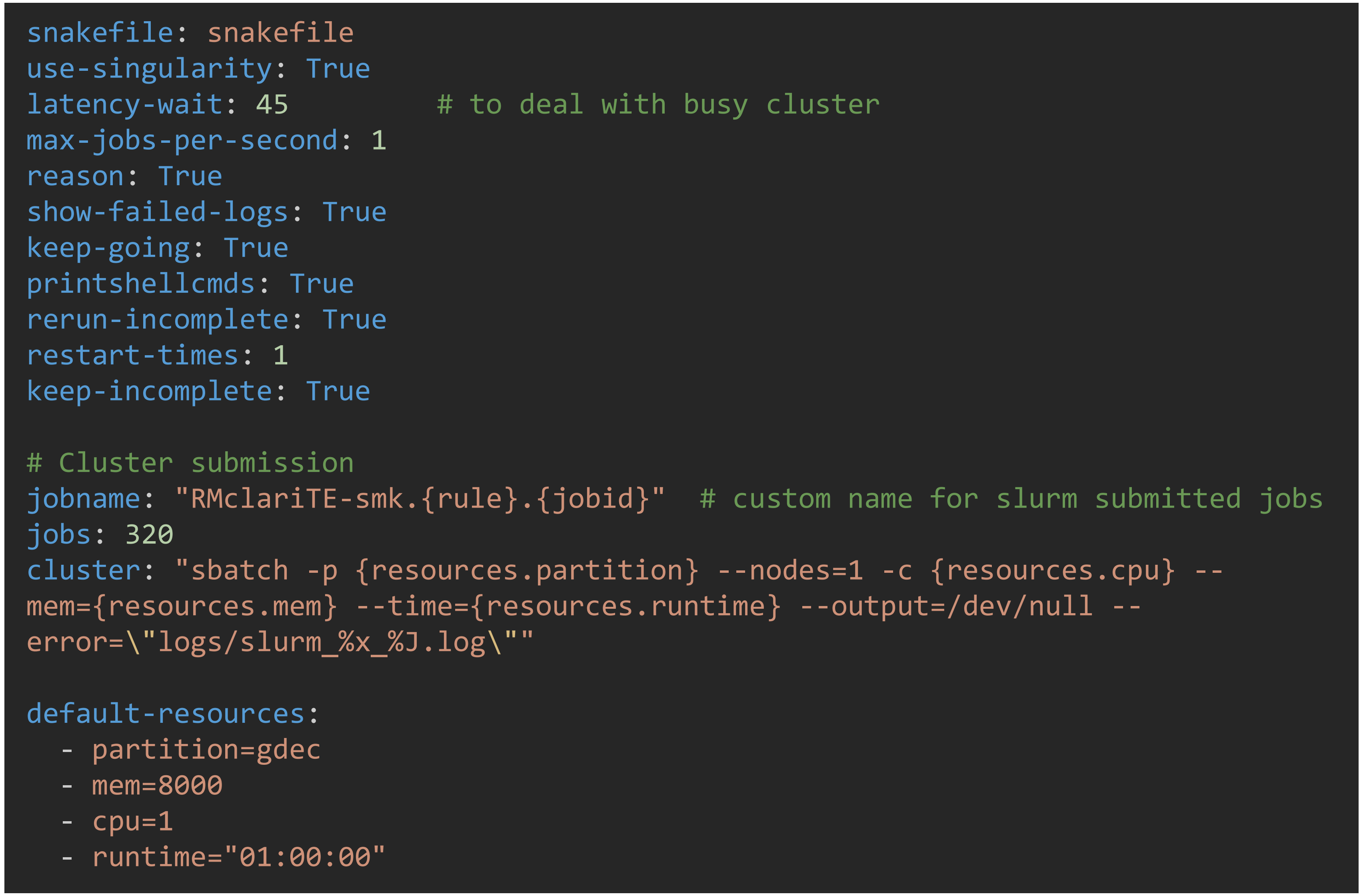
- ❑ **Cluster_profile/config.yaml:**



```
snakefile: snakefile
use-singularity: True
latency-wait: 45         # to deal with busy cluster
max-jobs-per-second: 1
reason: True
show-failed-logs: True
keep-going: True
printshellcmds: True
rerun-incomplete: True
restart-times: 1
keep-incomplete: True

# Cluster submission
jobname: "RMclariTE-smk.{rule}.{jobid}"  # custom name for slurm submitted jobs
jobs: 320
cluster: "sbatch -p {resources.partition} --nodes=1 -c {resources.cpu} --
mem={resources.mem} --time={resources.runtime} --output=/dev/null --
error=\"logs/slurm_%x_%J.log\""

default-resources:
  - partition=gdec
  - mem=8000
  - cpu=1
  - runtime="01:00:00"
```

To construct the pipeline

- ❑ **Cluster_profile/config.yaml:**



```yaml
snakefile: snakefile
use-singularity: True
latency-wait: 45         # to deal with busy cluster
max-jobs-per-second: 1
reason: True
show-failed-logs: True
keep-going: True
printshellcmds: True
rerun-incomplete: True
restart-times: 1
keep-incomplete: True

# Cluster submission
jobname: "RMclariTE-smk.{rule}.{jobid}"  # custom name for slurm submitted jobs
jobs: 320
cluster: "sbatch -p {resources.partition} --nodes=1 -c {resources.cpu} --
mem={resources.mem} --time={resources.runtime} --output=/dev/null --
error=\"logs/slurm_%x_%J.log\""

default-resources:
  - partition=gdec
  - mem=8000
  - cpu=1
  - runtime="01:00:00"
```

The snakefile

singularity and **ressources**

```
rule repeatmasker:
    output:
        XM = "results/chrom/{chrom}/{i}.fa.out.xm",
        out = "results/chrom/{chrom}/{i}.fa.out"
    input:
        fasta = "results/chrom/{chrom}/{i}.fa"
    singularity:
        "clari-te_smk_latest.sif"
    resources:
        runtime="12:00:00",
        mem=32000,
        cpu=16
    threads: 16
    params:
        config['clariTE_lib']
    shell:
        """
        cd results/chrom/{wildcards.chrom}
        RepeatMasker -e crossmatch -lib ../../../{params} -xsmall -nolow -xm -pa {threads} -q $(basename 
{input.fasta})
        cd ../../../
        rm {input.fasta}.cat {input.fasta}.masked {input.fasta}.ori.out {input.fasta}.tbl {input.fasta}.log
        rm RM_*/{wildcards.i}.fa*
        """
```

To launch the pipeline

❑ **Installation**

Download the pipeline:

```
git clone https://forgemia.inra.fr/umr-gdec/clari-te_smk.git
```

Download the Singularity image:

```
singularity pull oras://registry.forgemia.inra.fr/umr-gdec/clari-te_smk:latest
```

❑ **Setting up:** user configfile parameters

❑ **Pipeline launch:**

```
module load gcc/8.1.0 python/3.7.1 snakemake/7.15.1
snakemake --singularity-args '--bind /home/user/data' --profile cluster_profile/
```

❑ **Snakemake options**

```
--rerun-trigger input
--allowed-rules check_gff3,merge_chrom_gff3
```

Thank you for your attention