THERAGEN BIO

Snakemake tutorial

Jonghwan Shin

2025. 10. 28

Content

THERAGEN BIO

Introduction

THERAGEN BIO

What is snakemake?

- **Workflow manager**: builds **reproducible, scalable** data analysis pipelines.
- **Language**: a **Python-based DSL** (Domain-Specific Language) with readable rule syntax.
- **Scaling**: runs on **local, server, cluster, and cloud—no workflow changes needed**.
- **Environment management**: declare required **envs/containers/conda**, auto-deployed anywhere.
- **Dependency/retries**: tracks **input → output dependencies using a DAG** and supports selective re-runs.
- **Supports Bioconda installation**: bioinformatics package channel.
- **Supports PyPI installation**: installable via pip (Python); shows supported Python versions.

THERAGEN BIO

Useful website

Workflow catalog
- https://snakemake.github.io/snakemake-workflow-catalog/
- https://github.com/snakemake-workflows

Plugin
- https://snakemake.github.io/snakemake-plugin-catalog/

Installation

THERAGEN BIO

Pre-requirements

Git

https://git-scm.com/install/

Docker

* Install Docker Desktop or Docker Engine (choose one).

- Docker desktop (for GUI OS. such as Windows, Mac, Linux Ubuntu GUI)

https://www.docker.com/products/docker-desktop/

- Docker engine (for CLI OS. such as Linux OS)

https://docs.docker.com/engine/install/

THERAGEN BIO

Use Conda

```
# Download
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

# Run the command below and press Enter when prompted
# Type "yes" when asked to set up conda init
/bin/bash Miniconda3-latest-Linux-x86_64.sh

# Activate Conda
source ~/.bashrc

# Create a snakemake environment for the snakemake tutorial
conda create -n snakemake_9.11.1 -c bioconda -c conda-forge snakemake=9.11.1

# conda activate
conda activate snakemake_9.11.1

# Install tutorial dependencis
conda install -c bioconda -c conda-forge bwa
conda install -c bioconda samtools
```

THERAGEN BIO

Use Docker

```
# Download
The --privileged option is required for Singularity.
docker run -it -v [local_volume]:[docker_volume] --privileged shinejh0528/snakemake:9.11.1_for_tutorial /bin/bash
```

* For more information, please visit my docker hub
https://hub.docker.com/r/shinejh0528/snakemake

THERAGEN BIO

Tutorial repository download

```
# Repository download
git clone https://github.com/Shin-jongwhan/snakemake_tutorial.git
```

* For more information, please visit my git repository
https://github.com/Shin-jongwhan/snakemake_tutorial

Preparation

THERAGEN BIO

Genome FASTA download

Genome download (hg38)

UCSC - download link

In this tutorial, we use this FASTA file.

Name	Last modified	Size	Descript
Parent Directory		-	
analysisSet/	2023-01-06 17:06	-	
est.fa.gz	2019-10-14 13:54	1.5G	
est.fa.gz.md5	2019-10-14 13:54	44	
genes/	2024-12-23 12:50	-	
hg38.2bit	2015-04-30 16:16	797M	
hg38.agp.gz	2014-01-15 20:55	842K	
hg38.chrom.sizes	2013-12-24 21:06	11K	
hg38.chromAlias.bb	2024-04-08 20:13	243K	
hg38.chromAlias.txt	2021-10-06 13:44	27K	
hg38.chromFa.tar.gz	2014-01-23 17:18	938M	
hg38.chromFaMasked.tar.gz	2014-01-23 17:10	487M	
hg38.fa.align.gz	2014-01-08 23:43	2.4G	
hg38.fa.gz	2014-01-15 21:14	938M	
hg38.fa.masked.gz	2014-01-15 21:24	487M	
hg38.fa.out.gz	2014-01-15 20:56	172M	

THERAGEN BIO

Genome FASTA download

Genome download (hg38)

Since it is organized as chr1, chr2, … chrX, chrY, and chrM, it is easier to configure an analysis pipeline.

```
$ cat hg38.fa  | grep ">"
>chr1
>chr10
>chr11
>chr11_KI270721v1_random
>chr12
>chr13
>chr14
>chr14_GL000009v2_random
>chr14_GL000225v1_random
>chr14_KI270722v1_random
>chr14_GL000194v1_random
>chr14_KI270723v1_random
>chr14_KI270724v1_random
>chr14_KI270725v1_random
>chr14_KI270726v1_random
>chr15
>chr15_KI270727v1_random
>chr16
>chr16_KI270728v1_random
```

Genome FASTA download

Leave only chr1 sequence to test and bwa indexing

```
# Using seqtk
echo "chr1" > list.txt
seqtk subseq hg38.fa list.txt | seqtk seq -l 50 > hg38_chr1.fa

# fasta index
bwa index hg38_chr1.fa
```

```
drwxr-xr-x 2 jhshin bioinfo       4096 Sep 26 13:55 ./
drwxr-xr-x 4 jhshin bioinfo       4096 Sep 16 16:43 ../
-rw-r--r-- 1 jhshin bioinfo  253935557 Sep 17 15:51 hg38_chr1.fa
-rw-r--r-- 1 jhshin bioinfo       2527 Sep 26 13:23 hg38_chr1.fa.amb
-rw-r--r-- 1 jhshin bioinfo         45 Sep 26 13:23 hg38_chr1.fa.ann
-rw-r--r-- 1 jhshin bioinfo  248956524 Sep 26 13:23 hg38_chr1.fa.bwt
-rw-r--r-- 1 jhshin bioinfo   62239107 Sep 26 13:23 hg38_chr1.fa.pac
-rw-r--r-- 1 jhshin bioinfo  124478264 Sep 26 13:25 hg38_chr1.fa.sa
```

Genome fasta (chr1 only)

Genome fasta (chr1 only) - bwa index files

THERAGEN BIO

Download Human example data (quick start)

hg38_chr1.fa
https://drive.google.com/file/d/1-oIZNcatSH9gT5iHFhxm0Yy4fP34CnC7/view?usp=sharing

hg38_chr.fa.idx.tar.gz (in the same directory as hg38_chr1.fa)
https://drive.google.com/file/d/1quRANcCjdxHGoKAfsPB3pkZkrOlozrw_/view?usp=sharing
* **Use this index data after decompressing with the command**
tar -xzf hg38_chr1.fa.idx.tar.gz

test fastq 0.25 M reads
R1 : https://drive.google.com/file/d/1AqONWh9GJZKQvs0ls6ltny4fLqapv0eE/view?usp=sharing
R2 : https://drive.google.com/file/d/1J4P5lZA8RpRK8LGATbquv52HUGQLHFf6/view?usp=sharing

THERAGEN BIO

Basic tutorial

THERAGEN BIO

1. Create single rule - mapping

file name : 'Snakefile' (default)

```
rule bwa_map:
    input:
        genome_fa = "/biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa",
        fastq_1 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/test_1.fastq.gz",
        fastq_2 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/test_2.fastq.gz"
    output:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.bam"
    params:
        thread = 8
    shell:
        """
        bwa mem -t {params.thread} {input.genome_fa} {input.fastq_1} {input.fastq_2} \
        | samtools view -Sb - \
        > {output.bam}
        """
```

THERAGEN BIO

1. Create single rule - mapping

file name : 'Snakefile' (default)
You can execute snakemake in Snakefile folder by default
Or you can specify Snakefile path by -s option

Option
-n : dry-run
-p : print command
--jobs [int] : number of parallel jobs (or use --cores)
--cores [int] : number of threads (or use --jobs)
-s [Snakefile_path] : specify Snakefile path
Last arguments [path] : Specify the output path as an absolute path. This ensures that Snakemake constructs the dependency graph (DAG) correctly and executes all preceding rules in order to generate the specified output.

```
# Activate conda
conda activate snakemake_9.11.1
# Or using docker
docker run -it -v [local_volume]:[docker_volume] shinejh0528/snakemake:9.11.1 /bin/bash

# Run pipeline In Snakefile path
snakemake --jobs 10 -np /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.bam
# Or specify Snakefile path
snakemake --jobs 10 \
   -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/1_mapping/Snakefile -p \
   /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.bam
```

THERAGEN BIO

1. Create single rule - mapping

Running Pipeline

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ snakemake -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/1_mapping/Snakefile -np /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
host: husky
Building DAG of jobs...
Job stats:
job        count
-------  -------
bwa_map        1
total          1


[Wed Sep 17 16:28:06 2025]
rule bwa_map:
    input: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_1.fastq.gz, /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_2.fastq.gz
    output: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
    jobid: 0
    reason: Missing output files: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
    resources: tmpdir=<TBD>
Shell command:
              bwa mem /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_1.fastq.gz /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_2.fastq.gz             | samtools view -Sb -           > /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam

Job stats:
job        count
-------  -------
bwa_map        1
total          1

Reasons:
    (check individual jobs above for details)
    output files have to be generated:
        bwa_map
This was a dry-run (flag -n). The order of jobs does not reflect the order of execution.
```

Dry-run

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ snakemake --jobs 10 -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/1_mapping/Snakefile -p /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
Assuming unrestricted shared filesystem usage.
host: husky
Building DAG of jobs...
Using shell: /bin/bash
Provided cores: 10
Rules claiming more threads will be scaled down.
Job stats:
job        count
-------  -------
bwa_map        1
total          1

Select jobs to execute...
Execute 1 jobs...

[Fri Sep 26 13:40:27 2025]
localrule bwa_map:
    input: /biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa, /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_1.fastq.gz, /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_2.fastq.gz
    output: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
    jobid: 0
    reason: Missing output files: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
    resources: tmpdir=/tmp
Shell command:
                bwa mem -t 8 /biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_1.fastq.gz /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_2.fastq.gz           | samtools view -Sb -           > /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
[Fri Sep 26 13:41:24 2025]
Finished jobid: 0 (Rule: bwa_map)
1 of 1 steps (100%) done
Complete log(s): /biarchive/project/jhshin/test/snakemake_totorial/test_data/.snakemake/log/2025-09-26T134027.015432.snakemake.log
```

Real run

1. Create single rule - mapping

Output

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ ll
total 74265
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 26 14:02 ./
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 17 16:16 ../
drwxr-xr-x 12 jhshin bioinfo     4096 Sep 26 14:02 .snakemake/
-rw-r--r--  1 jhshin bioinfo 15413564 Sep 17 16:07 test_1.fastq.gz
-rw-r--r--  1 jhshin bioinfo 15130109 Sep 17 16:07 test_2.fastq.gz
-rw-r--r--  1 jhshin bioinfo 45489622 Sep 26 14:03 test.bam
```

Output file

snakemake metadata

2. Mapping with wildcard

Snakefile

The **{sample}** wildcard can be used in other arguments.

```
rule bwa_map:
    input:
        genome_fa = "/biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa",
        fastq_1 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}_1.fastq.gz",
        fastq_2 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}_2.fastq.gz"
    output:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.bam"
    params:
        thread = 8
    shell:
        """
        bwa mem -t {params.thread} {input.genome_fa} {input.fastq_1} {input.fastq_2} \
        | samtools view -Sb - \
        > {output.bam}
        """
```

2. Mapping with wildcard

Running Pipeline

```
snakemake --jobs 10 \
   -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/2_mapping_with_wildcard/Snakefile \
   -p \
   /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.bam
```

The output is same as '1. Create single rule - mapping'

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ ll
total 74265
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 26 14:02 ./
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 17 16:16 ../
drwxr-xr-x 12 jhshin bioinfo     4096 Sep 26 14:02 .snakemake/
-rw-r--r--  1 jhshin bioinfo 15413564 Sep 17 16:07 test_1.fastq.gz
-rw-r--r--  1 jhshin bioinfo 15130109 Sep 17 16:07 test_2.fastq.gz
-rw-r--r--  1 jhshin bioinfo 45489622 Sep 26 14:03 test.bam
```

THERAGEN BIO

3. Sorting read alignments

Snakemakefile

```
rule samtools_sort:
    input:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.bam"
    output:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.sorted.bam"
    params:
        thread = 8
    shell:
        "samtools sort -@ {params.thread} -T {wildcards.sample} "
        "-O bam {input.bam} > {output.bam}"
```

Shell cmd Options

- T [str]

When sorting, Samtools splits the input BAM into **temporary files and then merges** them into the final output BAM. The –T option specifies the prefix for these temporary files.

Example: -T test → creates temporary files like test.0000.bam, test.0001.bam during sorting.

-@ [int]

specify thread

THERAGEN BIO

3. Sorting read alignments

Running Pipeline

```
snakemake --jobs 10 \
   -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/3_sorting_read_alignments/Snakefile \
   -p \
   /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.sorted.bam
```

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ snakemake --jobs 10 -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/3_sorting_read_alignments/Snakefile
 -p /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.sorted.bam
Assuming unrestricted shared filesystem usage.
host: husky
Building DAG of jobs...
Using shell: /bin/bash
Provided cores: 10
Rules claiming more threads will be scaled down.
Job stats:
job              count
-------------  -------
samtools_sort        1
total                1

Select jobs to execute...
Execute 1 jobs...

[Fri Sep 26 14:45:10 2025]
localrule samtools_sort:
    input: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
    output: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.sorted.bam
    jobid: 0
    reason: Missing output files: /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.sorted.bam
    wildcards: sample=test
    resources: tmpdir=/tmp
Shell command: samtools sort -@ 8 -T test -O bam /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.bam
 > /biarchive/project/jhshin/test/snakemake_totorial/test_data/test.sorted.bam
[Fri Sep 26 14:45:12 2025]
Finished jobid: 0 (Rule: samtools_sort)
1 of 1 steps (100%) done
Complete log(s): /biarchive/project/jhshin/test/snakemake_totorial/test_data/.snakemake/log/2025-09-26T144510.465119.
snakemake.log
```

THERAGEN BIO

3. Sorting read alignments

Output

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ ll
total 114499
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 26 14:32 ./
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 17 16:16 ../
drwxr-xr-x 12 jhshin bioinfo     4096 Sep 26 14:20 .snakemake/
-rw-r--r--  1 jhshin bioinfo 15413564 Sep 17 16:07 test_1.fastq.gz
-rw-r--r--  1 jhshin bioinfo 15130109 Sep 17 16:07 test_2.fastq.gz
-rw-r--r--  1 jhshin bioinfo 45489622 Sep 26 14:21 test.bam
-rw-r--r--  1 jhshin bioinfo 41198673 Sep 26 14:32 test.sorted.bam
```

THERAGEN BIO

4. Indexing bam

Snakefile

```
rule samtools_index:
    input:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.sorted.bam"
    output:
        "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.sorted.bam.bai"
    shell:
        "samtools index {input.bam}"
```

Running pipeline

```
snakemake --jobs 10 \
    -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/4_indexing_bam/Snakefile \
    -p \
    /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.sorted.bam.bai
```

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ ll
total 114850
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 26 14:54 ./
drwxr-xr-x  3 jhshin bioinfo     4096 Sep 17 16:16 ../
drwxr-xr-x 12 jhshin bioinfo     4096 Sep 26 14:44 .snakemake/
-rw-r--r--  1 jhshin bioinfo 15413564 Sep 17 16:07 test_1.fastq.gz
-rw-r--r--  1 jhshin bioinfo 15130109 Sep 17 16:07 test_2.fastq.gz
-rw-r--r--  1 jhshin bioinfo 45489622 Sep 26 14:21 test.bam
-rw-r--r--  1 jhshin bioinfo 41198673 Sep 26 14:45 test.sorted.bam
-rw-r--r--  1 jhshin bioinfo   358968 Sep 26 14:50 test.sorted.bam.bai
```

5. Let’s combine everything

Snakefile

You can use Python syntax !

```
sAnalysis_dir = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/"

rule bwa_map:
    input:
        fastq_1 = sAnalysis_dir + "{sample}_1.fastq.gz",
        fastq_2 = sAnalysis_dir + "{sample}_2.fastq.gz"
    output:
        bam = sAnalysis_dir + "result/mapping/{sample}/{sample}.bam"
    params:
        genome_fa = "/biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa",
        thread = 8
    shell:
        """
        bwa mem -t {params.thread} {params.genome_fa} {input.fastq_1} {input.fastq_2} \
        | samtools view -Sb - \
        > {output.bam}
        """
```

page 1

```
rule samtools_sort:
    input:
        bam = sAnalysis_dir + "result/mapping/{sample}/{sample}.bam"
    output:
        bam = sAnalysis_dir + "result/mapping/{sample}/{sample}.sorted.bam"
    params:
        thread = 8
    shell:
        "samtools sort -@ {params.thread} -T {wildcards.sample} "
        "-O bam {input.bam} > {output.bam}"


rule samtools_index:
    input:
        bam = sAnalysis_dir + "result/mapping/{sample}/{sample}.sorted.bam"
    output:
        bam_bai = sAnalysis_dir + "result/mapping/{sample}/{sample}.sorted.bam.bai"
    shell:
        "samtools index {input.bam}"
```

page 2

THERAGEN BIO

5. Let’s combine everything

Running pipeline

```
snakemake --jobs 10 \
   -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/5_combine_1-4_rules/Snakefile \
   -p \
   /biarchive/project/jhshin/test/snakemake_tutorial/test_data/result/mapping/test/test.sorted.bam.bai
```

Output

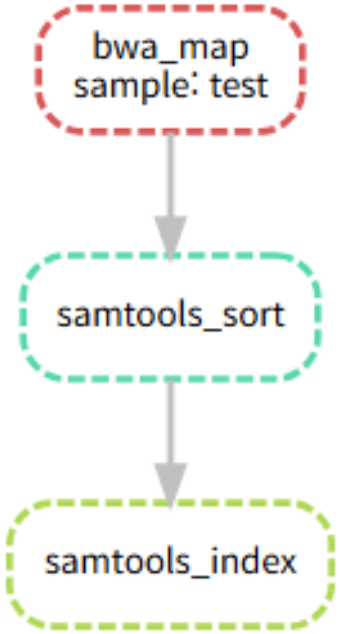
```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_tutorial/test_data
$ tree -h result/
result/
└── [4.0K]  mapping
    └── [4.0K]  test
        ├── [ 43M]  test.bam
        ├── [ 39M]  test.sorted.bam
        └── [ 351K]  test.sorted.bam.bai

2 directories, 3 files
```

THERAGEN BIO

6. Generate DAG graph

Command

```
snakemake -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/5_combine_1-4_rules/Snakefile \
   /biarchive/project/jhshin/test/snakemake_tutorial/test_data/result/mapping/test/test.sorted.bam.bai \
   --dag | dot -Tsvg > dag.svg
```



THERAGEN BIO

6. Generate DAG graph

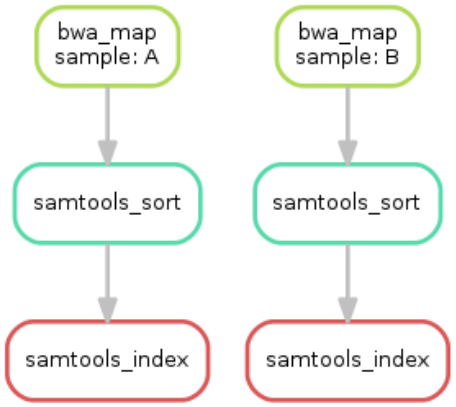
You can generate a DAG graph for each sample using {sample_1,sample_2,...}.

```
$ snakemake sorted_reads/{A,B}.bam.bai --dag | dot -Tsvg > dag.svg
```

Note

If you went with: Run tutorial for free in the cloud via Gitpod, you can easily view the resulting `dag.svg` by right-clicking on the file in the explorer panel on the left and selecting `Open With -> Preview`.

we create a **visualization of the DAG** using the `dot` command provided by Graphviz↗. For the given target files, Snakemake specifies the DAG in the dot language and pipes it into the `dot` command, which renders the definition into SVG format↗. The rendered DAG is piped into the file `dag.svg` and will look similar to this:



Additional rule option

THERAGEN BIO

Additional rule option

In this tutorial, we see

1. How to record benchmark

2. How to record log

3. How to use container

THERAGEN BIO

Additional rule option

The Snakefile from "**Basic tutorial – 2. Mapping with wildcard**" section, with additional benchmark, log, and container directives.

```
rule bwa_map:
    input:
        genome_fa = "/biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa",
        fastq_1 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}_1.fastq.gz",
        fastq_2 = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}_2.fastq.gz"
    output:
        bam = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/{sample}.bam"
    params:
        thread = 8
    log:
        stdout = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/log/bwa_map/{sample}.stdout.txt",
        stderr = "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/log/bwa_map/{sample}.stderr.txt"
    benchmark:
        "/biarchive/project/jhshin/test/snakemake_tutorial/test_data/benchmark/bwa_map/{sample}.txt"
    container:
        "docker://shinejh0528/bwa-mem2:1.2.0-samtools"
    shell:
        """
        bwa mem -t {params.thread} {input.genome_fa} {input.fastq_1} {input.fastq_2} 2> {log.stderr} \
        | samtools view -Sb - \
        > {output.bam} \
        2> {log.stderr}
        """
```

THERAGEN BIO

Additional rule option

Running pipeline

To run Snakemake with a container, use the options --use-singularity and --singularity-args.

* --singularity-args "--bind /biarchive": **specifies a mount point** in Singularity so that the /biarchive path is accessible inside the container.

```
snakemake --jobs 10 \
    --use-singularity \
    --singularity-args "--bind /biarchive" \
    -s /TBI/People/tbi/jhshin/pipeline/snakemake_tutorial/basic/6_mapping_with_wildcard_with_other_opt/Snakefile \
    -p \
    /biarchive/project/jhshin/test/snakemake_tutorial/test_data/test.bam
```

THERAGEN BIO

Additional rule option

Running pipeline

1. First, pull the container image from Docker Hub as a Singularity image.
2. Then, the rules run inside the container image.

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data
$ snakemake --jobs 10 --use-singularity --singularity-args "--bind /biarchive" -s /TBI/People/tbi/jhshin/pipeline/sna
kemake_tutorial/basic/6_mapping_with_wildcard_with_other_opt/Snakefile /biarchive/project/jhshin/test/snakemake_totor
ial/test_data/test.bam -p
Assuming unrestricted shared filesystem usage.
host: husky
Building DAG of jobs...
Pulling singularity image docker://shinejh0528/bwa-mem2:1.2.0-samtools.
Using shell: /bin/bash
Provided cores: 10
Rules claiming more threads will be scaled down.
Job stats:
job         count
-------  -------
bwa_map         1
total           1

Select jobs to execute...
Execute 1 jobs...
```

Additional rule option

Log

You can view the log files of each rule and sample.

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data/log/bwa_map
$ cat test.stderr.txt
[M::bwa_idx_load_from_disk] read 0 ALT contigs
[M::process] read 500000 sequences (75500000 bp)...
[M::mem_pestat] # candidate unique pairs for (FF, FR, RF, RR): (1, 18746, 16, 2)
[M::mem_pestat] skip orientation FF as there are not enough pairs
[M::mem_pestat] analyzing insert size distribution for orientation FR...
[M::mem_pestat] (25, 50, 75) percentile: (318, 367, 418)
[M::mem_pestat] low and high boundaries for computing mean and std.dev: (118, 618)
[M::mem_pestat] mean and std.dev: (370.05, 76.57)
[M::mem_pestat] low and high boundaries for proper pairs: (18, 718)
[M::mem_pestat] analyzing insert size distribution for orientation RF...
[M::mem_pestat] (25, 50, 75) percentile: (1153, 1920, 9747)
[M::mem_pestat] low and high boundaries for computing mean and std.dev: (1, 26935)
[M::mem_pestat] mean and std.dev: (4812.56, 4316.80)
[M::mem_pestat] low and high boundaries for proper pairs: (1, 35529)
[M::mem_pestat] skip orientation RR as there are not enough pairs
[M::mem_pestat] skip orientation RF
[M::mem_process_seqs] Processed 500000 reads in 391.234 CPU sec, 49.555 real sec
[main] Version: 0.7.18-r1243-dirty
[main] CMD: bwa mem -t 8 /biarchive/project/jhshin/reference/hg38/ucsc/hg38_chr1.fa /biarchive/project/jhshin/test/sn
akemake_totorial/test_data/test_1.fastq.gz /biarchive/project/jhshin/test/snakemake_totorial/test_data/test_2.fastq.g
z
[main] Real time: 53.727 sec; CPU: 392.737 sec
```

THERAGEN BIO

Additional rule option

Benchmark

In the benchmark data, you can check CPU, memory, and disk usage.

```
(snakemake_9.11.1) jhshin@husky:/biarchive/project/jhshin/test/snakemake_totorial/test_data/benchmark/bwa_map
$ cat test.txt
s       h:m:s   max_rss max_vms max_uss max_pss io_in   io_out  mean_load       cpu_time
54.0623 0:00:54 1852.50 2355.79 1847.91 1849.98 74.80   0.00    641.76  347.16
```

Introducing my template

THERAGEN BIO

Template

Template URL

https://github.com/Shin-jongwhan/snakemake_template/tree/main/impertoire/1.0.0

Features

1. Define configuration (config).
2. Organize Docker containers.
3. List samples in samples.tsv.
4. In workflow/rules/common.smk, define configuration, outputs, and options, then import each sub_rule.smk.

Notes

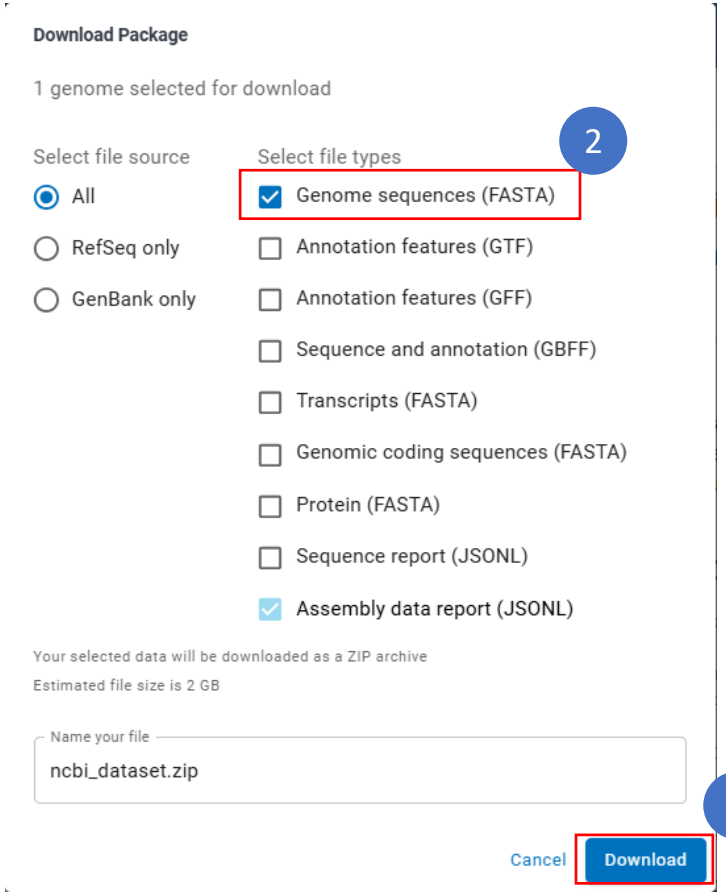
THERAGEN BIO

Genome FASTA download

Genome download (hg38)

NCBI - download link

GCF_000001405.26 is GRCh38.p13 (Genome Reference Consortium Human Build 38, patch 13), maintained by NCBI RefSeq.



Genome FASTA download

Genome download (hg38)

Since it is provided as an assembly, it may be difficult to set up an analysis pipeline.

```
$ cat GCF_000001405.26_GRCh38_genomic.fna  | grep ">"
>NC_000001.11 Homo sapiens chromosome 1, GRCh38 Primary Assembly
>NT_187361.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG1_UNLOCALIZED
>NT_187362.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG2_UNLOCALIZED
>NT_187363.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG3_UNLOCALIZED
>NT_187364.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG4_UNLOCALIZED
>NT_187365.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG5_UNLOCALIZED
>NT_187366.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG6_UNLOCALIZED
>NT_187367.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG7_UNLOCALIZED
>NT_187368.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG8_UNLOCALIZED
>NT_187369.1 Homo sapiens chromosome 1 unlocalized genomic scaffold, GRCh38 Primary Assembly HSCHR1_CTG9_UNLOCALIZED
>NC_000002.12 Homo sapiens chromosome 2, GRCh38 Primary Assembly
```

THERAGEN BIO

Thank you ! :)

jonghwan.shin@theragenbio.com