Snakemake vs Nextflow

SEDNA Computational Roundtable Series
17 April 2025

Overview

- Nextflow and Snakemake are pipeline frameworks / workflow management tools
- Give ability to manage and execute multistep computational analyses
- Both contribute to reproducibility, portability, and scalability of analyses
- Both have public repositories of workflows
  - https://snakemake.github.io/snakemake-workflow-catalog/index.html
  - https://nf-co.re/pipelines/
- Create reproducible environments with containers or Conda

	Snakemake	Nextflow
Language	Python	Groovy (combo of Java, Python, and others)
Organization	● File based/ file dependent: order of pipeline is defined by input and output at each step ● DAG constructed by file dependencies ● Rules	● Input and Output Channels ● Modularization via channels and modules ● Outputs do not have to be actual files on system
Execution	● **Pull-based** (rules executed when output is needed). Dependency graph results in efficient resource usage	● **Push-based** (data / processes occur in pipeline as data becomes available). This is better for large-scale or parallel workflows but can create bottlenecks
Potential Pros	● Popular in academia ● Easier debugging ● Dry-run mode	● Popular in industry ● Dynamic ordering (not file dependent) ● Nextflow tower UI monitor jobs
Potential Cons	● Clunky framework (output drives input / task) ● Dependency graph can get very complicated with lots of looping / complex workflows	● Steep learning curve with Groovy ● Compex to troubleshoot - no dry-run mode and hidden log file ● Large temp files / temp work directories

Questions and Discussion points

- Why?
  - Use for scalability and reproducibility of projects
- When to use?
  - May be tool dependent - might not be worth it if complex to set up
  - Small tasks (trimming, QC) may be easier to just run slurm script
- Logistics
  - Can both resume when failed?
  - Do both take up similar amounts of space?
  - How can I control resource usage?
  - Are both efficient with resources and space?
- What types of features are most important in a pipeline / workflow software?

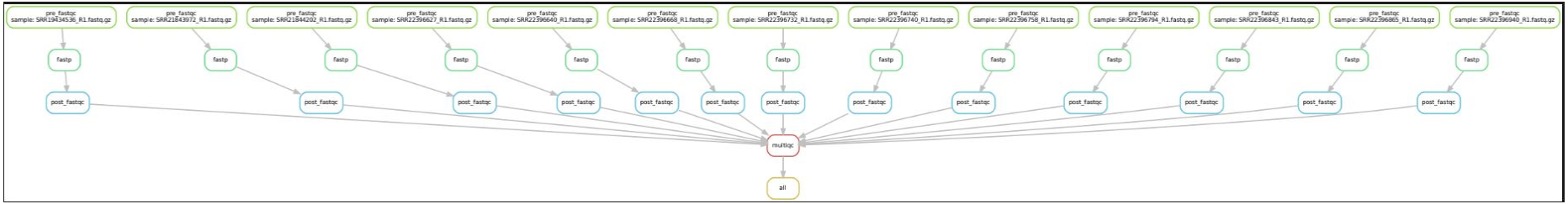
Working without a workflow manager

- Simple trim and quality control of sequencing data
  - Write slurm script for each step (or one big slurm script)
  - Wait for job to run before submitting job for next step
  - Manually check output and error files
  - Alternatively, run interactively and have no log of errors, output, nodes, etc
- Workflow managers track execution of each job and manages output
  - Extremely cumbersome for large workflows

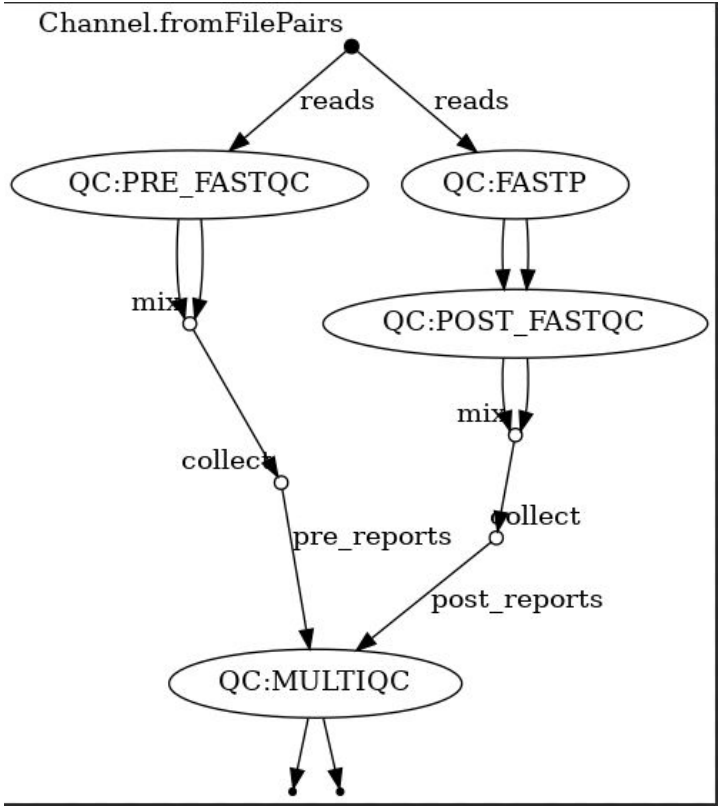
Workflow managers facilitate running large pipelines

- We have prepared two mirrored workflows for read trimming and QC and provided them on GitHub
  - SEDNA compatible
  - Executable in the queue and in interactive session
  - https://github.com/ccpowers-NOAA/PEMAD-PBB-fastp

Snakemake workflow diagram



Nextflow workflow diagram



basic script syntax

Nextflow

```
// PARAMETERS
params.input = “hello world”
// PROCESSES
process STEP1 {
        inputs:
        outputs:
        script:
}
process STEP2 {
        // inputs, outputs, script
}
// WORKFLOW
workflow {
        STEP1(params.input)
        STEP2(OUTPUT_FROM_STEP1)
}
```

Snakemake

```
rule all:
    input:
        "OUTPUT_FROM_STEP2"
rule STEP1:
    input:
        "hello world"
    output:
        "OUTPUT_FROM_STEP1"
    shell:
        script"
rule STEP2:
    input:
        "OUTPUT_FROM_STEP1"
    output:
        "OUTPUT_FROM_STEP2"
    shell:
        "script"
```