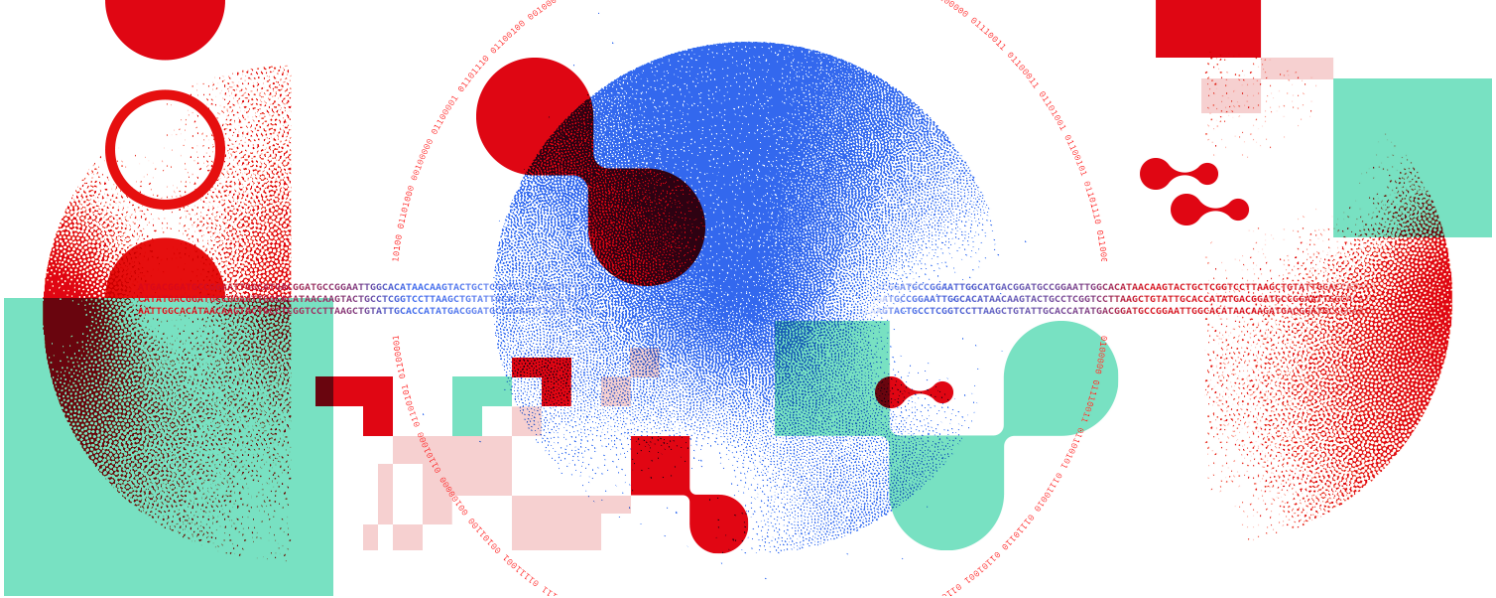

Snakemake for reproducible research

Additional advanced concepts

Centre hospitalier universitaire vaudois

Antonin Thiébaut
antonin.thiebaut@chuv.ch

SIB Swiss Institute of Bioinformatics

Running Snakemake non-locally

- Snakemake can interact with schedulers to run on clusters and cloud:
  - AWS
  - Azure
  - Flux
  - Google Batch
  - HTCondor
  - Kubernetes
  - LSF
  - Slurm

- Require almost no changes (runtime, memory…) to the rules
  - Scheduler command can take job information from rule definition
  - Resource managing is essential in a cluster/cloud environment
  - One key parameter: maximum number of jobs running in parallel: `-j / --jobs`

- Implemented with:
  - v7 and before: `--cluster “<scheduler_name>“` in the Snakemake command
  - v8+: install plugins then `--executor “<scheduler_name>“` in the Snakemake command

Working with remote inputs

- Snakemake can access remote files with many protocols:
  - AWS S3 (Amazon Simple Storage Service)
  - Azure (Microsoft Azure Blob Storage)
  - EGA (European Genome-phenome Archive), GenBank / NCBI Entrez
  - FTP (File transfer protocol), HTTP/S, SFTP (File transfer over SSH), locally mounted filesystem
  - GCS (Google Cloud Storage)
  - iRODS
  - Sharepoint (Microsoft Sharepoint)
  - Webdav
  - Zenodo

- Idea:
  - Install required plugins
  - Initiate remote provider in rule or set default provider with --default-storage-provider <provider_name>
  - Access remote files within a rule

- Files are downloaded in current working directory and deleted after job is completed

Execution profiles

- Execution profiles are presets of execution parameter values (-j <N>, --use-conda, --resources mem_mb=100…)
- Implemented as directory and stored in *~/.config/snakemake/<profile_name>/*
  - config.yaml with syntax <run_option>: <value>
- Profiles can be extended a lot, especially for HPC environments
  - Scripts to submit jobs
  - Scripts to check job status
  - ➢ Advanced customization
- Official list of Snakemake profiles here

Reminder on best practices

- One repository = one workflow
- Use Conda environments / Docker containers when possible
- Break out large workflow into modules with extension “.smk”
- Specify parameters in a config file located in a ‘config’ folder
- If you have many samples with information, use a sample sheet located in the ‘config’ folder
- Follow the official directory structure
- Use explicit rule and variable names
- Comment to explain your workflow; use docstring comments in rules