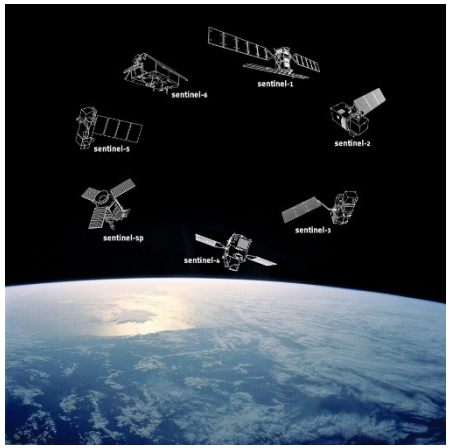
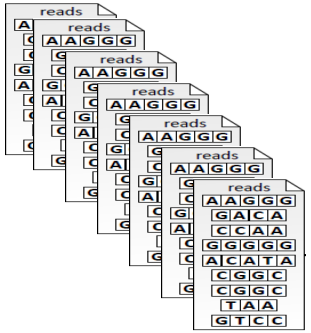
Workflowsprachen

Ulf Leser

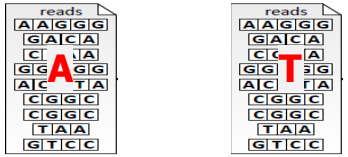
Big Scientific Data

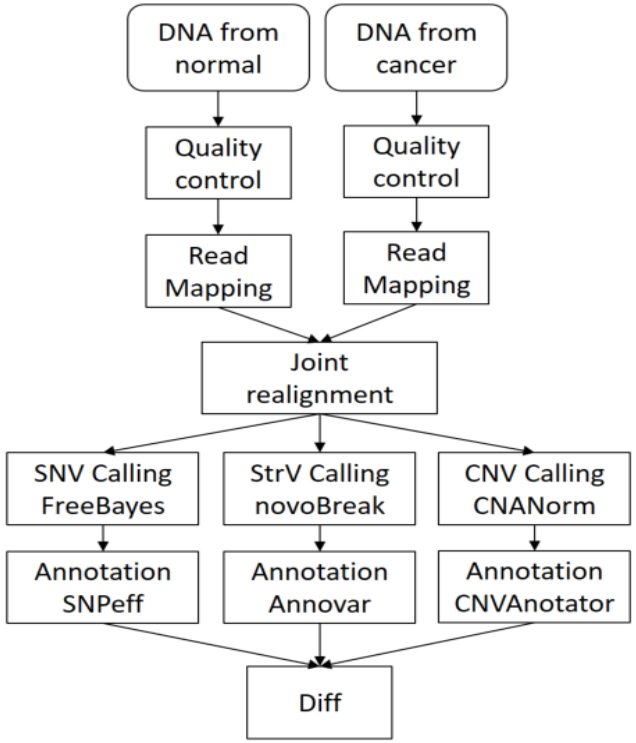


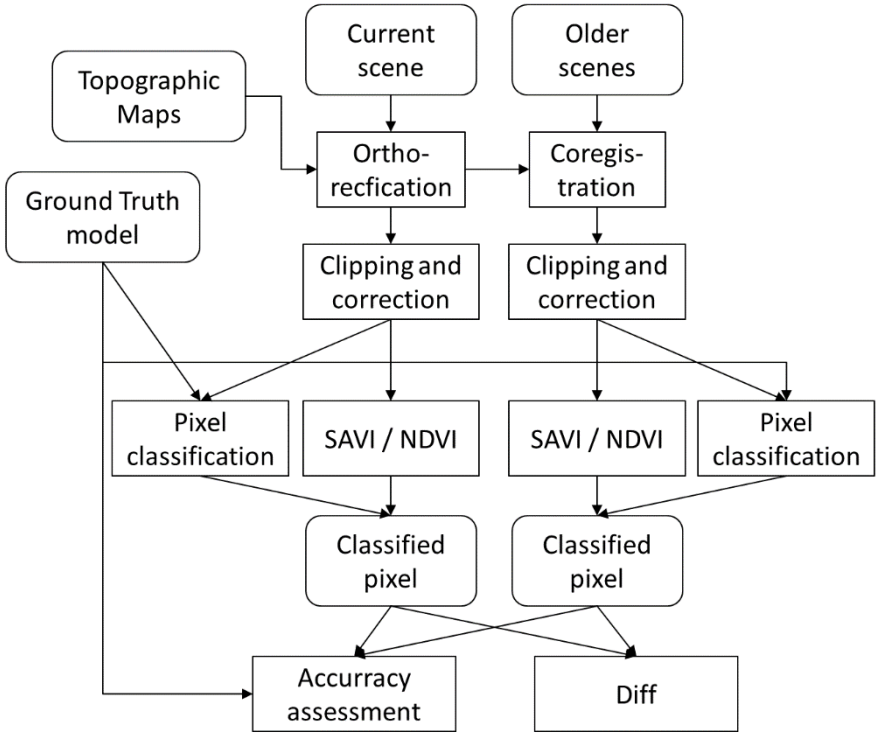


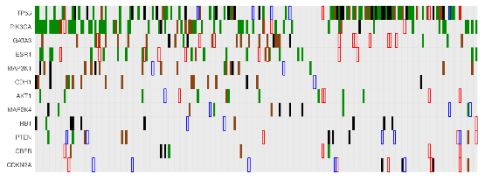
Note petabytes every day, but easily a few terabytes per week

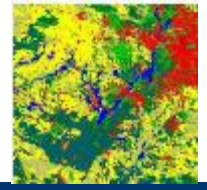
Data Analysis Workflows (DAWs)



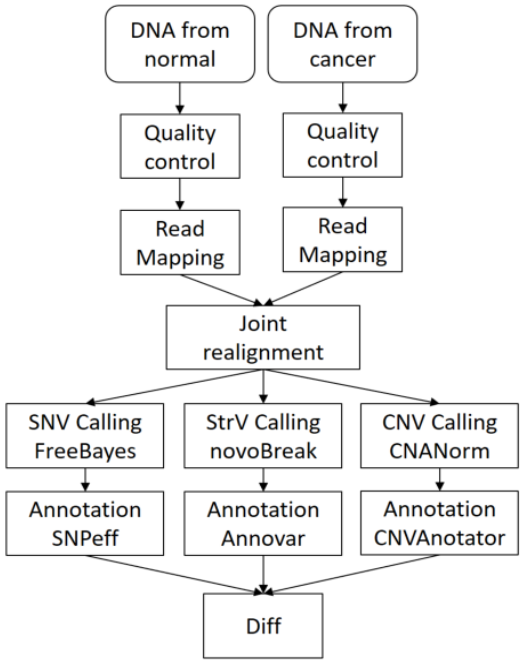








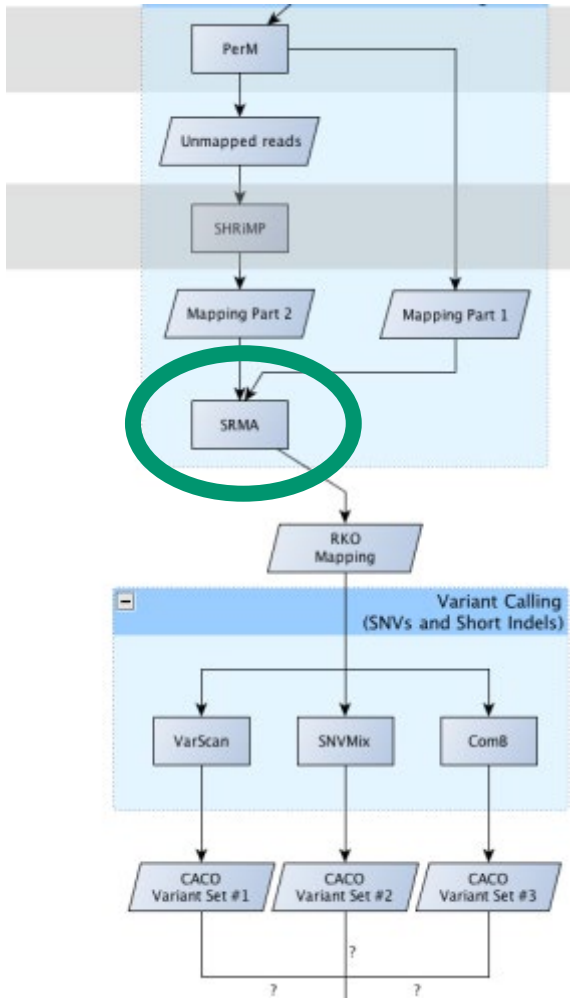
Distributed DAW Infrastructures



Workflows

Tasks

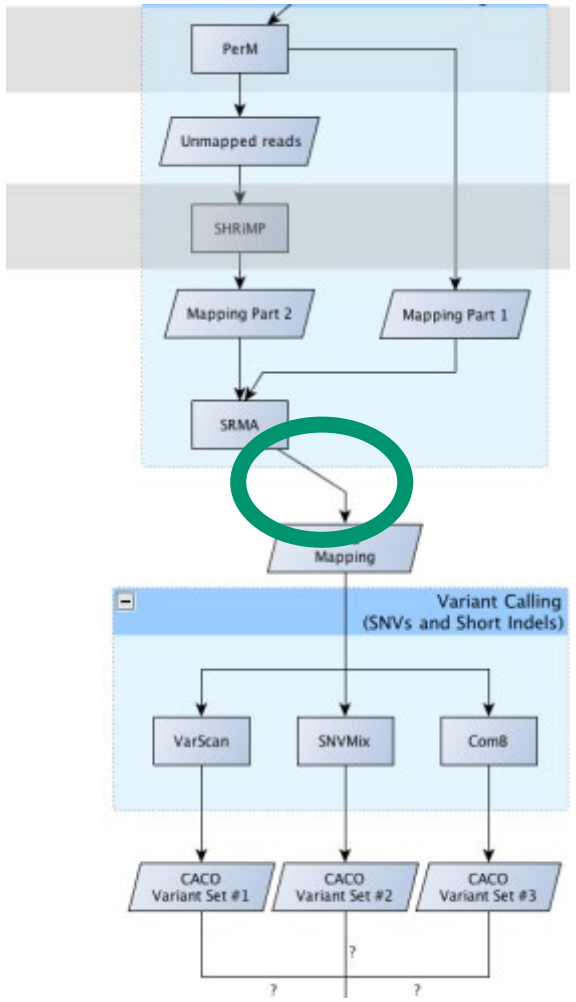
- Have (multiple) input and output "ports" (parameter)
- Must be executable
  - Or web services – deprecated in the Big Data area
- Black box model: Infrastructure has no notion on what a task does



Workflows: Tasks and Dependencies

Tasks

- Have (multiple) input and output "ports" (parameter)
- Must be executable
  - Or web services – deprecated in the Big Data area
- Black box: Infrastructure has no notion on what a task does

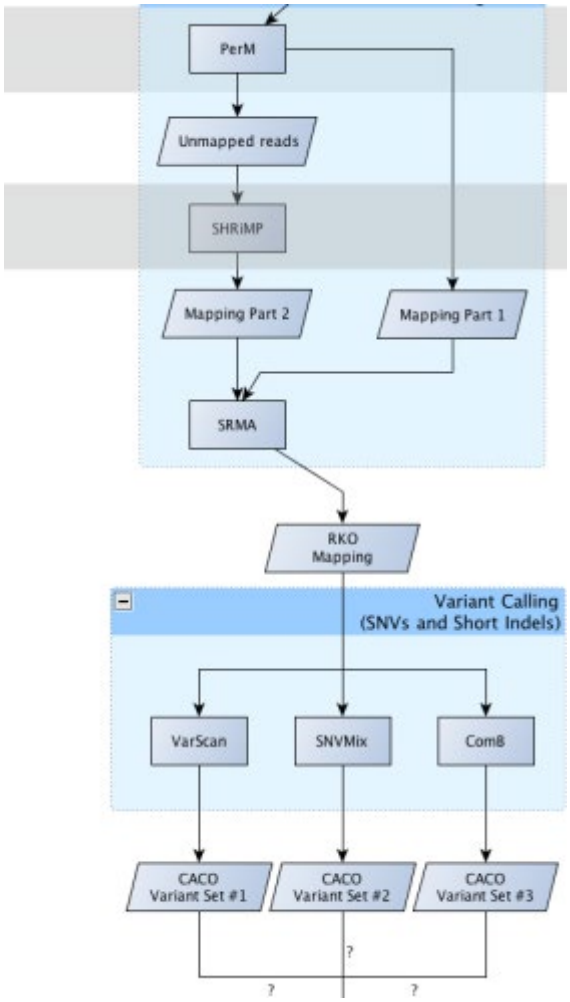


Dependencies

- Connect one input (upstream) with one output (downstream) port
- Implemented as files, pipes, in-memory, …
- Constrain the possible order of execution
- Black box model: Infrastructure has no notion on content (or format)

Dependency Graph

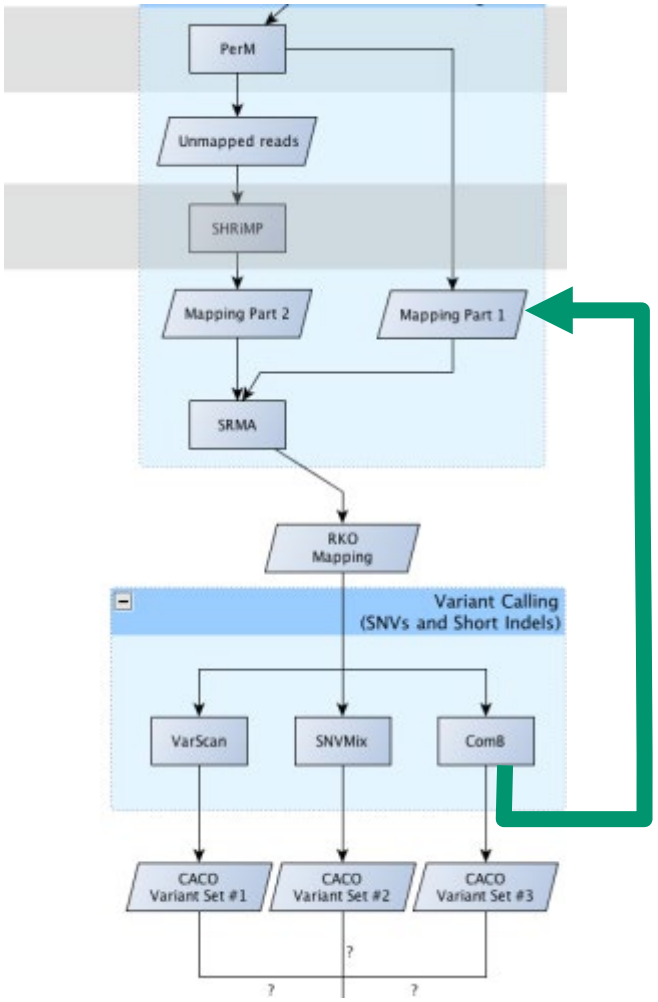
- Tasks become nodes (vertices)
- Dependencies become edges (arcs)
- Together a directed graph G = (T,D)



Dependency Graph

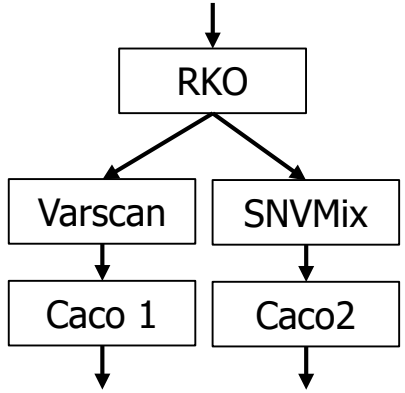
- Tasks become nodes (vertices)
- Dependencies become edges (arcs)
- Together a directed graph G = (T,D)

- Mostly a simple graph: No two arcs between the same pair of nodes
  - But: A task T1 may produce two files F1, F2 which both are input for task T2
- Mostly not a hyper-graph
  - But: Broadcasts can be modelled as n-ary arcs
- Mostly a DAG
  - But: Iteration / recursion introduces loops

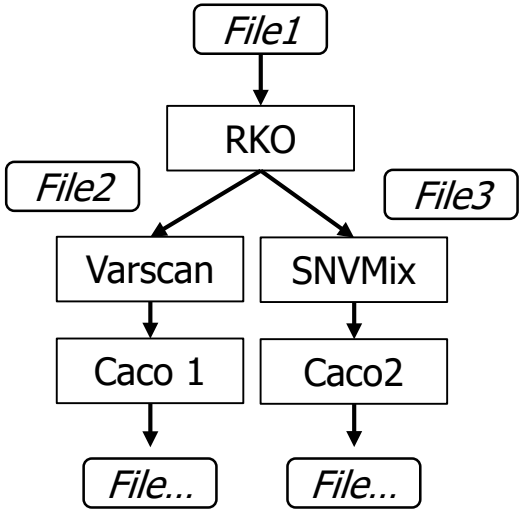


Abstract versus logical DAWs

- So far, our DAWs were abstract
  - Nodes have names, but there are no concrete files
  - Cannot be executed, but are easy to understand

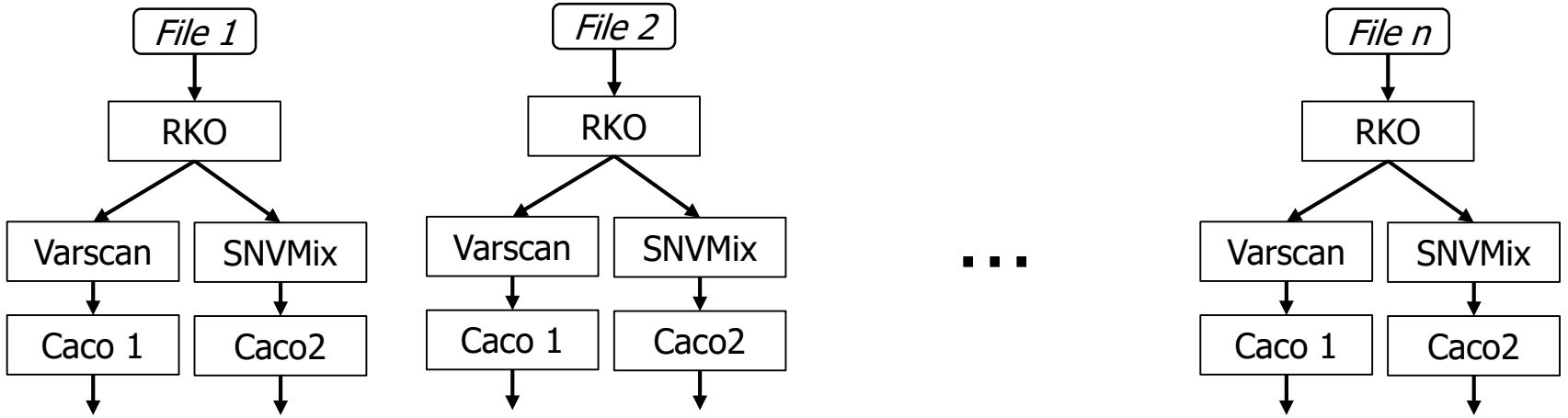


- A logical DAW also has concrete input files and produces concrete output files
  - A logical DAW instantiates an abstract DAW



Real Life

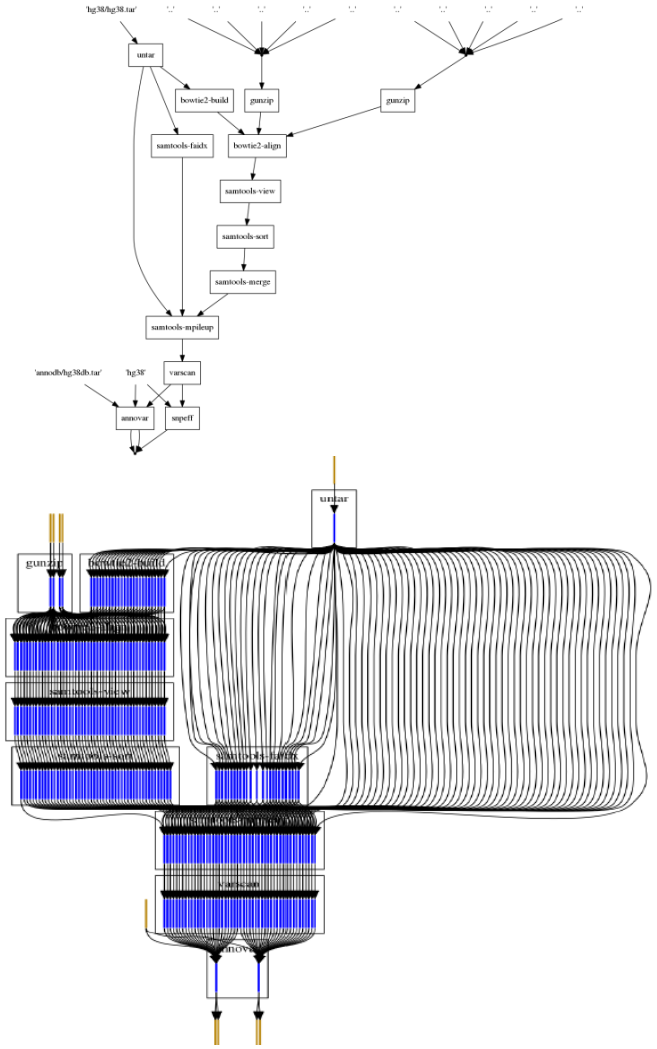
- Typically, DAWs are executed (from start or intermediate) over many files



- With multiple inputs, a single abstract (partial) DAW produces many logical (partial) DAWs
- More complicated as one may think, as the multiplicity of intermediate result files is only determined at runtime

Real Life with Real DAWs

- Multiplicities
  - Some tasks read one file and produce one
    - Read image and produce normalized image
  - Some tasks read one file and produce many
    - Read large image and partition into smaller ones
  - Some tasks read many files and produce one
    - Read partitions and combine into one file



Physical DAWs

- A physical DAW is a logical DAW plus
  - Every (logical) task is assigned to a node for execution
  - Every dependency is assigned to a method of communication

- Logical DAW + schedule + data exchange = physical DAW

Workflowsprachen

- There exist 100dres of workflow systems and hardly any standardization
- Workflow specification
  - Dedicated workflow language, DSL
  - Extension of common programming language with workflow abilities (extension of libraries)
- Different language styles
  - Imperative: Define tasks and dataflow
  - Declarative: Define tasks and dependencies
  - Functional: Define computation as side-effect free functions and function calls
- Large differences
  - Expressiveness: Control flow? (unlimited) loops? Sub-workflows? Recursion?
  - Execution: Tightly integrated workflow engine; steering of external scheduler
  - Development support (editor, debugger, parser, libraries, …)
  - …

Seminar Idea

- We build groups of two
- Every group selects a workflow system / language
- Implements a set of workflows in this language
  - Three simple ones: HelloWorld, sequential WordCount, parallel WordCount
  - One more complex: Bioinformatics read mapping
  - Must execute on your own machines; no distributed setup
- Presentation / thesis: Properties of language, experience with implementation

Sequential WordCount

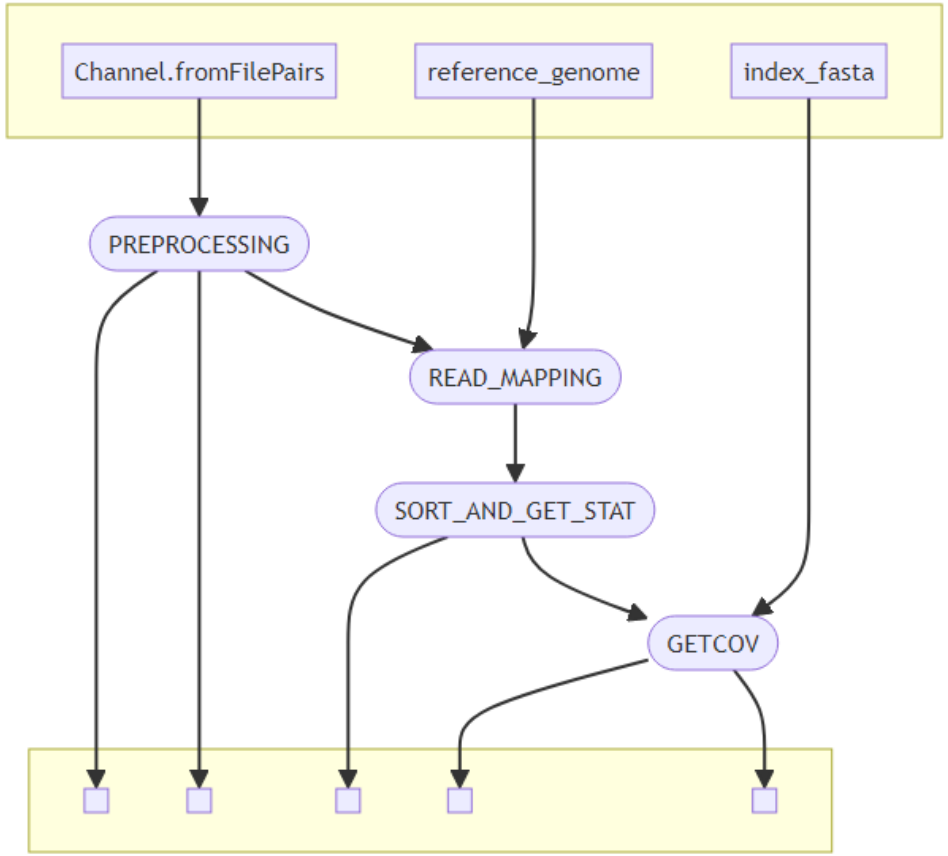
- Given a large ASCII file with natural language text
- Task 1: Read file and remove all special characters (non letter)
- Task 2: Tokenize (whitespace) into flat list of token
- Task 3: Sort file
- Task 4: Read sorted file and count each word
  - Output: Word, count

Parallel WordCount

- Given a large ASCII file with natural language text
- Task 1: Read file and partition into k equally large subfiles
- For every subfile in parallel
  - Task 2: Remove all special characters (non letter)
  - Task 2: Tokenize (whitespace) into flat list of token
  - Task 3: Sort file
  - Task 4: Read and count each word
- Task 5: Merge all k sorted token count files into one
  - Output: Word, count

Bioinformatics Workflow

- Simplified version of a real Bioinformatics workflow for handling sequencing data
- You will get
  - Documentation
  - Container for executables



Who should be here

- Bachelor Informatik (Mono or Kombi) or IMP
- Ability to read English papers
- Knowledge in software engineering, programming languages, distributed systems
- Willingness to work independently
  - Search suitable papers covering a topic, prepare presentations, write seminar thesis

How it will work

- Today: Presentation and choice of topics
  - If desired, we will group teams of 2 students
- 18.11.24: Send an outline of your topic (next slide)
- Before Christmas: Present your topic in 5min talk
- 31.01.25: Meet your advisor to discuss slides
- February: Present your topic in a Blockseminar (dates tba)
- 31.03.25: Write seminar thesis (10-15 pages)

The outline

- Topics will be rather abstract
- Find yourself a set of suitable papers
  - A specific focus is allowed and welcome
- Extract the most important information
- Structure into an outline of your thesis
  - Abstract, chapters, sections,
  - 1-2 sentences per section to describe the content
- Abstract
  - Roughly 20 lines – what is the topic, what will the thesis describe?
- Send us outline + references
  - Mark your top-3 references – those that most likely will form the basis of your work

The 5-min flash talk

- Focus on marketing – sell your topic to gain audience
  - What is the topic?
  - Why is it challenging?
  - Why is it cool?
  - What are important applications?
  - What will your talk be about?
- At most 5 slides
- Focus on figures & examples; omit details or algorithms

Presentation

- 25min presentations
- German or English
- Explain topic, methods, experiences, results
- Aim: Your audience should understand what you say

ToC

- Introduction
- Topics
- Assignment
- Hints on presenting your topic and writing your thesis

Topics

Topic	Assigned to
CWL	
Nextflow	
Snakemake	
PyComPSs	
DASK	
Airflow	
Galaxy	
WDL	
Pegasus	

CWL

- Common Workflow Language
- Attempt to define a standard workflow language compatible to many resource managers / schedulers
- Open source
- Elaborate syntax, local execution mode, many adapters
- Growing user base
- Imperative DSL
- https://www.commonwl.org/

Nextflow

- Open source system with commercial version
- Extends Groovy
- Increasingly popular across scientific disciplines
- Declarative nature, interpreted code
- Local engine or adapters to various RM / scheduler
- https://www.nextflow.io/

Snakemake

- Open source system
- Builds on make – rule-based, goal driven
- Very popular in the (European) Bioinformatics community
- Unclear scheduling
- https://snakemake.readthedocs.io/en/stable/

PyComPSs

- Open source system build by the Barcelona Supercomputing Center
- Extends python with a task-based model for distributed / parallel execution
  - Also available for other programming languages
- Can steer many different HPC scheduler
- https://workflows.community/systems/compss/
- https://github.com/bsc-wdc/compss

DASK

- Python extension for distributed computing
- Open source, quite strong traction also in commercial settings
- Closer to cloud than HPC
- https://www.dask.org/

Airflow

- Originally developed by Airbnb, now Apache open source project
- Very popular in cloud-based companies
- Some graphical elements
- https://airflow.apache.org/

Galaxy

- Open source, primarily graphical user interface
- Originally Bioinformatics, but now used in many disciplines
- Integrated with a large task library
- Runs on various servers (no local installation)
- https://galaxyproject.org/

WDL

- Open source system popular in US
- Relatively simple language
- Local execution with Cromwell scheduler
- https://openwdl.org/

Pegasus

- Long established open source workflow project
- Interface to DAGMan for workflow execution
- XML-based specification
- https://pegasus.isi.edu/

Introductory Topics

- Finding and assessing scientific literature
- (Scientific presentations)
- Writing a (seminar) thesis