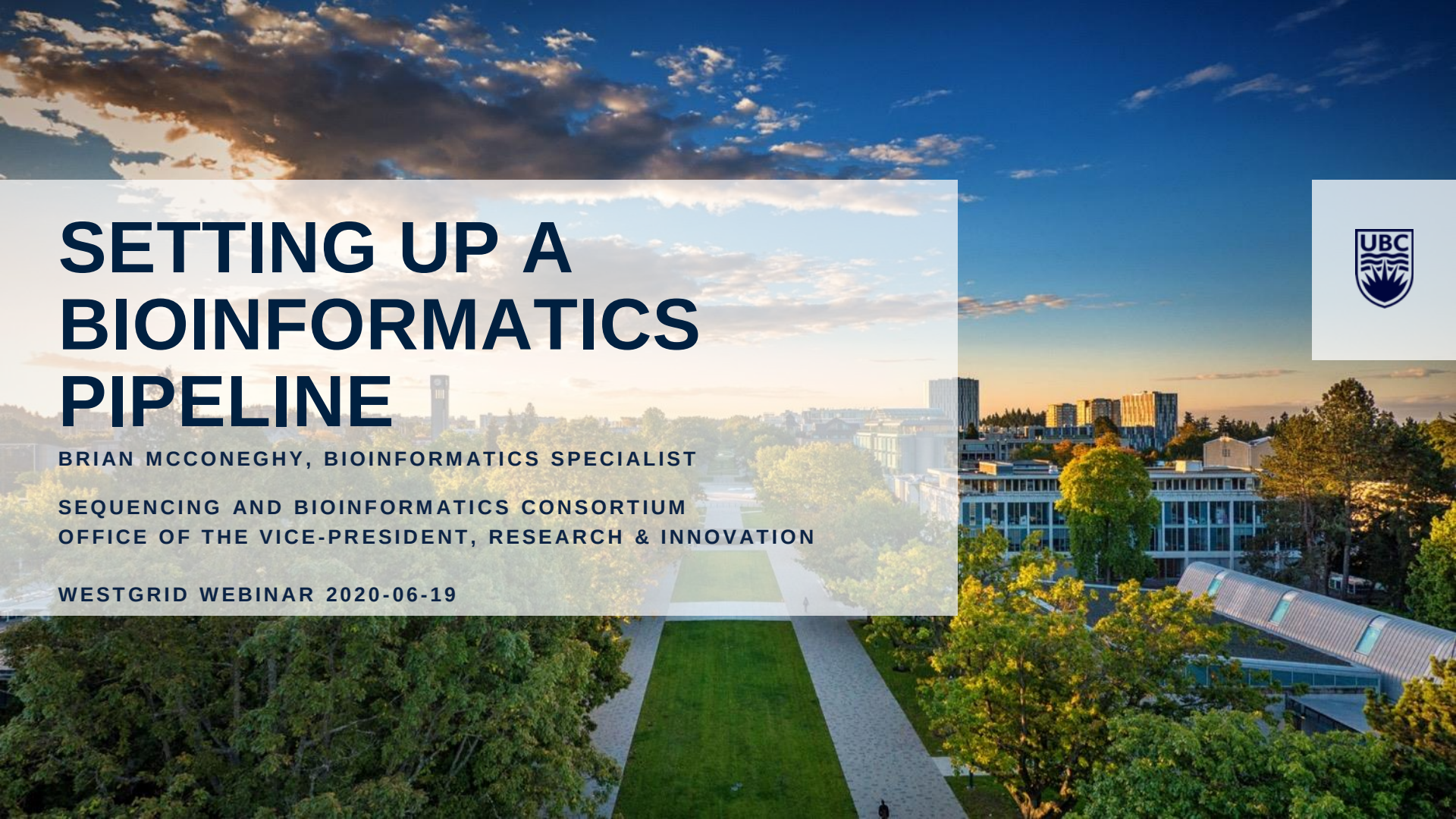


SETTING UP A BIOINFORMATICS PIPELINE

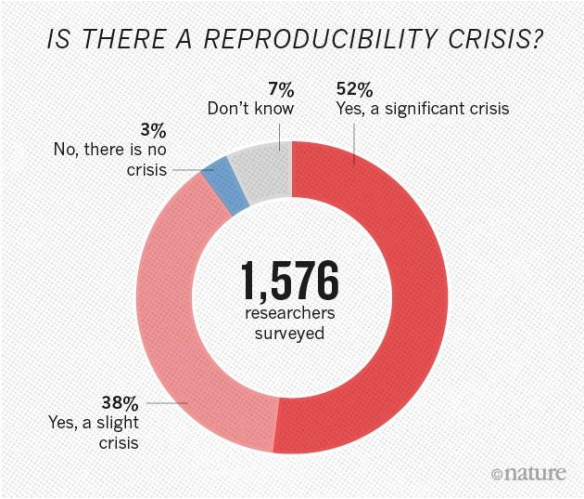
BRIAN MCCONEGHY, BIOINFORMATICS SPECIALIST

SEQUENCING AND BIOINFORMATICS CONSORTIUM
OFFICE OF THE VICE-PRESIDENT, RESEARCH & INNOVATION

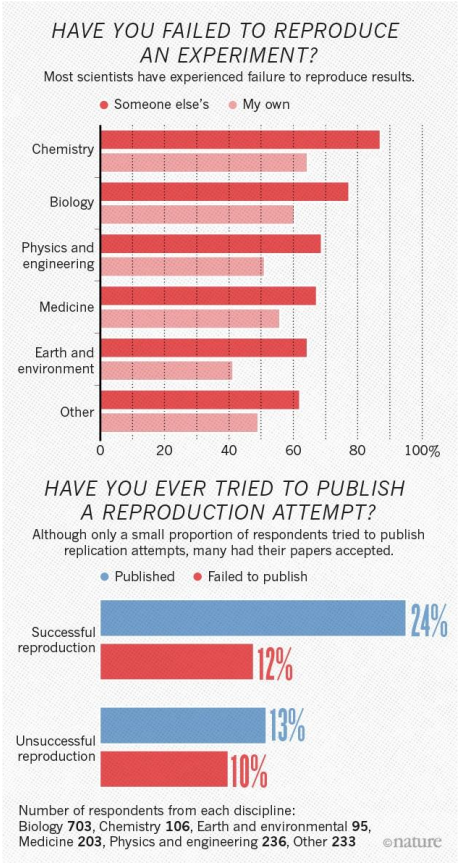
WESTGRID WEBINAR 2020-06-19

REPRODUCIBILITY CRISIS IN SCIENCE

“More than 70% of researchers have tried and failed to reproduce another scientist's experiments, and more than half have failed to reproduce their own experiments.”



Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604), 452-454. doi:10.1038/533452a



Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604), 452-454. doi:10.1038/533452a

Background

Pipelining Tools

Writing the Pipeline

Implementation

Conclusions

Background

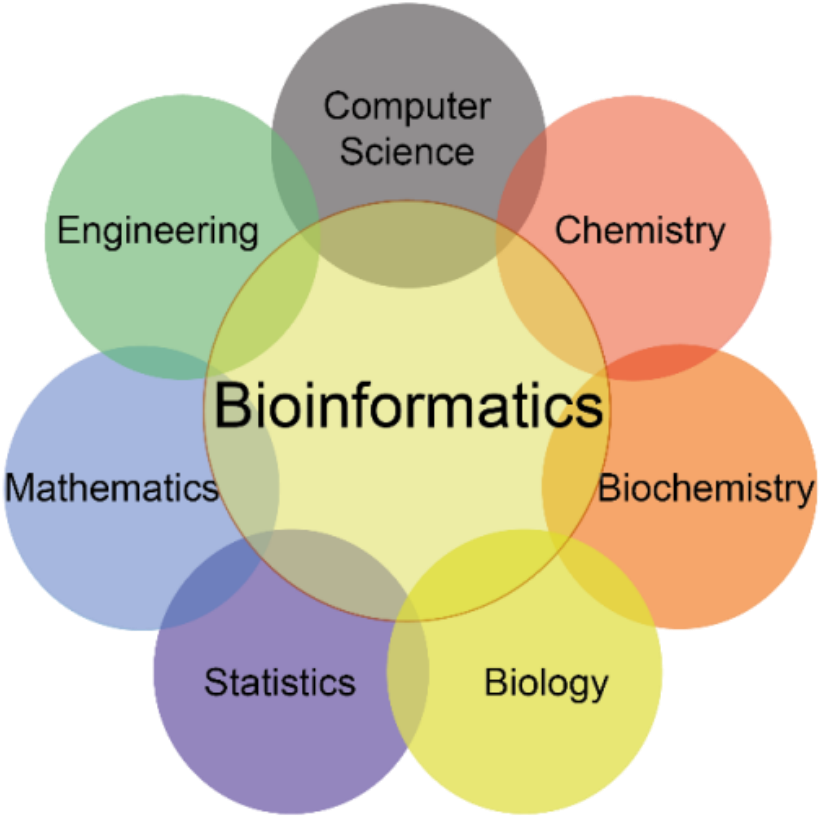
Pipelining Tools

Writing the Pipeline

Implementation

Conclusions

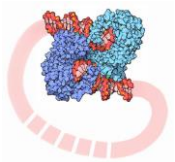
WHAT IS BIOINFORMATICS?



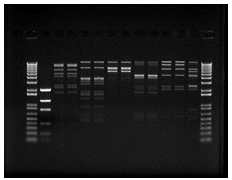
NEXT GENERATION SEQUENCING

Sample (input) QC → Sample Preparation → Sample (library) QC → Sequencing → Data!

https://www.nextadvance.com/



https://pdb101.rcsb.org/motm/84



www.thermofisher.com

www.Illumina.com

https://www.makeuseof.com/tag/best-linux-server-operating-systems/

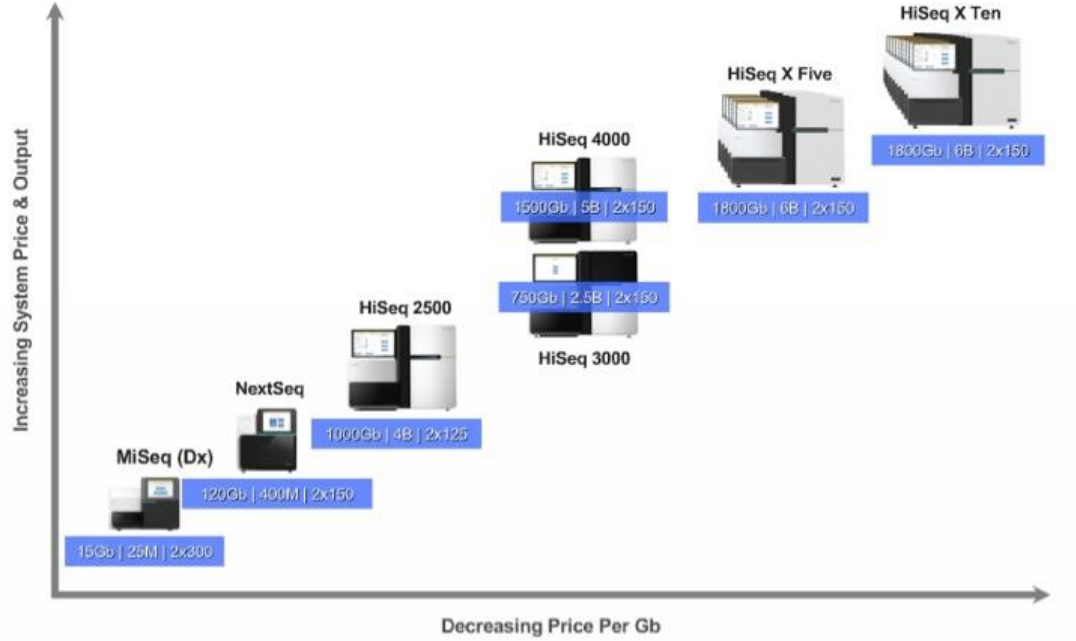
WHAT *IS* NEXT GEN SEQUENCING DATA, EXACTLY?

- High-throughput sequencing technology
  - Generates **millions** of ‘reads’
  - Reads are just strings of G’s, A’s, T’s, and C’s (with associated quality values)

```
@NB999999:999:ZZZZZZZZ9:1:11101:16570:1094 1:N:0:1
AAAGCNGCTGAATTGTTCGCGTTTACCTTGCGTGTACGCGCAGGAAACACT
+
AAAAA#A6EA66EEEAEEE/E//EAE/E//EEEEEEEEAAEEEAEEEEAEE
```

phiX 174 control DNA

Sequencing Power For Every Scale.



https://bloggenohub.files.wordpress.com/2015/01/slide1.jpg

PIPELINES

“In computing, a pipeline, also known as a data pipeline, is a set of data processing elements connected in series, where the output of one element is the input of the next one. The elements of a pipeline are often executed in parallel or in time-sliced fashion. Some amount of buffer storage is often inserted between elements.” - Wikipedia

PIPELINES

- Concept of a pipeline - The good old Unix "Pipe" symbol (|)
  - Essentially sends output from the command to the left of the pipe as input to the command to the right of the pipe
- Evolution of pipeline structure
  - Flat File Era: Data saved locally on game servers
  - Database Era: Data staged in flat files and then loaded into a database
  - Data Lake Era: Data stored in Hadoop/S3 and then loaded into a DB
  - Serverless Era : Managed services used for storage and querying

Background

Pipelining Tools

Writing the Pipeline

Implementation

Conclusions

WHAT AND WHY

- Workflow management system
- Reproducible and scalable data analysis
- Rules, inputs, and outputs

SCALABILITY IS INCREASINGLY IMPORTANT

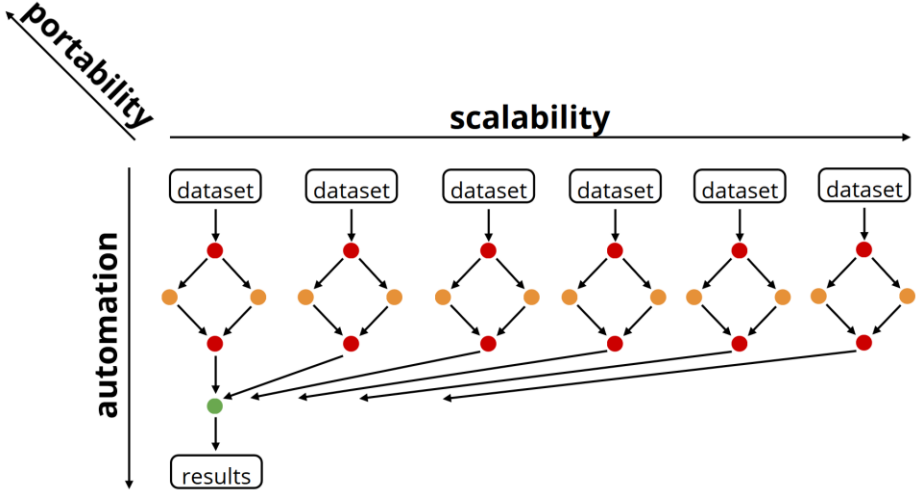
- Cost of NGS is rapidly decreasing and machines are more available which has caused an increase in dataset size, the number of datasets, and hence the number of users

- Must scale up (vertically) with respect to the resources on a single compute node, since the resource usage of some analyses increases with dataset size

- Must scale out (horizontally) to take advantage of compute clusters and clouds

NEEDS

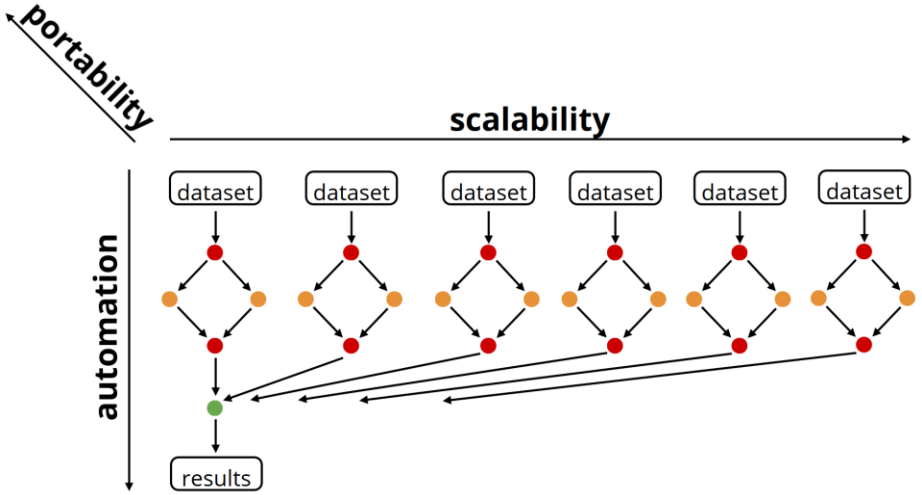
- Reproducible
- Scalable
- Efficient (Parallelizable)
- Portable
- Automated



https://slides.com/johanneskoester/snakemake-short#/3

WANTS

- Ease of development
- Unix-compatible
- FREE

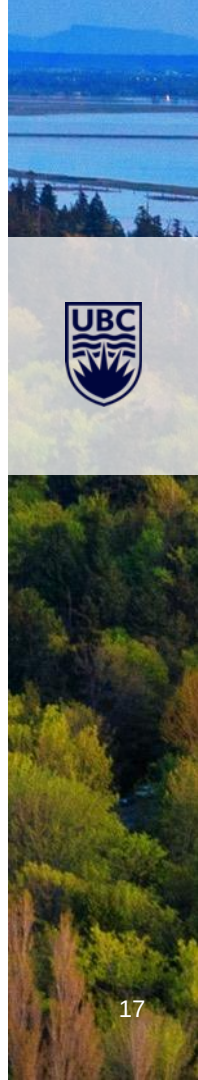


https://slides.com/johanneskoester/snakemake-short#/3

HOLD ON… COULDN'T I JUST WRITE A SCRIPT?

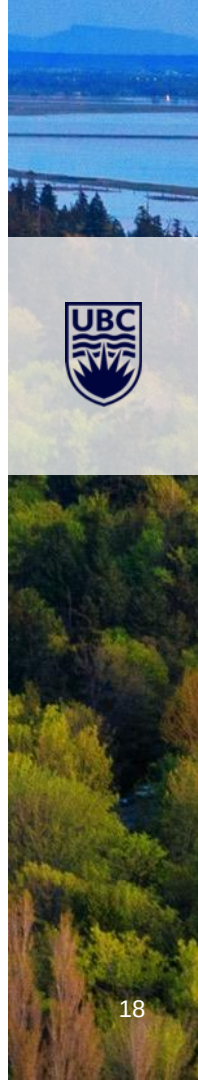
- Written in Unix shell or other scripting languages such as Perl, scripts can be seen as the most basic form of pipeline framework

- Scripting allows variables and conditional logic to be used to build flexible pipelines



HOLD ON… COULDN'T I JUST WRITE A SCRIPT?

- However, in terms of 'robustness', scripts tend to be quite brittle
  - Scripts lack **two key features** necessary for the efficient processing of data: support for '**dependencies**' and '**reentrancy**'
  - **Dependencies** refer to upstream files (or tasks) that downstream transformation steps require as input
  - When a dependency is updated, associated downstream files should be updated as well
  - **Reentrancy** is the ability of a program to continue where it left off if interrupted, obviating the need to restart from the beginning of a process



COMPARISON

- Galaxy
- Ruffus
- Snakemake

COMPARISON

- Galaxy
- Ruffus
- **Snakemake**

Background

Pipelining Tools

Writing the Pipeline

Implementation

Conclusions

CONSIDERATIONS

- What type of data will be processed?
- How many files will be processed?
- How many files will be created and what size will they be?
- What level of resources will be necessary in terms of CPUs, RAM, time?

RESOURCES - HARDWARE

- Cedar - Compute Canada
  - 94,528 Cores
  - 485,826 GB of RAM
  - 10TB scratch space

https://medium.com/monplan/how-we-automated-deployments-and-testing-with-bitbucket-pipelines-bb478c12c55f

RESOURCES - HARDWARE

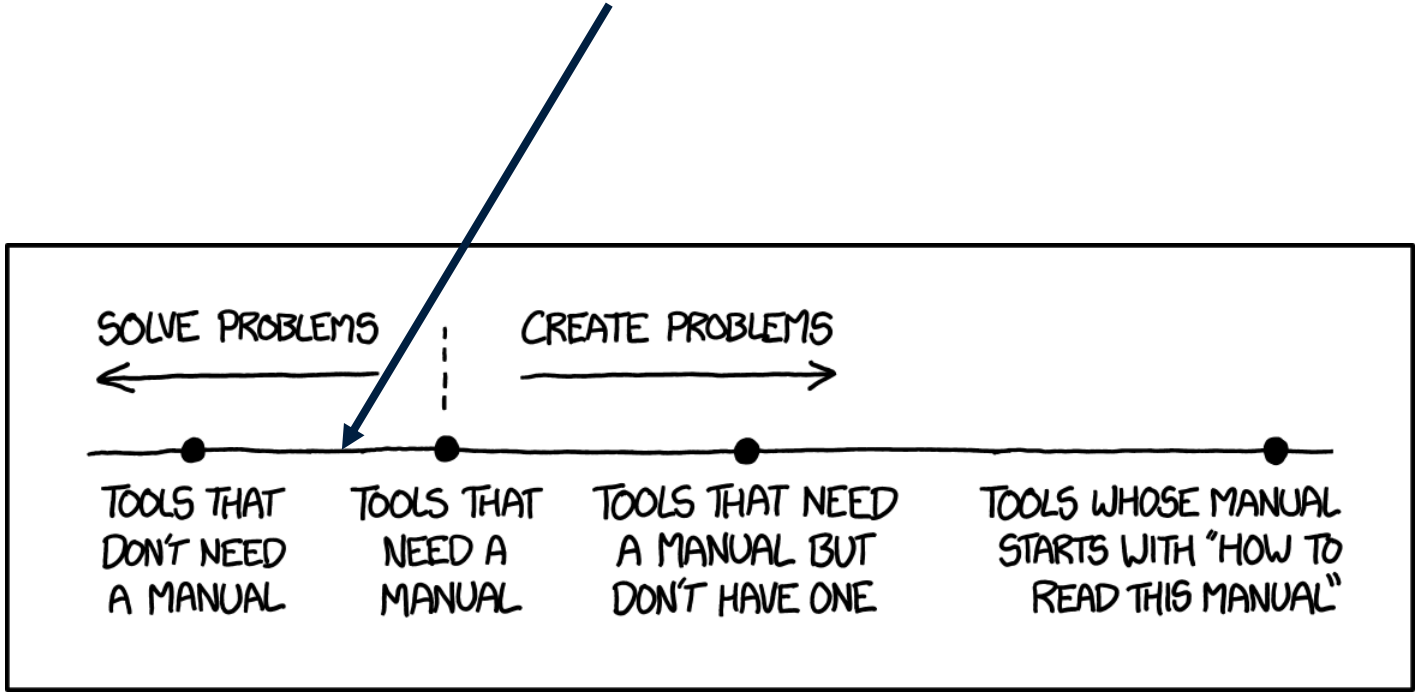
- Advanced Research Computing (ARC)
  - Jamie Rosner
  - Venkat Mahadevan
- VP Research & Innovation (VPRI)
  - Dr. Helen Burt

https://medium.com/monplan/how-we-automated-deployments-and-testing-with-bitbucket-pipelines-bb478c12c55f

SNAKEMAKE

- Decompose workflow into rules
- Rules define how to obtain output files from input files
- Snakemake infers dependencies and execution order

SNAKEMAKE



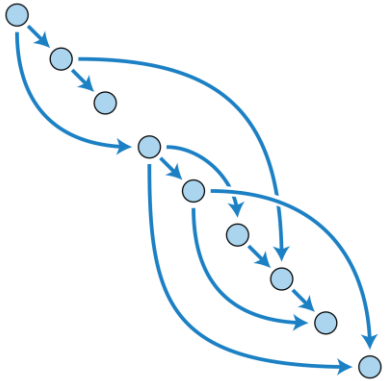
https://xkcd.com/1343/


And now
for something
completely TECHNICAL ...

DIRECTED ACYCLIC GRAPH (DAG)

- Finite, directed graph, with no cycles

- Consists of finitely many vertices and edges
  - Each edge directed from one vertex to another, such that there is no way to start at any vertex v and follow a consistently-directed sequence of edges that eventually loops back to v again



https://en.wikipedia.org/wiki/Directed_acyclic_graph

SIMPLICITY!

```
rule sort:
    input:
        "path/to/dataset.txt"
    output:
        "dataset.sorted.txt"
    shell:
        "sort {input} > {output}"
```

GENERALIZE RULES WITH NAMED WILDCARDS

```
rule sort:
    input:
        "path/to/{dataset}.txt"
    output:
        "{dataset}.sorted.txt"
    shell:
        "sort {input} > {output}"
```

SPECIFY MULTIPLE INPUTS (AND OUTPUTS) REFER BY INDEX

```
rule sort_and_annotate:
    input:
        "path/to/{dataset}.txt",
        "path/to/annotation.txt"
    output:
        "{dataset}.sorted.txt"
    shell:
        "paste <(sort {input[0]}) {input[1]} > {output}"
```

CAN SPECIFY MULTIPLE INPUTS (AND OUTPUTS), AND REFER BY NAME

```
rule sort_and_annotate:
    input:
        a="path/to/{dataset}.txt",
        b="path/to/annotation.txt"
    output:
        "{dataset}.sorted.txt"
    shell:
        "paste <(sort {input.a}) {input.b} > {output}"
```

USE PYTHON WITHIN RULES

```
rule sort:
    input:
        a="path/to/{dataset}.txt"
    output:
        b="{dataset}.sorted.txt"
    run:
        with open(output.b, "w") as out:
            for l in sorted(open(input.a)):
                print(l, file=out)
```

USE EXTERNAL SCRIPTS WITHIN RULES

```
rule NAME:
    input:
        "path/to/inputfile",
        "path/to/other/inputfile"
    output:
        "path/to/outputfile",
        "path/to/another/outputfile"
    script:
        "scripts/script.py"
```

- Have access to an object, “snakemake”, within the script that provides access to the same objects that are available in the run and shell directives (e.g. input, output, params, etc.)

A REAL RULE

- Used in DNA QC pipeline

```
rule bwa_mem_map_reads:
    input:
        get_trimmed_reads
    output:
        temp('mapped/{sample}-{unit}.sorted.bam')
    log:
        'logs/bwa_mem/{sample}-{unit}.log'
    params:
        index = get_genome_index,
        rg = get_read_group_bwa
    threads: 46
    shell:
        '(bwa mem -t {threads} {params.rg} {params.index} {input} | '
        'samtools sort -T $SLURM_TMPDIR/ -o {output} -) 2> {log}'
```

JOB EXECUTION

A job only executes if:

1. output file is the target requested and does not exist
2. output file needed by another executed job (i.e. is an input to another job) and does not exist
3. input file is newer than the output file
4. input file will be updated by other job
5. execution is forced

CLUSTER EXECUTION

- Can set up pipeline profiles
- Execute DAG by way of cluster job submission
- Configuration file
    - Max jobs at a time
    - CPUs
    - MEM
    - General (per profile) or granular (per rule)

Background

Pipelining Tools

Writing the Pipeline

Implementation

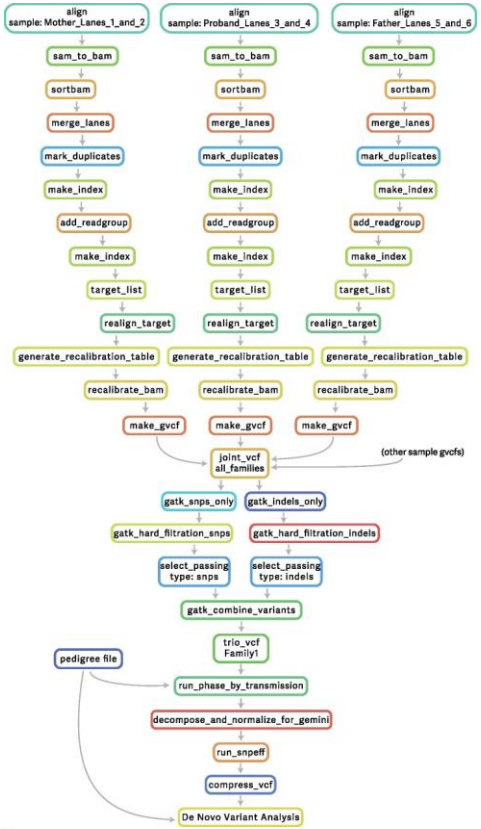
Conclusions

IMPLEMENTATION

- Conda environment
- Version controlled - GitHub
- Some of SBC's pipelines
    - Paired-end (or single-end) DNA QC
    - Paired-end RNA QC
    - RNA-seq differential gene expression
    - 16S Metagenomics

DNA QC WORKFLOW – 2 SAMPLES

WORKFLOWS CAN BE COMPLEX



Leipzig, J. (2016). A review of bioinformatic pipeline frameworks. *Briefings in Bioinformatics*. doi:10.1093/bib/bbw020

Background

Pipelining Tools

Writing the Pipeline

Implementation

Conclusions

CONCLUSIONS

- Snakemake satisfies many of the needs of the SBC and is simple to work with
- The complex pipelines the SBC has implemented are being tracked, in an automated fashion
- Implementation allows for reproducible, scalable, flexible, trackable QC


THE UNIVERSITY OF BRITISH COLUMBIA
Link to GitHub with workshop instructions:
https://bit.ly/2Xf4HN6
brian.mcconeghy@ubc.ca
GenomeCanada