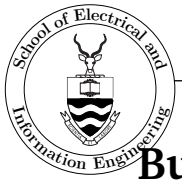


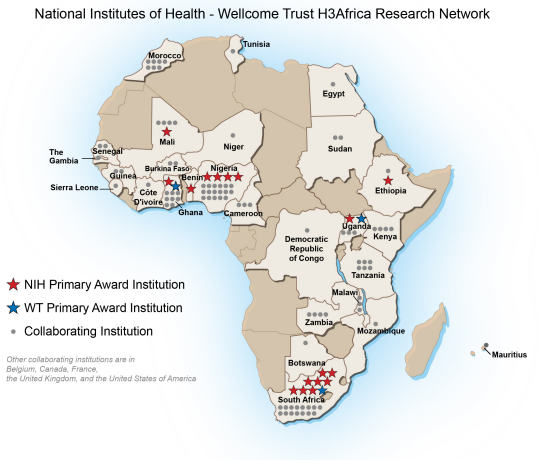
Building portable pipelines for reproducible scientific workflows: The H3ABionet Pipelines Project

Scott Hazelhurst
University of the Witwatersrand, Johannesburg

April 2018

Intro

H3A: Human Heredity and Health in Africa



Key components of H3Africa

Advancing genomics research in Africa

- >20 research projects and collaborative centres
- Training projects
- Biorepositories
- Pan-African Bioinformatics Network for H3Africa

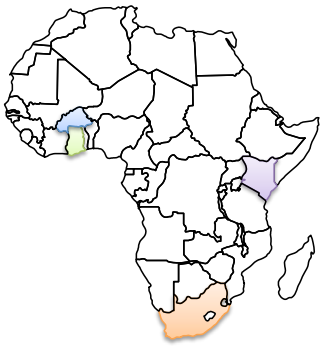
Significant collaborative work – harmonisation, projects.

AWI-Gen

Given as an example project.

AWI-Gen Project
Genetic & environmental factors in cardio-metabolic disorders in African populations – hub at Wits

- DPHRU, Wits, Soweto
- Wits Agincourt Research Unit
- Dikgale HDSS, University of Limpopo
- APHRC, Nairobi, Kenya
- Navrongo Health Research Centre, Ghana
- CRUN, Nanoro, Burkina Faso

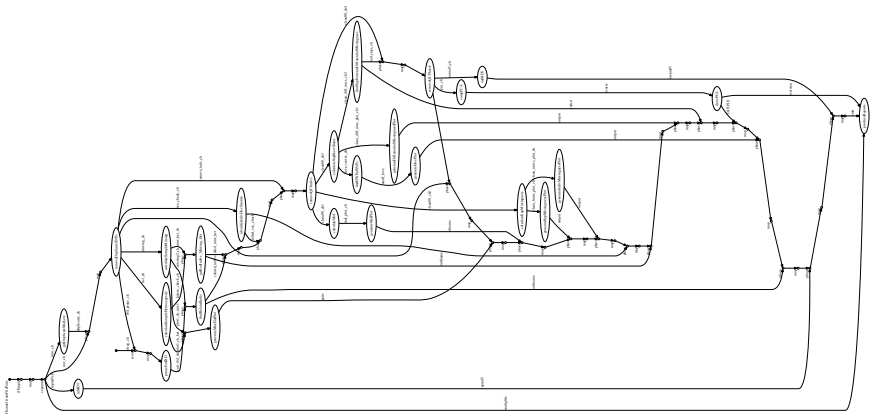


Data to be collected

12000 participants in all – data collected at site

- Extensive personal histories
- Measured, weighed, scanned, blood, urine samples
- DNA extracted, genotyped

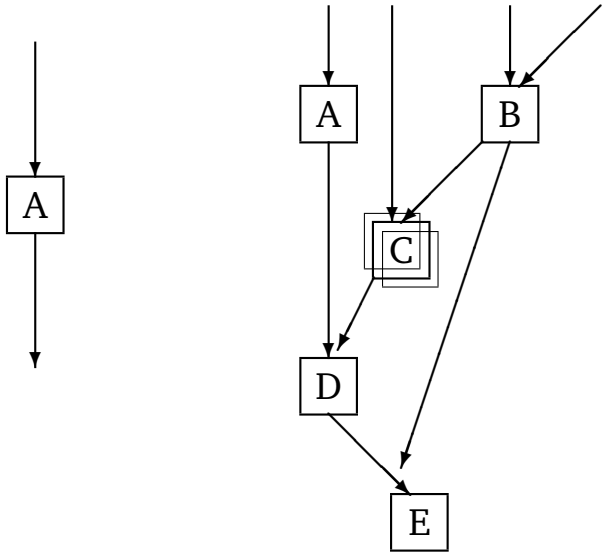
Genome-wide association test: relating genotype data to the phenotype data



Why a pipeline?

- **large data set**
- data can be sliced in many ways
- different phenotypes
- **quality control crucial**
- needs to be reproducible
- needs to be portable
- **under tight deadlines**

Why pipelines?



- Many scientific applications are complex – so complex to
  - install
  - run
- Computationally expensive
- Must be reproducible
  - run with different parameters
  - so that other people can reproduce
- Must be portable

Primary goals:

- managing complexity in the environment
- managing complexity of the workflow

Also:

- exploiting heterogeneous environments
  building laptop-to-HPC, desktop to cloud

H3ABioNet

Pan-African Bioinformatics Network for H3Africa

H3ABioNet created by NIH in 2012 to complement H3Africa research projects:

- bioinformatics key for the projects
- bioinformatics capacity in Africa sparse

Goals:

- Support H3A projects
- Build capacity in bioinformatics
- > 30 nodes in 15 African countries; US partner
- Central node at UCT – Overall PI, Nicola Mulder

Overall goal: improve capacity through training and infrastructure development

- Education & Training
- Pipelines and Computing
- Tools and Webservices
- Health Informatics
- Database and resources

1 Overview

H3A BioNet Pipelines Project

1. Strategic decision by BioNet to explore
    - "cloud computing"
    - build skills in pipelines
2. Needs of H3A partners
3. BioNet partners at the University of Illinois/NCSA
4. Work at Wits from 2014-2016

Overview of project

- Launched in May 2016
- Involved about 30 people from over 10 institutions, led by Sumir Panji.
- Identified key people, planning stared
- Ran 5-day "Cloud Hackathon" at the University of Pretoria in August 2016
- Pipelines published, paper written

Goals

1. Develop production-quality pipelines for key workflows
    - Direct support for stated needs
    - Position BioNet strategically
2. Develop human capacity for building pipelines
3. Explore different technologies

Constraints

1. Must be highly portable, scalable
    - Ideally laptop to CHPC
    - Support cloud environments
2. Must have skills within the network : both workflow and technology
3. Limited resources : start with focus and explore other technologies and workflows later

Technology solutions

- Containers
- Worfklow languages

2 Containers

Containerisation

Abstraction from the environment

- "Escaping dependancy hell"

Challenges

Environment complex and heteroegeneous

- Individual pieces of software are complex, may have specific library, OS requirements
- Requirements may conflict with our environment

  e.g. smc requires a library which requires LIBC 2.14. We run 2.12 – can't upgrade
- Multiple packages even more complex

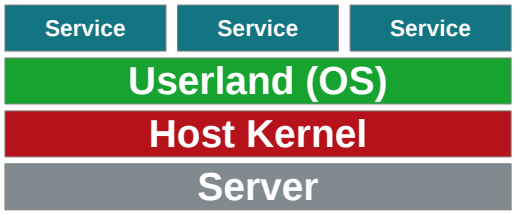
  Requirements may conflict with each other

Containerisation

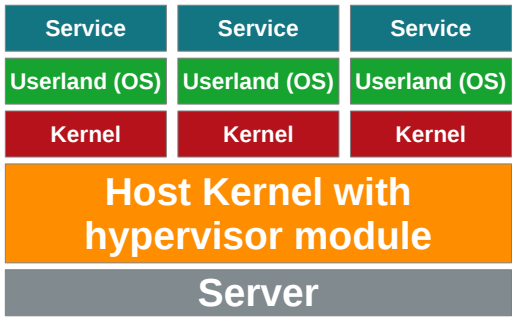
"Light-weight virtualisation" – kernel provides support for **containers**

- Can run jobs/systems in containers
- resource isolation and management
- CPU, memory
- file system
- namespace

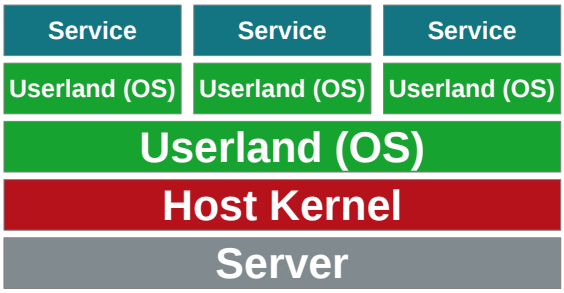
Several examples: Docker, Singularity, Shifter, rkt



Traditional Setup



Virtualisation



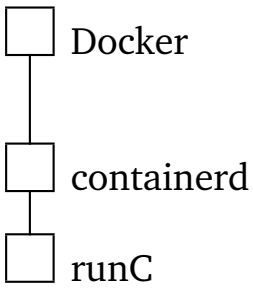
Containerisation

Picture credit: Long et al. Use of containerisation as an alternative to full virtualisation in grid environments. *Journal of Physics: Conference Series* **664**, 2015. doi:10.1088/1742-6596/664/2/022027

Docker

Best known containerisation software

- Linux
- macOS
- Windows 10 Pro, Enterprise, Education + HyperV



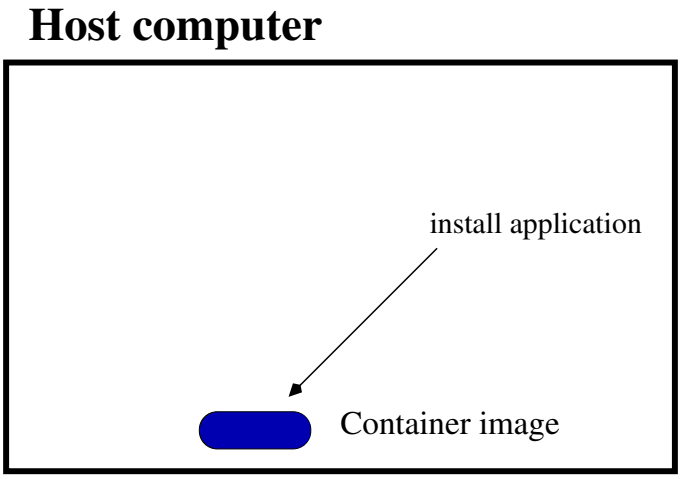
Building and deploying

Several ways of creating Docker images

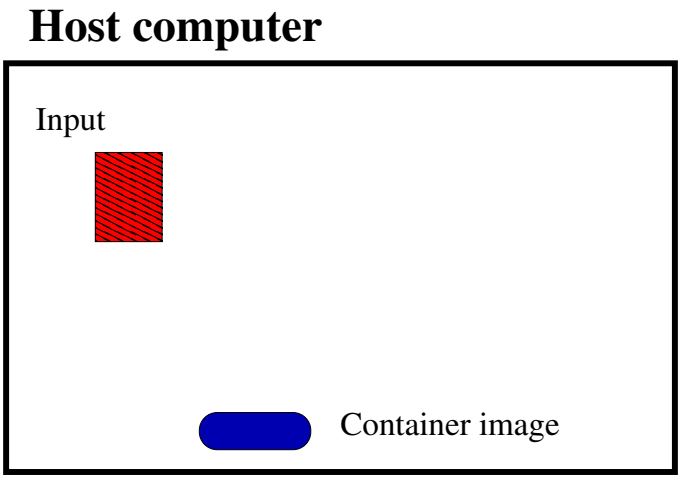
Can build and deploy from services such as

- DockerHub
- quay.io

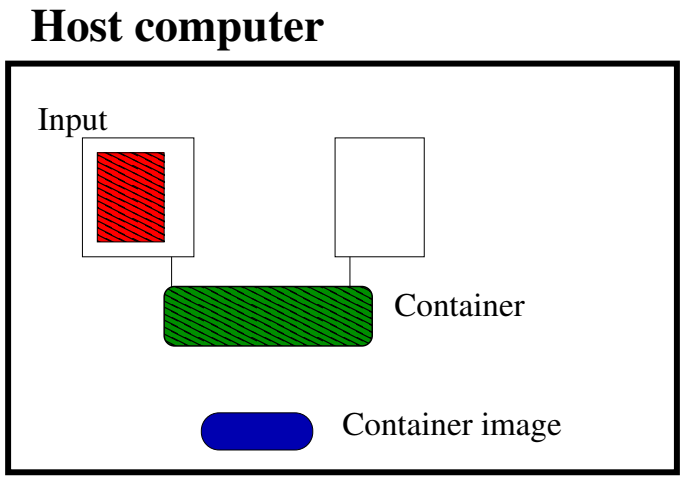
Typical use for scientific application



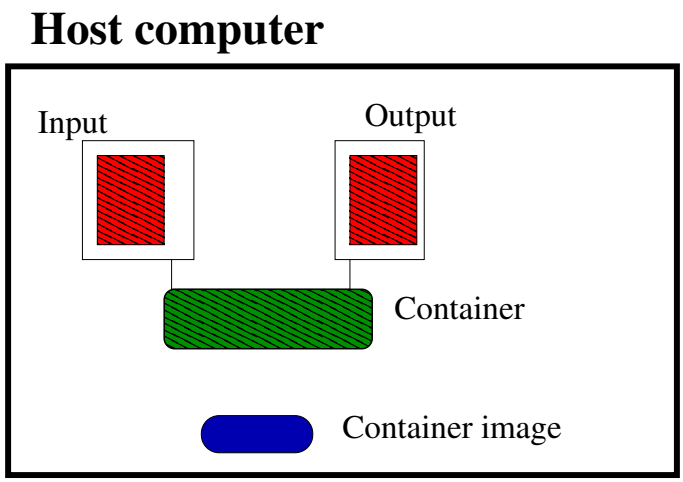
Typical use for scientific application



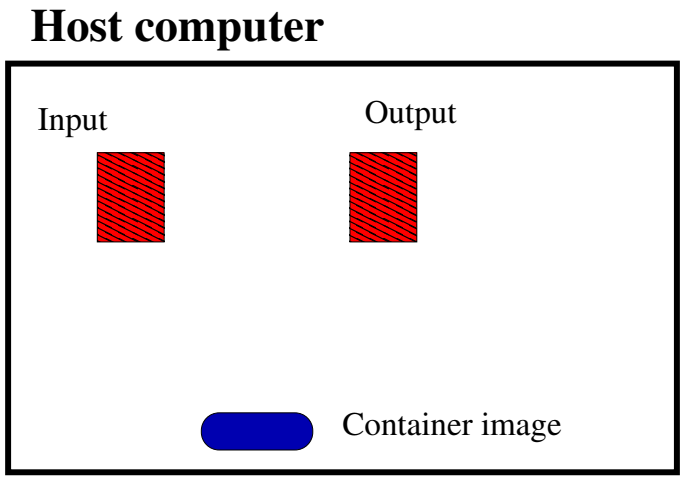
Typical use for scientific application



Typical use for scientific application



Typical use for scientific application



Containers for workflows
Each step in workflow has its own container – abstracts the environment

- Choose right OS for each application
- Only need to install dependancies for that application
- Highly portable : install once, deploy everywhere

Workflow languages

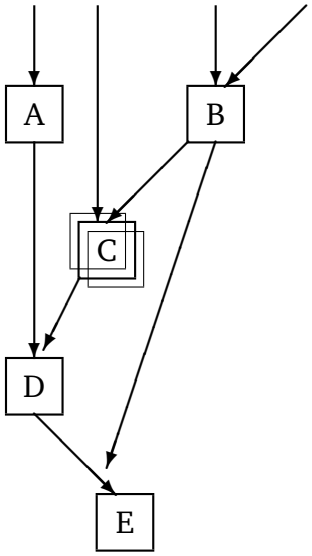
Managing complex workflows

Workflow

Challenges

Scientific applications require

- Multiple data files
- Multiple programs
- Perhaps different parameters
- Want to exploit parallelism



General purpose languages not well suited

- Too low a level of abstraction
- Does not separate workflow from application
- Not reproducible
- Lack of portability – how to exploit parallelism

Workflow languages

Designed to coordinate work rather than doing the work

- Long history
- many different languages and systems available
- hard problem . . .

Examples:

- Galaxy!!
- Taverna, Snakemake, Ruffus, BPipe, JDL, Amazon SWF

Common Workflow Language

Language specification rather than a tool – several tools support it.

- Community-driven, Multi-vendor

- Supports Docker, parallelism
- Language based on YAML
- Extensible

Came out of the bioinformatics community (BOSC) but general purposes

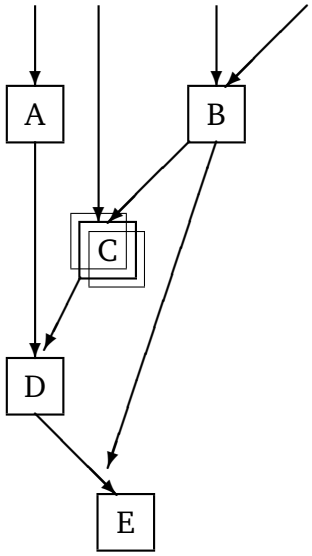
- has buy in from major players

```
class commandLineTool
inputs:
  fastqFile:
    type: File
    inputBinding:
      position: 1
baseCommand: [ fastqc, "--outdir", "res" , "--extract" ]
outputs:
  zippedFile:
    type: File
    outputBinding:
      glob: "*.zip"
  report:
    type: Directory
    outputBinding:
      glob: "res"
```

Nextflow

Developed by the Comparative Bioinformatics group at the Barcelona Center for Genomic Regulation (CRG)

- General purpose worflow system
- DSL based on Groovy
- Portable
- Scalable
- Very easy to install
- Supports Docker
- Supports a range of scheduling systems, cloud




```
process fastQCDo {
    input:
       file input from input_ch
    output:
       file "ids" into id_ch
       file "$input" into orig_ch
    script:
       "fastqc --outdir $output  --extract $input
}
```

Running workflow

```
nextflow run myexample.nf

nextflow run myexample.nf -resume

nextflow run myexample.nf -profile docker

nextflow run myexample.nf -profile pbs
```

What we did

Identified four workflows, two workflow technologies

Nextflow	GWAS	Imputation
CWL	NGS data	Metagenomics

Status of project

- Developed skills in pipeline creation
- Still work in progress but should be finalised soon
  - `github.com/h3abionet/h3agwas`
  - `github.com/h3abionet/chipimputation`
  - `github.com/h3abionet/h3abionet16S`
  - `github.com/h3abionet/h3agtk`

Workflows very portable

Used

- local computer (with or without Docker)
- local cluster (with or without Docker)
- Amazon EC2 AMI
- Docker Swarm
- OpenNebula (NCSA, ARC)

Experiences with the workflow languages
Both Nextflow and CWL worked well

- Both have responsive communities
- ?? Nextflow has an easier learning curve
- ?? When we started Nextlow was maturer – but very significant momentum behind CWL.
- ?? CWL may have an advantage in packaging workflows

Process

Successful training

- Experience in how to run hackathons
- Developing pipelines

Future work

1. Extend, maintain
2. Rigorous comparison
3. Look at other systems, e.g., JMS

Acknowledgements

Funded by NIH NHGRI grants U41HG006941, HG006938. Work at different institutions and individuals.

Sumir Panji Anmol Kiran Abayomi Mosaku Ayton Meintjes Brian O'Connor Don Armstrong Eugene de Beste Fourie Joubert Gerrit Botha Hocine Bendou Lerato Magosi Long Yee Luda Mainzer Mustafa Alghali Michael Crusoe Nicola Mulder Oussema Souiai Peter van Heusden Phelelani Mpangase Rob Clucas Scott Hazelhurst Shaun Aron Shakuntala Baichoo Segun Jung Alex Rodriguez Ravi Madduri Yassine Souilmi