Genomic Analysis in the Cloud

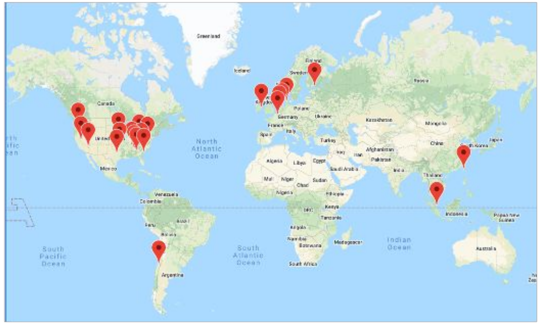
Samantha Zarate
Computational Biomedical Research

October 11, 2021

Slide courtesy of Mike Schatz

Cloud Architecture

- The cloud is built from several very large clusters of computers
  - Effectively infinite resources
  - High-end servers with many cores, many GB RAM, high speed networking, and exabytes of storage



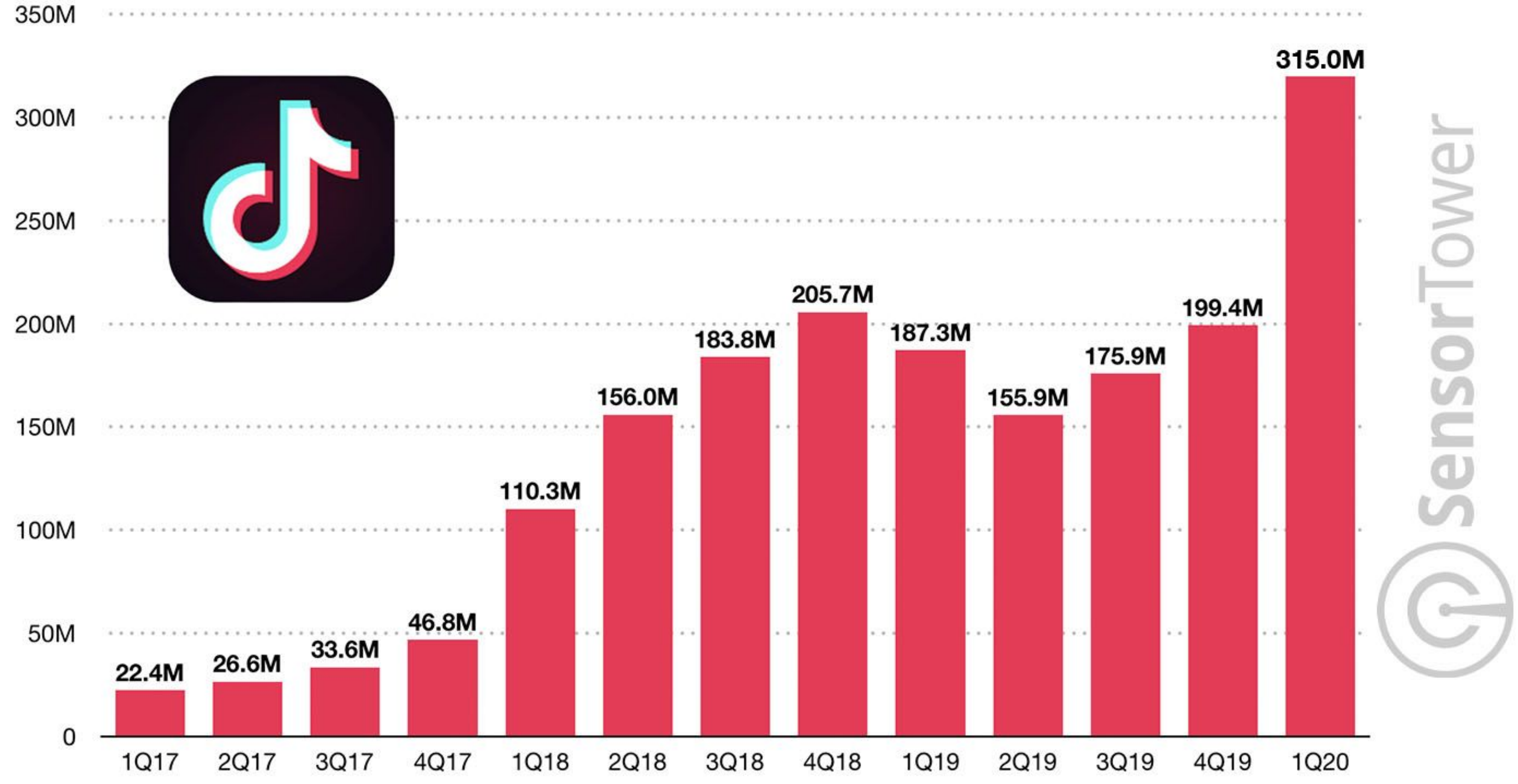
https://www.google.com/about/datacenters/locations/

- Computers run in a virtualized environment
  - Cloud providers subdivide large nodes into smaller instances
  - You are 100% protected from other users on the machine
  - You get to pick the operating system, all software installed

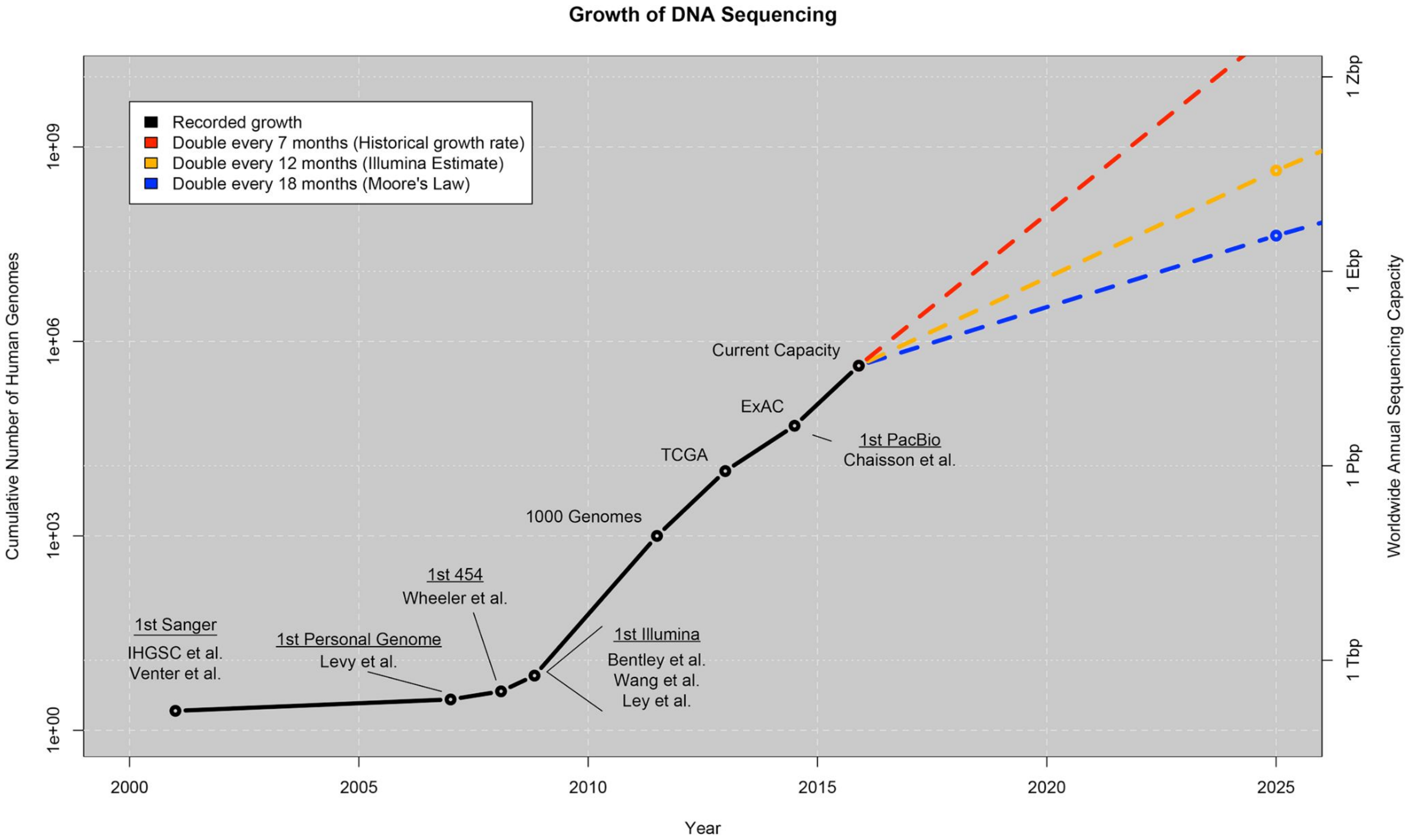
https://en.wikipedia.org/wiki/Virtual_machine

MUSIC ⚡ BAND

TikTok Global Downloads by Quarter

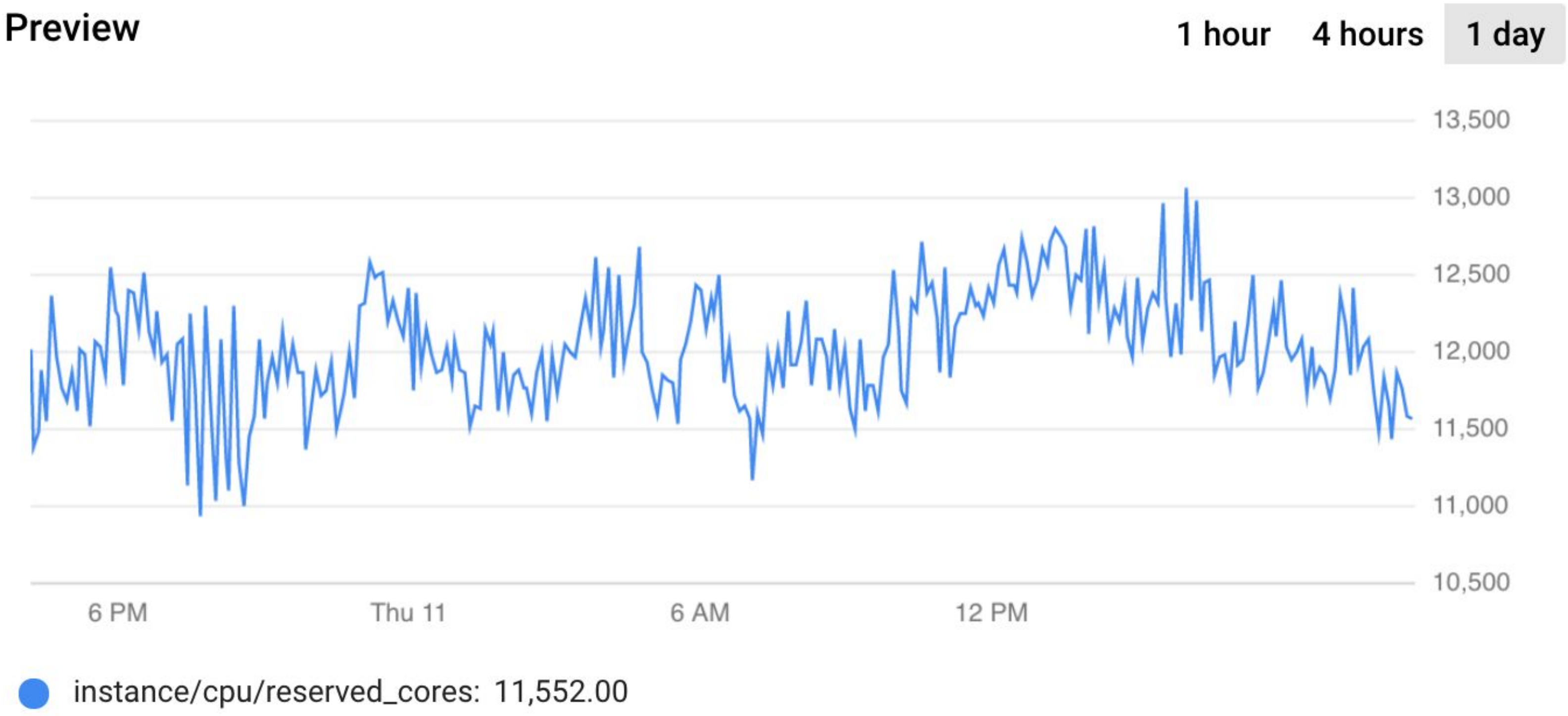


Note: Does not include downloads from third-party Android stores in China or other regions. **Source: Sensor Tower Store Intelligence**



Source: Stephens, Zachary D., *et al*. "Big data: astronomical or genomical?" PLoS Biology (2015). https://doi.org/10.1371/journal.pbio.1002195

Genomic analysis at scale



● instance/cpu/reserved_cores: 11,552.00

The Cloud: What is Terra?

- "A scalable platform for biomedical research"
- Lets you upload your data to the cloud, develop tools, and analyze your data
- Collaborate on projects with other people and share data
- Integrated Jupyter Notebooks

The Cloud: What is AnVIL?

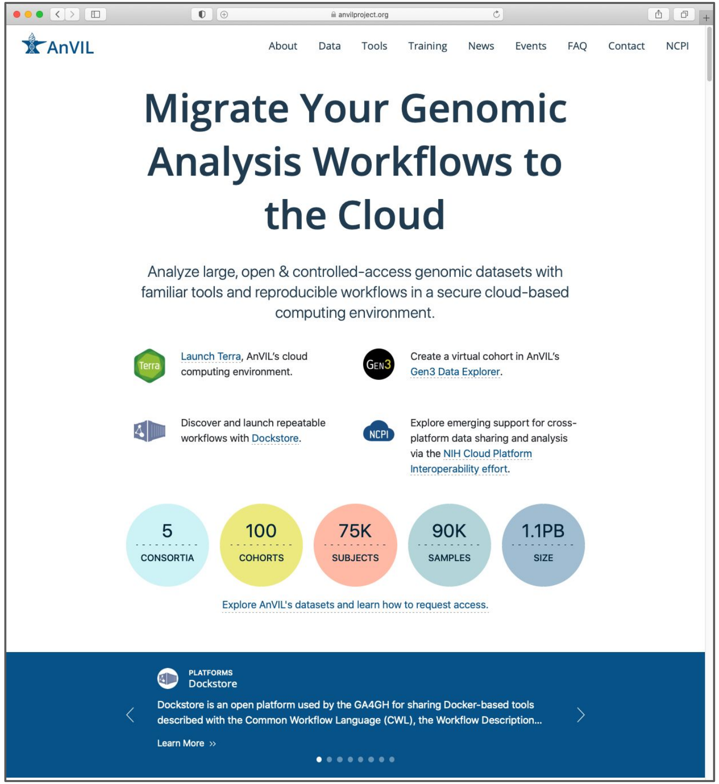
- Featured pipelines demonstrating different techniques (variant calling, RNA-Seq analysis, GWAS)
- Large-scale datasets from different consortia
- Uses Terra as a cloud computing environment
    - Uses Dockstore to share Docker-based tools using CWL, WDL, or NFL

Slide courtesy of Mike Schatz

What is the AnVIL?

Scalable and interoperable computing resource for the genomics scientific community

- **Cloud-based infrastructure**
  - Highly elastic; shared analysis and computing environment
- **Data access and security**
  - Genomic datasets, phenotypes and metadata
  - Large datasets generated by NHGRI programs, as well as other initiatives / agencies
  - dbGaP Authenticated sharing of primary and derived datasets
- **Collaborative computing environment for datasets and analysis workflows**
  - Storage, scalable analytics, data visualization
  - Security, training & outreach, with new models of data access
  - ...for both users with limited computational expertise and sophisticated data scientist users



Migrate Your Genomic Analysis Workflows to the Cloud

Analyze large, open & controlled-access genomic datasets with familiar tools and reproducible workflows in a secure cloud-based computing environment.

Launch Terra, AnVIL's cloud computing environment.

Create a virtual cohort in AnVIL's Gen3 Data Explorer.

Discover and launch repeatable workflows with Dockstore.

Explore emerging support for cross-platform data sharing and analysis via the NIH Cloud Platform Interoperability effort.

5	100	75K	90K	1.1PB
CONSORTIA	COHORTS	SUBJECTS	SAMPLES	SIZE

Explore AnVIL's datasets and learn how to request access.

PLATFORMS
Dockstore

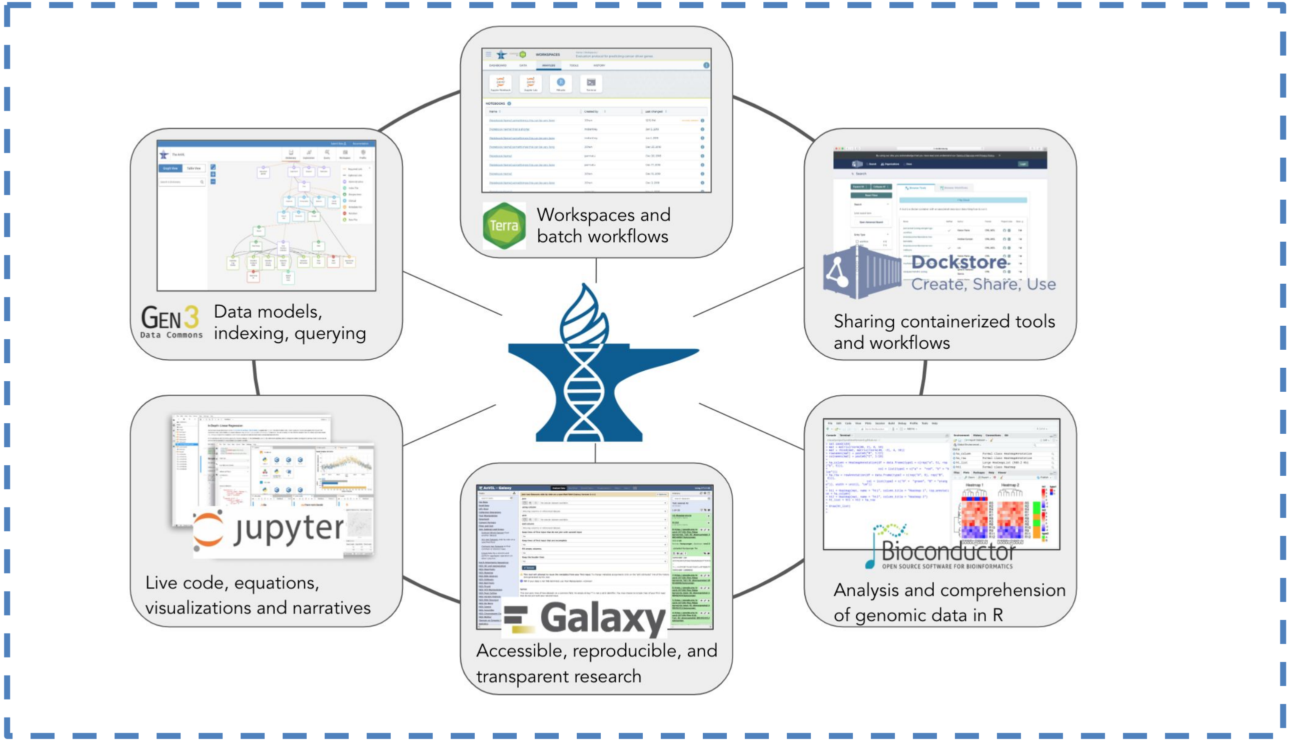
Dockstore is an open platform used by the GA4GH for sharing Docker-based tools described with the Common Workflow Language (CWL), the Workflow Description...

Learn More »

https://anvilproject.org

Slide courtesy of Mike Schatz

All data use and analysis in a FISMA moderate environment



Terra Workspaces and batch workflows

Dockstore Create, Share, Use

Sharing containerized tools and workflows

Bioconductor OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Analysis and comprehension of genomic data in R

Galaxy

Accessible, reproducible, and transparent research

jupyter

Live code, equations, visualizations and narratives

GEN3 Data Commons

Data models, indexing, querying

FISMA Moderate

2 ATOs

Pursuing FedRAMP

Implemented on Google Cloud Platform

Primary data storage costs covered by AnVIL,

user private data and compute billed directly through Google

The Cloud: Why use Terra?

- One lightning strike won't take out your servers
- Share data and projects with other people
- Data provenance and reproducibility
- Access to AnVIL data
- Portability
    - Import WDL workflows

WDL: What is WDL?

The Workflow Description Language (WDL) is a way to specify data processing workflows with a human-readable and -writeable syntax. WDL makes it straightforward to define analysis tasks, chain them together in workflows, and parallelize their execution.

https://openwdl.org/

WDL: What is WDL?

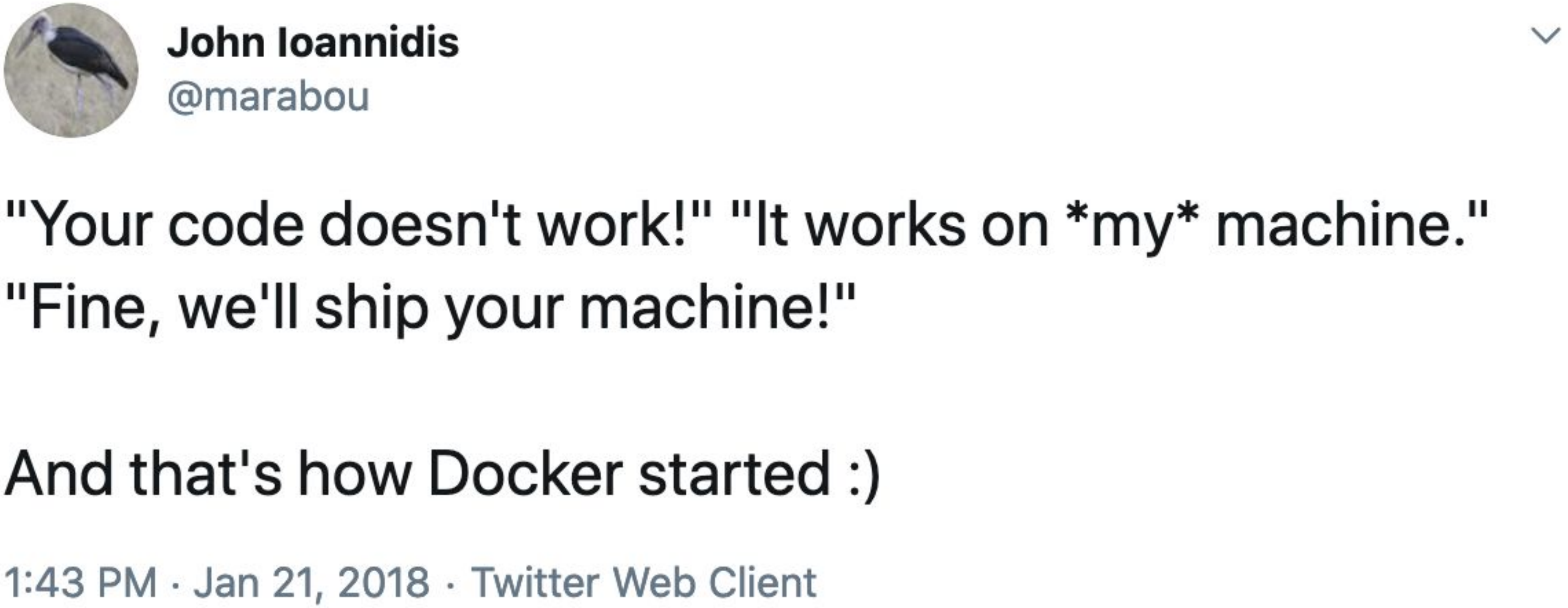
The Workflow Description Language (WDL) is a way to specify data processing workflows with a human-readable and -writeable syntax. WDL makes it straightforward to define analysis tasks, chain them together in workflows, and parallelize their execution.

https://openwdl.org/

WDL: What is WDL?

- Initially developed by the Broad Institute
- Now open-source and led by individuals from the Broad, DNAstack, UCSC, and DNAnexus (+1 freelancer)
- Not an execution engine -- needs an engine to run it
  - Cromwell
  - miniwdl
  - dxWDL

Sidebar: What is Docker?



John Ioannidis
@marabou

"Your code doesn't work!" "It works on *my* machine." "Fine, we'll ship your machine!"

And that's how Docker started :)

1:43 PM · Jan 21, 2018 · Twitter Web Client

Sidebar: What is Docker?

- Docker image: "everything needed to run an application: code, runtime, system tools, system libraries and settings"
- Docker container: an image at runtime (`docker run <image_id>`)

WDL: Example WDL

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

Specifies which version of openWDL to use

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

Workflow definition

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

Tasks: building blocks of WDL files

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

Task definition

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

Specify inputs

WDL: Example WDL

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL variable called with ~

WDL: Example WDL

Docker image defines what's installed on machine

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

Define outputs based on files generated

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

Pass outputs to workflow

WDL: Example WDL

```
version 1.0
workflow FindInFile {
  input {
     String needle
     File haystack
   }

  call find {
     input:
          to_find = needle,
          in_file = haystack
   }

  output {
     File locationsInFile = find.found
   }
}
```

```
task find {
     input {
          String to_find
          File in_file
     }

     command <<<
          grep "~{to_find}" "~{in_file}" > found.txt
     >>>

     runtime {
          docker : "ubuntu:20.04"
     }

     output {
          File found = "found.txt"
     }
}
```

WDL: Example WDL Run (with miniwdl)

```
$ miniwdl run find.wdl needle=wind haystack=corpus.txt
2021-10-11 11:54:36.368 wdl.w:FindInFile workflow start :: name: "FindInFile", source: "find.wdl", line: 2,
column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile"
2021-10-11 11:54:36.374 wdl.w:FindInFile miniwdl :: version: "v1.3.0"
2021-10-11 11:54:36.387 wdl.w:FindInFile issue :: job: "call-find", callee: "find"
2021-10-11 11:54:36.388 wdl.w:FindInFile.t:call-find task start :: name: "find", source: "find.wdl", line:
19, column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile/call-find", thread: 123145585725440
2021-10-11 11:54:36.767 wdl.w:FindInFile.t:call-find docker swarm resources :: workers: 1, max_cpus: 4,
max_mem_bytes: 5587193856, total_cpus: 4, total_mem_bytes: 5587193856
2021-10-11 11:54:36.787 wdl.w:FindInFile.t:call-find docker image :: tag: "ubuntu:20.04", id:
"sha256:bb0eaf4eee00c28cb8ffd54e571dd225f1dd2ed8d8751b2835c31e84188bf2de", RepoDigest:
"ubuntu@sha256:cbcf86d7781dbb3a6aa2bcea25403f6b0b443e20b9959165cf52d2cc9608e4b9"
2021-10-11 11:54:38.790 wdl.w:FindInFile.t:call-find docker task exit :: state: "complete", exit_code: 0
2021-10-11 11:54:39.270 wdl.w:FindInFile.t:call-find done
2021-10-11 11:54:39.271 wdl.w:FindInFile finish :: job: "call-find"
2021-10-11 11:54:39.272 wdl.w:FindInFile done
{
  "outputs": {
      "FindInFile.locationsInFile":
"/Users/slz/wdl/20211011_115436_FindInFile/out/locationsInFile/found.txt"
  },
  "dir": "/Users/slz/wdl/20211011_115436_FindInFile"
}
```

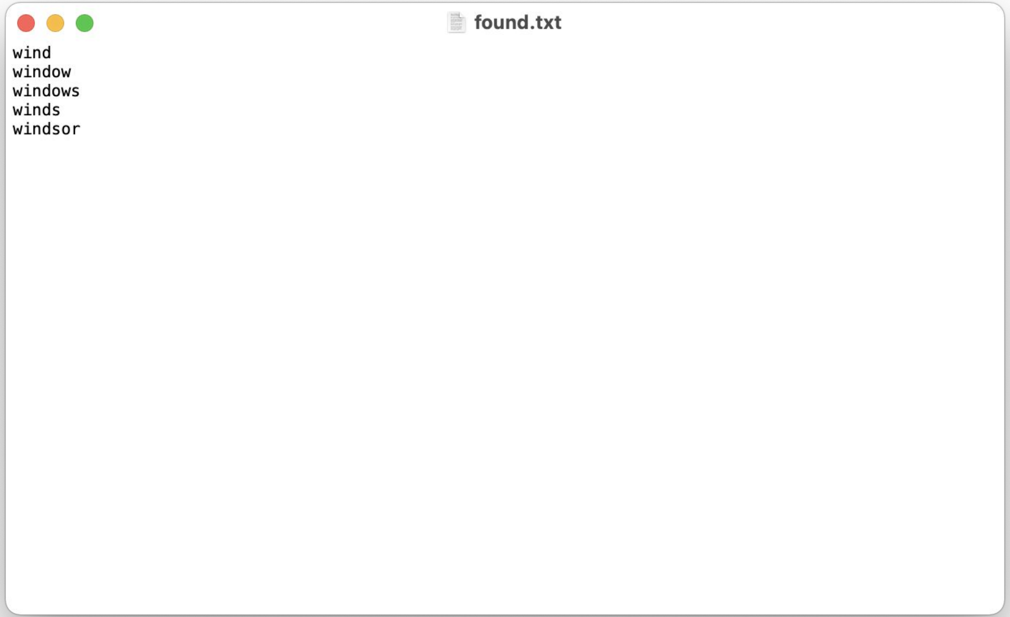
WDL: Example WDL Run (with miniwdl)

```
$ miniwdl run find.wdl needle=wind haystack=corpus.txt
2021-10-11 11:54:36.368 wdl.w:FindInFile workflow start :: name: "FindInFile", source: "find.wdl", line: 2,
column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile"
2021-10-11 11:54:36.374 wdl.w:FindInFile miniwdl :: version: "v1.3.0"
2021-10-11 11:54:36.387 wdl.w:FindInFile issue :: job: "call-find", callee: "find"
2021-10-11 11:54:36.388 wdl.w:FindInFile.t:call-find task start :: name: "find", source: "find.wdl", line:
19, column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile/call-find", thread: 123145585725440
2021-10-11 11:54:36.767 wdl.w:FindInFile.t:call-find docker swarm resources :: workers: 1, max_cpus: 4,
max_mem_bytes: 5587193856, total_cpus: 4, total_mem_bytes: 5587193856
2021-10-11 11:54:36.787 wdl.w:FindInFile.t:call-find docker image :: tag: "ubuntu:20.04", id:
"sha256:bb0eaf4eee00c28cb8ffd54e571dd225f1dd2ed8d8751b2835c31e84188bf2de", RepoDigest:
"ubuntu@sha256:cbcf86d7781dbb3a6aa2bcea25403f6b0b443e20b9959165cf52d2cc9608e4b9"
2021-10-11 11:54:38.790 wdl.w:FindInFile.t:call-find docker task exit :: state: "complete", exit_code: 0
2021-10-11 11:54:39.270 wdl.w:FindInFile.t:call-find done
2021-10-11 11:54:39.271 wdl.w:FindInFile finish :: job: "call-find"
2021-10-11 11:54:39.272 wdl.w:FindInFile done
{
  "outputs": {
      "FindInFile.locationsInFile":
"/Users/slz/wdl/20211011_115436_FindInFile/out/locationsInFile/found.txt"
  },
  "dir": "/Users/slz/wdl/20211011_115436_FindInFile"
}
```

WDL: Example WDL Run (with miniwdl)

```
$ miniwdl run find.wdl needle=wind haystack=corpus.txt
2021-10-11 11:54:36.368 wdl.w:FindInFile workflow start :: name: "FindInFile", source: "find.wdl", line: 2,
column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile"
2021-10-11 11:54:36.374 wdl.w:FindInFile miniwdl :: version: "v1.3.0"
2021-10-11 11:54:36.387 wdl.w:FindInFile issue :: job: "call-find", callee: "find"
2021-10-11 11:54:36.388 wdl.w:FindInFile.t:call-find task start :: name: "find", source: "find.wdl", line:
19, column: 1, dir: "/Users/slz/wdl/20211011_115436_FindInFile/call-find", thread: 123145585725440
2021-10-11 11:54:36.767 wdl.w:FindInFile.t:call-find docker swarm resources :: workers: 1, max_cpus: 4,
max_mem_bytes: 5587193856, total_cpus: 4, total_mem_bytes: 5587193856
2021-10-11 11:54:36.787 wdl.w:FindInFile.t:call-find docker image :: tag: "ubuntu:20.04", id:
"sha256:bb0eaf4eee00c28cb8ffd54e571dd225f1dd2ed8d8751b2835c31e84188bf2de", RepoDigest:
"ubuntu@sha256:cbcf86d7781dbb3a6aa2bcea25403f6b0b443e20b9959165cf52d2cc9608e4b9"
2021-10-11 11:54:38.790 wdl.w:FindInFile.t:call-find docker task exit :: state: "complete", exit_code: 0
2021-10-11 11:54:39.270 wdl.w:FindInFile.t:call-find done
2021-10-11 11:54:39.271 wdl.w:FindInFile finish :: job: "call-find"
2021-10-11 11:54:39.272 wdl.w:FindInFile done
{
  "outputs": {
      "FindInFile.locationsInFile":
"/Users/slz/wdl/20211011_115436_FindInFile/out/locationsInFile/found.txt"
  },
  "dir": "/Users/slz/wdl/20211011_115436_FindInFile"
}
```

WDL: Example WDL Output



```
wind
window
windows
winds
windsor
```

WDL: Why use WDL?

- Reproducibility
  - Docker images to "snapshot" tools & installations
- Portability
  - Run locally or in the cloud
- Readability
- Everyone else is doing it

Overview of assignment 4

Acknowledgements

- Schatz lab
  - Mike Schatz
  - Sergey Aganezov
  - Melanie Kirsche
  - Srividya Ramakrishnan
  - Sam Kovaka
  - Mike Alonge
  - Arun Das
  - Katie Jenike
  - Margaret Starostik
  - Bohan Ni

WDL, Terra/AnVIL, DNAnexus:

- Mike Lin (self-employed)
- John Didion (DNAnexus)
- Frederick Tan (Carnegie Institution of Washington)
- Broad & DNAnexus support staff

Thanks!

Sources

- OpenWDL: https://openwdl.org/
- Docker: https://www.docker.com/resources/what-container
- Terra: https://static1.squarespace.com/static/5c5a38e12727be0ca6a81209/t/5ccc979c54b774000177f809/1556912029107/Terra_OnePage_Information.pdf
- Meme: https://knowyourmeme.com/memes/how-do-you-do-fellow-kids
- TikTok: https://sensortower.com/blog/tiktok-downloads-2-billion