INTRODUCTION TO NEXTFLOW

Elixir workflows workshop - 1 Dec 2021

Paolo Di Tommaso - CTO Seqera Labs

AGENDA

- 9.30 - 9.50: Intro to Nextflow core concepts

  Paolo Di Tommaso, Seqera Labs

- 9.50 - 10.00: Deploy in the cloud with Nextflow Tower

  Evan Floden, Seqera Labs

- 10.00 - 12.00: Nextflow hands-on tutorial

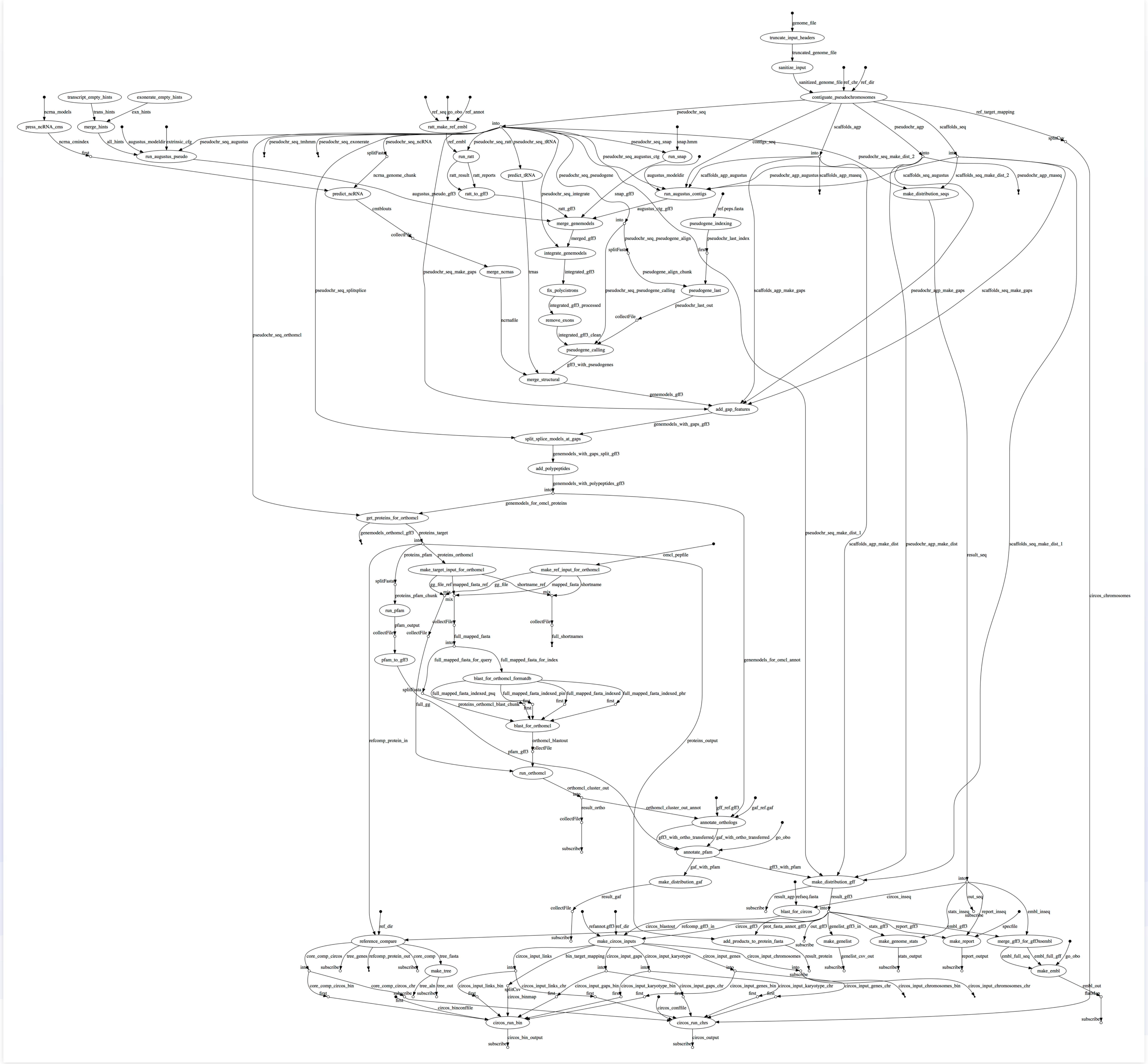
  Luca Cozzuto, Centre for Genomic Regulation (CRG)

QUICK BIO

- Paolo Di Tommaso
- Software engineer
- Creator & maintainer of Nextflow project
- CTO & Co-founder Seqera Labs

GENOMICS WORKFLOWS

- Data analysis applications to extract information from large genomic datasets (TB)
- Embarrassingly parallelisation, can spawn 100s-100k jobs over distributed cluster
- Mash-up of many different tools and scripts
- Complex executions and dependencies graphs → very fragile ecosystem



- 70 tasks
- 55 custom scripts
- 39 software tools & libraries

Complexity ⇒

instability ⇒

not reproducible result

Steinbiss et al., *Companion parassite genome annotation pipeline*, DOI: 10.1093/nar/gkw292

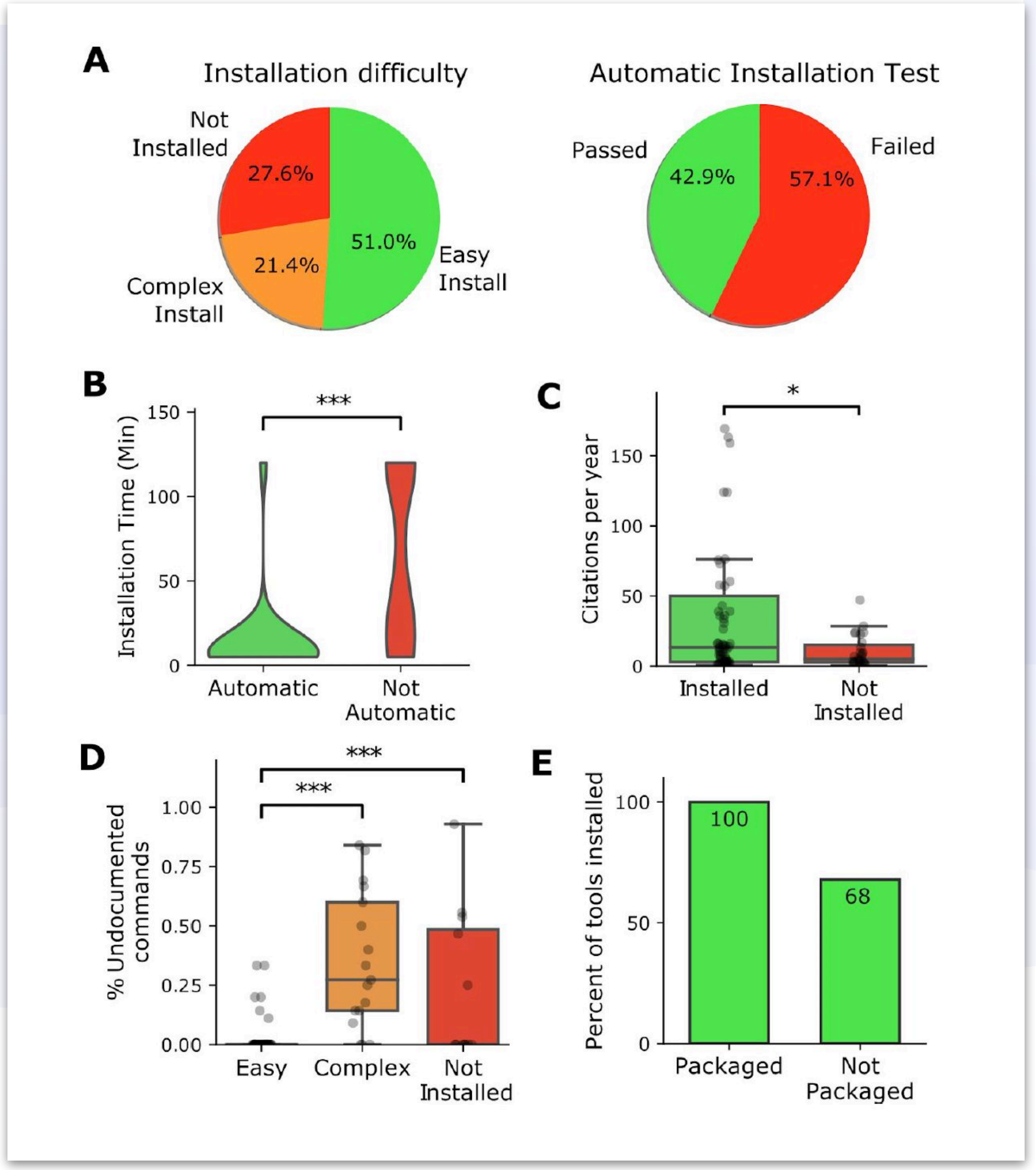
THE PROBLEM WITH REPRODUCIBILITY

A comprehensive analysis of the usability and archival stability of omics computational tools and resources

Serghei Mangul, et al. BioRxiv **doi:** https://doi.org/10.1101/452532

We found that 26% of all omics software resources are currently *not accessible* through URLs published in the paper.

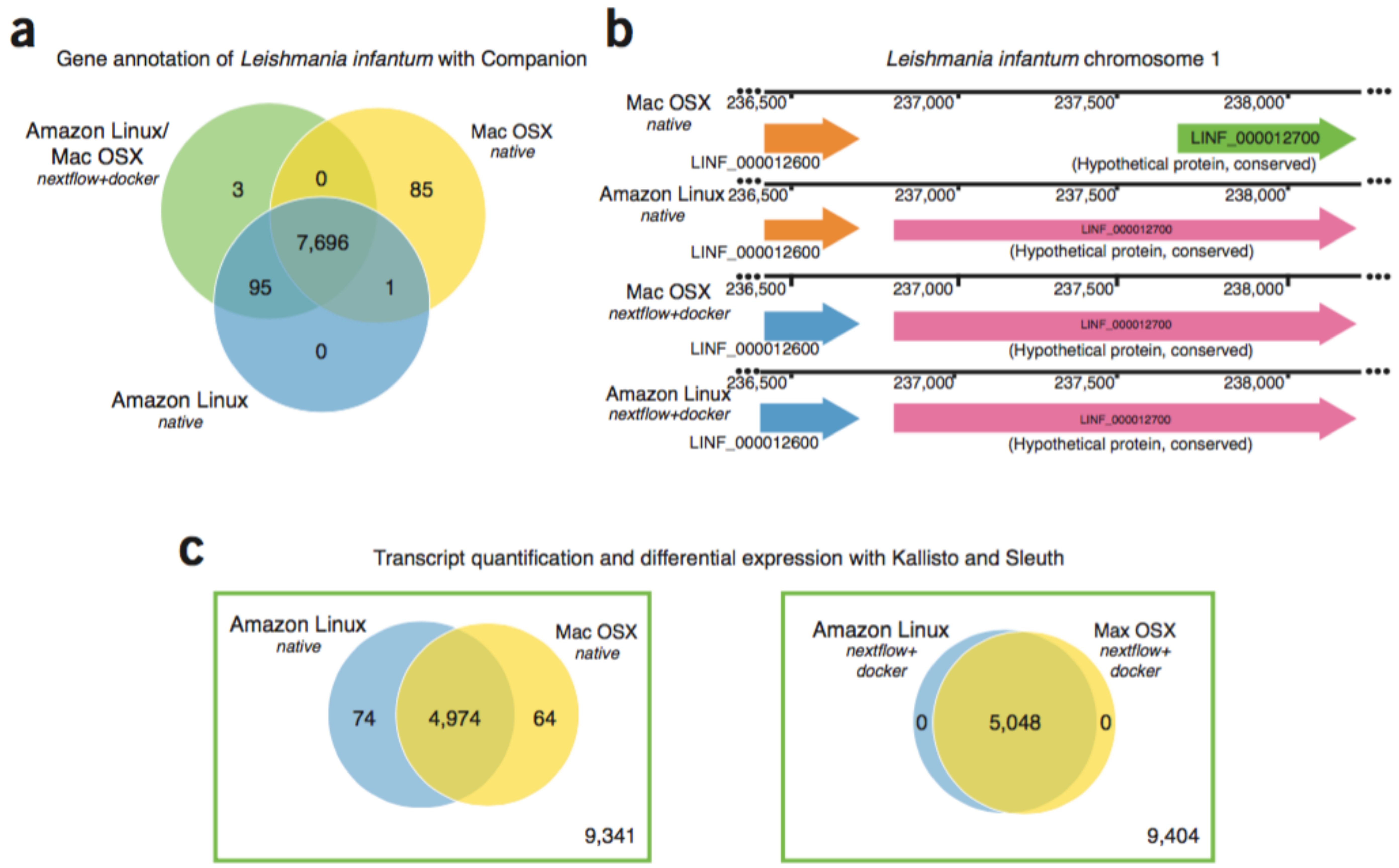
Among the tools selected 49% were deemed "difficult to install," and 28% *of the tools failed to be installed* due to problems in the implementation.



Nextflow enables reproducible computational workflows

Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo & Cedric Notredame

NATURE BIOTECHNOLOGY VOLUME 35 NUMBER 4 APRIL 2017



Comparison of the Companion pipeline annotation of *Leishmania infantum* genome executed across different platforms *

Platform	Amazon Linux	Debian Linux	Mac OSX
Number of chromosomes	36	36	36
Overall length (bp)	32.032.223	32.032.223	32.032.223
Number of genes	7.781	7.783	7.771
Gene density	236,64	236,64	236,32
Number of coding genes	7.580	7.580	7570
Average coding length (bp)	1.764	1.764	1.762
Number of genes with multiple CDS	113	113	111
Number of genes with known function	4.147	4.147	4.142
Number of t-RNAs	88	90	88

* Di Tommaso P, et al., *Nextflow enables computational reproducibility*, Nature Biotech, 2017

MAIN CHALLENGES

Scalability & parallelisation

Enable portability

Guarantee reproducibility

THE nextflow FUNDAMENTALS FOR SCALEABLE GENOMIC WORKFLOWS

Write code in any language

BASH THE BOURNE-AGAIN SHELL

R

Declarative parallelisation

Isolate dependencies with containers

Deploy everywhere

Google Cloud

GRID ENGINE

slurm workload manager

aws

Azure

kubernetes

Version control

Git

GitHub

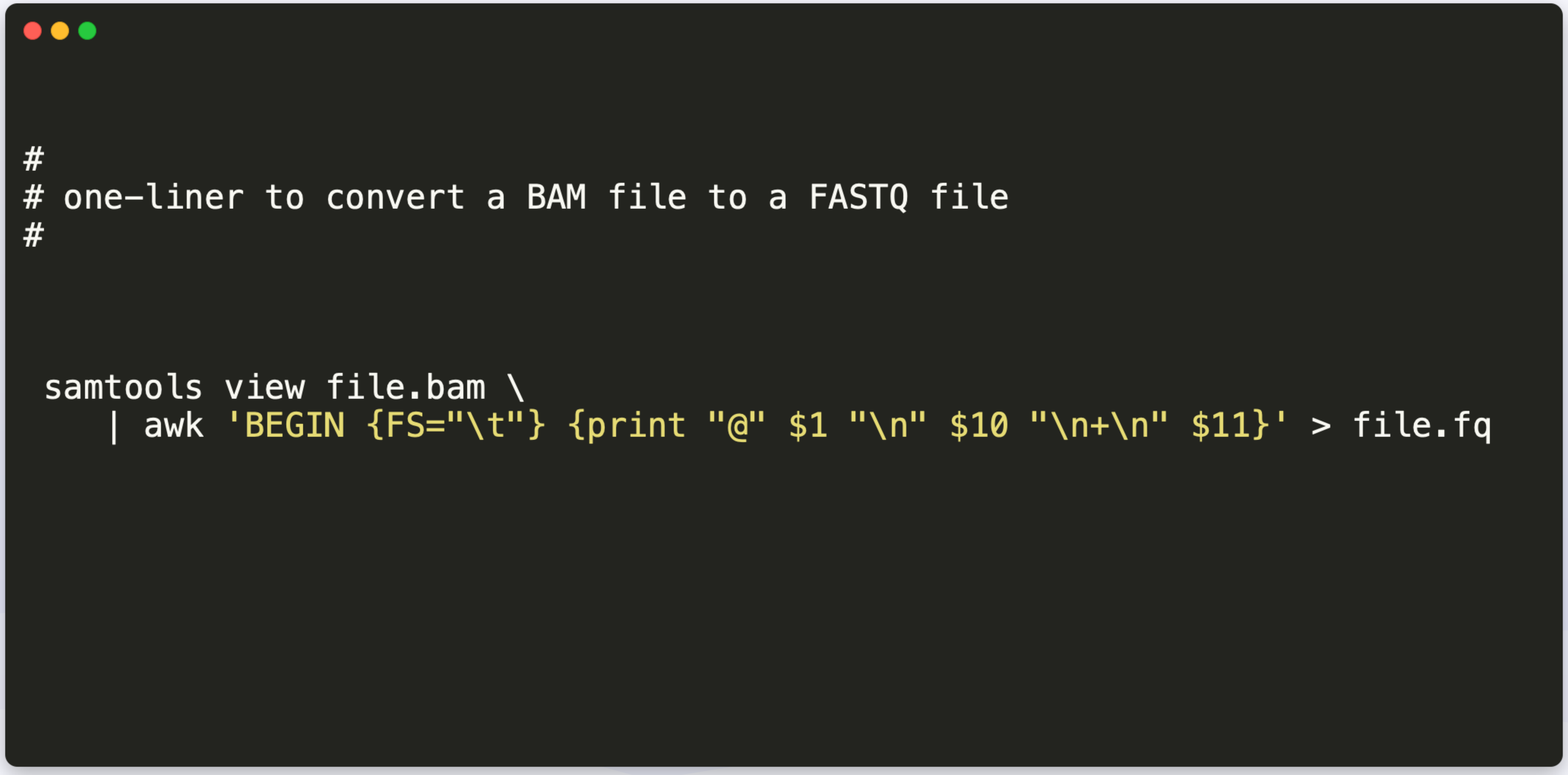
GitLab

Bitbucket

Open source

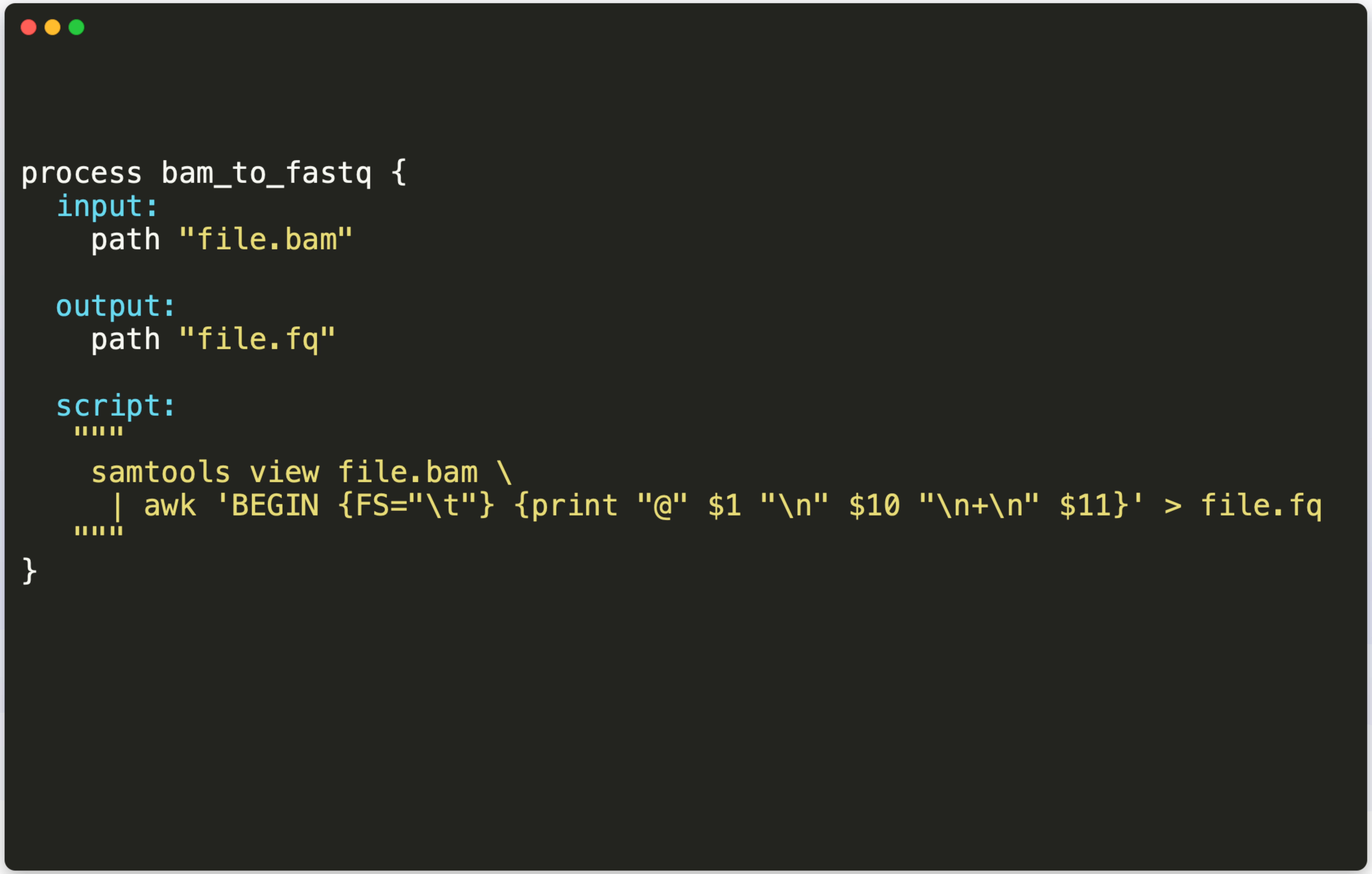
APACHE

SIMPLE EXAMPLE



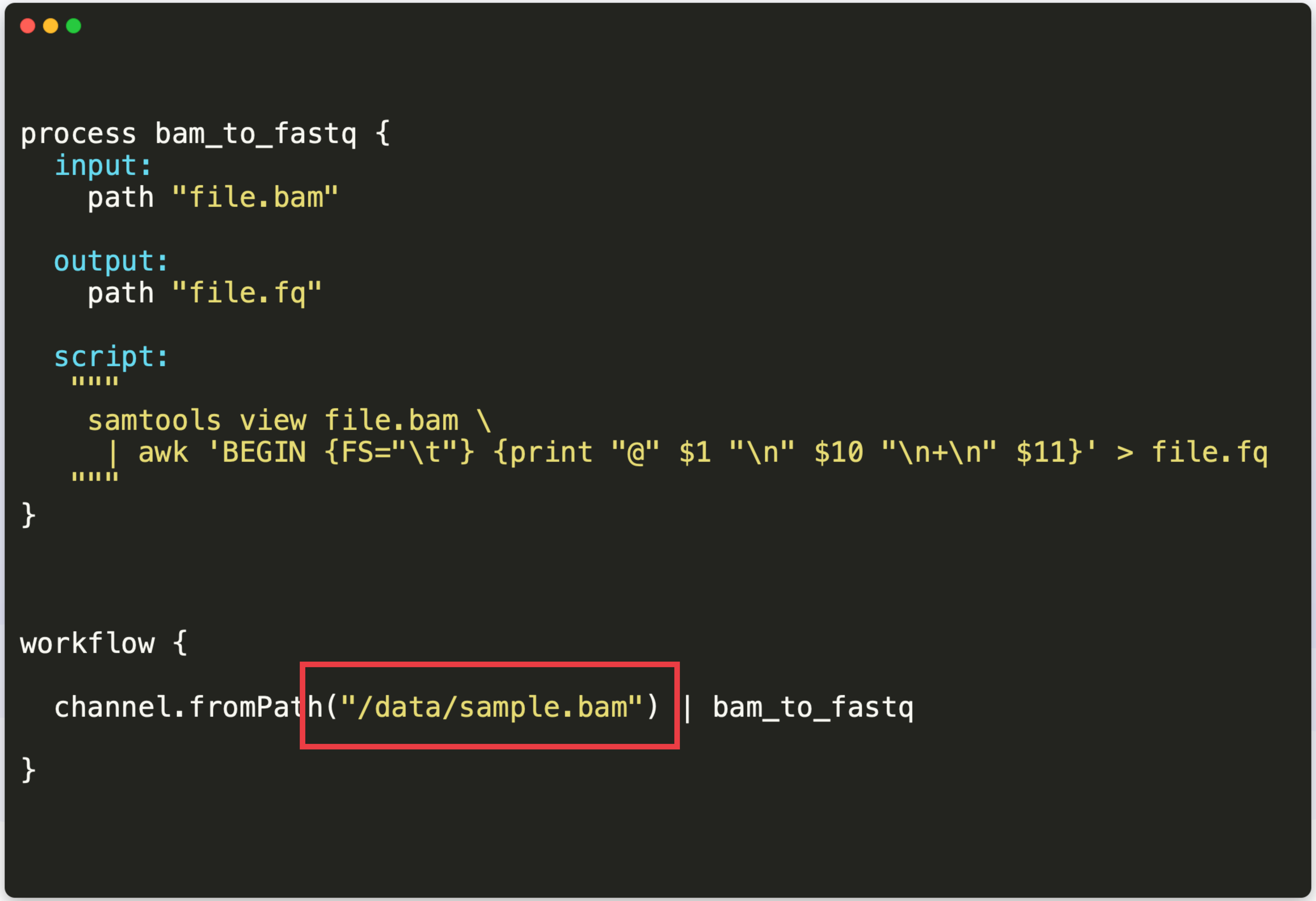
```
#
# one-liner to convert a BAM file to a FASTQ file
#


 samtools view file.bam \
    | awk 'BEGIN {FS="\t"} {print "@" $1 "\n" $10 "\n+\n" $11}' > file.fq
```

NEXTFLOW PROCESS



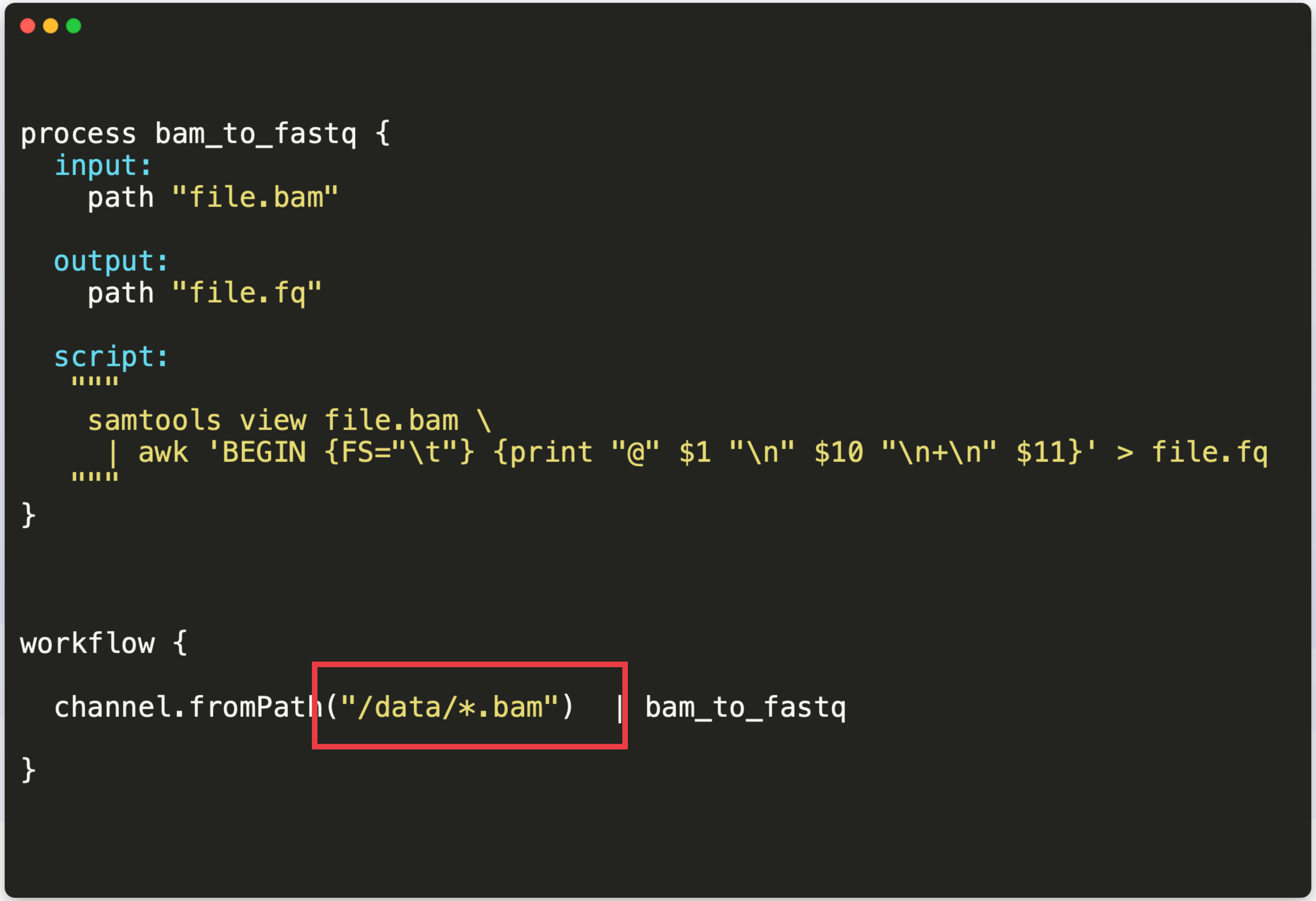
```
process bam_to_fastq {
  input:
    path "file.bam"

  output:
    path "file.fq"

  script:
   """
    samtools view file.bam \
      | awk 'BEGIN {FS="\t"} {print "@" $1 "\n" $10 "\n+\n" $11}' > file.fq
   """
}
```

NEXTFLOW WORKFLOW



```
process bam_to_fastq {
  input:
    path "file.bam"

  output:
    path "file.fq"

  script:
   """
    samtools view file.bam \
     | awk 'BEGIN {FS="\t"} {print "@" $1 "\n" $10 "\n+\n" $11}' > file.fq
   """
}



workflow {

  channel.fromPath("/data/sample.bam") | bam_to_fastq

}
```

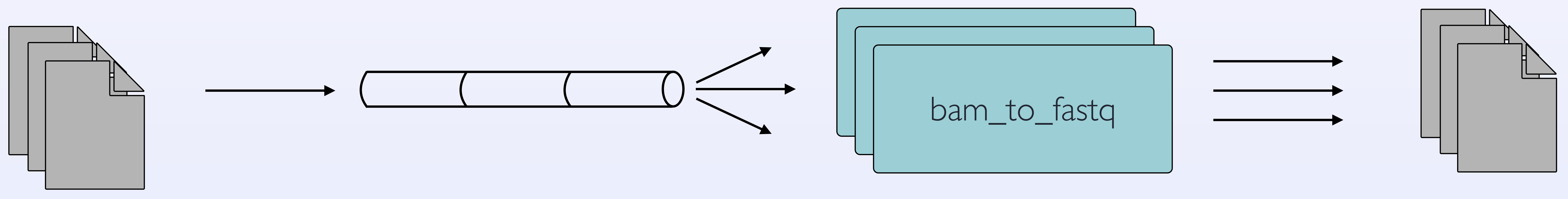
IMPLICIT PARALLELISM



```
process bam_to_fastq {
  input:
    path "file.bam"

  output:
    path "file.fq"

  script:
   """
    samtools view file.bam \
      | awk 'BEGIN {FS="\t"} {print "@" $1 "\n" $10 "\n+\n" $11}' > file.fq
   """
}



workflow {

  channel.fromPath("/data/*.bam")   | bam_to_fastq

}
```

IMPLICIT PARALLELISM

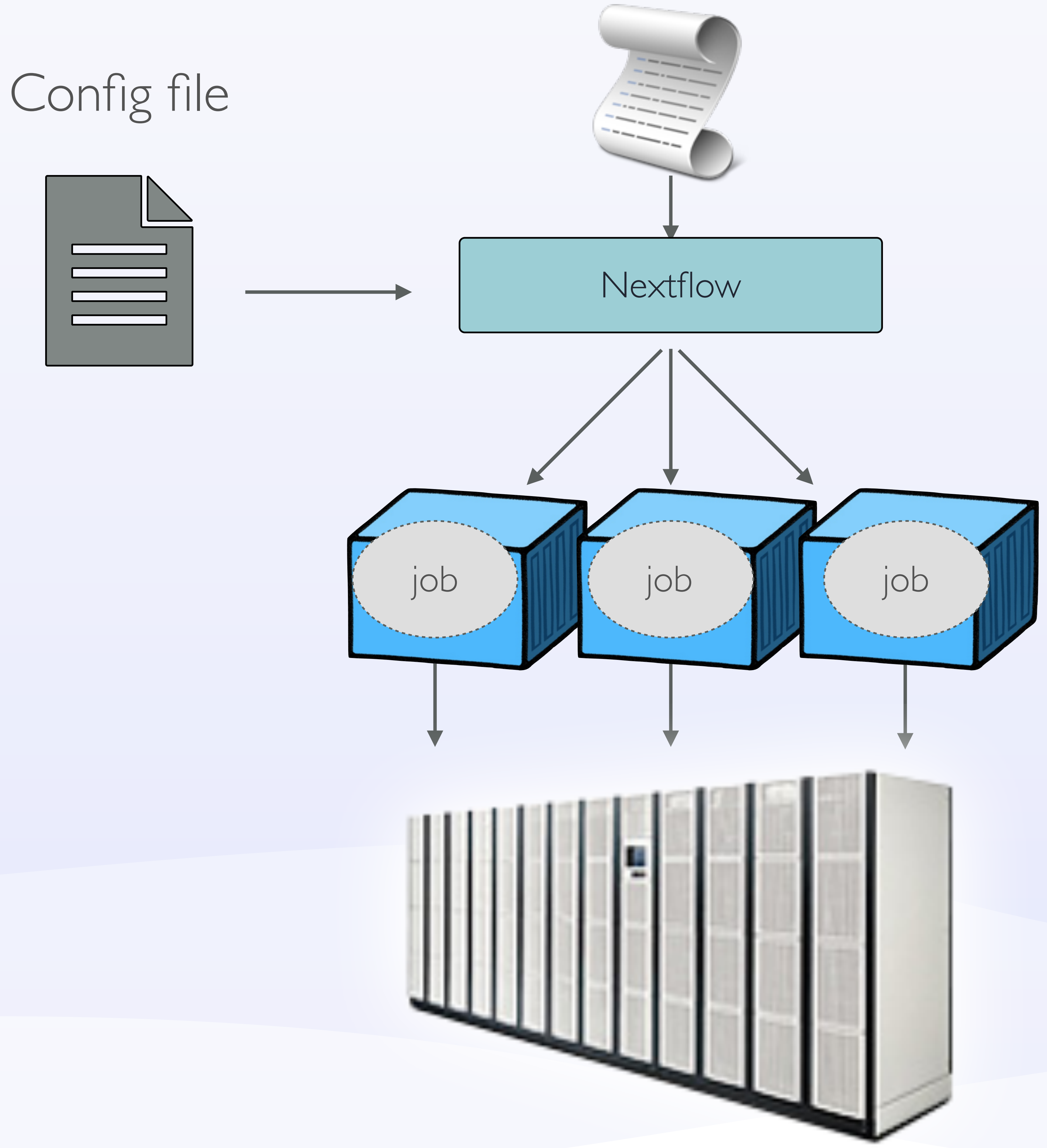
```
channel.fromPath("/data/*.bam")
```



BAM files

FASTQ files

CONTAINERISATION



- Nextflow envisioned the use of software containers to fix computational reproducibility
- Mar 2014, support for Docker
- Dec 2016, support for Singularity
- Jan 2020, support for Podman

MULTI-PLATFORM

slurm
workload manager

amazon
web services™

kubernetes

grid engine

Google Cloud

IBM Spectrum LSF

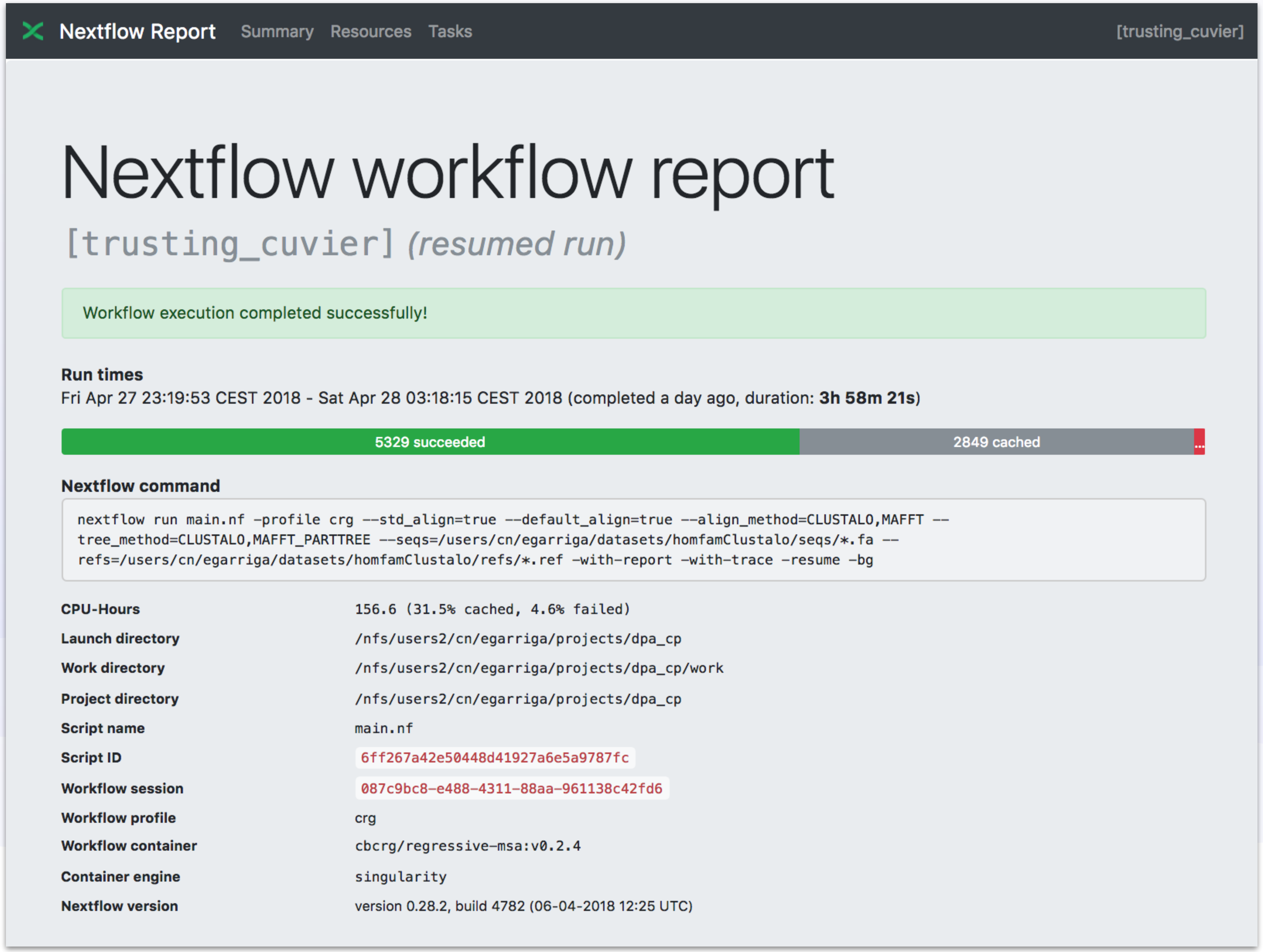
PBS Works™

Azure

HTCondor
High Throughput Computing

Charliecloud

EXECUTION REPORT



Nextflow workflow report

[trusting_cuvier] *(resumed run)*

Workflow execution completed successfully!

Run times

Fri Apr 27 23:19:53 CEST 2018 - Sat Apr 28 03:18:15 CEST 2018 (completed a day ago, duration: **3h 58m 21s**)

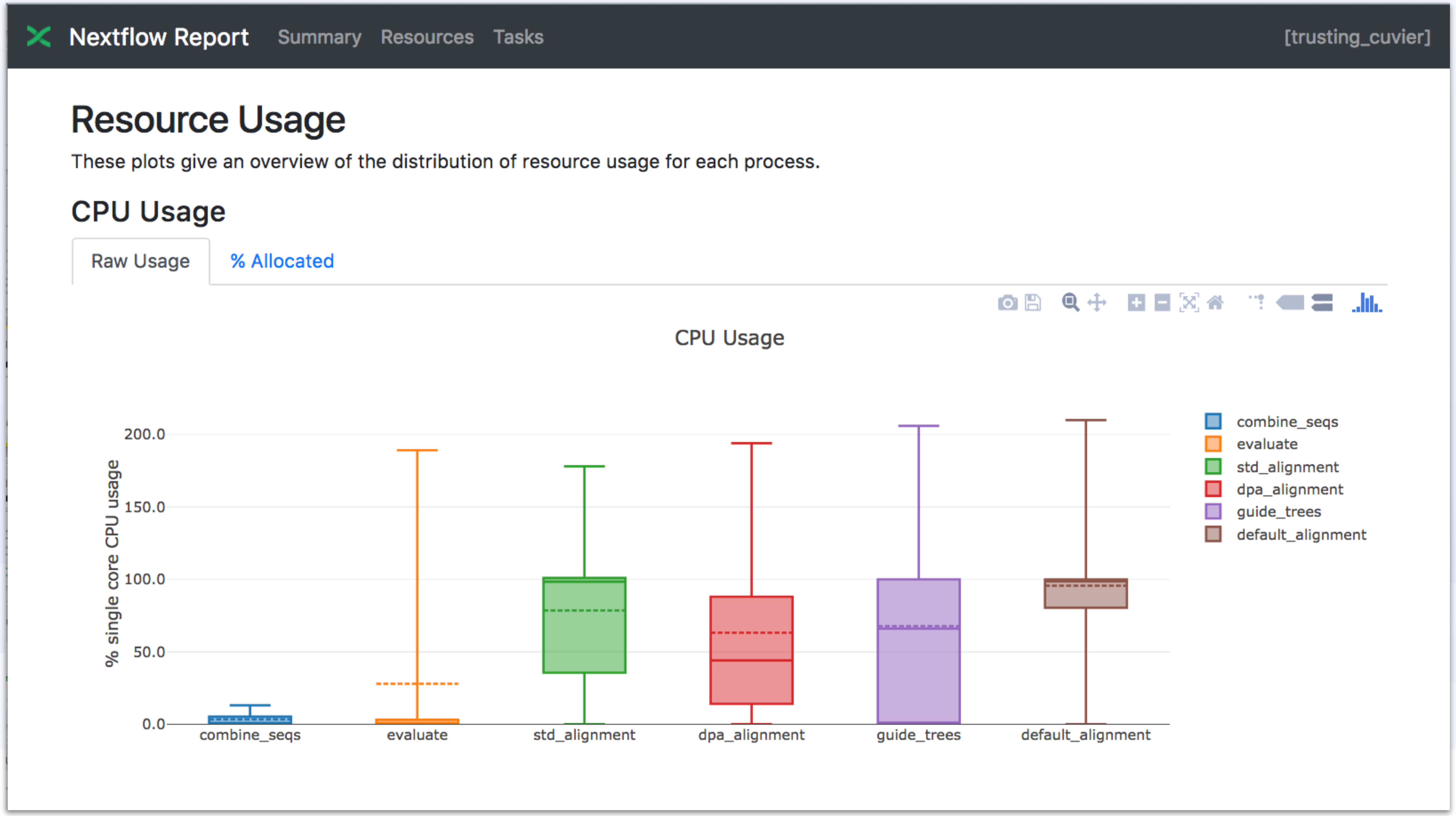
5329 succeeded 2849 cached ...

Nextflow command

```
nextflow run main.nf -profile crg --std_align=true --default_align=true --align_method=CLUSTALO,MAFFT --
tree_method=CLUSTALO,MAFFT_PARTTREE --seqs=/users/cn/egarriga/datasets/homfamClustalo/seqs/*.fa --
refs=/users/cn/egarriga/datasets/homfamClustalo/refs/*.ref -with-report -with-trace -resume -bg
```

CPU-Hours	156.6 (31.5% cached, 4.6% failed)
Launch directory	/nfs/users2/cn/egarriga/projects/dpa_cp
Work directory	/nfs/users2/cn/egarriga/projects/dpa_cp/work
Project directory	/nfs/users2/cn/egarriga/projects/dpa_cp
Script name	main.nf
Script ID	6ff267a42e50448d41927a6e5a9787fc
Workflow session	087c9bc8-e488-4311-88aa-961138c42fd6
Workflow profile	crg
Workflow container	cbcrg/regressive-msa:v0.2.4
Container engine	singularity
Nextflow version	version 0.28.2, build 4782 (06-04-2018 12:25 UTC)

EXECUTION REPORT

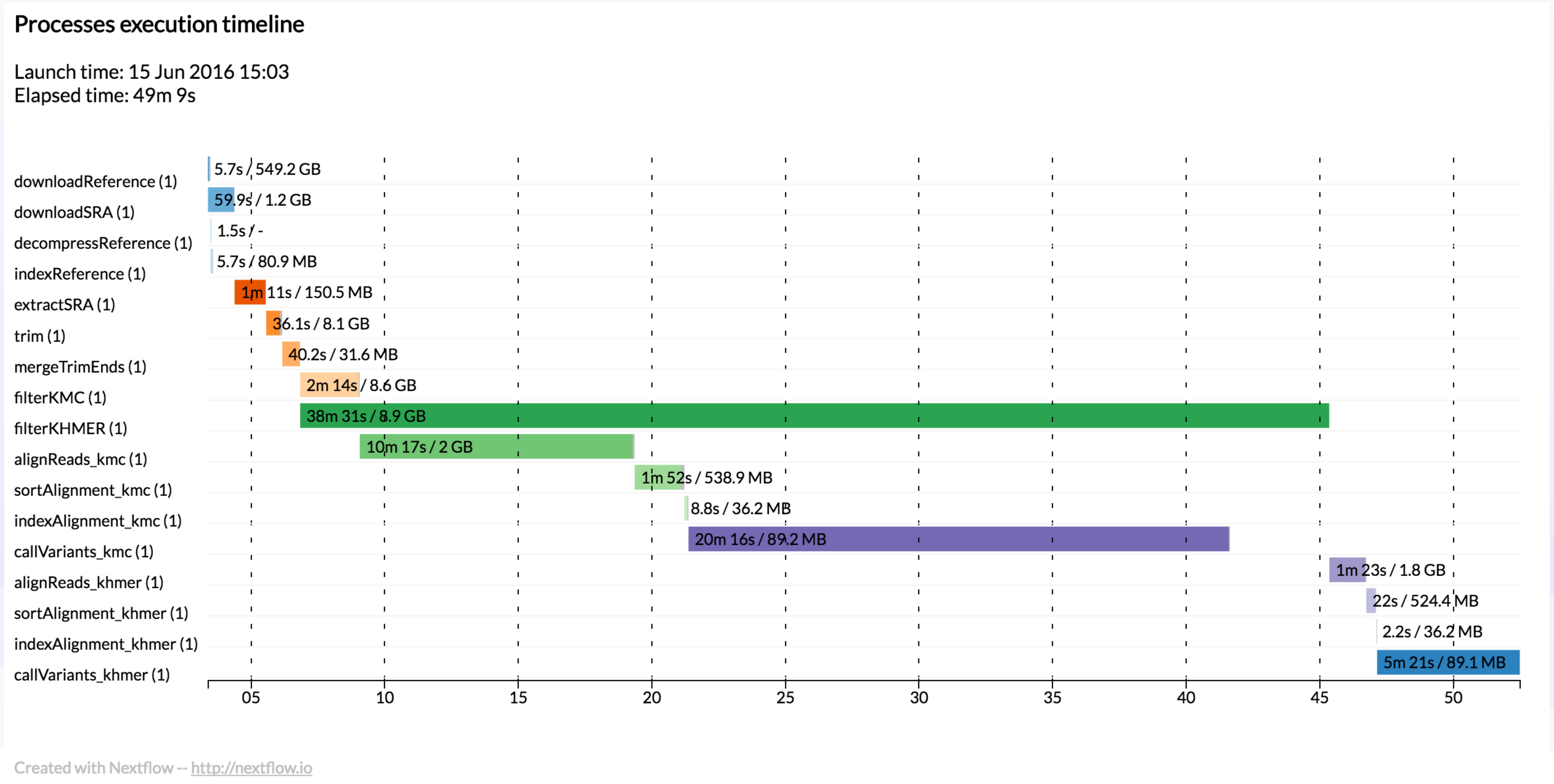


Resource Usage

These plots give an overview of the distribution of resource usage for each process.

CPU Usage

TIMELINE CHART

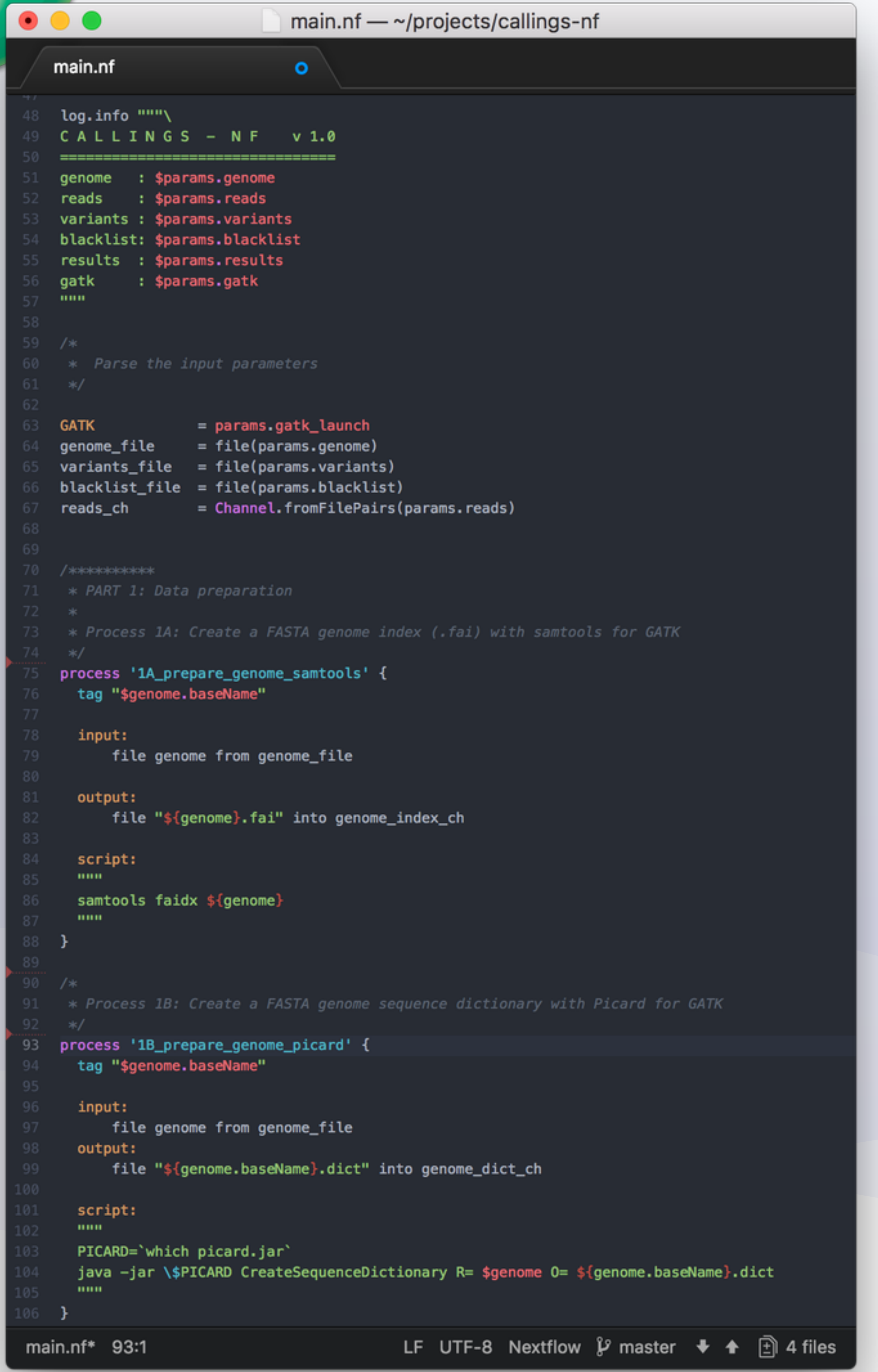


Processes execution timeline

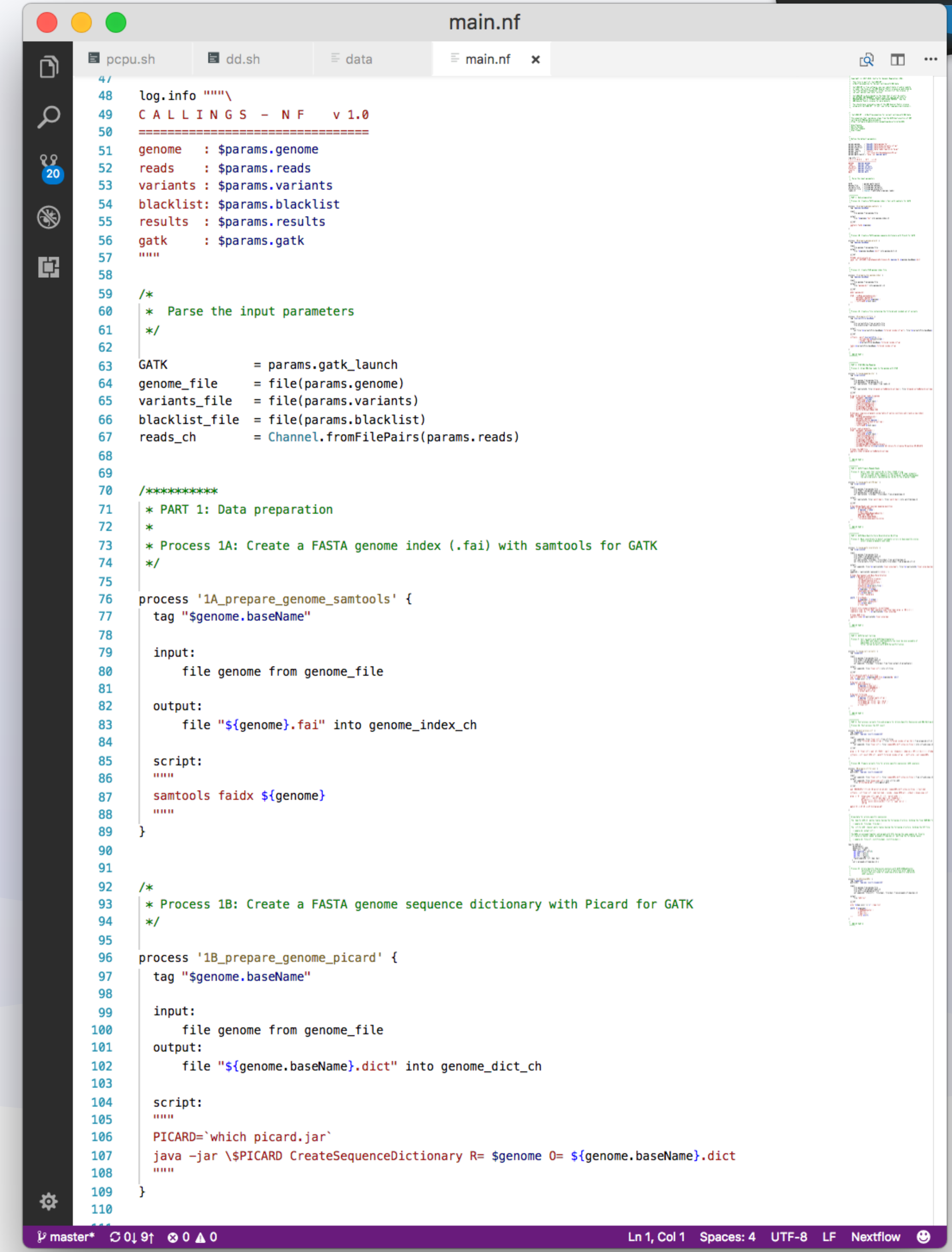
Launch time: 15 Jun 2016 15:03
Elapsed time: 49m 9s

Created with Nextflow -- http://nextflow.io

EDITORS



```
log.info """\
C A L L I N G S  -  N F    v 1.0
================================
genome   : $params.genome
reads    : $params.reads
variants : $params.variants
blacklist: $params.blacklist
results  : $params.results
gatk     : $params.gatk
"""

/*
 *  Parse the input parameters
 */

GATK              = params.gatk_launch
genome_file       = file(params.genome)
variants_file     = file(params.variants)
blacklist_file    = file(params.blacklist)
reads_ch          = Channel.fromFilePairs(params.reads)


/**********
 * PART 1: Data preparation
 *
 * Process 1A: Create a FASTA genome index (.fai) with samtools for GATK
 */
process '1A_prepare_genome_samtools' {
  tag "$genome.baseName"

  input:
      file genome from genome_file

  output:
      file "${genome}.fai" into genome_index_ch

  script:
  """
  samtools faidx ${genome}
  """
}

/*
 * Process 1B: Create a FASTA genome sequence dictionary with Picard for GATK
 */
process '1B_prepare_genome_picard' {
  tag "$genome.baseName"

  input:
      file genome from genome_file
  output:
      file "${genome.baseName}.dict" into genome_dict_ch

  script:
  """
  PICARD=`which picard.jar`
  java -jar \$PICARD CreateSequenceDictionary R= $genome O= ${genome.baseName}.dict
  """
}
```



```
log.info """\
C A L L I N G S  -  N F    v 1.0
================================
genome   : $params.genome
reads    : $params.reads
variants : $params.variants
blacklist: $params.blacklist
results  : $params.results
gatk     : $params.gatk
"""

/*
 *  Parse the input parameters
 */

GATK              = params.gatk_launch
genome_file       = file(params.genome)
variants_file     = file(params.variants)
blacklist_file    = file(params.blacklist)
reads_ch          = Channel.fromFilePairs(params.reads)


/**********
 * PART 1: Data preparation
 *
 * Process 1A: Create a FASTA genome index (.fai) with samtools for GATK
 */

process '1A_prepare_genome_samtools' {
  tag "$genome.baseName"

  input:
      file genome from genome_file

  output:
      file "${genome}.fai" into genome_index_ch

  script:
  """
  samtools faidx ${genome}
  """
}


/*
 * Process 1B: Create a FASTA genome sequence dictionary with Picard for GATK
 */

process '1B_prepare_genome_picard' {
  tag "$genome.baseName"

  input:
      file genome from genome_file
  output:
      file "${genome.baseName}.dict" into genome_dict_ch

  script:
  """
  PICARD=`which picard.jar`
  java -jar \$PICARD CreateSequenceDictionary R= $genome O= ${genome.baseName}.dict
  """
}
```

SUMMARY

- Data analysis reproducibility is hard and it's often underestimated.
- Nextflow does not provide a magic solution but enables best-practices and provides support for community and industry standards.
- It strictly separates the application logic from the configuration and deployment logic, enabling self-contained workflows.
- Applications can be easily deployed across different environment in a reproducible manner with a single command.
- The functional/reactive model allows applications to scale with ease.