RNAseq in Nextflow

Yuan Zhang
HPC Facility
October 12, 2023

Topics

- Nextflow basics
- A simple nextflow example
- RNAseq in nextflow

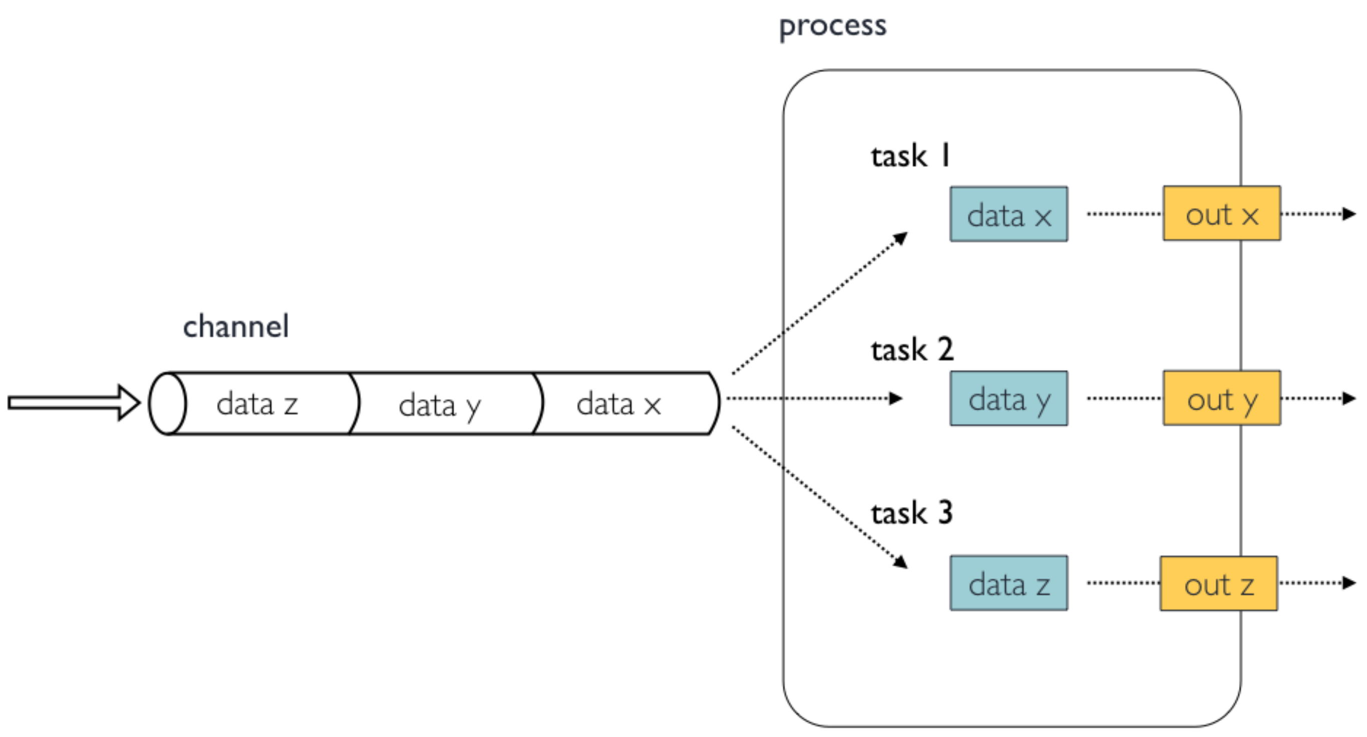
What is Nextflow?

It is a workflow management software which enables the writing of scalable and reproducible scientific workflows.

Scalability - It implements a Domain Specific Language (DSL) that simplifies the implementation and running of workflows on cloud or HPC infrastructure.

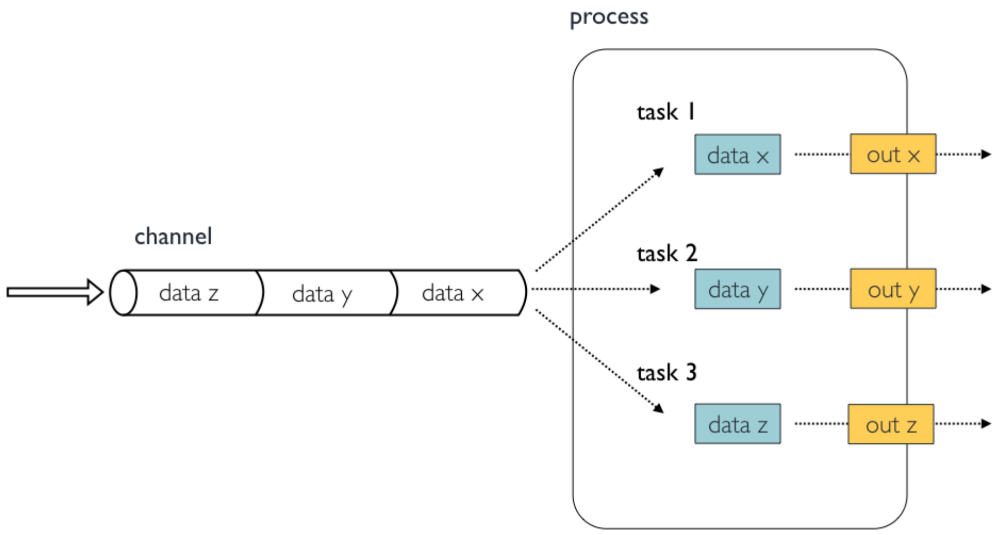
Reproducibility - It integrates various software packages and environment management system such as Docker, Singularity, and Conda.

Nextflow Diagram



Nextflow Basics - Process

- Describes a task to be run.
- The script can be written in any language (Bash, Perl, Python, R, etc.)
- One task for each input set.



Example - Process

```
process NUM_LINES {

    input: path read

    output: stdout

    script:
    """
    printf '${read}: '
    gunzip -c ${read} | wc -l
    """
}
```

Nextflow Basics - Channel

Stores the data as input or output of the processes. Processes communicate with each other through channels.

Value channel – can be used multiple times

- Channel.value

Queue channel – FIFO queues, used only once.

- Channel.of
- Channel.fromList
- Channel.fromPath
- Channel.fromFilePairs
- Channel.fromSRA

Example - Channel

Channel contains one data set

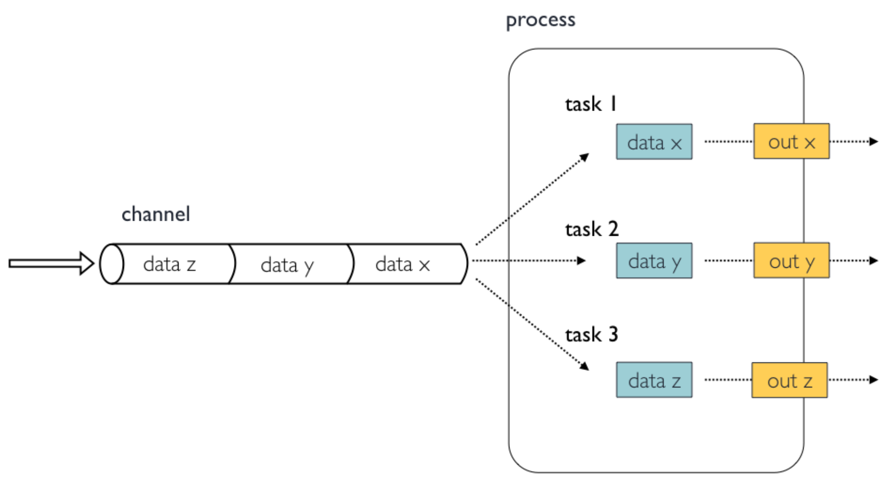
```
groovy> input_ch =
 Channel.fromPath("~/data/fastqs/SRR12005075.fastq.gz")
groovy> input_ch.view()

/home/gdhpcgroup/yxz006/data/fastqs/SRR12005075.fastq.gz
```

Channel contains multiple data set

```
groovy> input_ch =
 Channel.fromPath("~/data/fastqs/*.fastq.gz")
groovy> input_ch.view()
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005057.fastq.gz
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005058.fastq.gz
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005059.fastq.gz
...
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005073.fastq.gz
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005074.fastq.gz
/home/gdhpcgroup/yxz006/data/fastqs/SRR12005075.fastq.gz
```

Nextflow Basics - Workflow

- Defines a sequence of tasks that processes a set of data
- The execution environment (local, slurm, aws, etc.) is defined in config file.
- Can be run on local computer, HPC cluster, or cloud.



Example - Workflow

```
params.input =
”~/data/fastqs/SRR12005075.fastq.gz"

workflow {
    input_ch = Channel.fromPath(params.input)
    NUM_LINES(input_ch)
    NUM_LINES.out.view()
}
```

A Simple Workflow – wc.nf

```
nextflow.enable.dsl=2

params.input = ”~/data/fastqs/SRR12005075.fastq.gz"

workflow {
    input_ch = Channel.fromPath(params.input)
    NUM_LINES(input_ch)
    NUM_LINES.out.view()
}

process NUM_LINES {
    input: path read
    output: stdout
    script:
    """
    printf '${read}: '
    gunzip -c ${read} | wc -l
    """
}
```

Run the Workflow on Local Computer

```
[yxz006@r1pl-hpcf-log02 wc]$ ml nextflow

[yxz006@r1pl-hpcf-log02 wc]$ nextflow run wc.nf
N E X T F L O W  ~  version 22.10.6
Launching `wc.nf` [zen_lorenz] DSL2 - revision: 28933bf913
executor >  local (1)
[b1/b70dc6] process > NUM_LINES... [100%] 1 of 1 ✔
SRR12005075.fastq.gz: 71499804
```

Run the Workflow on HPC Cluster

```
[yxz006@r1pl-hpcf-log02 wc]$ cat nextflow.config
process.executor = 'slurm’

[yxz006@r1pl-hpcf-log02 wc]$ nextflow run wc.nf
N E X T F L O W  ~  version 22.10.6
Launching `wc.nf` [infallible_franklin] DSL2 - revision: 28933bf913
executor >  slurm (1)
[35/46dd0f] process > NUM_LINES (1) [100%] 1 of 1 ✔
SRR12005075.fastq.gz: 71499804
```

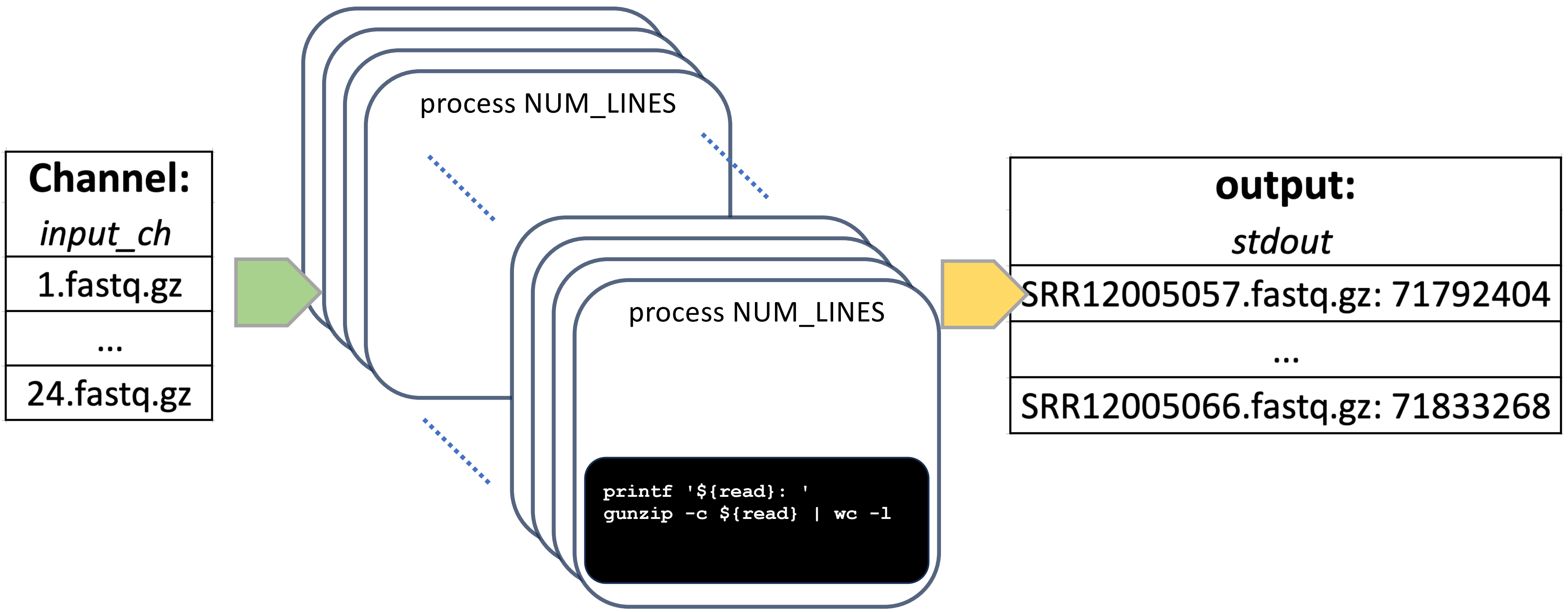
Change the Input Parameter

```
[yxz006@r1pl-hpcf-log02 wc]$ nextflow run wc.nf  -ansi-log false \
         --input="/home/gdhpcgroup/yxz006/data/fastqs/*.gz”

...
Launching `wc.nf` [suspicious_roentgen] DSL2 - revision: 0e75a1e416
[a7/24860b] Submitted process > NUM_LINES (1)
...
[a8/25498a] Submitted process > NUM_LINES (23)
[b5/56c3de] Submitted process > NUM_LINES (24)
SRR12005057.fastq.gz: 71792404
...
SRR12005065.fastq.gz: 71545878
...
```

The Resume Option

```
[yxz006@r1pl-hpcf-log02 wc]$ nextflow run wc.nf  -ansi-log false \
         --input="/home/gdhpcgroup/yxz006/data/fastqs/*.gz” \
         -resume
...
Launching `wc.nf` [extravagant_marconi] DSL2 - revision: 64b7f7c6e0
[a7/24860b] Cached process > NUM_LINES (1)
...
[a8/25498a] Cached process > NUM_LINES (23)
[b5/56c3de] Submitted process > NUM_LINES (24)
SRR12005057.fastq.gz: 71792404
...
SRR12005065.fastq.gz: 71545878
SRR12005066.fastq.gz: 71833268
```

Nextflow Process Flow Diagram



```
printf '${read}: '
gunzip -c ${read} | wc -l
```

A Simple Workflow – wc.nf

```
nextflow.enable.dsl=2

params.input = ”~/data/fastqs/SRR12005075.fastq.gz"

workflow {
    input_ch = Channel.fromPath(params.input)
    NUM_LINES(input_ch)
    NUM_LINES.out.view()
}

process NUM_LINES {
    input: path read
    output: stdout
    script:
    """
    printf '${read}: '
    gunzip -c ${read} | wc -l
    """
}
```

Matt’s RNAseq Pipeline

- `Input file`
  - `misc/MiiiceIiiiinSpaaaaaaaaaaaaace.csv`
- `Slurm scripts`
  - `sbatchCmds/getSRAData.sh`
  - `sbatchCmds/fastqc.sh`
  - `sbatchCmds/align_hisat.sh`
  - `sbatchCmds/featureCounts.sh`
- `Workflow`
  - `MainRNAseqAnalysis.Rmd`

Converting the Pipeline to Nextflow

- `misc/MiiiceIiiiinSpaaaaaaaaaaaaace.csv`
- `sbatchCmds/getSRAData.sh`
  - ➢ `getSRAData.nf`
- `sbatchCmds/fastqc.sh`
  - ➢ `fastqc.nf`
- `sbatchCmds/align_hisat.sh`
  - ➢ `align_hisat.nf`
- `sbatchCmds/featureCounts.sh`
  - ➢ `featureCounts.nf`
- `MainRNAseqAnalysis.Rmd`
  - ➢ `rnaseq.nf + MainRNAseqAnalysis.R + pca_de.nf`

Nextflow Script - getSRAData.nf

```
 process getSRAData {

      input:
      val (sra_number)

      output:
      path("*.gz"),          emit: fastq

      shell:
      '''
      module purge
      module load SRAToolkit/3.0.1
      prefetch -O ./ !{sra_number}

      fasterq-dump -e 4 -S -O ./ ./!{sra_number}/!{sra_number}.sra

      pigz ./!{sra_number}.fastq
      '''
}
```

Nextflow Config – nextflow.config

```
process {
    executor = 'slurm'
    cpus = 10
    time = '12hours'
    queue = 'general'
    memory = ''
    email = ''

    withName: getSRAData {
        cpus = 4
        time = '24hours'
    }

    withName: fastqc {
        cpus = 4
    }

}
```

Nextflow Script – rnaseq.nf

```
 params.samples = "$params.sampledir/MiiiceIiiiinSpaaaaaaaaaaaaace.csv"
 workflow RNASEQ {
     ch_samples = Channel.fromPath(params.samples)
     ch_samples.splitCsv ( header:true, sep:',' )
     .map { it.Run }
     .set {sra_array}

     getSRAData (sra_array)

     ch_fastq = getSRAData.out.fastq
     fastqc ( ch_fastq )

     align_hisat (ch_fastq)

     ch_bam = align_hisat.out.bam.collect()
     featureCounts (ch_bam)

     ...
}
```

Nextflow Script – main.nf

```
nextflow.enable.dsl = 2

// INPUT DIRECTORIES
params.projDir = "/home/gdhpcgroup/yxz006/training/2023/rnaseq/"
params.samples = "$params.projDir/misc/MiiiceIiiiinSpaaaaaaaaaaaaace.csv"
params.fastqs = "$params.projDir/input/fastqs"

//    OUTPUT DIRECTORIES
params.outdir = "$params.projDir/output"
params.fastqc = "$params.outdir/fastqc"
params.multiqc = "$params.outdir/multiqc"
params.align = "$params.outdir/align"
params.geneCounts = "$params.outdir/geneCounts"
params.figures = "$params.outdir/figures"
params.rdata = "$params.outdir/rdata"
params.de = "$params.outdir/de"
params.gsea = "$params.outdir/gsea"

// RUN WORKFLOW RNASEQ
include { RNASEQ } from './workflows/rnaseq'
workflow {
    RNASEQ ()
}
```

Nextflow Script – pca_de.nf

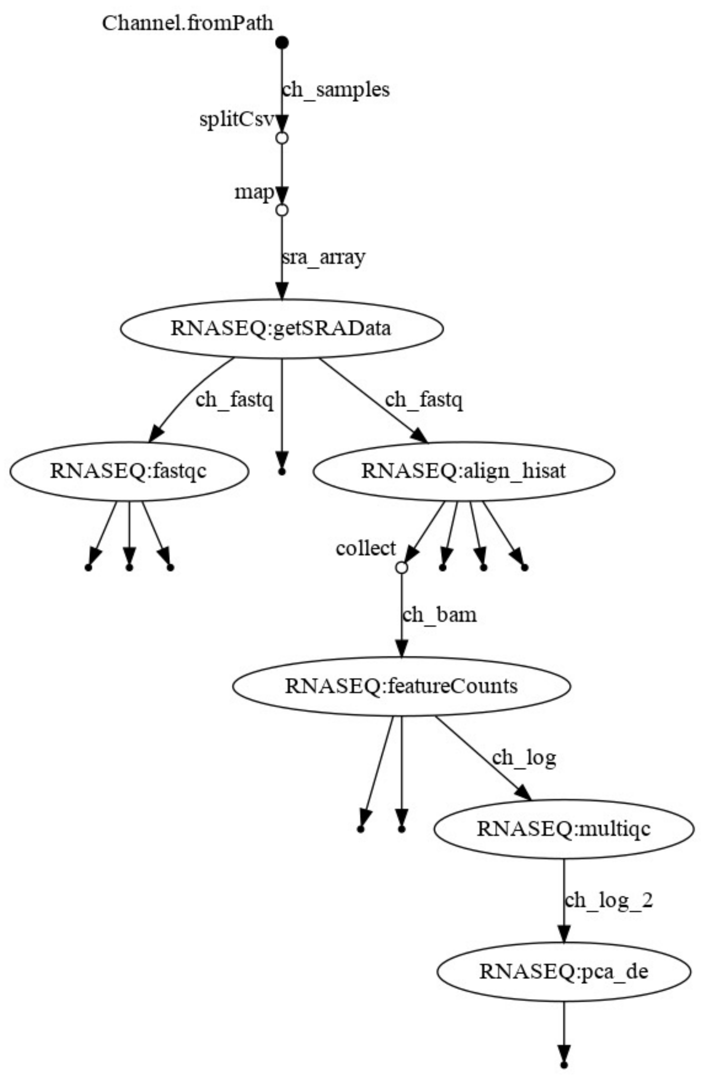
```
process pca_de {
    input:
    path ("*multiqc.log")

    shell:
    '''
    mkdir -p !{params.figures} !{params.rdata} !{params.de}
!{params.gsea}

    ml purge
    ml GCC/9.3.0 OpenMPI/4.0.3 R/4.2.2

    workDir=$(pwd)
    cd !{params.projDir}
    Rscript ./MainRNAseqAnalysis.R

    # save the slurm output
    cd ${workDir}
    cat .command.log > ${SLURM_JOB_ID}_pca_de.log
    '''
}
```

Run RNAseq in Nextflow

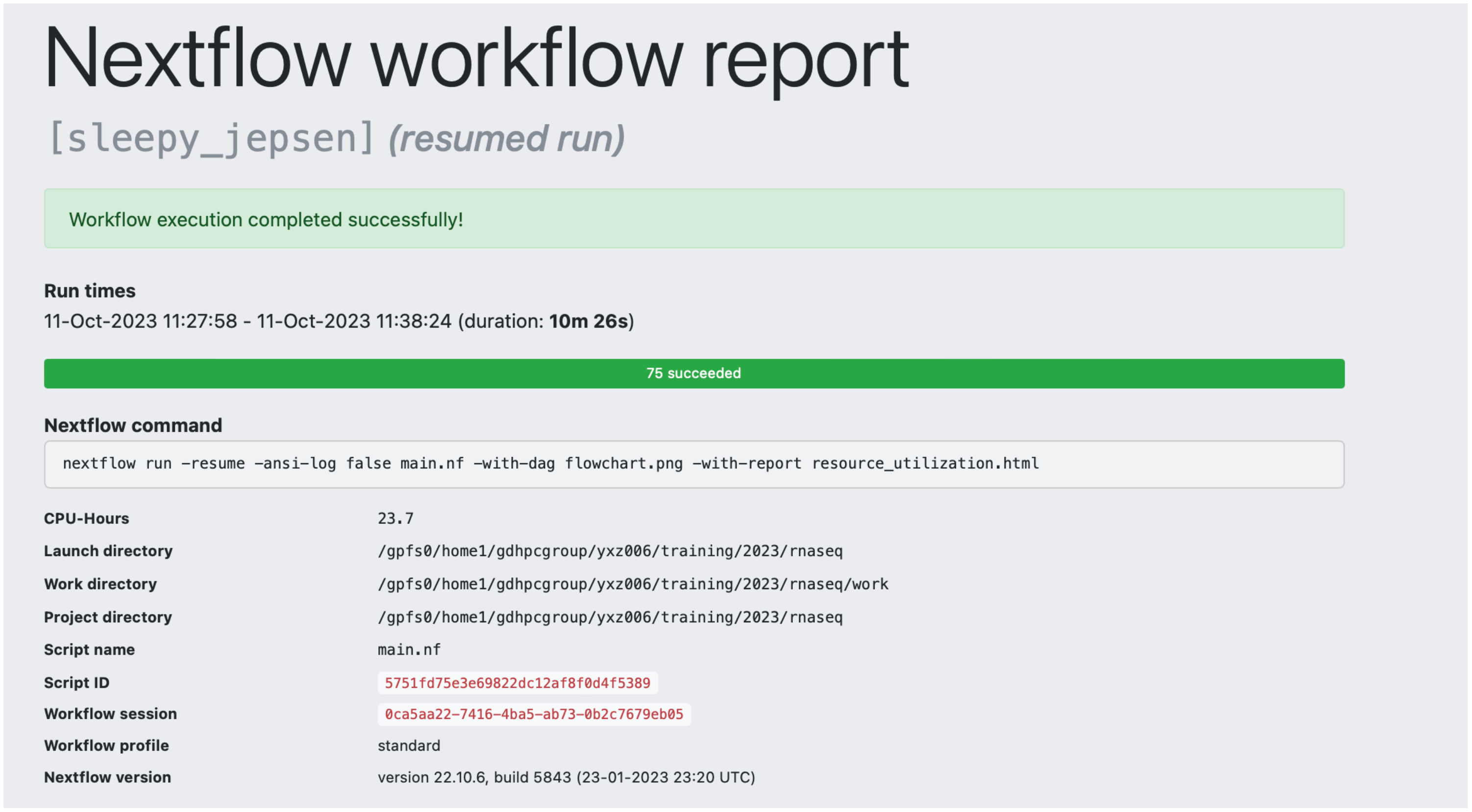
```
[yxz006@r1pl-hpcf-log02 rnaseq]$ nextflow run main.nf \
            -resume \
            -ansi-log false \
            -with-dag flowchart.png \
            -with-report resource_utilization.html
```

RNAseq pipeline in DAG Format

DAG – Directed acyclic graph



Resource Utilization Report



Nextflow workflow report

[sleepy_jepsen] *(resumed run)*

Workflow execution completed successfully!

Run times

11-Oct-2023 11:27:58 - 11-Oct-2023 11:38:24 (duration: **10m 26s**)

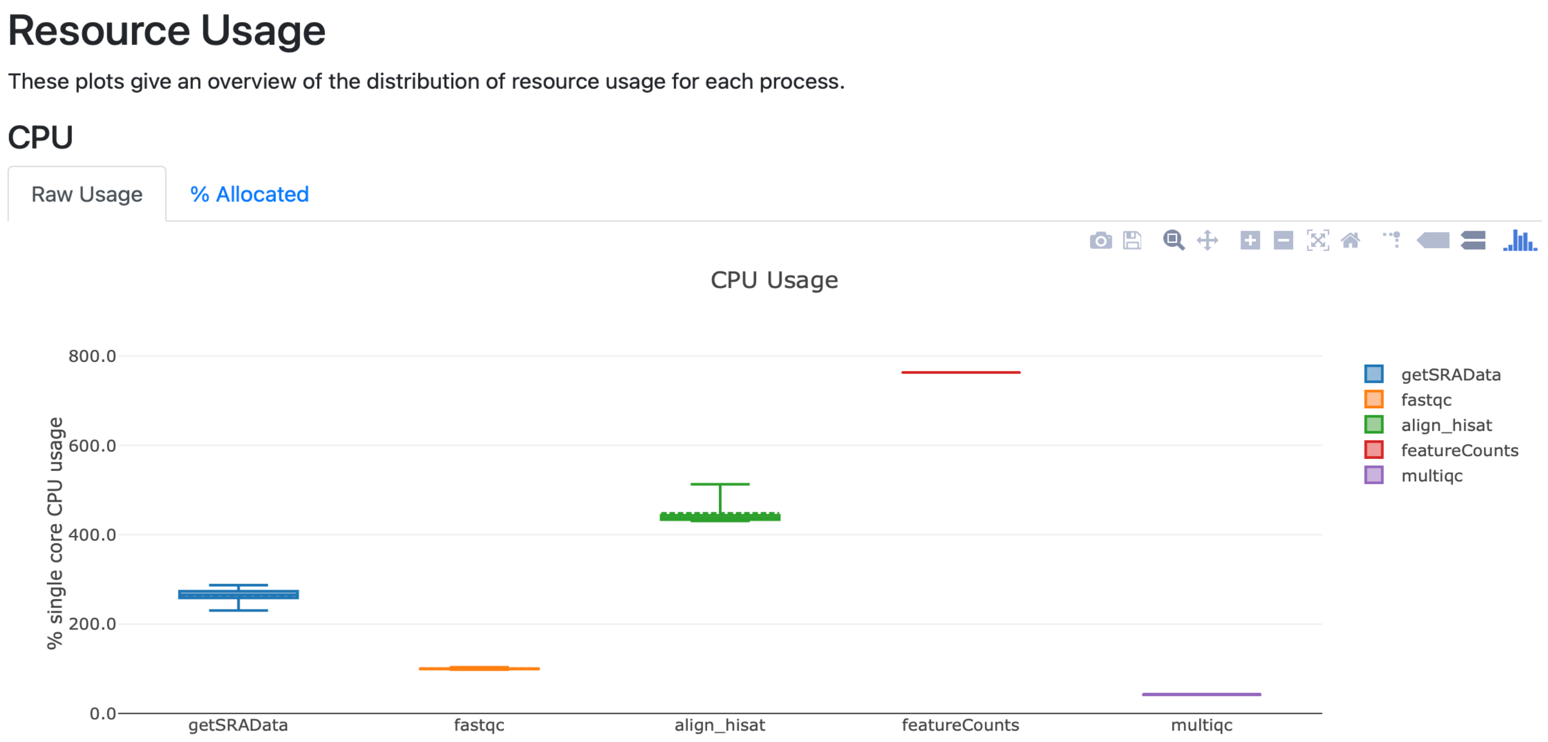
75 succeeded

Nextflow command

```
nextflow run -resume -ansi-log false main.nf -with-dag flowchart.png -with-report resource_utilization.html
```

CPU-Hours	23.7
Launch directory	/gpfs0/home1/gdhpcgroup/yxz006/training/2023/rnaseq
Work directory	/gpfs0/home1/gdhpcgroup/yxz006/training/2023/rnaseq/work
Project directory	/gpfs0/home1/gdhpcgroup/yxz006/training/2023/rnaseq
Script name	main.nf
Script ID	5751fd75e3e69822dc12af8f0d4f5389
Workflow session	0ca5aa22-7416-4ba5-ab73-0b2c7679eb05
Workflow profile	standard
Nextflow version	version 22.10.6, build 5843 (23-01-2023 23:20 UTC)

Resource Utilization Report - 2



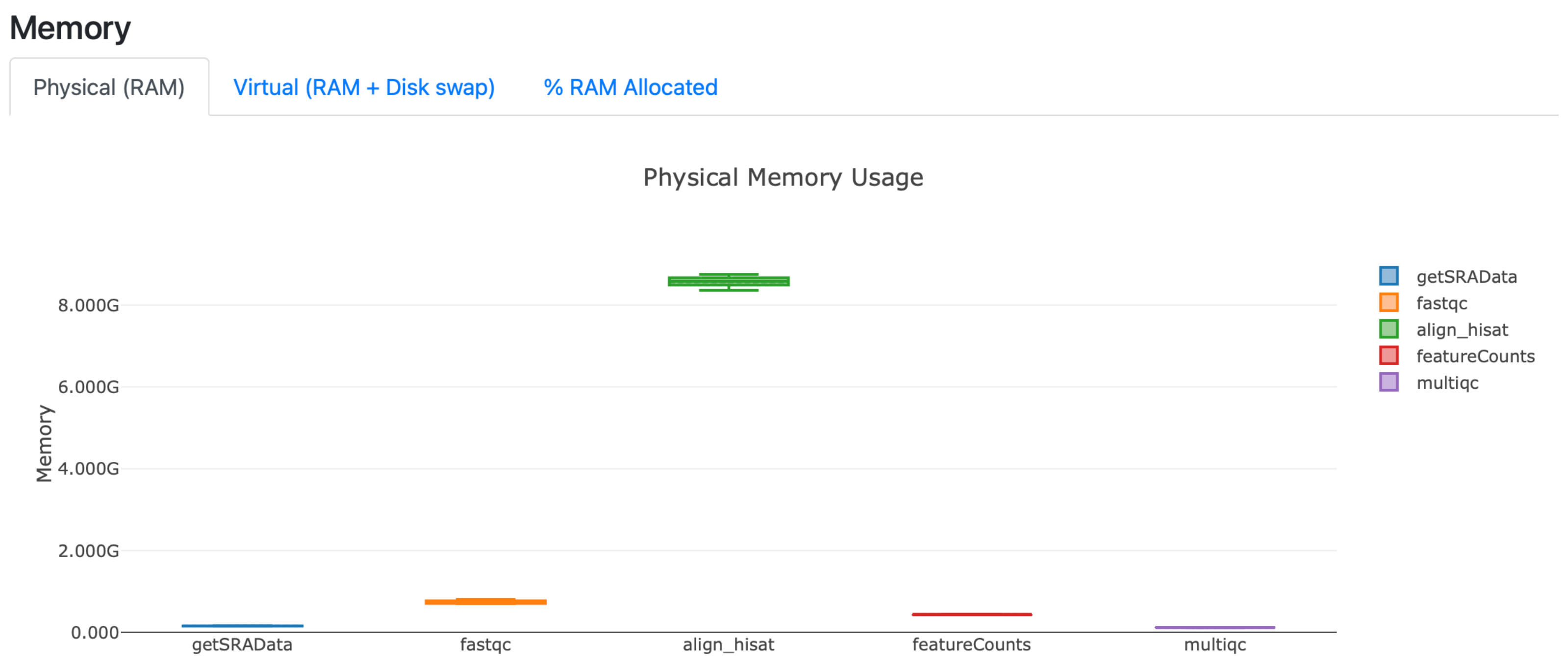
Resource Usage

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

CPU

Raw Usage | % Allocated

Resource Utilization Report - 3



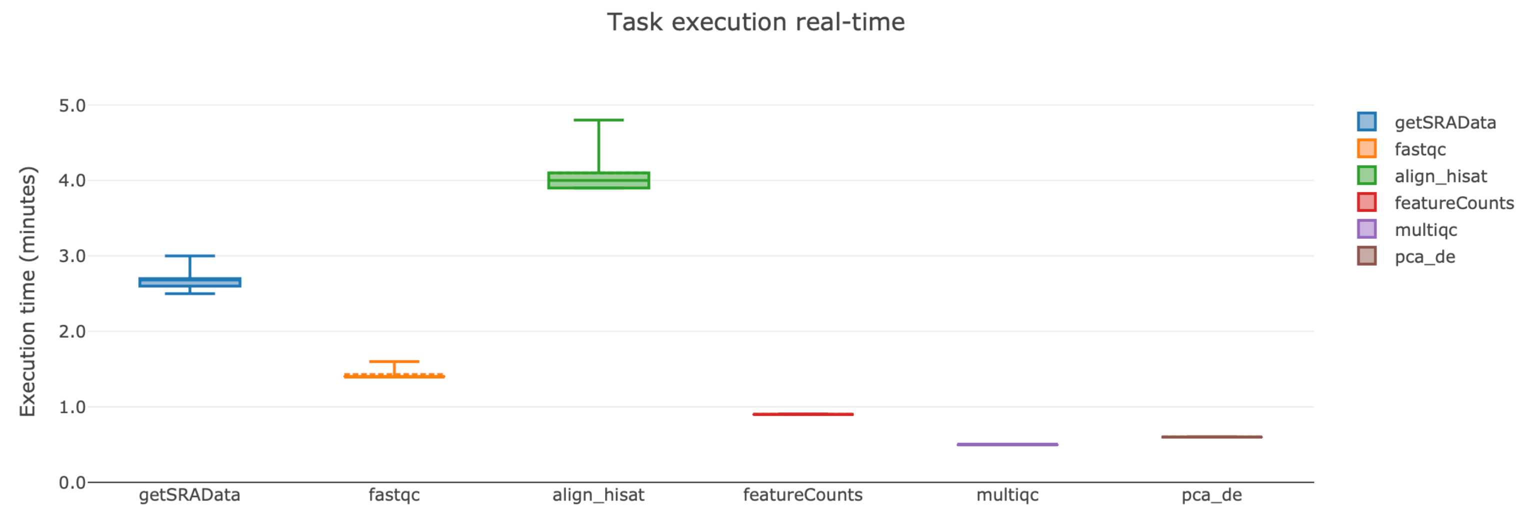
Memory

Physical (RAM) | Virtual (RAM + Disk swap) | % RAM Allocated

Resource Utilization Report - 4

Job Duration

Raw Usage | % Allocated



Benefits of Running RNAseq in Nextflow

Integrate various software packages

- QC, trimming, alignment, featurecounts, multiqc, differential expression analysis

Reproducible using environment management system

- Docker, Singularity, Conda

Portable

- local, Slurm, AWS

Continuous checkpoints

- Run the pipeline from the last successfully executed steps.

References:

- https://training.seqera.io/
- https://carpentries-incubator.github.io/workflows-nextflow/
- https://github.com/MVesuviusC/bioinformatics_meeting