DIFFRNA : A RNA-Seq analysis pipeline

Izem Mouhoubi Théo Roncalli Gustavo Magaña López

2021-11-29

Contents

1 Prerequisites **3**

1.1 Hardware and Operating System 3

1.2 BioPipes: a Biosphere-commons app 4

1.3 Main Tools 4

2 Dependencies **6**

2.1 Software 6

3 Workflow **7**

4 Execution **8**

5 Parameters **9**

6 Caveats **11**

Chapter 1

Prerequisites

1.1 Hardware and Operating System

The pipeline was developed and tested on Ubuntu 20.04.3 LTS on top of the (`GNU/Linux 5.4.0-88-generic x86_64`) kernel. The output of the commands `uname` and `neofetch` are provided to further detail our configuration.

```
$ uname -a
Linux machine329a7396-059f-41aa-94c7-4c41b4ec8290 5.4.0-88-generic #99-Ubuntu SMP Thu Sep 23 17:2

$ neofetch
            .-/+oossssoo+/-.               ubuntu@machine3f9ae3dc-6a4d-46b7-8131-04f00a1be146
        `:+ssssssssssssssssss+:`           --------------------------------------------------
      -+ssssssssssssssssssyyssss+-         OS: Ubuntu 20.04.3 LTS x86_64
    .ossssssssssssssssssdMMMNysssso.       Host: OpenStack Compute 18.2.1-1.el7
   /ssssssssssshdmmNNmmyNMMMMhssssss/      Kernel: 5.4.0-88-generic
  +ssssssssshmydMMMMMMMNddddyssssssss+     Uptime: 13 hours, 14 mins
 /sssssssshNMMMyhhyyyyhmNMMMNhssssssss/    Packages: 719 (dpkg), 4 (snap)
.ssssssssdMMMNhsssssssssshNMMMdssssssss.   Shell: bash 5.0.17
+sssshhhyNMMNyssssssssssssyNMMMysssssss+   Theme: Adwaita [GTK3]
ossyNMMMNyMMhsssssssssssssshmmmhssssssso   Icons: Adwaita [GTK3]
ossyNMMMNyMMhsssssssssssssshmmmhssssssso   Terminal: /dev/pts/0
+sssshhhyNMMNyssssssssssssyNMMMysssssss+   CPU: Intel (Haswell, no TSX, IBRS) (16) @ 2.294GHz
.ssssssssdMMMNhsssssssssshNMMMdssssssss.   GPU: 00:02.0 Cirrus Logic GD 5446
 /sssssssshNMMMyhhyyyyhdNMMMNhssssssss/    Memory: 635MiB / 64323MiB
  +sssssssssdmydMMMMMMMMddddyssssssss+
   /ssssssssssshdmNNNNmyNMMMMhssssss/
    .ossssssssssssssssssdMMMNysssso.
      -+sssssssssssssssssyyyssss+-
        `:+ssssssssssssssssss+:`
```

```
             .-/+oossssoo+/-.
```

This configuration was actually a virtual machine hosted on Biosphere's RAIN-Bio a cloud service maintained by the French Institue of Bioinformatics (*Institut Français de Bioinformatique*).

1.2 BioPipes: a Biosphere-commons app

The instance of the virtual machine we used is called *BioPipes*. It provides the most notable bioinformatics pipeline tools:

- nextflow
- snakemake
- cwltool

1.3 Main Tools

Their versions are specified to maximise reproducibility:

```
$ conda --version
conda 4.11.0
$ nextflow -v
nextflow version 21.10.0.5640
$ docker --version
Docker version 20.10.11, build dea9396
```

Detailed information about our development docker installation :

```
$ docker info
Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 4
 Server Version: 20.10.11
 Storage Driver: overlay2
  Backing Filesystem: xfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
 Network: bridge host ipvlan macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
Swarm: inactive
Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 7b11cfaabd73bb80907dd23182b9347b4245eb5d
runc version: v1.0.2-0-g52b36a2
init version: de40ad0
Security Options:
 apparmor
 seccomp
  Profile: default
Kernel Version: 5.4.0-88-generic
Operating System: Ubuntu 20.04.3 LTS
OSType: linux
Architecture: x86_64
CPUs: 16
Total Memory: 62.82GiB
Name: machine3f9ae3dc-6a4d-46b7-8131-04f00a1be146
ID: XT4Y:2HUL:HXEA:CDXV:ERC7:Z7JZ:YYRU:WZBT:ERCU:6GGA:OBZ6:QLXE
Docker Root Dir: /mnt/docker-data
Debug Mode: false
Registry: https://index.docker.io/v1/
Labels:
Experimental: false
Insecure Registries:
 127.0.0.0/8
Live Restore Enabled: false
```

Chapter 2

Dependencies

2.1 Software

The pipeline runs on nextflow a domain-specific language created to automate data-analysis pipelines whilst maximising reproducibility. Nextflow enables scientists to focus on their analyses, isolating different parts of the pipeline into processes whose dependencies can be dealt with using containers and virtual environments with technologies such as Docker, Singularity, and Anaconda.

The recommended way to install `nextflow` is via `conda`, using the environment file.

```
conda env create -f nextflow_conda_env.yml # will create an env called "nextflow"
conda activate nextflow
# You can edit the file at your choice, specially if the environment name conflicts
# with a preexisting conda env on your system
```

Docker should be installed as well:

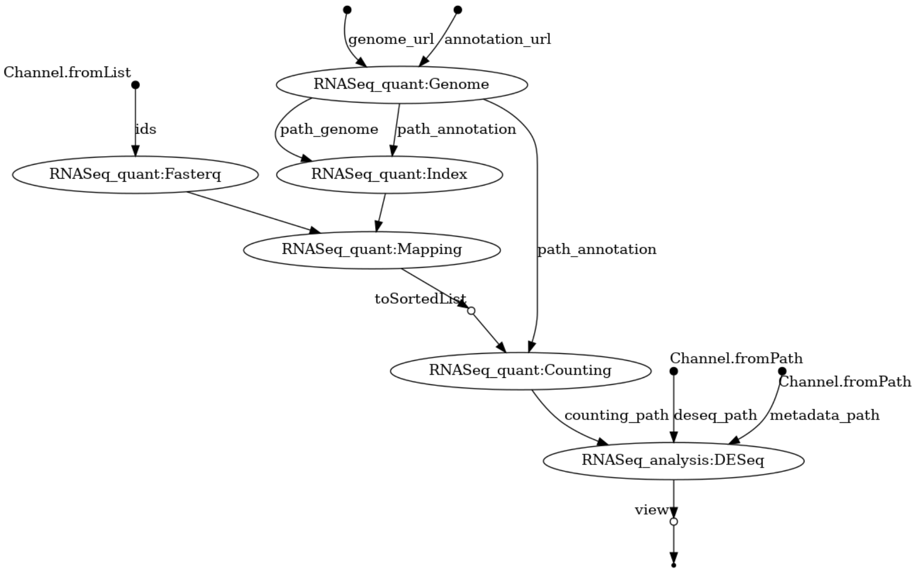
```
sudo apt install docker
```

Once nextflow is installed, it will automatically retrieve the docker images used within the pipeline.

Chapter 3

Workflow

Nextflow workflows should form a *DAG* (i.e. directed acyclic graph), which represents the flow of data through the different steps required to produce the final result.



This pipeline will generate a set of figures, representing differential gene expression analysis of RNA-Seq data.

Chapter 4

Execution

1. Clone the repo to your machine

```
git clone https://github.com/bio-TAGI/Hackathon.git
cd Hackathon
```

2. Create and activate the virtual environment

```
conda env create -f nextflow_conda_env.yml
conda activate nextflow
```

3. Run the wokflow with default parameters.

```
cd Nextflow
nextflow run main.nf
```

4. If you had to stop the workflow run, or if some error occurred, you can always resume the execution as follows:

```
nextflow run main.nf -resume
```

5. Specifying parameters from the command line

```
nextflow run main.nf --param1 value1\
--param2 value2\
--paramn valuen # these are generic names, not actual parameters for the pipeline
```

Chapter 5

Parameters

- `index_cpus` (number of cpus reserved for the genome indexation process. `default=14`)
- `mapping_cpus` (idem. for the mapping process, used to create BAM files. `default=14`)
- `counting_cpus` (idem. for the counting process. `default=7`)
- `mapping_memory` (RAM reserved for mapping. `default=50GB`)

If you already possess some of the files needed to execute the pipeline, you can specify them as follows:

- `reads` (path pointing to a directory containing the `fasterq` files)
- `genome` (path pointing to a directory containing the genome FASTA file)
- `index` (Répertoire contenant les fichiers d'index)
- `mapping` (Répertoire contenant les fichiers BAM)
- `counting` (Chemin d'accès entier au fichier de comptage – comprend le fichier lui-même)
- `metadata` (Chemin d'accès entier au fichier de métadonnées – comprend le fichier lui-même)

If unspecified, the pipeline will be executed using default values from the config file : nextflow.config. These too, can be tweaked and overriden:

- `ids` List of SRR accession number to fetch paired-end fastq files.
  - default=`['SRR628582', 'SRR628583', 'SRR628584', 'SRR628585', 'SRR628586', 'SRR628587', 'SRR628588', 'SRR628589']`
- `genome_url` URL to download the reference genome.
  - default `ftp://ftp.ensembl.org/pub/release-101/fasta/homo_sapiens/dna/Homo_sapiens.GRCh38.d`
- `annotation_url` URL to donwload the reference genome's annotation.
  - default `ftp://ftp.ensembl.org/pub/release-101/gtf/homo_sapiens/Homo_sapiens.GRCh38.101.chr`
- `sjdbOverhang` (a STAR-specific parameter. `default=99`)

  - For further information about this parameter, see this tutorial, or the STAR manual.

Chapter 6

Caveats

- A good internet connection is required. Retrieving `fastq` can be really slow and is thus a bottleneck.
- `fasterq-dump` will randomly segfault. At first we thought this was caused by connection problems, but running `ping` ruled this out. Apparently, the segfault is a known issue.
- The workflow will inevitably fail if you try building the genome's index on a machine with less than ~30 GB of RAM available.
  - As a general rule, tweak all parameters to reasonable values that fit your setup and needs. We don't know your hardware, you do ;)