Reproducible computational pipelines with Docker and Nextflow

Paolo Di Tommaso - Notredame Lab
Center for Genomic Regulation (CRG)
HPC Advisory Council - 22 March 2016, Lugano
@PaoloDiTommaso
Research software engineer
Comparative Bioinformatics,
Notredame Lab
Center for Genomic Regulation (CRG)
TWO MAJOR CHALLENGES IN COMPUTATIONAL BIOLOGY
COMPUTATIONAL
COMPUTATIONAL
REPRODUCIBILITY CRISIS

Announcement: 1
24 April 2013

Over the past year, Nature has and reproducibility of published
problems arise in laborat
erat scrutiny over it
information for other research

Assessing the validity and reproducibility

Lizzen A. Sugden, 1 Michael R. Tacker, 2 Yiannis A. Sany
Author information | Article notes | Copyright and License inform

Abstract

Motivation: Validation and reproducibility of results recent embarrassing incidents involving the irreproducible importance of this issue and the need for rigorous m

Results: Here, we describe an existing statistical meth
its utility for assessing the reproducibility of validated
me a statistical method for planning validation experimen
cidence limits, which, for a fixed total number of i
for the study.

Journals unite for reproducibility

Consensus on reporting principles aims to improve quality control in biomedical research and encourage public trust in science.

05 November 2014

Reproducibility, rigour, transparency and independent validation are cornerstones of the scientific method. Of course, just because a result is reproducible does not make it right, and just because it is not reproducible does not make it wrong. A transparent and rigorous approach, however, will almost always shine a light on issues of reproducibility. This light ensures that science moves forward, through independent verifications as well as the course corrections that come from refutations and the objective examination of the resulting data.
To replicate the result of a typical computational biology paper requires 280 hours!
WHAT'S WRONG WITH COMPUTATIONAL WORKFLOWS?
COMPLEXITY

• Dozens of dependencies (binary tools, compilers, libraries, system tools, etc)

• Experimental nature of academic SW tends to be difficult to install, configure and deploy

• Heterogeneous executing platforms and system architecture (laptop ➔ supercomputer)
CONTAINERS ARE THE THIRD BIG WAVE IN VIRTUALISATION TECHNOLOGY
BENEFITS

• Smaller images (~100MB)

• Fast instantiation time (~1 sec)

• Almost native performance

• Easy to build, publish, share and deploy

• Transparent build process
docker run -v $PWD:$PWD -w $PWD <image> cmd_x --opt file.txt
NAIVE APPROACH
SCALING OUT
CONTAINERS ORCHESTRATION

• Swarm
• Fleet
• Kubernetes
• Marathon
NOT THE RIGHT ANSWER FOR COMPUTATIONAL PIPELINES
SERVICES ORCHESTRATION ≠ TASKS SCHEDULING
OUR SOLUTION

Nextflow

Registry

Host file system
- A workflow framework that allows the same pipeline to run across different platforms
- Provides a high level parallelisation model
- Isolates task dependencies using containers
- It enables fast prototyping reusing any existing piece of software
process foo {
    input:
    val str from 'Hello'

    output:
    file 'my_file' into result

    script:
    ""
    echo $str world! > my_file
    ""
}
REACTIVE NETWORK
DATAFLOW

• Declarative computational model for concurrent processes

• Processes wait for data, when an input set is ready the process is executed

• They communicate by using dataflow variables i.e. async stream of data called channels

• Parallelisation and tasks dependencies are implicitly defined by process in/out declarations
PLATFORM AGNOSTIC

- DSL interpreter
- Dataflow
- Task dispatcher
- Executors
- POSIX processes
- qsub/bsub/...
SUPPORTED PLATFORMS
BATCH SCHEDULER

login node

nextflow

submit tasks

batch scheduler

NFS

cluster node

cluster node

cluster node
DISTRIBUTED MODE

Login node

NFS/Lustre

Job request

HPC cluster

nextflow cluster

nextflow driver

nextflow worker

nextflow worker

nextflow worker

cluster node

cluster node

#!/bin/bash
#$ -q <queue>
#$ -pe ompi <nodes>
#$ -l virtual_free=<mem>
mpirun nextflow run <your-pipeline> -with-mpi
USE CASE

- Deploying phylogenetic pipeline in BSC MareNostrum
- 500 lines of Nextflow scripting
- ~ 400k jobs
- 512 cores - 32 nodes
- ~ 50k cpu/h
process {
  executor = 'slurm'
  queue = 'cn-el6'
  memory = '10GB'
  cpus = 8
  time = '2h'
  container = 'your/image:latest'
}
PROS

• Dead easy deployment procedure
• Self-contained and precise controlled runtime
• Rapidly reproduce any former configuration
• Consistent results over time and across different platforms
CONS

• Requires a modern Linux kernel ($\geq 3.10$)

• Security concerns

• Containers/images cleanup
SHIFTER

• Container technology developed at NERSC

• Nextflow has built-in support for Shifter

• Experimental feature, under test

• It only requires an extra setting in the configuration file
WHAT ABOUT PERFORMANCE?
**Table 1** Mean execution times for pipelines and tasks with and without Docker. Time is expressed in minutes. The mean and the standard deviation were estimated from 10 separate runs. Slowdown represents the ratio of the mean execution time with Docker to the mean execution time when Docker was not used.

<table>
<thead>
<tr>
<th>Pipeline</th>
<th>Tasks</th>
<th>Mean task time</th>
<th>Mean execution time</th>
<th>Execution time std. deviation</th>
<th>Slowdown</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Native</td>
<td>Docker</td>
<td>Native</td>
<td>Docker</td>
</tr>
<tr>
<td>RNA-Seq</td>
<td>9</td>
<td>128.5</td>
<td>128.7</td>
<td>1,156.9</td>
<td>1,158.2</td>
</tr>
<tr>
<td>Variant call.</td>
<td>48</td>
<td>26.1</td>
<td>26.7</td>
<td>1,254.0</td>
<td>1,283.8</td>
</tr>
<tr>
<td>Piper</td>
<td>98</td>
<td>0.6</td>
<td>1.0</td>
<td>58.5</td>
<td>96.5</td>
</tr>
</tbody>
</table>
RNA-Seq toy pipeline

A proof of concept of a RNA-Seq pipeline intended to show Nextflow scripting and reproducibility capabilities.
$ nextflow run nextflow-io/rnatoy -with-docker
$ nextflow run nextflow-io/rnatoy -with-docker

NEXTFLOW ~ version 0.14.3
Pulling nextflow-io/rnatoy ...
downloaded from https://github.com/nextflow-io/rnatoy.git

Launching 'nextflow-io/rnatoy' - revision: 9c61bf5ac5 [master]
R N A T O Y   P I P E L I N E
=================================
genome             : /User/..../data/ggal_1_4885000_49020000.Ggal71.500bp.fa
annotat            : /User/..../data/ggal_1_4885000_49020000.bed.gff
pair1              : /User/..../data/*_1.fq
pair2              : /User/..../data/*_2.fq
[warm up] executor > local
[02/b08c28] Submitted process > buildIndex (ggal_1_4885000_49020000.Ggal71)
[ea/97d004] Submitted process > mapping (ggal_gut)
[98/16c9e5] Submitted process > mapping (ggal_liver)
[b5/38a0c7] Submitted process > makeTranscript (ggal_gut)
[00/e5efd6] Submitted process > makeTranscript (ggal_liver)
Saving: transcript_ggal_gut.gtf
Saving: transcript_ggal_liver.gtf
$ nextflow run nextflow-io/rnatoy -revision v1.0

NEXTFLOW ~ version 0.14.3
Launching 'nextflow-io/rnatoy' - revision: 0d0443d8f7 [v1.0]
R N A T O Y   P I P E L I N E
=================================
[35/cb611b] Submitted process > prepareTranscriptome (1)
[cd/239926] Submitted process > buildIndex (1)
[c6/f6488d] Submitted process > mapping (2)
[bc/b3ea76] Submitted process > mapping (1)
[f4/8d4628] Submitted process > makeTranscript (1)
[eb/92db7f] Submitted process > makeTranscript (2)
Saving: transcript_ggal_alpha.gtf
Saving: transcript_ggal_beta.gtf
$ vim nextflow.config

process {
    executor = 'slurm'
    memory = 10.GB
    cpus = 32
}

WHO IS USING NEXTFLOW?
WHO IS USING NEXTFLOW?

Writing **reproducible** and **scalable** bioinformatics pipelines using nextflow, docker and github

Matthieu Foll
Nov. 12th 2015

International Agency for Research on Cancer
Lyon, France
CONCLUSION

• Containers are a game-changer for computational workflows packaging and deployment

• Nextflow is a reactive/functional framework for computational workflows.

• Docker + Nextflow = Reproducible self-contained pipelines.
ACKNOWLEDGMENT

Evan Floden, CRG
Emilio Palumbo, CRG
Maria Chatzou, CRG
Cedric Notredame, CRG
THANKS
LINKS

project home
http://nextflow.io

GitHub repository
http://github.com/nextflow-io/nextflow

Docker benchmark
https://peerj.com/articles/1273/

Docker-Univa white paper
http://www.nextflow.io/misc/Univa-Docker-Whitepaper_FINAL.pdf