Workflow-Based TSS Prediction

Chong Lu,
Thomas Dutschmann,
Minyue Qi

SS2017
Eberhard Karls Universität Tübingen
Data Management of Quantitative Biology

Content

- TSS prediction
- Workflow management system
- Container system
- Project architecture
- Obstacles during the implementation
- Unsolved problem

TSS prediction

- TSS: Transcription start sites
- Main goal: We want to define where in a genome a transcription starts, as exact as possible
- Reads from wet labs are mapped onto a known genome
- A tool developed in Tübingen, **`TSSpredator`**, is able to detect TSS from wiggle input files
- Hence, we must
  - Map our reads onto the genome
  - Translate the mapping files to wiggle files
  - Run TSSpredator with the generated wiggle files

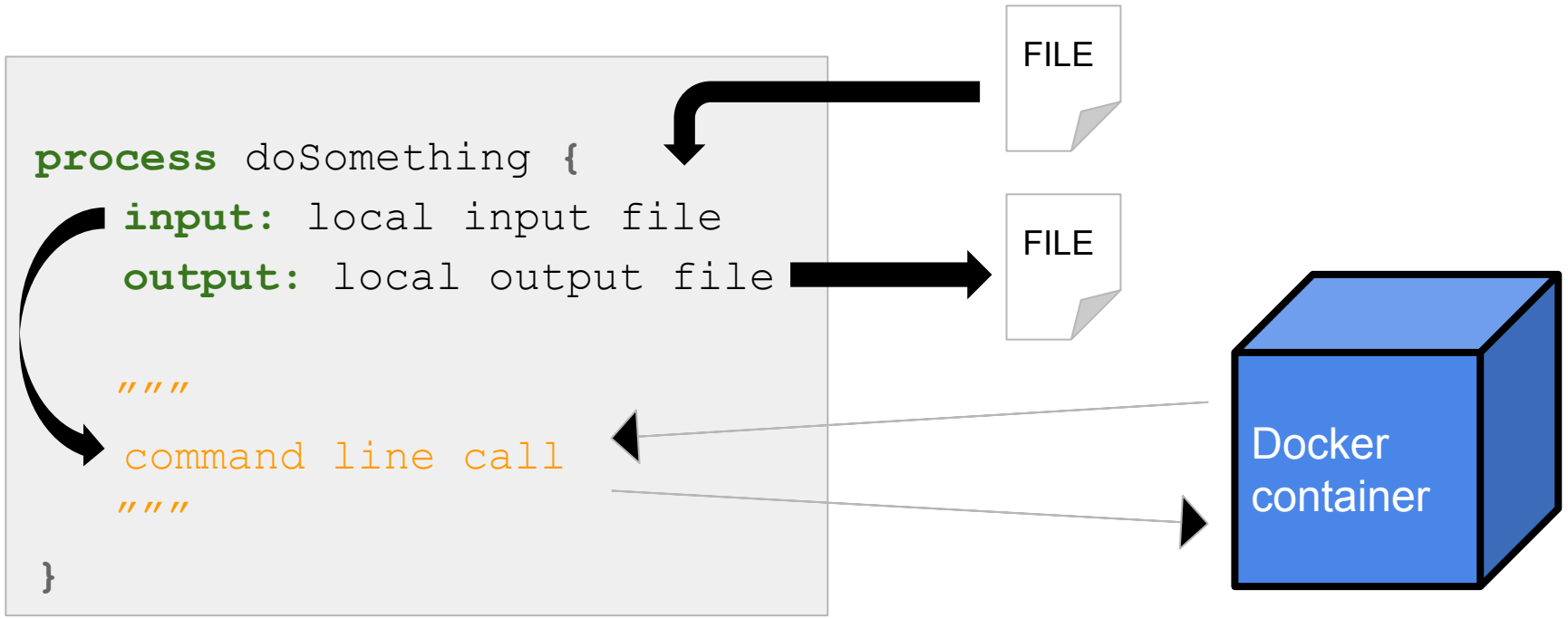
Workflow management system

- A workflow language is a language to glue together command line tools
- Calling applications by command line can be automated
- For our project, we had to choose one out of three currently available workflow languages

- We chose Nextflow, because...

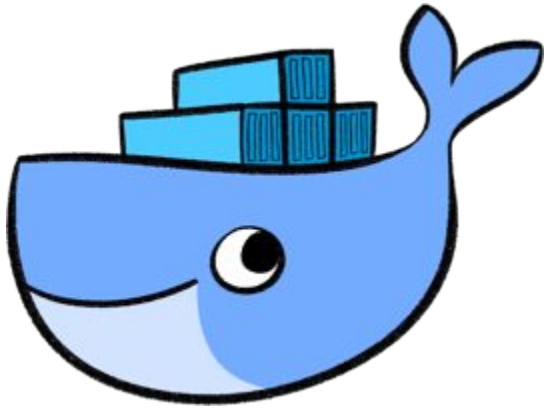
Workflow management system

- In our workflow, we call separate applications which are shipped inside their own Docker containers
- Conceptually, our workflow processes work like this:



Container

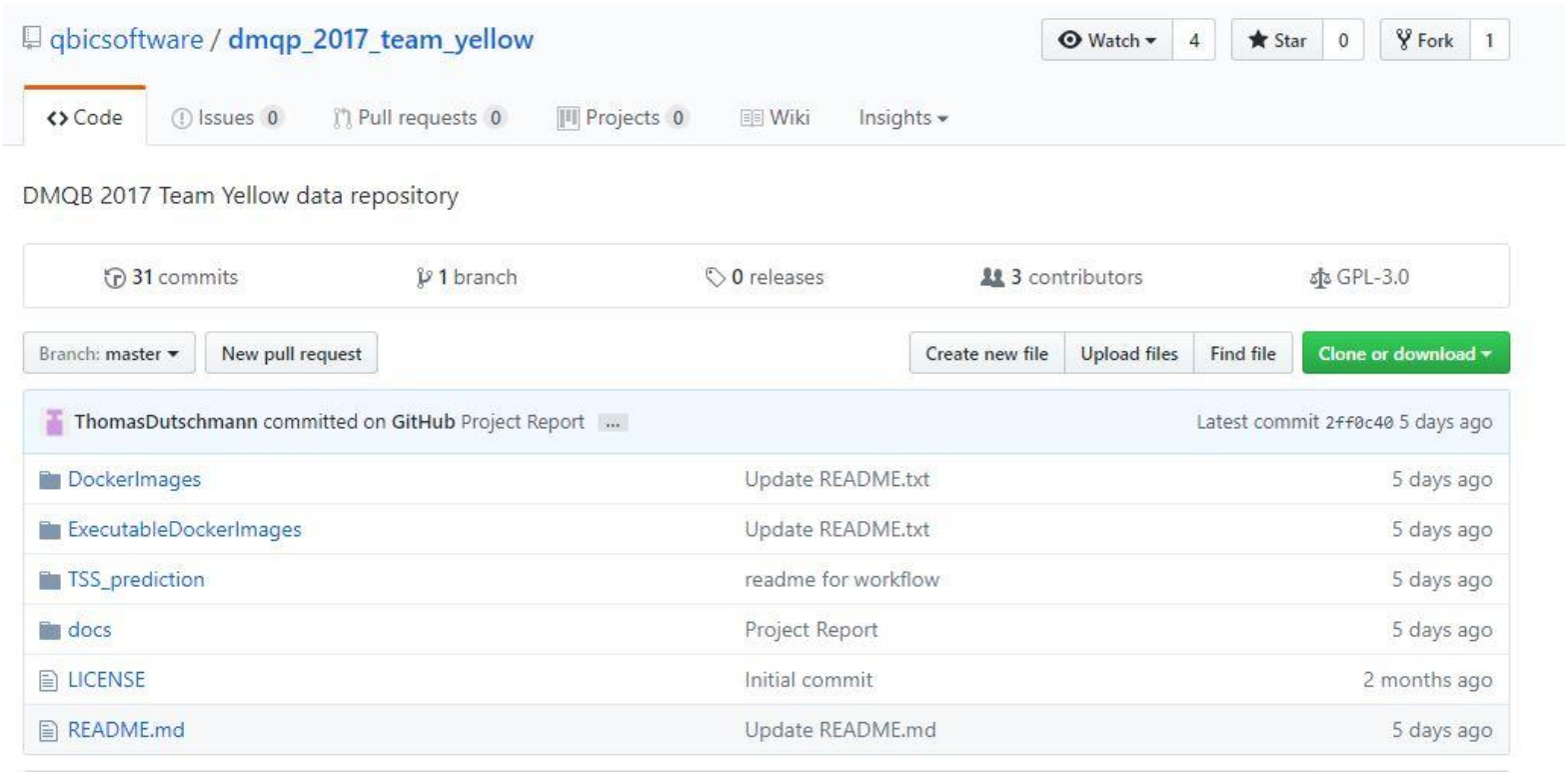
- Explained simplified, a container is a (runnable) environment
- The idea is based on virtual machines, but containers involve the hosts' kernel, so no external operating system is started
- Applications that require separate (or even incompatible) prerequisites can be "shipped" in separate containers, each with the necessary environment



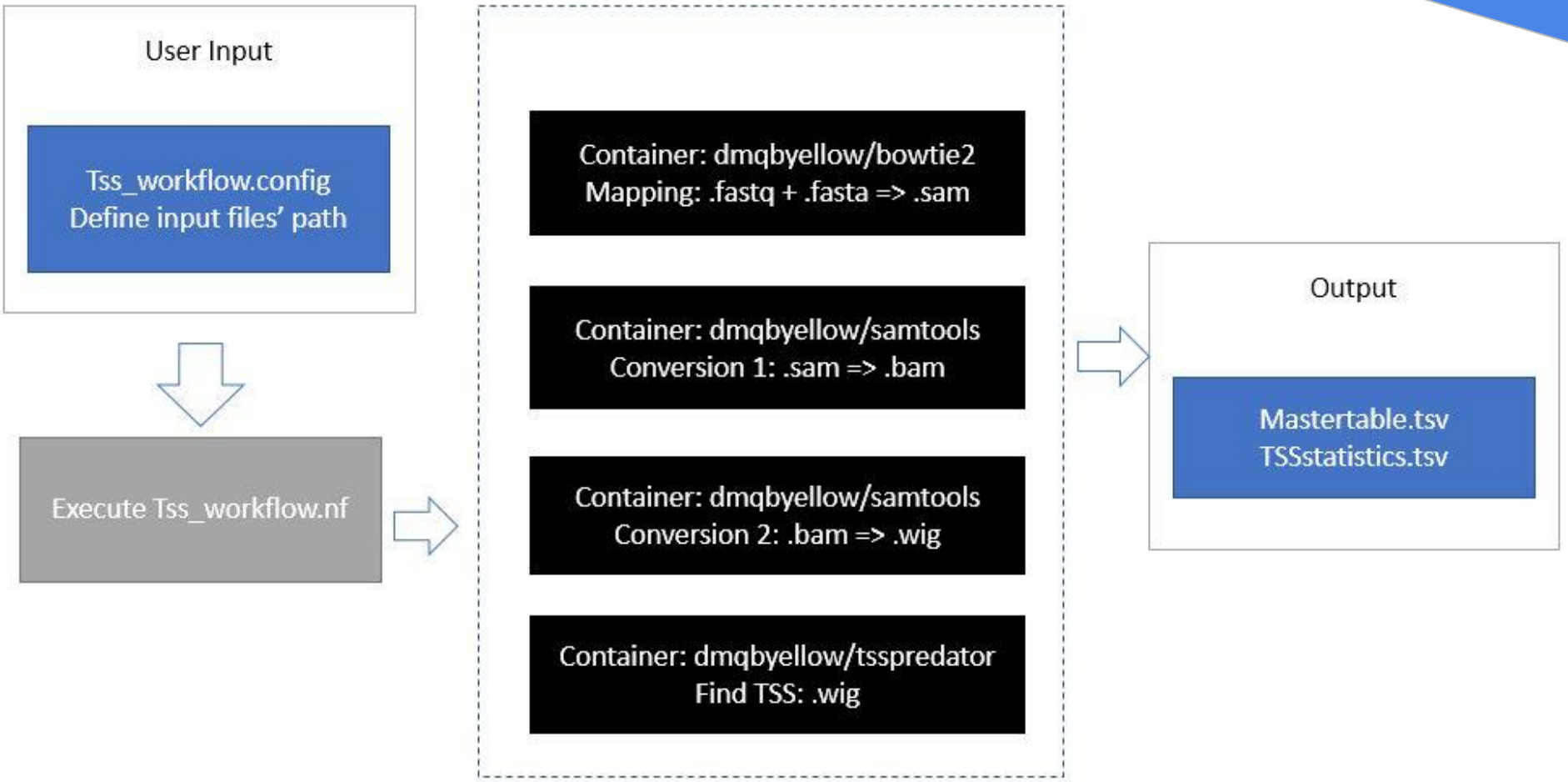
- We applied Docker, which is easy to use and already involves a vast collection of usable containers

Project architecture

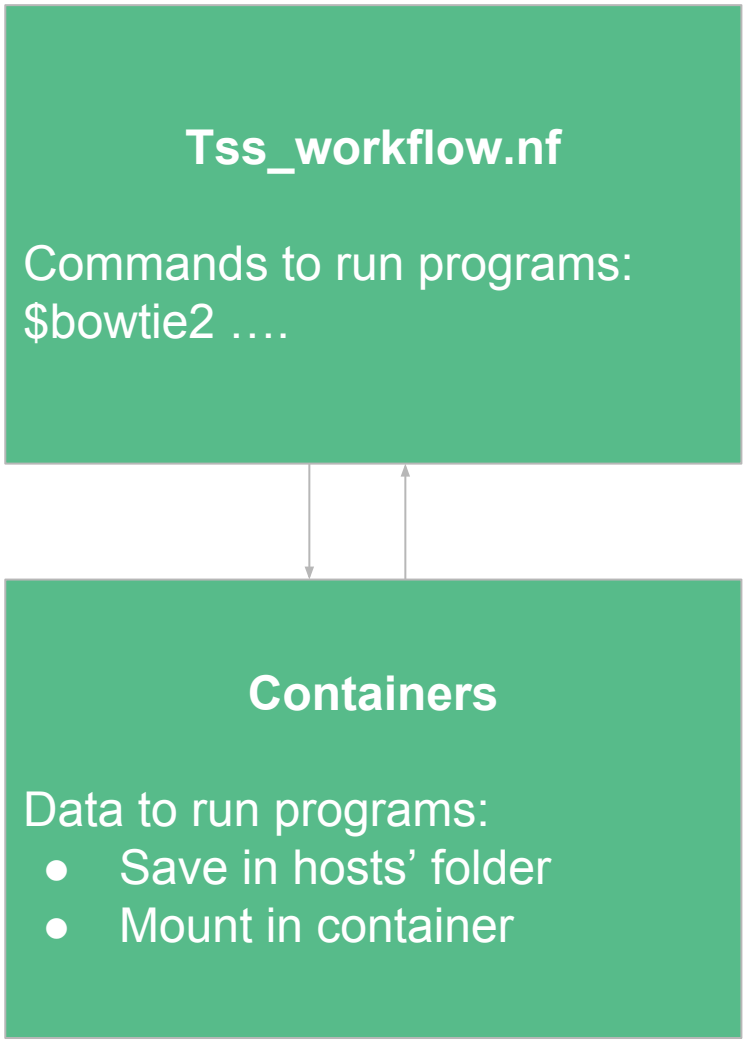
- Input files:
  - dRNA-seq data: .fastq
  - genome sequences: .fasta
  - genome annotation: .gff
- Output files:
  - MasterTable.tsv
  - TSSstatistics.tsv



Project architecture



Project architecture

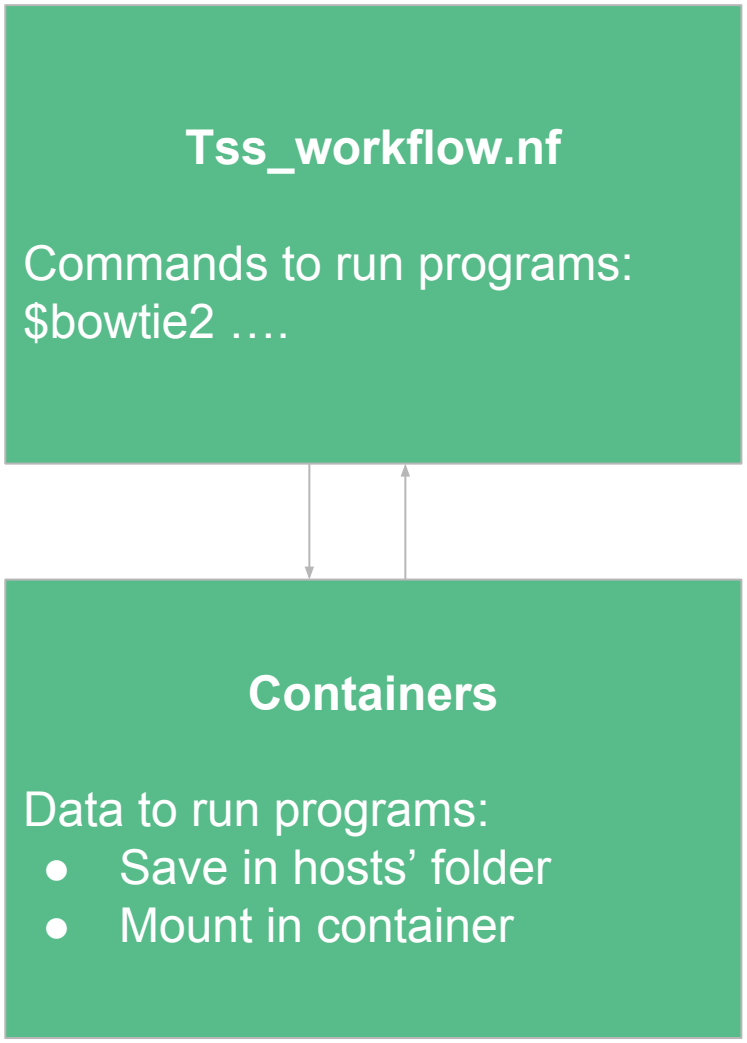


- Easier to adjust commands
  - compared to defining commands in Dockerfile

- No need to install additional softwares
  - Already done in Docker images

- Avoid environments' confliction
  - Java 8 for one program
  - Java 7 for the other

Project architecture



- Reasons for this structure:
  - Unable to read multiple files in container from Nextflow command: mysterious "*"
  - Each step yields intermediate files

- Reproducibility
  - Original input data is too large to upload
  - Provided example config file in our Github repository
  - Test each step separately

Obstacles: processes

process	tools	problems
mapping	bowtie 2.2.7 samtools	samtools requires many libraries as prerequisites to be installed; users need to keep the dockerfile up-to-date
conversion	tsstools 1.0_beta	tsstools will report errors when out directory doesn't exist
tsspredator	tsspredator 1.06	Java 8 required; paths of input files should be given by hand (as .config file); lack of space capacity when using ENTRYPOINT in Docker image

Obstacles: overall

Problem 1:

- Data stream of Nextflow for input and output files can't be used directly in a Docker container
- Tools generates output files in assigned paths

Solution:

Write path for each output to make it callable in a Docker container volume

Obstacles: overall

Problem 2:

- **File** object of Nextflow is invalid inside a Docker volume
- *Channel.fromPath()* method can't be used when reading multiple files, e.g '*.fasta'

Solution:

Give up on reading all files at the same time, let users input all files as a string 'FASTQ1 FASTQ2 ...'

Unsolved problem

- Unable to test a mapping software beside bowtie2
- Implementation of each tools still depends on Nextflow (command of each tool is inside Nextflow process)
- Alternative: build other docker images with **ENTRYPIONT**, but not integrate them into Nextflow

References

[1] “Docker explained.” https://www.docker.com/what-docker. Accessed 2017/6/5.

[2] “Wdl specification.” https://software.broadinstitute.org/wdl/documentation/topic?name=wdl-spec. Accessed 2017/6/5.

[3] “Bowtie2 homepage.” http://bowtie-bio.sourceforge.net/bowtie2/index.shtml. Accessed 2017/6/5.

[4] “Samtools.” http://samtools.sourceforge.net. Accessed 2017/6/5.

[5] “Bowtie2, how it performs, benchmark-based.” http://www.ecseq.com/support/benchmark. Accessed 2017/6/5.

Thanks for your attention!