Introduction to the Workflow Management System
-- Nextflow & nf-core

Yanni Chen

Graduate Research Assistant
High Performance Computing Center

April 20, 2022
Target Audience

This session is intended for an audience who …

• currently runs, plans to run workflow / pipeline / series analysis
• has to analyze data using different platform (personal computer, lab cluster, HPC, or cloud computing)
• tries to improve reproducibility of data analysis
• writes proposals for workflow development
• is simply curious about workflow management
Purposes of Training

Purposes of this training are …

• To introduce workflow management system to cluster users

• To introduce some existing workflows and their development community

• To demonstrate how to run the workflow on RedRaider Cluster
Part I: Workflow Management System

Part II: Nextflow, data-driven computational pipelines

Part III: nf-core pipelines, a community curation of bioinformatics pipelines

Part IV: Getting Help
Computation Workflow Management System
Fourth paradigm is using the computational power to generate, curate, analyze, archive a massive scientific data

Liew et al. 2016
Biological Experiment
Sequencing Data Prep and Data Size
(Illumina NovaSeq 6000)

In a single sequencing run on NovaSeq 6000:
about 4 days library prep and sequencing work could generate 6000 Gb data.
Challenges of scientific data explosion

• Large amount of data are generated from sequencing steps for different experiments
• Data need to go through a series of processing steps based on analysis
• Data processing may be executed on different platforms
The bioinformatic researchers are trying to automate the computational data processes to assist the data preparation process.
Part I: Workflow Management System

Bioinformatic Data Analysis Pipeline

Data Integration
- Read data from storage

Data Process
- Data trimming
- Quality control
- Data analysis

Data Summary
- Storage output
Part I: Workflow Management System

Advance Solution
Model biological data as workflows and use a workflow management system to organize their execution.

Pipeline + HPC infrastructure -> workflow management
Dictionary: the sequence of industrial, administrative, or other processes through which a piece of work passes from initiation to completion.

Wikipedia:
Workflows may be viewed as one fundamental building block to be combined with other parts of an organization's structure such as information technology, teams, projects and hierarchies.
A workflow management system (WfMS) is a software system for setting up, performing, and monitoring of a defined sequence of processes and tasks, with the broad goals of increasing productivity, reducing costs, becoming more agile, and improving information exchange within an organization.
Data intensive workflow often involves:

- Moving data from data sources to computational resources
- Cleaning, calibrating and normalizing data
- Constructing a model using part of that preprocessed data
- Validating the model with the remaining data
- Visualizing the results
- Moving the results to a storage system

Liew et al. 2016
Common roles for workflows

• Support for collaborative research by enabling scientific communities to share automated and formalized processes such as data analysis

• Construction free from distracting details about workflow management and execution

• The ability to automat workflow steps, that is, their mapping and execution, and to repeat in silicon experiments
Common roles for workflows (cont.)

• Integrating resources from distributed and heterogeneous enactment platforms
• Handling large volumes of data and complex computations
• Improving the execution through various optimization strategies
The benefit of using workflow

- **Use Domain Language**
  - handling I/O file between flows
  - provide standard log file along with analysis

- **Work with scheduler**
  - Each process get its job submission
  - Build-in checkpoint

- **Work with container**
  - Reproducibility / version control
  - Avoid install conflicts
Using prototyping to choose a bioinformatics workflow management system

Michael Jackson, Kostas Kavoussanakis, Edward W. J. Wallace

Published: February 25, 2021 • https://doi.org/10.1371/journal.pcbi.1008622

Abstract

Workflow management systems represent, manage, and execute multistep computational analyses and offer many benefits to bioinformaticians. They provide a common language for describing analysis workflows, contributing to reproducibility and to building libraries of reusable components. They can support both incremental build and re-entrancy—the ability to selectively re-execute parts of a workflow in the presence of additional inputs or changes in configuration and to resume execution from where a workflow previously stopped. Many workflow management systems enhance portability by supporting the use of containers, high-performance computing (HPC) systems, and clouds. Most importantly, workflow management systems allow bioinformaticians to delegate how their workflows are run to the workflow management system and its developers. This frees the bioinformaticians to focus on what these workflows should do, on their data analyses, and on their science.
A reactive workflow framework

A programming DSL /DSL2 that eases the writing of data-intensive computational pipelines.
Part II: Nextflow

Nextflow Highlights

• Domain specific language
• Work with cluster scheduler
• Error handling
• Reproducibility
Scripting Language

- **Groovy**
- **Bash**
- **Python**
- **R**
- **Perl**

Mixing scripting language of Nextflow: https://www.nextflow.io/example2.html
Part II: Nextflow

HPC Systems

• Slurm
• SGE
• LSE

Cloud Systems

• AW Batch
• Google Life Science
Part II: Nextflow

Benefit of Work with Different Scheduler:
• adaptability;
• parallelism of jobs:
  • create job channels
  • split job using scheduler

Scatter Execution of Nextflow Examples
Part II: Nextflow

Version Control

- Docker
- Apptainer / Singularity
- Shifter
- Podman
Exercise #0 Install NextFlow
Part II: Nextflow

Exercise #0 Install Nextflow (Solution)

1. **Check prerequisites**
   - Java 8 or later is required
   - Make sure 8 or later is installed on your computer by using the command:
     ```shell
curl -s https://get.nextflow.io | bash
```
   (it creates a file `nextflow` in the current dir)
   - Note: version numbers "1.8.y_2" and "8" identify the same Java release.

2. **Set up**
   - Dead easy to install
   - Enter this command in your terminal:
     ```shell
curl -s https://get.nextflow.io | bash
```
   (it creates a file `nextflow` in the current dir)
   - Note: it can also be downloaded from GitHub or installed by using Bioconda package manager.

3. **Launch**
   - Try a simple demo
   - Run the classic Hello world by entering the following command:
     ```shell
     ./nextflow run hello
     ```
Exercise #0: Install Nextflow (Solution)

- curl -s https://get.nextflow.io | bash
  Or
- conda install -c bioconda nextflow

Test Installation
- ./nextflow run hello
  Or
- nextflow run hello
Potential Problem:

1. nextflow installation needs java 8+ version:
   - `Java -version`

2. conda install needs Bioconda package manager:
   - `conda config --add channels defaults`
   - `conda config --add channels bioconda`
   - `conda config --add channels conda-forg`e

3. Install miniconda
Part II: Nextflow

Nextflow Scripting Structure

```bash
#!/usr/bin/env nextflow

params.str = 'Hello world!

process splitLetters {
  output:
  file 'chunk_*' into letters

  printf '{params.str}' | split -b 6 - chunk_
}

process convertToUpper {
  input:
  file x from letters.flatten()

  output:
  stdout result

  cat $x | tr '[a-z]' '[A-Z]' ...
}

result.view { it.trim() }
```

https://www.nextflow.io/docs/latest/getstarted.html#your-first-script
Part II: Nextflow

Your First Script:

• tutorial.nf

```bash
#!/usr/bin/env nextflow

params.str = 'Hello world!'

process splitLetters {
    output:
    file 'chunk_*' into letters

    printf '${params.str}' | split -b 6 - chunk_
}

process convertToUpper {
    input:
    file x from letters.flatten()

    output:
    stdout result

    cat $x | tr '[a-z]' '[A-Z]' }

result.view { it.trim() }
```
Exercise #1 Run first nextflow script

- Copy existing script to personal directory
- Run the script
- Modify script and run
Exercise #1 Run first nextflow script (solution)

```bash
#!/usr/bin/env nextflow

params.str = 'Hello world!

process splitLetters {
  output:
    file 'chunk_*' into letters

    printf '{params.str}' | split -b 6 - chunk_
}

process convertToUpper {
  input:
    file x from letters.flatten()

  output:
    stdout result

  cat $x | tr '[a-z]' '[A-Z']
}

result.view { it.trim() }
```

nextflow run tutorial.nf
Exercise #1 Run first nextflow script (solution)

```bash
nextflow run tutorial.nf

nextflow run tutorial.nf --resume

nextflow run tutorial.nf --str 'Bonjour le monde'
```
Part II: Nextflow

DSL2

nextflow.enable.dsl=2

process foo {
    output:
        path 'foo.txt'
    script:
        """""
            your_command > foo.txt
        """
}

process bar {
    input:
        path x
    output:
        path 'bar.txt'
    script:
        """""
            another_command $x > bar.txt
        """"
}
Part II: Nextflow

DSL2

• allow the definition of module, workflow, sub-workflow

• They can be import from library and re-use
Part II: Nextflow

**Exercise #2 Check available modules**

- using `nf-core/tools` helper package
Part II: Nextflow

Exercise #2 Check available modules (Solution)

• **Install nf-core**
  * conda install nf-core

• **list all remote modules**
  * nf-core modules list remote
A community effort to collect a curated set of analysis pipelines built using Nextflow.
Part III: nf-core pipelines

Use nf-core pipelines

check available nf-core pipeline, through website
Part III: nf-core pipelines

Exercise #3 Check available pipelines

• using `nf-core/tools` helper package
Exercise #3 Check available pipelines (Solution)

- **Install nf-core**
  - `conda install nf-core`

- **list all remote modules**
  - `nf-core list`
Run nf-core pipelines on RedRaider Cluster

- Ask scheduler for an interactive session
- Download nf-core pipelines
- Adapt the config file
  - Enable singularity container
  - Define the executor
- Run the job

nextflow run <pipeline>/workflow/ -c <config file>

or

nextflow run <pipeline>/workflow/ -c <config file> -resume
Config File

- Enable singularity container

```yaml
singularity {
  enabled = true
  cacheDir = "<singularity image location>"
}
```

- Define the executor

```yaml
process {
  executor = 'slurm'
  clusterOptions = '-p nocona -N 1 -n 2'
}
```

- Define the input files

Part III: nf-core pipelines
Exercise #4

- Download nf-core/RNASeq
- Run nf-core/RNASeq
Part III: nf-core pipelines

Exercise #4

• Download nf-core/RNASeq

• Run nf-core/RNASeq

  • nextflow pull nf-core/<pipeline> -r <version>

  • nextflow run nf-core/<pipeline> -r <version>

  • nf-core download nf-core/<pipeline> -r <version>
Part III: nf-core pipelines

Develop nf-core pipelines

• Discuss with nf-core
• Download nf-core template
• Check available modules
• Write your own pipelines
• Push to nf-core github
Getting Help
Part IV: Getting Help

- **slack**: communicate with developers, trouble shooting
- **GitHub**: see source code
- **Twitter**: milestone announcement
- **YouTube**: Tutorial videos
Part IV: Getting Help

- Visit Our Website:
  - Nextflow
  - nf-core
  - TTU Training Material

- Submit a support ticket:
  - Send an email to hpccsupport@ttu.edu
Part IV: Getting Help

Please help us improve this workshop and develop further training sessions:

- Ask questions at the end of this session
- Fill out survey forms
- Email any questions you have
• A Long run a partition RedRaider cluster
  Hardware: cores, clock speed, RAM
  Scheduler: job length
• Other workflow management systems