Going with the flow

DR. IAN SUDBERY
MOLECULAR BIOLOGY AND BIOTECHNOLOGY
I.SUDBERY@SHEFFIELD.AC.UK
@IanSudbery
www.github.com/sudlab
Using workflow managers to co-ordinate multistep analysis pipelines across multiple compute nodes in a reproducible manner.
Traditional HPC jobs are single monolithic programs using multi-node parallelism
Today many researchers use notebooks on clusters to do interactive/interpretive analysis of datasets.
Research computing spectrum

Single, large, long running, multimode jobs

? (A question mark)

Single core, quick running, interpretive analysis

e.g. a climate model

Regression analysis of a (quite) big dataset
Carrying out multi-step analyses by hand
Reproducibility

\[ \text{results} = \text{code(data)} \]

- Typing at a terminal is BAD NEWS for reproducibility
- Notebooks (for low intensity work)
- Containers
- Neither very easily work with multi-node parallelism
1. Easy/Automatic
2. Reproducible
3. Generalizable/Scalable
Workflow manager

- Specify dependencies between tasks
- Check if which dependencies need updating
- Only run tasks that need updating
- Do all this unsupervised.

Step1.gz → Step2.dat → Step3.txt
Modern workflow managers

• Either DSLs, configuration based or library
• Allow more complex forms of dependency
• Automatically submit each job to the cluster
• Monitor for successful completion and automatically submit next job
• Parameterizable
• Extensive logging
Modern Workflow managers

May also provide:

• conda/singularity/docker integration

• Use cloud compute and/or storage as well as local cluster

• Allow (distributed) execution of arbitrary code as well as shell scripts

• Helper functions for common analysis tasks
Some modern WFM

Ease of development

Flexibility, customisation

Scalability, portability, performance

Ruffus

SNAKEMAKE

nextflow

LUigi

TOLL

COMMON WORKFLOW LANGUAGE
<table>
<thead>
<tr>
<th>Feature</th>
<th>CGAT</th>
<th>SNAKEMAKE</th>
<th>NEXTFLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>Python</td>
<td>DSL</td>
<td>DSL</td>
</tr>
<tr>
<td>Dependency Paradigm</td>
<td>Explicit</td>
<td>Implicit (pull)</td>
<td>Implicit (push)</td>
</tr>
<tr>
<td>Rich dependency graphs</td>
<td>Yes</td>
<td>Partial</td>
<td>Yes</td>
</tr>
<tr>
<td>Conda integration</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Singularity/docker</td>
<td>Coming soon</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Arbitrary code</td>
<td>Python</td>
<td>Python</td>
<td>Any interpreted</td>
</tr>
<tr>
<td>Cloud Execution</td>
<td>No</td>
<td>Kubernetes</td>
<td>Amazon Batch</td>
</tr>
<tr>
<td>Cloud storage</td>
<td>Google/S3</td>
<td>Many</td>
<td>Many</td>
</tr>
<tr>
<td>Functions for common analysis</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Demonstration
It should take less time, effort and thought to do it the right way than to do it the wrong way.
## Ruffus dependency types

<table>
<thead>
<tr>
<th>Dependency Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Originate</td>
<td>None to one</td>
</tr>
<tr>
<td>Transform</td>
<td>One to One</td>
</tr>
<tr>
<td>Split</td>
<td>One to many</td>
</tr>
<tr>
<td>Merge</td>
<td>Many to one</td>
</tr>
<tr>
<td>Collate</td>
<td>Many to fewer</td>
</tr>
<tr>
<td>Subdivide</td>
<td>Many to more</td>
</tr>
<tr>
<td>Follows</td>
<td>Dependency without common files</td>
</tr>
<tr>
<td>Files</td>
<td>Arbitrary relationship</td>
</tr>
<tr>
<td>Permutations</td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td></td>
</tr>
<tr>
<td>Combinations</td>
<td>Combinatorics</td>
</tr>
<tr>
<td>Combinations_with_replacement</td>
<td></td>
</tr>
</tbody>
</table>
Pipelines can get quite complex...
Really very complicated!
Summary

• Automated farming and monitoring of pipelines of jobs to the cluster

• Create fully logged and reproducible workflows

• Generalizable and scalable

• Should be easier than writing a SGE submission script and faster than running in an interactive session

• Install with
  conda install -c bioconda -c conda-forge cgatcore
Acknowledgements

Sudbery Lab for Computational Genomics @ TUOS

Dr. Cristina Alexandru-Crivac
Jaime Alvarez-Benayas
Justin Coyne
Magdelena Dabrowska
Sumeet Deshmurkh
Jacob Parker
Ivaylo Yonchev

MRC Computational Genomics Analysis and Training/Tools

Dr. Adam Cribs
Sebastian Luna-Valero
Dr. Charlotte George
Dr. Antonio Berlanga-Taylor
Dr. Stephen Sansom
Dr. Tom Smith
Dr. Nicholas Ilott
Dr. Jethro Johnson
Jakub Scaber
Dr. Katherine Brown
Dr. David Sims
Dr. Andreas Heger

Dr. Leo Goodstat (Ruffus)
https://cgaticore.readthedocs.io

https://snakemake.readthedocs.io
Köster, J and Rahmann, S. *Bioinformatics* 2012, 28:2520

https://nextflow.io