DagOn*: Executing direct acyclic graphs as parallel jobs on anything

Raffaele Montella[1], Diana Di Luccio[1], Sokol Kosta[2],
Dante Sánchez Gallegos[1,3] and *José Luis González Compeán*[3]

raffaele.montella@uniparthenope.it diana.diluccio@uniparthenope.it sok@cmi.aau.dk
dsanchez@tamps.cinvestav.mx jgonzalez@tamps.cinvestav.mx

[1]Department of Science and Technologies
University of Napoli "Parthenope" - Italy

[2]CMI, Department of Electronic Systems
University of Aalborg - Denmark

[3]Adaptivez Heterogeneous Storage Research Group
Cinvestav-Tamaulipas - Mexico

Supported in part by the University of Napoli Parthenope, Italy (projects "Modelling mytilus farming System with Enhanced web technologies (MytiluSE - DSTE333)" funded by Campania Region/Veterinary sector and "Scientific computing advanced application for numerical modelling (ACSMN - DSTE347)". Thanks to the Mexico's National Council of Science and Technology (CONACYT) and Campania Center for Atmospheric and Marine Monitoring and Modelling (CCMMMA)

Introduction

- Any science field can be **now** considered as *computational science*.
- Workflow engines have existed since the beginning of the grid computing era.

***General purpose**, or **dedicated** tools, that is the question.*

Ascione, Isabella, Giulio Giunta, Patrizio Mariani, Raffaele Montella, and Angelo Riccio. "**A grid computing based virtual laboratory for environmental simulations.**" In European Conference on Parallel Processing, pp. 1085-1094. Springer, Berlin, Heidelberg, **2006**.

```
<jfdl:jfs project="experiment01">
  <!-- Job definition -->
  <jfdl:jobs>
    <jfdl:job name="downloadConditions"
          target="dgric.uniparthenope.it"
          rsl="downloadConditions.rsl"/>

    <jfdl:job name="runMM5"
          target="dgbeobi.uniparthenope.it"
          rsl="runMM5.rsl"/>
```

Montella, Raffaele, Alison Brizius, Diana Di Luccio, Cheryl Porter, Joshua Elliot, Ravi Madduri, David Kelly, Angelo Riccio, and Ian Foster. "**Using the FACE-IT portal and workflow engine for operational food quality prediction and assessment: An application to mussel farms monitoring in the Bay of Napoli, Italy.**" Future Generation Computer Systems (**2018**).

Galaxy PROJECT

Really, do we need for a "Yet Another Workflow Engine" ?

Direct Acyclic Graphs as parallel jobs on anything

DagOn* is a production-oriented workflow engine:

- **Integration** in the Python environment.
- **Minimal** footprint for external software components execution.
- **Avoiding** any workflow engine **centered data management**.
- **Straightforward** definition of tasks:
  - Python scripts.
  - Web interaction.
  - External software components.
- **Execution** sites **independence**:
  - Local / scheduler (SLURM).
  - Containers (Docker).
  - Clouds (AWS, OpenStack, DigitalOcean).

DagOn*

Named after the Phoenician god-fish *Dagon* known by ancient Greeks as *Triton*.

NB: The * symbol is the wildcard for **anything**.

Programming Model

Python Script: "DagOn* Hello World App"

```
import dagon
…
workflow=Workflow("myapp",settings)
workflow.add_task(new Task("a","...”))
workflow.add_task(new Task("b","workflow:///a"))
workflow.add_task(new Task("c","workflow:///a"))
workflow.add_task(new Task("d","workflow:///b workflow:///c"))
workflow.run()
sys.exit(0)
```

Task Flow — Defined by task dependencies.

Data Flow — Defined by data dependencies.

Data Flow

- Dealing with actual data files instead of high-level defined datasets.
- Performing backward data references in order to create dependencies.
- Having more Workflow instances in the same Python application.

DagOn* has been designed starting from the desidered programming model.

Architecture

Components

Python function.

Concurrent application thread.

REST & SOAP web services.

Websocket subscribe & publish.

DagOn Service.

External software.

SSH or scheduler.

Task embedded in container.

Task embedded in a virtual machine.

Instance status management.

Cloud independent.

The workflow:// schema

The **Batch** component takes charge of the management of data dependencies using the **workflow:// schema**.

workflow://workflow_unique_name/task_unique_name/

The schema label

The workflow unique name
An UUID could be used
If empty means “current workflow”

The task unique name
Can be dynamically generated by the Python script when the workflow is created programmatically.

Task scratch directory root

- Local
- Shared File System
- Remote scratch directory on physical machine, virtual instance or container

The garbage collector

- Tracks the storage and computational resources allocated during tasks execution.
- Proceeds to dispose them when no longer needed.

Make Dependencies

```
For each batch task in the <workflow> ...
    For each workflow://<workflow>/<task>/ reference in the task command line ...
        Increment the number of reference to <task>
```

On Task Finish

```
For each workflow://<workflow>/<task>/ reference in the task command line ...
    Decrement the number of reference to <task>
    If the number of reference to <task> is 0, clean up the involved resource
```

Clean Up

Local, remote or shared file system:
Remove the scratch directory.

Virtual machine instance:
Stop the instance.

Container:
Stop the container.

Container tasks

- Deployable in any Docker machine.
- Share a volume with the operating system host file system.
- If the tasks are on the same machine, the data transfer is done using shared memory.
- In a remote environment, data is copied to the volume shared between containers.

Cloud tasks

- Deployable in private, public and hybrid clouds.
- Define programmatically the flavour and configuration of the instance.
- Interoperable with other types of tasks (batch, containers, etc).
- SSH is used to make the DagOn app controlling the virtual machine instance.
- Data is transferred between tasks using the Stager component.
- Leverage on Apache Libcloud
- Tested with:
  - AWS
  - OpenStack
  - Digital Ocean
  - **Google Cloud**

Staging

```
globus-url-copy -vb -p 4 gsiftp://X/tmp/a/f1 gsiftp://Y/tmp/b/f2
```

Globus Connect Server

- Manages the data movement between all type of tasks.
- Fallback strategy:
  a. GridFTP
  b. Secure Copy
- Local tasks: memory, [shared] file system.

Application lifecycle

- As the workflow runs, all tasks are created in WAITING state.
- Each Workflow instance acts as DagOn Service client.
- When a task is going to be in RUNNING, the scratch directory is created.
- When a task is completed the garbage collector manage the used resources.

Operational weather forecast apps

Operational sea/weather forecast on the European regions with higher resolution on the Southern Italy area.

Applications

1. ROMS and WaComM for **mussels** farms **quality assessment** and management.
2. WaveWatch III for **coastal flooding** forecast in case of extreme sea storm events.
3. CALMET/CALPUFF for **wildfire** smoke tracing.

Model	Domain	Δt [sec.]	Grid dimension [N. of cells]			Grid spacing [m]	Output File Dimension [Gb] (1-hour of simulation)	N. of output file for each run (7 days)	N. of MPI processes for each Δt	Execution time [mm:ss] (1-hour of simulation)
			West-East	South-North	Bottom-Top					
WRF	d01	150	230	209	28	25000	0.11	169	96 (12 core*8 nodes)	05:05
	d02	30	361	336		5000	0.29			
	d03	6	301	306		1000	0.22			
ROMS	d03	30	1375	1021	30	200	2.28		80 (10 core*4 nodes)	07:12

Models, problem size and wall clock time.

Operational weather forecast apps

ROMS and WaComM for **mussels** farms **quality assessment** and management (MytiluSE app).

Use case: MytiluSE app

- Part of the DagOn* App running in production at http://meteo.uniparthenope.it
- The workflow is defined dynamically.
- Tasks are defined as Batch/Slurm ones.
- Once all the tasks have been added, the workflow is run.

```
# Create the orchestration workflow
workflow=Workflow("roms",config)

# Some beauty logging
workflow.logger.info("initialization date: %s",i_date)

# The myocean2roms task executed using Slurm
taskMyocean2roms=batch.Slurm("myocean2roms",command_dir_base+"/myocean2roms "+i_date+" "+gridFilename+" "+domainId)

# The wrf2roms task executed using Slurm
taskWrf2roms=batch.Slurm("wrf2roms",command_dir_base+"/wrf2roms "+i_date+" "+str(hours)+" "+gridFilename+" "+domainId+" "+wrfDom
ainId)

# The roms task executed using Slurm
taskRoms=batch.Slurm("roms",command_dir_base+"/roms "+i_date+" "+gridFilename+" "+domainId+" workflow://myocean2roms/ini-"+domai
nId+".nc workflow://myocean2roms/bry-"+domainId+".nc workflow:///wrf2roms/wind-"+domainId+".nc")

# The wrf2roms task executed using Slurm
taskPublishRomsOutput=batch.Slurm("publishRomsOutput",command_dir_base+"/publishRomsOutput "+i_date+"00 "+i_date+"00 rms3 workfl
ow://roms/output/ocean_his_")

# add tasks to the workflow
workflow.add_task(taskMyocean2roms)
workflow.add_task(taskWrf2roms)
workflow.add_task(taskRoms)
workflow.add_task(taskPublishRomsOutput)
```

```
frontend: 20181108Z00 data download completed!
Running workflow...
2018-11-08 21:35:25,829 root         INFO     initialization date: 20181108Z00
2018-11-08 21:35:25,830 root         DEBUG    Running workflow: roms
2018-11-08 21:35:25,831 root         DEBUG    myocean2roms: Status.WAITING
2018-11-08 21:35:25,831 root         DEBUG    myocean2roms: Status.RUNNING
2018-11-08 21:35:25,831 root         DEBUG    myocean2roms: Executing...
2018-11-08 21:35:25,832 root         DEBUG    wrf2roms: Status.WAITING
2018-11-08 21:35:25,832 root         DEBUG    wrf2roms: Status.RUNNING
2018-11-08 21:35:25,832 root         DEBUG    roms: Status.WAITING
2018-11-08 21:35:25,833 root         DEBUG    wrf2roms: Executing...
2018-11-08 21:35:25,833 root         DEBUG    publishRomsOutput: Status.WAITING
2018-11-08 21:35:25,848 root         DEBUG    myocean2roms: Scratch directory: /home/ccmmma/dev/yawe/tmp//1541712925832-myocean2roms
2018-11-08 21:35:25,856 root         DEBUG    wrf2roms: Scratch directory: /home/ccmmma/dev/yawe/tmp//1541712925833-wrf2roms
```

Local queue status (HPC scenario)

```
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
 4193       gpu     wrf_6   ccmmma  R    1:09:57      8 node[1,3-5,9-12]
 4195     hicpu wrf2roms   ccmmma  R       3:01      1 node13
 4196     hicpu myocean2   ccmmma  R       3:01      1 node14
```

Use case: MytiluSE app results

10m Wind speed and direction

Sea Surface Current

http://meteo.uniparthenope.it

Forecast: 23Z08NOV2018 ca001

http://meteo.uniparthenope.it

Forecast: 23Z08NOV2018 ca008

http://meteo.uniparthenope.it

November 8th, 2018
23:00 UTC

Simulation: 168h
Computing: ~24h

Potential sea pollution

Forecast: 23Z08NOV2018 ca000

Related Work

	http://swift-lang.org	http://parsl-project.org	https://galaxyproject.org	
	swift	Parsl	Galaxy PROJECT	dagon*
Workflow creation	Scripting	Python scripts Annotations	Intuitive Web Interface	Python scripts and JSON files **Object Oriented**
Representation	Graphs Data structure	Dependent task graph Futures	Direct Acyclic Graphs Data structure	Direct Acyclic Graphs, **workflow:// schema**
Data exchange	Files	Python object serialization	Datatype/Dataset Shared File System	Files Shared memory objects, Shared File System, Secure Copy, Grid-Ftp **Garbage collector**
Deployment	Local, HPC, cloud	Local, HPC, containers	Local, HPC, cloud	Local, HPC, containers, **cloud**
Disadvantages	The access to the filesystem could generate a bottleneck.	Can just manage local files and files accessible by Globus.	Requires the deployment of several different software packages.	Still work in progress. Quantitative performance evaluation on experimental use cases.

Conclusion and future ...tasks

- Python based tool for data intensive scientific workflows targeting **production applications**.
- DagOn* programming model enables the developer to **embed workflows in** already existing Python **scientific applications**.
- Peculiar features as the dependencies management via the **workflow://** schema and the ***garbage collector***.
- Tasks can be executed **locally**, using a local **scheduler**, on local or remote **containers** and on public, private or hybrid **clouds**.

http://meteo.uniparthenope.it

Real world use case application for **weather and marine forecasts.**

Conclusion and future... tasks

- **Single sign on system** component and a **credential repository.**
- **Checkpoint** and **recovery** workflow features.
- The DagOn* Service needs for **improvements** in web **user interface**, REST **APIs**, usage **data analysis** and **resource** allocation **management**.
- New **real** world test cases focusing on massive **weather station** data **processing**.
- Compare and contrast **evaluation** versus Swift and Parsl.
- Possible **integration** with Parsl.

Open Source, Apache 2.0 license: http://github//dagonstar

Introduction, contextualization and motivations

- Any science field can be **now** considered as *computational science*.
- Workflow engines play a **primary role** in computational sciences.

- The Cloud technology provides **elastic** and virtually **infinite** computational **resources**.
- Workflow engines ensure computational experiment **reproducibility**.
- The **scientific approach** has been enriched by the use of computational **models** and data analysis.

2000s

*_Science + Computer Science = "*Computational Science*"...

Present Days

...the "computational" is be **in** each Science **by default**.

Outlook

...any Science is "**computational**" by design.

Introduction, contextualization and motivations

- Workflow engines for data-intensive science have existed since the beginning of the grid computing era.

Ascione, Isabella, Giulio Giunta, Patrizio Mariani, Raffaele Montella, and Angelo Riccio. "**A grid computing based virtual laboratory for environmental simulations.**" In European Conference on Parallel Processing, pp. 1085-1094. Springer, Berlin, Heidelberg, **2006**.

```
<jfdl:jfs project="experiment01">
  <!-- Job definition -->
  <jfdl:jobs>
    <jfdl:job name="downloadConditions"
          target="dgric.uniparthenope.it"
          rsl="downloadConditions.rsl"/>

    <jfdl:job name="runMM5"
          target="dgbeobi.uniparthenope.it"
          rsl="runMM5.rsl"/>
```

Montella, Raffaele, Alison Brizius, Diana Di Luccio, Cheryl Porter, Joshua Elliot, Ravi Madduri, David Kelly, Angelo Riccio, and Ian Foster. "**Using the FACE-IT portal and workflow engine for operational food quality prediction and assessment: An application to mussel farms monitoring in the Bay of Napoli, Italy.**" Future Generation Computer Systems (**2018**).

Workflows gained the role of first-class citizen in nowaday applications.
Ranging from mobile/embedded computing at the edge to large scale science.
...embedding workflows in regular applications could be the next big thing...

Introduction, contextualization and motivations

***General purpose**, or **dedicated** tools, that is the question.*

- Define workflows **programmatically** (production).
- Execute tasks in very **heterogeneous** environments.
- Use **multiple workflows** in the same applications, interacting each other if needed.
- **Move** data across **different domains** or, vice-versa, move the **computation close** to the data.

Really, do we need for a "Yet Another Workflow Engine" ?