Scientific Workflow Repeatability through Cloud-Aware Provenance

Khawar Hasham, Kamran Munir, Jetendre Shamdasani, Richard McClatchey

University of the West of England (UWE)
Introduction

- Scientific community is using workflows to process and analyse large amounts of data
- Recent studies [1] are evaluating workflow execution on Cloud
- Cloud provides resources on-demand and nature of resources are dynamic
  - They are acquired or released on-demand
- Provenance is an important data;
  - Help in verifying and debugging the execution
  - Verify inputs and results
  - Repeating an experiment
Motivation

• Around 80% of the workflows cannot be reproduced, and 12% of them are due to the lack of information about the execution environment [2]

• Workflow Provenance capturing in Cloud
  • Challenging due to the dynamic and geographically distributed nature of Cloud computing [3]

• Workflow Repeatability in Cloud
  • Exact resource configuration should be known
    • Resources are acquired by passing on configuration information
    • Helps in achieving similar performance with similar resources

• Workflows executed on Cloud need to provide existing provenance information along with the Cloud resource information
Literature Survey

- Provenance applications
  - Enables a researcher to reproduce to verify experimental results

- Provenance approaches or systems in Grid
  - Chimera [4] stores physical location of machine
  - However, They lack computational environment or infrastructure information in the collected provenance

- Provenance approaches in Cloud
  - Hypervisor level information [7] (physical to virtual resource); mainly to detect data flow
  - File access level information [8]
  - These approaches don’t consider workflow executions, which is the focus of this paper
Literature Survey

- Literature survey on Reproducibility
  - Systems like ReproZip [8] or CARE [9]
    - Work on individual job or execution, thus not applicable to the workflow level.
    - They don’t consider the resource on which execution is taking place
    - Takes a lot of storage space.
  - Recently published (August 2014) [10], Semantic based approach uses annotations exhaustively
    - Depends on user provided data for resource provisioning and configuration

- Outcome: We can collect workflow provenance as well as virtualization level information, but we need to combine them
Re-provision an execution infrastructure on Cloud

- Focus of this paper is on re-provisioning the similar resources on Cloud, thus enabling repeatability of workflow execution on similar resources

- Information required for provisioning Cloud resources
  - Parameters require to request a new resource on the Cloud, to have similar execution infrastructure
    - CPU, RAM, Hard disk, OS Image used
  - Is this information sufficient to re-provision a resource on the Cloud? Yes

- In this paper, it is assumed that OS image contains all the required software libraries
Augmenting Provenance information with Cloud resource information

**Capture**
- Collect provenance data from the underlined Workflow Management System using APIs or database
  - Pegasus database
- Find APIs to interact with Cloud middleware to collect infrastructure (Cloud resource) information.
  - Apache libcloud

**Combine**
- How to combine them together
  - Identify the mapping points between two sets of information.
  - In current prototype, it is based on IP address of the resource.

- This provides detailed information about resources used for a workflow execution on Cloud

1: https://libcloud.apache.org
Proposed architecture

Cloud environment

Application Layer

Workflow Management System

Virtualized Layer

VM1
Condor

VM2
Condor

Physical Layer

Scientist

Stores/selects an existing workflow

Submits workflow

Provenance Aggregator

Workflow Provenance

Cloud Layer Provenance

Provenance API

Workflow Provenance Store

Existing tools

Prototype components

Prototype Storage

Workflow provenance

Infrastructure information

Cloud-aware provenance

Scientist

Stores/selects an existing workflow
Provenance Aggregator: combining workflow provenance with Cloud information

Start (wfid)

Get Workflow Jobs (wfJobs)

Get VM list from Cloud (vmList)

Establish mapping (wfJobs, vmList)

Has more jobs?

Yes

Insert mapping (job, flavour, image)

No

VM.ip = job.ip

Yes

Insert Mapping

No

Next record

End
Repeat a workflow execution using Cloud-aware Provenance

1) Repeat request
2(a) get Workflow (wfid)
2(b) get Cloud Resource (wfid)
3) Provision resource (flavour, image, name)
4) Submit workflow
5) monitor
6) Collect provenance information

Scientist

RepeatWorkflow

ProvenanceAggregator

Workflow Provenance Store

VM_1

Cloud

VM_n

Condor

Condor
Prototype Implementation

- A proof of concept prototype has been developed using Apache libcloud API to interact with Openstack Cloud middleware.
- Pegasus WMS along with condor cluster on virtual machines is used to submit and execute workflows.
- A simple WordCount workflow with 4 jobs is executed.
- Mapping between Pegasus Provenance and Cloud Virtual layer information is established.
  - The virtual layer information is mapped with the workflow provenance retrieved from Pegasus.
- Workflow re-execution after re-provisioning the Cloud resources using Cloud-aware provenance.
**Initial results – similar infrastructure**

**Original workflow execution – infrastructure information on Cloud**

<table>
<thead>
<tr>
<th>wf_id</th>
<th>hostip</th>
<th>nodename</th>
<th>middleware</th>
<th>flavorid</th>
<th>minRAM</th>
<th>minHD</th>
<th>vCPU</th>
<th>image_name</th>
<th>image_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>114</td>
<td>172.16.1.49</td>
<td>osdc-vm3.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
<tr>
<td>114</td>
<td>172.16.1.98</td>
<td>mynode.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
</tbody>
</table>

**Reproduced workflow execution – infrastructure information on Cloud**

<table>
<thead>
<tr>
<th>wf_id</th>
<th>hostip</th>
<th>nodename</th>
<th>middleware</th>
<th>flavorid</th>
<th>minRAM</th>
<th>minHD</th>
<th>vCPU</th>
<th>image_name</th>
<th>image_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>117</td>
<td>172.16.1.183</td>
<td>osdc-vm3-rep.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
<tr>
<td>117</td>
<td>172.16.1.187</td>
<td>mynode-rep.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
<tr>
<td>122</td>
<td>172.16.1.114</td>
<td>osdc-vm3-rep.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
<tr>
<td>122</td>
<td>172.16.1.221</td>
<td>mynode-rep.novalocal</td>
<td>OpenStack</td>
<td>2</td>
<td>2048</td>
<td>20</td>
<td>1</td>
<td>wf Peg_repeat</td>
<td>f102960c-557c-4253-8277-2df5ffe3c169</td>
</tr>
</tbody>
</table>
Future work

- Improve the initial prototype
- Detailed evaluation
  - Comparison of outputs produced by two workflow runs
  - Detailed provenance comparison of two workflow runs to evaluate the workflow reproducibility
    - Workflow structure
    - Outputs
    - Infrastructure
- Evaluate the impact of different resource configurations on workflow’s overall execution performance
Thank you!


