Scientific Workflows and Cloud Computing

Gideon Juve
USC Information Sciences Institute
gideon@isi.edu
Scientific Workflows

- Loosely-coupled parallel applications
- Expressed as directed acyclic graphs (DAGs)
  - Nodes = Tasks, Edges = Dependencies
- Data is communicated via files

Small Montage Workflow
Workflow Management System

- **Pegasus** – workflow planner
  - Efficiently maps tasks and data to resources
- **DAGMan** – workflow engine
  - Tracks dependencies, releases tasks, retries tasks
- **Condor** – task manager
  - Dispatches tasks (and data) to resources
Amazon Web Services (AWS)

- IaaS Cloud
- Services
  - Elastic Compute Cloud (EC2)
    - Provision virtual machine instances
  - Simple Storage Service (S3)
    - Object-based storage system
    - Put/Get files from a global repository
  - Elastic Block Store (EBS)
    - Block-based storage system
    - Unshared, SAN-like volumes
  - Others (queue, key-value, RDBMS, MapReduce, etc.)
Workflows and Clouds

• Benefits
  – User control over environment
  – On-demand provisioning / Elasticity
  – SLA, support, reliability, maintenance

• Drawbacks
  – Complexity (more control = more work)
  – Cost
  – Performance
  – Resource Availability
  – Vendor Lock-In
Questions About Clouds

• How can we deploy workflows in the cloud?
  – Install and configure software
  – Execute workflow tasks
  – Store workflow data

• How well do workflows perform in the cloud?
  – Compared to grids and clusters
  – Using various storage systems

• How much does it cost to run a workflow?
  – To provision resources
  – To store data
  – To transfer data
Deploying Workflows in the Cloud

• Virtual Machines/Virtual Machine Images
  – Clouds provide resources, but the software is up to the user

• Virtual Clusters
  – Collections of virtual machines used together
  – Configured to mimic traditional clusters

• Contextualization
  – Dynamically configuring virtual clusters is not trivial
  – Nimbus Context Broker – automates provisioning and configuration of virtual clusters
Execution Environment

Cloud

Submit Host

Amazon EC2

Grid

Submit Host

NCSA Abe
Workflow Storage In the Cloud

• Executables
  – Transfer into cloud
  – Store in VM image

• Input Data
  – Transfer into cloud
  – Store in cloud

• Intermediate Data
  – Use local disk (single node only)
  – Use distributed storage system

• Output Data
  – Transfer out of cloud
  – Store in cloud
Resource Type Experiments

• Run workflows on single instances of different resource types (using local disk)

• Goals:
  – Compare performance/cost of cloud resources
  – Compare performance of grid and cloud
  – Characterize virtualization overhead
  – Quantify performance benefit of network/file system

<table>
<thead>
<tr>
<th>Type</th>
<th>Arch.</th>
<th>CPU</th>
<th>Cores</th>
<th>Memory</th>
<th>Network</th>
<th>Storage</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>m1.small</td>
<td>32-bit</td>
<td>2.0-2.6 GHz Opteron</td>
<td>1/2</td>
<td>1.7 GB</td>
<td>1-Gbps Ethernet</td>
<td>Local disk</td>
<td>$0.085/hr</td>
</tr>
<tr>
<td>m1.large</td>
<td>64-bit</td>
<td>2.0-2.6 GHz Opteron</td>
<td>2</td>
<td>7.5 GB</td>
<td>1-Gbps Ethernet</td>
<td>Local disk</td>
<td>$0.12/hr</td>
</tr>
<tr>
<td>m1.xlarge</td>
<td>64-bit</td>
<td>2.0-2.6 GHz Opteron</td>
<td>4</td>
<td>15 GB</td>
<td>1-Gbps Ethernet</td>
<td>Local disk</td>
<td>$0.68/hr</td>
</tr>
<tr>
<td>c1.medium</td>
<td>32-bit</td>
<td>2.33-2.66 GHz Xeon</td>
<td>2</td>
<td>1.7 GB</td>
<td>1-Gbps Ethernet</td>
<td>Local disk</td>
<td>$0.17/hr</td>
</tr>
<tr>
<td>c1.xlarge</td>
<td>64-bit</td>
<td>2.33-2.66 GHz Xeon</td>
<td>8</td>
<td>7.5 GB</td>
<td>1-Gbps Ethernet</td>
<td>Local disk</td>
<td>$0.68/hr</td>
</tr>
<tr>
<td>abe.local</td>
<td>64-bit</td>
<td>2.33 GHz Xeon</td>
<td>8</td>
<td>8 GB</td>
<td>10-Gbps InfiniBand</td>
<td>Local disk</td>
<td>N/A</td>
</tr>
<tr>
<td>abe.lustre</td>
<td>64-bit</td>
<td>2.33 GHz Xeon</td>
<td>8</td>
<td>8 GB</td>
<td>10-Gbps InfiniBand</td>
<td>Lustre</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Storage System Experiments

• Investigate different options for storing intermediate data in a virtual cluster

• Goals
  – Determine how to deploy storage systems
  – Compare performance/cost of storage systems
  – Determine which storage system

• Amazon Issues
  – EC2 does not allow kernel patches (no Lustre, Ceph)
  – EBS volumes cannot be shared between nodes

• Use c1.xlarge resources
Storage Systems

• Local Disk
  – RAID0 across available partitions with XFS

• NFS: Network file system
  – 1 dedicated node (m1.xlarge)

• PVFS: Parallel, striped cluster file system
  – Workers host PVFS and run tasks

• GlusterFS: Distributed file system
  – Workers host GlusterFS and run tasks
  – NUFA, and Distribute modes

• Amazon S3: Object-based storage system
  – Non-POSIX interface required changes to Pegasus
  – Data is cached on workers
Example Applications

• Montage (astronomy)
  – I/O: High
  – Memory: Low
  – CPU: Low

• Epigenome (bioinformatics)
  – I/O: Low
  – Memory: Medium
  – CPU: High

• Broadband (earthquake science)
  – I/O: Medium
  – Memory: High
  – CPU: Medium
• Virtualization overhead is less than 10%
• Network/file system are biggest advantage for grid
• c1.xlarge is good, m1.small is bad
• Montage (high I/O) likes Lustre, Epigenome (high CPU) doesn’t care
• GlusterFS (NUFA) is best overall
• Epigenome file system doesn’t matter
• NFS, PVFS perform relatively poorly
• S3 performs poorly when reuse is low, and # files is large
Cost Components

• Resource Cost
  – Cost for VM instances
  – Billed by the hour

• Transfer Cost
  – Cost to copy data to/from cloud over network
  – Billed by the GB

• Storage Cost
  – Cost to store VM images, application data
  – Billed by the GB-month, # of accesses
- The per-workflow cost is not bad
- m1.small is not the cheapest
- m1.large is most cost-effective
- Resources with best performance are not cheapest
- Per-hour billing affects price/performance tradeoff
Resource Cost (by Storage System)

- Cost tracks performance
- Adding resources does not reduce cost (except in unusual cases)
- S3, NFS are at a disadvantage because of extra charges
Transfer Cost

<table>
<thead>
<tr>
<th>Application</th>
<th>Input</th>
<th>Output</th>
<th>Logs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montage</td>
<td>4291 MB</td>
<td>7970 MB</td>
<td>40 MB</td>
</tr>
<tr>
<td>Broadband</td>
<td>4109 MB</td>
<td>159 MB</td>
<td>5.5 MB</td>
</tr>
<tr>
<td>Epigenome</td>
<td>1843 MB</td>
<td>299 MB</td>
<td>3.3 MB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application</th>
<th>Input</th>
<th>Output</th>
<th>Logs</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montage</td>
<td>$0.42</td>
<td>$1.32</td>
<td>&lt; $0.01</td>
<td>$1.75</td>
</tr>
<tr>
<td>Broadband</td>
<td>$0.40</td>
<td>$0.03</td>
<td>&lt; $0.01</td>
<td>$0.43</td>
</tr>
<tr>
<td>Epigenome</td>
<td>$0.18</td>
<td>$0.05</td>
<td>&lt; $0.01</td>
<td>$0.23</td>
</tr>
</tbody>
</table>

Transfer Sizes

- Cost of transferring data to/from cloud
  - Input: $0.10/GB (first 10 TB, free till June 30)
  - Output: $0.17/GB (first 10 TB, now $0.15)

- Transfer costs are a relatively large
  - For Montage, transferring data costs more than computing it

- Costs can be reduced by storing input data in the cloud and using it for multiple workflows
Storage Cost

• **Storage Charge**
  – Price for storing data
  – Per GB-month

• **Access Charge**
  – Price for accessing data
  – Per operation

• **S3**
  – Storage: $0.15 / GB-month
  – Access: PUT: $0.01 / 1,000
  – GET: $0.01 / 10,000

• **EBS**
  – Storage: $0.10 / GB-month
  – Access: $0.10 / million IOs

<table>
<thead>
<tr>
<th>Application</th>
<th>Volume Size</th>
<th>Monthly Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montage</td>
<td>5GB</td>
<td>$0.66</td>
</tr>
<tr>
<td>Broadband</td>
<td>5GB</td>
<td>$0.60</td>
</tr>
<tr>
<td>Epigenome</td>
<td>2GB</td>
<td>$0.26</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image</th>
<th>Size</th>
<th>Monthly Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>32-bit</td>
<td>773 MB</td>
<td>$0.11</td>
</tr>
<tr>
<td>64-bit</td>
<td>729 MB</td>
<td>$0.11</td>
</tr>
</tbody>
</table>

Storage of Inputs in EBS  Storage of VM images in S3
Conclusions

• Deployment and Usability
  – Easy to start using, but some work is required to generate images and automate configuration
  – Tools like Nimbus Context Broker can help
  – Little maintenance, good reliability

• Performance
  – Not bad given resources, but not as good as dedicated clusters & grids
  – VM overhead is less than 10% for apps tested
  – c1.xlarge has best performance overall
  – Avoid using m1.small
Conclusions

• Cost
  – m1.small is not always the cheapest resource
  – Transferring data is relatively expensive
  – Store inputs long-term if possible
  – Using multiple nodes is not cost-effective
Web Resources

• Pegasus
  – http://pegasus.isi.edu
• Condor/DAGMan
  – http://cs.wisc.edu/condor
• Nimbus Context Broker
  – http://www.nimbusproject.org/
• Amazon Web Services
  – http://aws.amazon.com