Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 9 (2012) 1630 – 1634

Procedia Computer Science

International Conference on Computational Science, ICCS 2012

Early Cloud Experiences with the Kepler Scientific Workflow System

Jianwu Wang, Ilkay Altintas*

San Diego Supercomputer Center, UCSD, 9500 Gilman Drive, MC 0505, La Jolla, CA 92093, U.S.A.

Abstract

With the increasing popularity of the Cloud computing, there are more and more requirements for scientific workflows to utilize Cloud resources. In this paper, we present our preliminary work and experiences on enabling the interaction between the Kepler scientific workflow system and the Amazon Elastic Compute Cloud (EC2). A set of EC2 actors and Kepler Amazon Machine Images are introduced with the discussion on their different usage modes. Through two bioinformatics usecases, we demonstrate the capability of our work for both Cloud resource coordination and workflow execution on virtual Cloud resources.

Keywords: Scientific workflows, data-intensive, bioinformatics, Amazon EC2, Cloud computing

1. Introduction

Because of its virtualization, abundance and scalability characteristics, Cloud is becoming a popular environment for both scientific and business applications. For example, Amazon Elastic Compute Cloud (Amazon EC2)[1] provides infrastructure as a service (IaaS), which includes virtualized hardware (including storage and network) and pre-configured software stack. Users can start one or more virtual instances from one Amazon Machine Image (AMI), and administrate them as their own machines. Users can also get storage service from Amazon Simple Storage Service (S3)[2] or Elastic Block Store (EBS)[3]. Instead of making investments to purchase their own computing resources, users can start using the available Cloud resources for compute and storage instantly via these services and pay as they go based on their usage.

The popularity of the Cloud computing introduced new requirements for scientific workflows [1]. First, workflow users should be able to utilize available Cloud resources and packages since more and more domain-specific toolkits are available on the Cloud, especially on Amazon EC2. As a typical instance, Bio-Linux project[4] provides an AMI containing more than 500 bioinformatics programs. By initiating the Bio-Linux AMI on EC2, users can run these

*Corresponding author. Tel.: +1-858-822-5453; fax: +1-858-822-3693
Email address: {jianwu, altintas}@sdsc.edu (Jianwu Wang, Ilkay Altintas)

[1] Amazon EC2: http://aws.amazon.com/ec2/, 2012.

[2] Amazon S3: http://aws.amazon.com/s3/, 2012.

[3] Amazon EBS: http://aws.amazon.com/ebs/, 2012.

[4] Bio-Linux Project: http://nebc.nerc.ac.uk/tools/bio-linux/, 2012.

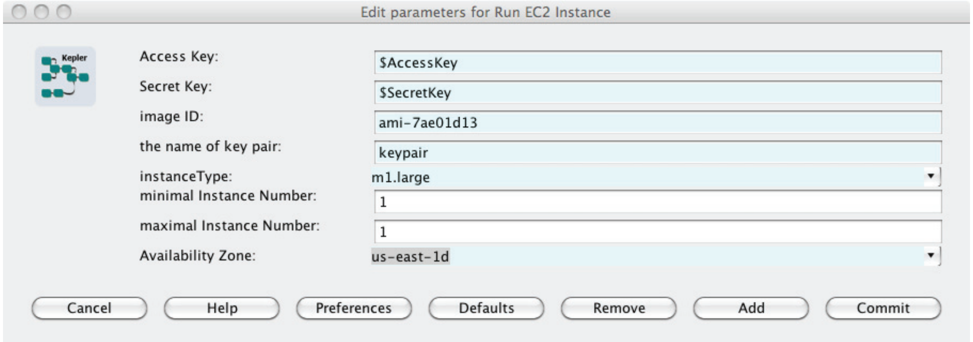
1877-0509 © 2012 Published by Elsevier Ltd.
doi:10.1016/j.procs.2012.04.179

programs directly without a separate installation. A complex application may have to execute toolkits on multiple AMIs. Workflow systems could provide a way for scientists to utilize these available tools by scheduling tasks with proper dependency control. The tasks include Cloud instance management (*e.g.*, initiation and termination), data transfer between Cloud instances, and program execution on Cloud instances. Another new requirement is data-intensive workflow application execution on the Cloud. For data-intensive applications, localities of data and programs are important for the overall performance. With virtualization and other techniques, workflow applications could get both data locality and program locality. We will discuss this in more detail in Section 3.

In this paper, we present our early work utilizing Amazon EC2 for the Kepler scientific workflow System[5] [2, 3]. The rest of this paper is organized as follows. In Section 2, we describe our Kepler EC2 Actors and AMIs to to interact with Amazon Cloud. Different usage modes of these Actors and AMIs are discussed in Section 3. Two workflow usecases in bioinformatics domain are demostrated in Section 4. In Section 5, we discuss our conclusions and plans for future work.

2. Kepler Amazon EC2 Actors and Amazon Machine Images

Kepler EC2 Actors. We have implemented an EC2 module in Kepler which contains a set of EC2 actors. These actors can be used to manage EC2 virtual instances in Amazon Cloud environment and attach EBS Volumes. Users can use these actors to start, stop, run or terminate EC2 instances. Through these EC2 actors and other actors in Kepler, users can easily build workflows to run their applications on EC2 Cloud resources. One EC2 workflow can link a variety of applications residing on different platforms by connecting multiple AMIs, *e.g.*, Hadoop[6] and Linux AMIs. Furthermore, through the Kepler user interface, researchers could analyze bottlenecks of their processes and accelerate their execution by switching to better AMI instance types or by activating multiple parallel AMI instances for their tasks.



Edit parameters for Run EC2 Instance

Access Key: $AccessKey
Secret Key: $SecretKey
image ID: ami-7ae01d13
the name of key pair: keypair
instanceType: m1.large
minimal Instance Number: 1
maximal Instance Number: 1
Availability Zone: us-east-1d

Figure 1: Configurations of the Kepler RunEC2Instance actor.

Figure 1 shows the configuration parameters for the *RunEC2Instance* actor. To run an EC2 instance, users need to provide account information including ’Access Key’, ’Secret Key’ and credential ’Key Pair’ file. ’Image ID’ parameter indicates the AMI ID for the instance. Users can also configure ’Instance Type’, ’Availability Zone’, and ’Minimal Instance Number’ for the instance they want to run. Using these configuration parameter settings, the RunEC2Instance actor interacts with Amazon EC2 Web service in order to get configured instances. If the actor can successfully get an instance, the information of the instance, *e.g.*, URL and ID, will be sent out through its output. Otherwise, the actor will pop up error message dialogue explaining why it cannot run successfully.

Kepler Amazon Machine Images and EBS Volumes. We created a few demonstrative Amazon Machine Images and EBS Volumes for Kepler. The Images and Volumes contain the Kepler system, demonstrative Kepler workflows and third-party bioinformatics tools including the BLAST tool [4]. The difference between Kepler Images and Volumes is that Kepler Images can be used directly to run virtual instances, while Kepler Volumes have to be attached

[5] Kepler website: http://kepler-project.org/, 2012.

[6] Hadoop Project: http://hadoop.apache.org, 2012.

to running instances. These Kepler Images and Volumes can be used to provide scalable and virtualized workflow execution engine. For example, we discussed how to build a virtual Hadoop and Oracle Grid Engine[7] cluster with an attached Kepler Volume to accelerate certain Kepler workflow executions in [5]. Virtual cluster could also be built directly based on a Kepler Image using toolkits like StarCluster[8] .

3. Usage Modes

The Kepler EC2 actors and Images introduced in Section 2 could be used in different usage modes. We explain the requirements and capabilities of each usage mode based on example scenarios below.

1) Kepler EC2 Actors + Third Party AMIs : If users want to access tools on third party AMIs, such as the Bio-Linux Image, they can build workflows to access the related AMIs using the EC2 actors. Some other actors, *e.g.*, SSHExecution and SSHFileCopier, are also needed to transfer data and call third party programs remotely. Dependencies can be easily depicted in the workflow. An example for this usage mode is discussed as the first usecase in Section 4.

2) Kepler EC2 Actors + Kepler AMI: If users want to run their existing local workflows on EC2, they can build an auxiliary workflow with EC2 actors to call their existing workflows on the virtual Kepler EC2 instance. For example, a user might have a workflow working locally, and want to run it on a faster compute node. In this usage mode, the user can find a proper instance type provided by Amazon for the Kepler AMI and configure it in the EC2 actors of the auxiliary workflow. The auxiliary workflow will initiate a Kepler virtual instance for this instance type from the Kepler AMI, upload his/her original workflows to the instance. Then the auxiliary workflow can remotely call the Kepler execution engine to execute uploaded workflows on the instance. After the execution on the instance, the results can be copied back to local machine.

3) Kepler EC2 Actors + Kepler AMI + Third Party AMIs: If users already have workflows for the first usage mode, they might want to run them via a virtualized Kepler execution engine. An auxiliary workflow can be built to initiate a Kepler virtual instance from the Kepler AMI, upload the existing workflows and run them on the instance. Then the uploaded workflows will use the EC2 actors in the workflows to launch other instances and run tools there. This usage mode can be seen as a special combination of the above two. Particularly in this mode, user account information needs to be uploaded to the Kepler virtual instance to launch other virtual instances. An example for this usage mode is discussed as the second usecase in Section 4.

4) Virtual Cluster based on Kepler AMI: This mode is suitable for distributed parallel workflow execution on the Cloud. Amazon EC2 provisions 'Cluster Compute Instance' type and there are toolkits like StarCluster to help setting up virtual clusters based on the Kepler Image. Using such toolkits, a virtual cluster can be generated with Kepler installed on each node and fast inter-connection. This usage mode fits well with the framework for distributed data-parallel workflow execution in Kepler [6]. By utilizing distributed data-parallel execution engines like Hadoop, data can be partitioned and distributed on the virtual cluster. So with both data and Kepler execution engine located locally on each compute node, the program and data transfer among nodes could be minimized and better performance could be achieved for data-intensive workflow applications.

Besides the above typically usage modes, there could be other usage modes based on particular requirements. For instance, a user could launch Kepler virtual instances using the EC2 toolkit and directly login to the instances to run workflows.

4. Bioinformatics Usecases

To validate the proposed work in scientific applications, two experimental bioinformatics workflows running the BLAST tool [4] are shown in Figures 2 and 3. The BLAST tool detects the similarities between biological sequence datasets, and is one of the most widely used tools in bioinformatics.

[7] Oracle Grid Engine System: http://www.oracle.com/technetwork/oem/grid-engine-166852.html, 2012

[8] StarCluster Toolkit: http://web.mit.edu/star/cluster/, 2012.

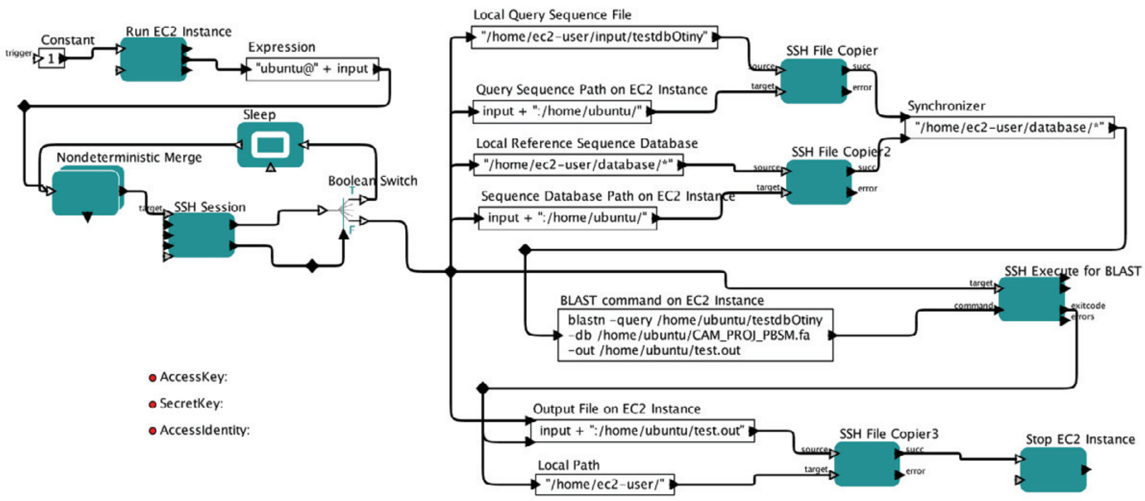

Constant
trigger
1
Run EC2 Instance
Expression
"ubuntu@" + input
Sleep
Nondeterministic Merge
target
SSH Session
Boolean Switch
T
F
Local Query Sequence File
"/home/ec2-user/input/testdbOtiny"
Query Sequence Path on EC2 Instance
input + ":/home/ubuntu/"
Local Reference Sequence Database
"/home/ec2-user/database/*"
Sequence Database Path on EC2 Instance
input + ":/home/ubuntu/"
SSH File Copier
source
target
succ
error
SSH File Copier2
Synchronizer
"/home/ec2-user/database/*"
SSH Execute for BLAST
target
command
exitcode
errors
BLAST command on EC2 Instance
blastn -query /home/ubuntu/testdbOtiny
-db /home/ubuntu/CAM_PROJ_PBSM.fa
-out /home/ubuntu/test.out
AccessKey:
SecretKey:
AccessIdentity:
Output File on EC2 Instance
input + ":/home/ubuntu/test.out"
Local Path
"/home/ec2-user/"
SSH File Copier3
Stop EC2 Instance

Figure 2: A Kepler workflow running BLAST on a third-party virtual instance.

Figure 2 shows a Kepler workflow that runs BLAST on EC2 via the first usage mode in Section 3. The workflow first runs a virtual instance from the Bio-Linux Image using the RunEC2Instance actor. Because EC2 virtual instance might take a while to be ready for user access, the workflow uses a loop to monitor its accessibility. After the virtual instance is accessible, two SSHFileCopier actors are employed to copy the reference database and query sequence file to the Bio-Linux instance. Then BLAST tool is executed on the instance via a SSHExecute actor. After the execution is done, its output is copied back to the local machine via another SSHFileCopier actor.

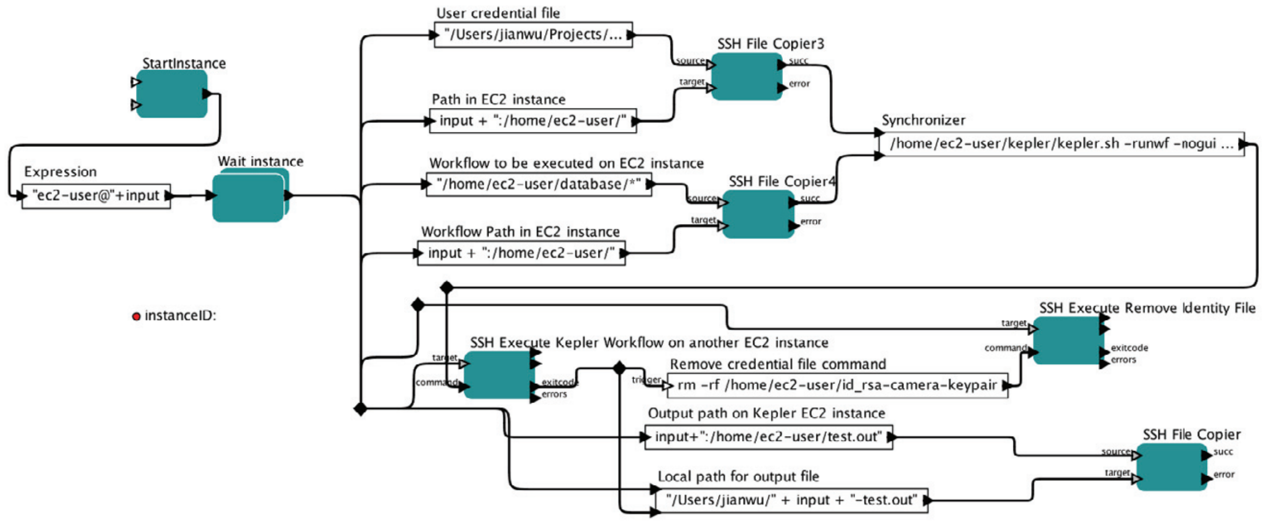

User credential file
"/Users/jianwu/Projects/...
StartInstance
Path in EC2 instance
input + ":/home/ec2-user/"
SSH File Copier3
source
target
succ
error
Synchronizer
/home/ec2-user/kepler/kepler.sh -runwf -nogui ...
Expression
"ec2-user@"+input
Wait instance
Workflow to be executed on EC2 instance
"/home/ec2-user/database/*"
SSH File Copier4
Workflow Path in EC2 instance
input + ":/home/ec2-user/"
instanceID:
SSH Execute Remove Identity File
target
command
exitcode
errors
SSH Execute Kepler Workflow on another EC2 instance
Remove credential file command
trigger
rm -rf /home/ec2-user/id_rsa-camera-keypair
Output path on Kepler EC2 instance
input+":/home/ec2-user/test.out"
SSH File Copier
Local path for output file
"/Users/jianwu/" + input + "-test.out"

Figure 3: A Kepler workflow running BLAST on a Kepler virtual instance and a third-party virtual instance.

Figure 3 shows a Kepler workflow that runs BLAST on EC2 via the third usage mode in Section 3. First, this workflow starts a virtual instance of a Kepler AMI. Then similar to the above workflow, it needs to wait until the virtual instance is accessible. After the virtual instance is running, user credential file to use EC2 resources and the workflow shown in Figure 2 are uploaded to the Kepler EC2 instance. The credential file copy is necessary to start another virtual EC2 instance. Then the Kepler engine on the Kepler instance is run in batch mode to execute the uploaded workflow on another EC2 instance based on the Bio-Linux Image. After the execution, the uploaded user credential file is deleted for account security. In the end, the output file is downloaded to the local machine.

We note that domain scientists do not have to build workflows from scratch like the workflows discussed above for all their requirements. We can build generic workflow templates or compose the above workflows into composite actors, only leaving parameters, such as program name and data path, for scientists to configure for their different requirements.

5. Conclusions and Future Work

Utilizing advantages of Cloud computing, scientific workflows could serve scientific application easier or more efficiently. In this paper, we discuss our early Cloud experiments with the Kepler scientific workflow system, which includes our EC2 actors, Kepler AMI and their usage discussion. Through bioinformatics usecases, we demonstrate how to use our Kepler EC2 actors and Images to coordinate Cloud resources in a workflow and how to execute workflows on virtual Cloud resources.

The Kepler EC2 actors are in a separate module in the Kepler repository. We will release this module in a new Kepler suite. Along with the release, the Kepler Images and EBS Volumes will also be updated. We will publish the information of the Images and Volumes in the Kepler project website.

With the promising results so far, we will improve our work in the future in several directions. First, we will compare the different usage modes in Section 3 through experiments. We also plan to study how to optimize data-intensive workflow execution on EC2.

6. Acknowledgments

The authors would like to thank the rest of Kepler and CAMERA teams for their collaboration. This work was supported by NSF SDCI Award OCI-0722079 for Kepler/CORE, NSF ABI Award DBI-1062565 for bioKepler, the Gordon and Betty Moore Foundation award to Calit2 at UCSD for CAMERA and an SDSC Triton Research Opportunities grant. We also thank the AWS in Education Research Grants from Amazon.com, Inc for EC2 usage credit.

References

[1] G. Juve, E. Deelman, Scientific workflows and clouds, Crossroads 16 (2010) 14–18. doi:http://doi.acm.org/10.1145/1734160.1734166.

[2] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludaescher, S. Mock, Kepler: An extensible system for design and execution of scientific workflows, in: Proceedings of 16th International Conference on Scientific and Statistical Database Management (SSDBM 2004), Santorini Island, Greece, 2004.

[3] B. Ludaescher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. A. Lee, J. Tao, Y. Zhao, Scientific workflow management and the Kepler system, Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows 18 (10) (2006) 1039–1065.

[4] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, D. J. Lipman, Basic Local Alignment Search Tool, Journal of Molecular Biology 215 (3) (1990) 403 – 410. doi:10.1016/S0022-2836(05)80360-2.

[5] J. Wang, P. Korambath, I. Altintas, A physical and virtual compute cluster resource load balancing approach to data-parallel scientific workflow scheduling, in: Proceedings of 2011 IEEE World Congress on Services (SERVICES 2011), 2011, pp. 212 –215. doi:10.1109/SERVICES.2011.50.

[6] J. Wang, D. Crawl, I. Altintas, A framework for distributed data-parallel execution in the Kepler scientific workflow system, in: Proceedings of 1st International Workshop on Advances in the Kepler Scientific Workflow System and Its Applications, ICCS 2012, 2012.