Integrated Scientific Cloud Computing Workflows

SC2IT: A Toolkit to Bring Novel Computational Methods to Non-specialists

K. Jorissen, W. Johnson, J.J. Rehr
Department of Physics, University of Washington, Seattle, USA
E-mail: kevinjorissen.pdx@gmail.com

***Abstract*—Scientific Cloud Computing (SCC) applies the advantages of Clouds to computational science. We present the SC2IT Scientific Cloud Computing tools that abstract complex computational workflows and allow state-of-the-art scientific high-performance computing software to be distributed in a user-friendly manner. We present Linux and Java "SC2IT" toolsets to manage virtual cloud clusters and implement scientific workflows. We demonstrate several GUIs that implement the "SC2IT" interface for materials science applications. In particular, the ClusterManager is a general-purpose SCC GUI to which new workflows can be added easily.**

Keywords-Cloud Computing; High-Performance Computing; Scientific Computing; Workflows; User Interface Design

I. INTRODUCTION

High-Performance Computing (HPC) has become indispensable to many areas of research. Computational studies are now done in widely varying fields, including economics, genetics, and computational linguistics. The current paper showcases applications in materials science. HPC users have traditionally needed to make a significant investment of time and money. They would either acquire a Unix or Linux cluster requiring configuration and maintenance; or they would buy time at a supercomputing center and learn to work in its specialized environment. To aggravate the steepness of the learning curve, state-of-the-art codes in computational materials science (MS) and other fields often offer limited user-friendliness compared to commercial software. They may require challenging compilation and installation, or may operate from a Linux command line and depend on poorly formatted text files. Such difficulties have traditionally limited computational MS to "specialists". This is unfortunate, because many researchers who are not specialized computational scientists could benefit from direct access to state-of-the-art theoretical simulations. E.g. many experimentators in MS rely on theoretical modeling for interpretation of their measurements. We believe that innovative theoretical methods could permeate the field of MS faster if 1/ their software implementation were easier to use, and 2/ HPC infrastructure were more accessible. This insight carries over to computational research beyond MS also.

This outlines the central problem that this paper discusses: how can we bring the best available computational science to the researchers who need it? Keeping in mind that the latter are likely not computational specialists; that novel methods can be expected to have non-trivial hardware requirements; and can be slow to acquire user-friendliness.

Cloud Computing (CC), which is very successful in business and personal computing, has the potential to address these challenges for scientists. CC stands for the on-demand availability of scalable, virtualized computing resources. In the CC paradigm, the researcher no longer purchases hardware. Instead, she rents as much of it as she needs for the time that she needs it. Such virtualized on-line cloud resources are now available and capable of handling state-of-the-art HPC workloads.[1-5] An additional benefit lies in the relationship between the developer of a scientific code and its users. While a developer would traditionally rely on users to install and configure the code (potentially including an environment of dependencies), he can now pre-install and optimize the entire software ecosystem in the cloud. It is a great advantage for the user to no longer have to learn and perform configuration tasks, and it helps guarantee the production of valid results. Thus the adoption threshold to the end user can be lowered dramatically and the scope of computational resources available to a typical researcher can be expanded.

Within this "Scientific Cloud Computing (SCC or SC2)" paradigm [5] we created a "virtual platform" or blueprint for the virtual machines in the cloud cluster. This blueprint (SC2VP) contains libraries, compilers, a parallel computing environment, and preconfigured applications typically useful for materials scientists. E.g. these applications can calculate structural and electronic properties of materials. Similar virtual platforms can be created for other fields of research with a selection of codes representative to that field. Secondly, we designed a set of interface tools (SC2IT) for the Amazon EC2 cloud [6] that configure virtual cloud computers to form a "cloud cluster" suitable for parallel calculations. Third, we tested the performance of this setup to prove that HPC calculations for materials science can be done efficiently in a cloud environment. Finally, we embedded the interface and blueprint in a GUI environment that enables MS end users to perform specific SCC calculations with a few mouse clicks. The same approach can be followed for SCC enhancement of GUIs for other fields of research.

In this paper we describe recent developments of our platform. In Section II we present the latest version of our interface tools, which are now available as a Java library. This enables other developers to add SCC functionality to existing GUIs in a cross-platform way.

To truly bring advanced science to a broad class of end users, another step is necessary beyond launching a parallel MS program on a cloud cluster. The development of novel scientific software is often modular: computational scientists link existing codes together and combine them with new developments to produce state-of-the-art results. For example, we use a popular Density Functional Theory code to calculate the dynamical matrix of a material; secondly, a new module to next derive vibrational properties; and thirdly, an existing spectroscopy code to finally calculate an X-ray spectrum incorporating the vibrational information. Each of these building blocks may be written in different programming languages and have widely varying requirements, dependencies, I/O conventions, and scaling. The resulting installation and usage threshold is often insurmountable for non-specialist users, who may not "catch on" until a more streamlined single software package has been produced years later. Academic scientists do not currently receive the needed incentives and support to make development of user-friendly software a priority.

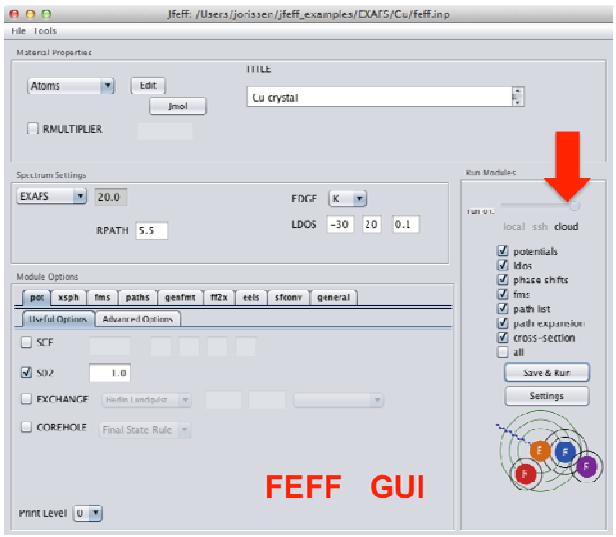

Figure 1 JFEFF, the Graphical User Interface for the X-ray spectroscopy simulation code FEFF9 [8]. Note the slider marked by the red arrow, which lets a FEFF user decide where the calculation should be executed: on the local machine, on a traditional cluster, or on the EC2 cloud. After the slider is adjusted, the user simply clicks "Save and Run". The JFEFF GUI can automatically drive the entire cloud calculation from there, including the launch of a cloud cluster, the two-way transfer of files corresponding to input and to results, controlling the FEFF calculation on the remote cloud cluster, and terminating the cloud cluster when it is no longer needed. The user does not need to understand any of the details of this process. The SC2IT Java library implements all the necessary functionality. AWS account details are provided in a Settings window (not shown here)

Clearly there is a need for streamlined, abstracted workflows that relieve the end user from this burden of complexity. Again, cloud computing allows the developer or distributor to fully shoulder the setup and address the complexity in a fully controlled environment. In Section III we present a new Graphical User Interface (GUI) that implements our JSC2IT toolkit and can easily control defined workflows, so that users can benefit from complex workflows being executed on the cloud without having to actively address their complexity.

II. The JSC2IT Toolkit

Launching a set of virtual machines from a cloud provider is easy but does not produce a fully functional HPC cluster. Further configuration tasks are required. The SC2IT [5] is a collection of interface tools for the Amazon EC2 cloud that launches and configures individual cloud instances to create such a parallel cluster. In its second function it simplifies interaction with the cloud cluster by managing many of the technical aspects of addressing this collection of remote virtual machines. Using SC2IT a user has access to simple commands for essential tasks, and the user experience closely resembles that of a regular Linux computer. The SC2IT interface is currently tailored to the Amazon Web Services (AWS) API for the EC2 cloud [6]. This design choice will make generalization with another cloud provider straightforward as AWS is currently an industry standard.

SC2IT exists in two forms. First is a bash implementation for use from the Linux or Mac OS X Terminal. This version gives more flexibility for users who prefer a command line environment for low-level control of the cloud cluster. Second is a Java implementation designed to be incorporated in Graphical User Interfaces. This allows UI developers to provide user-friendly cloud computing interfaces. As an example, we show the JFEFF interface for the FEFF9 code in Fig. 1. Aside from linking the JSC2IT.jar library, only minimal changes to the pre-existing GUI were required to endow it with SCC functionality.

The bash version of the toolset follows a procedural structure, with programs equivalent to essential Linux instructions ("get", "put", ...; Table I). The Java SC2IT is a Java library and is implemented along an object-oriented scheme. These two versions offer identical functionality. What's more, they are able to work side-by-side. A user can launch and manage a cloud cluster from a Java-based interface, and later access it with the bash interface for troubleshooting calculations or administrator tasks such as installing additional software.

The Java version of the SC2IT toolset contains approximately thirty classes, sorted a handful of packages, and builds upon the official AWS SDK.[7] Tables II-IV give the most important packages and the classes they contain. SC2IT-Java comes with documentation that provides more information on usage (including technical notes on the so-called "credential files" that identify a user to AWS for security and billing purposes), and javadoc are also provided. The Sandbox class is essentially a demo of SC2IT and can also be of use for debugging purposes.

TABLE I. SC2IT for bash

Tools in the SC2IT interface for bash		
Name of tool	***function***	***analog***
sc2launch N	Launch cluster with N instances	boot
sc2connect	Connect to the cluster	ssh

Tools in the SC2IT interface for bash		
Name of tool	***function***	***analog***
sc2put	Upload files to the cluster	scp
sc2get	Download files from the cluster	scp
sc2list	Display the active clusters	top
sc2terminate	Terminate the cluster	shutdown
sc2desc	Describe the active clusters	ps
sc2run	Launch a job on a cluster	.
sc2connectr	Connect to a cluster as root	ssh
sc2load	Monitor load in a cluster	loadavg
sc2usage	Monitor CPU usage in a cluster	top

TABLE II. THE SCC PACKAGE

Class	***Description***
EC2Cluster	Primary class. API for all method calls in Table V.
ClusterManager	Tracks list of active clusters. Constructs EC2Cluster objects.
Sandbox	Simple implementation as example and for debugging.

TABLE III. THE SCC.SSH PACKAGE

Class	***Description***
Console	Connect to a cloud node and open the terminal window
ParallelSetup	Configures all nodes simultaneously
NodeUpdater	Reconfigure existing cluster when number of nodes changes.
ScpTask	Transfer setup files to cloud nodes and configure SSH channels appropriately.
ExecTask	Run setup tasks on the cloud server
Workflow	Defines a workflow of codes and I/O that together form one calculation

TABLE IV. THE SCC.UTIL PACKAGE

Class	***Description***
ClustConfig	Manages cluster settings (AWS/EC2 identifiers, IP addresses, ...)
ScriptWriter	Writes setup scripts and data files for cluster.
NewClusterSpecs	Bundles specifications for EC2 launch requests.
Machine	Defines Cluster profiles for specific tasks, e.g. AMI and instance types suitable for FEFF9.
ClusterResult	Output stream handler for AWS requests.
ClustAuth	Bundles AWS credentials (e.g. SSH keys and credentials) required for EC2 access.

TABLE V. METHODS IN THE EC2CLUSTER CLASS

Method	***Description***
EC2Cluster	Launch new cluster with n nodes ; or load existing one.
AddNodes (n)	Add n nodes to a cluster.
Connect	Starts a terminal window via SSH. Login as root or regular user.
DeleteNodes (n)	Delete n nodes from a cluster. Update configuration files accordingly.
Run	Execute a command on a cloud cluster.
Get (rFile, lFile)	Download remote file rFile to local lFile.
GetConfig	Return EC2ClustConfig object.
GetEC2Client	Return AmazonEC2Client object.
GetName	Return name of a cluster.
Put (lFile, rFile)	Upload local file lFile to rFile on a cluster.
Terminate	Terminate a cluster and clean up.

III. WORKFLOWS IN THE CLUSTERMANAGER

A. Workflows

MS applications typically start from a small set of input, e.g. text files specifying options, atomic coordinates, and some properties such as the electron density of the material. They often require very large amounts of memory and CPU, producing large or very large amounts of intermediary scratch data and moderate to large amounts of output data. Many modern MS codes need to be run in parallel on a cluster or supercomputer.

Workflows handle the required input, execution step, and desired output of each code/step in the workflow. The JSC2IT contains a Workflow Class that makes it very easy to define these ingredients. The JSC2IT Toolkit has simple commands for the upload/download of files between the end user's computer and the cloud cluster, and for the execution of a command on the cloud cluster. The stdout/stderr of tasks on the cloud cluster can be directed to the end user's computer. If the tasks have defined error states, these can be monitored and taken into account. Programming new workflows using JSC2IT is therefore easy and concise.

B. *The ClusterManager GUI*

ClusterManager, shown in Fig. 2, is a GUI built around the JSC2IT library. It contains three principal elements. First, an interface for registering an Amazon Web Services (AWS) account with ClusterManager and setting up the "credentials" required for accessing the EC2 cloud. Second, the main interface window showing all cloud clusters active for the current user, and the elementary functions of JSC2IT: Upload and Download of files, Execution of a command, Launching and Terminating a cluster, Opening a terminal, etc. Third, a section of the main interface window containing integrated workflows. Here a user can start a complex task using a single button. ClusterManager executes the attached workflow, gathering the required input files from the working directory on the end user's computer, bringing them to the cloud, executing multiple scientific codes present on the cloud cluster, and bringing back the desired results as specified in the workflow. Additionally ClusterManager can monitor the wellbeing of the workflow e.g. by checking for error files and messages, or watching the usage of resources; or bring intermediary results back and allow or request the user to edit intermediary input files.

IV. CONCLUSION

We have presented a Scientific Cloud Computing (SCC) platform that enables researchers to use on-demand HPC clusters in the Amazon EC2 cloud. Additionally, developers can use it to build state-of-the-art applications that are easy to use for researchers lacking expertise in parallel computing or access to conventional HPC resources. The main elements of our new platform include a virtual cloud computer blueprint or AMI, which contains preinstalled and optimized scientific codes and utilities. Next, we offer the SCC Interface Tools (SC2IT), available as bash programs for work in a command-line environment and alternatively as a Java library for GUI building. Several widely used MS codes are already equipped to perform user-friendly cloud calculations due to enhancing their dedicated GUIs with the SC2IT, for example the JFEFF interface for FEFF9. In this way FEFF users can expediently and conveniently obtain results from demanding calculations in the Cloud without any technical effort. Finally, we presented the ClusterManager, a general-purpose SCC GUI that allows developers to easily define new Workflows that include several scientific codes, as is typical for novel developments in computational science. Thus their users will be able to execute complex workflows without having to address the underlying complexity that would be prohibitive to most non-specialists. This paradigm will be of value to other areas of research to which CC is appropriate.

ACKNOWLEDGMENT

UW SCC is supported by NSF grant OCI 1048052. EC2 computing resources are generously contributed through an Amazon Web Services educational research grant. The FEFF project acknowledges support by DOE BES grant DE FG03-97ER45623. Our thanks go to Deepak Singh at AWS for valued support, and to J. Gardner at UW for feedback and discussion.

REFERENCES

[1] Vaquero, L. M., Rodero-Merino, L., Caceres, J., and Lindner, M., "A Break in the Clouds: Towards a Cloud Definition", Comp. Comm. Rev. 39 (2009) 50.

[2] Masud, R., "High Performance Computing with Clouds", http://ix.cs.uoregon.edu/~raihan/HPC_with_Clouds_Raihan_Masud.pdf (2011)

[3] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., and Zaharia, M., "Above the Clouds: A Berkeley View of Cloud Computing", EECS Dept., Univ. of California, Berkeley, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html (2009).

[4] Fenn, M., Holmes, J., and Nucciarone, J., "A Performance and Cost Analysis of the Amazon EC2 Cluster Compute Instance", Research Computing and Cyberinfrastructure Group, Penn State Univ., http://rcc.its.psu.edu/education/white_papers/cloud_report.pdf (2011).

[5] K. Jorissen, F.D. Vila, J.J Rehr, "A high performance scientific cloud computing environment for materials simulations", Comp. Phys. Comm. 183 (2012) 1911.

[6] Amazon Elastic Compute Cloud, http://aws.amazon.com/ec2/ (2012).

[7] AWS SDK for Java, http://aws.amazon.com/sdkforjava/ (2012).

[8] Rehr, J. J., Kas, J. J., Vila, F. D., Prange, M. P., and Jorissen, K., "Parameter-free calculations of X-ray spectra with FEFF9", Phys. Chem. Chem. Phys. 12 (2010) 5503.

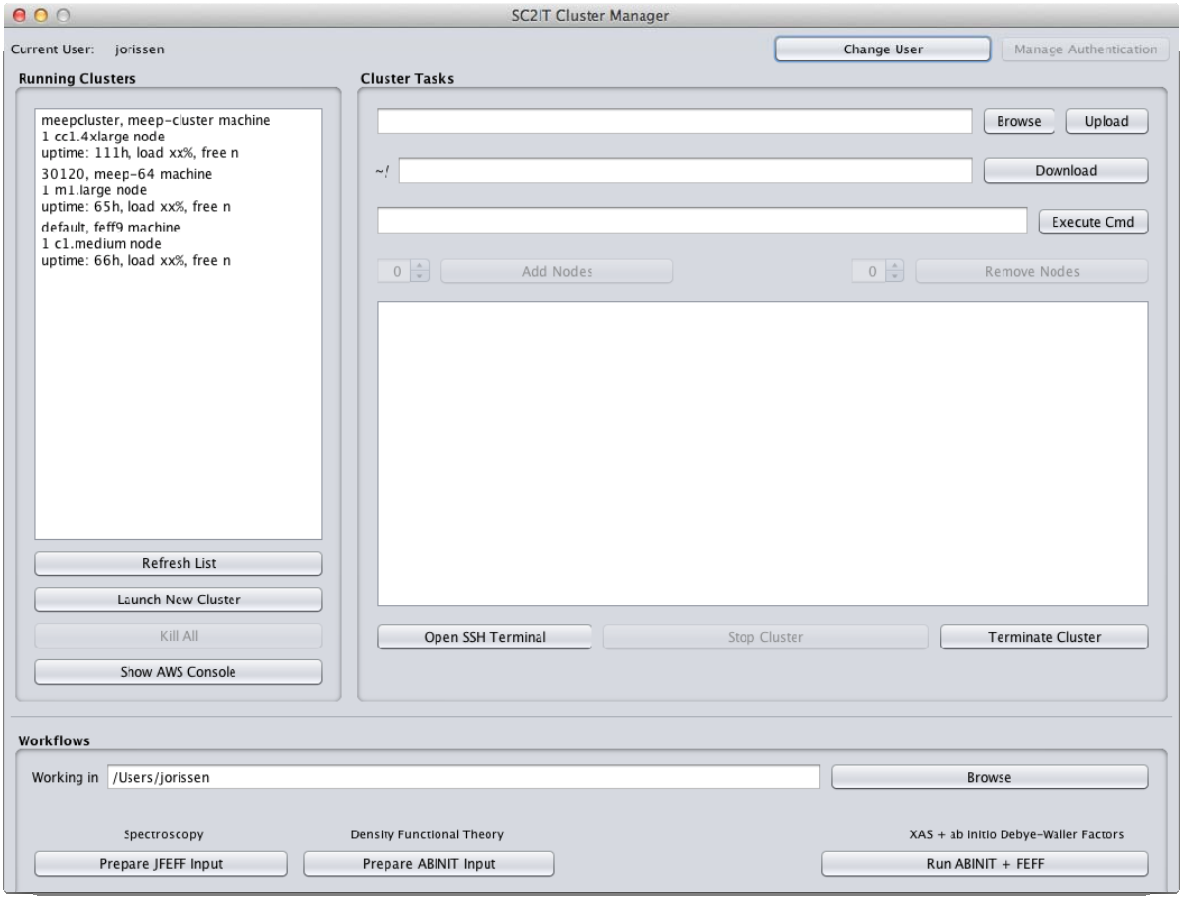

Figure 2 The main window of the ClusterManager GUI. The left column lists the EC2 cloud clusters that are active for the current user. The right-hand panel offers basic tasks for the selected cluster, such as up/downloading files and executing commands. It also contains a stdout/stderr window. Finally, the bottom panel loads a workflow. In this case, the "ABINIT+FEFF" workflow takes appropriate input files from the working directory, copies them to the cloud cluster, and executes three consecutive codes: the ABINIT code to calculate the dynamical matrix of the material; the DMDW program to derive the vibrational properties of the material; and finally the FEFF9 code [8] to arrive at the X-ray Absorption Spectrum of the material. Important output files are then copied back to the user's computer. The cloud cluster can then safely be terminated. The two leftmost buttons, labeled "Prepare JFEFF Input" and "Prepare ABINIT Input", help the user prepare the required input files. To implement a new workflow one needs to change the definitions of the required files and required execution steps in the Java code and adjust the interface elements of this lower panel. Naturally, the cloud cluster must be equipped with the proper compiled applications to receive the workflow tasks.