International Journal of Cloud Computing and Services Science (IJ-CLOSER)
Vol.1, No.4, October 2012, pp. 156~172
ISSN: 2089-3337

The Design and Implementation of the VMD Plugin for NAMD Simulations on the Amazon Cloud

Adam K. L. Wong*, Andrzej M. Goscinski*
* School of Information Technology, Deakin University

Article Info

Article history:

Received Aug 08th, 2012
Accepted Sept 23th, 2012

Keyword:

Cloud
Molecular Dynamics
High Performance Computing
VMD/NAMD Simulations
Amazon EC2

ABSTRACT

High Performance Computing (HPC) clusters have enabled Molecular Dynamics (MD) simulations of biological systems for a sufficiently large complexity and long simulation period. VMD and NAMD are two major MD simulations software packages, which can work together for mining structural information of bio-molecules. However, HPC clusters are in many cases unaffordable and when accessed force users to wait long time for the results of their computations. Recent years, clouds have provided HPC clusters on demand that allow users to benefit from their flexibility, elasticity, provision services on demand and flexible pay-as-you-go payment method. Although cloud computing promises to provide seamless access to HPC clusters through abstractions of services that hide the details of the underlying software and hardware infrastructure, users without in depth computing knowledge are still forced to cope with many low level details. Therefore, we have designed and developed a software plugin of VMD, which can provide an integrated framework for VMD and NAMD to be executed on Amazon EC2 cloud. The proposed Amazon EC2 Plugin for VMD frees users from performing many tedious computing tasks such as launching, connecting and terminating Amazon EC2 compute instances; configuring a HPC cluster; and installing middleware and software applications before the system is readily available for any scientific investigation. Thus, it allows VMD/NAMD users to spend less time getting applications to work on HPC clusters but more time on researching.

Corresponding Author:

Adam K. L. Wong,
School of Information Technology,
Deakin University,
Waurn Ponds, VIC3216, Australia.
Email:aklwong@deakin.edu.au

1. INTRODUCTION

In the pre-cloud era, organizations with high performance computing (HPC) needs have been required to fund expensive, in-house compute clusters by either purchasing and maintaining them or paying an inflexible amount of subscription fee to HPC service providers. The demand for HPC clusters often exceeds the capacity of many organizations because of these upfront investments. Thus, many projects are cut altogether or application jobs wait in long queues to access shared resources. Recently, some public cloud vendors including the Amazon Elastic Compute Cloud (Amazon EC2) [2] have provided computer instances, which are specifically designed for running HPC applications and other demanding network-bound applications. Thus, businesses and researchers now have access on demand to the high performance computing capabilities that they need with a very flexible pay-as-you-go pricing method. A HPC cloud gives users the opportunity to test and run their parallel applications in the cloud at a price and performance level

that can leverage their budgets and computation requirements. Furthermore, it also provides the ability to scale on-demand as the users' requirements change.

Many scientific research areas require running computationally demanding software in HPC systems such as the molecular dynamics simulation of bio-molecules in life sciences research. Molecular dynamics simulation is an important tool for biological scientists to study the physical basis of the structure and function of proteins (bio-molecules) since the internal motions of individual atoms play an essential role in the functional mechanism of proteins in living organisms. For example, NAMD [14] is a parallel molecular dynamics program designed for high-performance simulation of large bio-molecular systems. This parallel software program can be scaled well to run on a large number of processors in computer clusters or a large number of cores in graphics processing unit (GPU) hardware. NAMD usually works together with its sister molecular graphics program VMD [23], which provides easy-to-use tools that explore structure information of bio-molecules. In a nutshell, VMD is a molecular visualization program for simulation preparation, displaying, animating, and analyzing large bio-molecular systems using 3D graphics and built-in scripting. Both of NAMD and VMD are distributed free of charge with source codes and have supports on various OS platforms such as Linux, Windows and Mac OS.

Although cloud technologies promise to provide seamless access to HPC clusters through abstractions of services that hide the details of the underlying software and hardware infrastructure, users without in depth computing knowledge are still forced to cope with many details in setting up and configuring a HPC cluster, and installing middleware and software applications before the system is readily available for any scientific investigation. To run any NAMD simulation in a public cloud, a typical VMD/NAMD procedure would require a discipline scientist to first, prepare simulation input data in VMD on a host computer; second, prepare a HPC cluster for example in Amazon EC2 where NAMD must also has been made ready; third, transfer the prepared simulation input data from the host computer to the HPC cluster and start the simulation; and fourth, transfer the simulation result back to the host computer for analyzing in VMD. Furthermore, many operations for HPC cluster setup and configuration must be carried out in a Linux command-line platform; a command-line averse scientist would face more inconvenience rather than operating comfortably in the user-friendly GUI environment of VMD. Thus, instead of concentrating on research activities in the discipline, the scientist must also act as a system administrator and to learn in-depth HPC computing knowledge.

An important component included in the VMD package is its plugin interfaces which provide a feature for extending VMD at run-time without the necessity to recompile the program. By using plugins, new functionality or updated code can be dynamically loaded from Tcl/Python scripts and shared object files for Linux/Unix systems, or dynamic link libraries for Windows systems [24]. This also allows third party developers to distribute extensions to the VMD user community. Common scripting extensions, which implement new commands and user interfaces for performing tasks such as structure building, data importing and exporting, simulation setup, molecule visualization as well as result analyzing, can be found in [24].To relive researchers of molecular dynamics from performing tedious computing tasks in an ad-hoc way as listed above, we propose a generic solution for it by integrating VMD, NAMD and Amazon EC2 together into a software plugin of VMD. This Amazon EC2 Plugin for VMD supports NAMD simulations on the Cluster Compute Instances of Amazon EC2. It allows users to 1) create instantly a HPC cluster on Amazon EC2; 2) submit a NAMD parallel simulation from VMD to Amazon EC2; and 3) collect result from Amazon EC2 to the host computer running VMD for post-processing. Additionally, our Amazon EC2 Plugin has also integrated with the Interactive Molecular Dynamics (IMD) plugin [11], which can make a standard simulation interactive and display the simulation in real-time. Most importantly, it hides completely the details of any NAMD simulation deployment in the underling HPC cloud. All of those tasks can be carried out by just a few button clicks.

This paper describes the design and implementation of the Amazon EC2 Plugin that we have developed for VMD; its prototype and key features were presented in [26]. The rest of this paper is organized as follows. Section 2 provides the background of molecular dynamics simulations using VMD and NAMD, and a brief description of Amazon EC2. Section 3 describes the design and implementation of the Amazon EC2 Plugin. A case study of molecular dynamics simulations of Satellite Tobacco Mosaic Virus on the Amazon EC2 via the support of our plugin is detailed and its benefits are discussed in section 4. Finally, conclusion and future work are presented in section 5.

2. BACKGROUND

A molecular dynamics simulation is a computer simulation of physical movements of atoms and molecules in a molecular system. In most cases, the trajectories of atoms and molecules are modelled in the Newton's equations of motion for a system of interacting particles, where forces between the particles and potential energy are defined by molecular mechanics force fields [1]. MD simulations have provided detailed

information on the fluctuations and conformational changes of proteins and nucleic acids [12]. These methods are now routinely used to investigate the structure, dynamics and thermodynamics of biological molecules and their complexes [9].

With continue grows of high performance computing technology, MD simulations are capable of handling very large and complex bio-molecules, e.g., viruses. Also, the simulation time for a bio-molecule can be increased in a timescale of microsecond instead of a timescale of nanosecond or picosecond[1] as in the past. Thus, more functional details of the bio-molecules can be studied nowadays.VMD and NAMD are two major MD simulations software packages, which can work together for mining structural information of bio-molecules. In the following sections, brief descriptions abstracting the key functionalities of VMD and NAMD and an introduction to Amazon EC2 are provided.

2.1. VMD

VMD is developed by the Theoretical and Computational Biophysics Group at the Beckman Institute of the University of Illinois at Urbana-Champaign [20]. It is designed for modelling, visualization, and analysis of biological systems such as proteins, nucleic acids, etc. It can be used to view molecules in the standard of Protein Data Bank (PDB) format and display the contained structure. VMD provides a wide variety of methods for rendering and colouring a molecule: simple points and lines, CPK spheres and cylinders, licorice bonds, backbone tubes and ribbons, cartoon drawings, and others. VMD can be used to animate and analyze the trajectory of a MD simulation. In particular, VMD can act as a graphical front end for an external MD software application such as NAMD by displaying and animating a molecule undergoing simulation on a remote computer.

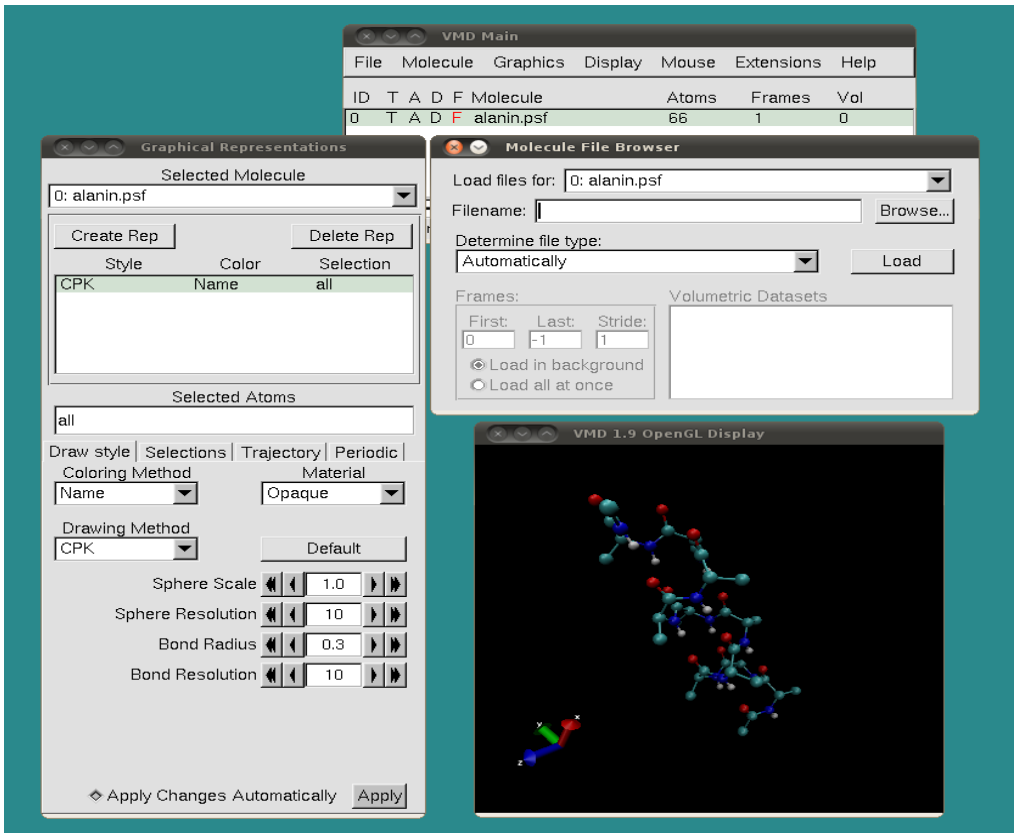

Figure 1. A snapshot of VMD displaying a deca-alanine molecule

Figure 1 shows VMD displaying the 3D structure of deca-alanine, a protein with 66 atoms. When VMD is started, two windows (the VMD Main and VMD OpenGL Display) are first popped up. The VMD Main window provides various functions, extensions and other interfaces to manipulate bio-molecular structures whereas the VMD OpenGL Display window is used to display the bio-molecule being

[1] One picosecond is one trillionth of a second (10-12).

manipulated. The first file needed to be loaded into VMD is the protein structure file, which contains important information about the bio-molecule system such as which atoms are bonded together, what charge they are, and the mass of each atom The second file needed to be loaded into VMD is the Protein Data Bank file, which contains the coordinates of the atoms. These files are loaded via the Molecule File Browser window, which can be popped up by selecting File -> Load Data Into Molecule function from the VMD Main window. To adjust the methods for rendering and colouring a bio-molecule, the Graphical Representations window is used which can be popped up by selecting File -> Graphical Representations function from the VMD Main window.

2.2. NAMD

Same as VMD, NAMD is developed by the Theoretical and Computational Biophysics Group at the Beckman Institute of the University of Illinois at Urbana-Champaign. NAMD is a parallel, object-oriented molecular dynamics program designed for high performance simulation of large bio-molecular systems. NAMD employs the Charm++/Converse parallel runtime system [5] to handle inter-process communication among processes by default. It allows excellent parallel scaling on both massively parallel supercomputers and commodity workstation clusters. Alternatively, NAMD also supports the use of MPI [21] libraries in place of Charm++ for portability related issues.

In order to run any MD simulation, a user needs to prepare the followings:

1. a Protein Data Bank (pdb) file which stores atomic coordinates and/or velocities for the system. The pdb files may be generated by hand, but they are also available in the Protein Data Bank repository at [15] for many proteins.
2. a Protein Structure File (psf) file which stores structural information of the protein, such as various types of bonding interactions. This file must be prepared by the user. Instructions for psf file generation can be found in [4].
3. a force field parameter file which defines the force fields such as bond strengths, equilibrium lengths, etc. for atoms in the system. Parameter files for a given class of molecule could be obtained from CADD [6].
4. a NAMD configuration (conf) file which specifies all the options that NAMD should adopt in running a simulation. This file must also be prepared by the user.

Among the four files as presented above, the NAMD configuration file is used as the input for a NAMD simulation. It specifies the search paths of pdb file, psf file, parameter file as well as other options which define properties of the simulation. The output of a NAMD simulation consists of two major files, one containing the final coordinates (coor) and another containing the final velocities (vel) of all atoms in the simulation. Additional output file such as the dcd file, which stores the trajectory of all atom position coordinates of an animation, can also be produced. Figure 2 shows the execution flow of a NAMD simulation. Once a NAMD configuration file is prepared, the NAMD stimulation can be run on a HPC cluster by invoking the following command:

```
>charmrun ++remote-shell ssh ++nodelist nodelistfile namd2 filename.conf ++p Num
```

where multiple namd2 instances are launched by the charmrun run-time. The option ++remote-shell specifies the remote shell to use in the cluster, option ++nodelist specifies a list of compute nodes to use in the cluster, and option ++p specifies the number of namd processes to be launched.

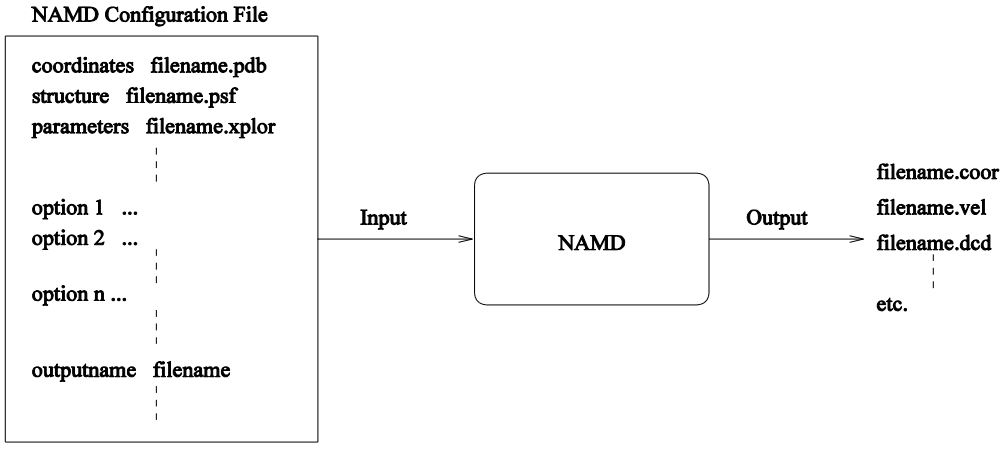

Figure 2. A typical NAMD simulation

2.3. Amazon EC2

The simplest way to use the Amazon EC2 services is by accessing the AWS Management Console [3] with a web-based interface as shown in Figure 3.

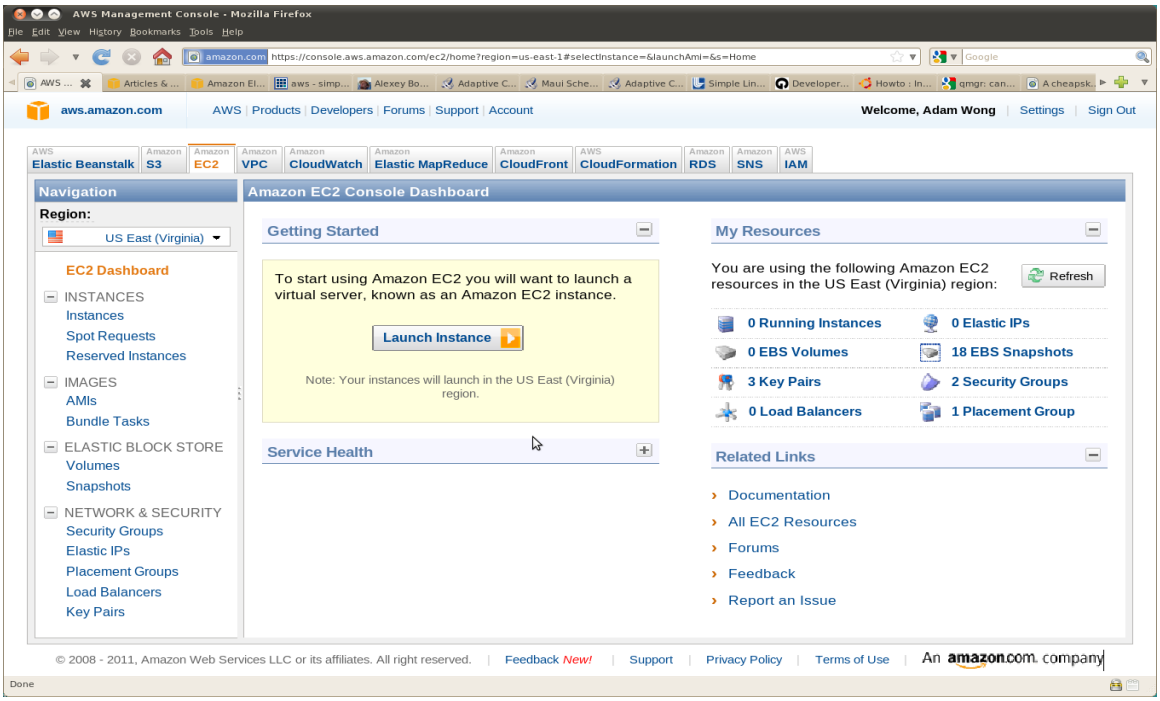

Figure 3. A snapshot of the AWS Management Console

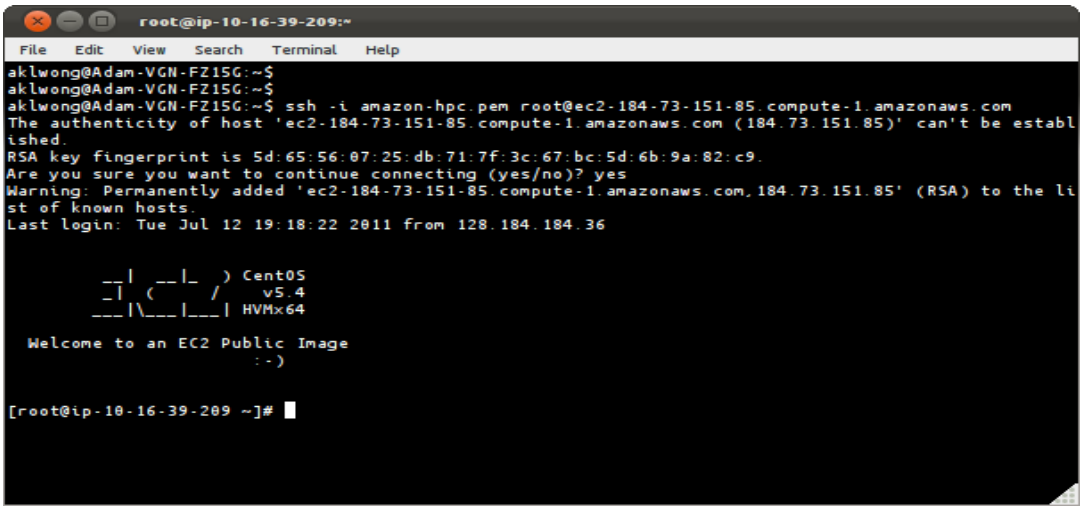

Figure 4. A snapshot of the Command Prompt Shell made on the Running AWS EC2 Instance

After logging on to the AWS Management Console, a user can get started with Amazon EC2 by performing the following tasks:

1. Launch instance (assuming Linux/Unix instance)
    a. Choose an Amazon Machine Image (AMI) for creating a new computer instance.
    b. Create or select a key pair (a security credential), which will be used by the user to securely connect to the instance after it is running.
    c. Create or select a security group, which defines firewall rules for the instance.
    d. Launch the specified number of instances using the selected AMI.
2. Connect to instance (See Figure 4)
    a. From a Linux host, use the ssh (secure shell) command to connect to the Linux instance.
    b. Install software applications, if they are not already available, in the instance.

c. Transfer any files, which are required for the applications to the instance.
d. Run the applications.
e. Transfer any files back to the host if necessary.

3. Terminate instance
   a. In the Management Console, locate the instance(s) in the list of instances on the Instances page.
   b. Confirm to terminate the instance(s).

3. AMAZON EC2 PLUGIN

As presented in Section 2, the current model of adopting public HPC cloud service such as Amazon EC2 to carry out NAMD simulations is quite ad-hoc and could be tedious for the researchers of molecular dynamics who have little background in HPC. On top of the work in launching, connecting and terminating AWS computer instances, they are also forced to deal with many details in setting up and configuring a HPC cluster; and installing middleware and software applications before the system is readily available for any actual scientific investigation. For this reason, we have developed a software plugin for VMD which can provide an integrated framework for VMD, NAMD and Amazon EC2 to work together for exploring structure information of bio-molecules.

3.1. Design

VMD has provided a plugin interface to facilitate dynamic extensions to its core parts. Users are free to extend the VMD functionalities by providing new software plugins without the need to recompile the entire program from scratch. The simplest type of plugins to be created for VMD is Tcl/Tk plugin [25]. Since Tcl is a scripting language, Tcl packages are relatively easy to develop, test and deploy. On the other hand, the Tk toolkit, a graphical user interface library, makes graphical application development become easy. Our Amazon EC2 Plugin for VMD is implemented in Tcl/Tk as a GUI application which provides various functions to automate the tasks of:

1. Create instantly a HPC cluster on Amazon EC2.
2. Shutdown and terminate a HPC cluster on Amazon EC2.
3. Submit a NAMD parallel simulation from VMD at a host computer to Amazon EC2.
4. Integrate with the Interactive Molecular Dynamics (IMD) plugin, which can make a standard simulation running on Amazon EC2 become interactive and the simulation is displayed in real-time.
5. Collect result from Amazon EC2 to the host computer running VMD.

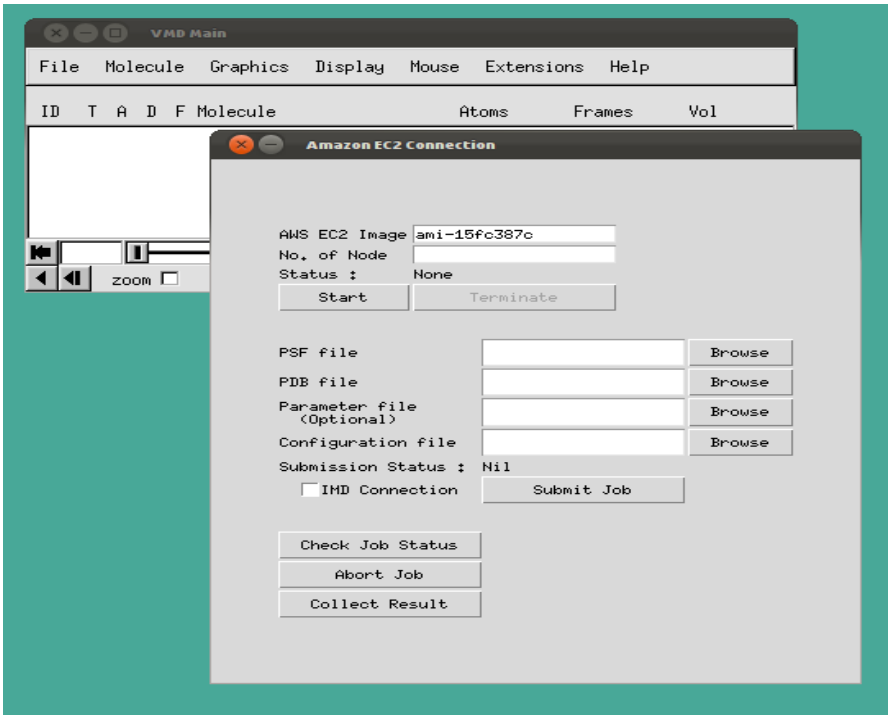

Figure 5. A snapshot of the VMD Amazon Plugin

Figure 5 shows the VMD Amazon Plugin window which can be popped up by selecting Extensions ->Amazon EC2 Connection function from the VMD Main window. The first set of buttons handles HPC cluster creation and termination in Amazon EC2. The second set of buttons handles NAMD job submission and IMD connection to Amazon EC2. Finally, the third set of buttons handles NAMD results retrieval from Amazon EC2. Users do not need to access the AWS Management Console at all.

3.2. Implementation

As described in section 2.3, the AWS Management Console has provided users a point-and-click web interface for Amazon Web Services. Although the AWS Management Console can provide an interactive and convenient management of compute, storage, and other cloud resources to general users, it fails to support software developers to create applications which are built on AWS. Alternatively, there are other methods to access Amazon EC2 such as 1) coding directly to the web service API; 2) using a programming library for languages such as Java, PHP, Python, Ruby and Windows .Net; and 3) using the command line tools provided either by Amazon or other developers from AWS developer community. Among these three methods, we have chosen to use the last one to implement our Amazon EC2 Plugin as that is the simplest way to access Amazon EC2 services effectively from within any software programs. Currently, we have provided its implementation in the Linux platform.

3.2.1. Amazon EC2 Accessing Scripts

To implement our Amazon EC2 Plugin for VMD, we have selected to use the Amazon EC2 command line tools developed by Timothy Kay [18] because it is simple to install and simple to use. The only dependent tools used by the command line tools are Perl [22] and Curl [7] and they are normally included in many of the common Linux distributions. Table 1 shows the full list of command line tools provided by Timothy Kay's implementation for Amazon EC2 accessing.

Table 1. A list of supported commands for accessing Amazon EC2

Command Syntax	Description
aws add-group GROUP aws addgrp GROUP	Create a security group named GROUP.
aws add-keypair KEYNAME aws addkey KEYNAME	Create a new key pair named KEYNAME and displays the private key.
aws allocate-address aws allad	Allocate a new public IP address.
aws associate-address ... aws aad ...	Associate an IP address with an instance.
aws authorize ... aws auth ...	Create a new key pair named KEYNAME and displays the private key.
aws delete-keypair KEYNAME aws delkey KEYNAME	Delete the key pair named KEYNAME.
aws delete-group GROUP aws delgrp GROUP	Delete the security group named GROUP.
aws describe-groups [GROUP...] aws dgrp [GROUP...]	Describe security groups.
aws describe-keypairs [KEY...] aws dkey [KEY...]	Describe the key pairs in used.
aws describe-images ... aws dim [IMAGE...] [-o OWNER] [-u USER]	Describe EC2 images.
aws describe-instances aws din [INSTANCE...]	Describe all EC2 server instances.
aws describe-regions aws dreg	Describe available EC2 regions. When issuing EC2 commands specify --region=REGION to indicate a region.
aws reboot-instances INSTANCE [INSTANCE...] aws reboot INSTANCE [INSTANCE...]	Reboot EC2 server instances.
aws run-instances ... aws run [--wait=SEC] [--simple]	Start N EC2 server instances. To poll for instances to leave "pending" or to display the instanceId's in tab-separated format
aws terminate-instances INSTANCE [INSTANCE...] aws tin INSTANCE [INSTANCE...]	Terminate EC2 server instances.

By using this command line tools, we have implemented a set of bash shell scripts, which are programmed to handle HPC cluster creation and termination in Amazon EC2; NAMD job submission and

IMD connection to Amazon EC2; and NAMD results retrieval from Amazon EC2 as described in Section 3.1. Table 2 below shows the full list of bash shell scripts that we have developed.

Table 2. Bash scripts for handling NAMD simulations on Amazon EC2 HPC cluster

Shell Script	Description
awsConnect [-n number]	Connect to Amazon EC2 and start n number of Cluster Compute instants (an EC2 cluster).
awsReady	Check readiness of the EC2 cluster created for accepting instructions.
awsTerminate	Terminate the EC2 cluster created.
awsSubmit	Submit a NAMD simulation to the EC2 cluster.
awsIMD	Enable Interactive Molecular Dynamics within the Plugin.
awsCheck	Check readiness of the simulation result.
awsCollect	Collect simulation result from Amazon to a local host for post-processing.

Finally, a Tcl/Tk package was developed that implements the GUI-based plugin for VMD as shown in Figure 5 as well as the programming logic to invoke any one of the bash shell scripts as listed in Table 2 in order to perform the Amazon EC2 handling tasks as specified in Section 3.1. The code fragments listed in the appendix: App1, App2 and App3 show how the plugin was implemented as tcl/tk programs. In App1, the code fragment shows that major variables were declared for the tcl/tk package. In App2, the code fragment shows how the tk widgets as shown in the plugin GUI (see Figure 5) were created. In App3, the code fragment shows the procedures where each of them was associated with a widget element, e.g. a command button, in the plugin GUI. For example, the highlighted text in App2 shows a button widget named Start which has associated with a command procedure of IMD::connect. When the Start button is clicked, IMD::connect would be executed which in turn will invoke the awsConnect script (see Table 2) in order to start an EC2 cluster.

3.2.2. Providing public AMI that supports NAMD simulation

When a HPC cluster is to be created and started on Amazon EC2, users of our VMD Amazon Plugin must provide two parameters: 1) an Amazon Machine Image (AMI) for creating an instance and 2) a total number of instances for the HPC cluster. The former contains all information necessary to boot instances of software such as operating system, middleware and applications. The latter quantifies the size of the HPC cluster, which in turn defines the maximum computing power to be provided.

There is a large selection of public AMIs available from Amazon and the Amazon EC2 community. However in many cases, users may need to customize a public AMI and then save that customized AMI or even create new AMI from scratch for their own use. To facilitate VMD/NAMD users with little computing and HPC knowledge, an AMI has been provided, which was constructed specifically for running NAMD simulations. The AMI runs Ubuntu Linux and NAMD version 2.7. This AMI has been set publicly accessible from Amazon EC2. Users of our VMD Amazon Plugin can access this AMI by referencing its identifier, the AMI ID. This way, the construction of a HPC cluster and the installation and configuration of applications (NAMD in this case) has been abstracted into selecting an AMI from Amazon to use.

In the current version of our plugin, this AMI is pre-coded in the bash shell script, awsConnect, of the Tcl/Tk package. We are planning to supply more AMIs in the future, which will define different machine images to support various NAMD versions including those with GPU acceleration.

3.2.3. Embedding the Amazon EC2 Plugin into VMD

To embed a plugin into VMD, a plugin directory is needed to store the Tcl/Tk package in a host computer, which runs VMD. Then, the plugin directory and the plugin name must be known by VMD through their settings in the VMD start-up file .vmdrc. The variable auto_path is used to set a searchable path for the locations of plugin directories and variable vmd_install_extension allows third party plugins be recognized by VMD as shown in App4.

Finally, credentials of Amazon Web Services of a user account must be made accessible to the VMD Amazon Plugin in order to authenticate one's requests to any AWS service. Two types of access credentials must be provided in this case. The first one is the Access Keys pair, which is used to make secure REST or Query protocol requests to any AWS service API. The second one is the Amazon EC2 Key Pair, which is used to launch and then securely access Amazon EC2 instances. Details in setting up these credentials of AWS can be found in AWS webpage.

4. A CASE STUDY OF MOLECULAR DYNAMICS SIMULATIONS

Section 3 has presented the design and implementation of our Amazon EC2 Plugin for VMD. In this section, a case study of molecular dynamics simulations is used to demonstrate how accessibility and user-friendliness is improved by using the plugin. This case study also provides a performance evaluation of the

molecular dynamics simulations in the HPC cluster created on Amazon EC2 and an account on the total cost involved to carry out such simulations.

4.1. Satellite Tobacco Mosaic Viruse

Satellite Tobacco Mosaic Virus (STMV) [17] is a small icosahedral plant virus, which lives on both a host cell and a host virus, the tobacco mosaic virus in this case, for it to reproduce. The entire STMV particle consists of 60 identical copies of a single protein that make up the viral capsid (coating), and a 1063 nucleotide single stranded RNA genome, which codes for the capsid and one other protein of unknown function. Although the full crystal structure of the STMV with both its capsid and RNA has been solved by Alexander McPherson [13] using X-ray crystallography and Atomic Force Microscopy (AFM) techniques and that has revealed a great deal of information about STMV, some unresolved questions remain regarding this virus. One of such is how does the crystal structure of STMA assemble and disassemble inside a host cell, because it has been impossible to obtain a high enough resolution electron map for that part of the structure. Therefore, STMV makes a good candidate for study using molecular dynamics simulations that can provide both a visual representation and quantitative data regarding the thermal motion of a system's structure over time, or its response to applied forces [8].

STMV was chosen for our NAMD simulations experiments because the fully solvated STMV system consists of approximately 1 million atoms, which is on the high end of what is feasible to simulate using molecular dynamics. This can demonstrate well for the needs of researchers to run their simulations on HPC hardware. The purpose of our experiments carried out however is not meant to provide any study on the virus itself.

4.2. Experiments

The experimental testbed used in this case study of molecular dynamics simulations is a computer cluster making up of 8 Cluster Compute Instances created in Amazon EC2. The computer cluster offers high computing power and low network latency features for optimal performance with HPC applications. Each Cluster Compute Instance has 23 GB of memory, 33.5 EC2 Compute Units[2] (2 x Intel Xeon X5570, quad-core "Nehalem" architecture), 1690GB of instance storage and a 10 Gigabit Ethernet network interface. Thus, this is a HPC cluster with 64 CPUs. Besides, a standalone workstation computer (Intel Xeon dual-core, 1.6GHz) with 4GB of memory is used to connect remotely to the cluster in Amazon so that a terminal can be created in the cluster to control the experiment.

Two sets of the experiment were performed. In the first experiment set, we calibrated our Amazon HPC cluster for its speedup performance by running NAMD simulations of STMV. The simulations source of STMV was obtained from [19]. It contains the structure file (stmv.psf), coordinates file (stmv.pdb), parameters file (par_all27_prot_na.inp) and configuration file (stmv.conf). Two workloads of NAMD simulations of STMV were made up by adopting two different simulation periods of 0.5 picosecond and 2 picoseonds where all other simulation parameters were unchanged as stated in the original source. Each of these two simulation workloads were run on the HPC cluster using various number of CPUs in the exponent of 2 increasing from 1 to 64 (i.e. 1, 2, 4, 8, 16, 32 and 64). In the second experiment set, a workload of NAMD simulations of STMV with a simulation period of 30 picoseonds were separately run on a standalone workstation computer and the HPC cluster with 64 CPUs. The job execution time was measured for the simulations in both cases.

4.3. Using the Amazon EC2 Plugin

With the present of this Amazon EC2 Plugin for VMD, users are free from performing many tedious computing tasks such as launching, connecting and terminating Amazon EC2 compute instances; configuring a HPC cluster; and installing the NAMD software application before the system is readily available for running a NAMD simulation in the Amazon EC2. The simulation result can be transferred back to the user easily as well. All these tasks have been encapsulated into a few button clicks as illustrated in Figure 6, Figure 7, Figure 8 and Figure 9 rather than requiring users to perform those tasks manually as described in Section 2.3.

2The Elastic Compute Unit (ECU) was introduced by Amazon EC2 as an abstraction of compute resources. One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz Opteron or Xeon processor.

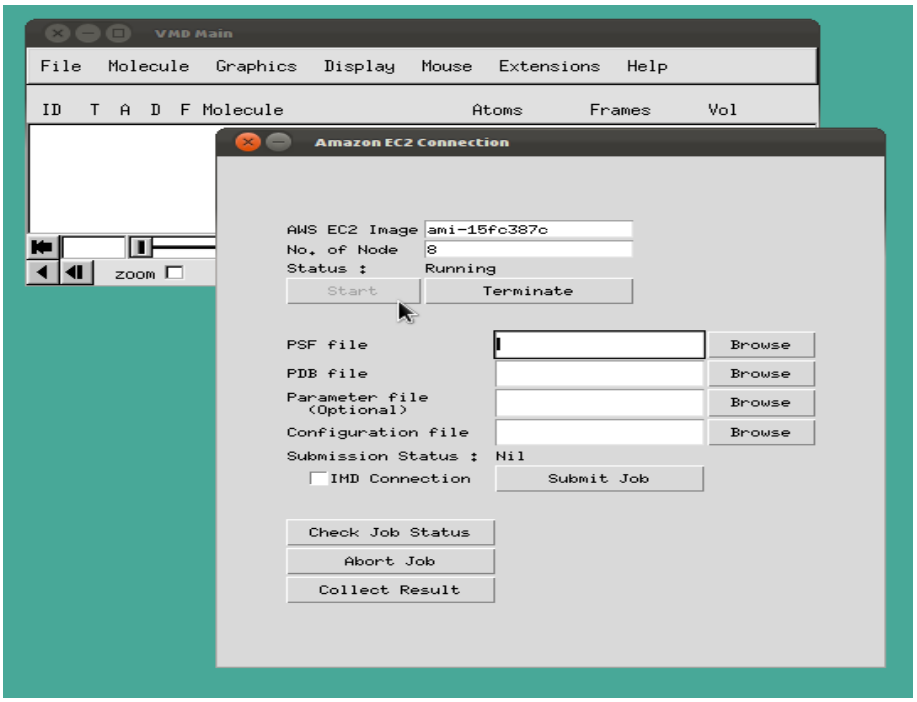

Figure 6. A snapshot of the VMD Amazon Plugin showing 8 compute nodes have been started

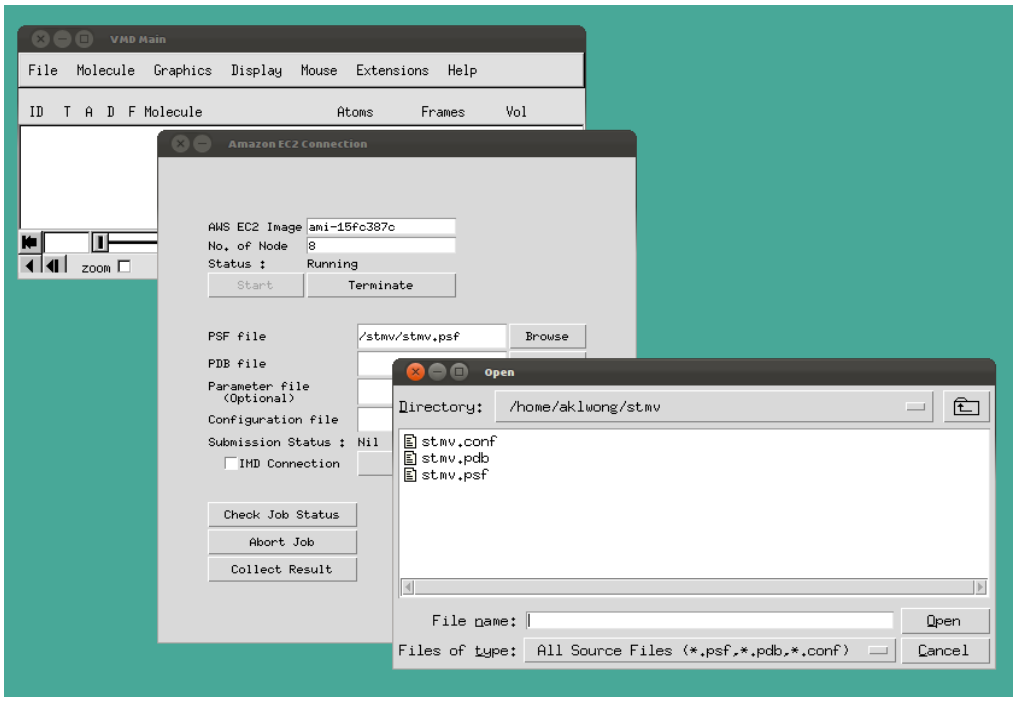

Figure 7. A snapshot of the VMD Amazon Plugin showing input files for a NAMD simulation are being submitted

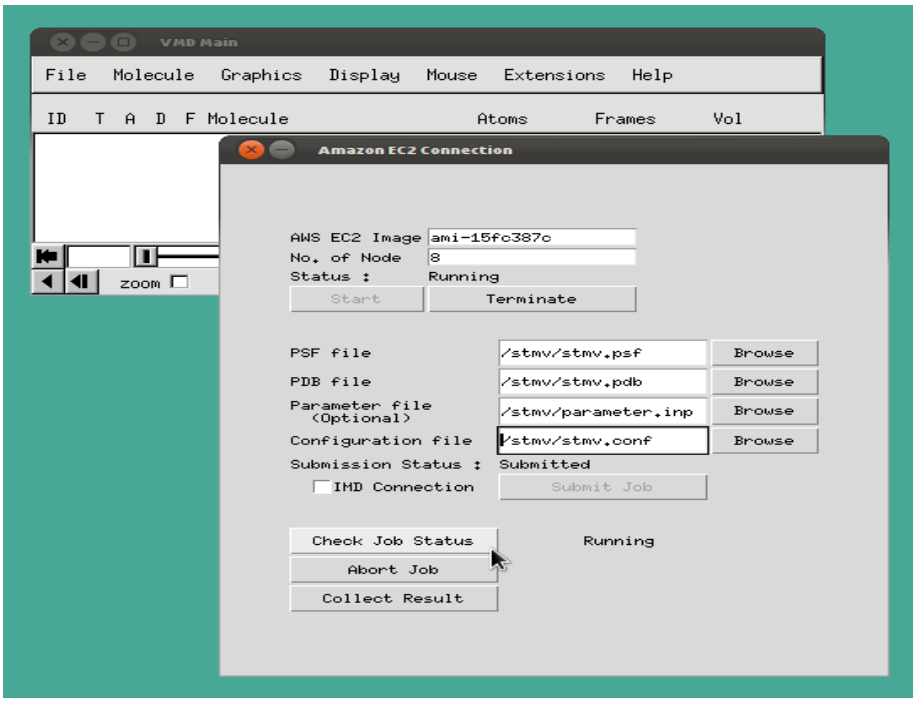

Figure 8. A snapshot of the VMD Amazon Plugin showing a NAMD simulation has been submitted for execution

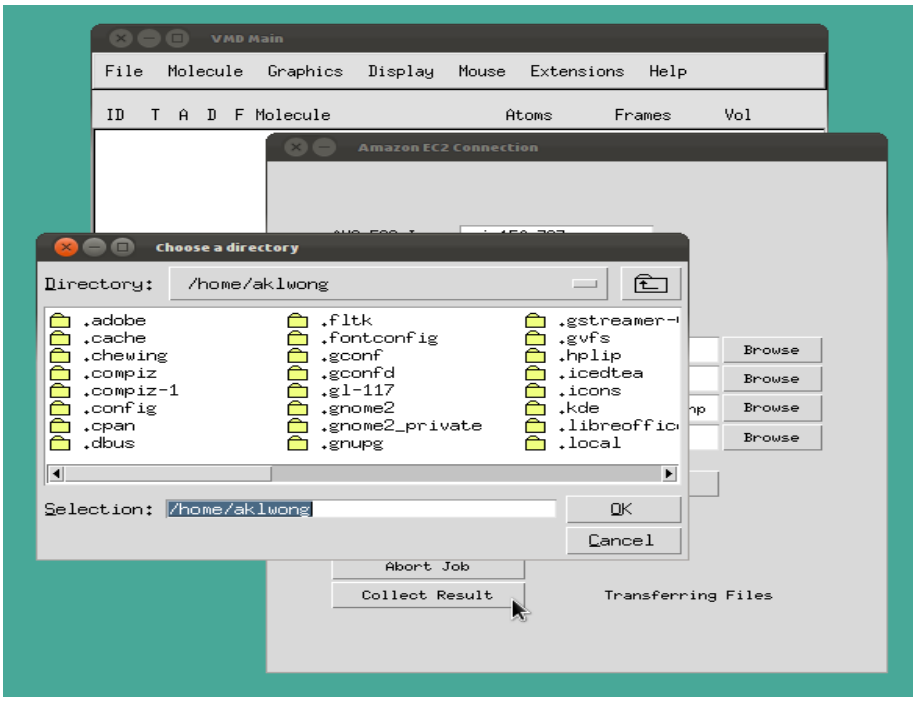

Figure 9. A snapshot of the VMD Amazon Plugin showing a NAMD simulation has been finished and the result files are being collected

4.4. Results

Figure 10 and Figure 11 have captured the outcome of the first experiment set, that is, a calibration of our 8 instances (64 CPU cores) Amazon EC2 HPC cluster for running NAMD simulations of STMV. Figure 10 shows that the execution time of STMV simulations decrease as the number of CPU cores used increase; and the execution time of STMV simulations increase as the simulation periods increase. Figure 11 is the speedup representation of Figure 10. The speedup is defined here as follow:

$$Speedup = \frac{Sequential\ Execution\ Time\ \ (on\ single\ CPU)}{Parallel\ Exection\ Time\ (on\ multiple\ CPUs)}$$

This calibration has shown that speedup of STMV simulations is achievable by using the Amazon EC2 HPC cluster. The result has confirmed that the Cluster Compute Instance type offered by Amazon EC2, which features high computing power and low network latency, can deliver good performance for communication-bounded HPC applications.

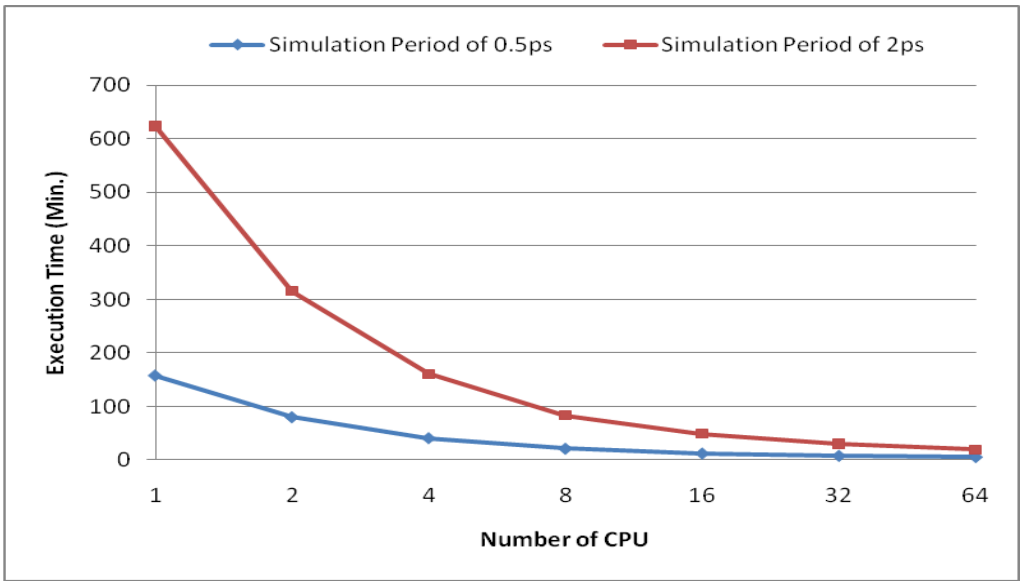

Figure 10. STMV simulations on a 64-CPUs Amazon EC2 HPC Cluster

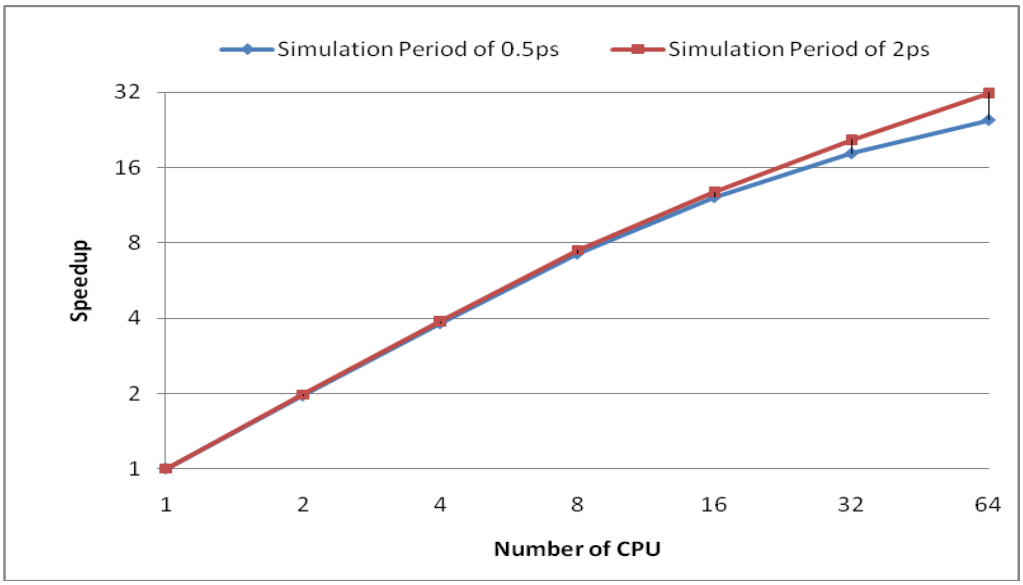

Figure 11. Speedup of STMV simulations on a 64-CPUs Amazon EC2 HPC Cluster

Figure 12 presents the result obtained from the second experiment set. The workload of NAMD simulations of STMV was 30 picoseonds. The execution time of the NAMD simulations obtained from running on a standalone workstation and the Amazon EC2 HPC cluster (all 64 CPUs) are 129.6 hours (5.4 days) and 4.6 hours correspondingly. The first scenario of using a standalone workstation for the NAMD simulations represented the case that no upfront investment in HPC hardware was affordable by the user. On the other hand, the second scenario of employing Amazon EC2 for the same NAMD simulations represented

a contrasting case that a very flexible pay-as-you-go method of buying HPC hardware could be afforded by the user. According to the latest pricing information that can be found in the Amazon EC2 website, the standard cost of a Cluster Compute Instance is $1.60 per hour[3]. The 30 picoseonds NAMD simulation has run on 8 Cluster Compute Instances (64-CPUs) for 4.6 hours and thus has incurred a total cost of $64.00($1.60 $times$ 8 $times$ 5).

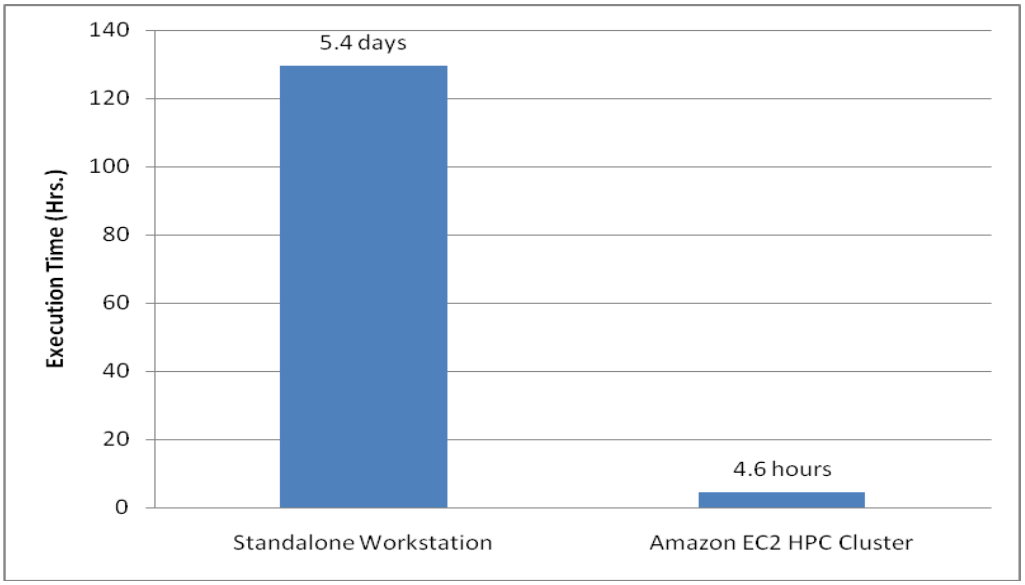

Figure 12. STMV simulations (30ps) on i) a workstation and ii) an Amazon EC2 HPC Cluster

5. CONCLUSION

We have designed and implemented a software plugin of VMD for NAMD simulations on the Amazon EC2. The plugin has provided an integrated platform for the molecular dynamics users of VMD/NAMD to harness a flexible pay-as-you-go method of purchasing HPC resource from the Amazon EC2. More importantly, it has relieved the users from performing many tedious computing tasks such as setting up and configuring a HPC cluster; and installing middleware and software applications before the system is readily available for any actual scientific investigation. As presented in Section 3.1, the graphical user interface of the plugin has abstracted many low level details in HPC cluster creation, NAMD job submission as well as result retrieval into just a few button chicks in the plugin. The idea was further testified with a NAMD simulations case study as presented in Section 4 where the plugin allows VMD/NAMD users to spend less time getting applications to work on HPC clusters but more time on researching.

The NAMD simulations case study has also served to demonstrate a second objective of our work. That is the use of cloud computing for running HPC applications. Recently, some public cloud vendors including the Amazon EC2 have provided computer instances which are specifically designed for HPC applications and other demanding network-bound applications. Our performance result obtained from running NAMD simulations on Amazon EC2 Cluster Compute Instances is in-line with other studies [10] [16], which has confirmed that a communication-bounded parallel application, like NAMD, can also be benefited by using Amazon EC2 HPC cluster. The flexible cost advantage of Amazon EC2 gives users the opportunity to test and run their high performance computing applications in the cloud at a price and performance level that can leverage their budgets and computation requirements.

Currently, we are working toward a generic solution for developing software plugins and add-ons of cloud services. We have proposed the High-Performance-Computing-as-a-Service (HPCaaS) model [27] to abstract high performance computing resources including both the hardware: networks, storages and servers (physical/virtual) and the software (operating systems, middleware and user level HPC applications) so that these HPC resources can be exposed to the cloud community as software libraries of 1) a set of virtual machine images with pre-built HPC applications and 2) a set of program scripts which can be used to create, manage and terminate a computer cluster, to access HPC applications, and to transfer data between users and

3 Pricing is per instance-hour consumed for each instance, from the time an instance is launched until it is terminated. Each partial instance-hour consumed will be billed as a full hour.

the computer cluster.

ACKNOWLEDGEMENTS

This work was supported by the Amazon Web Services (AWS) in Education Research Grant.

REFERENCES

[1] B. J. Alder and T. E. Wainwright, "Studies in Molecular Dynamics. I. General Method", J. Chem. Phys. 31 (2): 459-466, 1959.
[2] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2. Last access: August 2011.
[3] Amazon Web Service (AWS) Management Console. http://aws.amazon.com/console. Last access: August 2011.
[4] M. Bhandarkar et al. "NAMD User Guide Version 2.8", Theoretical Biophysics Group University of Illinois and Beckman Institute. 2011.
[5] Charm++. http://charm.cs.uiuc.edu. Last access: August 2011.
[6] Computer-Aided Drug Design Center, School of Pharmacy, University of Maryland. http://mackerell.umaryland.edu/CADD/CADD_center.html. Last access: August 2011.
[7] Curl. http://curl.haxx.se. Last access: August 2011.
[8] P. L. Freddolino, A. S. Arkhipov, S. B. Larson, A. McPherson, and K. Schulten. "Molecular dynamics simulations of the complete satellite tobacco mosaic virus", Structure, 14:437-449, 2006.
[9] D. Frenkel and B. Smit, "Understanding Molecular Simulation: from algorithms to applications", San Diego, California: Academic Press, 2001.
[10] A. Goscinski, M. Brock and P. Church. "High Performance Computing Clouds. Cloud computing: methodology, system, and applications", CRC, Taylor & Francis group. 2011.
[11] Interactive Molecular Dynamics Simulation. http://www.ks.uiuc.edu/Research/vmd/imd. Last access: August 2011.
[12] J. McCammon, J. B. Gelin, M. Karplus, "Dynamics of folded proteins", Nature 267 (5612): 585–590, 1977.
[13] A. McPherson. Department of Biological Sciences, Molecular Biology and Biochemistry, University of California, Irvine. http://darwin.bio.uci.edu/~faculty/mcpherson. Last access: August 2011.
[14] NAMD Molecular Dynamics Simulator. http://www.ks.uiuc.edu/Research/namd. Last access: August 2011.
[15] Protein Data Bank. http://www.pdb.org. Last access: August 2011.
[16] J. K.R. Ramakrishnan, L. M. K. Canon, S. Cholia, S. Shalf, H.J. Wasserman and N.J. Wright. "Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud", In Proceedings of IEEE Second International Conference in Cloud Computing Technology and Science (CloudCom), pp. 159-68, 2010.
[17] Satellite Tobacco Mosaic Virus. http://www.ks.uiuc.edu/Research/STMV. Last access: August 2011.
[18] Simple Command-Line Access to Amazon EC2. http://aws.amazon.com/developertools/739. Last access: August 2011.
[19] STMV simulation source. http://www.ks.uiuc.edu/Research/namd/utilities. Last access: August 2011.
[20] Theoretical and Computational Biophysics Group at the Beckman Institute of the University of Illinois at Urbana-Champaign. http://www.ks.uiuc.edu/Overview/People. Last access: August 2011.
[21] The MPI Forum. http://www.mpi-forum.org. Last access: August 2011.
[22] The Perl Programming Language. http://www.perl.org. Last access: August 2011.
[23] VMD Molecular Graphics Viewer. http://www.ks.uiuc.edu/Research/vmd. Last access: August 2011.
[24] VMD Plugin Library. http://www.ks.uiuc.edu/Research/vmd/plugins. Last access: August 2011.
[25] B. Welch, K. Jones and J. Hobbs. "Practical Programming in Tcl and Tk", Prentice Hall, Upper Saddle River, NJ, USA, 2003.
[26] A. K. L. Wong and A. Goscinski, A VMD Plugin for NAMD Simuations on Amazon EC2, In Preceedings of the International Conference on Computational Science, ICCS 2012, Omaha, Nebraska, USA, June, pp. 135-145.
[27] P. Church, A.K.L. Wong, A. Goscinski, and M. Brock. Toward Exposing and Accessing HPC Applications in a SaaS Cloud, In Proceedings of the 19th IEEE International Conference on Web Services (ICWS 2012), June 24-29 2012, Honolulu, Hawaii, USA, pp. 692-699.

6. APPENDIX

```
# Tk-based Amazon EC2 Plugin window in VMD
# $Id: AmazonEC2Plugin.tcl
package provide AmazonEC2Plugin 1.0

namespace eval IMD {
  # Define variables
  variable imdmol        -1   ;# molecule ID of the live IMD simulation
  variable hostname
  variable awsImageID     "ami-15fc387c"
  variable numberOfInstance
  variable transferrate    0
  variable keeprate        0
  variable w
  variable defaultText        ""
  variable status2            "None"
  variable submitStatus       "Nil"
  variable psf
  variable pdb
  variable config
}
```

App1. Grobal variables declaration

```
# Tk-based Amazon EC2 Plugin window in VMD
# $Id: AmazonEC2Plugin.tcl
package provide AmazonEC2Plugin 1.0
   :
# the GUI
proc IMD::startGUI { } {
  variable w
  global env

  # make the initial window
  set w [toplevel ".imd"]
  wm title $w "Amazon EC2 Connection"
  wm resizable $w no no
  set textfont [font create -family helvetica -size 8]
  set labelfont [font create -family helvetica -size 8]
  set buttonfont [font create -family helvetica -weight bold -size 7]

  # 1st set of widgets
  frame $w.win ;# Main contents
  pack $w.win -padx 40 -pady 40
  frame $w.win.server
  label $w.win.server.hostlabel -text "AWS EC2 Image" -font $labelfont -anchor w
  label $w.win.server.portlabel -text "No. of Node" -font $labelfont -anchor w
  entry $w.win.server.awsImageID -width 20 -relief sunken -bd 1 -textvariable IMD::awsImageID -font $textfont
  entry $w.win.server.numberOfInstance -width 10 -relief sunken -bd 1 -textvariable IMD::numberOfInstance -font $textfont
  button $w.win.server.connect -text "Start" -font $buttonfont -command IMD::connect
  button $w.win.server.terminate -text "Terminate" -font $buttonfont -command IMD::terminate -state disabled
  label $w.win.server.status1 -text "Status : " -font $labelfont -anchor w
  label $w.win.server.status2 -text "Not Connected " -textvariable IMD::status2 -font $labelfont -anchor w
   :
  # 2nd set of widgets
  frame $w.win.settings
  label $w.win.settings.psflabel -text "PSF file" -font $labelfont -anchor w
  label $w.win.settings.pdblabel -text "PDB file" -font $labelfont -anchor w
  label $w.win.settings.configlabel -text "Configuration file" -font $labelfont -anchor w
  entry $w.win.settings.psf -width 20 -relief sunken -bd 1 -textvariable IMD::psf -font $textfont
  entry $w.win.settings.pdb -width 20 -relief sunken -bd 1 -textvariable IMD::pdb -font $textfont
  entry $w.win.settings.config -width 20 -relief sunken -bd 1 -textvariable IMD::config -font $textfont
  button $w.win.settings.browse1 -text "Browse" -font $buttonfont -command "doIt $w.win.settings.psf"
  button $w.win.settings.browse2 -text "Browse" -font $buttonfont -command "doIt $w.win.settings.pdb"
  button $w.win.settings.browse3 -text "Browse" -font $buttonfont -command "doIt $w.win.settings.config"
  label $w.win.settings.status1 -text "Submission Status : " -font $labelfont -anchor w
  label $w.win.settings.status2 -text "Nil " -textvariable IMD::submitStatus -font $labelfont -anchor w
  button $w.win.settings.submit -text "Submit Job" -font $buttonfont -command IMD::submit
   :
  # 3rd set of widgets
  frame $w.win.controls
  button $w.win.controls.check -text "Check Job Status" -font $buttonfont -command IMD::stop
  button $w.win.controls.abort -text "Abort Job" -font $buttonfont -command IMD::connect
  button $w.win.controls.collect -text "Collect Result" -font $buttonfont -command IMD::collect
  label $w.win.controls.status1 -text "" –textvariable IMD::defaultText -font $labelfont -anchor w
   :
  # pack the frames
  set padx 0
  set pady 10
  pack $w.win.server -fill x -expand yes -anchor nw -pady $pady
  pack $w.win.settings -fill x -expand yes -padx $padx -pady $pady
  pack $w.win.controls -fill x -expand yes -padx $padx -pady $pady
}
```

App2. Widgets declaration

```
# Tk-based Amazon EC2 Plugin window in VMD
# $Id: AmazonEC2Plugin.tcl
package provide AmazonEc2Plugin 1.0
  :
proc IMD::connect { } {
  variable hostname
  variable awsImageID
  variable numberOfInstance
  variable w
  set IMD::status2 "Running"
  puts stdout "awsImageID = $IMD::awsImageID\n"
  puts stdout "number of instance = $IMD::numberOfInstance\n"
  set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsConnect numberOfInstance} msg]
  puts "rc = $rc\n"
  if { $rc } {
                puts stderr "Error: Could not connect to AWS EC2!\n$msg"
                exit 1
  }
  $IMD::hostname = $rc
  $w.win.server.terminate configure -state normal
  $w.win.server.connect configure -state disable
}

proc IMD::terminate { } {
  variable w
  set IMD::status2 "Terminated"
  set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsTerminate $IMD::hostname} msg]
  puts "rc = $rc\n"
  if { $rc } {
                puts stderr "Error: Could not terminate AWS EC2!\n$msg"
                exit 1
  }
  $w.win.server.connect configure -state normal
  $w.win.server.terminate configure -state disable
}

proc IMD::submit { } {

  variable w
  puts "psf = $IMD::psf"
  puts "pdb = $IMD::pdb"
  puts "config = $IMD::config"
  set myList [split $IMD::config /]
  set listSize [llength $myList]
  set doSomething "$listSize - 1"
  set lastIndex [expr $doSomething]
  set configFileName [lindex $myList $lastIndex]
  puts $configFileName
  set IMD::submitStatus "Submitted"
  set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsSubmit $IMD::psf $IMD::pdb $IMD::config $configFileName $IMD::hostname &} msg]
  $w.win.settings.submit configure -state disable
}

proc IMD::collect { } {
  variable w
  set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsCollect $IMD::hostname} msg]
  $w.win.settings.submit configure -state normal
  set IMD::submitStatus "Nil"
}

proc IMD::check { } {
  variable w
  set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsCheck $IMD::hostname} msg]
  if { $rc } {
                puts stderr "Error: Could not terminate AWS EC2!\n$msg"
                exit 1
  }
  $IMD::defaultText = $rc
}

proc IMD::abort { } {
  variable submitStatus
  variable Status2
  variable w
  if {{ $submitStatus == "Submitted"} && { $Status2 == "running"} } {
    set rc [catch {exec $SYSTEM/vmdplugins/Amazon/awsTerminate $IMD::hostname} msg ]
    $IMD::defaultText = $rc
  } else {
    set msg "Abort failed."
    tk_messageBox -title "IMD Error" -parent $w -type ok -message $msg
  }
}
```

App3. Command procedures for widget elements

```
# .vmdrc
# Bring main menu back
menu main on

# Prepend the path to third party plugins
set suto_path [linsert $auto_path 0 [file job $env(Home) vmdplugins]]

# Add third party plugins
vmd_install_extension amazonNamd amazonNamd_tk_cb "Amazon EC2 Connection"
```

App4. A sample VMD start-up file

BIOGRAPHY OF AUTHORS

Adam K. L. Wong received his PhD. in Computer Science from the University of Queensland in 2004. He is currently working as a research fellow in the School of Information Technology at Deakin University conducting research in Cloud Computing, High Performance Computing and Bioinformatics. From 2009 to 2010, he worked as a bioinformatican at the Victorian Partnership for Advanced Computing (VPAC) in Australia and his major role was in the development of a HPC bioinformatics server for the bioinformatics research group (BioDeakin) at Deakin University. Adam has participated in many professional activities. He has served as a reviewer of paper for many international workshops/conferences and journals in the research areas of parallel and distributed computing, Clusters, and Grid. They include the Future Generation Computer Systems, Parallel Computing, IEEE/ACM Supercomputing, IEEE Cluster Computing and the Grid, and IEEE High Performance Distributed Computing. He is also a committee member of the International Conference on Applied Computing (IADIS).

Professor A. Goscinski has had a long-standing interest in distributed systems, parallel processing, virtualization, autonomic and service computing, and clouds and cloud computing. Since 2004, he has successfully concentrated his research on autonomic grids based on SOA, the abstraction of software and resources as a service, and cloud computing. The major achievement in the area of autonomic grids based on SOA is the development of the concept of a broker that led to its use in clouds. The major achievement in the area of the abstraction of software and resources as a service and cloud computing is the development of the Resource Via Web Services (RVWS) framework that contains service's dynamic state and characteristics, and service publishing, selection and discovery; the contribution to level of cloud abstraction in the form of CaaS (Cluster as a Service); and comparative study of High Performance Computing clouds. The results of this research have been published in the high quality journals and conference proceedings.