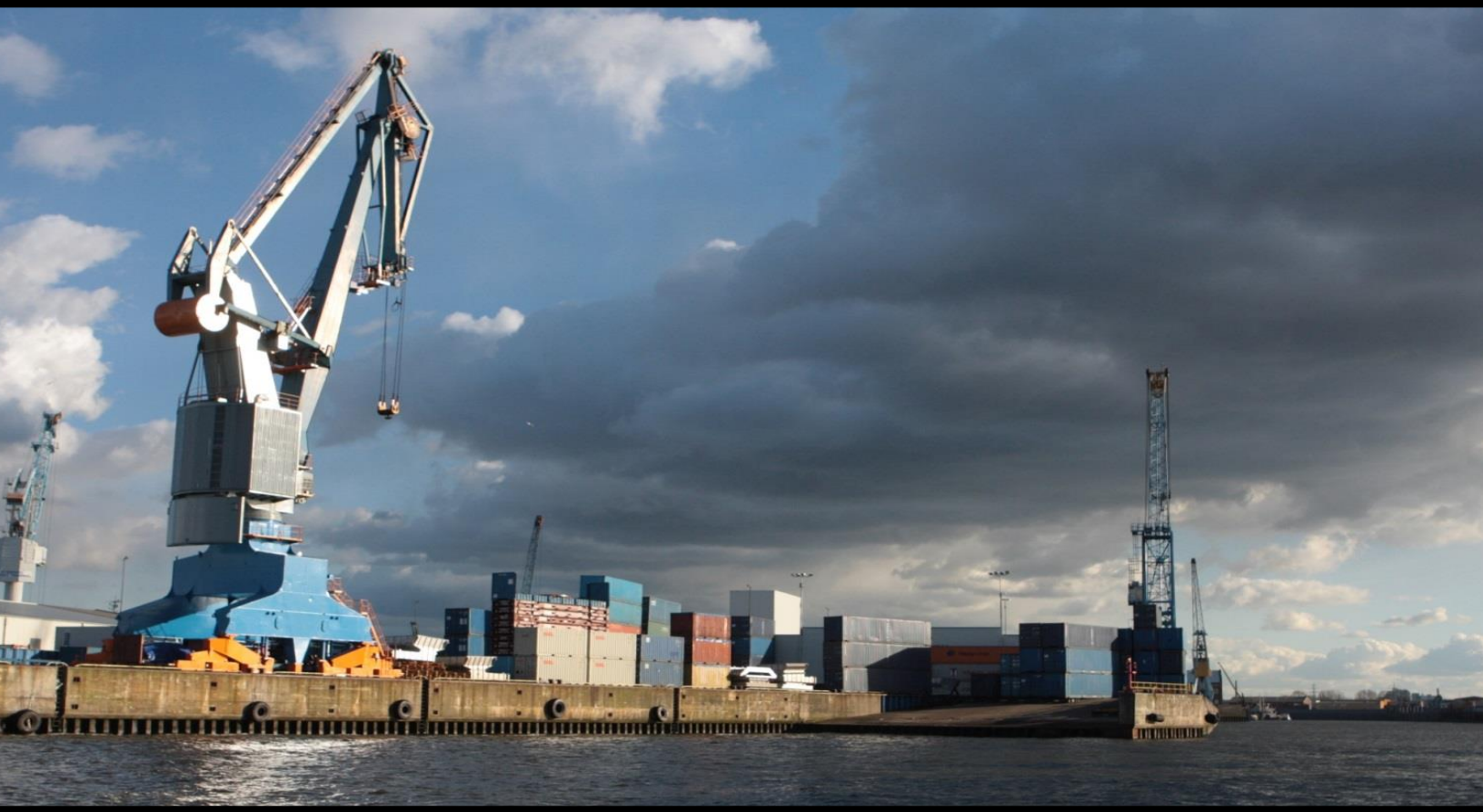
The Simr* Projects

Selected Case Studies on Amazon AWS

*) Simr, formerly known as UberCloud



Simr Compendium of
11 Engineering Simulation Case Studies on AWS

https://www.Simr.com

Engineering Case Studies on Amazon AWS

Starting in July 2012, and 230 HPC cloud experiments later, with 150+ case studies, and a ton of hands-on experience gained, that's the harvest of 10 years of UberCloud Experiments (now Simr Experiments) for engineering simulations. We are now able to measure cloud computing progress, quite objectively. Looking back these years at our first 50 cloud experiments in 2012, 26 of them failed or didn't finish, and the average duration of the successful ones was about three months. Already five years later, looking at the next 50 cloud experiments, none failed; and the average duration of these engineering cloud experiments is now just a few weeks. This includes defining the application case, preparing and accessing the engineering application software in the cloud, running the simulation jobs, evaluating the data via remote visualization, transferring the final results back on premise and writing a case study.

The goal of the UberCloud CAE Experiments is to perform engineering simulations in the Cloud with real engineering applications, in order to understand the roadblocks to success and how to overcome them. Our so far 23 Compendiums of Case Studies are a way of sharing these results with our broader community of engineers, their managers, and their service providers.

We found that enterprises would strongly benefit from technical computing in the cloud. By gaining access on-demand from their desktop workstation browser to additional remote and powerful computing resources, their major benefits now are: the agility gained by immediate access to any computing, storage, and networking resources: by shortening product design cycles through shorter simulation run times; the superior quality achieved by simulating more and more sophisticated applications; and by running many more iterations to look for the best product design. These are benefits that can increase a company's ability to innovate and compete, dramatically.

But how far away are we today from an ideal cloud model for engineers and scientists? In the beginning, we did not know. Engineers were just facing challenges like security, privacy, and trust; lack of cloud control; conservative software licensing models; slow data transfer; uncertain cost & ROI; identifying best suited resources; and lack of standardization, transparency, and cloud expertise. However, in the course of the UberCloud Experiments, as we followed each of the (so far) 230 teams closely and monitored their challenges and progress, we have got an excellent insight into these roadblocks, and how our engineering and scientific teams have tackled them, and how we are now able to reduce or even fully resolve them.

Here's now Compendium #23: a selection of 11 engineering simulation projects performed on Amazon AWS, originated from the need to solve complex problems in manufacturing, faster, more efficient, and with higher quality. Our customers simulated a wide variety of applications, such as multi-resonant antenna system, turbo-machinery, spatial hearing, next-generation sequencing, weather research and forecasting, quantitative finance historical data modeling, clinical cancer genomics, CFD compressor map generation, pulsatile flow in a coronary artery, peptide benchmark using molecular dynamics, and accelerating product development at Fortune 50 innovator on Amazon AWS.

Wolfgang Gentzsch and Burak Yenier
Simr, Los Altos, CA, May 2024

Table of Contents

Team 223

Simr Accelerates Product Development at Fortune 50 Innovator on Amazon AWS

PROBLEM: Highly innovative R&D company is building the next generation of products to delight their customers…

- **Product Quality** and **Time to Market** are a competitive differentiator
- Need a platform that design engineers can use with high performance and **No New Learning**
  - An enterprise grade platform supporting complex multi-physics solutions, custom solvers and concurrent access to multiple ISV products
  - Deployed in customer tenant – Simr does not provide compute!
- Can't use generic ISV cloud solutions due to **Security Considerations**
  - Must work with customers existing GRC model and IT processes including directory services and access control
- Needs to be **Low-Overhead** and is not a burden to IT
  - Must provide IT with insights as to use, costs and change management associated with CAE environment

⇒ *They need a secure **"Simulation Capability"** with easy access and use, under their exclusivecontrol, to scale IP generation across the company*

SIMЯ

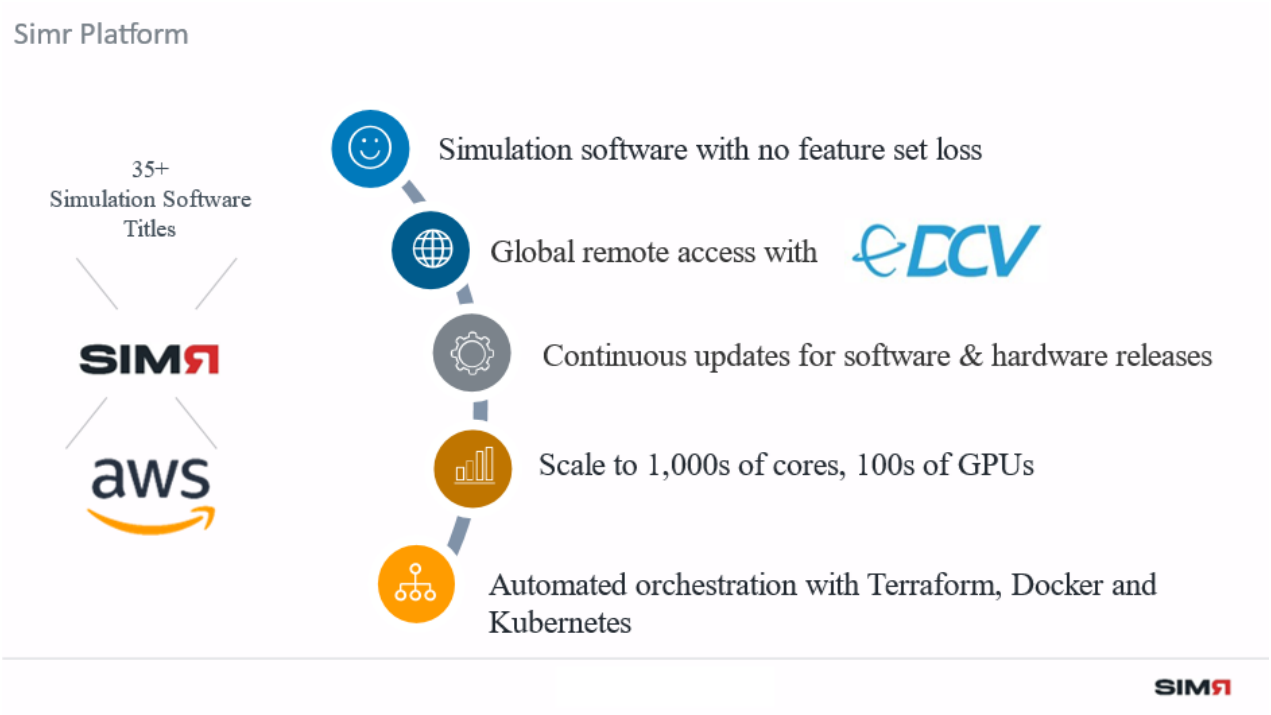
SOLUTION: Simr built a value-added HPC simulation platform on AWS

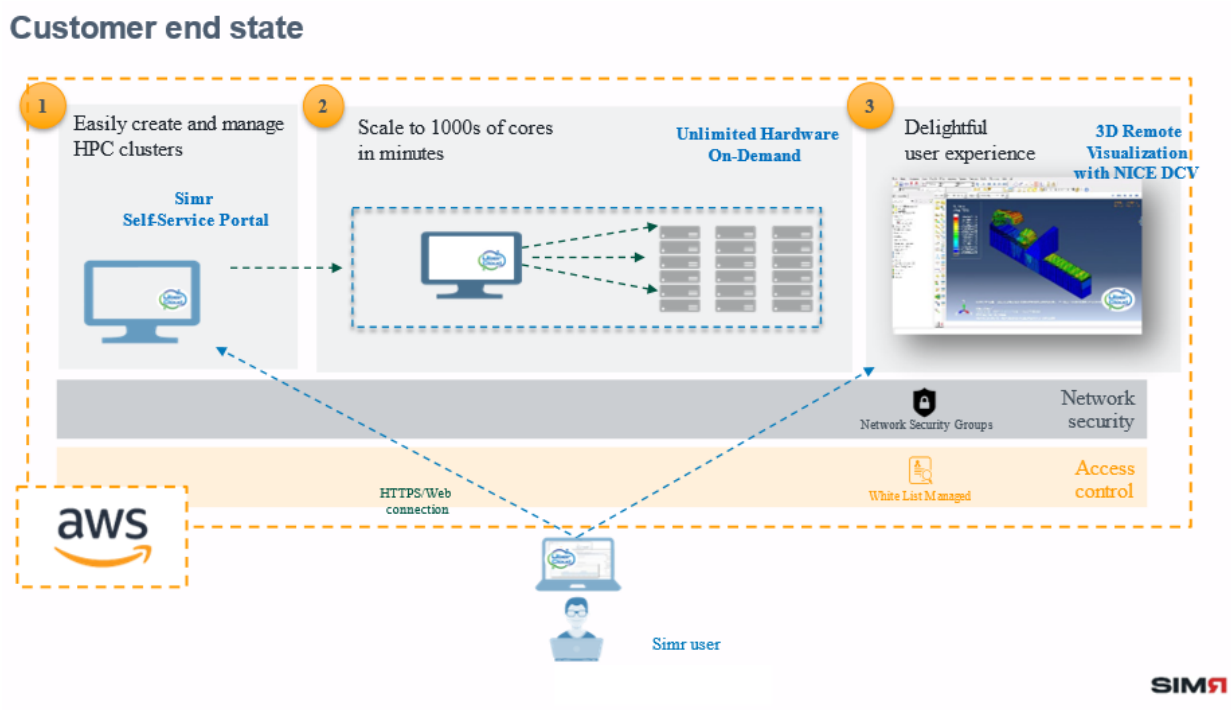
- ✓ Enables simulation-driven R&D
- ✓ Low administrative overhead even at scale
- ✓ Operates within the customer's cloud security policies

Used by 20+ product teams for all engineering simulations

- ✓ Computational Fluid Dynamics
- ✓ Multiphysics
- ✓ Audio Engineering
- ✓ Electronics Design
- ✓ Thermal Analysis
- ✓ Optical Simulations
- ✓ Structural Analysis
- ✓ Fracture Mechanics
- ✓ Materials Simulations

SIMЯ


Simr Platform
35+
Simulation Software
Titles
SIMR
aws
Simulation software with no feature set loss
Global remote access with
DCV
Continuous updates for software & hardware releases
Scale to 1,000s of cores, 100s of GPUs
Automated orchestration with Terraform, Docker and
Kubernetes
SIMR


Customer end state
1
Easily create and manage
HPC clusters
Simr
Self-Service Portal
2
Scale to 1000s of cores
in minutes
Unlimited Hardware
On-Demand
3
Delightful
user experience
3D Remote
Visualization
with NICE DCV
Network Security Groups
Network
security
White List Managed
Access
control
HTTPS/Web
connection
aws
Simr user
SIMR

Simr value-add to different stakeholders

*Simr-enabled AWS consumption - **$13 Million** in 2022*

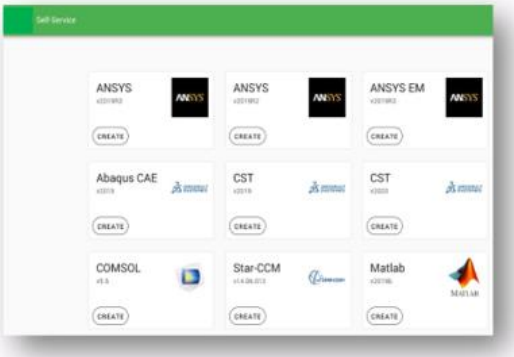
SIMR

Simr value add – software enablement

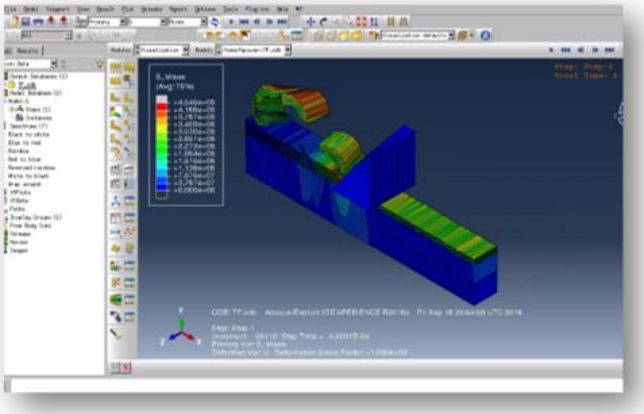
. . . and more.

SIMR

Simr value add – easy for design engineers to use

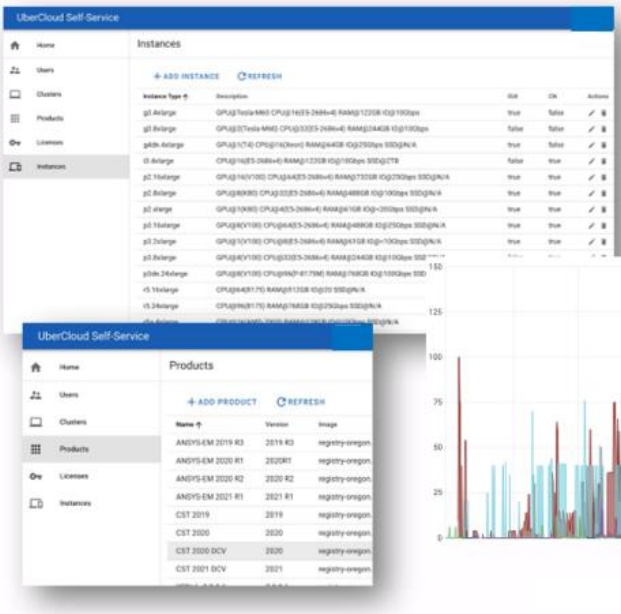


- User friendly launch process for HPC clusters
- High end graphics and monitoring features

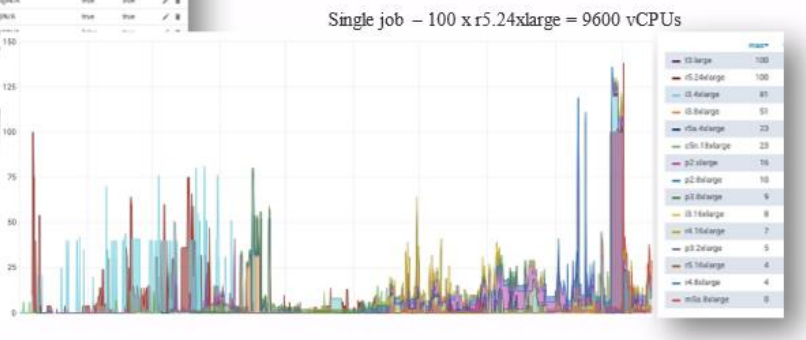


SIMR

Simr value add – easy for IT to administer



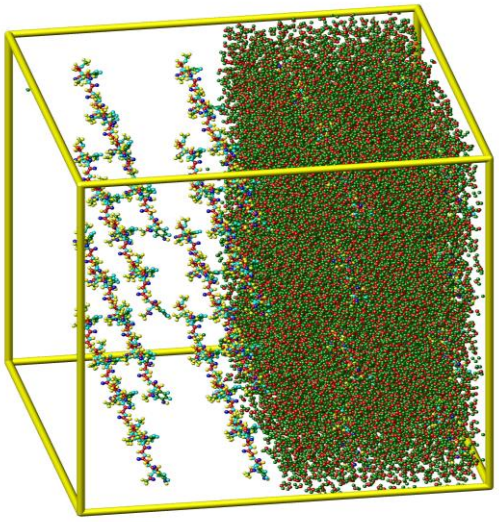
- Simple admin interfaces for configuring relevant AWS features
- Custom charts that showcase the value of AWS



SIMR

Team 199

HPC Cloud Performance of Peptide Benchmark Using LAMMPS Molecular Dynamics Package



"HPC software container-based cloud computing is an easy process compared to building and maintaining your own cluster in the cloud."

Figure 1: *Simulation snapshot using LAMMPS, studying the minimum energy configuration of peptide chains in a background water solution.*

MEET THE TEAM

End User – National Renewable Energy Lab (NREL), Tech-X Research
Software Provider – LAMMPS open source software and Steven J. Plimpton (Sandia National Lab)
Resource Provider – Amazon Web Services (AWS)
HPC Expert – Dr. Scott W. Sides, Senior Scientist, Tech-X Research Boulder, CO., Fetican Coskuner and Ender Guler, UberCloud.

USE CASE

In order to address realistic problems in the nanomaterials and pharmaceutical industries, large-scale molecular dynamics (MD) simulations must be able to fully utilize high-performance computing (HPC) resources. Many small- and medium-sized industries that could make use of MD simulations do not use HPC resources due to the complexity and expense of maintaining in-house computing clusters.

Cloud computing is an excellent a way of providing HPC resources to an underserved sector of the simulation market. In addition, providing HPC software containers with advanced application software can make the use of these codes more straightforward and further reduce the barriers for entry to small- and medium-sized businesses.

The molecular dynamics package LAMMPS is widely used in academia and some industries. LAMMPS has potentials for solid-state materials (metals, semiconductors) and soft matter (biomolecules, polymers) and coarse-grained or mesoscopic systems. It can be used to model atoms or, more generically, as a parallel particle simulator at the atomic, meso, or continuum scale.
Cloud computing service provider Amazon Web Services provided a number of virtual machines each with up to 16 cores for this experiment with different levels of network communication performance.

Technical Details of the Simulation

Figure 2 shows the parallel scaling performance of LAMMPS containers running on an AWS multi-node cluster with each of the nodes having 16 cores available. A simple peptide chain model that is included in the tests for LAMMPS was used for performance scaling. The initial peptide input file only contains 2004 particles, but using the 'replicate' keyword available in LAMMPS the initial simulation cell may be copied in the x,y,z directions an arbitrary number of times.

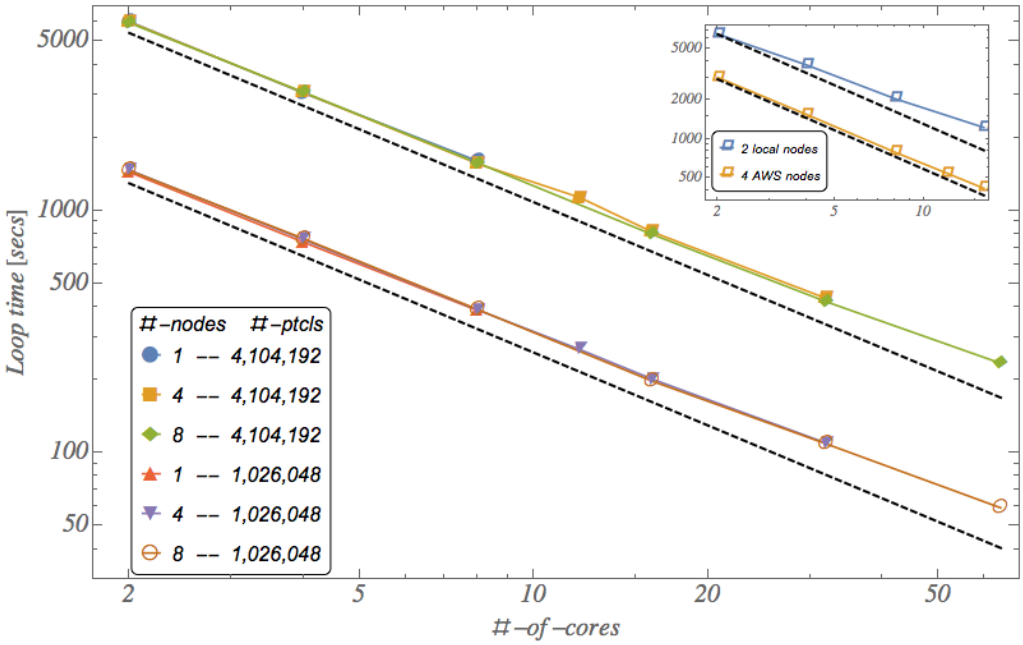


Figure 2: *LAMMPS parallel scaling performance on an AWS multi-node cluster with each of the nodes having 16 cores available. Inset upper right: Comparison of the parallel scaling performance between LAMMPS running on the bare-metal 2-node test cluster at Tech-X and LAMMPS containers running on a 4-node remote AWS cluster. The dotted lines indicate the optimal scaling behavior, showing that the performance of the LAMMPS containers running in the cloud is excellent.*

The simulations in Figure 2 show two system sizes using ≈ 10^6 and ≈ $4.1*10^6$ particles run for 300 update steps for reasonable timing statistics. The inset in the upper right shows a comparison of the parallel scaling performance for a system with ≈ $2.0*10^6$ particles between LAMMPS running on the bare-metal 2-node test cluster at Tech-X and LAMMPS containers running on a 4-node remote AWS cluster. The dotted line in the main figure and inset is the optimal scaling trend. The main figure shows that the LAMMPS multi-node container performance persists as the number of nodes in the cloud cluster increases. There was degraded performance when the number of processors/node reaches the maximum number of cores available as listed by AWS and is due to hyper-threading. But, there appears to be no degradation of performance as the size of the cluster increased, suggesting that an arbitrary number of processors can be used for HPC molecular dynamics simulations using LAMMPS in the cloud.

Summary of the SBIR project

This cloud experiment was initially funded as part of a Small Business Innovation Research (SBIR) grant. The solicitation called for enabling modern materials simulations in a larger sector of the industrial research community. High performance computing (HPC) is a technology that plays a key role in materials science, climate research, astrophysics, and many other endeavors. Numerical simulations can provide unique insight to physical phenomena that cannot be easily obtained by other means. Numerical simulations complement experimental observations, help in validating models, and advance our understanding of the world. Advances in HPC software development and algorithms are becoming increasingly important in materials science and for industries developing novel materials. According to a recent survey by the US Council on Competitiveness, faster time to market, return on investment, and enabling work that could not be performed by any other means are cited as the most common justifications for using HPC in industry. For instance, Goodyear was able to significantly reduce the time to bring new tires to market through a collaboration with Sandia National Laboratory by leveraging high performance clusters. The oil, aeronautic, and automobile industries are examples of big industries where HPC technologies have been leveraged for decades. The growing penetration of HPC into engineering fields has been fueled by the continued performance improvements of computer chips as well as the emergence of hardware accelerators such as general-purpose graphics processing units (GPUs) and the Intel Xeon Phi co-processor (also known as many integrated core architecture, or MIC).

However, one of most striking features of the US Council on Competitiveness survey, is how underrepresented are the companies that would be most likely to take advantage of soft materials simulations. The biosciences sector accounted for only 5.9% and the chemical engineering sector accounted for only 4.0% of respondents on their use of HPC resources. The Phase I SBIR proposal granted to Tech-X addresses this call and the two issues outlined above, by using an extensible object-oriented toolkit (STREAMM) for linking quantum chemistry (DFT) and classical molecular dynamics (MD) simulations and making this code suite available to take advantage of HPC cloud computing.

Process Overview

1. Kickoff team meeting of the experiment using WebEx.
2. Organization of project tasks, communication and planning through RedMine.
3. The end user, Scott Sides, obtained an AWS account and provided ssh-keys to UberCloud in order to setup a project specific security group that is used to configure the multi-node multi-container environment.
4. A specialized installer was created for LAMMPS and made available to the team.
5. The end user performed an MD scaling study on a 1-node, 4-node, and 8-node cluster.
6. The end user analyzed performance data and communicated the results to the rest of the team.

CHALLENGES

End user perspective - The cloud computing service at Amazon Web Services (AWS) provided high-quality compute nodes with efficient communication networks that enabled the good scaling seen in Figure 2. There is quite a bit of manual setup that needs to be performed by the end-user for AWS. For any cloud computing project, the first step is to create the remote compute instances. One must apply for an account at AWS, and use the AWS web interface to navigate to the services for the Elastic Compute Generation 2 (EC2). The 'elastic' refers to the ability to expand or shrink the hardware usage for a particular task at a given time. Then the desired number, type and security

settings for the EC2 instances must be selected. For a first-time setup, an ssh-key pair is generated and stored within the user's account information. The web interface instructs the user how to setup their local ssh configuration so that access to any remote AWS instance can be obtained. This procedure is straightforward but again, must currently be done manually. The security group must also be specified manually, and is one that is configured by UberCloud in order for the networking modules to function. Now the separate instances must be assembled and configured into a multi-node cluster.

The next steps are to copy setup applications, scripts and configuration files needed to install Docker, pull all needed Docker images, and start the computational images with all of the appropriate network configuration settings. The remote copy requires the DNS addresses generated by the AWS instance startup outlined above and must currently be performed manually. Then one of the compute instances must be designated as the 'Master' node which has two main purposes: (i) to run the 'Consul' container which is part of the framework that manages the network setup for all of the cluster instances and (ii) to provide a remote entry access point for the cluster. When launching simulations on this remote cloud cluster a user executes an SSH login command using the public IP address for the master node (again obtained manually through the AWS web tool) and a password that is automatically generated within the secure container and emailed to the user. These security measures are all part of the networking image layer in the UberCloud simulation containers. However, once these steps are in place, then running on a cloud cluster is much the same as running on an HPC cluster at a university or national lab.

BENEFITS

End user perspective

- Gained an understanding of the cloud computing philosophy and of what is involved in using a cloud-based solution for computational work.
- Cloud computing using novel HPC software containers based on Docker is an easy process compared to building and maintaining your own cluster and software environment.
- Developed an effective workflow for constructing additional HPC cloud containers.

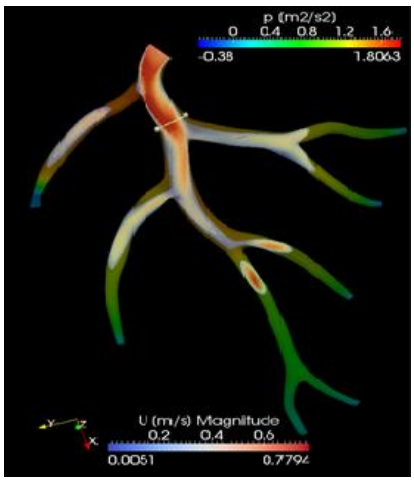
CONCLUSIONS AND RECOMMENDATIONS

For the Phase II proposal based on this case study, Tech-X will add additional codes to the UberCloud marketplace for targeted industries and applications including those in nanotech and the pharmaceutical industries. We will also investigate ways to add functionality to our STREAMM framework to streamline the setup steps described in the ‘end-user perspective’ section. We will also check all our current scaling results on the Microsoft Azure cloud platform and compare with AWS and bare-metal. The Azure setup is reported to have ways of streamlining the setup process to make utilizing cloud HPC resources even easier.

Case Study Authors – Dr. Scott W Sides and Wolfgang Gentzsch

Team 156

Pulsatile flow in a Right Coronary Artery Tree



"The UberCloud system was found to be far more user friendly than the NSF XSEDE supercomputing resources, in terms of both simplicity and ease of access."

MEET THE TEAM

End User – Prahlad G. Menon, PhD, Dept. of Electrical & Computer Engineering, Carnegie Mellon University

Resource Providers – Amazon AWS

Software Provider – UberCloud OpenFOAM container with OpenFOAM and ParaView.

USE CASE

Modeling of vascular hemodynamics has become increasingly popular for the assessment of the underlying patient-specific biomechanical traits of vascular disease. This modeling approach typically begins with the three-dimensional (3D) segmentation and reconstruction of a smooth surface model of the vascular region of interest from patient-specific medical image volumes obtained as a series of tomographic slices using either magnetic resonance imaging (MRI) or computed tomography (CT) imaging. The resulting 3D surface model is used to generate a discretized volume mesh for the purpose of analysis of flow using computational fluid dynamics (CFD) solution techniques.

In this study, blood flow inside a patient-specific right coronary artery tree (see Figure 1), including one inlet and a total of 7 outflow branches, was studied computationally under realistic unsteady flow conditions after segmentation from tomographic MRI slice images obtained across the whole human body of a male volunteer, at 2mm intervals.

METHODS

A finite volume mesh consisting of 334,652 tetrahedral elements / cells was prepared in the mesh building toolkit, GAMBIT (Ansys). This mesh was then passed over as input to the icoFoam solver in OpenFOAM – a free, open source CFD software package – to solve for hemodynamics.

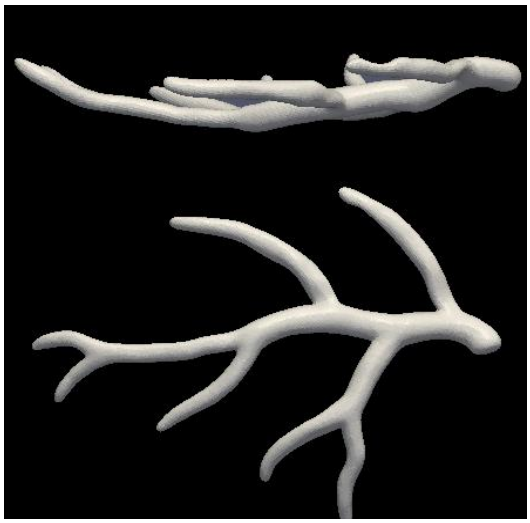


Figure 1: 3D reconstruction of the studied patient-specific human right coronary artery tree.

Hemodynamics (meaning literally "blood flow, motion and equilibrium under the action of external forces") is the study of blood flow or circulation. It explains the physical laws that govern the flow of blood in the blood vessels under pulsatile inflow conditions. A realistic right coronary artery inflow waveform was obtained from the scientific literature based on reports from catheterization studies for the purpose of this CFD simulation study (see Figure 2).

The solution was obtained assuming a plug-flow inlet velocity profile, zero-pressure outlet boundary conditions and rigid, impermeable, no-slip arterial wall boundaries. The choice of equal, zero-pressure outflow boundary conditions resulted in a naturally distribution of blood flow to the seven outlets of the geometry, based on their respective intrinsic geometry-based resistance pathways.

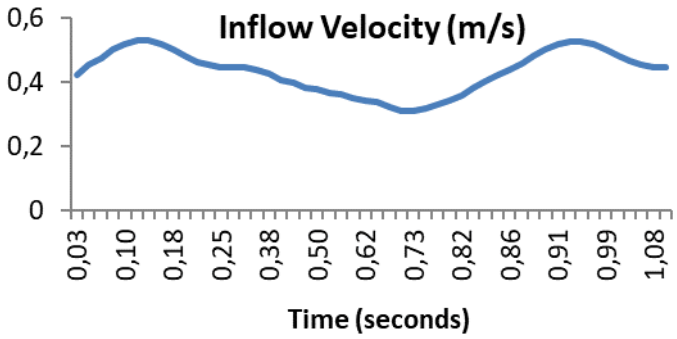


Figure 2: Realistic pulsatile right coronary artery inflow waveform considered in this study.

Blood was modeled as a Newtonian fluid with constant density (1060 kg/m3) and viscosity (0.00371 Pa.s). One full cardiac cycle was simulated with a time step of 0.00001 seconds, with first order implicit time integration and second order Gaussian integration based finite volume spatial discretization with a linear cell-to-face center interpolation scheme. The generalized geometric-

algebraic multi-grid (GAMG) solver was chosen for this study for its properties of being faster than most standard solvers. This is due to its strategy of first generating a flow solution for velocity and pressure on a coarser mesh using a method of agglomeration of cells, and then mapping the coarse solution onto the full (i.e. fine) mesh to obtain a more accurate solution. The solution was obtained using the remote UberCloud multi-core computing framework – including UberCloud's new software containers – after first parallelizing the problem automatically on to 16 cores using OpenFOAM's computational domain decomposition and then submitting the job (including all necessary input files) via a secure shell (SSH) client.

RESULTS AND DISCUSSION

Pulsatile flow results were examined using ParaView, running on a remote visualization machine. Fig. 3 shows an instantaneous snapshot of flow in the coronary artery tree under peak flow conditions.

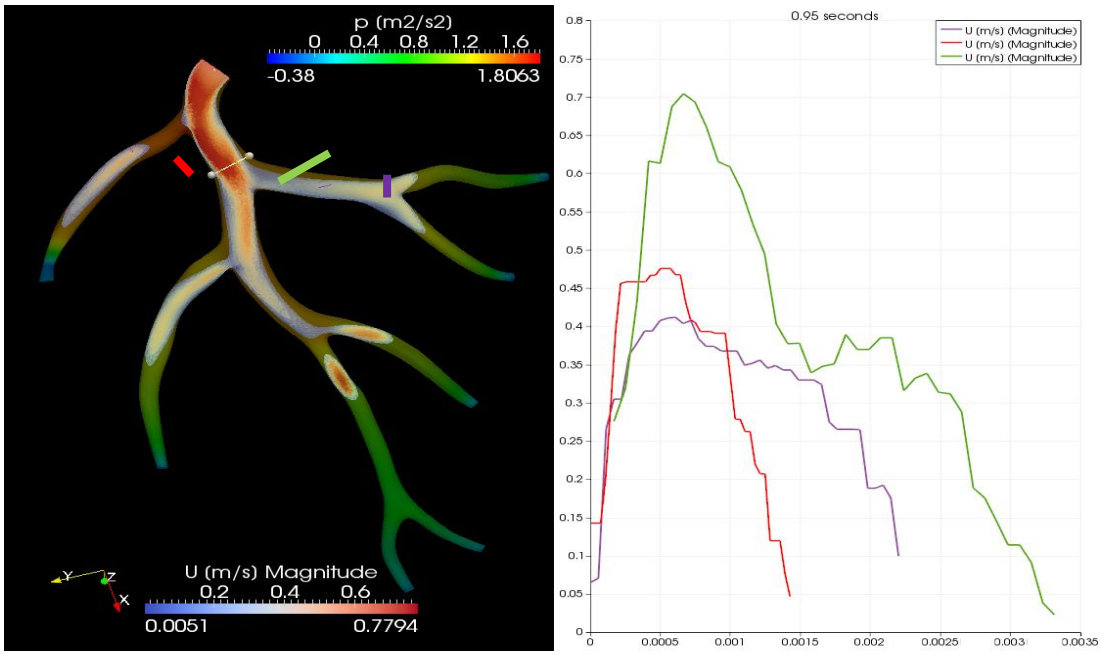


Figure 3: Visualization of hemodynamics at the instantaneous peak-flow instant, in ParaView. The full time-resolved simulation result is available for viewing at: http://www.youtube.com/watch?v=hcS_k9xhfIo

Shown on the left of Figure 3, the velocity through the inlet is visualized through a longitudinal slice (blue to red color-scale) superimposed with a translucent rendering of the arterial wall colored by the arterial pressure field (green to red color-scale). The units of velocity are meters per second (m/s), whereas pressure is plotted in m2/s2, which is equivalent to Pascals per unit density units (i.e. as Pascals per unit density units = $(kg.m/s^2) / (kg/m^3) = m2/s2$; where density = 1060 kg/m^3). On the right of Figure 3, velocity profiles through three different diametric lines across the geometry (marked in the figure on the left), are shown. The x-axis of the velocity profile plot is distance measured along the diameter of vessel segment (in SI units of meters), across which the velocity profile is visualized. Realistic pulsatile flow profiles which resembled expectations of Womersley flow profiles were possible to observe over the simulated cardiac cycle.

Finally, the average simulation speed under 16-core parallelism was analyzed as a function of clock time taken per unit of simulated pulsatile flow time. This worked out to be ~34 clock hours per simulation second on the UberCloud Nephoscale system, as illustrated in Figure 4.

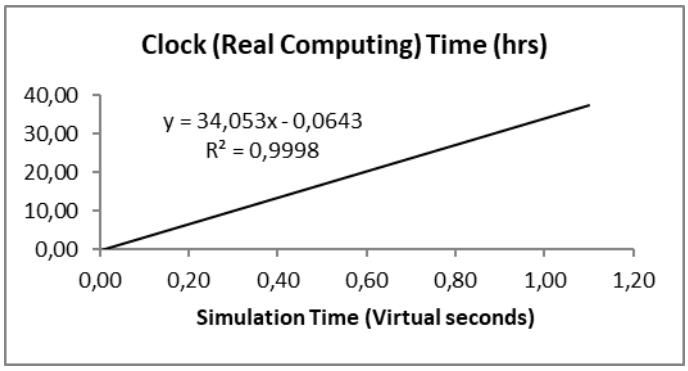


Figure 4: Illustration of the relationship between clock time (in hours) and simulated pulsatile flow time (in seconds) in the coronary artery.

BENEFITS

The UberCloud system was found to be far more user friendly – in terms of both simplicity and ease of access (via SSH) – than the National Science Foundation (NSF) XSEDE resources, which I use on a daily basis simply because the OpenFOAM cases ran without any issues with MPI libraries. On the other hand, access to XCEDE resources is less expensive than commercial cloud resources.

Parallel scalability of the simulated OpenFOAM problem was found to be quite good on the remote computing resource. A 2- to 3-fold speed improvement was noted using 16 core parallelism on the remote UberCloud machine in contrast with an equivalent local simulation run on just 4 parallel cores of a local quad-core machine.

CONCLUSIONS AND RECOMMENDATIONS

First, I like the ease of access and simplicity of the UberCloud system and would consider it as a preferred computation system if up to 128 cores were available for parallel simulation. CFD simulations generate large data volumes; it would be ideal if the UberCloud system could facilitate remote visualization of results without having to download large amounts of data to a local machine for further analysis (Now the UberCloud Container comes with ParaView and remote viz integrated).

I used SSH to download and locally process the results. This was accomplished with consistent upload/download speeds of ~700 KBps – which is pretty good. While it is possible to get 10+ MBps, inside a local network 700 MBps is a good speed for transferring data to a remote server. Availability of ParaView as a visualization software along with VNC (remote desktop) access capabilities would enhance ease of uploading / downloading simulation data and simultaneously facilitate remote processing as well as visualization of simulation results.

Second, it is very important to have sudo / root privileges for some installation and compilation procedures in the interest of customizing boundary conditions and solver settings in OpenFOAM. For example, if you were to use a 3rd party library like *Bison* for a boundary condition implementations in OpenFOAM, at the present time you would have to contact the UberCloud administrator in order to install this library and subsequently add access to this library to your Linux user environment. Most computational specialists require compiling and using their own code, and install 3rd party libraries often. Therefore, the lack of sudo-user privileges to install software (e.g., using the apt-get

command) or customize environment variables may be seen as a limitation to an advanced user of the UberCloud environment, when the user is not the administrator (i.e. 'root').

Case Study Author – Prahlad G. Menon, PhD, Dept. of Electrical & Computer Engineering, Carnegie Mellon University

Team 147

Fast and Cost-Effective Compressor Map Generation Using Cloud-Based CFD

"A compressor map of 39 operating points was obtained in approximately 2.5 hours at a total cost of $580 in the cloud."

MEET THE TEAM

End-user: Anonymous independent consultant that does not have the interest or the financial capability to invest in large clusters and CFD licenses.
Software Provider: Michael Ni – Product Manager - AeroDynamic Solutions Inc. - a provider of CFD software and consulting services.
Resource Provider: AWS Amazon Web Services.

USE CASE

When analyzing or designing centrifugal compressors, understanding the performance characteristics of the compressor is a significant challenge for any individual or small business without the hardware and software computing resources of their larger industry counterparts. Many complex simulations must be performed, and often the individual must tradeoff between turnaround time (and a reduced number of design iterations) and software and hardware expense.

The end-user in this case study was interested in understanding the effectiveness of cloud-based CFD for analyzing the NASA CC3 compressor. The NASA CC3 compressor consists of

an impeller with a splitter and a vaned diffuser and is representative of a typical centrifugal compressor configuration.

END TO END PROCESS

Cloud Leo is a cloud-based turbomachinery CFD simulation service based on the RANS solver Code Leo and designed to take advantage of Amazon Web Services. Designed for compressor and turbine aerodynamic design, Cloud Leo allows users to provision CFD analysis capacity on demand and use it on a pay-as-you-go basis. Cloud Leo runs on cloud computing infrastructure provided by Amazon Web Services (AWS). AWS has many years of experience in designing large data centers and has comprehensive policies in place to ensure physical security, secure services and data privacy. AWS is SAS 70 Type II certified, ISO 27001 certified and PCI compliant. Cloud Leo simplifies the process by automatically provisioning instances, loading the appropriate AMI, and configuring the firewall.

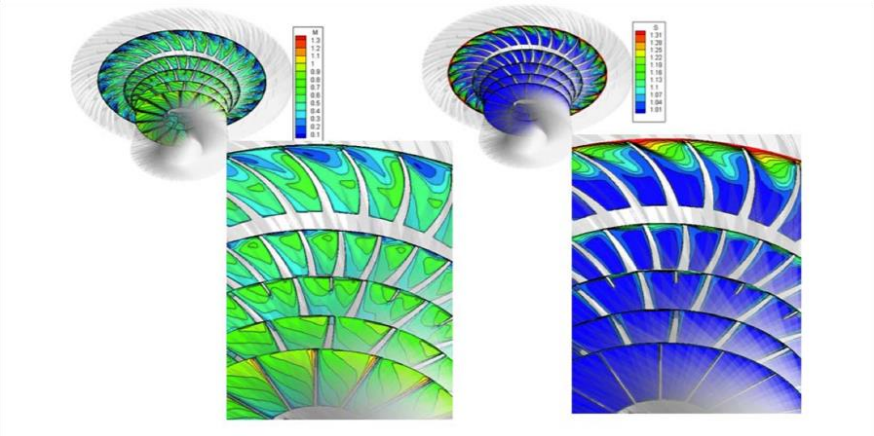


Figure 1: Mach Number and Entropy Contours for the NASA CC3 simulation.

The service is accessed through a graphical workbench running on a local computer. The workbench enables cases to be configured locally—including mesh generation with Code Wand—and transferred securely to specially developed AWS AMIs for cloud-based execution. The workbench contains a queuing and batching system to ensure that cloud resources are not over-subscribed and are utilized efficiently. As execution occurs, the status of the simulation can be monitored within the workbench. When cases complete, results are downloaded securely back to the workbench for post- processing and permanently deleted from the AWS instance as an added measure of security.

A user begins by simulating an operating point at the design point condition for each of the 5 operating conditions. The initial points are then examined for convergence prior to generating the rest of the map. Once validated, the workbench generates 39 operating points configured to run in parallel on 15 cc2.8xlarge AWS EC2 instances. These instances are equivalent to 2 Intel Xeon ES-2670 processors with 16 cores, 60GB of RAM, and interconnected via 10Gb Ethernet. The simulations completed in 2.5 hours including transfer of data to and from the instances.

CHALLENGES

There were many challenges in making the service seamless to the end-user. To begin, an intensive study of the effects of various AWS instance types was done. It was found that outside of the compute cluster instances, the network delays of the less expensive virtual instances were so large that it made using them impossible for any significant computations.

Another significant challenge was in developing the user experience so that it was intuitive and simple. The target end-users were not expected to understand IT infrastructure, networking, securing data over the Internet, or resource load balancing. This challenge was a significant source of development effort as it was built to be resource provider agnostic.

Lastly, architecting and implementing the licensing structure such that it was truly charged per hour and secure against fraud was a challenge. Our licensing technology had to be re-implemented to handle metered simulations as nothing existed that could be integrated with our software.

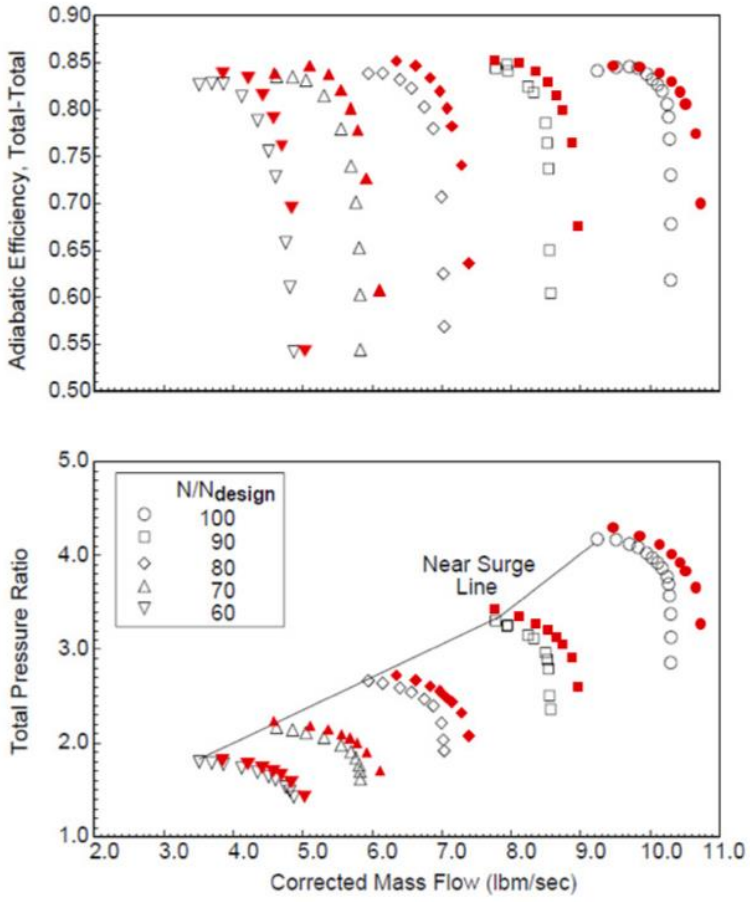


Figure 2: CFD Results compared against experimental data.

BENEFITS

The end-user benefitted from the team collaboration and the knowledge gained from this project. These include:

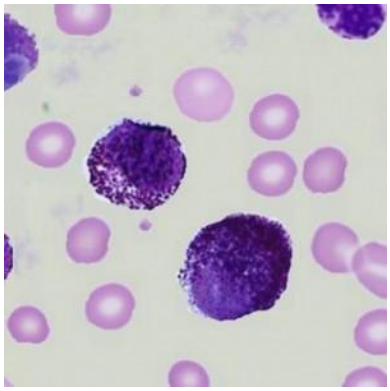
- The costs of demanding CFD simulations will not exceed their budget.

- The user can now scale CFD related expenditures as demand rises, rather than before demand exists.
- The user can apply CFD technologies that were not previously available to them.
- The user can turn around CFD design and analysis in the same design windows as their larger competitors.
- The user can now size their CFD IT expenditures to their typical simulation load rather than sizing to their peak load.

"UberCloud software container images were easily ported between different environments by the Operations Engineer."

Team 143

Open Source Clinical Cancer Genomics Pipeline in the Cloud



MEET THE TEAM

End-User: Erinija Pranckeviciene, Bioinformatics Research Scientist, Department of Human & Medical Genetics, Vilnius University

Resource Provider: Amazon AWS

HPC Cloud Experts: Fethican Coskuner, Operations Engineer, UberCloud.

USE CASE

Rapid progress in next generation sequencing is paving the way to individualize cancer treatment based on sequencing of the cancer genome. This process is still evolving and validated open source pipelines to process the enormous amounts of data involved are not available to everyone interested in this field.

Identifying informative genomic variants – potential disease-causing candidates – through the use of next generation sequencing (NGS) data is becoming a routine diagnostic tool. In cancer, exomes of

tumor and normal tissues aligned to the reference genome serve as primary data source to start identifying these genomic variants for a better understanding of the disease.

Software programs used to align NGS data to the reference genome are governed by user-defined parameters and map the short sequenced reads to the reference genome with varying accuracy [4]. Most of the alignment tools require a long time to obtain accurate mappings of the short reads to the reference genome.

Decisions as to which of the alignment tools have to be included into the NGS data processing pipeline depend on many factors such as ease-of-use of tools, their public availability, and their ability to align NGS data from various sequencing platforms (Illumina and AB Solid). Among those factors the time required to accomplish the alignment plays a very important role. The time required for the alignment program to complete the mapping, depends on the size of the NGS data set and scales with available computational resources.
The main goal of this project was to start preparing guidelines on the optimal choices of open source alignment tools to be included into the NGS data processing pipelines by matching these tools to the available computational resources. We started by estimating how much time the alignment tools needed to complete the task of alignment of the cancer data-set to the human reference genome by running these tasks on various configurations of high performance computing (HPC) resources.

At this stage two aligners were chosen: Mapping and Assembly with Qualities (MAQ 0.6.6 and 0.7.1) [1] and Short Read Mapping Package (SHRiMP 2.2.3) [2].

MAQ is one of the most accurate aligners and supports gaped alignments of paired end reads. However, MAQ was programmed in such way that it can't take advantage of available multiple-core processor architectures.

SHRiMP is an exhaustive mapping tool because it finds multiple candidate alignment locations (CALs) and then selects the best hits out of those CALs. SHRiMP can fully utilize and take advantage of the available memory and multiple-core processor architectures. The computational resources used in this project are listed in Table 1.

Table 1: Computing Resources

A) AWS	**B) BARE METAL #1**	**C) BARE METAL #2**	**D) WORKSTATION AT THE DEPT. OF HUMAN AND MEDICAL GENETICS, VILNIUS UNIVERSITY #3**
CPU: Intel® 8 CPU cores (Model is unknown) RAM: 68GB	CPU: Intel Xeon CPU E5-2650, 32 cores RAM: 32GB	CPU: Intel Xeon(R) CPU E5620, 16 cores RAM: 144GB	Allocated resource: CPU: AMD Opteron (tm) Processor 6172, 12 cores RAM: 98GB
OS: Ubuntu 12.10 LTS Server MAQ: 0.6.6	OS: Ubuntu 12.04.2 LTS Server MAQ: 0.6.6 SHRIMP: 2.2.3	OS: Ubuntu 12.04.4 LTS Server MAQ: 0.7.1 SHRIMP: 2.2.3	OS: CentOS 5.10 MAQ: 0.7.1 SHRIMP: 2.2.3

To enable portability of the computation environment (including the aligners, scripts, configuration settings) Linux container images were used. The same image was used in the AWS, Bare Metal #1, and Bare Metal #2 test. The Linux container images were easily ported between environments by the Operations Engineer.

PROJECT OBJECTIVES

Objective 1 – Document how much time it took for MAQ and SHRiMP to align 2 million 75 base-pair (bp) length paired end sequence reads of cancer exome data [3] to the human reference genome. The programs were tested on several HPC configurations summarized in Table 1.

Objective 2 – To compute a complete mapping of the mast-cell leukemia exome data (tumor and germline available in Illumina Genome Analyzer FASTQ format) [3] to the Human Genome Build 37 (GRCh37 hg19).

RESULTS

Total number of paired end reads in a whole sample consisted of 32,971,877 reads in the germline exome and of 37,495,826 reads in the tumor exome. To compare chosen alignment programs in terms of their execution time on different HPC configurations, 2 million paired end reads of the germline exome was used. Durations in which MAQ completed alignments of 2 million reads on tested computing resources are shown in Table 2. The shortest duration was achieved on the HPC configuration **C**. Next, MAQ was tested by using UberCloud Linux container images on HPC architecture C. The alignments of 2 million paired end reads were executed simultaneously on 4 nodes. Each completed in 2 hours and 5 minutes.

SHRiMP can fully utilize multiple-core processor architectures. We were interested in how much the alignment time could be reduced by using the aligner, which takes advantage of available multi-core computing resources. SHRiMP can use as many threads as set by a parameter. Durations in which SHRiMP aligned 2 million reads on different HPC architectures are shown in Table 2.

To summarize the comparative results, we note that MAQ completed the task in 75 minutes and SHRiMP completed the same task in 40 minutes by using 16 threads. Both of them completed the alignment of 2 million paired end reads within similar time range.

Table 2: Running times of the alignment tasks

ALLIGNMENT TASK	A) AWS	B) BARE METAL #1	C) BARE METAL #2	D) WORKSTATION AT THE DEPT. OF HUMAN AND MEDICAL GENETICS, VILNIUS UNIVERSITY #3
MAQ (2 mln reads) version 0.6.6	cr1.8xlarge more than 2 days experiment stopped			
MAQ (2 mln reads) version 0.7.1	m1.medium 2 h 20 min	1h 30 min	1 h 13 min	4 h 30 min
SHRiMP (2 mln reads)				

using 1 thread	3 h 20 min
using 16 threads	1 h 40 min
using 24 threads	38 min

We estimated that the time required for MAQ to complete the mapping of 2 million reads on the architecture C within the virtualization layer takes a little over two hours. Based on this estimate, we predicted that time, required to map exomes of germline and tumor divided into 36 parts of 2 million reads each, should take approximately 36*2= 72 hours. By using 4 computing nodes simultaneously for this task the mapping should complete in 18 hours. The estimated time for SHRiMP using 24 threads to map 2 million reads takes about 40 minutes. Based on this estimate, we predicted that SHRiMP will align the whole sample in 25 hours. A summary of the actual time spent in the mapping of whole sample on the HPC architecture C is shown in Table 3.

We used 24 threads in mapping with SHRiMP. The mappings were performed in batch on a single node. This mapping task completed in 29 hours, which is more or less consistent with previous expectation of 25 hours that were predicted for the whole sample based on the time estimate of mapping 2 million reads (see Table 2).

Mapping by MAQ was performed using UberCloud Linux containers on four nodes. Two nodes were used to map the germline and another two nodes were used to map the tumor sample. Alignment of the whole sample by MAQ completed in 35 hours, which is almost twice as long as expected.
There are several reasons why MAQ had prolonged mapping time. The first is that, in general, reads have different sequence complexity, which influences mapping time. The average actual mapping time of a part of 2 million reads was approximately three hours. The second reason was unexpected interruptions in a batch mapping because of unexpected symbols in some parts of the data. The second reason is minor – the data was cleaned and processed again.

Table 3: Alignment of the whole exome of germline and tumor by using HPC configuration C

Alignment program	Germline sample 0-16 parts x 2million reads	Tumor 0-18 parts x 2million reads
MAQ version 0.7.1 utilized 4 nodes in parallel	In batch: Node1 (part 0-6) From Jun 7 21:02 Till Jun 9 07:42 Node2 (part 12 - 14) From Jun 9 02:38 Till Jun 9 11:57 Realigned: Parts 7,8,9, 10,11,14,15,16	In batch : Node3 (part 0-6) From Jun 7 21:04 Till Jun 9 10:42 Node4 (part 10-16) From Jun 7 21:04 Till Jun 9 07:18 Realigned: Parts 7.8,9 17,18
SHRiMP ver 2.2.3 using 24 threads on 1 node	From Jun 7 22:02 Till Jun 8 11:59 Total 14 hours	From Jun 8 12:57 Till Jun 9 04:00 Total 15 hours

CONCLUSION AND RECOMMENDATIONS

The team gained two particularly useful insights from this project:

Some software programs are created in a such a way that they can't take advantage of modern multiple core architectures. Therefore, a virtualization approach was utilized to perform mapping with simple alignment program MAQ simultaneously on four nodes. Running simple programs in virtualization layers on HPC resources proved to be a very productive use of those resources resulting in a considerable reduction of execution times of simple processes. In NGS data analysis, numerous useful software programs exist that can't use modern HPC architectures. Using these programs to process big data is challenging because of the time required to obtain results. One possible solution to this challenge is to use the UberCloud Linux containers on available HPC architecture.
The predicted mapping times resulting from estimates of mapping times computed on small randomly selected sample of reads are only approximate. This most likely occurs because sequence complexity impacts finding a candidate alignment location of the read in the reference genome.

Alignment of the short reads to the reference genome in NGS exome analysis pipeline is only a single intermediate step in obtaining information on which genomic variants are present in the exome. A second step after the alignment is search and identification of those genomic variants. This step is computationally demanding as well. It would be beneficial to assess a computational intensity and create resource usage guidelines for genomic variant identification tools such as Genome Analysis Toolkit (GATK), Samtools and MAQ.

REFERENCES

1. Li, H, Ruan, J, Durbin, R (2008). “Mapping short DNA sequencing reads and calling variants using mapping quality scores.” *Genome Res.*, 18, 11:1851-8.
2. David, M, Dzamba, M, Lister, D, Ilie, L, Brudno, M (2011). “SHRiMP2: sensitive yet practical SHort Read Mapping.” *Bioinformatics*, 27, 7:1011-2.
3. Spector, MS, Iossifov, I, Kritharis, A, He, C, Kolitz, JE, Lowe, SW, Allen, SL (2012). “Mast-cell leukemia exome sequencing reveals mutation in the IgE mast-cell receptor b chain and KIT V654A.” *Leukemia, 26(6):1422-5.* PMID:22173243
4. Hatem, A, Bozda, D, Toland, AE, Catalyurek, UV (2013). “Benchmarking short sequence mapping tools.” *BMC Bioinformatics*, 14:184, p.1-25.

Case Study Author – Erinija Pranckeviciene

Team 116:
Quantitative Finance Historical Data Modeling

"This experiment used the cloud for processing large amounts of data in a massively parallel fashion."

MEET THE TEAM
End Users – These are thousands of end users around the world, using the back-testing service through a web browser.
Software Provider - Jared Broad, Founder and CEO of QuantConnect Corporation, a start-up company based in New York City providing an online back-testing service.
Resource Provider - Amazon AWS.

USE CASE
This experiment used the cloud for processing large amounts of data in a massively parallel fashion. We have terabytes of financial data, and can scale unlimited nodes to process it in finite time periods. Users access this cloud through a coding environment in a web browser. A financial back-test is a process of testing algorithms on historical time series data to verify a trading strategy.

Currently it is difficult to process financial data as an individual or on your home PC due to the large volumes of data created each day and the very high expense to purchase financial data. The Software Provider QuantConnect makes the data freely available in a cloud platform, allowing the user access to the data temporarily. It is not uncommon to hear stories of back-tests taking 24-48 hours on users home PCs.

The experiment end-user required extensive processing power to quickly analyze six years of a FOREX tick data, equating to roughly 500MB per simulation. The user would design an algorithm and then perform 10-20 simulations on the dataset, often slightly adjusting the algorithm to optimize variables of different simulations. This problem was ideally suited to a cloud of nodes, where each node processes one long running simulation.

Over the period of two months the user's back-tests were performed and the logs were analyzed. The user's strategy iterated over 10 times during the period, and he performed roughly 1000 hours of compute in total on powerful nodes.

End User - The typical profile of an end-user is either a financial engineer working in a bank or hedge fund, who designs algorithms in his free time, or a general software engineer who has an interest in trading and automating his strategy. They are generally very educated individuals. One such user is Eyal Zach, a C and C++ embedded systems engineer and quality assurance engineer who designed a FOREX trading system inside QuantConnect.

CHALLLENGES

Amazon EC2 cluster performed well and each node was robust during the trials. Our key limitation was in scaling quickly to user demand as the nodes took several minutes to load an image. This load period was unacceptable so a constant baseline of nodes was required which added to the project cost.

Costs - Using AWS comes with significant cost, which is highly unpredictable and depends on the server demand and complexity of the algorithms we process. QuantConnect made significant innovations to lower the overhead costs. The primary result is an advanced load prediction and machine image scaling algorithm, which learns from historical usage per user and accounts for many environmental factors.

AMI Wait Time - The primary reason for the evolution in the load balancing algorithm was the long delays in machine image spin up time. Amazon currently takes between 3-5 minutes to load the 30GB machine image. This is an unacceptable wait time for an end-user looking to get quick back-test results.

With a fully scaled cloud, running a massively parallel back-test we were able to process 5GB of data within minutes, a process which would normally take 12-24 hours on a home desktop computer.

DLL Upload - The software provider recently launched DLL upload support [1] so registered users could compile the algorithm locally on their PC and upload the DLL with any supporting supplementary files they require. It was a challenging process to ensure the security of the nodes from 100% arbitrary code.

We solved this problem by making the algorithm worker nodes 100% disposable for paying clients. After a single run the potentially infected nodes are disposed and a fresh set are created

for the next user. It takes machine images use scenario to the extreme – potentially only using a worker node for 10 minutes before disposing of it.

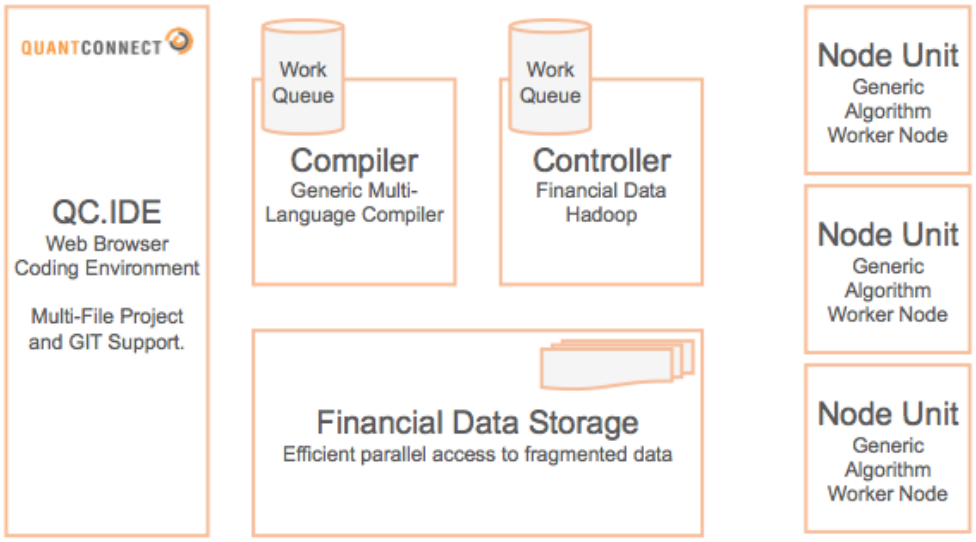


Figure 1: Block Diagram of the QuantConnect platform on AWS.

CONCLUSIONS AND RECOMMENDATIONS

We recommend there be a global standard version of Hadoop developed for financial data. It is quite a unique problem as the data storage and retrieval of a typical Apache Hadoop is relatively inefficient. Therefore, this experiment used QuantConnect's own proprietary implementation specifically for fragmented time series data and implemented a custom Hadoop controller to splinter the jobs and collect the result data.

There is potential for an easy instantly scalable cloud based from Amazon machine images. Users expect responses within 1-2 seconds and the image turn on time of EC2 is unacceptable for user interaction. There is also potential for a private company to offer an alternative to Amazon Simple Queue Service (SQS). SQS is slow, expensive and has limited data storage capacity. We would gladly substitute Amazon for a private company offering a better product that is in line with SQS API. Alternative open source queues such as RabbitMQ don't use a REST API and are often blocking designs, which require code changes to implement in the platform.

Through fragmenting data and massively parallel analysis, the solution provider was able to speed up time series analysis several orders of magnitude. This has the important limitation that each node unit is ignorant of the past. There is no easy solution to make the nodes aware of the past data or calculations as most efficient formulas are calculated online (on streams of data), which requires the data to appear in order.

Some end users required series analysis (knowledge of past events), so they preferred to use QuantConnect cloud to run 10+ versions of the same algorithm concurrently to optimize algorithm variables.

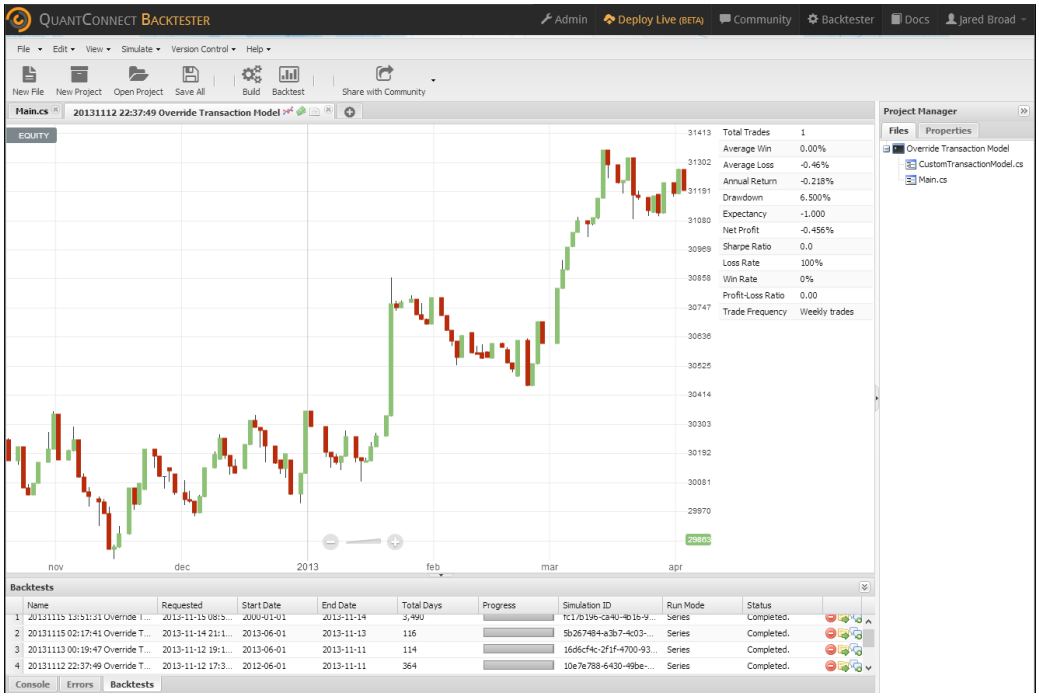


Figure 2: Result of the data analysis is displayed through a rich graphical UI.

REFERENCE

[1] https://www.quantconnect.com/blog/algorithm-sharing-private-groups-and-dll-upload-support/)

Case Study Author – Jared Broad

Team 70:
Next Generation Sequencing Data Analysis

"A good working relationship with our internal IT group, including Networking and Information Security, as well as familiarity with their processes, was essential for the success of this project."

MEET THE TEAM

End User - A medical devices company with over 20,000 employees and products in diagnostics, medical and biosciences markets.

Software Provider - Brian O'Connor, CEO, Nimbus Informatics, which provides consulting and cloud execution services for workflows that utilize the open source SeqWare application.

Resource Provider - Amazon Web Services (AWS).

HPC/CAE Expert – Ted Pendleton, Carl Chesal, Ian Alderman, James Cuff, Rob Futrick, Cycle Computing.

USE CASE

In this experiment, we explored using cloud computing for next-generation sequencing data analysis. We used SeqWare, an open source framework, running on AWS to perform variant calling. Variant calling is the reporting of differences between input sample DNA and a reference genome, using targeted exome data generated in-house on Ion Torrent's Personal Genome Machine.

Next-generation sequencing (NGS) is a relatively new high-throughput sequencing technology revolutionizing medicine. Its accuracy and breadth of coverage makes it useful in diagnosing and treating hereditary diseases and cancer, as well as infectious disease. However, the advantages of NGS lead to corollary challenges: storing, sharing, and analyzing the data strains disk space, network bandwidth and processor resources.

Cloud computing, though no panacea, can alleviate some of these problems by flexibly providing virtualized compute and storage resources on-demand. Cluster computing, running on top of virtualized infrastructure, as provided by Cycle Computing, can assist in coordinating the compute and storage resources for high-throughput applications, such as processing NGS data. Bioinformatics workflow services, such as those provided by Nimbus Informatics, have emerged. These services, built on top of infrastructures like AWS, promise to provide an end-to-end platform that automates

storing, organizing, and processing large amounts of data in the cloud. Our group has several Mac workstations, and several Linux servers that adequately served the needs of the group before the advent of our NGS work. Although we had unused local server capacity to perform most of the current NGS data analysis, we could see that additional storage and compute capacity would be needed soon.

Our initial NGS plan was to procure a medium-size cluster with storage and HPC capabilities. However, management decided against up-front capital expenditure, so we investigated cloud-based options that could be procured on an as-needed basis, which would cost less initially and provide flexibility if project activities changed. Although the strategy had advantages in flexibility, it also assumed:

- Adequate connectivity to cloud resources from our facility
- Cooperation from IT & security groups to approve and facilitate putting data outside the firewall
- A significant acquisition of skills by our team

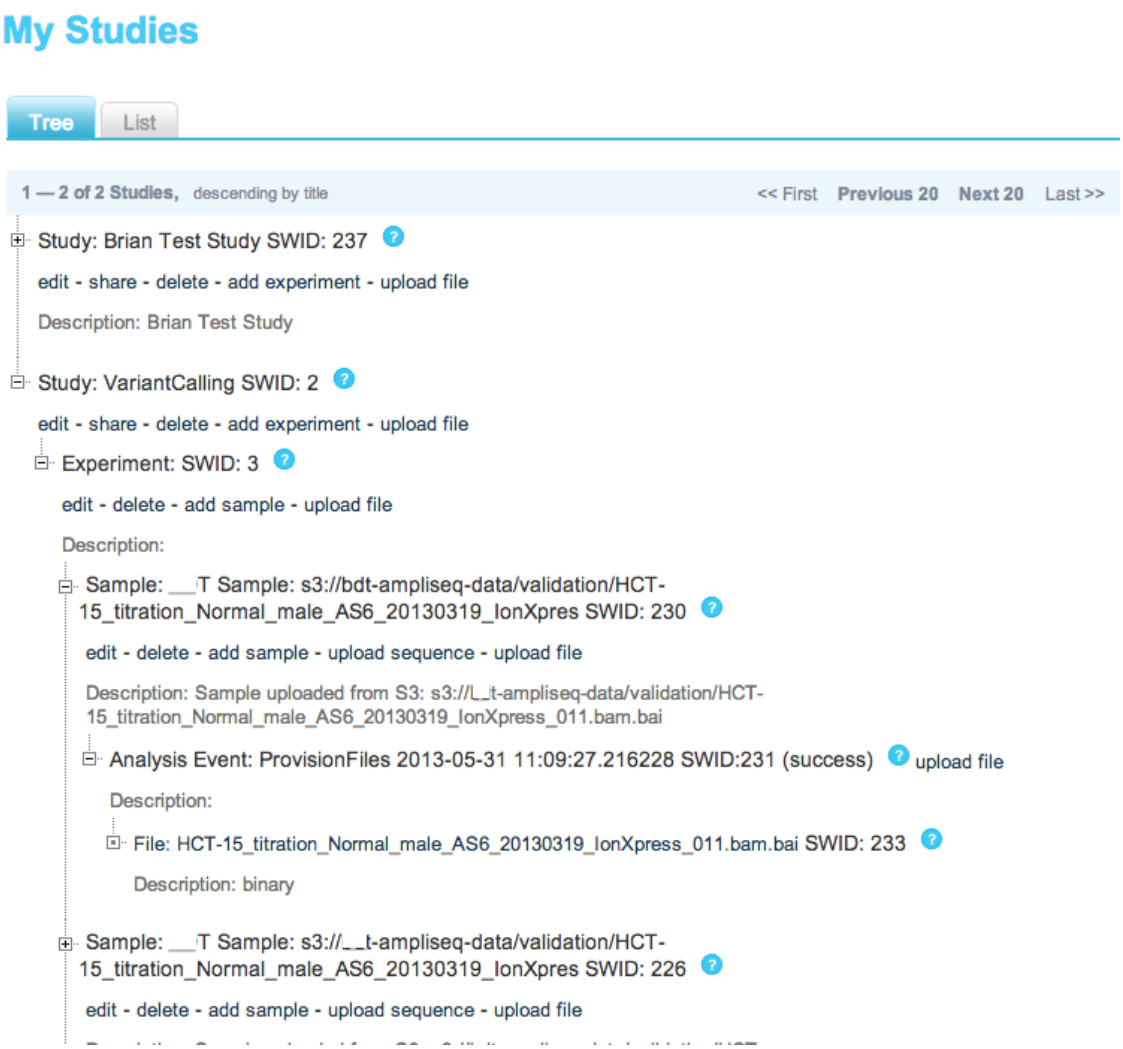


Figure 1: User data is organized hierarchically in SeqWare, with top-level Studies objects that contain one or more Experiments, which contain Samples. Samples map one-to-one to data files, and also have the workflow analysis results attached to them. They keep a detailed record of all analyses performed.

This project with the HPC Experiment is one of the ways our team is developing the necessary skills, and we are investigating options for cloud-based NGS data management and analysis. The project definition phase consisted of consulting with HPCExperiment staff to comment on our initial problem statement. They suggested a team, which began with Cycle Computing as the domain expert. Shortly after that, Tom Dyar learned about SeqWare and Nimbus Informatics, which provides a complete genome-center scale and decided to add the principal Brian O'Connor to the team. The work plan to achieve the goals included:

- Launch SeqWare AMI and connect to it via SSH through the corporate firewall
- Add ~10 GB NGS sequence raw data samples and annotate within SeqWare MetaDB
- Construct variant calling pipeline, using SeqWare Java library
    - Organize components such as diBayes and ion torrent GATK jar
    - Write/Debug Java code and test on standard BAM file
    - Upload bundle to SeqWare
    - Test on standard BAM file
- Run pipeline on samples, in parallel, and record time to complete, and other metrics
    - Brian O'Connor and Nimbus assisted in providing the team with updated AMI's, accounts on their hosted service, and set up our customized AMI running within Nimbus' infrastructure, so that our custom workflow could easily be run. Cycle Computing was helpful in suggesting ways in which other customers of theirs have performed large-scale sequencing work on AWS.

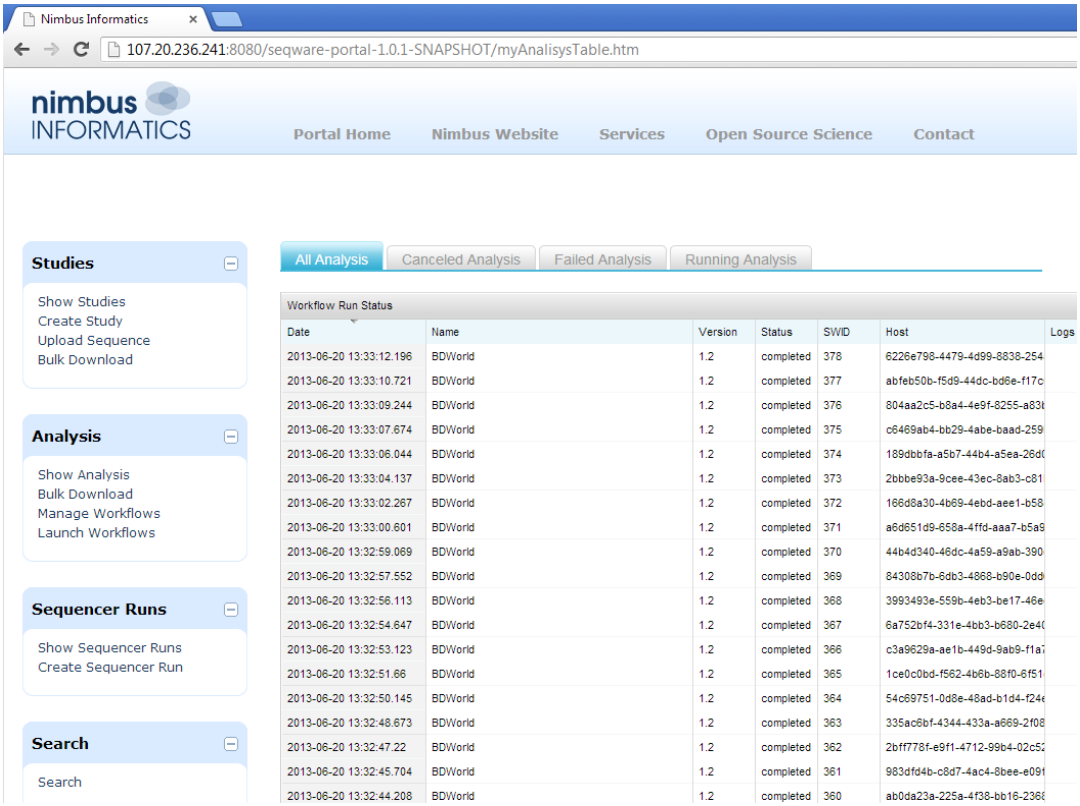


Fig 2: Workflow run status. We ran approx.. 30 workflows simultaneously on Nimbus' AWS-based system.

CHALLENGES

The primary challenges we faced revolved around getting good connectivity from our IT group, and approval to proceed with UberCloud's HPC Experiment from our management and corporate legal team. Secondary challenges were technical – learning and using SeqWare.

As our group performed more work on the AWS infrastructure, we encountered several connectivity issues. Typically, we requested ports be opened to support various transfer protocols such as VNC or Remote Desktop Protocol (RDP). Our company also maintains a proxy server through which most traffic passes. The server performs virus and content scanning, which has caused issues such as corrupting large files when downloaded, because the scanning process took so long that the connection timed out. Also, we have adapted to changes made by our information Security group to increase security, such as closing the SSH port 23, which we were using to connect to EC2 instances on AWS.

Finally, we had a globally distributed team with associates in Singapore using a slightly different networking infrastructure that needed modifications. For example, while diagnosing slow connectivity to the Singapore AWS datacenter, we discovered that traffic was being routed first back to the eastern United States before reaching the AWS datacenter. All of these issues typically involved one or several requests to IT, and slowed our progress significantly.

Overall, our management has been supportive of HPC Experiment, but that support is predicated upon sign-off from the Legal department. Since cloud services is a bit of a "hot button" issue, especially since our project involves human DNA data that some might consider highly sensitive, the review process took more time than expected. Multiple individuals with expertise from procurement, information technology, and health information privacy, have all weighed in on the overall project and the pros and cons of performing this type of R&D on AWS and other cloud providers.

SeqWare is a comprehensive open source toolbox, billed as a "genome center in a box," so we expected a considerable learning curve. However, the toolbox is fairly well documented, and we had personal interaction with the principal developer, Brian O'Connor, so our experience overall was positive.

However, we encountered a few difficulties. The first step in using SeqWare is to ingest data into the system, which involves uploading files, and registering them within the metadata database as samples. The samples are then processed by workflows. One option is to use a web interface to the MetaDB system. However, we found the web user interface was slow and had some bugs that made it difficult to work with, so our main interaction with SeqWare was through the command line tools. Yet the command-line tools required invoking very long and complicated commands, which made it difficult to diagnose problems. Once we got some experience with the tools, though, the process was smooth.

To process the samples, SeqWare uses workflows developed in Java. When developing a workflow with the SeqWare Java library, we found that some aspects of working with large files during development need to be sorted out. For example, NGS workflows typically make use of large sequence files used as references. Using the SeqWare tools, it was not apparent how you would transition from a fast develop-compile-debug cycle in which the reference files were not continually re-transferred as part of the workflow packaging process, to a deployed workflow bundle that includes the correct reference file. Although SeqWare generates and uses comprehensive workflow bundles, which ensures a reproducible process, some facilities need to be implemented to reduce the burden on the developer in managing large files. Brian was very helpful, and using our public repository, rewrote and improved our workflow code.

After we had our samples registered, and a workflow developed and tested, we were ready to process our data. The publicly available SeqWare AMI is pre-set to run workflows directly on the EC2 instance without an external cluster, which is a great option for development and testing. However, we wanted to test the scalability of the system by processing the samples in parallel. For this, we utilized the Nimbus Informatics service. Our MetaDB instance and workflow were transferred to Nimbus, and using our account credentials, we were able to initiate workflow runs from the command line of our SeqWare EC2 instance. This configuration and the speed with which Nimbus set it up, underscored the flexibility and maturity of the tools, and was quite impressive. However, once we submitted our jobs via the command line, there was a considerable delay before any samples were actually processed, and the monitoring tools available from the command line (and the web UI) were not very helpful.

BENEFITS

Nimbus Informatics participated in the UberCloud HPC Experiment project to evaluate Nimbus' cloud-based genomics tools. Nimbus Informatics is a provider of hosted computational workflows that enable the analysis of complex, genomics data for customers that lack local HPC infrastructure. In the evaluation, Nimbus and the end-user collaborated on the creation of a sequence variant calling pipeline used to find genomic differences in samples sequenced by the end-user. Like other Nimbus projects, this work leveraged the extensive infrastructure available through the open source SeqWare project (http://seqware.io), which provides workflow and other tools to aid in genomics analysis.

The benefits of this solution are three-fold. First, through partnering with Nimbus, the end-user was able to write workflows using an open source infrastructure. This increased transparency of the engineered solution since both the end-user and Nimbus were able to use the core SeqWare software with complete transparency and few barriers to use.

Second, because of the open nature of the software, the end-users were able to create their own analytical workflows and test them locally. SeqWare, for example, provides virtual machines (VMs) that anyone, including the end-user, can download and use for their custom workflow development. This is a powerful model since most genomics cloud provides are closed source and do not support users creating their own custom workflows.

By providing an open and user-driven solution, Nimbus allowed the end-user to create an analytical workflow that used standard tools but was customized for the particulars of their analytical challenges. The final benefit of the Nimbus solution was the cloud-based hosting of the SeqWare system. While users can choose to build and run their own SeqWare infrastructure on their own hardware, Nimbus adds considerable value by providing the infrastructure on Amazon's cloud transparently to the user. In the case of the trial by the end-user, this meant they could write and test their workflow locally, upload it to the Nimbus infrastructure, and simply "order" workflows on specified samples. Nimbus handled the provisioning of SeqWare infrastructure behind the scenes, monitored workflows, ensured sufficient server capacity was available, and invoiced the end-user for the cloud time used, all without the need for the end-user to interact directly with the cloud. Together, these benefits provided a streamlined and efficient system capable of processing massive amounts of genomic data using an analysis workflow specific to the customer's needs.

CONCLUSIONS AND RECOMMENDATIONS

Since considerable effort was spent working through connectivity issues with IT, a good working relationship with that group, as well as familiarity with their processes, is essential. Recently, our IT department has formed an "R&D IT" sub-group to liaise between R&D associates and the wider IT organization including Networking and Information Security, on a project-specific basis.

These dedicated resources have made a large difference in the efficiency of the request process, and provided a high level of visibility to our projects within the IT management organization. However, our corporation's networking infrastructure is complicated, and there is a tension between security and functionality that will be a continual battle. Also, our group is competing for bandwidth with the rest of the R&D groups and business functions, which will limit our ability to utilize cloud resources going forward. One possibility we are investigating is to purchase our own dedicated bandwidth directly from our Internet vendor. This would give us ample bandwidth for our needs, and allow us to sidestep a lot of the security settings that are in place to protect sensitive corporate data.

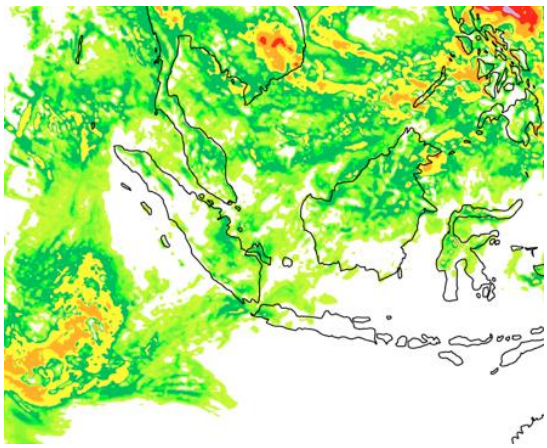
We determined that SeqWare/Nimbus is a comprehensive solution with an attractive open source business model, with promising prospects. We do think that if our group were to use it extensively, we would develop some wrappers around the command line tools to make it user-friendlier. Also, we would work with Nimbus to improve the monitoring capabilities, and push for a better web user interface to the metadata database.

In future rounds of HPC Experiment, we plan to work with Nimbus to explore the processing of large numbers of samples, and the storage, management and analysis of the resulting variant data in the SeqWare Query Engine (SQE). Since SQE is built on open source Hadoop and HBase database systems, there is a vibrant ecosystem of related projects that can be leveraged and integrated to support a wide range of use cases as our group's needs evolve.

Case Study Authors – Thomas Dyar, Senior Software Engineer, Betty Diegel, Senior Software Engineer, Brian O'Connor, CEO Nimbus Informatics

Team 65:

Weather Research and Forecasting: Performance and Evaluation of WRF



"The primary benefit of participating in the HPC Experiment is that we were able to evaluate the cluster compute instances in AWS."

MEET THE TEAM

End Users - The beneficiary of this research is Temasek Marine Sciences Institute, National University of Singapore. The experiments were conducted by students and faculty from Department of Electrical and Computer Engineering, NUS and the Institute for Infocomm Research, A*STAR, Singapore.

Software Providers - Open source WRF software from UCAR

Resource Provider - Amazon Web Services (AWS)

HPC Expert - Prof. Bharadwaj Veeravali, Department of Electrical and Computer Engineering, NUS and Institute for Infocomm Research, A*STAR, Singapore.

USE CASE

In this experiment, we attempted to evaluate the performance of WRF (Weather Research and Forecasting) open source software on cloud services provided by Amazon Web Services. The WRF software is currently setup to run on a Beowulf-style computing cluster comprised of 12 nodes, each node featuring an 8-core CPU. The cluster is used 24x7. Execution time is 24 hours for a 12 hour weather prediction cycle.

Experiments – We used an 8 node cluster of AWS CC1.4xlarge instances located within a placement group.

CPUs:	2 x Intel Xeon 5570
RAM:	23 GB
Memory Topology:	Socket L#0 + L3 L#0 (8192KB) L2 L#0 (256KB) + L1 L#0 (32KB) + Core L#0 PU L#0 (P#0) PU L#1 (P#8)

Memory Topology, cont'd:	L2 L#1 (256KB) + L1 L#1 (32KB) + Core L#1 PU L#2 (P#1) PU L#3 (P#9) L2 L#2 (256KB) + L1 L#2 (32KB) + Core L#2 PU L#4 (P#2) PU L#5 (P#10) L2 L#3 (256KB) + L1 L#3 (32KB) + Core L#3 PU L#6 (P#3) PU L#7 (P#11) Socket L#1 + L3 L#1 (8192KB) L2 L#4 (256KB) + L1 L#4 (32KB) + Core L#4 PU L#8 (P#4) PU L#9 (P#12) L2 L#5 (256KB) + L1 L#5 (32KB) + Core L#5 PU L#10 (P#5) PU L#11 (P#13) L2 L#6 (256KB) + L1 L#6 (32KB) + Core L#6 PU L#12 (P#6) PU L#13 (P#14) L2 L#7 (256KB) + L1 L#7 (32KB) + Core L#7 PU L#14 (P#7) PU L#15 (P#15) HostBridge L#0 PCI 8086:7010 Block L#0 "sda" PCI 1013:00b8

Table 1: Details of AWS CC1.4xlarge instances.

End-to-end process – Since the goal of this experiment was to analyze the performance of WRF on different configurations of a cloud based computer cluster, we set up an 8 node CC1.4xlarge compute cluster within a placement group at AWS. The installation of WRF was fairly straightforward as it is open source and relatively well documented. We decided to use a MPICH2 cluster configuration to run the jobs in parallel.

The experiments were setup to run up to 16 processes on a single instance and run up to 8 parallel instances/nodes yielding a total of 128 parallel processes. Afterword, the experiments were repeated by using the CPU core pinning option of the hydra MPI manager. Both the best case of all physical cores and worst case of physical/virtual core pairs were documented.

Results - We extracted the performance characteristics of executing WRF on varying number of cluster sizes - from 1 instance to 8 instances, and varied the number of threads/processes in each instance. In general, we observed that the same number of threads running on greater number of nodes had better performance than running on fewer nodes. Figures 2 - 5 show the varying performance of multiple threads in 1, 2, 4 and 8 instances.

The trend of each chart has been consistent whereby threads that run on physical/virtual cores have highest run time regardless of number of instance used. Besides, the practice of core pinning does achieve better performance in comparison to default allocation as the number of thread increases. Further increase of number of threads beyond 32 threads does not reduce the runtimes of WRF.

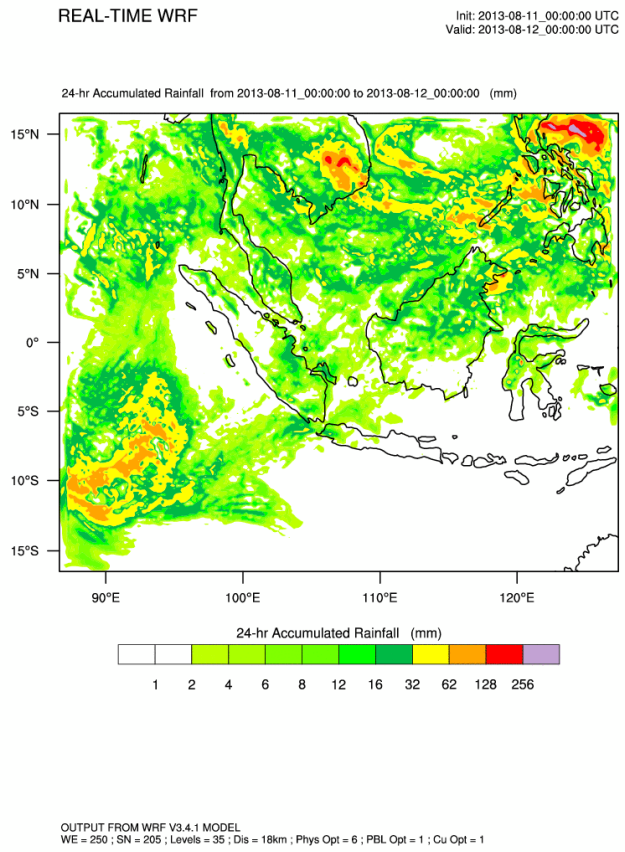


Figure 1: WRF simulation showing 24-hr accumulated rainfall for the Singapore region.

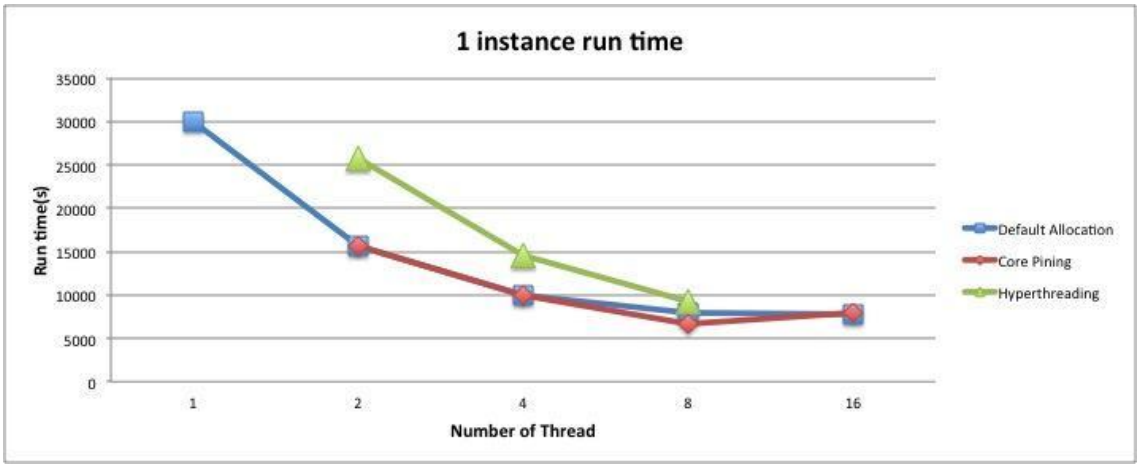


Figure 2: WRF runtimes Vs threads in 1 instance cluster using multiple pinning options.

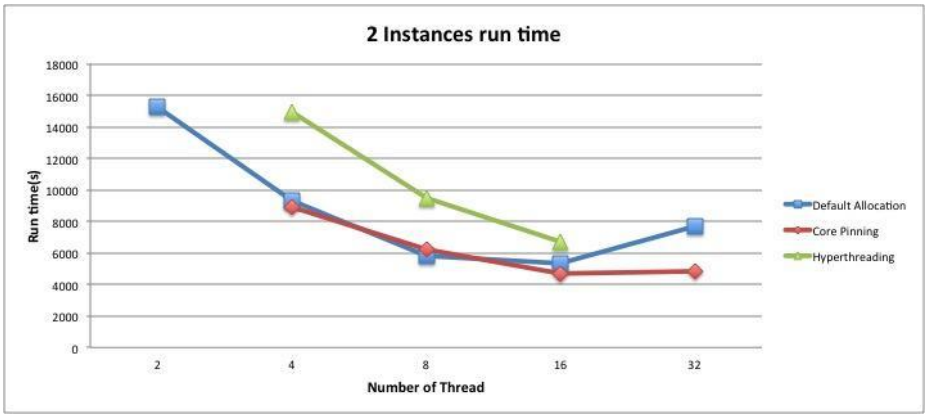


Figure 3: WRF runtimes Vs threads in 2 instance cluster using multiple pinning options.

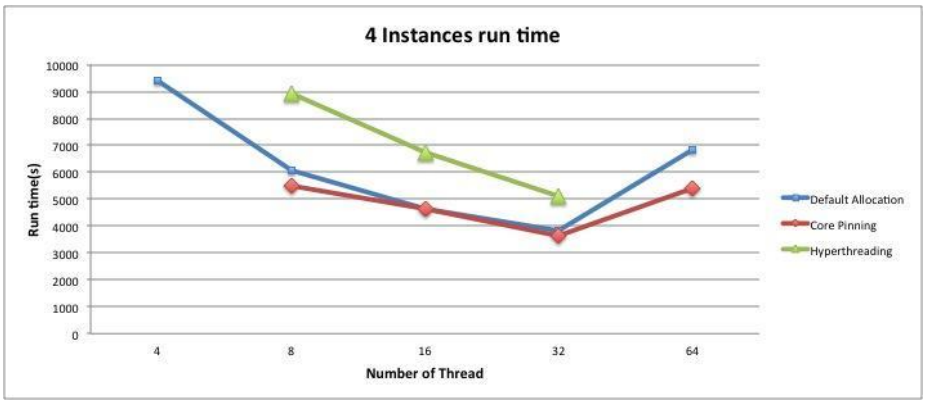


Figure 4: WRF runtimes Vs threads in 4 instance cluster using multiple pinning options.

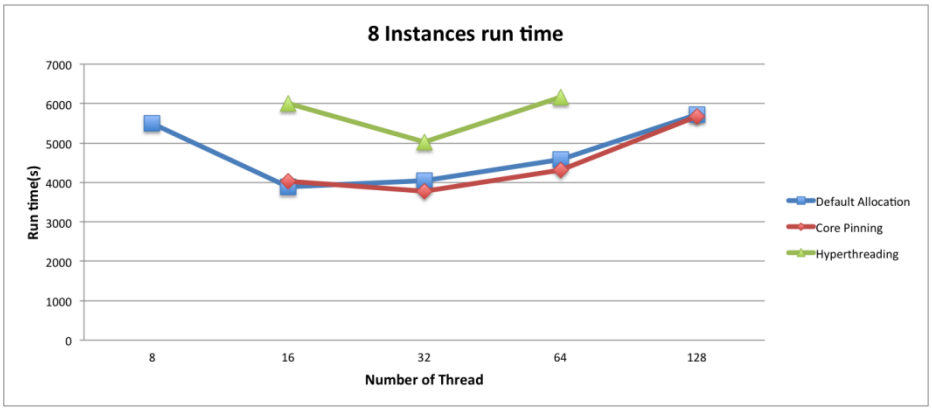


Figure 5: WRF runtimes Vs threads in 8 instance cluster using multiple pinning options.

CHALLENGES

In order to save costs, we implemented a scheme of shutting down the instances when they finished one set of experiments. On restarting the instances, the IP addresses assigned to them were changed by the resource provider. This meant that the cluster had to be reconfigured frequently.

More importantly, this meant that the number of network hops was not a constant. Specifically, if two nodes are in the same network subnet then it is 2 hops (node - to - network switch - to - node). If the same two nodes, on reboot, end up on two different network subnets then it means 4 hops (node - to - network switch - to - router - to 2nd network switch - to node). Figure 6 shows these IP

variations. It indicates that there is no pattern in the allocations although all of the nodes are within the same placement group in the AWS computer cluster.

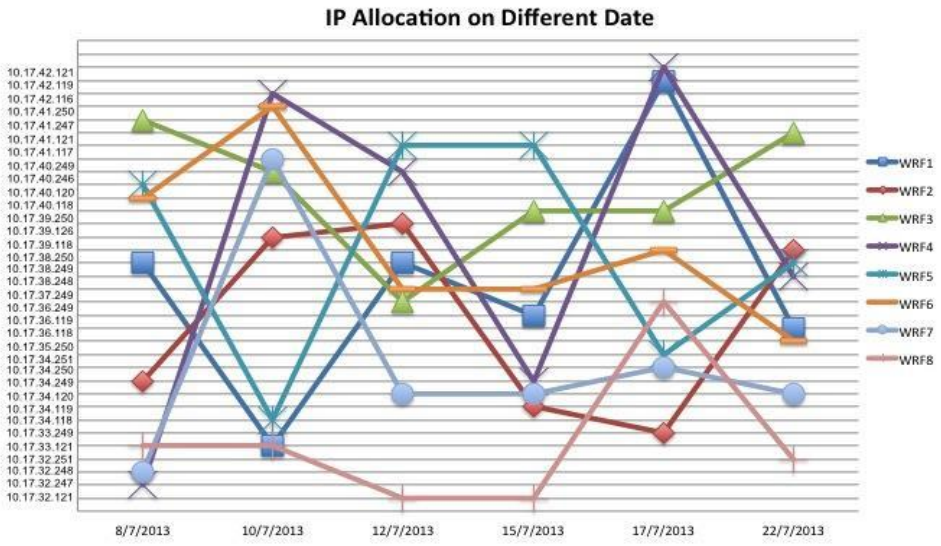


Figure 6: IP assignment variations on instance reboots.

BENEFITS

We were able to evaluate the cluster compute instances in AWS. Technically, the main benefit was to determine the optimum number of instances to use for WRF for a given data set size. Specifically, given a sufficiently high bandwidth, was there is a performance improvement when increasing the number of nodes across which the software was executed? Also, we learned that the ease with which the cluster size on the cloud can be expanded is remarkable, and is of immense use for dealing with performance bottlenecks.

CONCLUSION AND RECOMMENDATIONS

Core pinning is essential because the probability of running threads on virtual cores is higher when the number of threads used is high. We found that running WRF in hyper-threading results in longer run times – therefore, it is undesirable to have virtual cores running WRF. Also, providing additional threads does not speed up compute time. Instead it increases communication latency between the additional cores because the decomposition of the WRF workload is so small that all the threads are running at low utilization.

REFERENCES

- WRF Model - How to Compile: http://www.mmm.ucar.edu/wrf/users/wrfv2/install.html
- Getting Started with Amazon EC2 Linux Instances: *Amazon Elastic Compute Cloud*. http://docs.aws.amazon.com:AWSEC2:latest:UserGuide:EC2_GetStarted.html
- Using the Hydra Processing Manager:
- http://wiki.mpich.org/mpich/index.php/Using_the_Hydra_Process_Manager
- ARSC WRF Benchmarking Suite: http://weather.arsc.edu/BenchmarkSuite/

Case Study Author – Prof. Bharadwaj Veeravali

Team 25
Simulation of Spatial Hearing

"Our main challenge was to develop interactive visualization tools for simulation data that was stored in the cloud."

MEET THE TEAM

End User
The end user is a manufacturer of consumer products. The end-user tasks were related to the planning of the simulations and post-processing the simulated data.

Software Provider and HPC Expert – Antti Vanne, Kimmo Tuppurainen, Tomi Huttunen
These team members are with Kuava Ltd. Kuava provides services for computational technology and simulations. Kuava's software products are Waveller Cloud platform for running and visualizing simulations in the cloud; and Datain, a tool for data acquisition, analysis and storage.

HPC Experts – Ville Pulkki, Marko Hiipakka
Pulkki and Hiipakka are researchers from Aalto University. They provided technical expertise for acoustic analysis in the post-processing of the simulation results.

USE CASE

A sound emitted by an audio device is perceived by the user of the device. The human perception of sound is, however, a personal experience. For example, the spatial hearing (the capability to distinguish the direction of sound) depends on the individual shape of the torso, head and pinna (i.e. so-called head-related transfer function, HRTF).

To produce directional sounds via headphones, one needs to use HRTF filters that "model" sound propagation in the vicinity of the ear. These filters can be generated using computer simulations, but, to date, the computational challenges of simulating the HRTFs have been enormous due to: the need of a detailed geometry of head and torso; the large number of frequency steps needed to cover the audible frequency range; and the need of a dense set

of observation points to cover the full 3D space surrounding the listener. In this project, we investigated the fast generation of HRTFs using simulations in the cloud. The simulation method relied on an extremely fast boundary element solver, which is scalable to a large number of CPUs.

The process for developing filters for 3D audio is long, but the simulation work of this study constitutes a crucial part of the development chain. In the first phase, a sufficient number of the 3D head-and-torso geometries needed to be generated. A laser-scanned geometry of a commercially available test dummy was used in these simulations. Next, acoustic simulations to characterize acoustic field surrounding the head-and-torso were performed. This was our task in the HPC experiment. Finally, the filters were generated from the simulated data and they were evaluated by a listening test. The final part will be done by Aalto University and the end-user after the data from the HPC experiment is available.

The Environment
Simulations were run via Kuava's Waveller Cloud simulation tool using the system described below. The number of concurrent instances ranged between 6 and 20.

- Service: Amazon Elastic Compute Cloud. Total CPU hour usage: 341h. Type: High-CPU Extra Large Instance.

- High-CPU Extra Large Instance: 7 GiB of memory. 20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each). 1690 GB of instance storage.

- One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor. This is also the equivalent to an early-2006 1.7 GHz Xeon processor.

CHALLENGE

Our main challenge was to develop interactive visualization tools for simulation data that was stored in the cloud.

BENEFITS

The main benefit was realized from the flexible resource allocation that is necessary for efficient acoustic simulations – that is, a large number of instances can be obtained for a short period of time.

There was no need to invest in our own computing capacity. Especially in audio simulations, the capacity is needed in short bursts for fast simulation turnaround times and the time between the simulation bursts while the next simulation is planned. The fact that no computational capacity is needed is significant.

CONCLUSIONS AND RECOMMENDATIONS

The main lessons learned were related to the optimal use of the cloud capacity. In particular, we obtained important experience on running large simulations in the cloud. For example, the optimal number of instances was dependent on the size of the simulation task and the amount of data needed to transfer to and from the cloud.

The man hours logged during the experiment were: Kuava (60h), Aalto (1h) and end-user (4h). Total CPU hour usage during the experiment was 341h using High-CPU Extra Large Instance.

Team 20

NPB2.4 Benchmarks and Turbo-machinery Application on Amazon EC2

"We obtained first-hand experience in running engineering applications in the commercial cloud computing environment."

MEET THE TEAM

End User and Software Provider
The team consisted of Haibao Chen, Ph.D. Candidate, Zhenjiang Xie, Master Student, and Song Wu, Professor, from the Services Computing Technology and System Lab & Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology in China.

Resource Provider – Amazon AWS with EC2

HPC Expert – Wenguang Chen
Professor Chen is with the Department of Computer Science and Technology, Tsinghua University in China.

USE CASE

We used an application from machinery manufacturing in this experiment. The engineers who use CAE software on physical workstations or compute clusters can do the same operations on cloud resources based on their skills and knowledge of creating, configuring and connecting to instances on Amazon EC2.

Advantages and Challenges of Complex Engineering Applications in Clouds
The rapid development and popularity of cloud computing are profoundly affecting and changing the resource supply, resource management, and computing modes of future applications. As one of the main supporting technologies of cloud computing, virtualization technology enables cloud computing to have the features of dynamical scalability, flexible customization and isolation of the environment, and transparent fault tolerance of applications, which are missing in the traditional platform of engineering applications. Therefore, cloud computing provides a good choice to solve

engineering applications demand, but also brings new opportunities for the development of engineering applications and software.

Compared to traditional computing environments, two main advantages of cloud computing for engineering applications are a *customizable environment,* and *flexible usage and management of resources*.

However, the majority of current cloud systems and the corresponding techniques primarily aim at Internet-based applications. Engineering applications, especially complex engineering applications, bring grand challenges to cloud computing since they are significantly different from those service-oriented Internet-based applications due to their inherent features, such as workload variations, process control, resource requirements, environment configurations, lifecycle management, and reliability maintenance.

The End-to-End Process

1. **Define end-user project with help from end-user –** During the definition of our project, we consulted with researchers from School of Mechanical Science and Engineering, Huazhong University of Science and Technology, and then decided to choose the NAS Parallel Benchmarks (NPB) and a real engineering application as the two experimental subjects.
2. **Contact resource providers, set up project environment –** Amazon EC2 was our resource provider. After obtaining an **"**Amazon EC2 redeem code" voucher from the project organizers, we redeemed the resource immediately, then launch and login an instance, which was made into an AMI file later. This set the experiment environment with openmpi-1.4.3, which is the commonly used MPI Communication Library, NAS Parallel Benchmarks (NPB2.4), and industry-renowned CFD software. We use scripts to build virtual clusters with the AMI file created above as the experiment environment.

Three types of EC2 instances were used in our experiment, namely EC2 Cluster Compute Instances (cc1.4xlarge), EC2 High CPU Extra Large Instances (c1.xlarge) and EC2 Extra Large Instances (m1.large). Each CCI has 8 cores (the computing capability is equal to 33.5 EC2 Compute Units), with 23GB of memory. Each High CPU Extra Large Instances has 8 cores (the computing capability is equal to 20 EC2 Compute Units), with 7GB of memory. Each Extra Large Instances has 4 cores (the computing capability is equal to 8 EC2 Compute Units), with 23GB of memory.

We built virtual clusters consisting of 9 or 17 nodes to run the NPB programs. One node of the virtual cluster is the NFS Server, which is an EC2 Extra Large Instance; the other nodes are compute nodes, which run EC2 Cluster Compute Instances or EC2 High CPU Extra Large Instances. The detailed setup of NPB experiment is depicted in Table 1.

Number of Process	Compute Node Instance Type	Number of Compute Node	Number of NFS node
64	EC2 Extra Large Instance (m1.large)	16	1
64	EC2 Cluster Compute Instance (cc1.4xlarge)	8	1
128	EC2 High CPU Extra Large Instance (c1.xlarge)	16	1

Table 1: Detailed Setup of the NPB Experiment.

3. **Initiate execution of the end-user project –** After setting the experiment environment, we first carried out both class C and Class D of the NPB benchmarks on three virtual clusters consisting of 16 m1.large instances (64 cores), 16 c1.xlarge instances (128 cores) and 8 cc1.4xlarge (64 cores) instances separately. We ran each benchmark 10 times, and the experimental results are shown in the appendix.

Since we used Amazon EC2 for the first time, and had the expectation of cheap prices, we did the test in the same way that we would have done on the HPC cluster in our laboratory. This approach incurred significant costs at this stage of the experiment – so, it is not cheap, or not as cheap as we originally thought.

4. **Monitoring –** We did not use specialized monitoring tools, but instead used the Xschell4, which is a terminal emulator for Windows platforms, to connect to virtual cluster consisting of EC2 instances, and to monitor the progress of our experiment. We used the runtime of the benchmark as the metric for evaluating the performance of the cloud resource.

5. **Review results (where needed**) – Experimental results show that there is substantially no performance fluctuation of tightly coupled benchmarks on the virtual cluster consisting of CCI instances. By contrast, the performance fluctuation of tightly coupled benchmarks on the virtual cluster consisting of non-CCI instances is obvious, and we find that the application performance on the virtual cluster consisting of non-CCI instances shows an increasingly positive trend from the first round to the tenth round.

CHALLENGES

- An objective fact is that China's Internet speed is still lagging behind comparable network speeds worldwide.
- Currently, Chinese users cannot connect to the Amazon EC2 instances. In order to do the scientific experiments, we access EC2 instances temporarily through some technical means.
- During uploading large files to EC2 instances by Xschell4, we often encountered the problem of instance crash. Our solution was to first upload files to DropBox, which is a free service that lets you bring your photos, docs, and videos anywhere and share them easily, and then download them in the EC2 instances.
- We also had budget issues. Running the NPB test excessively on virtual clusters consisting of non-CCI instances at the first step of the experiment led to a budget overrun. The project organizers have solved this issue.

BENEFITS

We are very grateful to the organizer for the opportunity of participating in the project – we obtained first-hand experience in running engineering applications in the commercial cloud computing environment.

We got a more profound understanding of the EC2 billing methods through the experiment.

We verified two main advantages of cloud computing for engineering applications, which are a *customizable environment,* and the *flexible usage and management of resources*.

We found that the Standard Deviation of the benchmarks' performance on the virtual cluster consisting of CCI instances is small. By contrast, the performance fluctuation of benchmarks on the virtual cluster consisting of non-CCI instances is obvious.

We observed that the performance of the real application on a virtual cluster consisting of non-CCI instances showed an increasingly positive trend from the first to the last round.

CONCLUSIONS AND RECOMMENDATIONS

Do not use commercial cloud resources in the same way that you use them in your own physical cluster, because each operation in commercial cloud resources may incur costs.

Although you don't need upfront investment on infrastructure and commercial software in the cloud environment, you still need to pay for the hardware and software you have consumed. That is to say, you need comprehensive schedules regarding using the cloud resource in order to adapt to the billing method before running engineering applications in the cloud.

Although the price of non-CCI instance is cheaper, manufacturers need a precise model to choose the suitable type of instance according to their budget and the deadline of tasks.

Manufacturers need solutions, which are the basis of the comprehensive schedules mentioned above, to predict the running times of engineering application when the physical environment turns into a cloud environment.

Relative to the general engineering applications, we are more concerned about performance issues of complex engineering applications in the cloud computing environment, because it is more challenging! The key issues are:

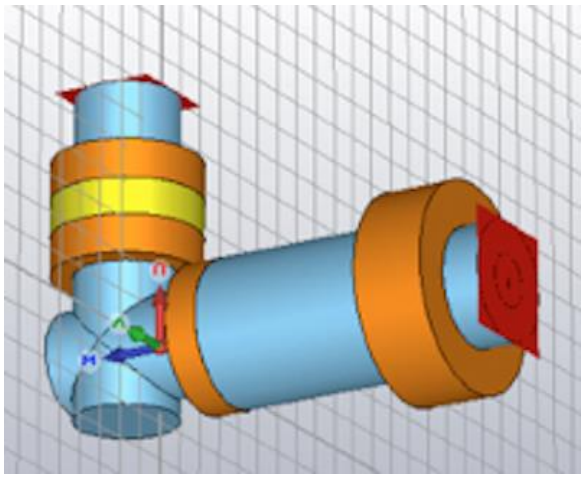
- How to build a new cloud resource organization model for the characteristics of complex engineering applications

- How to design virtualized resource management techniques for complex engineering applications in the cloud environment

- How to schedule the cloud resources in order to enhance complex engineering applications performance and cloud system capacity

Team 2

Simulation of a Multi-resonant Antenna System Using CST Microwave Studio



"The cloud is normally advertised as "enabling agility" and "enabling elasticity" but in several cases it was our own project team that was required to be agile/nimble simply to react to the rapid rate of change within the AWS environment."

MEET THE TEAM

End User – Dr. Nicolas Freytag
Freytag is a PhD physicist and engineer working for Bruker Biospin Corporation as innovation manager responsible for new sensor applications, new markets and fundamental research. Bruker is one of the world's leading analytical instrumentation companies with 6,000 employees at more than 70 locations around the globe. The local site has 500 employees.

Software Provider – Dr. Felix Wolfheimer
Wolfheimer is a Senior Application Engineer with CST AG

Resource Provider – Amazon Web Services

HPC Expert – Chris Dagdigian
Dagdigian is a Principal Consultant employed by the BioTeam.

USE CASE

The end user uses CAE for virtual prototyping and design optimization on sensors and antenna systems used in NMR spectrometers. Advances in hardware and software have enabled the end-user to simulate the complete RF-portion of the involved antenna system. Simulation of the full system is still computationally intensive although there are parallelization and scale-out techniques that can be applied depending on the particular "solver" method being used in the simulation.

The end-user has a highly-tuned and over-clocked local HPC cluster. Benchmarks suggest that for certain "solvers" the local HPC cluster nodes are roughly 2x faster than the largest of the cloud-based Amazon Web Services resources used for this experiment. However, the local HPC cluster

averages 70% utilization at all times and the larger research-oriented simulations the end-user was interested in could not be run during normal business hours without impacting production engineering efforts.

Remote cloud-based HPC resources offered the end-user the ability to “burst” out of the local HPC system and onto the cloud. This was facilitated both by the architecture of the commercial CAE software as well as the parallelizable nature of many of the “solver” methods.

The CST software offers multiple methods to accelerate simulation runs. On the node level (single machine) multithreading and GPGPU computing (for a subset of all available solvers) can be used to accelerate simulations still small enough to be handled by a single machine. If a simulation project needs multiple independent simulation runs (e.g. in a parameter sweep or for the calculation of different frequency points) that are independent of each other, these simulations can be sent to different machines to execute in parallel. This is done by the CST Distributed Computing System, which takes care of all data transfer operations necessary to perform this parallel execution. In addition, very large models can be handled by MPI parallelization using a domain decomposition approach.

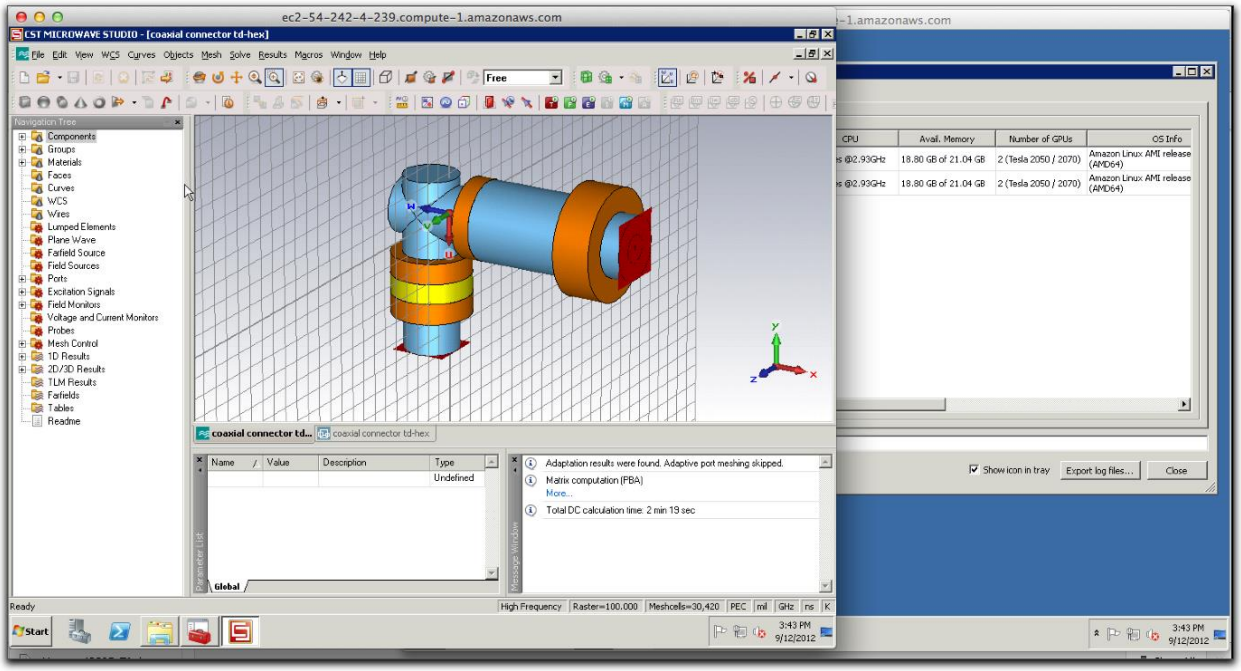
End-user effort: >25h for setup, problems and benchmarking. >100h for software related issues due to large simulation projects, bugs, and post-processing issues that would also have occurred for purely local work.

ISV effort: ~2-3 working days for creating license files, assembling documentation, following discussions, debugging problems with models in the setup, debugging problems with hardware resources.

PROCESS

1. Define the ideal end-user experiment
2. Initial contacts with software provider (CST) and resource provider (AWS)
3. Solicit feedback from software provider on recommended “cloud bursting” methods; secure licenses
4. Propose Hybrid Windows/Linux Cloud Architecture #1 (EU based)
5. Abandon Cloud Architecture #1; User prefers to keep simulation input data within EU-protected regions. However, AWS has resources we require that did not yet exist in EU AWS regions. End-user modifies experiment to use synthetic simulation data, which enables the use of US, based cloud systems.
6. Propose Hybrid Windows/Linux Cloud Architecture #2 (US based) & implement at small scale for testing
7. Abandon Cloud Architecture #2. Heavily secured virtual private cloud (VPC) resource segregation front-ended by an internet-accessible VPN gateway looked good on paper however AWS did not have GPU nodes (or the large cc2.* instance types) within VPC at the time and the commercial CAE software had functionality issues when forced to deal with NAT translation via a VPN gateway server.
8. Propose Hybrid Windows/Linux Cloud Architecture #3 & implement at small scale for testing.
9. The third design pattern works well; user begins to scale up simulation size

10. Amazon announces support for GPU nodes in EU region and GPU nodes within VPC environments; end-user is also becoming more familiar with AWS and begins experimenting with Amazon Spot Market to reduce hourly operating costs by very significant amount.
11. Hybrid Windows/Linux Cloud Architecture #3 is slightly modified. The License Server remains in the U.S. because moving the server would have required a new license file from the software provider. However, all solver and simulation systems are relocated to Amazon EU region in Ireland for performance reasons. End-user switches all simulation work to inexpensively sourced nodes from the Amazon Spot Market.
12. The "Modified Design #3" in which solver/simulation systems are running on AWS Spot Market Instances in Ireland, while a small license server remaining in the U.S. reflects the final "design." As far as we understood, the VPN-Solution that did not work in the beginning of the project would actually have worked at the end of the project period because of changes within the AWS. In addition, the preferred "heavily secured" solution would have provided fixed MAC addresses, thus avoiding having to run a license instance all the time.



Front-end and two GPU solvers in action

CHALLENGES

Geographic constraints on data – End-user had real simulation and design data that should not leave the EU.

Unequal availability of AWS resources between Regions – At the start of the experiment, some of the preferred EC2 instance types (including GPU nodes) were not yet available in the EU region (Ireland). This disparity was fixed by Amazon during the course of the experiment. At the end of the experiment, we had migrated the majority of our simulation systems back to Ireland.

Performance of Remote Desktop Protocol – The CAE software used in this experiment makes use of Microsoft Windows for experiment design, submission and visualization. Using RDP to access remote

Windows systems was very difficult for the end-user, especially when the Windows systems were operating in the U.S.

CAE Software and Network Address Translation (NAT) – The simulation software assumes direct connections between participating client, solver and front-end systems. The cloud architecture was redesigned so that essential systems were no longer isolated within secured VPC network zones.

Bandwidth between Linux solvers & Windows Front-End – The technical requirements of the CAE software allow for the Windows components to be run on relatively small AWS instance types. However, when large simulations are underway a tremendous volume of data flows between the Windows system and the Linux solver nodes. This was a significant performance bottleneck throughout the experiment. The project team ended up running Windows on much larger AWS instance types to gain access to 10GbE network connectivity options.

Node-locked software licenses – The CAE software license breaks if the license server node changes its network hardware (MAC address). The project team ended up leveraging multiple AWS services (VPC, ENI, ElasticIP) in order to operate a persistent, reliable license serving framework. We had to leave the license server in the US and let it run 24/7 because it would have lost the MAC-address upon reboot. Only in the first setup did it have a fixed MAC and IP.

Spanning Amazon Regions – It is easy in theory to talk about cloud architectures that span multiple geographic regions. It is much harder to implement this "for real." Our HPC resources switched between US and EU-based Amazon facilities several times during the lifespan of the project. Our project required the creation, management and maintenance of multiple EU and US specific SSH keys, server images (AMIs) and EBS disk volumes. Managing and maintaining capability to operate in the EU or US (or both) required significant effort and investment.

BENEFITS

End-User

- Confirmation that a full system simulation is indeed possible even though there are heavy constraints, mostly due to the CAE software. Model setup, meshing and post-processing are not optimal and require huge efforts in terms of manpower and CPU-time.
- Confirmation that a full system simulation can reproduce certain problems occurring in real devices and can help to solve those issues.
- Realize the reasonable financial investment for additional computation resources needed for cloud bursting approaches.
- Realize that the internet connection speed was the major bottleneck for a cloud bursting approach but also very limiting for RDP work.

Software Provider

- Confirmation that the software is able to be setup and run within a cloud environment and also, in principle, using a cloud bursting approach (see comments regarding the network speed). Some very valuable knowledge was gained on how to setup an "elastic cluster" in the cloud using best practices regarding security, stability and price in the Amazon EC 2 environment.
- Experience the limitations and pitfalls specific to the Amazon EC2 configuration (e.g. availability of resources in different areas, VPC needed to preserve MAC addresses for licensing setup, network speed, etc.).

- Experiencing the restrictions of the IT department of a company when it comes to the integration of cloud resources (specific to the cloud bursting approach).

HPC Expert

- Chance to use Windows-based HPC systems on the cloud in a significant way was very helpful
- New appreciation for the difficulties in spanning US/EU regions within Amazon Web Services

CONCLUSIONS AND RECOMMENDATIONS

End-User

- Internet transfer speed is the major bottleneck for serious integration of cloud computing resources to the end users design flow and local HPC systems.
- Internet transfer speed is also a limiting factor to allow for remote visualization.
- Security and data protection issues as well as fears of the end users IT department create a huge administrative limitation for the integration of cloud based resources.
- Confirmation that a 10 GbE network can considerably speed up certain simulation tasks compared to the local clusters GbE network. The local cluster has been upgraded in the meantime to an IB network.

HPC Expert

- Rapid evolvement of our provider's capability constantly forced the project team to re-architect the HPC system design. The cloud is normally advertised as "enabling agility" and "enabling elasticity" but in several cases it was our own project team that was required to be agile/nimble simply to react to the rapid rate of change within the AWS environment.
- The AWS Spot Market has huge potential for HPC on the cloud. The price difference is extremely compelling and the relative stability of spot prices over time makes HPC usage worth pursuing.
- Our design pattern for the commercial license server is potentially a useful best-practice. By leveraging custom/persistent MAC addresses via the use of Elastic Network Interfaces (ENI) within Amazon VPC we were able to build a license server that would not "break" should the underlying hardware characteristics change (common on the cloud).
- In a "real world" effort we would not have made as much use of the hourly on-demand server instance types. Outside of this experiment it is clear that a mixture of AWS Reserved Instances (license server, Windows front-end, etc.) and AWS Spot Market instances (solvers and compute nodes) would deliver the most power at the lowest cost.
- In a "real world" effort we would not have done all of our software installation, configuration management and patching by hand. These tasks would have been automated and orchestrated by a proper cloud-aware configuration management system such as Opscode Chef.

Software Provider:

- The setup of a working setup in the cloud is quite complex and needs quite some IT/Amazon EC2 expertise. Supporting such a setup can be quite challenging for an ISV as well as for an end user. Tools to provide simplified access to EC2 would be helpful.