A HYBRID COMPUTING INFRASTRUCTURE FOR CLIMATE SIMULATION

by

Kai Liu
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Earth Systems and Geoinformation Sciences

Committee:

______________________	Dr. Chaowei Yang, Dissertation Director
______________________	Dr. Ruixin Yang, Committee Member
______________________	Dr. Songqing Chen, Committee Member
______________________	Dr. Matthew T. Rice, Committee Member
______________________	Dr. Anthony Stefanidis, Department Chairperson
______________________	Dr. Donna M. Fox, Associate Dean, Office of Student Affairs & Special Programs, College of Science
______________________	Dr. Peggy Agouris, Dean, College of Science
Date: ______________________	Summer Semester 2017 George Mason University Fairfax, VA

A Hybrid Computing Infrastructure for Climate Simulation

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Kai Liu
Master of Science
George Mason University, 2014
Bachelor of Science
Wuhan University, 2006

Director: Chaowei Yang, Professor
Department of Geography and Geoinformation Science

Summer Semester 2017
George Mason University
Fairfax, VA

DEDICATION

This dissertation is dedicated to my wife Huifen Wang and my son Thomas Liu

And my parents – Tongfang Liu, Wangxian Yu

ACKNOWLEDGEMENTS

First, I would like to express the deepest appreciation to my adviser and committee chair, Prof. Chaowei Yang, for his great mentorship, excellence guidance and enormous support during my graduate studies at George Mason University. I would also like to thank my committee members, Dr. Ruixin Yang, Dr. Songqing Chen and Dr. Matthew T. Rice, for their expertise and time in advising and encouraging throughout the entire dissertation process. In addition, I would like to thank Steve McClure to improve my English. Finally, I would like to thank my colleagues and friends at GGS department and Spatiotemporal Center: Zhenlong Li, Qunying Huang, Jing Li, Wenwen Li, Yunfeng Jiang, Jizhe Xia, Min Sun, Weiwei Song and others.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

Atmospheric Model Intercomparison Project	AMIP
Assessment Report5	AR5
Amazon Web Service	AWS
Berkeley Open Infrastructure for Network Computing	BOINC
Control Data Corporation	CDC
Coupled Model Intercomparison Project5	CMIP5
Data as a Service	DaaS
Floating-point Operations Per Second	FLOPS
GNU Compiler Collection	GCC
Global Earth Observation System of Systems	GEOSS
Goddard Institute for Space Studies	GISS
Graphic User Interface	GUI
Infrastructure as a Service	IaaS
Intergovernmental Panel on Climate Change	IPCC
National Aeronautics and Space Administration	NASA
National Oceanic and Atmospheric Administration	NOAA
New Credit System	NCS
Operating System	OS
Platform as a Service	PaaS
Periodically Upload Mechanism	PUM
Software as a Service	SaaS
UK Met Office Unified Model	UM
United Nations Environment Programme	UNEP
Virtual Disk Image	VDI
Volunteer Computing	VC
Virtual Machine	VM
Virtual Machine Monitor	VMM
World Meteorological Organization	WMO

ABSTRACT

A HYBRID COMPUTING INFRASTRUCTURE FOR CLIMATE SIMULATION

Kai Liu, Ph.D.

George Mason University, 2017

Dissertation Director: Dr. Chaowei Yang

The climatological community relies increasingly on computing intensive models and applications to study atmospheric chemistry, aerosols, carbon cycle and other tracer gases. These models and applications are becoming increasingly complex and bring computing challenges including: 1) the enormous computational power required for running these models and applications to produce results in a reasonable timeframe; 2) the challenging in providing convenient and fast solutions distributing and storing the massive climate model outputs; 3) the lack of methods for visualizing the climate simulation results efficiently and reliably. Volunteer computing provides a potential solution for tackling the computational power problem by obtaining large amounts of computational resources from global volunteers. Meanwhile, virtualization technology allows researchers to run climate models in a predefined virtual machine. Cloud computing storage provides advantages for distributing and storing climate data and outputs with low-cost. The Load Balancer and Auto Scaling in cloud computing provides a good solution to visualize the climate simulation results. This dissertation reports our research on integrating and optimizing volunteer computing, virtualization technology and cloud computing for climate simulation by: 1) using volunteer computing resources to leverage large number of home computers to support climate simulations; using virtualization technology to enable the climate models run on heterogeneous computers while providing bit-level homogeneous computing environment; optimizing the output collection mechanism to periodically upload climate model output; and optimizing the credit system to grant credits periodically to volunteers for volunteer retention to support long time climate simulation tasks. 2) Using cloud Simple Storage Service provided by leading cloud providers to develop a global replication storage to distribute cloud models and data to global volunteers. 3) Using Load Balancing to distribute incoming WMS requests across multiple cloud instances to improve the performance; Using Auto Scaling to help to maintain climate visualization availability and allows climate scientists to dynamically scale the cloud capacity. A prototype is developed to demonstrate the feasibility and efficiency of proposed techniques. The prototype is further tested in the Climate@Home project, a hybrid computing project using volunteer computing and cloud computing. Result shows that this research provides a computationally efficient and usable approach to accelerate climate simulation.

CHAPTER ONE: INTRODUCTION

Climate has changed during earth's history over millions of years and the change of climate has been one of the biggest threats for our earth. It is very important for humans to study and analyze climate change since it has significant impacts on ecosystems, agriculture and economic sectors. Global warming is one of the biggest climate change issues. Since the early 20th century, there is increasing mean temperature in various regions. Researchers analyze the Monthly mean maximum and minimum temperatures for over 50% of the Northern and 10% of the Southern Hemisphere landmass which is about 37% of the global landmass. It indicates that the rise of the minimum temperature has occurred at a rate three times that of the maximum temperature in that landmass during the period 1951–90 (0.84°C versus 0.28°C) (Karl et al. 1993). To analyze the climate change, engineers and climatologists have been working together to collect massive climate data. Particularly, climate data sets began growing rapidly after the first weather satellite TIROS-1 has been launched. Scientists use climate models to analyze these data and simulate the interactions of the atmosphere, oceans, land surface, and ice. Since NOAA's Geophysical Fluid Dynamics Laboratory in Princeton developed the first general circulation climate model which combined both oceanic and atmospheric processes in the late 1960s[1], a variety of models have been created and improved in the

[1] https://celebrating200years.noaa.gov/breakthroughs/climate_model/

recent 50 years. For example, Energy Balance Models (EBMs) calculate a balance between the radiation arriving at the Earth and the energy leaving the Earth. Radiative Convective Models (RCMs) are based on the radiative and convective energy transport up to the atmosphere. General Circulation Models (GCMs) include the physics of the atmosphere, land, see and ice (Sellers et al. 1986, Liang et al. 1994). These models vary in complexity to answer the questions like: how the hurricanes interact with oceans? How the global warming change the polar glaciers? What are the future sea level changes in 100 years?

In recognition of the problem of global warming and other climate change issues, the Intergovernmental Panel on Climate Change (IPCC) has established in 1988 by the World Meteorological Organization (WMO) and United Nations Environment Programme (UNEP) to review and assess these climate models and related reports, papers and other literatures 1. It has published five comprehensive assessment reports and the latest assessment report Assessment Report5 (AR5) is based on the fifth phase of the Coupled Model Intercomparison Project5 (CMIP5), which incorporates the latest versions of climate models for long-term simulations (Taylor et al., 2012). Compared to the previous assessment reports, the models and scenarios have been changed to make a comparison with earlier literature challenging (Rogelj et al., 2012).

NASA GISS ModleE

National Aeronautics and Space Administration (NASA) Goddard Institute for Space Studies (GISS) has developed a series of General Circulation Models (GCMs) to simulate global climate (Hansen et al. 1983, DelGenio and Yao 1993). The GISS climate model development process has been documented in papers stretching over 30 years (Schmidt et al., 2006). During these years, various models have been deployed to track climate change.

ModelE version is the latest GISS model incorporating numerous improvements in basic physics, the stratospheric circulation, and forcing fields (Schmidt et al. 2006). The main objective of ModelE is to simulate many different configurations of Earth System Models, including interactive atmospheric chemistry, aerosols, carbon cycle and other tracers, as well as the standard atmosphere, ocean, sea ice and land surface components . Numerous projects have been conducted using the GISS ModelE (Hansen et al. 2007, Koch and Hansen 2005, Shindell el al. 2006). Among these projects, Climate@Home (CH) is a volunteer project computer using ModelE (Sun et al. 2012) and is used as the experimental project in this dissertation.

Climate models may contain different parameters, which are sensitive to the models. If some parameters are changed, the model may get significantly different results. For example, there are seven parameters in modelE used in Climate@Home project (Table 1).

Table 1 Seven tested atmospheric parameters in the Climate@Home project

Parameter	Definition	Range
funio_denominator	Affects fraction of time that clouds in a mixed-phase regime over ocean are ice or water	5-25
autoconv_multiplier	Multiplies rate of auto conversion of cloud condensate	0.5–2
radius_multiplier	Multiples effective radius of cloud droplets	0.5-2
entrainment_cont1	Entrainment coefficient for convective plume	0.1-0.9
entrainment_cont2	Entrainment coefficient for secondary convective plume	0.1-0.9
U00a	Relative humidity threshold for stratus cloud formation	0.4-0.8
U00b	Relative humidity multiplier for low clouds	0.9-2.5

GISS ModelE uses rundecks (E001*.R) to set the running configurations. These rundeckes represent runs with the standard fixed SST atmosphere only run (E001.R), a Qflux run (E001q.R), a coupled ocean-atmosphere run (E001o.R), a varying SST (AMIP style) run (E001a.R), a Qflux with deep diffusion (E001qd.R), a stratospheric model (E001M23.R), a tracer model (E001tr.R) etc[1]. The Climate@Home project runs 300

[1] https://www.giss.nasa.gov/tools/modelE/HOWTO.html

experiments to test seven atmospheric parameters (Li, Z. et al. 2015). Table 2 shows some of the example task configurations. The differences of parameter values in these configurations will get different climate simulation results. Table 3 shows the differences in computing, programming and price of the three computing types.

Table 2 ModelE sample configurations

E4M20a_000001.R
funio_denominator=17.9737
autoconv_multiplier=0.7379
radius_multiplier=1.5081
entrainment_cont1=0.5915
entrainment_cont2=0.5336
U00a=0.4313
U00b=1.8472
	E4M20a_000002.R
	funio_denominator=24.0795
	autoconv_multiplier=0.7502
	radius_multiplier=1.7480
	entrainment_cont1=0.7495
	entrainment_cont2=0.4054
	U00a=0.7810
	U00b=1.6530

E4M20a_000003.R
funio_denominator=20.6640
autoconv_multiplier=0.7485
radius_multiplier=0.9256
entrainment_cont1=0.2560
entrainment_cont2=0.4723
U00a=0.6409
U00b=1.6727
	E4M20a_000004.R
	funio_denominator=6.6350
	autoconv_multiplier=1.1759
	radius_multiplier=1.9318
	entrainment_cont1=0.3056
	entrainment_cont2=0.4527
	U00a=0.4710
	U00b=1.3763

E4M20a_000005.R
funio_denominator=5.8805
autoconv_multiplier=1.4649
radius_multiplier=1.7026
entrainment_cont1=0.1514
entrainment_cont2=0.1257
U00a=0.7009
U00b=1.8787
	E4M20a_000006.R
	funio_denominator=16.1758
	autoconv_multiplier=0.5874
	radius_multiplier=1.1231
	entrainment_cont1=0.4154
	entrainment_cont2=0.3816
	U00a=0.6191
	U00b=1.6897

Using the GISS models, climate scientists can get the monthly values of about 300 variables such as: evaporation, land coverage, latent heat flux, precipitation, sea surface temperature.

Challenges to Run Climate Models

Although climate models could help scientists to better understand, simulate the climate change, they also bring many unique challenges that need to be carefully addressed.

Climate Model Processing Challenge

Climate modeling is a compute-intensive application, demanding a lot of computations (Allcock et al. 2002). Climate modeling varies in complexity and some models may be very complex and need massive computing resources to run. Using NASA GISS ModleE as example, a standard fixed SST atmosphere only run (E001.R) requires 3 days to finish a 10-year simulation task on a regular 2 GHz, 1G memory desktop. If scientists use 8 core servers to run 1000 models, it may take more than one year. In addition, the computing time increases when the resolution scale gets higher and time scale gets longer. Khaleel et al. (2009) reported one study of throughput rates (with a simulation time one thousand times wall clock) and computer capability requirements matches a 30-km global resolution with a 1 petaflop sustained computational speed. A 10-km grid resolution requires 5 petaflops, while a 1-km grid resolution requires 20 petaflops sustained and 100 terabytes of main memory.

In addition, scientists need to a model run many times to determine its accuracy before public release. Thousands of configurations need to be processed to determine which configuration is ideal specific cases. Thus, scientists typically require enormous computing resources to develop, test, run and fine-tune climate models. However, after testing and finishing the climate simulation, scientists may not need same enormous computing power.

Climate models normally require specific computational environments and it takes a lot time to configure the model. Most of the models are UNIX based and cannot run on other platforms directly because various libraries, programming language, and compilers are required. For example, Network Common Data Form (NetCDF) 4 libraries (Rew and Davis 1990), Fortran 95 (Metcalf et al. 2004) and GNU Compiler Collection (GCC) 4.4 (Griffith 2002) compilers are required to compile and run ModelE.

Yang et al. (2010) described performance pattern as one of the major research pursuits for a Geospatial Cyberinfrastructure (GCI) system. Climate simulation requires dedicated computing resources. Traditionally, scientists solve by increasing speed and number of CPUs by improving the network and Disk I/O. Some distributed architectures, e.g., Message Passing Interface (MPI) have also been created to support data processing. However, the traditional way to use physical computing resources may have disadvantages for processing climate data since it needs to be handled by the increasing speed of CPU, network and storage (Bertino et al. 2011). Hence, enormous computing resources are needed to run climate models in an efficient way: 1) the computing resources need to be easily scaled up or down, which means more computing resources could be acquired if there are many simulation tasks and idle computing resources could be released if there are less simulation tasks; 2) an effective way is needed to help to package climate models and run these models in different machines; 3) these computing resources should be acquired with low-cost if economic impact is important in climate simulation tasks.

Climate Data Storage Challenge

Climate modeling is a data-intensive application, creating big simulation outputs (Allcock et al. 2002). Taking ModelE as an example, the outputs of a 10-year monthly simulation task would be 10 Gigabytes of which 9 Gigabytes are temporary and 1 Gigabytes is monthly simulation results to be uploaded or returned for the further analysis. If scientists run 300 ModelE tasks, 300 Gigabytes results are collected. With the big climate modelE outputs. The evolution of technologies and human understanding of the data have shifted data handling from the more traditional static mode to an accelerating data arena characterized by volume, velocity, variety, veracity and value (Marr 2015). Data storage challenges are posed by the volume, velocity and variety of big data. It is essential to transfer and host big volume of data efficiently with low cost. In addition, the velocity of Big Data requires the storage systems to be able to scale up quickly which is difficult to achieve with traditional storage systems.

Internet software providers may provide multiple locations for users to download software, e.g., CentOS provides 34 locations for users to download CentOS installation package[1]. Scientists and users of climate projects may distribute around the world as well. A reliable global replication storage system is needed for users to download climate

[1] http://isoredirect.centos.org/centos/7/isos/x86_64/CentOS-7-x86_64-DVD-1611.iso

models and applications and to upload simulation results, or it will be difficult for users to transfer climate packages.

In addition, the storage traffic management is also a challenge if many users access the climate data storage. With the increase of climate simulation tasks and simulation timeframe, it is a challenge for the storage system to provide reliable storage with high scalability.

Climate Data Visualization Challenge

Data analysis is an important phase in the value chain of Climate Data for information extraction and predictions (Fan and Liu 2013; Chen et al. 2014b). However, analyzing Climate Data challenges the complexity and scalability of the underlying algorithms (Khan et al. 2014).

Geovisual analytical tools have the potential to provide an intuitive and convenient method for scientists to access climate data, discover the relationships between various climate parameters, and communicate the results across different research communities (Sun et al. 2012). However, implementing a geovisual analytical tool for complex climate data in a distributed environment poses a challenge to visualize the climate data with scalability. Various paradigms have been proposed for the computing vision (Buyya et al. 2009). These paradigms either build a super-fast computer or use distributed computing (Anderson 2010). Many have been done to evaluate and compare these computing paradigms (Kondo et al 2009, Foster et al 2008). Some tools have been developed to help users to better select cloud services for scientific computing (Gui et al. 2014).

Objective and Summary of Contribution

Toward tackling the aforementioned challenges, we use Volunteer Computing (VC) , virtualization technology and cloud computing for climate simulation by: 1) Using volunteer computing resources to leverage large number of home computers to support climate simulations; using virtualization technology to enable the climate models run on heterogeneous computers while providing a bit-level homogeneous computing environment; optimizing the output collection mechanism to periodically upload climate model output; and optimizing the credit system to grant credits periodically to volunteers for volunteer retention to support long time climate simulation tasks. 2) Using cloud Simple Storage Service provided by leading cloud providers to develop a global replication storage to distribute cloud models and data to global volunteers. 3) Using Load Balancing to distribute incoming WMS requests across multiple cloud instances to improve the performance; Using Auto Scaling to help to maintain climate visualization availability and allows climate scientists to dynamically scale the cloud capacity.

The dissertation is organized into several sections. Section 2 reviews related literature including three computing paradigms (supercomputer, cloud computing and volunteer computing), virtualization technology, container technology, cloud storage, auto scaling and load balancing in cloud. Section 3 analyzes the lifecycle of climate simulation tasks and present the hybrid computing infrastructure used in our approach. Section 4 introduces Volunteer Computing as a Service (VCaaS) to solve the climate model processing challenge and demonstrate the results in CH project. Section 5 states the methodology two which is a cloud based storage system to solve climate data storage challenge. Section 6 introduces auto scaling and load balancing which are used to tackle the climate data visualization challenges. Section 7 introduces the Climate@Home prototype and Section 8 draws the conclusion and discuss the future work.

CHAPTER TWO: LITERATURE REVIEW

Computing Framework

Supercomputer

A supercomputer (Hayes et al. 1986) is a computer that has highest processing capacity, which is far higher than a normal one. Supercomputers were first introduced in the 1960's primarily by Seymour Cray at Control Data Corporation (CDC). The first generation of supercomputers used only a few processors. Since then, more and more supercomputers have been created with increasing computing power. In 2013, the world's fastest supercomputer, Tianhe-2, was deployed at the National Supercomputer Center in Guangzhou, China. It provided a performance of 33.86 petaFLOPS (Floating-point Operations Per Second) on the Linpack benchmark.

In 2004 NASA built the Columbia supercomputer to increase NASA's high-end computing capability ten-fold for missions in aeronautics, space exploration, and earth and space sciences. Columbia is a constellation comprised of 20 nodes, each containing 512 Intel Itanium2 processors and running on the Linux operating system (Brooks et al. 2005). Scientists have used Columbia to solve many problems across climatological disciplines. For example, Menemenlis et al. (2005) used Columbia to estimate ocean circulation constrained by in situ and remotely sensed observations; and Mavriplis et al. (2007) investigated the high-resolution aerospace applications.

The supercomputer provides fast computing for complex applications and simulations. However, supercomputers have the disadvantages of being extremely expensive to develop and use. With thousands of processors, they consume large amounts of electrical power and generate lots of heat. The cost to power and cool the supercomputer is significant. Moreover, supercomputers require special programming and hardware skills to write the code for applications.

Cloud Computing

Cloud computing enables end users to access pool of computing resources (e.g., networks, servers, storage, applications, and services) via network. The resources can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell and Grance 2009). Cloud computing provides 'Infrastructure as a Service' (IaaS), 'Platform as a Service' (PaaS), 'Software as a Service' (SaaS), and 'Data as a Service' (DaaS) for end users in a 'pay-as-you-go' mode. Yang et al. (2011) coined "Spatial Cloud Computing" to refer to the cloud computing paradigm driven by the geospatial sciences and optimized by spatiotemporal principles.

Cloud Computing provides an alternative way for scientific applications (Keahey et al. 2008, Vecchiola el al. 2009). Scientists can easily access the various cloud resources by purchasing computing resources, storage resources and network resources. Furthermore, scientists can scale up or down the resources based on the application requirements. Cloud computing is cheaper than supercomputers (Losup et al. 2011). Huang et al. (2010) used Amazon AWS cloud to support Global Earth Observation System of Systems (GEOSS) Clearinghouse (Liu et al. 2011) deployment. Li et al. (2014)

proposed Model as a Service (MaaS) to use cloud computing to enable geoscience models.

However, cloud computing is still expensive for climate simulations. Given cloud computing ‘pay as you go’ mode, the more computing resources consumed, the more one needs to pay. The scales of climate simulations may vary from days to hundreds of years, the time needs for the simulations may be very long. With hundreds or thousands of simulation tasks, the cost to buy cloud computing resources may be out of the budget.

Volunteer Computing

Volunteer Computing (VC) uses internet-connected computers volunteered by their owners, as a source of computing power and storage (Anderson and Fedak 2006). Kondo et al. (2009) determined the cost-benefits of cloud computing versus VC applications and concluded the VC overheads for platform construction, application deployment, compute rates, and completion times. Interest in volunteer computing (VC) reflects general interest in end-user participation in scientific activities. In the geospatial domain, this interest is referred to as Volunteered Geographic Information (VGI, Goodchild 2005, 2007), or geocrowdsourcing, a significant research trend reviewed by Rice et al. (2012, 2013).

Since 1996 in which the first VC project "Great Internet Mersenne Prime Search" was launched, VC has been used in a wide range of scientific projects such as SETI@Home (Anderson et al. 2002) and Folding@Home (Larson et al. 2002). Climateprediction.net is a successful VC project in climate research using BOINC (Stainforth et al. 2002).

In these VC projects, most volunteers' computers run Microsoft Windows operational system, however, most climate models are UNIX based. Efforts are needed to modify the models to different systems. Stainforth et al. (2002) transferred the Climateprediction.net code to windows and found the main difficulties were the identification of suitable compiler options and changes in the way environment variables are used.

Comparing the three computing type: supercomputer, VC and cloud computing. Table 3 shows the differences in computing, programming and price of the three computing types.

Table 3 Comparison of super computer, cloud computing and VC

Computing type	**Computing Efficiency**	**Programming**	**Price**
Supercomputing	Data moved between processors rapidly	Difficult to write programs	More expensive than other two
Cloud Computing	Less efficiency	Standalone programming	Cheaper than supercomputing
VC	Less efficiency	Standalone programming	Cheaper than cloud computing

In light of the advantages and disadvantages, VC is the most cost-efficient paradigm to run the climate simulations so we selected VC as the computing type for the CH project. This paper extends the VC and virtualization technology by minimizing the size of Virtual Disk Image (VDI), uploading results periodically and new credit system for the climate simulation applications.

Virtualization Technology

Thanks in part to the facilities provided by virtualization technologies, VC now can melt with Virtual Machine Monitor (VMM) to provide an easy way for scientists to publish their applications to different operations platforms. Numerous systems have been designed to subdivide the ample resources of a modern computer (Barham2003, Liu and Abali 2009). Using virtualization technologies, Virtual Machine (VM) could be created from the underlying hardware resources and act like a real computer with an Operating System (OS). Despites the underlying OS, VMs could be launched with different OS. For example, a computer that is running Microsoft Windows may host a VM that looks like a computer with the Ubuntu Linux operating system; Ubuntu-based software can be run on the VM (Turban et al. 2008).

There are advantages for VC to use virtualization technology. First, it packages libraries, programming language, compilers and operating systems to a blackbox and makes it easier for scientists to develop applications. Second, it provides increased security for volunteers since VMs are a very strong security barrier. VM blocks the program running in it to access to the files on the volunteer's machine. Third, VM apps are automatically "restartable". The contents of the VM are written to disk every few minutes, and if your computer is turned off the application can restart close to where it left off. Recognizing the extreme benefits of using VMs, Several VC projects have been using VMs, including Test4Theory (Lombraña, 2012), Beauty@LHC and RNAworld.

These applications are different from climate simulation tasks since they don't create large outputs like the climate simulations tasks. For example, Test4Theory create 3-4 Megabytes per 24 hours. Compared to these projects, climate models and applications require a convenient and fast way to upload big outputs.

Docker Container

Recently container technology grows rapidly and becomes an important virtualization solution. Where the traditional virtual machine utilizes the virtualization at machine level, container implements virtualization at the operational system level, so many containers share the same kernel instance (Fink, J. 2014). Docker container is one of the most popular container solutions, since it provides the ability to package applications and their dependencies into lightweight containers that move easily between different distros, start up quickly and are isolated from each other (Merkel, D. 2014).

Compared to the traditional virtual machine, container technology has the advantages to provide lightweight virtualization so container can be built, modified, rebuilt and shared far more rapidly (Stubbs el al. 2015). Felter e al. (2015) explore the performance of traditional virtual machine deployments and contrast them with the use of Linux containers and their results show that containers result in equal or better performance than VMs in almost all cases.

Auto Scaling and Load Balancing

Cloud computing can provide two significant advantages: auto scaling and load balancing. Auto scaling is a method used in cloud computing to automatically scale up or down computing resources based on the measurement of metrics (CPU, network etc).

Load Balancing automatically distributes incoming application traffic across multiple cloud instances[1].

The auto scaling and load balancing are widely used since they can provide advantages to: 1) save the cost to use the cloud computing by scale up more machines for the computing tasks if needed and scale down machines to save cost if tasks have finished or only a few machines are needed; 2) improve the performance by distributing traffic to two or many machines. With the advantages provided by auto scaling and load balancing, Huang et al. (2010) deployed Global Earth Observation System of Systems (GEOSS) Clearinghouse (Liu et al., 2011) to Amazon AWS cloud. Mao et al. (2011) presented an approach to use auto scaling to minimize cost and meet application deadlines in cloud workflows and specify performance requirements by assigning (soft) deadlines to jobs. Dougherty et al. 2012 presented a model-driven engineering approach to optimizing the configuration, energy consumption, and operating cost of cloud auto-scaling infrastructure to create greener computing environments that reduce emissions resulting from superfluous idle resources. Kolb et al. (2012) proposed and evaluated two approaches for such skew handling and load balancing to support blocking techniques to reduce the search space of entity resolution, utilize a preprocessing MapReduce job to analyze the data distribution, and distribute the entities of large blocks among multiple reduce tasks.

[1] https://aws.amazon.com/elasticloadbalancing/

CHAPTER THREE: A HYBRID COMPUTING INFRASTRUCTURE

Climate Simulation Lifecycle

Climate models are developed using different languages and based on various libraries. Figure 1 shows the lifecycle for a typical climate simulation task. Scientists need to preprocess the climate models, i.e., select a specific operation system with specific version then install and configure models on the specific machine. Then scientists can use scripts (some command lines tools, e.g., "runE_new" command is used in NASA GISS ModelE) to process the climate models to start simulation. In the process of simulation tasks, climate models will generate results (e.g., hourly, daily or monthly results) continuously. Then scientists to store these results to a stable storage system which could be disks or network storage such as NAS (Network-Attached Storage), SAN (Storage Area Network). Finally, scientists need to analyze and visualize the simulation results. and pedestrians cross eight lanes of traffic to walk from a new metro stop to nearby workplaces (**Error! Reference source not found.**).

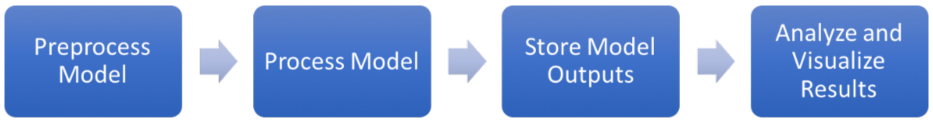


Figure 1 Lifecycle of climate simulation project

During the lifecycle, scientists may face the challenge discussed in chapter 1. Hence, a hybrid infrastructure is proposed and implemented in this dissertation to improve the climate simulation based on the lifecycle.

Hybrid Computing Infrastructure

The hybrid computing infrastructure uses cloud computing and volunteer computing to provide a low-cost, scalable, reliable and distributed environment for processing, storing, analyzing and visualizing climate simulations. Figure 2 shows the different components in hybrid computing infrastructure.

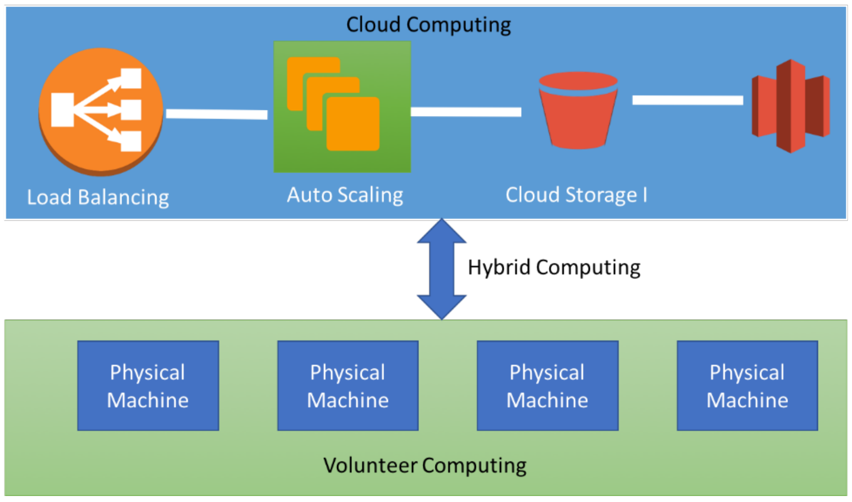


Figure 2 Hybrid computing infrastructure for climate simulation

Volunteer computing provides massive computing resources with low-cost or no-cost. The prepressing and processing may take long time form climate simulation tasks using traditional infrastucture. Li et al. (2016) analyzed climate models and stated that it needs 12 days to set up ModelE environments and 38 days to run ModelE for a Ph.D.

student from GMU collaborated with a NASA scientist. With the advantages of Volunteer Computing as a Service (VCaaS), volunteer computing can provide a significant improve to preprocess and process climate models.

Cloud computing offers the ability to store data in its various storage systems such as Simple Storage Service (S3), Glacier, Elastic Block Storage (EBS). These options could provide reliable storages with low-cost. In addition, there are storages cloud be easily scale up or down and charge the cost based on “pay-as-you-go” mode.

Auto scaling and load balancing are used in the hybrid computing infrastructure to help to visualize and analyze climate simulation results. First, two or more instances with visualization tools can be registered in load balancer to distribute the incoming and outcoming traffics. Second, an auto scaling group could be created to scale up or down instances based on the requirements.

CHAPTER FOUR: VOLUNTEER COMPUTING AS A SERVICE

Introduction

Although volunteer computing has many advantages for providing scalable and free computing resources for scientific applications, significant efforts are needed to set-up a volunteer computing project. In particular, climate simulation projects need specific environments to run and it needs professional knowledge to convert the climate models to different OS versions.

A Volunteer Computing as a Service (VCaaS) is proposed in the dissertation to help climate scientists to use volunteer computing.

Definition: *VCaaS is a service that provides volunteer computing resources allowing consumers to run and manage applications on volunteers' computers.*

VCaaS Architecture

The infrastructure used in the dissertation has a four-layer structure (Figure 3).

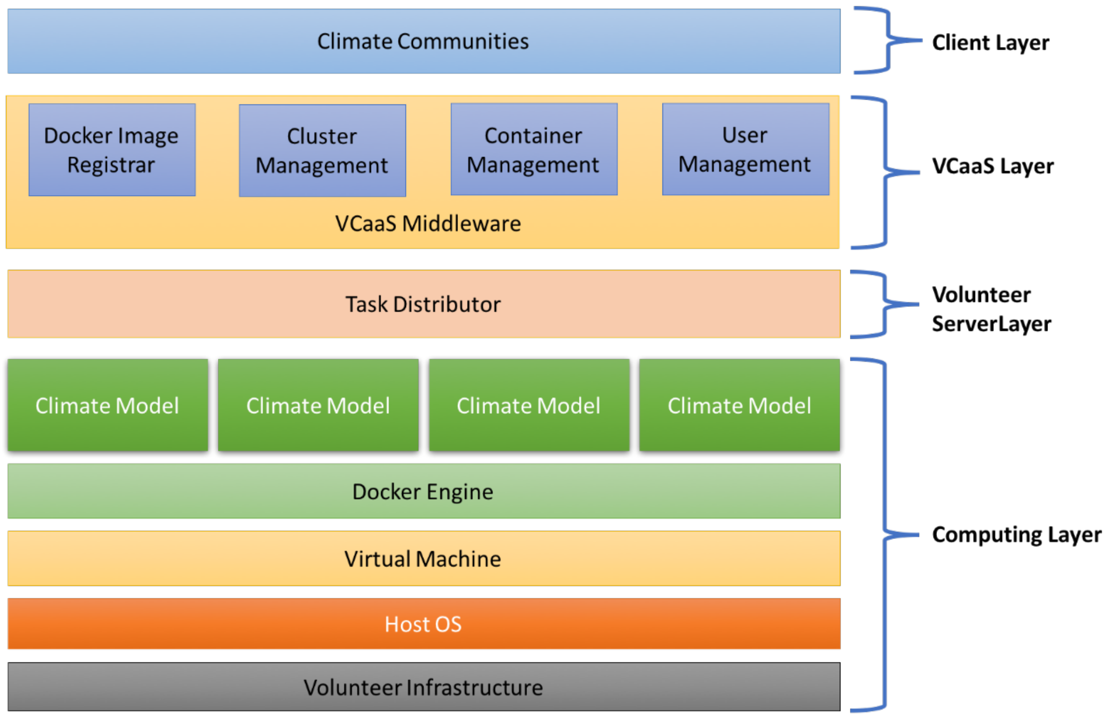


Figure 3 Volunteer computing infrastructure for climate simulation

The first layer comprises the computing layer. In this layer, volunteer computing client middleware and virtualization software are installed on volunteer's machine. In Cliamte@Home project, Berkeyley Open Infrastructure for Network Computing (BOINC) is used as volunteer computing client middleware, and The BOINC client is responsible for communication with the BOINC server, science application and data downloading, and configuration of donated computational resources such as (e.g., how much CPU, RAM and hard disk to be donated). The climate model runs in the virtual machine launched by the virtualization software. In Climate@Home project, VirtualBox is used as virtualization software to host a CentOS 7 virtual machine on volunteer's computer. A Docker CE (Community Edition) is running in the virtual machine. With the Docker service, different climate models could be running as container.

The second tier is the server, containing volunteer computing server middleware. The BOINC server middleware is installed in Climate@Home project and (i.e., geosciences models, applications and necessary environments), assigns climate simulation tasks, gets feedback from the volunteer's hosts, and grants credits to the volunteers who have already run the climate simulation.

The third tier is the VCaaS layer. It is a web portal used to upload, manage, delete the climate model container images and related data. After uploading, the model and data could be replicated to different cloud storage endpoints.

The top tier is client, which presents the project status for scientists. Meanwhile, A Graphic User Interface (GUI) is deployed to visualize the climate simulation results.

VCaaS Workflow

A typical workflow for volunteer computing and virtualization for Climate@Home (Figure 4) highlights several features of the research. There are six steps in the workflow to preprocess, run climate models and store results. In this workflow, scientists don't need to know how to do parallel programing to write their code and don't need to understand how to distribute their applications onto a volunteer computing platform.

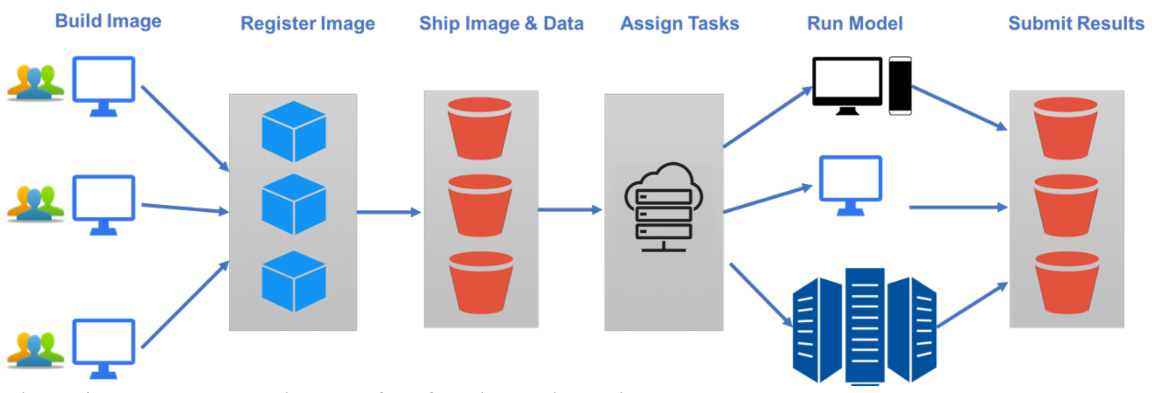


Figure 4 Volunteer computing workflow for climate simulation

Step 1: Build image: a lightweight Docker image is built to run climate models in this step. The Docker image includes everything needed to run climate model, including related libraries, model code, a runtime, environment variables and config files. With the Docker image, a Docker container could be created.

Step 2: Register image: scientists need to upload and register the container image in Step 1 to VCaaS portal.

Step 3: Ship image and data: this step is automatically implemented by VCaaS portal. Container images and climate data are replicated to multiple copies and shipped to different web locations for download. In this dissertation, Amazon Simple Storage Service (S3) which provides object storage with a simple web service interface to store and retrieve any amount of data from anywhere on the web is used save and distribute and image and data. Amazon S3 is designed to deliver 99.999999999% durability, and scale past trillions of objects worldwide .

Step 4: Assign tasks: the volunteer computing server assigns tasks to global volunteers in this step.

Step 5: Run models: volunteers' computers automatically run climate models in this step.

Step 6: Submit results: After finish, submit the result to cloud storages.

Shrink Volunteer Computing data

The Virtual Disk Image (VDI) file of VirtualBox is used in the Climate@Home project to store ModelE and data. In addition, the VDI file also contains an operation system and required libraries and tools. Climate@Home version 1 which was terminated before 2015 uses Ubuntu 12.04 as the basic operation system in the project. After the Ubuntu 12.04 has been installed in the VDI, GCC 4.4, GFortran, OpenJDK 1.7 and NetCDF are installed before ModelE's installation. Finally, ModelE is installed and model data are transferred to the VDI. In addition, a periodically upload application is also installed in the VDI.

The VDI file is uploaded to BOINC server middleware, and Volunteers download it automatically to run ModelE after they added Climate@Home project in their BOINC client. To save download time for the virtual image and volunteers' network cost, the following steps minimize the size of the VDI file:

1. Compress the virtual image before sending to the volunteers. GZIP (Deutsch 1996) is used, which reduces the size of the VDI file by about 50%.

2. Shrink the virtual image. The VDI file uses "Dynamically Expanding Storage" option to allocate disk storage and save the ModelE outputs. The option expands the VDI size on preparation of the original VDI by expanding the disk when install new applications (GCC 4.4, GFortran, OpenJDK 1.7) in the VM. However, the VDI size does

not shrink or return to its previous size when the installation is finished. In this project, zerofree patch (Boutcher and Chandra, 2008) frees the expanded spaces. Then, clonehd (Dash 2013) creates a small copy of the VDI file. The small copy operates same as the original VDI file and is uploaded to the BOINC middleware.

3. In the installation of GCC, GFortran, and OpenJDK, it will create the cache of package files. By default, the Ubuntu keeps all the packages it has downloaded in case they are needed in the future. This makes the Virtual Image larger than necessary. Since the virtual machine is only used to run ModelE and volunteers don't need to modify or upload it, the cache of packages files are cleaned to make the virtual machine image smaller.

Since the first version of Climate@Home released in November 2012, 16 versions have been released through April 23, 2014 in Climate Version 1. The first version of VDI (Figure 5) was about 2.4 Gigabytes, and declined to 1.2 Gigabytes using GZIP to compress in version 2.0. Subsequently, the size dropped to about 850 Megabytes after shrinking in version 4.0. The current size is less than 700 Megabytes after all the cache of package files are cleaned.

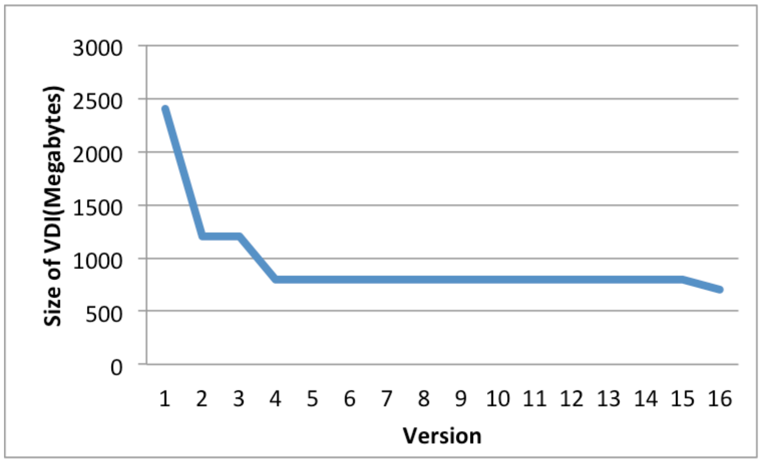


Figure 5 Size history of virtual image history in Climate@Home Version 1

Periodically Upload Mechanism

In BOINC's traditional upload mechanism, the output from volunteers machine are uploaded after application finishes. This presents two problems for climate simulation tasks. First, the climate simulation task always creates large output files which are difficult to upload incrementally. Second, the output for climate simulation tasks is always organized in time series with uniform time intervals (e.g., hourly, daily, monthly, yearly). To recognize the specialty of the climate simulation outputs, a Periodically Upload Mechanism (PUM) has been used in the Climate@Home project (Figure 6).

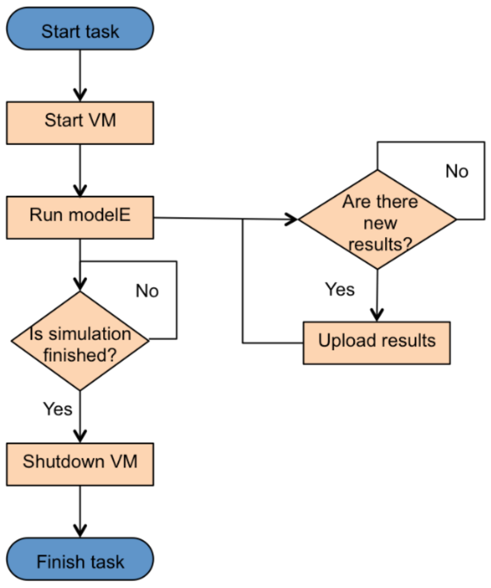


Figure 6 Periodically Upload Mechanism

Two checkers are added in the PUM. The first checker determines if a new result is created. The second checker determines if the task is completed. The PUM starts to run once the VM is launched. The two checkers run in set time intervals (25 minutes is used in Climate@Home project). If the results checker finds that a new new result is created, the result is uploaded to FTPS server. If the task checker finds that the simulation task is finished, the VM is shutdown.

Credits are used to present how much work a computer, a user, or a team has contributed to the Volunteer Computing project. It is very critical for volunteer retention since increasing credit gives individual volunteers confirmation that their continuing contribution and credit provide a basis for competition among users and teams (Anderson and Koren 2004). Coleman et al. (2009, 2010) Paez (2014), and Rice et al. (2014, 2015) discuss the motivations and complexities of volunteer-centered data production, and the difficulty of recruiting and retaining volunteers. Thus, it is one of the most important incentives for participation in the Volunteer Computing project. The BOINC provides two credit systems: 1) the first credit system which grants the credits based on the CPU runtime; 2) the second credit system which grants the credits based on the FLOPS performed by the application. They are awarded in small increments, according to the type and length of time to run a climate model.

However, these are not suitable for Climate Simulation projects since they only grant credits for completed tasks and climate projects are always take a long time to run. Thus, it is not acceptable for some users to wait a long time to accumulate the credits. A New Credit System (NCS) was created to address these problems and increase users' incentives.

In the NCS, credits are granted in the two steps: check at intervals the FTPS server to get the number of monthly results and granting the host based on the hourly, daily, monthly or yearly results. The NCS grants credits to those successful tasks using Equation 1,

Equation 1 New Credit System

$$C = n * CM$$

C is the granted credits, n is the number of finished sub results (e.g., hourly, daily, montly or yearly) and CM is granted credits for every month. The value for CM in the Climate@Home Version 1 is 30.

Climate simulation visualization is critical for both scientists and volunteers. However, interactive geovisual analytics over the Internet to facilitate collaborative climate research are immature (Sun et al. 2012). In order to analyze the ModelE output, a web-based climate visualization system in the Climate@Home project was developed.

Validate Climate Computing Results

Volunteer computing is a public computing ecosystem which means anyone (i.e., academic researchers, students, hobbyists, malicious hackers) can participate and donate their computing resources to run climate models. It brings several security problems: 1) some hackers may use their computer to provide incorrect results; 2) some computer may lack of security protections and provide incorrect results as well. Hence, scientists face a critical problem to distinguish correct results and incorrect results.

VCaaS assigns same computing job to different hosts with different users and then compare the results. Suppose there are one hundred hosts and five of them are hacked to provide incorrect results. Table 4 shows the trust percentage is growing when the replications increase. When there is only one replication, the trust percentage is 95% since 5 of 100 volunteers' hosts are hacked. When there are two replications increate to 2, the chance to get two incorrect results for a job is 0.25% and the trust percentage is 0.25%. The trust percentage for is 99.994% for 3 replications and 99.99988% for 4 replications.

Table 4 Trust percentages for different replications when 5 of 100 volunteer hosts are hacked

Replication	Trust Percentage
1	95%
2	99.8%
3	99.994%
4	99.99988%

Climate@Home project uses 3 replications to improve the trust percentage. In Climate@Home project, volunteer server assigns a modleE job to 3 different hosts. In addition, Climate@Home project adds a constraint that the 3 hosts must belong to 3 different users since hackers may have multiple hosts and may result some error results if the job assign to one user. Figure 7 shows the example replication to assign 32 tasks to 24 hosts. Each host can run up to 4 tasks in this case, and Climat@Home assign tasks to different hosts. For example: task 1 is assigned to host 1, host 15 and host 16 while task 2 is assigned to host 1, host 12 and host 17.

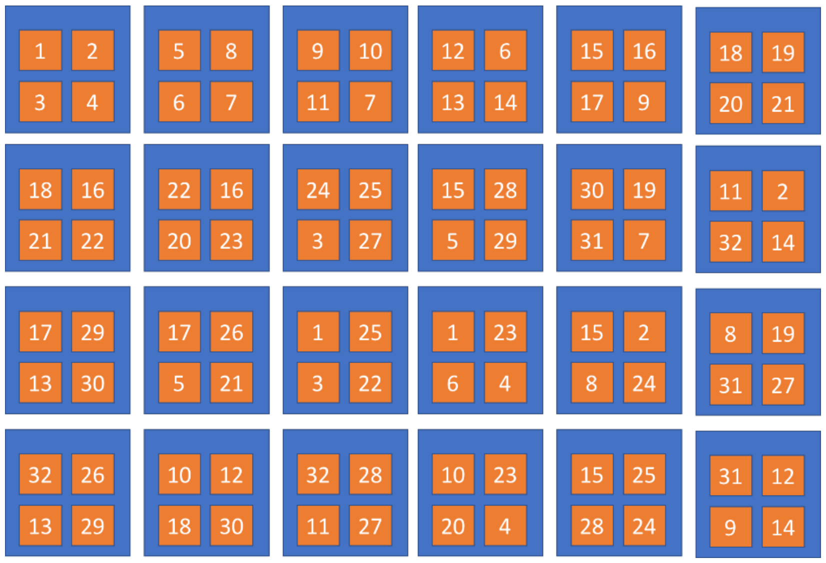


Figure 7 Climate task replication

A validator is used in VCaaS to decide whether results submitted by volunteers are correct and which completed jobs are "valid". Figure 8 is the pseudocode to validate results in Climate@Home project. The validator queries and reads all results for a specific task first. Then the first result is supposed to be correct and is used to compare to reset results byte by byte. If there are some difference, then the simulation task failed and Volunteer Computing server will create a new job. If all three results are same, the validator will mark the simulation task is valid and keep one result to reduce redundancy.

```
input: climatetask: a climate simulation task
Validate
        results = Get all results for climatetask
        result1 = first result
        i = 1;
        while i < results.size
                result2 = results.get(i);
                valid = compare result1 and result2;
                if valid is false
                        create new climatetask;
                        return;
        climatetask is valid;
        Keep one result to reduce redundancy
        Update the status of climatetask in database;
```

Figure 8 Validator used in Climate@Home project

CHAPTER FIVE: METHODOLOGY II: CLOUD STORAGE FOR VCAAS

Introduction

After registering in the volunteer computing project and adding the project to volunteer client, volunteers need to download project applications and data from locations provided in the project. However, the download may be terminated because of the network situations. Anyone around world could register in volunteer computing to provide computing hosts, hence, the hosts may distribute world around.

To support a continuously increasing demand for internet data traffic, the Global Submarine Cable Infrastructure is made up of fiber optic cables that lie on the ocean floor (Omer et al. 2009). Figure 9 shows the Global Submarine Cable map of 2016[1]. The reliability of submarine cables is high and speed is fast. However, the different countries may have different network speed to connect to volunteer computing project website. It causes many volunteers can not download climate applications and data fluently, some volunteers can not upload outputs as well.

[1] http://submarine-cable-map-2016.telegeography.com/

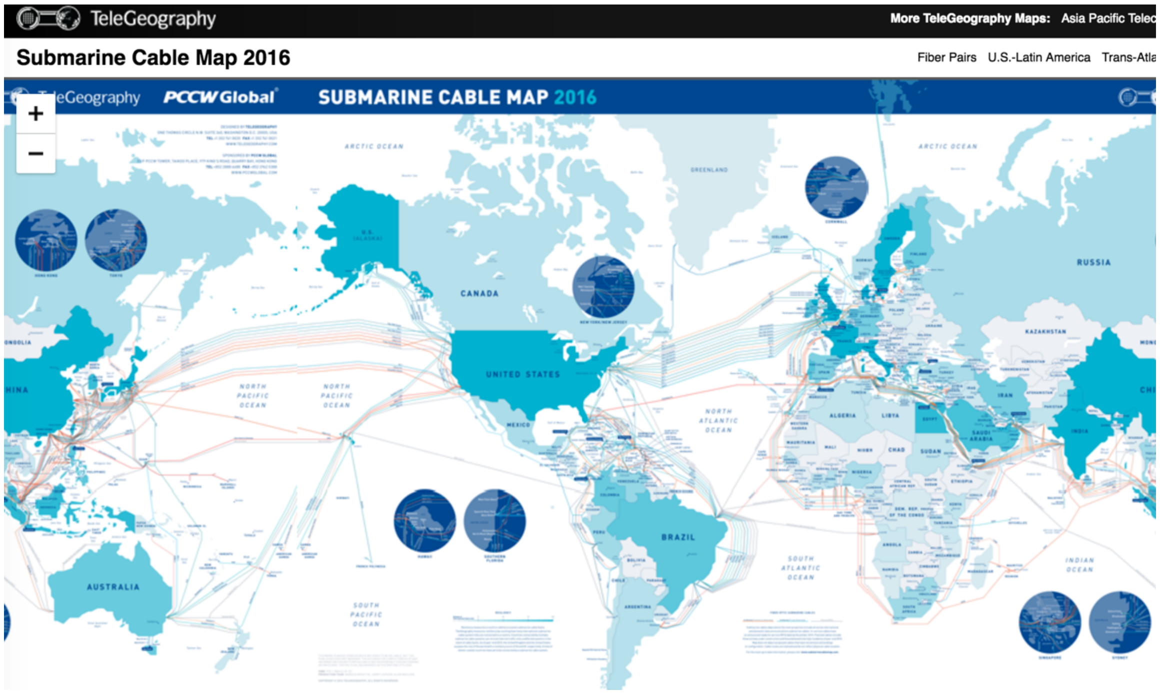


Figure 9 Submarine Cable Map 2016

Volunteer Global Distribution Pattern

Since the BOINC does not provide the location information about the hosts. That information is derived from external IP of hosts from the BOINC database (Figure 10 shows the algorithm used to get the location and add to google map). For each host, BOINC saves is Internet Protocol (IP) address (both external IP address and private IP address) in host table. Then use the following Structured Query Language (SQL) language to get all the external IP addresses and linking that to the web IP locater API to get the location of the host's location.

```
MapHosts
    IPStatus Sets = Run SQL "select external_ip_addr, status from host";
    For each IPStatus of IPStatus Sets; Do
        IF IP is Valid; Then
            Get Coordinates using IP;
            Placemarker = "activate";
            IF status is "inactivated"; Then
                Placemarker = "inactivate"
            Add Placemarker in Google Map;
```

Figure 10 Algorithm used in Climate@Home to map hosts

From the hosts map (Figure 11), the volunteers are from 40 different countries with most located in United States and Europe. The different locations connect to the Climate@Home using the submarine cables and they have different network traffics. The various network traffics cause a problem for volunteers to download climate models and data, e.g., in the Climate@Home project almost all hosts in China can not download the ModelE virtual machine and data successfully.

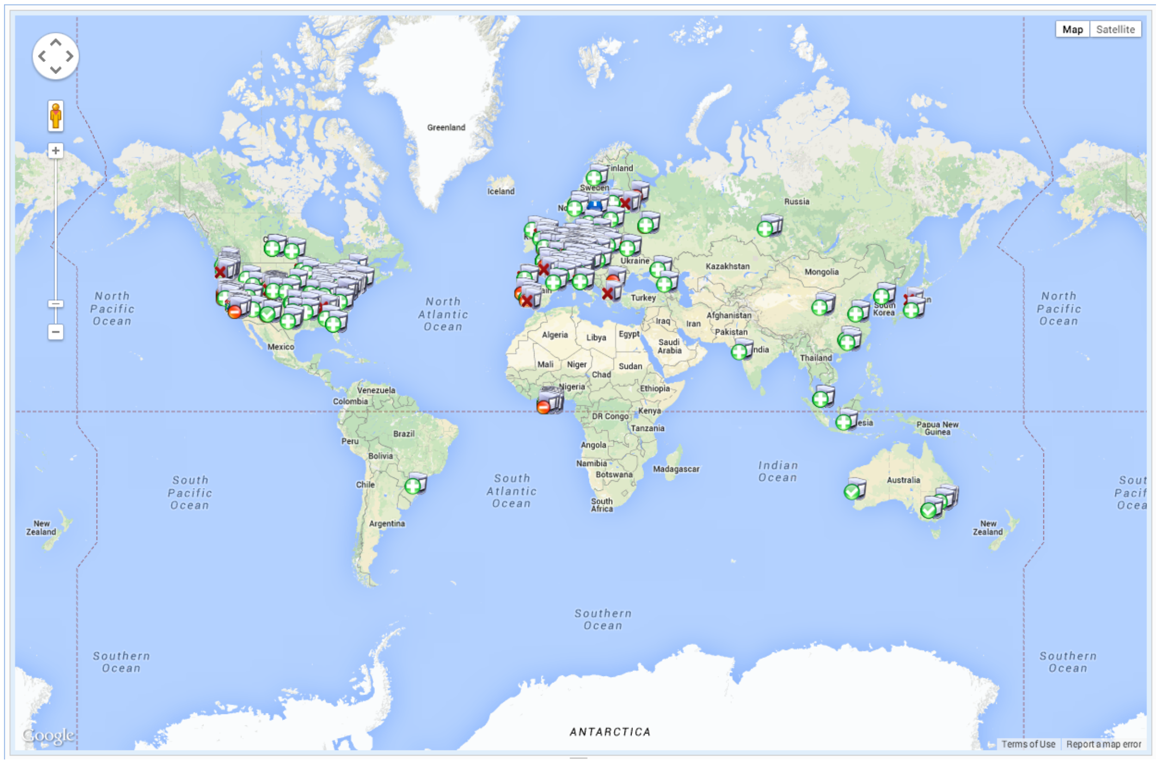


Figure 11 Hosts location in Climate@Home project

Hence, it requires an efficient and reliable way to transfer a big virtual image, a docker image and climate data to volunteers' hosts. Yang et al. (2015) described spatiotemporal computing which the computing paradigm that utilizes spatiotemporal principles to devise cutting-edge computing technologies and solutions to tackle this.

Cloud Global Resources

Fortunately, leading public commercial cloud providers have built many datacenters to compete in the global markets. For example, the AWS Cloud operates 43 Availability Zones within 16 geographic Regions around the world, with announced plans for 11 more Availability Zones and four more Regions[1]. Aliyun cloud has 14 global

[1] https://aws.amazon.com/about-aws/global-infrastructure/

data centers including 8 outside of China[1]. The global distribution of cloud computing datacenters provides an advantage for volunteer computing to host their data and application world-wide. Figure 12 shows the 16 data centers of Amazon AWS and Aliyun cloud used in Climate@Home project.

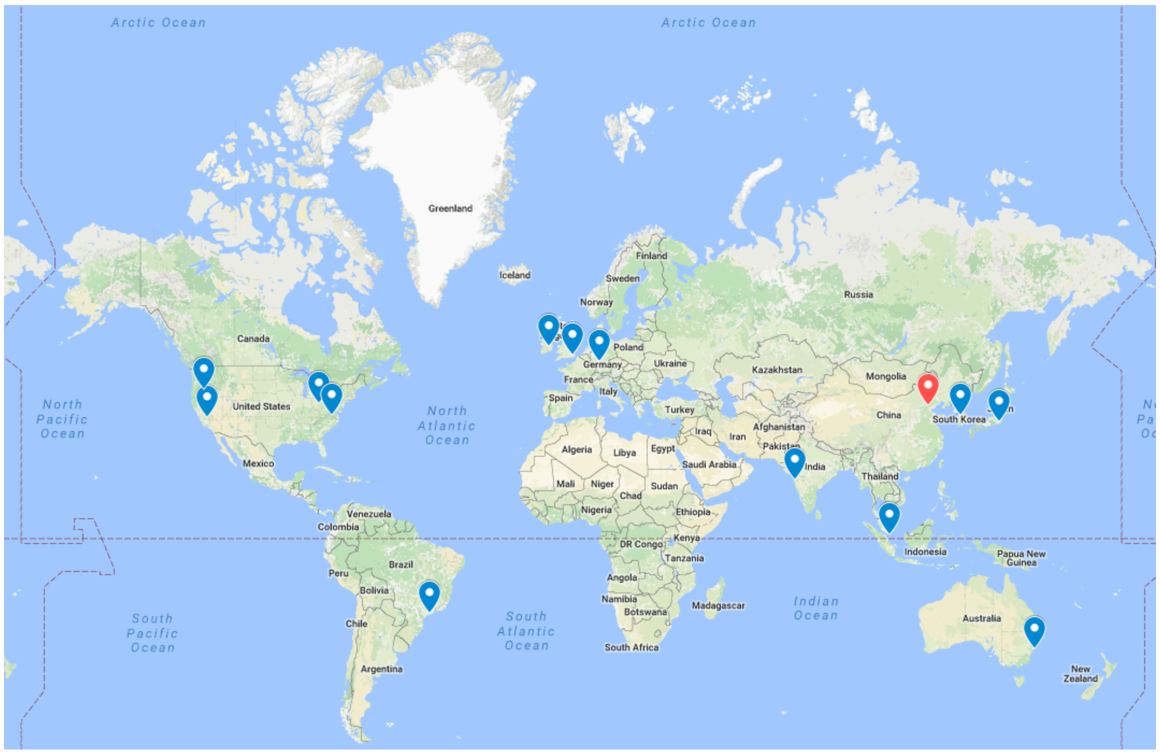


Figure 12 Cloud datacenters used in Climate@Home: blue icons are AWS datacenter while the red icon is Aliyun datacenter.

With the 16 datacenters, Climate@Home create 16 buckets which is the storage location in Amazon S3 and Aliyun S3. A bucket is a logical unit of storage (folder) in

[1] https://techcrunch.com/2017/02/27/alibaba-aliyun-cloud-computing/

cloud object storage service S3[1]. By default, customers can provision up to 100 buckets per AWS account[2] and 10 buckets per Aliyun account[3]. Buckets can be used to store objects which consist of data and metadata that describes the data. The maximum size of a single object could be up to 5 Gigabytes.

Cloud consumers can create S3 buckets in different datacenter, i.e, different regions. However, the bucket names are globally unique for one cloud provider, i.e., AWS bucket names are globally unique regardless of the AWS Region in which customers create the bucket. Aliyun bucket names are globally unique as well. For different cloud providers, the bucket names could be same. For example, a user create a climate" bucket in AWS US-EAST-1 region, other users can not create bucket with same name in any region in AWS. However, Aliyun could create a "climate" bucket if the name is not used by anyone in Aliyun cloud. Combine the bucket name and cloud region domain, S3 can provide a unique endpoint to upload and download data. Table 5 lists the bucket name, location and endpoint used in Climate@Home project.

[1] http://searchaws.techtarget.com/definition/AWS-bucket
[2] https://aws.amazon.com/s3/faqs/
[3] https://bbs.aliyun.com/simple/t112349.html

Table 5 Name, location and endpoint of 16 buckets used in Climate@Home project

Bucket Name	Location	Endpoint
climateinfooregon	US West (Oregon)	s3-us-west-2.amazonaws.com
climateinfoohio	US East (Ohio)	s3.us-east-2.amazonaws.com
climateinfoireland	EU (Ireland)	s3-eu-west-1.amazonaws.com
climateinfocentral	Canada (Central)	s3.ca-central-1.amazonaws.com
climateinfosydney	Asia Pacific (Syndney)	s3-ap-southeast-2.amazonaws.com
climateinfoseoul	Asia Pacific (Seoul)	s3.ap-northeast-2.amazonaws.com
climateinfovirginia	US East (N. Virginia)	s3.amazonaws.com
climateinfofrankfurt	EU (Frankfurt)	s3.eu-central-1.amazonaws.com
climateinfosaopaulo	South America (Sao Paulo)	s3-sa-east-1.amazonaws.com
climateinfosingapore	Asia Pacific (Singapore)	s3-ap-southeast-1.amazonaws.com
climateinfocalifornia	US West (N. California)	s3-us-west-1.amazonaws.com
climateinfomumbai	Asia Pacific (Mumbai)	s3.ap-south-1.amazonaws.com
climateinfotokyo	Asia Pacific (Tokyo)	s3-ap-northeast-1.amazonaws.com
climateinfolondon	EU (London)	s3.eu-west-2.amazonaws.com
climateinfooregon	China (Beijing)	oss-cn-beijing.aliyuncs.com

Data Download/Upload Matrix

When users download or upload data, they may have different network speed to access these different buckets. “ping” is a network software used to test the reachability of a host and it is widely used in different OS (Linux, Windows and Max OSX). “ping” is used to measure the round-trip time for messages sent from the volunteer’s host to S3 region endpoint in Climate@Home project.

Table 6 shows ping results from volunteer’s host in different country to S3 endpoint. The lower number means the host has better network speed to access that specific S3 region. For example, for ping results of the host in united states to in this table, the result to Virginia region is minimum as 3 and the result to Beijing is maximum as 277, the Virginia S3 endpoint is the best choice for this host to upload and download data and Beijing S3 endpoint is the worst choice. Another example is the host in China which has minimum ping result (5) to Beijing. The ping results for this host to any other region is bigger than 100 except Tokyo. Climate@Home version one hosts the download website in George Mason University and almost no Chinese hosts can download the virtual image and climate model data successfully.

Table 6 "ping" results from volunteers' host to S3 region.

Host / S3 Region	United States (Virginia)	Czech Republic (Prague)	Australia (Sydney)	Poland (Katowice)	Italia (Corato)	China (Beijing)
Virginia	3	116	228	137	175	289
Ohio	14	125	235	160	174	297
California	69	192	165	205	190	291
Oregon	68	204	190	220	219	230
Central	17	125	231	146	135	283
Mumbai	251	290	342	478	308	195
Seoul	194	306	259	368	333	150
Singapore	247	285	110	390	295	138
Sydney	207	339	11	391	345	231
Tokyo	185	284	210	313	305	84
Frankfurt	88	17	325	58	64	407
Ireland	88	39	310	67	64	389
London	78	31	303	58	78	350
São Paulo	128	256	352	253	428	402
Beijing	277	257	259	227	532	5

With the advantages of the global distribution of cloud storage, Climate@Home version uses the VCaaS infrastructure and replicate data to 16 S3 regions. After receiving the climate simulation tasks, each host will run an application to select the best endpoint and download data from that host. Figure 13 is the pseudocode to implement the steps.

```
GetBestRegionAndBucket
    Best_Ping = Max_value
    Best_endpoint=Virginia
    Best_bucket=cliamteinfovirginia
    For endpoint in endpoints Do
        Run ping command 5 times
        Get the average ping time
        If (ping < Best_ping) Then
            Best_Ping = Ping
            Best_Endpoint=S3_ENDPOINT
            Best_bucket = bucket in this region
DownloadDataFromBucket
    Best_Download_Url = Bestbucket + Best_Endpoint
    Download virtual image, docker image and data from Best_Download_Url
```

Figure 13 Pseudocode used in Climate@Home project to select best cloud storage bucket and download data

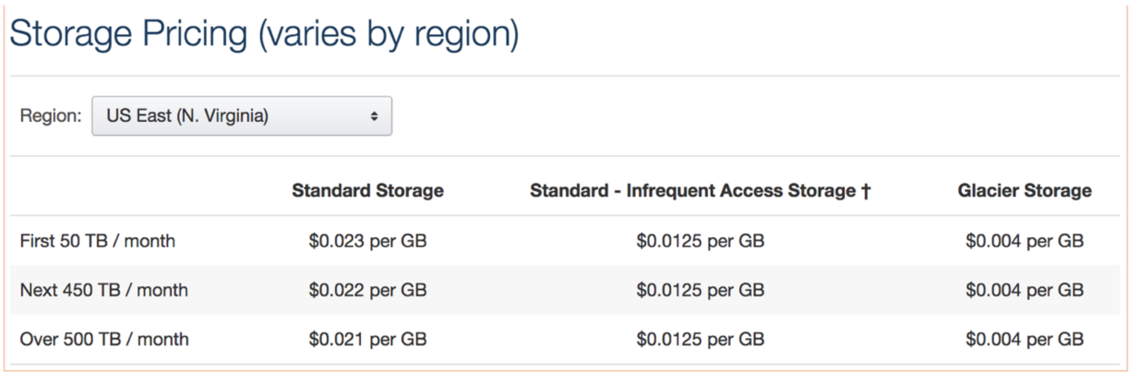
The same function to get best region and bucket can be used to upload outputs. Figure 14 shows the pseudocode used in Cliamte@Home to select best cloud storage bucket and upload outputs. Using PUM, hosts can upload outputs to that best hosts. With the naïve cloud storage command line, VCaaS can upload data to buckets easily. Following is the psedocode to upload results:

```
GetBestRegionAndBucket
    Best_Ping = Max_value
    Best_endpoint=Virginia
    Best_bucket=cliamteuploadvirginia
    For endpoint in endpoints Do
        Run ping command 5 times
        Get the average ping time
        If (ping < Best_ping) Then
            Best_Ping = Ping
            Best_Endpoint=S3_ENDPOINT
            Best_bucket = bucket in this region
UploadDataToBucket()
    While true Do
        If task has finished Then
            Shutdown Virtualmahine
        If BestRegion is AWS Region Then
            Upload to AWS region using AWS S3 command
        If BestRegion is Aliyun Region Then
            Upload to Aws region using Aliyun S3 command
```

Figure 14 Pseudocode used in Climate@Home project to select best cloud storage bucket and upload data

The Data Download/Upload Matrix

S3 storage could provide efficient and reliable storage. However, it is more expensive than another storage type: glacier. Amazon Glacier is an extremely low-cost storage service that provides highly secure, durable, and flexible storage for data backup and archival (Baron and Kotecha, 2013). Cloud consumers can reliably store their data for as little as \$0.004 per gigabyte per month with Amazon Glacier. Figure 15 lists and compares the AWS S3 storage, Glacier Storage in North Virginia region[1] and Aliyun S3 storage[2]. It shows the price of AWS glacier storage price is 20-25% of S3 storage.

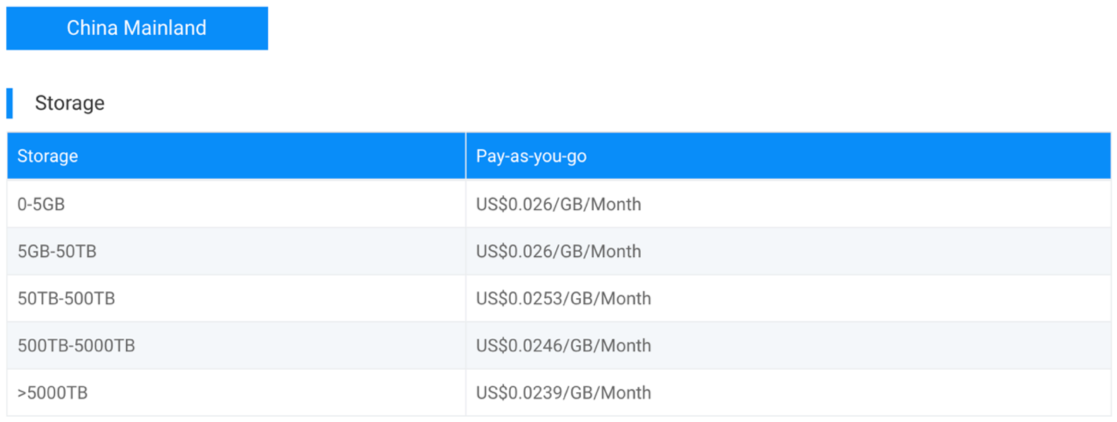


Storage Pricing (varies by region)

Region: US East (N. Virginia)

	Standard Storage	Standard - Infrequent Access Storage †	Glacier Storage
First 50 TB / month	$0.023 per GB	$0.0125 per GB	$0.004 per GB
Next 450 TB / month	$0.022 per GB	$0.0125 per GB	$0.004 per GB
Over 500 TB / month	$0.021 per GB	$0.0125 per GB	$0.004 per GB

(a) AWS S3, glacier price for US-EAST Virginia Region



China Mainland

Storage

Storage	Pay-as-you-go
0-5GB	US$0.026/GB/Month
5GB-50TB	US$0.026/GB/Month
50TB-500TB	US$0.0253/GB/Month
500TB-5000TB	US$0.0246/GB/Month
>5000TB	US$0.0239/GB/Month

(b) Aliyun S3 price for Beijing Region

Figure 15 Comparison of storage price: (a) AWS S3 price and Glacier price in North Virginia Region; (b) Aliyun S3 price in Beijing Region.

[1] https://aws.amazon.com/s3/pricing/
[2] https://www.alibabacloud.com/product/oss#OssStorageTableTrans_HZ

CHAPTER SIX: METHODOLOGY III: AUTO SCALING AND LOAD BALANCING FOR CLIMATE VISUALIZATION

Introduction

After processing the climate models, it is important to visualize the climate data. Climate visualization have the potential to provide such an intuitive and convenient method for scientists to access climate data, discover the relationships between various climate parameters, and communicate the results across different research communities (Sun et al. 2012).

Climate@Home uses netcdf[1] as the output format for climate simulations. Various software could be used to visualize netcdf outputs. For example, MATLAB[2] and Panpoly[3] desktop version could be used to visualize netcdf locally. Geoserver[4] and ncWMS[5] could be used to publish netcdf online. Figure 16 shows precipitation monthly visualization map of E4M20a_000006 from December 1949 to November 1950 using Panpoly. Scientists can analyze the precipitation changes from the visualization results. Figure 17 shows the visualization results for 6 different configurations in Cliamte@Home Project (from E4M20a_000001.R to E4M20a_000006.R). Since the 7 parameters used in

[1] https://www.unidata.ucar.edu/software/netcdf/
[2] https://www.mathworks.com/products/matlab.html
[3] https://www.giss.nasa.gov/tools/panoply/
[4] http://geoserver.org/
[5] https://reading-escience-centre.github.io/ncwms/

the R file have different values, the visualization shows the output differences between them.

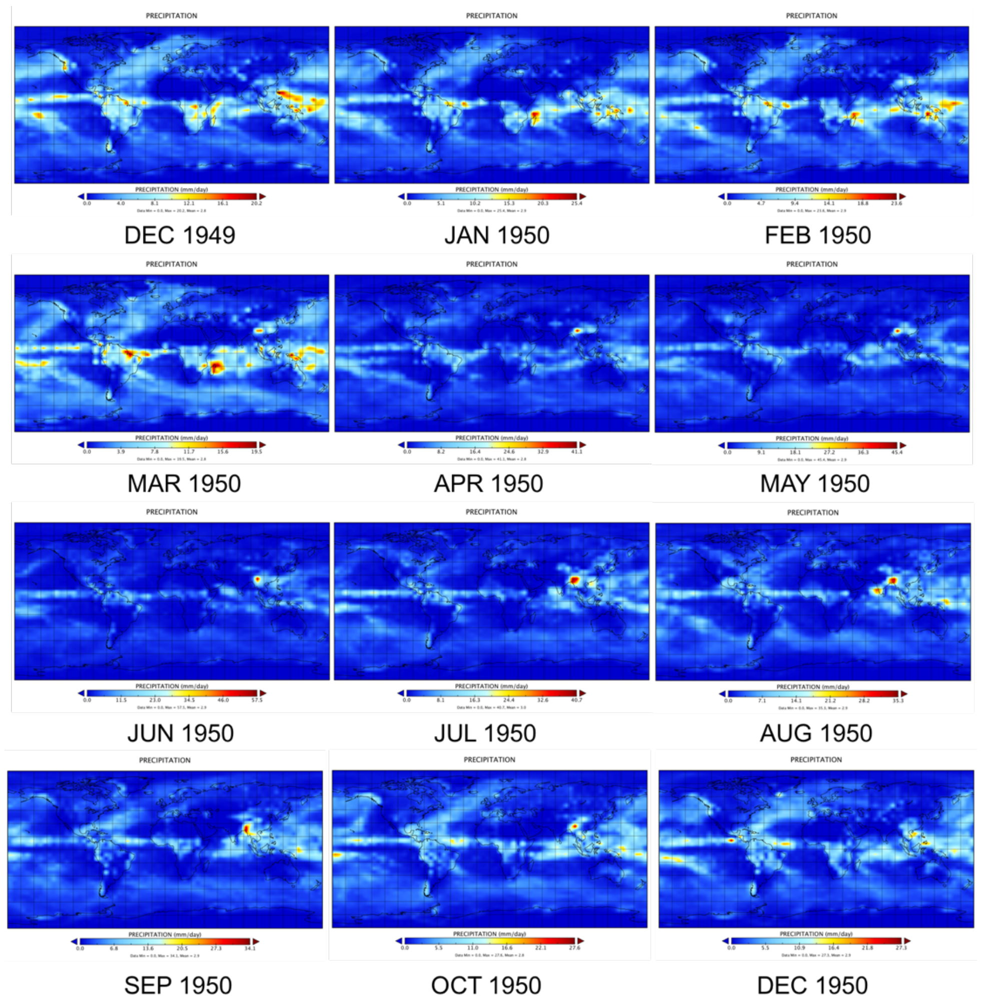


Figure 16 Precipitation monthly visualization map of E4M20a_000006

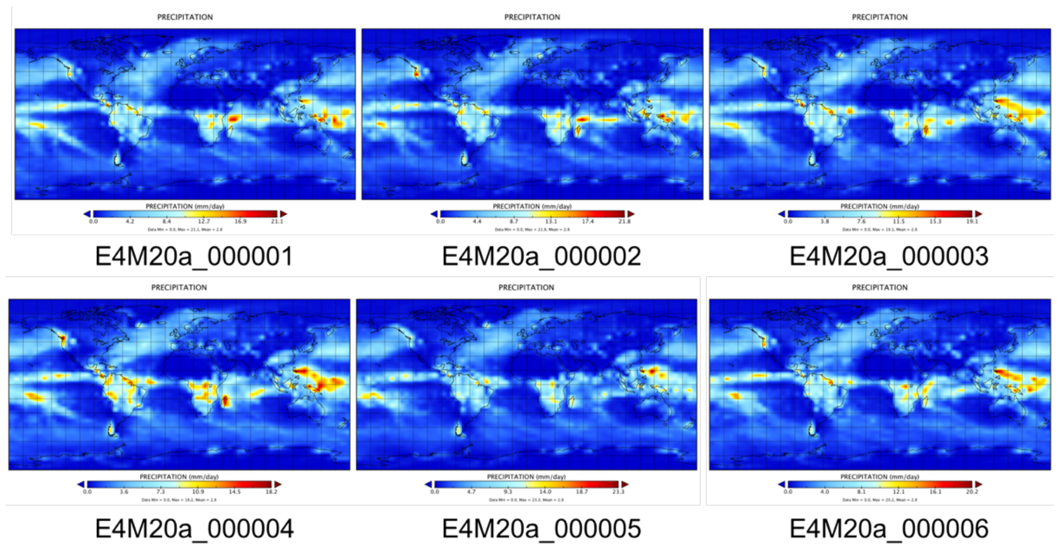


Figure 17 Precipitation monthly visualization map of Decebmer, 1949 of E4M20a_000001 to E4M20a_000006

Netcdf files could be published online to share to other scientists and users. Figure 18 shows the precipitation WMS layer of E4M20a_000006 published using ncWMS. Web Map Service (WMS) provides a GetCapbilities function to describe the capabilities including the layers, the description, the coordinates in the WMS. WMS includes a GetMap function as well to get the map based on requirements (such as bounding box, output format like png, tiff etc.). In addition, WMS could have a GetFeatureInfo option to query the information about specific point, line or polygon in the Web Map.

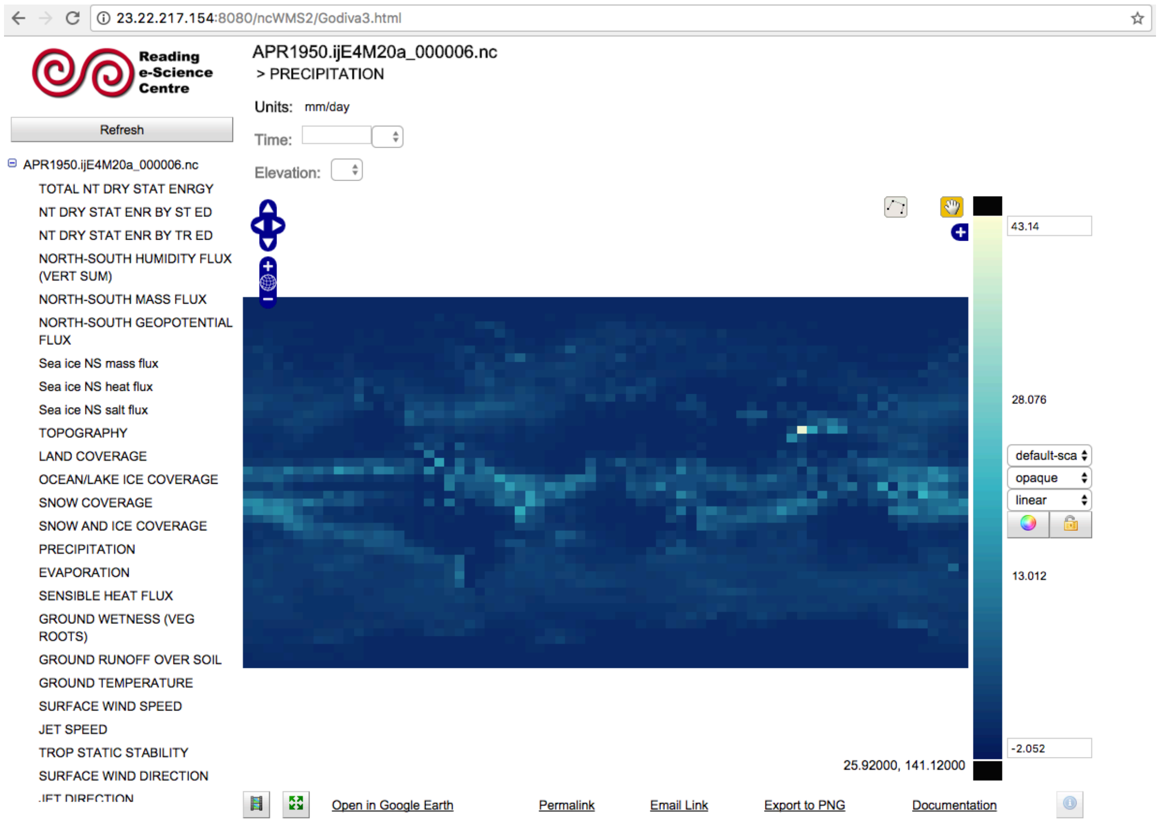
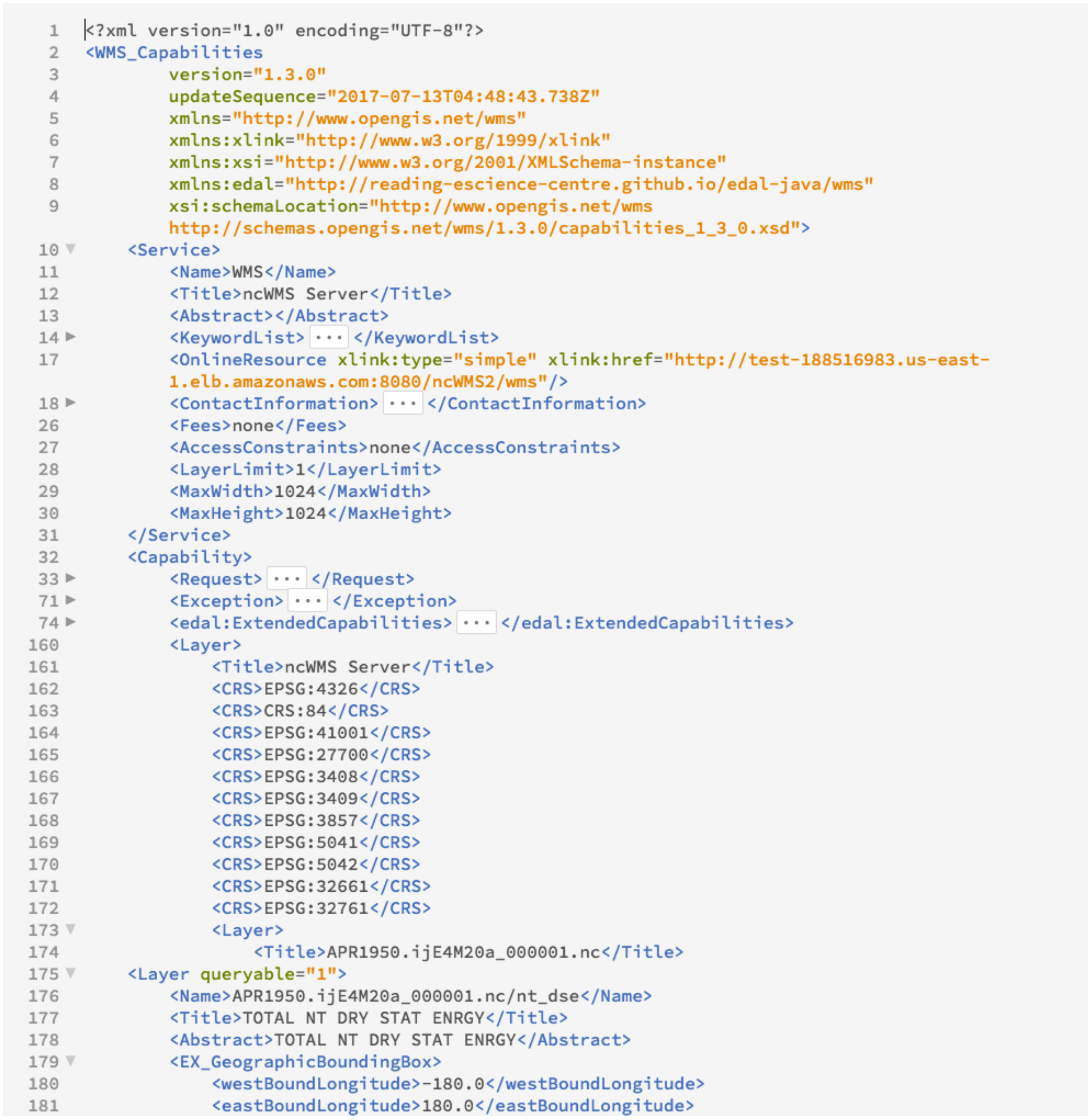


Figure 18 Using ncWMS to publish netcdf file

To use WMS, users always need to send a HTTP Get requests to get the capabilities first. Then analyze the capabilities and to send GetMap request to get online map. At last, users could send GetFeatureInfo request to get the attributes of Point Of Interect (POI). Figure 19 shows the GetCapabilities response of a WMS published by ncWMS.

When many concurrent requests are sending to the ncWMS server, it may cause some traffic and then the server will response slowly.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<WMS_Capabilities
        version="1.3.0"
        updateSequence="2017-07-13T04:48:43.738Z"
        xmlns="http://www.opengis.net/wms"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:edal="http://reading-escience-centre.github.io/edal-java/wms"
        xsi:schemaLocation="http://www.opengis.net/wms
        http://schemas.opengis.net/wms/1.3.0/capabilities_1_3_0.xsd">
    <Service>
        <Name>WMS</Name>
        <Title>ncWMS Server</Title>
        <Abstract></Abstract>
        <KeywordList>···</KeywordList>
        <OnlineResource xlink:type="simple" xlink:href="http://test-188516983.us-east-
        1.elb.amazonaws.com:8080/ncWMS2/wms"/>
        <ContactInformation>···</ContactInformation>
        <Fees>none</Fees>
        <AccessConstraints>none</AccessConstraints>
        <LayerLimit>1</LayerLimit>
        <MaxWidth>1024</MaxWidth>
        <MaxHeight>1024</MaxHeight>
    </Service>
    <Capability>
        <Request>···</Request>
        <Exception>···</Exception>
        <edal:ExtendedCapabilities>···</edal:ExtendedCapabilities>
        <Layer>
            <Title>ncWMS Server</Title>
            <CRS>EPSG:4326</CRS>
            <CRS>CRS:84</CRS>
            <CRS>EPSG:41001</CRS>
            <CRS>EPSG:27700</CRS>
            <CRS>EPSG:3408</CRS>
            <CRS>EPSG:3409</CRS>
            <CRS>EPSG:3857</CRS>
            <CRS>EPSG:5041</CRS>
            <CRS>EPSG:5042</CRS>
            <CRS>EPSG:32661</CRS>
            <CRS>EPSG:32761</CRS>
            <Layer>
                <Title>APR1950.ijE4M20a_000001.nc</Title>
    <Layer queryable="1">
        <Name>APR1950.ijE4M20a_000001.nc/nt_dse</Name>
        <Title>TOTAL NT DRY STAT ENRGY</Title>
        <Abstract>TOTAL NT DRY STAT ENRGY</Abstract>
        <EX_GeographicBoundingBox>
            <westBoundLongitude>-180.0</westBoundLongitude>
            <eastBoundLongitude>180.0</eastBoundLongitude>
```

Figure 19 GeCapabilities response for WMS in ncMWS

Load Balancing and Auto Scaling for Visualization

Using Load Balancing to distribute incoming WMS requests across multiple cloud instances to improve the performance; Using Auto Scaling to help to maintain climate visualization availability and allows climate scientists to dynamically scale the cloud capacity. Figure 20 shows the architecture used in Climate@Home to utilize auto scaling and load balancing.

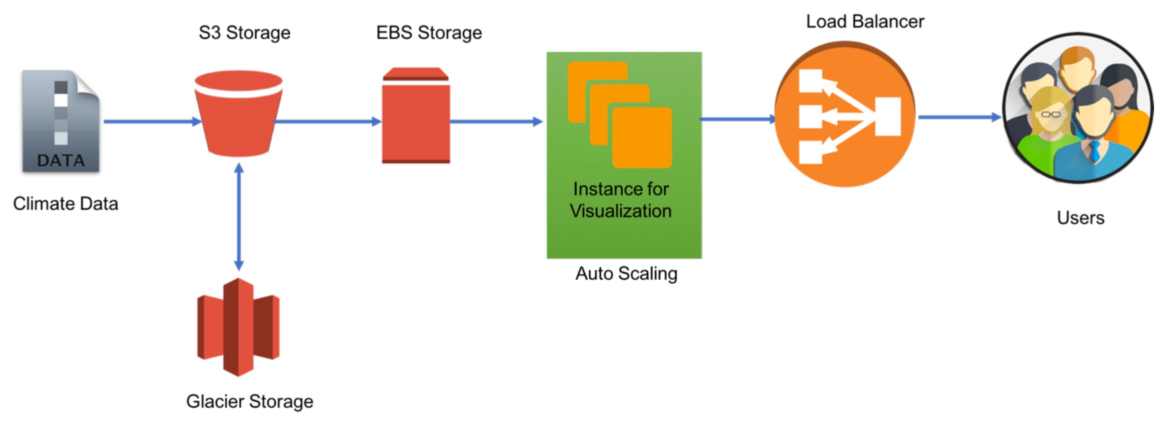
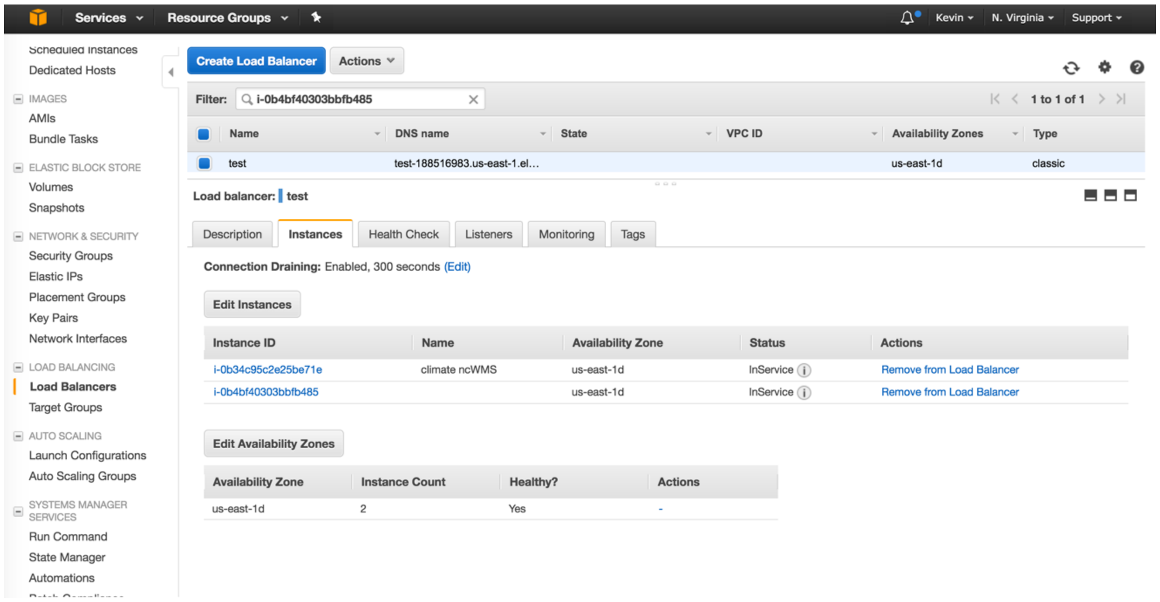


Figure 20 Architecture to use load balancing and auto scaling for visualization

In this architecture, climate outputs may be saved in S3 storage. Then scientists launch an instance to install ncWMS. In the installation, apache tomcat and java need to be installed for the ncWMS. In addition, the firewall needs to be configured to allow incoming traffic (Climate@Home Version 2 uses port 8080). At last, an automatically running script needs to be added in the instance to startup tomcat when the instance is rebooted. Then using the “building image” function provided by cloud computing, scientists could build an image for the server. Then Scientists could start to build load balancer and auto scaling group.



Filter: i-0b4bf40303bbfb485

Name	DNS name	State	VPC ID	Availability Zones	Type
test	test-188516983.us-east-1.el...			us-east-1d	classic

Load balancer: test

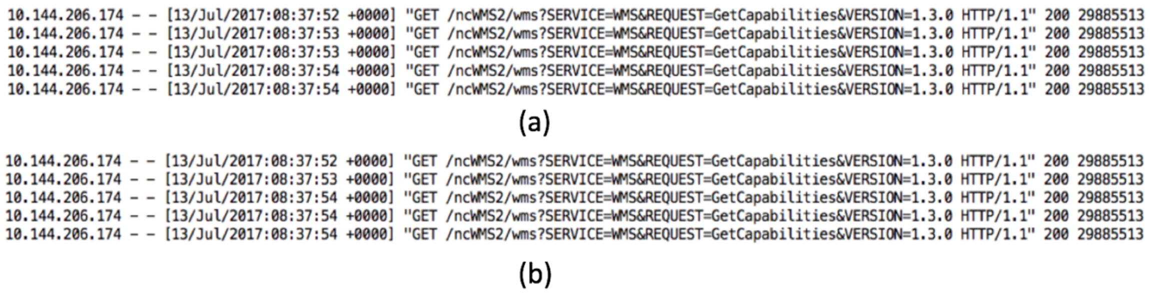
Connection Draining: Enabled, 300 seconds (Edit)

Instance ID	Name	Availability Zone	Status	Actions
i-0b34c95c2e25be71e	climate ncWMS	us-east-1d	InService	Remove from Load Balancer
i-0b4bf40303bbfb485		us-east-1d	InService	Remove from Load Balancer

Availability Zone	Instance Count	Healthy?	Actions
us-east-1d	2	Yes	-

Figure 21 Load Balancer used in Climate@Home for two instances

Figure 21 shows load balancer in AWS cloud which has two visualization instances for Climate@Home project. The load balancer has a listener to listen the load balancer port 8080 and the instance port 8080. When multiple users request the WMS resources through HTTP 8080 port, the load balancer can distribute the requests to two instances (figure 22).



```
10.144.206.174 - - [13/Jul/2017:08:37:52 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:53 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:53 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:54 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:54 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
```

(a)

```
10.144.206.174 - - [13/Jul/2017:08:37:52 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:53 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:54 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:54 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
10.144.206.174 - - [13/Jul/2017:08:37:54 +0000] "GET /ncWMS2/wms?SERVICE=WMS&REQUEST=GetCapabilities&VERSION=1.3.0 HTTP/1.1" 200 29885513
```

(b)

Figure 22 Load Balancer distributes WMS GetCapabilities requests to two instances.

Since load balancer can distribute the traffic to two instances, load balancer can significantly improve the performance of the WMS server since two instances are combined to support the WMS functions. Figure 23 shows the comparison of response time of concurrent GeCapabilities requests using one instance and two instances. With the load balancer of two instances, the response time decrease to almost half of the response time of one machine. It means the load balancer improve the WMS service about 1 time.

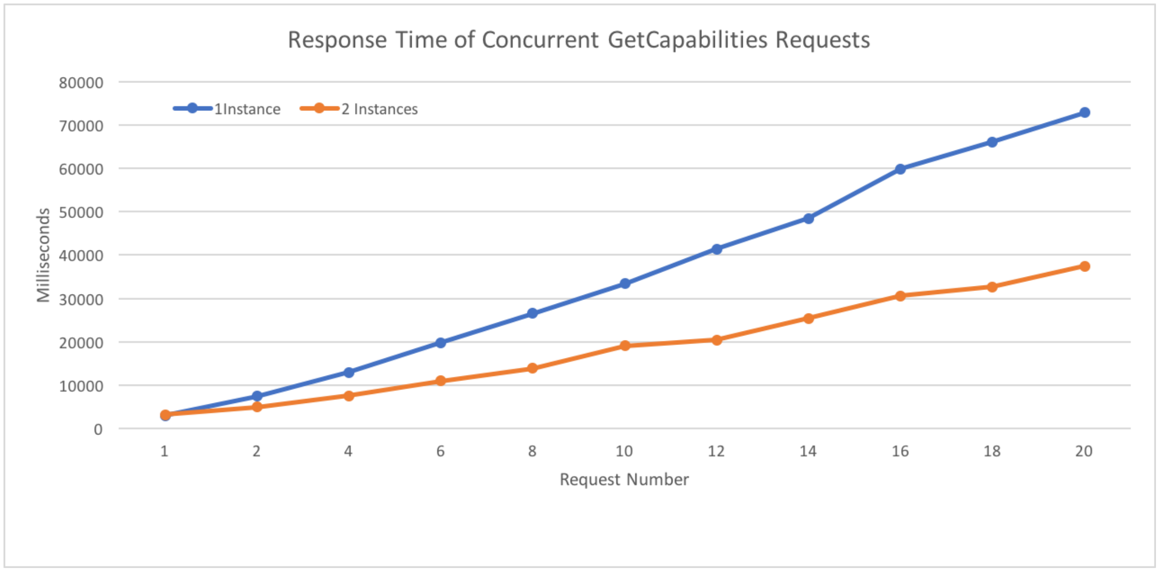
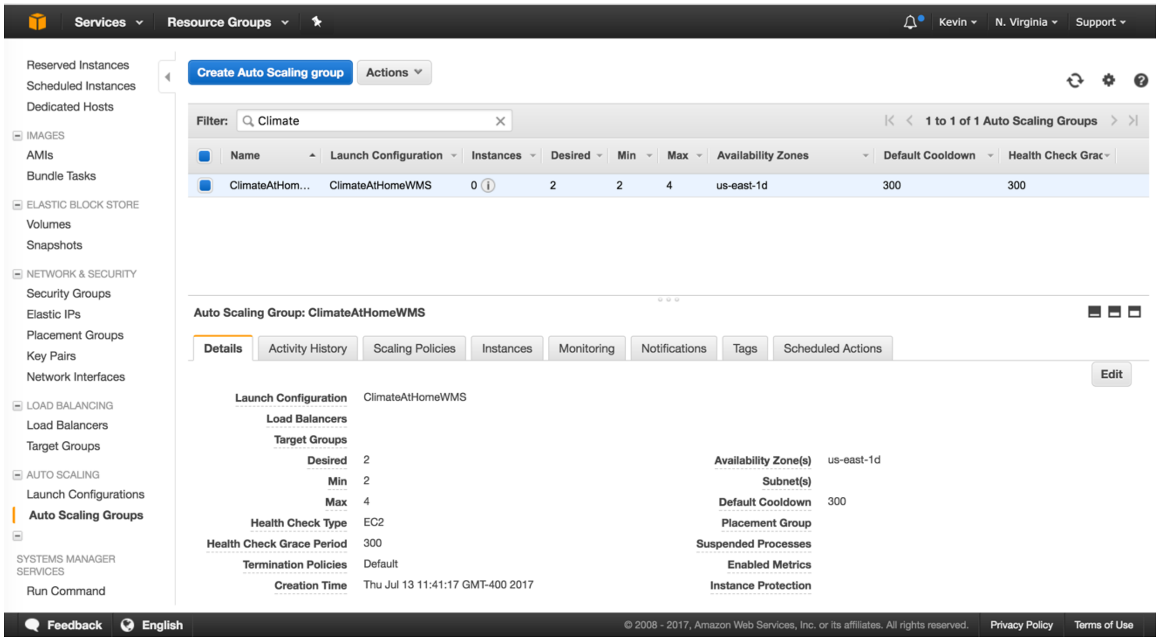


Figure 23 Load Balancer distributes WMS GetCapabilities requests to two instances.

Auto scaling is another way cloud computing provides to improve the WMS service performance and save the cost. Figure 24 shows the auto scaling group used in Climate@Home project. The Climate@Home WMS servicer image is used in this auto scaling group. The group has an auto scaling policy which monitor the incoming network. If the average incoming network is larger than 5000, the group will automatically scale up machines to support the traffic. The group can support 4 virtual machines at most. If the average incoming is less than 5000, the group will automatically terminate machines to save cost.



Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
ClimateAtHom…	ClimateAtHomeWMS	0	2	2	4	us-east-1d	300	300

Auto Scaling Group: ClimateAtHomeWMS

Launch Configuration	ClimateAtHomeWMS		
Load Balancers			
Target Groups			
Desired	2	Availability Zone(s)	us-east-1d
Min	2	Subnet(s)	
Max	4	Default Cooldown	300
Health Check Type	EC2	Placement Group	
Health Check Grace Period	300	Suspended Processes	
Termination Policies	Default	Enabled Metrics	
Creation Time	Thu Jul 13 11:41:17 GMT-400 2017	Instance Protection	

Figure 24 Auto scaling group used in Climate@Home project

CHAPTER SEVEN: EXPERIMENT & RESULT

Using volunteer computing and virtualization technology, the first version of Climate@Home Version 1 was launched on the website http://climateathome.org/climateathome in November 2012 and was terminated in September 2014. Climate@Home Version 2 was launched on the website https://climateathome.info/climateathome using hybrid computing infrastructure in April 2017.

Running Status

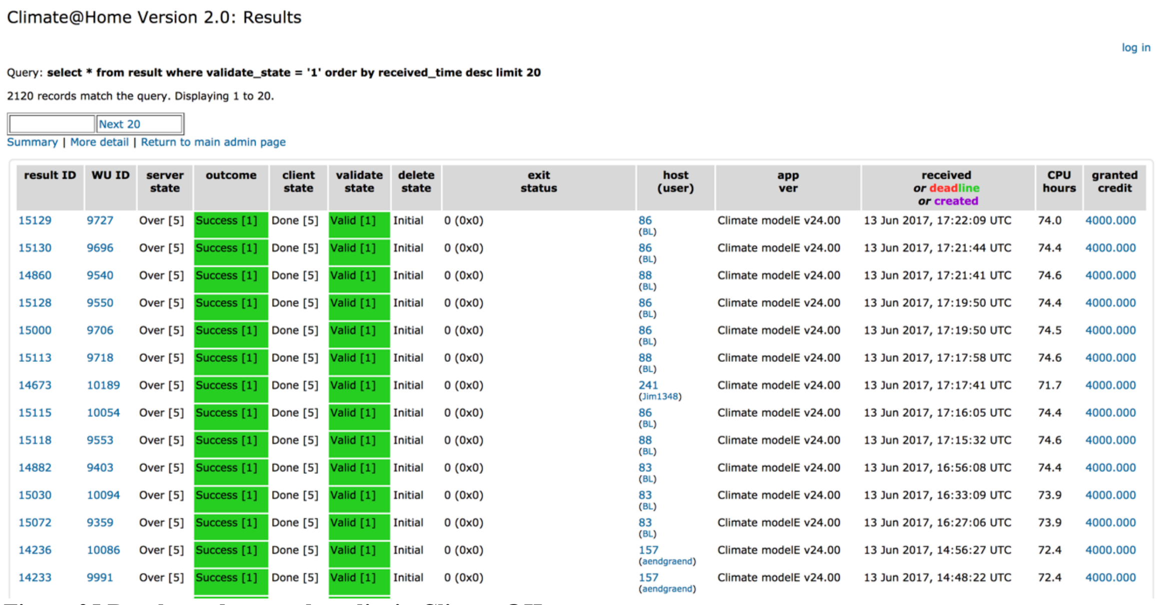
Of the activated hosts in Climate@Home project, Windows machines account for 92%, Linux machines for 3.7% and Apple machines for 4.3%. Windows machines have a total of 3399 cores, which account for 92.1% FLOPS. Linux machines have a total of 218 cores, which account for 3.3% FLOPS. And Apple machines have 137 core, which account for 4.6% FLOPS (Table 7).

Table 7 Statistics of volunteer hosts

OS	Number	CPU Core	GFLOPS
Windows	544	3399	1543.5
Linux	22	218	56
Apple	25	137	76.4
Total	591	3754	1675.9

Credits

Using validator, the Climeate@Home validate simulation tasks based on three results. Then using the New Create System, Cliatme@Home can grant credits to results. Figure 25 shows the 14 results submitted by different hosts, each result is granted 4000 credits.



Climate@Home Version 2.0: Results

log in

Query: **select * from result where validate_state = '1' order by received_time desc limit 20**

2120 records match the query. Displaying 1 to 20.

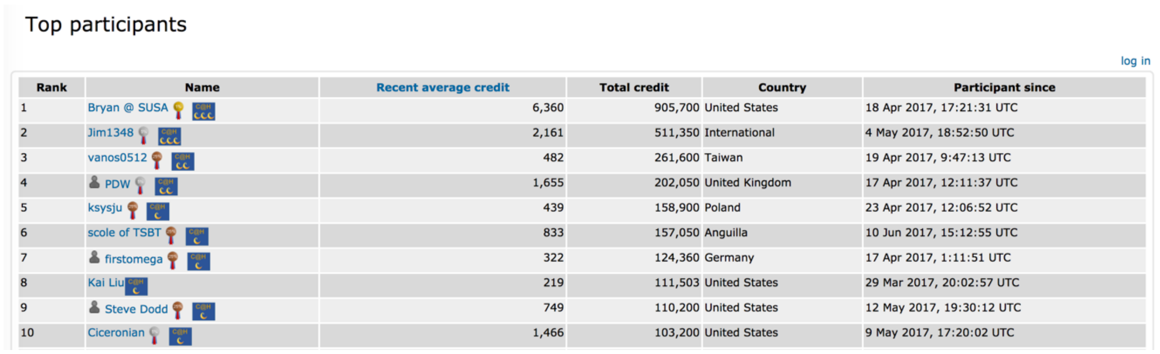
Next 20

Summary | More detail | Return to main admin page

result ID	WU ID	server state	outcome	client state	validate state	delete state	exit status	host (user)	app ver	received *or* deadline *or* created	CPU hours	granted credit
15129	9727	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	86 (BL)	Climate modelE v24.00	13 Jun 2017, 17:22:09 UTC	74.0	4000.000
15130	9696	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	86 (BL)	Climate modelE v24.00	13 Jun 2017, 17:21:44 UTC	74.4	4000.000
14860	9540	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	88 (BL)	Climate modelE v24.00	13 Jun 2017, 17:21:41 UTC	74.6	4000.000
15128	9550	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	86 (BL)	Climate modelE v24.00	13 Jun 2017, 17:19:50 UTC	74.4	4000.000
15000	9706	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	86 (BL)	Climate modelE v24.00	13 Jun 2017, 17:19:50 UTC	74.5	4000.000
15113	9718	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	88 (BL)	Climate modelE v24.00	13 Jun 2017, 17:17:58 UTC	74.6	4000.000
14673	10189	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	241 (Jim1348)	Climate modelE v24.00	13 Jun 2017, 17:17:41 UTC	71.7	4000.000
15115	10054	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	86 (BL)	Climate modelE v24.00	13 Jun 2017, 17:16:05 UTC	74.4	4000.000
15118	9553	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	88 (BL)	Climate modelE v24.00	13 Jun 2017, 17:15:32 UTC	74.6	4000.000
14882	9403	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	83 (BL)	Climate modelE v24.00	13 Jun 2017, 16:56:08 UTC	74.4	4000.000
15030	10094	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	83 (BL)	Climate modelE v24.00	13 Jun 2017, 16:33:09 UTC	73.9	4000.000
15072	9359	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	83 (BL)	Climate modelE v24.00	13 Jun 2017, 16:27:06 UTC	73.9	4000.000
14236	10086	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	157 (aendgraend)	Climate modelE v24.00	13 Jun 2017, 14:56:27 UTC	72.4	4000.000
14233	9991	Over [5]	Success [1]	Done [5]	Valid [1]	Initial	0 (0x0)	157 (aendgraend)	Climate modelE v24.00	13 Jun 2017, 14:48:22 UTC	72.4	4000.000

Figure 25 Results and granted credits in Climate@Home

Climate@Home can compare credits based on hosts and users. Figure 26 shows the top 10 users with high credits in Climate@Home Version 2.



Top participants

log in

Rank	Name	Recent average credit	Total credit	Country	Participant since
1	Bryan @ SUSA	6,360	905,700	United States	18 Apr 2017, 17:21:31 UTC
2	Jim1348	2,161	511,350	International	4 May 2017, 18:52:50 UTC
3	vanos0512	482	261,600	Taiwan	19 Apr 2017, 9:47:13 UTC
4	PDW	1,655	202,050	United Kingdom	17 Apr 2017, 12:11:37 UTC
5	ksysju	439	158,900	Poland	23 Apr 2017, 12:06:52 UTC
6	scole of TSBT	833	157,050	Anguilla	10 Jun 2017, 15:12:55 UTC
7	firstomega	322	124,360	Germany	17 Apr 2017, 1:11:51 UTC
8	Kai Liu	219	111,503	United States	29 Mar 2017, 20:02:57 UTC
9	Steve Dodd	749	110,200	United States	12 May 2017, 19:30:12 UTC
10	Ciceronian	1,466	103,200	United States	9 May 2017, 17:20:02 UTC

Figure 26 Top top 10 participants in Climate@Home Project

Budget Cost Comparison

Figure 27 compared the budget cost comparison using cloud computing and volunteer computing. The blue bar is to run 300 5-year simulation tasks and the red bar is to run 300 10-year simulation tasks. With different instance types, the cost may be different. However, volunteer computing can save significantly to for both 5-year and 10-year simulations.

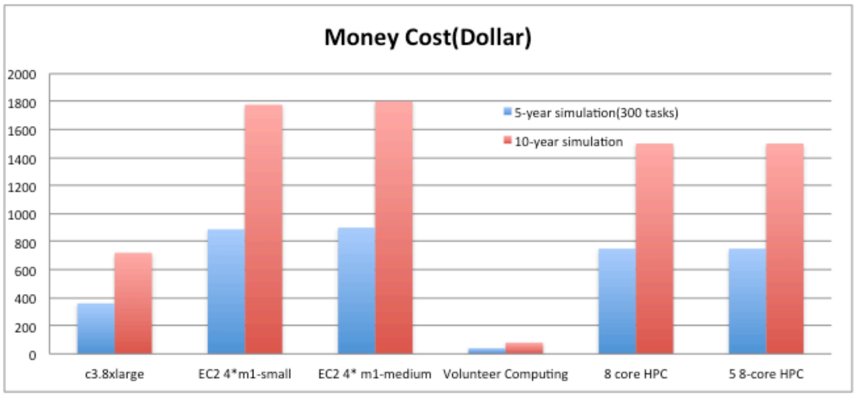


Figure 27 Budget cost comparison for model processing (exclude cloud storage and visualization)

CHAPTER EIGHT: CONCLUSIONS AND FUTURE RESEARCH

Summary and Concluding Remarks

Climate@Home is the first volunteer computing project using virtualization technology in climate domain. This thesis reports the research to integrate and optimize volunteer computing and virtualization technology for climate simulation. The most salient finding from this research are as follows:

- Compared to expensive supercomputer and cloud computing, the volunteer computing could provide enormous computational resources with essentially no cost. In addition, more volunteers and hosts are registered in the Climate@Home project, and consequentially computational resources are getting more powerful.
- Virtualization technology could improve the development of climate application in volunteer computing project. Generally, 32 bit virtual machines can run on both 32 bit and 64 bit machines. By packaging climate models, data, operation system, libraries and tools into a 32 bit virtual image, scientists only need to develop one version of their model rather than transferring their code to different platforms. In addition, deployers only need to update one virtual image if there is new version of model and then use the new virtual image to upload for different platforms. Comparing the traditional method which needs to update codes for different platforms, it saves a significant amount of time for the new version release. In addition, the virtual image saves the running status to a snapshot and it enables climate models to run again from the breakpoint once they

have crashed. These advantages of virtual technologies bring significant convenience to develop and test climate models.

- PUM can upload the climate model outputs periodically. On one hand, it saves time to upload outputs since uploading occurs when models are running rather than uploading them after the model finishes. Conversely, it reduces the network stress by uploading small monthly outputs incrementally.
- NCS improves the volunteers' incentives to participate. By using the new credit system, it is fairer to grant credits based on number of completed monthly tasks and the faster machine will get more credits as they complete more tasks.
- Using cloud Simple Storage Service provided by leading cloud providers to develop a global replication storage to distribute cloud models and data to global volunteers.
- Using Load Balancing to distribute incoming WMS requests across multiple cloud instances to improve the performance; Using Auto Scaling to help to maintain climate visualization availability and allows climate scientists to dynamically scale the cloud capacity.

Future Research

In spite of these advantages, this infrastructure has some shortcomings and is not suitable for all the climate applications. The principal shortcomings are as follows:

- There is no guarantee that all simulation tasks will be finished on time since volunteers may have a variety of site-specific issues such as hardware error, download error or abortion of the tasks. Although the virtualization technology enable a user to run climate models on virtual hardware, the models still can not continue once physical hardware problem occurs. Most download errors occur because of network problems. Due to the long running time of the climate simulation tasks, volunteers may abort some tasks to release computing resources for other projects. Although volunteer computing could provide a replication to assign each job to some different hosts to increase the reliability to run climate projects, it may waste volunteers' computing resources since only those unsuccessful tasks need to be assigned again.

- The infrastructure can not support real-time climate tasks. These tasks need that the computers can get the real-time data rapidly and execute the climate tasks as fast as possible. However, the virtual image and data transfer in the volunteer computing relies on network and volunteer's machine. The network speed is far lower than transfer data in the supercomputer. In addition, the machine hardware differs widely between different hosts and it causes those hosts finish tasks in different time.

- The infrastructure can not support applications which need parallel processing, such as Message Passing Interface (MPI) applications. Parallel processing emphasizes the feasible exploitation of available concurrency in a computational process. By the decomposition, parallel processing partitions the large-scale computational problems to small sub-domains. Parallel processing requires frequently and rapidly exchanging data between sub-domains.

More research and advancements in relevant fields could optimize the volunteer computing and virtualization for climate domain and these are listed below.

- A better task scheduler to improve the reliability of the climate simulation tasks from volunteers. The scheduler should be able to get the success rate of each host to run the simulation tasks and should use some optimization algorithm to assign the tasks to different hosts based on the success rate to improve the reliability. It will assign the unsuccessful tasks to different hosts rather than assign each task to different hosts.

REFERENCES

Aji, A., Wang, F., Vo, H., Lee, R., Liu, Q., Zhang, X., & Saltz, J. (2013). Hadoop gis: a high performance spatial data warehousing system over mapreduce. *Proceedings of the VLDB Endowment*, 6(11), 1009-1020.

Anderson, D., et al. (2002). SETI@ home: an experiment in public-resource computing. *Communications of the ACM*, 45(11), 56-61.

Anderson, D. (2004). Boinc: A system for public-resource computing and storage. *In: Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on* (pp. 4-10). IEEE.

Anderson, D., and Fedak, G. (2006). The computational and storage potential of volunteer computing. *In: Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on* (Vol. 1, pp. 73-80). IEEE.

Arakawa, A., & Lamb, V. R. (1977). Computational design of the basic dynamical processes of the UCLA general circulation model. *Methods in computational physics*, 17, 173-265.

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., ... & Warfield, A. (2003, October). Xen and the art of virtualization. In *ACM SIGOPS operating systems review* (Vol. 37, No. 5, pp. 164-177). ACM.

Baron, J., & Kotecha, S. (2013). Storage options in the aws cloud. *Amazon Web Services*, Washington DC, Tech. Rep.

Boutcher, D., & Chandra, A. (2008). Practical techniques for purging deleted data using liveness information. *ACM SIGOPS Operating Systems Review*, 42(5), 85-94.

Brooks, W., Aftosmis, M., Biegel, B., Biswas, R., Ciotti, R., Freeman, K., ... & Thigpen, W. (2005). Impact of the Columbia supercomputer on NASA science and engineering applications. *Lecture notes in computer science*, 3741, 293.

Caupp, C., Brock, J., & Runke, H. (1991). *Application of the dynamic stream simulation and assessment model (DSSAM III) to the Truckee River below Reno, Nevada: Model formulation and program description.* Raipd Creek Water Works.

Christensen, C., Aina T., and Stainforth, D., (2005). The challenge of volunteer computing with lengthy climate model simulations. *In e-Science and Grid Computing, 2005. First International Conference on* (pp. 8-pp). IEEE.

Coleman, D. J., Georgiadou, Y., & Labonte, J. (2009). Volunteered geographic information: The nature and motivation of produsers. *International Journal of Spatial Data Infrastructures Research*, 4(1), 332-358.

Coleman, D. J., Sabone, B., & Nkhwanana, N. (2010). Volunteering Geographic Information to Authoritative Databases: Linking Contributor Motivations to Program Effectiveness. *Geomatica*, 64(1), 383–396.

Cullen, M. (1993). The unified forecast/climate model. *Meteorological Magazine*, 122(1449), 81-94.

Dash, P., (2013). *Getting Started with Oracle VM VirtualBox*. Packt Publishing Ltd.

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. Communications of the ACM, 51(1), 107-113.

Del Genio, D., & Yao, S. (1993). Efficient cumulus parameterization for long-term climate studies: The GISS scheme. *The Representation of Cumulus Convection in Numerical Models, Meteor.* Monogr, 46, 181-184.

Deutsch, L. P. (1996). *GZIP file format specification version 4.3*.

Dougherty, B., White, J., & Schmidt, D. C. (2012). Model-driven auto-scaling of green cloud computing infrastructure. *Future Generation Computer Systems*, 28(2), 371-378.

Fedak, G., Germain, C., Neri, V., & Cappello, F. (2001). Xtremweb: A generic global computing system. *In Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on* (pp. 582-587). IEEE.

Felter, W., Ferreira, A., Rajamony, R., & Rubio, J. (2015). An updated performance comparison of virtual machines and linux containers. In Performance Analysis of Systems and Software (ISPASS), 2015 IEEE International Symposium on (pp. 171-172). IEEE.

Fink, J. (2014). Docker: a software as a service, operating system-level virtualization framework. Code4Lib Journal, 25.

Gao, S., Li, L., Li, W., Janowicz, K., & Zhang, Y. (2017). Constructing gazetteers from volunteered big geo-data based on Hadoop. Computers, Environment and Urban Systems, 61, 172-186.

Gates, W. L., (1992). AMIP: The atmospheric model intercomparison project. *Bulletin of the American Meteorological Society*, 73(12), 1962-1970.

Goodchild, M. F., Kyriakidis, P., Rice, M., & Schneider, P. (2005). Spatial Webs, Final Report and Position Papers. https://escholarship.org/uc/item/46z721n2

Goodchild, M. F. (2007) Citizens as sensors: The world of volunteered geography. *GeoJournal*, vol. 69, no. 4, pp. 211–221, Dec. 2007.

Gordon, C., Cooper, C., Senior, C. A., Banks, H., Gregory, J. M., Johns, T. C., ... & Wood, R. A. (2000). The simulation of SST, sea ice extents and ocean heat transports in a version of the Hadley Centre coupled model without flux adjustments. *Climate dynamics*, 16(2), 147-168.

Griffith, A., (2002). *GCC: the complete reference*. McGraw-Hill, Inc.

Hansen, J., Russell, G., Rind, D., Stone, P., Lacis, A., Lebedeff, S., ... & Travis, L. (1983). Efficient three-dimensional global models for climate studies: Models I and II. *Monthly Weather Review*, 111(4), 609-662.

Hansen, J., Sato, M., Ruedy, R., Kharecha, P., Lacis, A., Miller, R., ... & Aleinov, I. (2007). Climate simulations for 1880–2003 with GISS modelE. *Climate dynamics*, 29(7-8), 661-696.

Hayes, J. P., Mudge, T. N., Stout, Q. F., Colley, S., & Palmer, J. (1986, August). Architecture of a Hypercube Supercomputer. In *ICPP* (pp. 653-660).

Havnø, K., Madsen, M. N., & Dørge, J. (1995). MIKE 11–a generalized river modelling package. *Computer models of watershed hydrology*, 733-782.

Huang, Q., Yang, C., Nebert, D., Liu, K., & Wu, H. (2010, November). Cloud computing for geosciences: deployment of GEOSS clearinghouse on Amazon's EC2. In *Proceedings of the ACM SIGSPATIAL international workshop on high performance and distributed geographic information systems* (pp. 35-38). ACM.

Lee, Y., & Lee, Y. (2013). Toward scalable internet traffic measurement and analysis with hadoop. *ACM SIGCOMM Computer Communication Review*, 43(1), 5-13.

Li, Z. Yang, C., Jin, B., Yu, M., Liu, K., Sun, M., & Zhan, M. (2015). Enabling big geoscience data analytics with a cloud-based, MapReduce-enabled and service-oriented workflow framework. *PloS one*, 10(3), e0116781.)

Liu, K., Yang, C., Li, W., Li, Z., Wu, H., Rezgui, A., & Xia, J. (2011, June). The GEOSS clearinghouse high performance search engine. In *Geoinformatics, 2011 19th International Conference* on (pp. 1-4). IEEE.

Karl, T. R., Knight, R. W., Gallo, K. P., Peterson, T. C., Jones, P. D., Kukla, G., ... & Charlson, R. J. (1993). A new perspective on recent global warming: asymmetric trends of daily maximum and minimum temperature. *Bulletin of the American Meteorological Society*, 74(6), 1007-1023.

Koch, D., & Hansen, J. (2005). Distant origins of Arctic black carbon: a Goddard Institute for Space Studies ModelE experiment. *Journal of Geophysical Research: Atmospheres* (1984–2012), 110(D4).

Kolb, L., Thor, A., & Rahm, E. (2012, April). Load balancing for mapreduce-based entity resolution. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference* on (pp. 618-629). IEEE.

Larson, S., Snow, C., & Shirts, M. (2002). Folding@ Home and Genome@ Home: Using distributed computing to tackle previously intractable problems in computational biology. *arXiv preprint arXiv:0901.0866.*

Liang, X., Lettenmaier, D. P., Wood, E. F., & Burges, S. J. (1994). A simple hydrologically based model of land surface water and energy fluxes for general circulation models. *Journal of Geophysical Research: Atmospheres* (1984–2012), 99(D7), 14415-14428.

Liu, J., & Abali, B. (2009). Virtualization polling engine (VPE): using dedicated CPU cores to accelerate I/O virtualization. *In Proceedings of the 23rd international conference on Supercomputing (pp. 225-234).* ACM.

Liu, J., Schmidt, G. A., Martinson, D. G., Rind, D., Russell, G., & Yuan, X. (2003). Sensitivity of sea ice to physical parameterizations in the GISS global climate model. *Journal of Geophysical Research: Oceans*, 108(C2).

Liu, K., Yang, C., Li, W., Li, Z., Wu, H., Rezgui, A., & Xia, J. (2011, June). The GEOSS clearinghouse high performance search engine. In *Geoinformatics, 2011 19th International Conference on* (pp. 1-4). IEEE.

Lombraña González, D., Karneyeu, A., Buncic, P., Segal, B., Grey, F., Blomer, J., ... & Marquina, M. (2012). Virtual machines & volunteer computing: Experience from LHC@ Home: Test4Theory project. *PoS*, 036.

Omer, M., Nilchiani, R., & Mostashari, A. (2009). Measuring the resilience of the trans-oceanic telecommunication cable system. *IEEE Systems Journal*, 3(3), 295-303.

Marr, B. (2015). Big Data: *Using SMART Big Data. Analytics and Metrics To Make Better Decisions and Improve Performance*. Atrium: Wiley.

Mell, P., & Grance, T. (2009). The NIST definition of cloud computing. *National Institute of Standards and Technology*, 53(6), 50.

Menemenlis, D., Hill, C., Adcrocft, A., Campin, J. M., Cheng, B., Ciotti, B., ... & Lee, T. (2005). NASA supercomputer improves prospects for ocean climate research. *Eos, Transactions American Geophysical Union*, 86(9), 89-96.

Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014(239), 2.

Metcalf, M., Reid, J. K., & Cohen, M. (2004). *Fortran 95/2003 Explained* (Vol. 416). Oxford: Oxford University Press.

Paez Wulff, F. I. (2014). Recruitment, Training, and Social Dynamics in Geo-Crowdsourcing for Accessibility (Master's thesis, George Mason University, Fairfax, Virginia).

Palmer, T., et al. (2005). Representing model uncertainty in weather and climate prediction. *Annu. Rev. Earth Planet*. Sci., 33, 163-193.

Rew, R., & Davis, G. (1990). NetCDF: an interface for scientific data access. *Computer Graphics and Applications*, 10(4), 76-82.

Rice, M. T., Paez, F. I., Rice, R. M., Ong, E. W., Qin, H., Seitz, C. R., Medina, R. M. (2014). Quality assessment and accessibility applications of crowdsourced geospatial data: A report on the development and extension of the George Mason University Geocrowdsourcing Testbed (Annual No. BAA: #AA10-4733, Contract: # W9132V-11-P-0011) (p. 91). Fairfax, VA: George Mason University.

Rice, M. T., Curtin, K. M., Pfoser, D., Rice, R. M., Fuhrmann, S., Qin, H., Vese, R. D., Ong, E. W., Fayne, J. V., Paez, F. I., Seitz, C. R., Rice, M. A., Yu, M., Ober, S., Rice, C. A. (2015). Social Moderation and Dynamic Elements in Crowdsourced Geospatial Data: A Report on Quality Assessment, Dynamic Extensions and Mobile Device Engagement in the George Mason University Geocrowdsourcing Testbed. George Mason University Fairfax United States. Retrieved from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=AD100194
3

Rice, M. T., Paez, F. I., Mulhollen, A. P., Shore, B. M., & Caldwell, D. R. (2012). Crowdsourced Geospatial Data: A report on the emerging phenomena of crowdsourced and user-generated geospatial data (Annual No. AA10-4733). Fairfax, VA: George Mason University. http://www.dtic.mil/dtic/tr/fulltext/u2/a576607.pdf.

Rice, M. T., Curtin, K. M., Paez, F. I., Seitz, C. R., & Qin, H. (2013). Crowdsourcing to Support Navigation for the Disabled: A Report on the Motivations, Design, Creation and Assessment of a Testbed Environment for Accessibility (US Army Corps of Engineers, Engineer Research and Development Center, US Army Topographic Engineering Center Technical Report, Data Level Enterprise Tools Workgroup No. BAA: #AA10-4733, Contract: # W9132V-11-P-0011) (pp. 1–62). Fairfax, VA: George Mason University. http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA58847 4

Rogelj, J., Meinshausen, M., & Knutti, R. (2012). Global warming under old and new scenarios using IPCC climate sensitivity range estimates. Nature climate change, 2(4), 248-253.).

Sellers, P. J., Mintz, Y. C. S. Y., Sud, Y. E. A., & Dalcher, A. (1986). A simple biosphere model (SiB) for use within general circulation models. *Journal of the Atmospheric Sciences*, 43(6), 505-531.

Shindell, D. T., Faluvegi, G., Unger, N., Aguilar, E., Schmidt, G. A., Koch, D. M., ... & Miller, R. L. (2006). Simulations of preindustrial, present-day, and 2100 conditions in the NASA GISS composition and climate model G-PUCCINI. *Atmospheric Chemistry and Physics*, 6(12), 4427-4459.

Shvachko, K., Kuang, H., Radia, S., & Chansler, R. (2010, May). The hadoop distributed file system. In *Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on* (pp. 1-10). IEEE.

Stainforth, D., Kettleborough, J., Martin, A. P., Simpson, A., Gillis, R., Akkas, A., ... & Allen, M. (2002, November). Climateprediction. net: Design Principles for Publicresource Modeling Research. In *IASTED PDCS* (pp. 32-38).

Stainforth, D. A., Aina, T., Christensen, C., & Collins, M. (2005). Uncertainty in predictions of the climate response to rising levels of greenhouse gases. *Nature*, 433(7024), 403.

Stubbs, J., Moreira, W., & Dooley, R. (2015, June). Distributed systems of microservices using docker and serfnode. In *Science Gateways (IWSG), 2015 7th International Workshop on* (pp. 34-39). IEEE.

Sun, M., Li, J., Yang, C., Schmidt, G. A., Bambacus, M., Cahalan, R., ... & Li, Z. (2012). A web-based geovisual analytical system for climate studies. *Future Internet*, 4(4), 1069-1085.

Taylor K. E., Stouffer, R. J., & Meehl, G. A. (2012). An overview of CMIP5 and the experiment design. *Bulletin of the American Meteorological Society*, 93(4), 485-498)

Electronic Commerce A Managerial Perspective. Prentice-Hall. p. 27. Turban, E., King, D., Lee, J., & Viehland, D. (2008). Chapter 19: Building E-Commerce Applications and Infrastructure. *Electronic Commerce A Managerial Perspective*, 27.

Khaleel, M. A., Johnson, G. M., & Washington, W. M. (2009). *Scientific Grand Challenges: Challenges in Climate Change Science and the Role of Computing at the Extreme Scale* (No. PNNL-18362). Pacific Northwest National Laboratory (PNNL), Richland, WA (US).

Williams, K., Senior, C., & Mitchell, J. (2001). Transient climate change in the Hadley Centre models: The role of physical processes. *Journal of Climate*, 14(12), 2659-2674.

Yang, C., Goodchild, M., Huang, Q., Nebert, D., Raskin, R., Xu, Y., ... & Fay, D., 2011. Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing?. *International Journal of Digital Earth*, 4(4), 305-329.

Yang, C., Raskin, R., Goodchild, M., & Gahegan, M. (2010). Geospatial cyberinfrastructure: past, present and future. *Computers, Environment and Urban Systems*, 34(4), 264-277.

BIOGRAPHY

Kai Liu is a graduate student in the Department of Geography and GeoInformation Sciences in the College of Science at George Mason University. He received his Master of Science from George Mason University in 2014. Previously he was a visiting scholar at the Center of Intelligent Spatial Computing for Water/Energy Science, and worked for 4 years at Heilongjiang Bureau of Surveying and mapping in China. His formal education was acquired at Wuhan University, China, BA Geographic Information Science. His research focuses on Geospatial semantics, volunteer computing and spatial cloud computing.