2016 IEEE 40th Annual Computer Software and Applications Conference

Teaching Secure Cloud Computing Concepts with Open Source CloudSim Environment

Yong Shi, Dan Lo and Kai Qian

Department of Computing and
Software Engineering

Kennesaw State University

Marietta, GA, USA

yshi5,dlo2,kqian@kennesaw.edu

***Abstract*—Cloud Computing is growing very fast and many universities are offering Cloud Computing courses on undergraduate and graduate levels, however, most of them are seminar type courses without hands-on learning experience for students. Hands-on learning, i.e. learning by doing is very crucial to help students understand better on the concepts of cloud computing such as security, task scheduling and network topology. In this paper, we propose to enhance Cloud Computing education with open source simulation tools such as CloudSim. We apply CloudSim to create labware for students to gain hands-on experience and understand different aspects of Cloud Computing more efficiently and effectively. The preliminary feedback on student learning outcomes is very encouraging.**

Keywords-Cloud Computing; Simulation Tools; CloudSim; Security; Task Scheduling; Network Topology; MapReduce

I. INTRODUCTION

A cloud infrastructure provides a framework to manage scalable, reliable, on-demand access to applications. In fact, a cloud is the backend to a lot of our mobile applications while we might not be aware of it. There are many definitions of Cloud Computing. For example, according to Amazon Web Services, one of the pioneer Cloud Services, Cloud Computing refers to the on-demand delivery of IT resources and applications via the Internet with pay-as-you-go pricing [1]. It enables industry companies to consume compute resources as a utility rather than having to build and maintain computing infrastructures.

Cloud Computing gives the users the illusion of infinite computing resources available on demand, so the users do not need to plan far ahead for resource provisioning. It also eliminates an up-front commitment, so start-up companies can start small and gradually increase the hardware resources when needed in the future. The cloud customers are able to pay the costs of computing resources on a short-term basis as needed, and once they are done, they can immediately release the computing resources.

Cloud Computing provides various layers of services. Three most commonly used ones are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [2, 8]. A large amount of companies provide cloud services on IaaS, PaaS and SaaS layers. IaaS, such as Amazon EC2, is the delivery of technology infrastructure as an on demand scalable service. IaaS shares the physical hardware, where consumers can provision computing resources within provider's infrastructure upon which they can deploy and run arbitrary software, including OS and applications. PaaS, such as Google App Engine, shares the application framework, where consumers can create custom applications using programming tools supported by the provider and can deploy them onto the provider's cloud infrastructure. Different from IaaS and PaaS, SaaS, such as Salesforce.com, shares the entire software stack, where consumers use provider’s applications running on provider's cloud infrastructure.

Cloud Computing is one of the critical emerging technologies nowadays, and Clouds form a major industry thrust that keeps growing more and more investment. Therefore, there are many opportunities for new jobs in Cloud Computing. That is one of the reasons that a lot of universities are offering cloud computing courses so their students can learn the principle, technology and application of cloud computing, and prepare themselves for job market [3, 4, 5, 6]. Cloud Computing courses are offered via various forms. Some universities rent commercial cloud servers, which normally hide technology details to the public so students are not able to learn various aspects of cloud servers. Some other schools build their own cluster of servers that can take a lot of time, thus it is unrealistic course offerings in a semester. Also not all the schools can afford the cost of building cloud servers. There are also schools that teach Cloud Computing courses in which students are asked to do literature survey and presentation, and students are not able to understand the concepts and technology well [7].

These challenges motived us to design our unique Cloud Computing course. In the course titled “Cloud Computing” that we offered, we applied an open source simulation tool called CloudSim to design labware. There are many benefits of our approach, for example:

1) CloudSim is open source that promotes universal access via a free license. Any schools can afford it since it is free to use.

2) CloudSim allows students to build and configure the cloud system in a relatively short time, so the learning curve is short.

DOI 10.1109/COMPSAC.2016.201

3) Students can have hands-on experience on various aspects of Cloud Computing systems with CloudSim.

4) Students are able to conduct research activities with a lot of creativity, using both the labs and extensions provided by CloudSim and the labs created by ourselves that are designed particularly for our Cloud Computing course.

In our Cloud Computing class, students can not only learn conceptual knowledge of cloud computing, but also have hands-on experience, and learn the concepts more efficiently and effectively.

II. SIMULATION BASED CLOUD COMPUTING LABWARE

Cloud Computing systems have various aspects. Our labware is designed to help students understand all aspects of the Cloud Computing such as its principle and architecture, cloud datacenter design and networking, cloud virtualization and resource provisioning, cloud security, etc.. From the labware students will also get hands-on experience on distributed and parallel data processing programming, cloud performance analysis, and task scheduling. We refer to the CloudSim open source project website for basic practice labs [9], and we also create our own labs particularly for our Cloud Computing course. The feedback from the students on the lab activities is very positive.

Various aspects of cloud computing are implemented as stand-alone learning modules in our labware. The majority of labware modules provide three components including Pre-Lab Activities, Hands-on Lab Activities, and Post-Lab Activities.

Students start with Pre-Lab Activities, where they learn knowledge and information of a certain aspect of Cloud Computing. They then move on to the second part, Hands-on Lab Activities, following the instructions and analyzing the simulation results. Afterwards, students continue their work in the third part, Post-Lab Activities, where they implement various simulation tasks on their own. Since CloudSim is an open source simulation toolkit developed in Java, students will also improve their Java programming skills by learning how to use write programs CloudSim.

Here we demonstrate four modules in our simulated Cloud Computing labware:

Module 1. Network Topology

Purpose: In Cloud Computing environment, at any given time, multiple users send requests to Cloud for resource allocation and task processing. In order to have a totally-optimized solution or semi-optimized solution for all the users to have prompt response from Cloud and to be able to process their required tasks as soon as possible, we need to study how data centers are connected in the Cloud Computing system, as well as how users are positioned in the system, and find the shortest or best available path for each message transferring. It will help students understand better the advantages and disadvantages of various network topologies and how they affect the performance of Cloud Computing systems.

a) Pre-Lab Activities

In the Pre-Lab Activities, students will learn the basic concepts of network topology. Network topology is the arrangement of the various elements (links, nodes, etc.) of a computer [10] [11] or biological network [12]. Essentially, it is the topological [13] structure of a network, and may be depicted physically or logically. A good example is a local area network (LAN): Any given node in the LAN has one or more physical links to other devices in the network; graphically mapping these links results in a geometric shape that can be used to describe the physical topology of the network. Conversely, mapping the data flow between the components determines the logical topology of the network. The study of network topology recognizes various basic topologies [14] such as Point-to-point, Bus, Star, Ring or circular, Mesh, Tree, etc.

b) Hands-on Lab Activities

In this lab, we will experiment simulation of resource allocation for multiple users from multiple data centers. Each data center contains multiple virtual machines. Each node in CloudSim represents a data center or a user. There is an edge between two nodes if they are directly connected. In CloudSim, a file titled “topology.brite” is used to record the network topology information. This file has three sections [15] including model information number of nodes and edges such as number of nodes and edges, node information such as node id and node position, edge information such as edge id, length, connected nodes, propagation delay, and bandwidth. Propagation delay is important because it describes how fast information is transferred from one node to another one over the connecting edge.

When there are multiple users and each of them has tasks (represented as CloudLets in CloudSim) to process, the first thing is to find a suitable data center near this user that can possibly provide a virtual machine for the task processing. During this process, request for virtual machine can be failed for various reasons, such as *“[VmScheduler.vmCreate] Allocation of VM #0 to Host #0 failed by MIPS*”, etc.

CloudSim provides some simple java programs to show when there are one data center and one user, two data centers and one user, and two data centers and two users, how CloudLets are sent to virtual machines in different data centers based on the network topology. By testing and analyzing the sample codes, the students will understand how the network topology and resource allocation are simulated.

c) Post-Lab Activity

Students are required to create and design a network based on CloudSim, in which there are three data centers with one host each and three users running three CloudLets (one for each). There are 10 nodes, and 20 edges in the network saved in file "topology.brite":

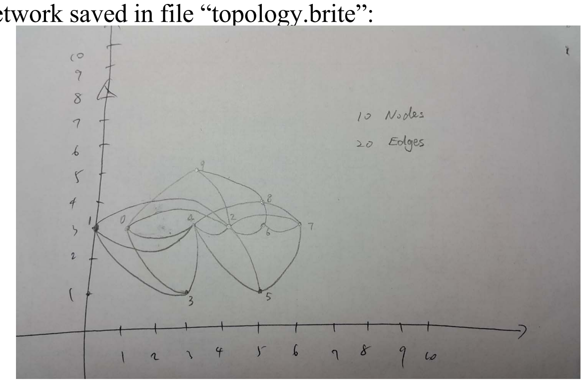

Figure 1 An example of network topology of Cloud

Datacenter 0 will correspond to BRITE node 0, Datacenter 1 will correspond to BRITE node 2, and Datacenter 2 will correspond to BRITE node 7. A Broker represents a user. Broker 1 will correspond to BRITE node 3, Broker 2 will correspond to BRITE node 5, and Broker 3 will correspond to BRITE node 9.

Students implement the network topology described above in their java program, and simulate the running result from CloudSim:

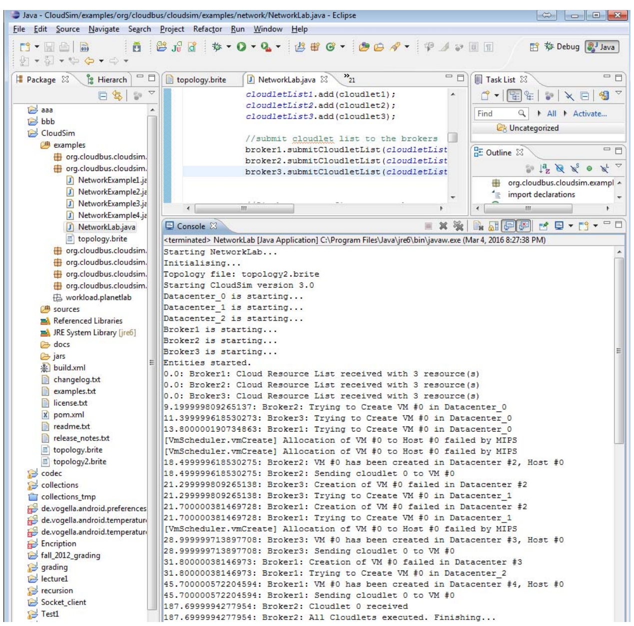

Figure 2 Network topology simulation result part 1

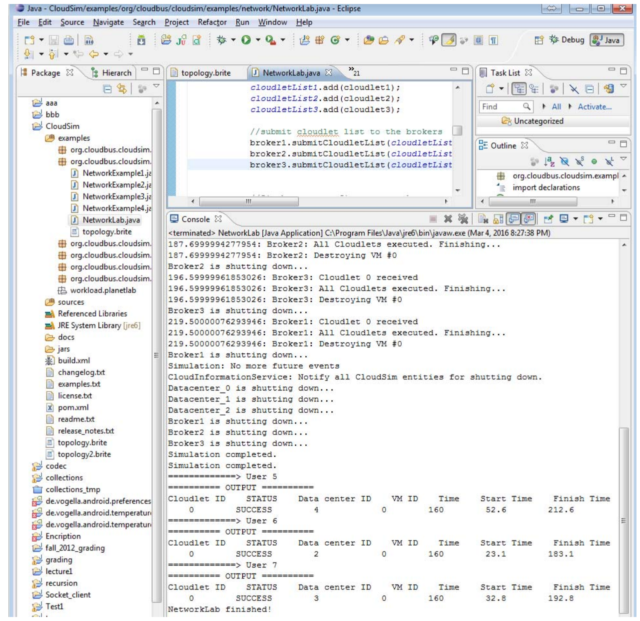

Figure 3 Network topology simulation result part 2

From the simulation result, we can see that, based on the network topology described in Figure 1, the task (CloudLet) of the first user is processed at time 52.6, and ends at time 212.6. The task of the second user is processed at time 23.1, and ends at time 183.1. The task of the third use is processed at time 32.8, and ends at time 192.8.

Module 2. Task Scheduling

Purpose: Task scheduling plays a very important role in Cloud Computing that aims to schedule the tasks effectively in order to reduce the turnaround time and improve resource utilization. This lab module will help students learn different kinds of task scheduling algorithms, as well as how to apply their own scheduling algorithms in CloudSim. It will help students understand better how task scheduling is performed while keeping load balancing in Cloud Computing systems.

a) Pre-Lab Activities

The need for a scheduling algorithm arises from the requirement for most modern systems to perform multitasking (execute more than one process at a time) and multiplexing (transmit multiple flows simultaneously).

The scheduler is concerned mainly with factors such as throughput (the total number of processes that complete their execution per time unit); latency, specifically, turnaround time (total time between submission of a process and its completion), response time (amount of time it takes from when a request was submitted until the first response is produced), and fairness/waiting time (equal CPU time to each process).

Traditional scheduling disciplines include First in first out that simply queues processes in the order that they arrive in the ready queue; shortest remaining time, which arranges processes with the least estimated processing time remaining to be next in the queue; round-robin scheduling that assigns a fixed time unit per process and cycles through them.

b) Hands-on Lab Activities

In this lab, we intend to implement task scheduling algorithms not provided by CloudSim. We implement Min-min, Max-min and Sufferage algorithms [16].

Min-Min [16] is designed in this way: let e_{ij} represent the expected execution time of processing task (CloudLet) t_i on virtual machine m_j. Let r_j denote the expected time that machine m_j will become ready to execute a task after finishing the execution of all tasks assigned to it at that point in time. Let c_{ij} represent the expected completion time of processing task t_i on virtual machine m_j. First the c_{ij} entries are computed using the e_{ij} and r_j values. For each task t_i, the virtual machine that gives the earliest expected completion time is determined by scanning the *i*th row of the c matrix (composed of the c_{ij} values). Once we have the earliest expected completion time for each task t_i, we find the task t_k that has the minimum earliest expected completion time, and assign t_k to the corresponding virtual machine. We then update the matrix c and vector r, and repeat the above process for tasks that have not yet been assigned a virtual machine.

The only difference between Max-min [16] and Min-min is, once we have the earliest expected completion time for each task t_i, we find the task t_k that has the maximum earliest expected completion time, and assign t_k to the corresponding virtual machine.

Sufferage [16] is designed in this way: for each CloudLet, we first define a value called *sufferage* that is the difference of its earliest completion time and second earliest completion time. If a CloudLet is not assigned a virtual machine that gives it the earliest completion time, *sufferage* is what the CloudLet will suffer in terms of time "wasted". In each round of the algorithm, if CloudLet A always "takes" a virtual machine V, but CloudLet B has its *sufferage* value that is greater than that of CloudLet A, CloudLet B will replace CloudLet A to occupy virtual machine V in this round.

We implement Min-min, Max-min and Sufferage algorithms in Class DatacenterBroker:

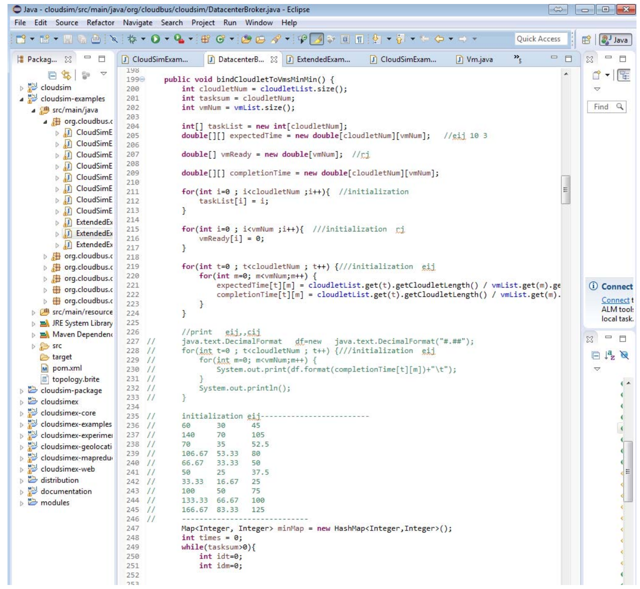

Figure 4 Min-min algorithm for task scheduling

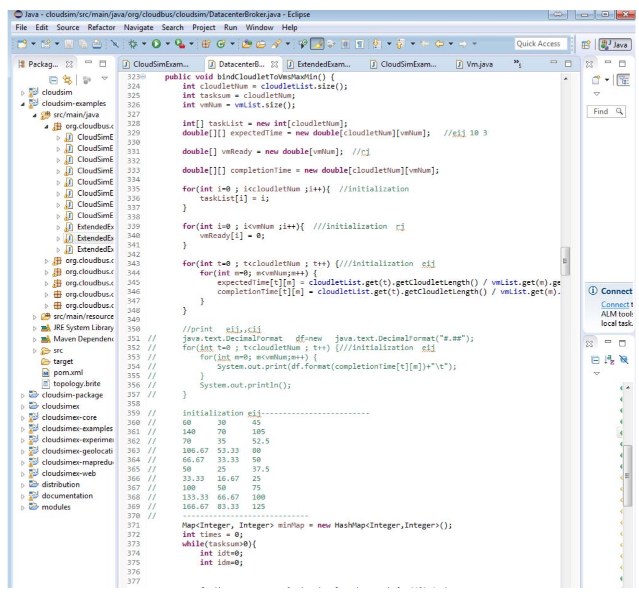

Figure 5 Max-min algorithm for task scheduling

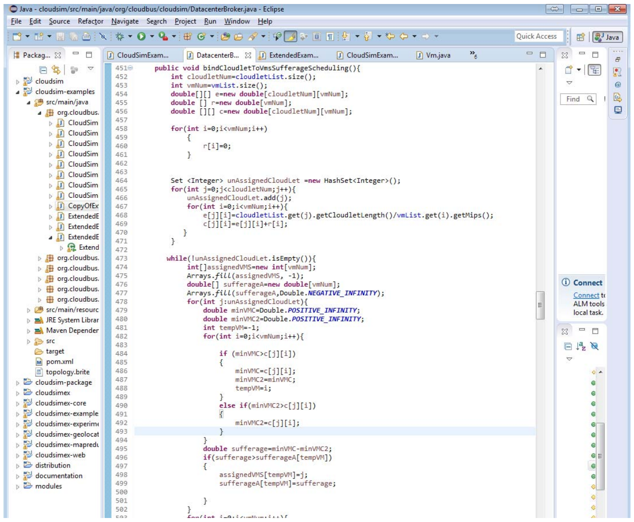

Figure 6 Sufferage algorithm for task scheduling

Before starting simulation, we call bindCloudletToVmMinMin(), bindCloudletToVmMaxMin() and bindCloudletToVmsSufferageScheduling() to apply Min-min, Max-min and Sufferage algorithms respectively:

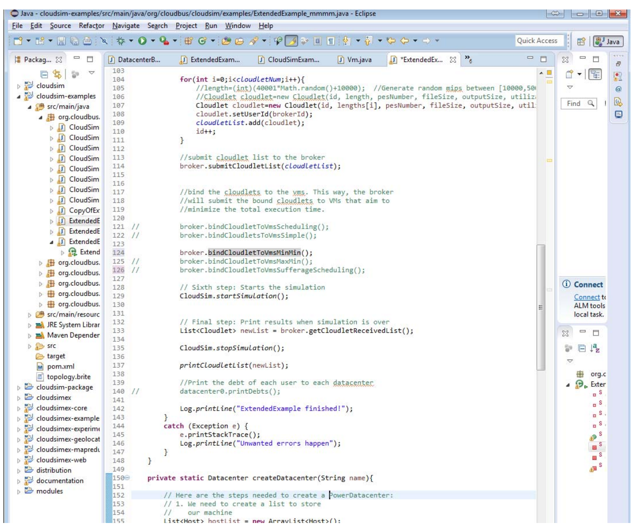

Figure 7 Test algorithm for task scheduling

Students analyzed the performance of Min-min, Max-min and Sufferage algorithms using Throughput and Average Turnaround Time. Throughput is the total number of processes that complete their execution per time unit, and Average Turnaround Time is the average time between submission of a process and its completion. In our experiment, we set the number of virtual machines as 3, and the number of CloudLets as 10. Our experiment shows that for Min-min algorithm, the Throughput is 0.0294 and the Average Turnaround Time is 136.683, for Max-min algorithm, the Throughput is 0.0395 and the Average Turnaround Time is 129.4332, and for Sufferage algorithm, the Throughput is 0.03869 and the Average Turnaround Time is 130.6. We can see that among these three algorithms, based on the input data values, Max-min algorithm can finish more tasks per time unit, and the average completion time of tasks in Max-min is the shortest.

c) Post-Lab Activity

Students are required to implement their own task scheduling algorithm and compare their performance with the existing algorithms using Throughput, Average Turnaround Time, and other measurements.

Module 3. MapReduce

Purpose: MapReduce is a programming framework that allows us to implement distributed algorithm programs that process massive amounts of unstructured data in parallel across a distributed cluster [17]. This module helps students learn how to simulate MapReduce in CloudSimEx environment which is a set of extensions for the CloudSim simulator, since CloudSim has not officially provided MapReduce functionality yet.

Students are required to load the CloudSimEx package in their IDE such as Eclipse, and analyze the MapReduce package. The Simulation class in the MapReduce package is executed, and the result of MapReduce simulation is saved in three Excel files: requests.csv, tasks.csv and vms.csv. These files record how tasks are assigned to various resources at different times, and how the results are grouped together to form the final result.

Module 4. Data Security

Purpose: Cloud Security is on every enterprise customer requirement list. To take advantage of cloud resources while meeting these cloud security requirements, enterprises need to make sure that there is a way for them to maintain control of their sensitive data within the cloud offering, and that using cloud service will not increase the organization's exposure to the soaring costs of data breaches. Thus it is important for the students to understand how crucial to maintain the data security while offering and using cloud services.

This module helps students learn how to simulate data security practice in CloudSim environment. Students are required to simulate the situation where multiple users (brokers) share the same set of data and practice how to make sure the data sharing among these users are private and secured. Students will explore various modules in CloudSim package for setting up the system for secure date sharing among multiple users residing on different network nodes, and simulate their communication to each other for private data sharing.

III. STUDENT FEEDBACK

Upon completion of each labware module, assignment and project, students will be asked to complete a survey of this particular labware module/assignment/project. Here we will demonstrate the survey result.

In the survey, we asked the student if they Agree/Neutral/Disagree on “Q1: I learned various network topology types such as Bus, Star, Mesh and Tree which help me understand how different network topology strategies are designed”, “Q2: The simulation of multiple brokers and multiple data centers in CloudSim helps me understand how each broker selects a data center based on the network topology”, “Q3: The concept of throughput and turnaround time simulated in CloudSim helps me understand how to analyze the performance of certain task scheduling strategies”, “Q4: The lab tutorials help me work on student add-on labs/assignments/projects” and “Q5: The project of implementing Min-min, Max-min and Sufferage algorithms in CloudSim helps me understand better how various task scheduling approaches are designed”. The survey result below shows that the students gave very positive feedback on the labware:

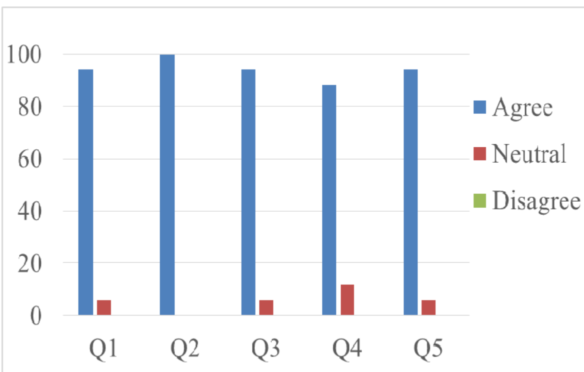

Figure 8 Survey feedback on CloudSim labware

IV. Conclusions

In this paper, we propose a simulation based Cloud Computing labware using open source CloudSim. It can be applied to any Cloud Computing classes without concern of any lab budget. The students learn various aspects of Cloud Computing using our hands-on labware with better understanding of how Cloud Computing provides services of different layers to customers in a more effectively and efficiently way. The preliminary feedback from the students on the lab activities is very positive.

References

[1] https://aws.amazon.com/what-is-cloud-computing/

[2] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya, CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, Software: Practice and Experience, Wiley, 2011

[3] R. A. Brown. Hadoop at home: large-scale computing at a small college. ACM SIGCSE, 2009.

[4] M. Johnson, R. H. Liao, A. Rasmussen, R. Sridharan, D. D. Garcia, and B. Harvey. Infusing Parallelism into Introductory Computer Science Curriculum using MapReduce. Technical Report EECS-2008-34, UC Berkeley, 2008.

[5] A. Kimball, S. Michels-Slettvet, and C. Bisciglia. Cluster computing for web-scale data processing. ACM SIGCSE, 2008.

[6] Ariel Rabkin, Charles Reiss, Randy Katz, David Patterson, Experiences Teaching MapReduce in the Cloud, SIGCSE’12, Raleigh, North Carolina, USA, 2012

[7] Above the Clouds: A Berkeley View of Cloud Computing, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf, 2009

[8] Above the Clouds: A Berkeley View of Cloud Computing, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf, 2009

[9] The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University of Melbourne,

http://www.cloudbus.org/cloudsim/, 2011

[10] Groth, David; Toby Skandier (2005). Network+ Study Guide, Fourth Edition'. Sybex, Inc. ISBN 0-7821-4406-3.

[11] ATIS committee PRQC. "mesh topology". ATIS Telecom Glossary 2007. Alliance for Telecommunications Industry Solutions. Retrieved 2008-10-10.

[12] Proulx, S. R.; Promislow, D. E. L.; Phillips, P. C. (2005)."Network thinking in ecology and evolution". Trends in Ecology and Evolution 20 (6): 345–353.doi:10.1016/j.tree.2005.04.004. PMID 16701391.

[13] Chiang, Mung; Yang, Michael (2004). "Towards Network X-ities From a Topological Point of View: Evolvability and Scalability". Proc. 42nd Allerton Conference.

[14] Bicsi, B., (2002). Network Design Basics for Cabling Professionals. City: McGraw-Hill Professional

[15] http://www.cs.bu.edu/brite/user_manual/node29.html

[16] Muthucumaru Maheswaran, Shoukat Ali, Howard Jay Siegel, Debra Hensgen, and Richard F. Freund. 1999. Dynamic mapping of a class of independent tasks onto heterogeneous computing systems. J. Parallel Distrib. Comput. 59, 2 (November 1999), 107-131. DOI=http://dx.doi.org/10.1006/jpdc.1999.1581

[17] https://en.wikipedia.org/wiki/MapReduce

Acknowledgment

This material is based in part upon work supported by the National Science Foundation under Grant Numbers 1438858, 1623724, and 1241651. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.