The Journal of Computing Sciences in Colleges

Papers of the 37th Annual CCSC Eastern Conference

October 22-23, 2021
Marymount University
Arlinton, VA

Baochuan Lu, Editor
Southwest Baptist University

John Wright, Regional Editor
Juniata College

Volume 37, Number 3 **October 2021**

The Journal of Computing Sciences in Colleges (ISSN 1937-4771 print, 1937-4763 digital) is published at least six times per year and constitutes the refereed papers of regional conferences sponsored by the Consortium for Computing Sciences in Colleges.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors **9**

CCSC National Partners **11**

Welcome to the 2021 CCSC Eastern Conference **12**

Regional Committees — 2021 CCSC Eastern Region **13**

Reviewers — 2021 CCSC Eastern Conference **14**

AI...Transformation — Keynote Adress **15**
Michael S. Groen, USMC

Integrating Computational Thinking Into K-12 Teacher Preparation: A Collaborative Partnership Between the Department of Computer Science and the College of Education — Panel Discussion **17**
Michael B. Flinn, Kristine McGee, Frostburg State University, Katelyn Barnes, Haylee Morton, Garrett County Public Schools

Using a Raspberry Pi to Teach Python Coding to Students of All Ages — Conference Tutorial **20**
Susan Conrad, Samantha King, Natalia Ermicioi, Marymount University

Teaching Mathematical Foundations of Computer Science — Conference Tutorial **22**
Pradip Peter Dey, Hassan Badkoobehi, National University

Project-Based Data Science Course Design **24**
Sunghee Kim, Gettysburg College

Fostering a Sense of Belonging in Female Computer Science Students **38**
James Geller, New Jersey Institute of Technology

Analyzing Brain Tumor MRI to Demonstrate GPU-based Medical Image Segmentation **47**
Sajedul Talukder, Southern Illinois University, Neil Noyes, Edinboro University

Opportunities for Using HBCU Culture to Teach Elementary Data Structures to Computing Students **65**
Gloria Washington, Saurav Aryal, Todd Shurn, Legand Burge III, Howard University, Marlon Mejias, UNC Charlotte

Introducing Computational Thinking to Pre-service Teachers **76**
Jiang Li, Paulette Shockey, Jennifer Cuddapah, Christy Graybeal, Marisel Torres-Crespo, Anthony Williams, Hood College

Checkpoint Variations for Deep Q-Learning in Reconnaissance Blind Chess **83**
Kyle Blowitski, Timothy Highley, La Salle University

Resource Management with Java **91**
Yaodong Bi, University of Scranton

A Hands-On Arm64 Lab for Computer Organization **104**
Robert Montante, Bloomsburg University of Pennsylvania

Development of a Virtualized Security Operations Center **110**
Robert de Céspedes III, George Dimitoglou, Hood College

Integrating Cloud Computing into Computer Science Curriculum **122**
Aijuan Dong, Hood College

A Successful Online Systems Class using Scaffolded Active Learning and Formative Assessment **134**
Michael O. Lam, Dee A. B. Weikle, James Madison University

How Colleges Can Better Prepare Technology Students for Internships Students Perception of Internship Programs **145**
Susan S. Conrad, Marymount University

Enhancing Alternative Proof of Work for Cryptocurrencies Using Machine Learning — Faculty Poster Abstract **157**
Sajedul Talukder, Southern Illinois University, Abdur R. Shahid, Robert Morris University

Analyzing the Optimal Level of Investment in Cybersecurity for Nonprofit Organizations(NPO) by Combining Gordon-Loeb and FAIR Models — Faculty Poster Abstract **158**
Natalia Ermicioi, Michelle (Xiang) Liu, Marymount University

Text-based Large-Scale Programming Assignment for Teaching Object-Oriented Analysis and Design — Faculty Poster Abstract **159**
Jessica Zeitz, Veena Ravishankar, University of Mary Washington

An Applied Machine Learning Approach to Detecting Fraud in Mobile Transactions — Faculty Poster Abstract **160**
Faleh Alshameri, Alex Mbaziira , Khadijah Makhadmi, Marymount University

Hybrid Student Service Learning through Creative Community Partnership — Faculty Poster Abstract **161**
Lily Liang, Briana Wellman, Uzma Amir, Jeffrey Enamorado, Jermel Watson, University of the District of Columbia, Abdeladim Elhamdani, Rockville Science Center

Survey of COVID Online Teaching Methodologies and How They Can Be Used in the Traditional Classroom — Faculty Poster Abstract **162**
Jennifer Polack, University of Mary Washington

Teaching Advanced Data Structures and Algorithms Online During a Pandemic — Faculty Poster Abstract **163**
Olumide Kayode, Frostburg State University

Machine Learning Android-Based Malware Classification — Faculty Poster Abstract **164**
Micheline Al Harrack, Marymount University

Applying Local Differential Privacy in Handwriting Recognition-based Systems — Faculty Poster Abstract **165**
Sajedul Talukder, Southern Illinois University, Abdur R. Shahid, Robert Morris University

Evaluation of Privacy-Preserving Logistic Regression and Naive Bayes Classifiers in Healthcare — Faculty Poster Abstract **166**
Abdur R. Shahid, Robert Morris University, Sajedul Talukder, Southern Illinois University

Privacy-Preserving Activity Recognition from Sensor Data — Faculty Poster Abstract **167**
Abdur R. Shahid, Robert Morris University, Sajedul Talukder, Southern Illinois University

Prevalence of PII within Public Malware Sandbox Samples and Implications for Privacy and Threat Intelligence Sharing — Student Paper Abstract **168**
Aaron Weathersby, Marymount University

How Can Romania Leverage Education to Become a Regional Cybersecurity Leader? — Student Paper Abstract **169**
Daria Pop, Georgetown University, Natalia Ermicioi, Marymount University

An Observational Assessment of CTI Standards for Blue Teams — Student Paper Abstract **170**
Jeremy McHugh, Dawn Childs, Jerry Jenkins, Marymount University

DReAM: A Deepfake Recognition Assessment Model — Student Paper Abstract **171**
JJustin Ubert, Sheryl Drake, Seth Moyers, Marymount University

Advantages of Python Programming Language in the World of Big Data — Student Poster Abstract **172**
Meharban Arora, Marymount University

Using Multiple Forensic Tools to Analyze the Data Acquisitions of Previously Used iOS & Android Devices — Student Poster Abstract **173**
Katrina Khanta, Marymount University

US Policy on the Use of Force in Cyberspace — Student Poster Abstract **174**
Roncs Etame-Ese, Marymount University

Cybersecurity Education for the Government Workforce: Lessons from Counterintelligence — Student Poster Abstract **175**
Vincent Pisani, Marymount University

An Examination of Consumer IoT Security & Privacy — Student Poster Abstract **176**
Darrell Andrews, Marymount University

An Analysis of the Rise of Data Breaches in the Healthcare Organizations During Covid-19 — Student Poster Abstract **177**
Laura Vera, Marymount University

Tool to Teach and Practice the GitHub Workflow — Student Poster Abstract **178**
Jacob Smith, Moravian University

The Consortium for Computing Sciences in Colleges Board of Directors

Following is a listing of the contact information for the members of the Board of Directors and the Officers of the Consortium for Computing Sciences in Colleges (along with the years of expiration of their terms), as well as members serving CCSC:

Karina Assiter, President (2022), (802)387-7112, karinaassiter@landmark.edu.

Chris Healy, Vice President (2022), chris.healy@furman.edu, Computer Science Department, 3300 Poinsett Highway Greenville, SC 29613.

Baochuan Lu, Publications Chair (2021), (417)328-1676, blu@sbuniv.edu, Southwest Baptist University - Department of Computer and Information Sciences, 1600 University Ave., Bolivar, MO 65613.

Brian Hare, Treasurer (2020), (816)235-2362, hareb@umkc.edu, University of Missouri-Kansas City, School of Computing & Engineering, 450E Flarsheim Hall, 5110 Rockhill Rd., Kansas City MO 64110.

Cathy Bareiss, Membership Secretary (2022), cathy.bareiss@betheluniversity.edu, Department of Mathematical Engineering Sciences, 1001 Bethel Circle, Mishawaka, IN 46545.

Judy Mullins, Central Plains Representative (2023), Associate Treasurer, (816)390-4386, mullinsj@umkc.edu, UMKC, Retired.

Michael Flinn, Eastern Representative (2023), mflinn@frostburg.edu, Department of Computer Science Information Technologies, Frostburg State University, 101 Braddock Road, Frostburg, MD 21532.

David R. Naugler, Midsouth Representative(2022), (317) 456-2125, dnaugler@semo.edu, 5293 Green Hills Drive, Brownsburg IN 46112.

Grace Mirsky, Midwest Representative(2023), gmirsky@ben.edu, Mathematical and Computational Sciences, 5700 College Rd. Lisle, IL 60532.

Lawrence D'Antonio, Northeastern Representative (2022), (201)684-7714, ldant@ramapo.edu, Computer Science Department, Ramapo College of New Jersey, Mahwah, NJ 07430.

Shereen Khoja, Northwestern Representative(2021), shereen@pacificu.edu, Computer Science, 2043 College Way, Forest Grove, OR 97116.

Mohamed Lotfy, Rocky Mountain Representative (2022), Information Systems & Technology Department, College of Engineering & Technology, Utah Valley University, Orem, UT 84058.

Tina Johnson, South Central Representative (2021), (940)397-6201, tina.johnson@mwsu.edu, Dept. of Computer Science, Midwestern State University, 3410 Taft Boulevard, Wichita Falls, TX 76308.

Kevin Treu, Southeastern Representative (2021), (864)294-3220, kevin.treu@furman.edu, Furman University, Dept of Computer Science, Greenville, SC 29613.

Bryan Dixon, Southwestern Representative (2023), (530)898-4864,

bcdixon@csuchico.edu, Computer Science Department, California State University, Chico, Chico, CA 95929-0410.

Serving the CCSC: These members are serving in positions as indicated:

Bin Peng, Associate Editor, (816) 584-6884, bin.peng@park.edu, Park University - Department of Computer Science and Information Systems, 8700 NW River Park Drive, Parkville, MO 64152.

Shereen Khoja, Comptroller, (503)352-2008, shereen@pacificu.edu, MSC 2615, Pacific University, Forest Grove, OR 97116.

Elizabeth Adams, National Partners Chair, adamses@jmu.edu, James Madison University, 11520 Lockhart Place, Silver Spring, MD 20902.

Megan Thomas, Membership System Administrator, (209)667-3584, mthomas@cs.csustan.edu, Dept. of Computer Science, CSU Stanislaus, One University Circle, Turlock, CA 95382.

Deborah Hwang, Webmaster, (812)488-2193, hwang@evansville.edu, Electrical Engr. & Computer Science, University of Evansville, 1800 Lincoln Ave., Evansville, IN 47722.

CCSC National Partners

The Consortium is very happy to have the following as National Partners. If you have the opportunity please thank them for their support of computing in teaching institutions. As National Partners they are invited to participate in our regional conferences. Visit with their representatives there.

Platinum Partner

Turingscraft
Google for Education
GitHub
NSF – National Science Foundation

Silver Partners

zyBooks

Bronze Partners

National Center for Women and Information Technology
Teradata
Mercury Learning and Information
Mercy College

Welcome to the 2021 CCSC Eastern Conference

The CCSCE Conference Committee is looking forward to a tremendous in-person event this year. We warmly welcome you to the 37th Annual Conference, where computer professionals, educators, and students come together to learn about contemporary topics involving computer science.

This year's conference will be held at Marymount University in Arlington, Virginia. After enduring a long year of zoom and online meetings, the committee carefully weighed all the data and decided to take this conference to an in-person event. We are delighted with the quality and quantity of submissions for papers, posters, workshops, and special sessions. Thank you to everyone who submitted a proposal to the conference. At this point in time, we have participants from more than 20 schools in the region which is sure to make the Eastern Region CCSC conference a great success.

We have two days of excellent programs planned for the professional enrichment of our audiences, which include an invited keynote, paper presentations, workshops, tutorials, poster presentations, and a programming competition for the students. Our speakers are tops in their fields, beginning with Lieutenant General Michael S. Groen, USMC Director, Joint Artificial Intelligence Center Department of Defense, and then hearing from Mr. Vint Cerf, often referred to as the "Father of the Internet." This year the conference had 19 professional paper submissions, out of which we have accepted 12 papers for an acceptance rate of 63%. All papers underwent a double-blind review process and on average papers were reviewed by 3 reviewers. The conference is supported by faculty from multiple institutions who served on the Conference Committee, as reviewers, etc. We want to express our sincere gratitude to everyone involved in making this conference a reality. Special thanks to all those who helped develop the program, coordinated the paper reviews, organized the programming competition, coordinated the keynote speaker, coordinated panels, workshops, tutorials, nifty ideas, and lightning talks. Our gratitude also goes to the judges for the poster awards, the session chairs, and student volunteers. We also appreciate the continuous effort and support from the CCSC Eastern Region Steering Committee, and we are very grateful for the generous supports of the CCSC National Partners, Sponsors, and Vendors.

We want to send special thanks to Mike Flinn (Frostburg State University) for serving as the CCSC Eastern representative and providing invaluable advice to the chairs of this year's conference. Also, our great appreciation goes to our colleague, John Wright (Juniata College), who tirelessly worked with authors to get the papers and abstracts ready for the proceedings. Lastly, thank you to our attendees for sharing your knowledge with colleagues to enhance education for everyone.

Susan Conrad and Natalia Ermicioi
Conference Co-Chairs, Marymount University

2021 CCSC Eastern Conference Committee

Susan Conrad, Co-Chair ..... Marymount University
Natalia Ermicioi, Co-Chair ..... Marymount University
John Wright, Faculty Papers, Web Site ..... Juniata College
George Dimitoglou, Student Papers ..... Hood College
Pranshu Gupta, 2022 Conference Chair ..... DeSales University
Karen Anewalt, Posters ..... University of Mary Washington
YuLin Bingle, Posters ..... Marymount University
Ian Finlayson ..... University of Mary Washington
Steven Kennedy, Programming Contest ..... Frostburg State University
Dave Hovemeyer, Programming Contest ..... Johns Hopkins University
Jiang Li, Zoom Master ..... Hood College
Nathan Green, Regional Treasurer ..... Marymount University
Michael Flinn, Regional Board Representative .... Frostburg State University

2021 CCSC Eastern Steering Committee

Elizabeth Adams ..... James Madison University
Steven Andrianoff ..... Bonaventure University
Karen Anewalt ..... University of Mary Washington
George Benjamin ..... Muhlenberg College
Elizabeth Chang ..... Hood College
Vincent Cicirello ..... Stockton University
George Dimitoglou ..... Hood College
Michael Flinn ..... Frostburg State University
Sister Jane Fritz ..... St. Joseph's College
Nathan Green ..... Marymount University
David Hovemeyer ..... Johns Hopkins University
John Meinke ..... University of Maryland University College
Karen Paullet ..... Robert Morris University
Donna Schaeffer ..... Marymount University
Jennifer Polack-Wahl ..... University of Mary Washington
John Wright ..... Juniata College

Reviewers — 2021 CCSC Eastern Conference

Faleh Alshameri ..... Marymount University, Arlington, VA
Karen Anewalt ..... The University of Mary Washington, Fredericksburg, VA
Ian Barland ..... Radford University, Radford, VA
David Conner ..... Christopher Newport University, Newport News, VA
Susan Conrad ..... Marymount University, Arlington, VA
Ian Finlayson ..... The University of Mary Washington, Fredericksburg, VA
Michael Flinn ..... Frostburg State University, Frostburg, MD
Reva Freedman ..... Northern Illinois University, DeKalb, IL
Alessio Gaspar ..... University of South Florida, Tampa, FL
Grigoriy Grinberg ..... Montgomery College, Gaithersburg, MD
Edward Lindoo ..... Regis University, Denver, CO
Robert Marmorstein ..... Longwood University, Farmville, VA
Alex Mbaziira ..... Marymount University, Arlington, VA
John McManus ..... Randolph-Macon College, Ashland, VA
John Pais ..... Ladue Horton Watkins High School, St. Louis, MO
Sofya Poger ..... Felician University, Rutherford, NJ
Veena Ravishankar . The University of Mary Washington, Fredericksburg, VA
Donna Schaeffer ..... Marymount University, Arlington, VA
Richard Scorce ..... St. John's University, Jamaica, Queens, NY
JoAnna Shore ..... Frostburg State University, Frostburg, MD
Alla Webb ..... Montgomery College, Gaithersburg, MD

AI...Transformation*

Keynote Address

Lieutenant General Michael S. Groen, USMC
Director, Joint Artificial Intelligence Center
Department of Defense

Biography

Lieutenant General Michael S. Groen assumed his current position as the Director, Joint Artificial Intelligence Center on 1 October 2020. As a member of the JAIC team, he leads the transformation of U.S. Joint warfighting and departmental processes through the integration of Artificial Intelligence.

Prior to this nomination, General Groen was assigned to the National Security Agency and served as the Deputy Chief of Computer Network Operations, leading this premier Computer Network Exploitation organization. In 2018/2019, he served as the Director for Intelligence, Joint Staff (J2) in direct support of the Chairman of the Joint Chiefs and the Joint Staff. He also served as the Vice J2. Prior to his Joint Staff assignments, General Groen served as the Director of Marine Corps Intelligence (DIRINT) where he championed the redesign of intelligence capabilities into a Marine Corps Intelligence, Surveillance, and Reconnaissance Enterprise (MCISRE).

General Groen has served in a variety of operational, ground, air, and naval units. His service has included Central America, the Western Pacific, the Philippines, the Balkans and Iraq, General Groen served afloat with the 31st Marine Expeditionary Unit and supported aviation units in the U.S. and Okinawa. In 2003, he was assigned to the 1st Marine Division. He initially served as the Deputy Intelligence officer, then became the Intelligence Officer (G-2) in 2004. He has supported both conventional combat and counter-insurgency operations. General Groen was a principal in the redesign of Marine Intelligence to meet the emerging demands of the Global War on Terror. Later, General Groen served with the U.S. European Command as the Chief of Intelligence Planning for Europe and Africa. He was instrumental in transitioning intelligence processes into the Joint Intelligence Operations Center.

General Groen has commanded intelligence and operational units including the 3d Radio Battalion (conducting its first deployment to the Southern Philippines in support of Operation ENDURING FREEDOM) and the Headquarters Battalion, 1st Marine Division in Camp Pendleton, California. He was also given additional duties as the Division's Chief of Staff. As a perpetual change-agent, he has served as a Combat Development, Requirements, and Acquisition officer. He served as the initial Director of the Commandant's Amphibious Capabilities Working Group (Capabilities), the Ellis Group (Expeditionary Futures) and as the Director of the Commandant of the Marine Corps Strategic Initiatives Group (SIG).

General Groen is a graduate of Calvin College in Grand Rapids, MI with a Bachelor's Degree in Engineering. He has received multiple Master's Degrees from the University of Southern California (Systems Management) and from the Naval Postgraduate School (Electrical Engineering, Applied Physics.) General Groen is a graduate of the Marine Corps Command and Staff College and the Naval War College.

General Groen's personal decorations include the Defense Superior Service Medal, the Legion of Merit, the Bronze Star, and the Combat Action Ribbon.

General Groen is a native of Michigan with three sons.

Integrating Computational Thinking Into K-12 Teacher Preparation: A Collaborative Partnership Between the Department of Computer Science and the College of Education*

Panel Discussion

Michael B. Flinn[1], Kristine McGee[2],
Katelyn Barnes[3], Haylee Morton[4]
[1]Computer Science Department, [2]College of Education
Frostburg State University, Frostburg, MD 21532
{mflinn, kmcgee}@frostburg.edu
[3,4]Garrett County Public Schools, Garrett County, MD
{katelyn.barnes, haylee.morton}@garrettcountyschools.org

Summary

Computer science education is more important than ever. The COVID-19 pandemic has highlighted our society's reliance on computing and its power to help businesses innovate and adapt, yet at the same time has surfaced greater disparities for students studying computer science. Computing is the number one source of all new wages in our economy, and there are currently 400,000 open computing jobs across the United States. However, unequal access to computer science instruction and opportunities to engage in computational thinking and practices remain prevalent (Google & Gallup, 2015). In the 2018-19 school year, only 15 percent of Maryland graduates took at least one computing related course, and there were significant gaps in the course taking patterns by gender and race (Maryland Center for Computing Education, 2021). To grow students' competencies with computational thinking and computer science, it is essential to build the capacity of teachers to integrate computational

thinking competencies and standards into their practice. The Frostburg State University College of Education's Maryland Accelerates (MA) Program provides aspiring teachers the opportunity to earn an accelerated Master of Arts in Teaching (Elementary or Secondary track) degree along with an innovative yearlong teacher residency in partnership with Frederick and Garrett County Public Schools. One of the primary goals of the program is to integrate mathematical problem solving and computational thinking to promote scientific inquiry in partnering elementary and secondary schools. To this end, Frostburg State University has developed an exciting opportunity for Master of Arts in Teaching (MAT) students to earn a microcredential in Computational Thinking that is aligned to the ISTE Educator Standards, the High Leverage Practice Standards from TeachingWorks, and the Interstate Teacher Assessment and Support Consortium InTASC Model as part of their teacher preparation program. In this panel, faculty from the Computer Science Department and the College of Education at Frostburg State University will discuss their partnership to develop a microcredential for Computational Thinking for pre-service teachers, how and why the credential was developed, and what competencies aspiring teachers develop. Two recent graduates of the Maryland Accelerates MAT Teacher Residency program will discuss their experience using these skills and knowledge during their residency and how they plan to apply and integrate computational thinking into their practice as new teachers.

Dr. Michael B. Flinn, Professor and Chair, Computer Science Department, Frostburg State University (MODERATOR)

Dr. Flinn will discuss how the collaboration between the computer science department and College of Education was formed, and how this initiative aligns to other work at the university and across Maryland to improve computer science education. Dr. Flinn's primary research interests include understanding how to advance student knowledge of Computing Science theories, concepts, and applications; discovering how students synthesize the "system thinker" with their course of study; and promoting computer science and computing in general as must-have skills in the modern, computer-driven, world. In addition to his position at Frostburg, Dr. Flinn serves on the steering committee of CCSC Eastern Conference and Tech at the Gap. Dr. Flinn is also the Co-PI on a Maryland Center for Computing Education (MCCE) grant to develop a Maryland framework for preservice coursework to lead to secondary computer science certification, or add-on certification and K-8 integration of computational thinking. Dr. Flinn holds a D.Sc. Information Systems and Communications from Robert Morris University, and a MS in Applied Computer Science, a BS in Computer Science, and a BS in English, all from Frostburg State University.

Dr. Kristine McGee, Associate Professor, College of Education, Frostburg State University
Dr. McGee will discuss how the program was integrated into the Elementary MAT in pilot year, how students applied computational thinking concepts into their teaching at the elementary level. Dr. McGee's passion and expertise include teaching and mentoring pre-service and in-service teachers to develop their literacy lives. She strives to create experiential learning opportunities for her students, to help them develop a strong connection to literacy and to the children in the community. Sharing her work, collaborating with others, and presenting at local, state, and national conferences bring her great pride and joy. She recently co-authored a chapter titled, "Using Technology to Build Interactions Within and Beyond the Literacy Classroom," in the Handbook of Research on Integrating Digital Technology with Literacy Pedagogies. Kris is an avid reader, "book dealer," and self-proclaimed "tech junkie." She is celebrating her 31st year as an educator.

Katelyn Barnes, Teacher, Garrett County Public Schools
Ms. Barnes will discuss her personal experience completing the microcredential in Computational Thinking, discuss her experience using these skills and knowledge during her residency, and her plans to apply and integrate computational thinking into her practice as a new teacher. Ms. Barnes holds a MAT in Elementary Education, and a BA in Liberal Studies with a concentration in elementary education from Frostburg State University. Teaching is her passion, and she is always looking to learn new things.

Haylee Morton, Teacher, Garrett County Public Schools
Ms. Morton will discuss her personal experience completing the microcredential in Computational Thinking, discuss her experience using these skills and knowledge during her residency, and her plans to apply and integrate computational thinking into her practice as a new teacher. Ms. Morton holds a MAT in Elementary Education, and a BA in Liberal Studies with a concentration in elementary education from Frostburg State University.

References

[1] Google and Gallup (2015). Searching for Computer Science: Access and Barriers in U.S. K-12 Education. Accessed on May 21, 2021 at `https://csedu.gallup.com/home.aspx`

[2] Maryland Center for Computing Education (2021). Computer Science Dashboard. Accessed on May 21, 2021 at `https://mldscenter.maryland.gov/ComputerscienceDashboard.html`

Using a Raspberry Pi to Teach Python Coding to Students of All Ages*

Conference Tutorial

Susan Conrad, Samantha King and Natalia Ermicioi
Department of IT and Cybersecurity
Marymount University
Arlington, Virginia 22207
{sconrad, srk25218, nermicio}@marymount.edu

Summary

This workshop will demonstrate how to set-up a Raspberry Pi and use it to teach students how to code. It will discuss what hardware is needed to set up the Raspberry Pi and how to set up both blue-tooth and wireless capabilities. It will discuss the open-source resources available to guide users in to pre-made exercises utilizing Raspberry Pi and providing students with hands-on experiences. The workshop will demonstrate several projects that can be used in coding classes for students of all abilities.

Dr. Susan Conrad

Dr. Susan Conrad is a professor at Marymount teaching introductory python classes. She has a PhD in Education and Technology and has a strong teaching background. Dr. Conrad is working with 2 honors students to code many of the Raspberry Pi open-source python coding projects and put together a guide along with videos to help teachers and students complete projects. Sue has many years in IT working in industry before coming to Marymount to teach.

Samantha King

Samantha King is a Honor student in her third year studying computer science. She has done research on teaching young students to code. She works

with young children over the summer and is passionate about teaching future generations about coding.

Natalia Ermicioi

Dr. Natalia Ermicioi is serving as an Assistant Professor - Tenure Track and Faculty Leadership Fellow (AY 2021-2022) in the College of BILT, School Technology and Innovation at Marymount University teaching undergraduate and graduate-level courses in IT and Cybersecurity; supervising senior-level IT and Cybersecurity Capstone projects and serving on various doctoral committees in the Doctorate of Science in Cybersecurity Program. Additionally, she serves as the faculty mentor for MU's ACM-Women Chapter and MU ISACA Student Group. Dr. Ermicioi considers that her diverse background and fluency in five languages have afforded her a well-rounded and diverse skill set and experiences around the world, which is an added value to such an interdisciplinary and multi-faced field as IT and cybersecurity.

Intended audience: This workshop is intended for instructors teaching introductory python, cybersecurity or data science classes. It is also intended for beginning coders who want to challenge themselves with a Python coded project on a personally built Raspberry Pi.

Description of materials provided to participants: Participants will receive access to powerpoints and specific project specifications demonstrated in the tutorial.

Teaching Mathematical Foundations of Computer Science*

Conference Tutorial

Pradip Peter Dey and Hassan Badkoobehi
Department of Engineering and Technology
National University
San Diego, CA 92123
{pdey, hbadkub}@nu.edu

Summary

The foundational aspects of computer science are usually explained with mathematical models such as Finite Automata (FA), Pushdown Automata (PDA) and Turing Machines (TMs). Challenges of teaching mathematical foundations can be overcome with innovative approaches and agile planning. For undergraduate students, teachers should start with a general overview of the area so that students get the big picture and become interested in learning more about foundational ideas and their relationships. Active participation of students should be encouraged from the beginning when the basic ideas are introduced. At the initial stages, proof ideas need to be emphasized which may be followed by rigorous mathematical proofs when students fully understand the intuitive ideas behind the proof.

Tutorial Description

In order to have good interactions with learners, multiple representations are used for initial discussion of mathematical ideas [1, 2, 3, 4]. The central ideas of computation are defined in an area commonly known as the theory of computation or automata theory where mathematical models including FA, PDA and TMs are explained with examples. Undecidable problems

are explained with examples such as integral polynomials [4]. Some supplemental materials from the following site are used in this tutorial: `http://www.asethome.org/mathfoundations/`

Expected outcomes

Attendees will exit the tutorial with a good understanding of mathematical foundations of computer science. Attendees will have free access to the supplemental materials at the following site: `http://www.asethome.org/mathfoundations/`

Target audience

Any faculty who desires to teach mathematical models of computation.

Prerequisites

None. Everybody is welcome to this tutorial where intuitive explanations are presented about teaching mathematical foundations.

References

[1] Ainsworth, S. DeFT: A conceptual framework for considering learning with multiple representations. Learning and Instruction, 16, 183-198, 2006.

[2] Cohen, D. Introduction to Computer Theory. (2nd. Ed.). John Wiley, 1996.

[3] Hopcroft, J., Motwani, R., and Ullman, J. Introduction to Automate Theory, Languages, and Computation. (2nd. Ed) Pearson Education, 2007.

[4] Sipser, M. Introduction to the Theory of Computation, (3rd Ed.) Cengage Learning, 2012.

Project-Based Data Science Course Design*

Sunghee Kim
Department of Computer Science
Gettysburg College
Gettysburg, PA 17325
skim@gettysburg.edu

Abstract

This paper describes a data science course design for undergraduate computer science students. The data science course is a major elective course that aims to equip undergraduate students majoring in computer science with working knowledge of data analytics methodologies and basic techniques in a single course. The course assessment consists of two exams, laboratory exercises, programming assignments, and group term projects. A group term project is a significant component of the course and requires two or three students to work together on a data science project, from defining the problem and data requirement to building and evaluating models. This paper offers ideas for the structure and content of a single data science course for computer science students as an upper-level elective course and presents sample project topics in various domains.

1 Introduction

In the past decade the popularity of data science has surged in many industry sectors. There is a shortage of workers with data analytics skills and the demand for data science professionals is expected to rise even faster in the next decade [8][7]. To address this trend and need, higher education institutions have started offering data analytics courses and data science degree programs

[3]. This paper aims to contribute to this curricular update effort by providing the structure and sample content for a data science course as a major elective for undergraduate computer science (CS) students. Hicks and Irizarry suggest that computing should be an essential component of a data science course [5]. A data scientist needs to work with data of all types, shapes, and sizes and in the era of big data manual processing is no longer practical. While a considerable amount of time and efforts are necessary to bring up programming skills in a data science course targeting students with little to no programming experience, a course for computer science students can include more computing-intensive activities and topics since they are already proficient in programming and problem-solving. The course presented in this paper attempts to cover the fundamental topics in data science including statistics and machine learning so that CS students can relatively quickly acquire data science principles and skills.

In the first offering of the course in fall 2018, lectures centered around case studies and examples to prompt students to think critically about the problems and to practice working with real data. Students were expected to gain a working knowledge of the principles of data science and learn the basic techniques from analysis of the problem to evaluation of data models. Students collaborated in groups of two or three on a real-world problem of their choice and applied their newly acquired knowledge to the problem from start to finish.

2 Course Design

2.1 Background and Development

At the time this course was being designed, an interdisciplinary workgroup with faculty representatives from all divisions was being formed to explore a data science minor program in the author's institution (SchoolA). In the academic year 2019-2020 the six-course minor program that is open to all majors was officially launched. There were a large number of interested computer science students who would not have the opportunity to take official data science courses and it was decided that a special topics course would focus on introducing data science methodology and tools and providing the students with opportunities to practice their skills on solving data science problems.

Java is the main programming language used in the CS major curriculum in SchoolA. Though the students are exposed to different programming languages in other core and elective courses, such as C and Assembly in computer organization, C++ in computer graphics, web programming languages in database management systems, they tend to feel apprehensive about writing programs in languages other than Java. More than half of the students in the course had just taken data structures, the third core course in the major and the only

pre-requisite course for the data science course, in the previous semester and lacked application-level programming experiences. After considering both R and Python, it was ultimately decided that Python would be used in the course along with Jupyter Notebook for an easier transition from Java to Python for the students. The course was programming intensive and included a number of laboratory exercises and programming assignments so that the students could gain a reasonable level of comfort in both writing code in Python and using data science libraries.

2.2 Learning Goals and Objectives

The learning goals of the course were for the students to gain an understanding of data science principles, to learn the basic techniques, and to familiarize themselves with the entire data science process. Specifically, students were expected to learn (1) the fundamental Data Science concepts and processes, (2) data preparation techniques including data collection, cleaning, and integration, (3) data analysis techniques such as exploratory data analysis, predictive modeling, and descriptive modeling, (4) data production and evaluation techniques, and (5) effective communication and presentation of the findings.

2.3 Course Assessment

The evaluation of the students' performance in the course was based on active participation (2%), eight programming assignments (38%), term project (26%), and two exams (34%). All programming assignments were to be completed individually while the term project was a collaborative effort in groups of two or three which were formed based on the students' interest.

2.4 Textbooks and Computing Environment

After examining several published books on data science and related topics, it was decided that two books that were freely accessible online would be used as main textbooks in the course:

- Think Stats, Exploratory Data Analysis in Python [4]
- Python Data Science Handbook, Essential Tools for Working with Data [13]

The computer science major degree in the author's institution does not require statistics or experiment design and it was important for the students to learn the fundamentals of statistical analysis as part of the course. Both books provide python code examples for every step of data analytics methodology and easy-to-understand case studies. While the lectures and laboratory exercises

were conducted in a computer classroom with Linux machines, the majority of the students used their own laptops for assignments and group project.

2.5 Course Schedule

The weekly topics, assignments, project milestones, and case studies are presented in Table 1. In the first week, students were introduced to the basics of Python and they were asked to write functions to process different data types and data structures in the first assignment. As the students were already familiar with programming in Java, they were able to learn Python syntax with class examples, exercises, and assignments. Throughout the semester in-class labs were used to introduce Python data structures and libraries which they incorporated in the assignments and project. From week 2 topics and libraries specific to data science were introduced.

3 Term Projects

3.1 Overview

One of the main components of the course assessment was a collaborative term project that students worked on throughout the semester. Early in the semester a class session was devoted to brainstorming and discussing possible data science project topics and students came up with twelve ideas. Students then submitted three topics from the list in the order of preference which was used to form groups of 2 or 3 students. Evaluation of the term project consisted of multiple milestone submissions and activities:

- Defining the problem and data requirement (week 6)
- Data acquisition and preprocessing (week 7)
- Preliminary analysis including exploratory data analysis and visualization (week 11)
- Presentation (week 14)
- Final analysis and report (week 15)
- Self and peer evaluation (week 15)

There were six groups in total; four groups of three and two groups of two students. In the next six sections, each group project is briefly described. Groups were required to use GitHub for collaboration and version controls and to include all data, references, and a Jupyter notebook with python code to load data, preprocess, analyze, and visualize in their final submission.

3.2 Group 1: Repurposing Cancer Drugs

The goal of this project was to investigate overlaps between gene mutations, chemotherapy drugs, and primary cancers in order to find which existing drugs for particular cancer could be used to treat other cancers based on a similar gene profile. Although there are many chemotherapy drugs available, not every type of cancer has a specific drug to treat it. While survival rates in some cancers have improved dramatically over time, this is not the case for other cancers such as biliary tract cancer. Based on the discovery that certain mutations may be highly associated with a range of primary cancers it may be possible to investigate whether the drugs for breast cancer could treat pancreatic cancer [6]. For example, BRCA gene mutations are known to be related to cancers of the breast, ovarian, pancreas, and prostate. Students in this group used data from DrugBank and National Cancer Institute Genomic Data Commons. Through their analysis, they discovered associations between cancer sites and their gene expressions that could suggest drugs that exist for one cancer that could be considered for other cancers with few approved drugs (Figure 1).

Table 1: Group 1 presentation of results

week	topics and deliverables
1	Overview Introduction to Python and Jupyter notebook
2	Python data structures, numpy, pandas data frames, data analytics methodology **Case study:** what impacts used vehicle price? [11] **Assignment 1:** functions, strings, lists, arrays, and loops in Python
3	Data pre-processing: missing values, format and type conversion; project topic brainstorming **Case study:** analyzing gender bias based on google search phrases [12] **Assignment 2:** data pre-processing and project topics
4	Data pre-processing, drawing charts, explorative data analysis **Assignment 3:** data pre-processing
5	Histogram, filtering **Case study:** do first babies tend to arrive late? [4] **Assignment 4:** world cuisine analysis [2]
6	Working with image data, convolution, filters; PMF, CDF, ANOVA, correlation **Project 1:** project problem definition and data requirement
7	Introduction to machine learning, supervised vs. unsupervised learning, linear regression **Assignment 5:** image processing **Project 2:** project data collection and preprocessing
8	Multiple linear regression, polynomial regression, pipeline, validation, evaluation, training-test split **Lab:** regression model for pricing used vehicle **Exam 1**
9	Best polynomial, regularization, grid search; Intro to classification, naïve Bayesian, decision tree **Case study:** can we diagnose cancer from biopsy? **Assignment 6:** Scikit-Learn
10	GINI coefficients, SVM **Labs:** decision tree for kyphosis prediction [1], SVM face recognition **Assignment 7:** feature engineering, naïve Bayesian classification
11	Intro to clustering, k-means clustering **Lab:** handwritten digit recognition **Project 3:** data preprocessing, exploratory analysis, preliminary analysis
12	Working with text data, natural language processing, intro to neural network **Lab:** sentiment analysis **Exam 2**
13	Neural network **Lab:** handwritten digit recognition using neural network **Assignment 8:** SVM classifier and k-means clustering
14	**Project 4:** presentation
15	**Project 5:** self and peer evaluation **Project 6:** final submission with full analysis and report

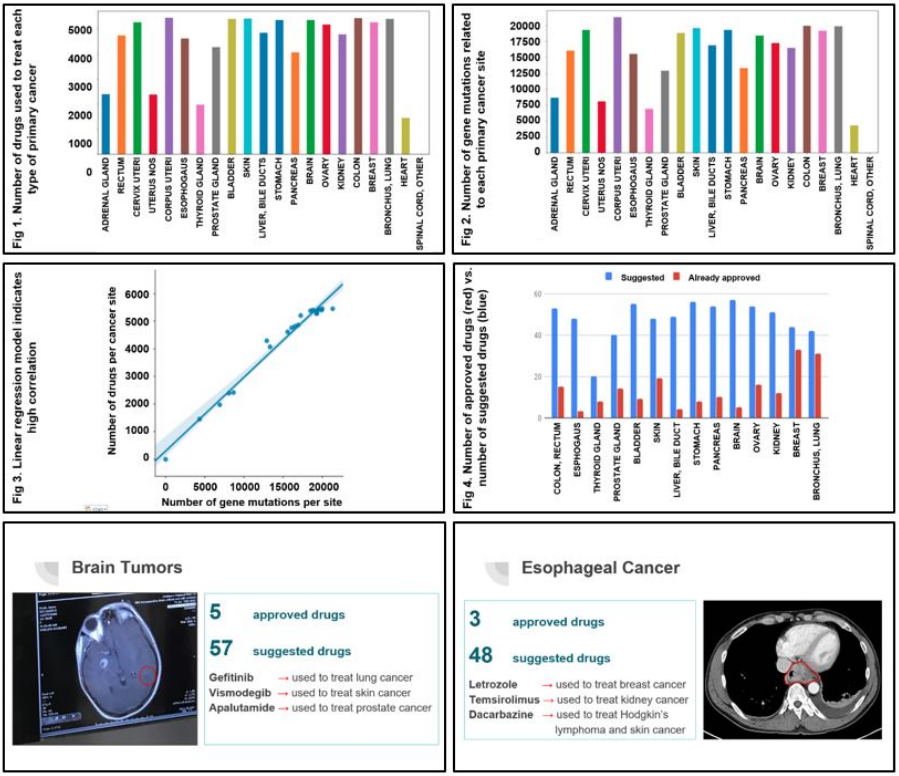


Figure 1: Group 1 presentation of results

3.3 Group 2: Weather-related Factors for Vehicle Accidents

Weather is known to greatly affect driving conditions [10]. This group built predictive models using weather and vehicle crash data to investigate the statement by the US Department of Transportation, "On average, there are over 5,891,000 vehicle crashes each year [9]. Approximately 21% of these crashes - nearly 1,235,000 - are weather-related." The factors included in the models were temperature, precipitation amount, wind speed, relative humidity, and time of day to predict whether or not a vehicle crash occurred. The vehicle crash data was from Montgomery County in Maryland. The group was surprised to find that precipitation was not a significant factor in crashes but that the time of day (rush hour or not) was. This contradictory finding may be due to the fact that the data they used to build the model was from a single county and it might not represent all types of road and driving conditions.

Figure 2: Extracts from group 2 presentation

3.4 Group 3: Predicting Success/Failure of Business

This group's goal was to determine the factors that make a company successful. They used open data 500 companies and attempted to find the factors highly correlated to the success of a company. There were many challenges for this project. In particular, the dataset with sufficient diversity and details could not be found. For example, the majority of the companies in the dataset were b2b (business-to-business) or b2c (business-to-consumer) and there were few companies of different business models. Most companies were young, small in size (number of employees), tended to be in technology or finances, and headquartered in California or New York. There were also numerous missing values that students had to spend considerable effort and time to search for and manually input.

They built regression models, Gaussian naïve Bayesian models, support vector machine models, and neural network models using their data but found a mostly low correlation between factors and success/failure. Although the students in the group were disappointed that they did not find significant factors for success/failure, they recognized that the challenges they faced in working on this project gave them a deeper understanding of the importance of data requirement and the difficulty of data preparation. As one of the reviewers pointed out, it was also possible that there did not exist any statistically significant correlation between the factors included in the models and the outcome of the company.

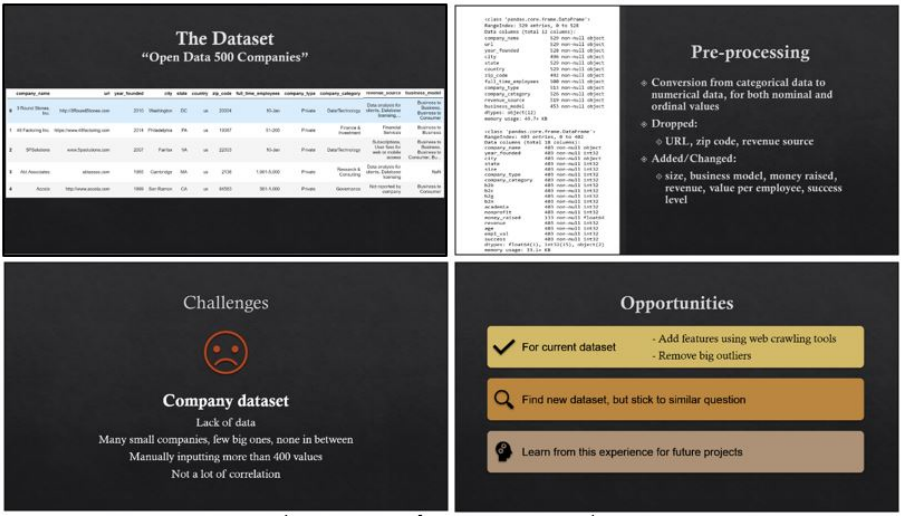


Figure 3: Extracts from group 3 presentation

3.5 Group 4: Fruit Recognition

The ultimate goal of this group's project was "automatic farming: automatically monitor and detect when to water, fertilize, and harvest with live monitoring with cameras." In the course, the group focused on a small but important step of the overall goal, fruit identification from images. The original dataset, Fruits-360, from Kaggle included 81 fruits, each with 360 images (from 0 359 degrees). Since the students were new to computer vision, they focused on a subset of 6 fruits: oranges, lemons, strawberries, bananas, pomegranates, and avocados.

The images were clean and uniform and did not require transformation but this also posed a serious drawback that they were not the typical fruit picture in the real farming environment whether from stationary cameras or drone-mounted cameras. The models they trained with the images in Fruits-360 were not able to reliably classify when they were tested on actual photos of the fruits in various stages of ripeness.

3.6 Group 5: Detect and Remove Fillers in Speech

This group was interested in reconstructing speeches without fillers by detecting fillers from a voice recording and reconstructing a new recording without them. They limited fillers they would remove to "uhhh", "umm", and "like." The group members created their own recordings of short sentences and used an existing library, Librosa, to process their recordings. The pre-processing steps extracted from their final presentation are shown in Figure 5. Upon building and evaluating different models (k-means clustering, spectral clustering, Gaussian mixture clustering, Gaussian naïve Bayesian classification, logistic regression, and kernel SVM) to detect fillers, they found that the kernel SVM model had the highest accuracy with a score of 96.47% in filler detection. The group's final submission included a user interface that allowed the user to record a speech and upon submission produced a downloadable recording without detected fillers.

3.7 Group 6: Predicting the Likelihood of Alzheimer Disease

This group attempted to predict the likelihood of Alzheimer's Disease (AD) based on the analysis of family history, medical records, gene expressions, and periodic brain MRI images. They performed ANOVA to compare the number of Alzheimer patients whose father or mother also had AD or dementia and found that more patients had a father who had AD or dementia.

Their preliminary analysis of recent medical history in the form of a word cloud shows the more common recent medical events the AD patients had at the time of diagnosis (Figure 6 left). In some patients whose yearly brain MRI

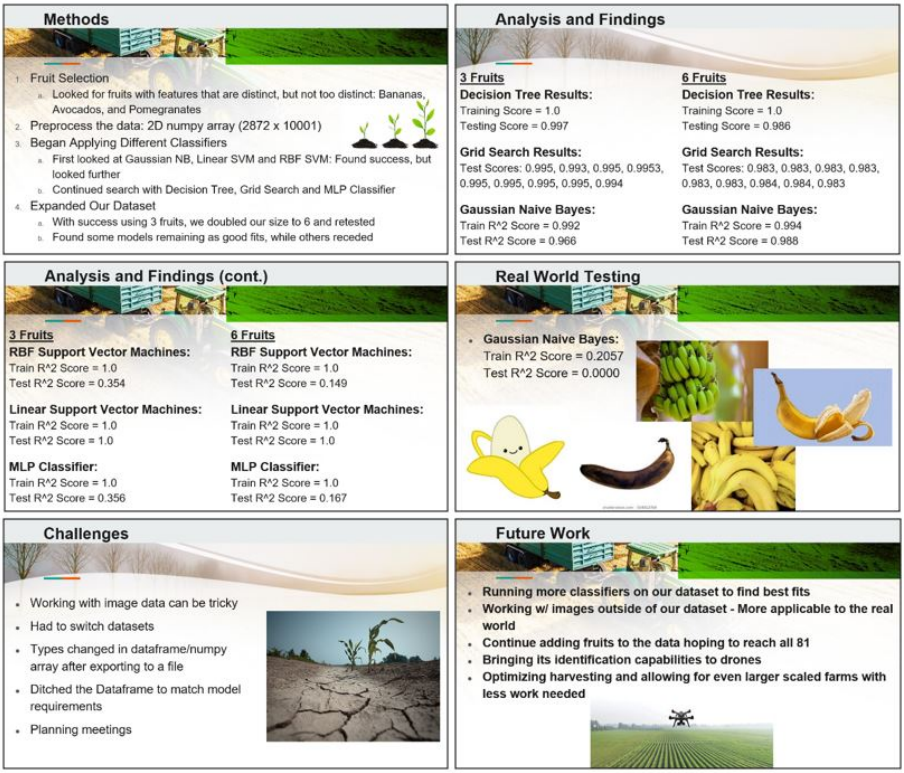


Figure 4: Group 4 presentation of results

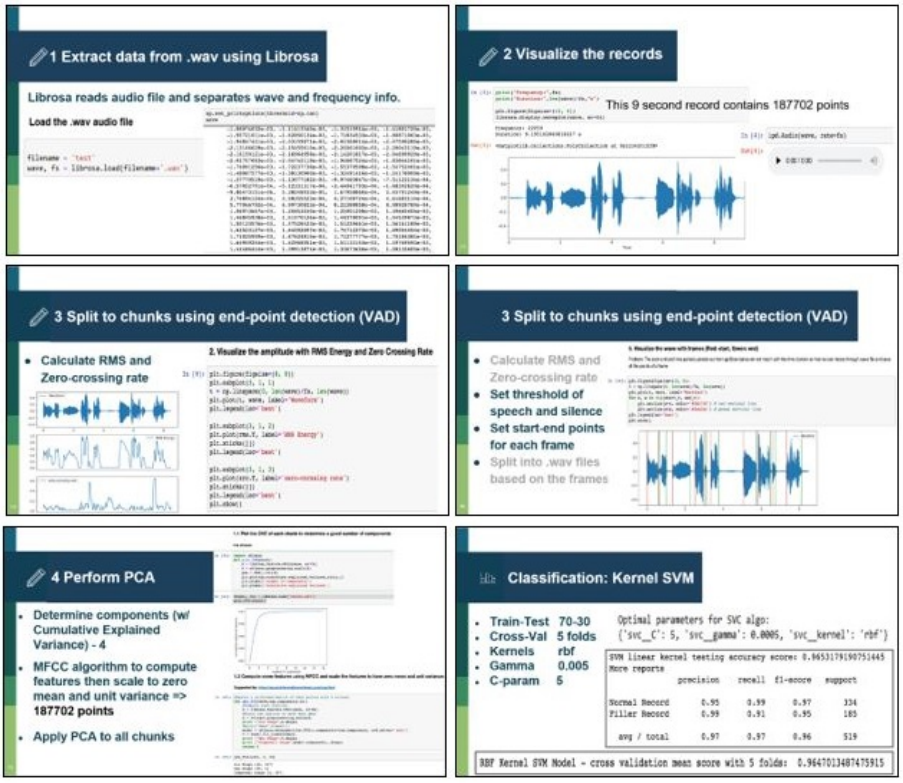


Figure 5: Pre-processing and detecting fillers from a speech recording

scans were available, the changes in the MRI images were clearly visible (Figure 6 right). More analysis is required to build a more accurate prediction model using the brain MRI scans.

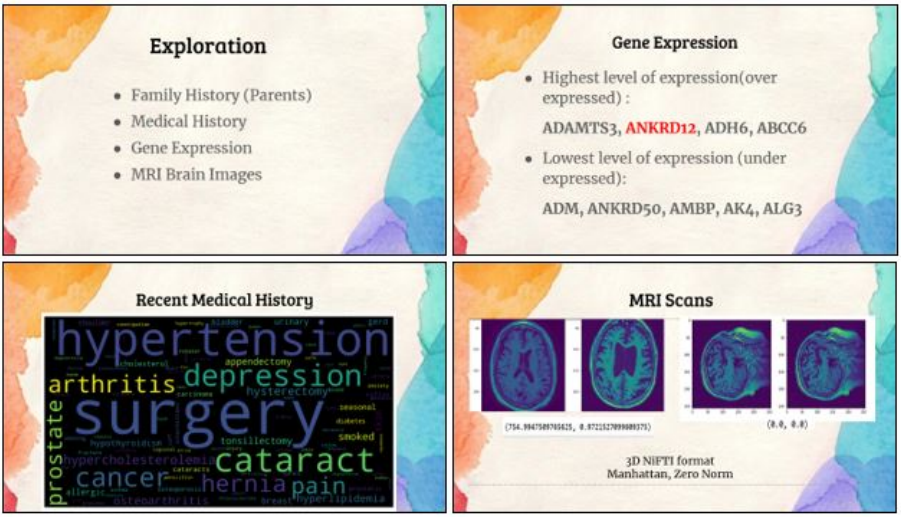


Figure 6: Group 6 presentation of results

4 Student Feedback

Overall student feedback indicates that the course successfully met the learning goals. Students reported that

- their commitment to this course was high (4.3 out of 5),
- they learned a lot (4.38 out of 5),
- they appreciated the course materials (4.15 out of 5), and
- overall the course was excellent (4.53 out of 5).

A sample of qualitative feedback on the course is as follows:

- learned a lot from this course but felt it was too broad in the topic; would have preferred the entire course on regression models with a follow-up course on other machine learning algorithms
- am now more comfortable with python and python data structures
- really enjoyed the class
- the materials in this class really range widely, appreciate the work of the professor trying to catch us up with what are offered currently in the field

- a great course with lots of interesting materials and topics
- enjoyed the topics covered
- this class could influence greatly what I want to do in the future in a positive way
- learned a great deal; knew no python or any data science prior to course
- interesting topic and enjoyed learning about it. Would definitely take another data science class and would like to dig deeper into the subject

5 Discussion

Hicks and Irizarry suggest the following set of general principles when developing a data science course [5]:

- Organize the course around a set of diverse case studies
- Integrate computing into every aspect of the course
- Teach abstraction, but minimize reliance on mathematical notation
- Structure course activities to realistically mimic a data scientist's experience
- Demonstrate the importance of critical thinking/skepticism through examples

Although the publication by Hicks and Irizarry was not referenced directly at the time of design, the structure and activities of the data science course presented in this paper were generally aligned with their guidelines.

The students in the course did not find it difficult to transition to Python as they were already familiar with programming in Java. However, they needed more examples in class and directed assignments before they felt comfortable using the many convenient functions in Python libraries. One of the challenges in designing a single course to introduce data science to undergraduate computer science students was that there were too many important topics to include. More than half of the students had taken data structures in the previous course and had no upper-level course experience and it could not be assumed that all students would have taken a statistics course e. In fact, only a couple of students in the course had taken a statistics course from the Economics department. Although an effort was made to strike a balance between breadth and depth of the topics, some students reported that they occasionally felt overwhelmed and thought the pace was too fast. In the future offering, an additional prerequisite of statistics course could be added or a couple of less central topics such as image processing could be dropped for a more deliberate coverage of unfamiliar topics.

Note that this course model can be adapted to include additional topics and case studies, to delve deeper into a subset of topics, or to replace some

of the topics with other topics according to the target student population. The experience of teaching this course provided a useful basis for designing a required core course in the newly created data science minor in SchoolA and the author hopes that this paper can serve as a resource for other faculty who are interested in developing an introductory data science course for students with programming familiarity.

References

[1] Datasets distributed with r. `https://forge.scilab.org/index.php/p/rdataset/source/tree/master/csv/rpart`.

[2] Yong-Yeol Ahn, Sebastian Ahnert, James Bagrow, and Albert-Laszlo Barabasi. Flavor network and the principles of food pairing. *Scientific Reports*, 1, 2011.

[3] Zachary Bennett. Why colleges are offering data science programs. US News, 2020, `https://www.usnews.com/education/best-colleges/articles/why-more-colleges-are-offering-data-science-programs`.

[4] Allen B. Downey. *Think Stats, Explorative Data Analytics in Python.* Green Tea Press, 2014.

[5] Stephanie Hicks and Rafael Irizarry. A guide to teaching data science. *American Statistics*, 72(4):382–391, 2018.

[6] Carla Mottini, Francesco Napolitano, Zhongxiao Li, Xin Gao, and Luca Cardone. Computer-aided drug repurposing for cancer therapy: Approaches and opportunities to challenge anticancer targets. *Seminars in Cancer Biology*, 68:59–74, 2021.

[7] Tanya Mulvey, Evren Esen, Jennifer Schramm, and Kathya Scanlan. *Jobs of the Future: Data Analysis Skills.* SHRM Survey Findings, 2016.

[8] Bureau of Labor Statistics. Fastest growing occupations. `https://www.bls.gov/ooh/fastest-growing.htm`.

[9] US Department of Transportation. How do weather events impact roads? `https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm`.

[10] Doyle Rice. Surprise: Rain is the deadliest weather driving hazard. USA Today, 2015, `https://www.usatoday.com/story/weather/2015/05/14/deadly-driving-hazards/27300165/`.

[11] Jeffrey Schlimmer. Automobile data set. `https://archive.ics.uci.edu/ml/datasets/automobile`.

[12] Seth Stephens-Davidowitz. Google, tell me. is my son a genius? New York Times, 2014, `https://www.nytimes.com/2014/01/19/opinion/sunday/google-tell-me-is-my-son-a-genius.html`.

[13] Jake VanderPla. *Python Data Science Handbook.* O'Reilly, 2016.

Fostering a Sense of Belonging in Female Computer Science Students*

James Geller
Department of Data Science
New Jersey Institute of Technology
Newark, NJ 07102
james.geller@njit.edu

Abstract

Female students are underrepresented in Computer Science degree programs at US colleges. This problem has resisted interventions for decades and average enrollment rates are stubbornly hovering around 18%, several success stories at selected institutions notwithstanding. Solutions to this problem require bringing in more female students and once enrolled keeping them in the program. To achieve the latter, improvements have been attempted at the curricular and pedagogical level on one hand, and on the social and community building level on the other. This paper describes a low cost approach to building a sense of community and belonging by making a Women in Computing club more "official" using a "token of belonging." Membership is made more rewarding by the promise of a conference trip

1 Introduction

The Computer Science Department at NJIT has been part of the BRAID multi-university consortium of 17 institutions with the declared goal of increasing the representation of women in Computer Science degree programs. (Regrettably, BRAID was sunsetted on May 18, 2021.) The consortium defined a multi-pronged approach dealing with curricular and pedagogic issues on one hand, and with community building, social capital creation, and fostering a sense of

belonging on the other hand. A good method for community building and creating a safe space for female students is the establishment of a club specifically designated for them. This was achieved early on in the project, and the local Women in Computing Society was strengthened by "cross-listing it" with an ACM-W chapter. Thus members of the local club were encouraged to sign up for the ACM-W chapter, and officers of the local club served as president and vice-president for the national ACM-W club chapter.

Famous female, out-of-town guest-speakers as well as presenters from our own university were invited and hosted by the Women in Computing Society. Before the COVID pandemic changed the realities of education, guest speaker engagements were always combined with pizza parties. Yet we observed a wide range of "reactions" to the club and the club activities. A small core of female students became heavily involved, volunteering to serve as club officers and at club events. A larger group of female students ignored club activities. Random sampling of female students "in hallways and elevators" uncovered that some of them did not know that the Women in Computing Society exists. This happened at a time when a guest speaker of the club was widely announced by bulk email messages and by posters on the wall of the CS building.

In this paper, we address the two issues of female students not seeing intrinsic value in being club members, while apparently not gaining a stronger sense of community when they join the Women in Computing Society. We suggest a preliminary attempt at improving this state of affairs: A "token of belonging" tied to club benefits, especially attendance at a conference.

2 Background

Academic success of female students in Computer Science cannot be achieved by improved pedagogy and curriculum manipulations alone, as long as the environment is not welcoming. A key factor for achieving persistence (retention) as a student is a sense of cultural and social belonging [6, 7, 2]. Feelings of belonging can be engendered by creating a community [10]. This has been investigated, for example, in the wider context of validation theory [8]. An important source of feelings of belonging is the validation provided by peers [1, 12]. Another source of a sense of belonging documented in the literature is the validation by faculty members and even by office staff [9]. The beneficial effect of a sense of belonging does not start in college and is not limited to gender-defined groups. Ethnic groups in high school are equally in need of it, even though differences have been observed between groups [4]. The need for community building has been established in many studies in different contexts.

Practical advice on how to create a sense of belonging is harder to come by, especially when teachers are hurried and peers are themselves lacking a

community that they feel they are a part of. An award-winning teacher [3] recommends five practical approaches:

- Shine a light on each student.
- Foster student identity building.
- Always leave one desk empty.
- Make sure that each child feels chosen.
- Weave social and emotional practices throughout the day.

Unfortunately, except for the "empty desk," which is intended for welcoming a student who might join the class later in the semester, none of these "practical" recommendations is prescriptive enough to inform an instructor in the classroom what to do, without providing further elaboration.

In a wide variety of organizations where a sense of belonging is essential for cohesion and functioning, "physical signs" are used to create a specific member identity. The use of uniforms for soldiers, firemen, etc. is a good example, but uniforms also appear in educational institutions [11]. Some expensive private schools require students to wear school uniforms even today. Richard Feynman describes that he had to wear a gown for dinner every night when he was a student at Princeton University [5].

The "physicality" of the visible signs of belonging inspired us to look for a "token of belonging" that could be used to strengthen the feeling of membership in our female students, connecting them to their peers.

At special college events, T-shirts are often given out to organizing committee members, creating a uniform. Everybody wearing the T-shirts "is an insider." For example, at the International Collegiate Programming Competition (ICPC), T-shirts in two colors are handed out (Figure 1), one color for staff members and a different color for competing participants. This creates a hierarchy and two "levels of belonging." However, the purpose of such "uniforms" is identification as opposed to achieving a feeling of belonging. Such T-shirts are rarely worn outside of the events where they are required. Thus, T-shirts were not a solution for us.

3 Methods

Working since 2015 with student members of a college-level Women in Computing Society, several observations were made about subgroups of students, reflecting different attitudes toward the club. We will distinguish between the local club and the ACM-W chapter, as they bring out different reactions from the students. One difference between these two organizations is that our local club does not charge the students a membership fee, while the ACM-W is an international organization that, at the time of this writing, charges an annual

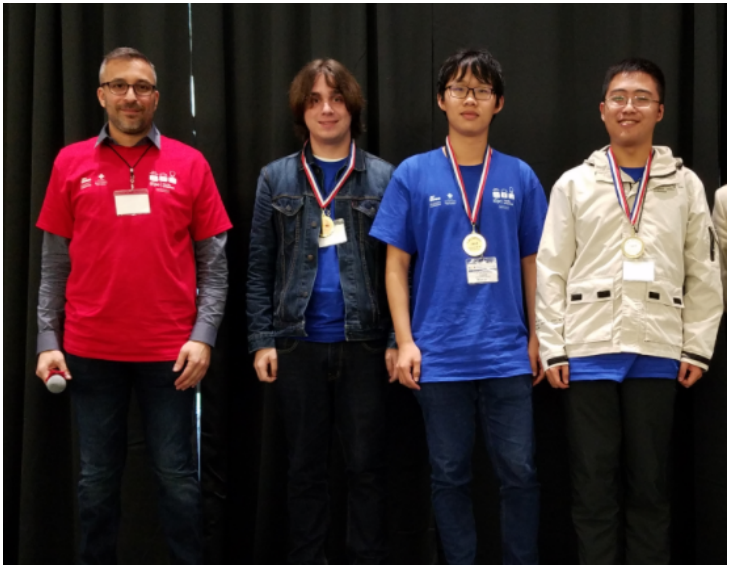


Figure 1: ICPC T-Shirts for Staff (left) and for three Participants.

fee of $19. The second difference is that the ACM-W requires students to sign up on the ACM website. For the local club no such requirement exists. Little can be said about the group of students that is potentially the largest, the "silent majority." We have to focus on students who speak up. Two major groups emerged.

- Subgroup 1: This group can be characterized by the question "Why should I join"? Members of this group of students do not think that community in itself is of any value to them. They do not see any advantages in fostering a closer connection with like-minded students in the local club. They are also not impressed by belonging to a world-spanning organization such as the ACM. They don't see any value in discounts for conference registration fees and academic publications for ACM members, as they have no conception that these would become valuable at a later stage in their education and career. Neither do they understand that advanced levels of membership can only be achieved after being a "basic member" for a few years and that "advanced memberships" (e.g., senior member or fellow) can be included in their resumes and increase their marketability. Finally, some students were reluctant to pay the ACM membership fee. To encourage such students to become members of the ACM-W, grant money was made available to cover the $19 membership

fees for them. Yet the simple tasks of clicking on a registration link and entering basic personal information seemed prohibitive for at least some of the students, who did not sign up after repeated email reminders.

- Subgroup 2: "How do I become a member of the NJIT Women in Computing club"? We used to tell students of this group that there was nothing they had to do. They were women, and they were Computer Science students. By dint of these two facts, they were automatically members of the club. They should just attend the programs and events of the club, without any limitations. The impression we received from this group was that "it was too easy." Members of this subgroup did not ask "why become a member" and appeared to understand the advantages of belonging to a community on an instinctive level. They were not bothered by the same issues as Subgroup 1. Rather they wanted "evidence" that they were members.

Thus, we wanted to find a method that was more concrete than "Make sure that each child feels chosen" (Bullet 4 in Section 2) and that gave the students of Subgroup 2 "proof" that they were members, but that did not impose obstacles, such as the effort of filling out a membership form, on the students of Subgroup 1. Furthermore, we needed to give Subgroup 1 a good reason why they should want to be members, at least in the local club, if not in the ACM-W chapter.

We had one powerful incentive at our disposal. Over the last few years, we have sponsored trips to the Grace Hopper Celebration/Conference for groups of our female students. Registration fees, hotel rooms, and airline tickets were all paid for, for close to 20 students each year. In 2020, when the Grace Hopper Conference was held as a virtual event, we were able to pay the registration fee for 50 of them. Once word got around that this opportunity existed, the demand for "tickets" exceeded the 50 that we had available, and students who were left out started asking about "next year" (2021).

As a result, these are the methods that we are using this year to increase interest and participation in the Women in Computing Club. Working through the eBoard (the officers of the club) we spread the "rumor" that only members of the club would be eligible for free registration at Grace Hopper, which is virtual again in 2021. This was intended to address the thinking of Subgroup 1 of students who felt they did not need to belong to the club, because it did not provide any obvious benefits.

To satisfy the need of Subgroup 2 to have "proof" that they are members of the club, without imposing the need to fill out a signup form (which would have discouraged Subgroup 1), we introduced an ID Card for members. All students had to do was to inform their eBoard officers that they wanted an ID. The eBoard provided us with a list of names and email addresses and we

Figure 2: ID Card Sample for the Women in Computing club.

ordered the cards for them.

The logo on the card had been created by previous club members, who had graduated several years earlier, for use on flyers and posters of the club. We offered the current eBoard the chance to either design a new logo or to organize a competition among the members for designing a new logo, however, the president of the club declared herself satisfied by the existing logo combining a power button, the biological symbol for females, and a C++ style multi-line comment with the symbols /* and */ with the name of the club between them (Figure 2). Cards are of the same shape and size as our college ID cards and can be worn hanging on a lanyard or chain.

To alleviate the concerns of the Campus Police Department (that we consulted) that the cards could be used as fraudulent identification, we made it a point not to include the name of the university on the cards. Figure 2 shows an example card, with the student name partially obscured for privacy reasons.

After the cards were printed, we sent email messages to all the students that due to COVID-19 we would not come to campus, but if they provided us with their home addresses we would be happy to mail them the IDs.

4 Results

A total of 39 students requested ID cards. It was encouraging for us to see that three of those students, who missed the original deadline, later wrote in a "desperate tone" that they really wanted one. We accommodated them, and we had clearly created interest in this "token of belonging."

As a side remark, we also found it encouraging that the commercial company printing the ID cards requested a signed letter from the university to verify that our order was legitimate. After we pointed out that the university name does not appear on the card, they settled for a letter from the Dean of the College. This encouraged us to put the name of the college on the card.

When we requested the students' home addresses, 32 students responded on short notice. (Most others followed soon after.) We speculate that had we asked for their home addresses at the beginning of the process, this would have deterred some of them from asking for a card, but once we had a good reason (COVID and mailing), students were forthcoming with the information. In the process we discovered that one of the students was in India and we decided not to send the card to her.

Our students are already "over-questionnaired" and we did not want to impose yet another survey on them. On the other hand, we wanted to get minimal feedback to see how much they cared about their new IDs. For this purpose, we sent an email message with a single question to them, and we made it a point NOT to send a follow-up message to students who did not respond the first time. Only students who cared enough to respond to the first message were of interest to us. We received 16 responses, most of them within a few hours of sending the question.

As Figure 3 shows, 62.5% of the respondents stated that they agreed or strongly agreed that the membership card made them feel more connected with "Women in Computing." Not a single student disagreed strongly with the statement, and 31.3% felt neutral about it. One student (6.2%) disagreed with the statement.

5 Discussion

We presented a preliminary study concerning the advantages of a "token of belonging" to a club for Women in Computing. In a year when no pizza parties could be held and all guest speakers of the Women in Computing Society gave their presentations on Zoom or WebEx, a physical connection appeared especially important. We intend to repeat the ID Card provisioning at the beginning of future semesters, as long as there is a budget available for it. We will actively involve the incoming eBoard of the club at the beginning of the fall

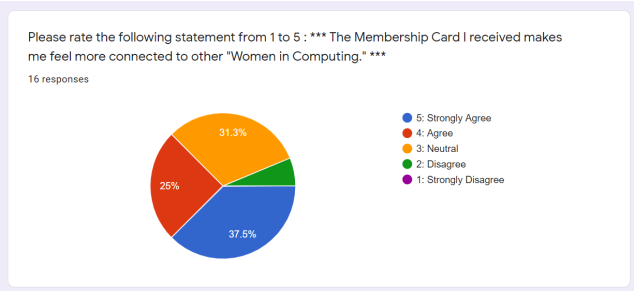


Figure 3: Responses of 16 of the card recipients.

semester and discuss variant ideas, e.g., creating cards in two different colors to distinguish high-achieving students or eBoard members.

Much of this work was informed by anecdotal evidence. In the future, we would like to use a questionnaire in a physical setting ("after a pizza party") to get a better understanding of the perceptions of the students and their attitudes towards club membership.

Many students wear the official university ID cards on a lanyard around their necks, to use the swipe card access to locked rooms that these cards provide. Once on-campus activities will be ramped up again, we will be looking out for students who also wear their Women in Computing club ID cards for all to see. This might further increase interest in club activities among their peers.

6 Conclusions

We presented the reasoning behind the introduction of a "token of belonging" for members of a Women in Computing club. A special ID card was printed at no cost to the students and with absolute minimal effort to "sign up for it." We spread the "rumor" that only club members would be eligible for free attendance at the virtual Grace Hopper Celebration. We are encouraged by the interest of the students and intend to continue with this initiative at the beginning of every future semester.

Acknowledgments

We thank Lisa Ryder for providing material support to this initiative.

This work was supported in part by the National Science Foundation under grant DRL-1837489. Any opinions, findings, conclusions or recommendations are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] A. M. Martinez Aleman. Race talk: Undergraduate women of color and female friendships. *The Review of Higher Education*, 23(2):133–152, 2000.

[2] R. J. Benbow and E. Vivyan. Gender and belonging in undergraduate computer science: A comparative case study of student experiences in gateway courses (WCER Working Paper No. 2016-2), 2016. `http://www.wcer.wisc.edu/publications/working-papers`.

[3] M. Dunlea. Every student matters: Cultivating belonging in the classroom. Edutopia, September 4, 2019. `https://www.edutopia.org/article/every-student-matters-cultivating-belonging-classroom`.

[4] B. S. Faircloth and J. V. Hamm. Sense of belonging among high school students representing 4 ethnic groups. *Journal of Youth and Adolescence*, 34(4):293–309, 2005.

[5] R. Feynman. *Surely you're joking Mr. Feynman.* W.W. Norton Company Inc., New York, NY, 1985.

[6] K. Garvin-Doxas and L. J. Barker. Communication in computer science classrooms: understanding defensive climates as a means of creating supportive behaviors. *Journal on Educational Resources in Computing*, 4(1):1–18, 2004.

[7] J. Margolis and A. Fisher. *Unlocking the clubhouse: Women in computing.* MIT Press, Cambridge, Massachusetts, 2002.

[8] L. I. Rendon. Validating culturally diverse students: Toward a new model of learning and student development. *Innovative Higher Education*, 19(1):33–51, 1994.

[9] P. Schuetz. A theory-driven model of community college student engagement. *Community College Journal of Research and Practice*, 32(4):305–324, 2008.

[10] D. B. Tatum. *Can we talk about race? And other conversations in an era of school resegregation.* Beacon Press, Boston, Massachusetts, 2007.

[11] G. Whitby. The pros and cons of school uniforms, March 24, 2017. `https://www.dailytelegraph.com.au/newslocal/central-sydney/school-uniform-pros-and-cons/news-story/cd111d0c459ffd4f53acd03d80b4ed9c`.

[12] C. Zhao and G. D. Kuh. Adding value: Learning communities and student engagement. *Research in Higher Education*, 45(2):115–126, 2004.

Analyzing Brain Tumor MRI to Demonstrate GPU-based Medical Image Segmentation*

Sajedul Talukder
Southern Illinois University
Carbondale, IL, USA
sajedul.talukder@siu.edu

Neil Noyes
Edinboro University
Edinboro, PA, USA
nn180872@scots.edinboro.edu

Abstract

Image processing, a computationally intensive task must be done quickly, efficiently, and painlessly in a variety of medical imaging applications to assure quality for both patients and clinicians. Graphics Processing Units (GPUs) have been increasingly used in medical image processing in recent years due to their high efficiency and parallel capabilities. In this paper, we compare the performance of several GPU-based image processing algorithms used in medical imaging against their CPU-based counterparts on a quantitative basis. To illustrate the possibilities of GPU segmentation on a brain MRI containing a significant brain tumor, we investigate an NVIDIA Clara-driven GPU segmentation extension within the program 3D Slicer.

1 Introduction

One of the most essential forms of computing technology, both for consumer and corporate computing, is the graphics processing unit (GPU) [13]. The

GPU was originally created to speed up the rendering of 3D graphics, but it is now utilized in a variety of applications, including graphics [13], digital automation [20, 12, 23] and video rendering. GPUs are becoming more popular for use in artificial intelligence (AI) [3], cybersecurity [24, 5, 22], and medical image segmentation [18, 25], despite their best-known capabilities in gaming and creative production [6]. Image segmentation, or the division of an image into disjoint sections based on image characteristics such as color detail, strength, or texture, is one of the most fundamental tasks in computer vision and image processing, and it is essential for many applications such as object detection, labeling, and recognition [14, 21]. Image segmentation is regarded the most important medical imaging procedure since it extracts the area of interest (ROI) using a semiautomatic or automated procedure. Picture segmentation is the process of dividing an image into ROIs based on a description. This is generally used for boundary detection, tumor detection/segmentation, and bulk detection by segmenting bodily tissue. Due to the enormous number of applications for image processing algorithms in the medical profession, there is a community committed to developing more efficient and speedier solutions.

The medical field is an excellent example of where rapid and efficient imaging techniques are required. Medical imaging is now used in a variety of clinical applications, ranging from medical scientific research to diagnoses and therapy planning. However, due to the huge three-dimensional medical datasets to analyse in real clinical applications, medical imaging techniques are typically computationally intensive. In comparison to traditional CPU-based computing frameworks, GPU computational speed has risen so quickly in recent years that it now serves as a better platform for considerable acceleration for many computationally complex activities.

Previous research in this field reveals that GPU-based picture segmentation and processing is far more efficient than CPU-based computing frameworks. We focus on the advantages that GPU-based image processing may provide when employed in a variety of medical applications, and we provide quantitative reasoning to propose the best strategy. We also implement an extension for the program 3D Slicer [1], a free, open-source, and multi-platform software package widely used for medical, biomedical, and related imaging research, to explore the capabilities of this computing framework by combining GPU-based computing with NVIDIA AI-Assisted Annotation (AIAA) [2].

Our Contributions. This paper presents the following contributions:

- **Survey Brain Image Segmentation**. Review the literature on image segmentation within the medical field, considering the GPU as the primary computation brain.

- **Segmentation Approaches**. Provide a comprehensive overview of the

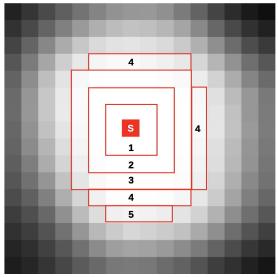


Figure 1: Parallel region growing with double buffering with S as the seed pixel. The numbers indicate at which iteration the pixels in the red regions are added to the final segmentation.

low-level and high dimensional medical image segmentation approaches that involves medical image processing such as image registration, segmentation, denoising, filtering, interpolation, and reconstruction.

- **GPU Segmentation Implementation: NVIDIA AIAA**. Implement an extension for the program 3D Slicer with AI-assistive technology (NVIDIA AIAA) to demonstrate the capabilities of GPU segmentation on a brain MRI consisting of a large brain tumor.

The rest of the paper is organized as follows. Section II presents the literature review that closely relate and contribute to our research. Section III discusses the image segmentation approaches. Section IV presents our implementation of the GPU segmentation using NVIDIA AIAA. Finally, Section V concludes the paper with a highlight on the scope of future work.

2 Related Work

There are lots of resources on image segmentation within the medical field, even when considering the GPU as the primary computation brain. The following are a few key examples that closely relate and contribute to our research. Sharma et al. [15] proposed automated medical image segmentation techniques. This related work deals with the differences, benefits, and limitations among MR (magnetic resonance) vs CT (computed topography) imaging and how they need to be effective for different types of application. Segmentation of a brain is much different from the segmentation of a thorax, so there needs to be clarity on which type of scan/segmentation method is used where and how the results will best reflect the desired outcome.

Taha and Hanbury [19] proposed metrics for evaluating 3D medical image segmentation. Since we are focusing on the efficiency of image segmentation algorithms, we need to clearly define ways in which to confirm quality. Comparing images to evaluate quality is essential to measuring progress within the space, and that is exactly what this related work shows us. The challenges in evaluating medical segmentation are: metric selection, the use in the literature of multiple definitions for certain metrics, inefficiency of the metric calculation implementations leading to difficulties with large volumes, and lack of support for fuzzy segmentation by existing metrics.

Di Salvo et al. [7] presented image and video processing on CUDA. The research proposed here argues for the use of CUDA for various applications of image and video processing. Since GPUs boast simple, data-parallel, deeply multi-threaded cores and high memory bandwidths, fields such as the medical industry can take advantage of the vast amount of efficiency that the platform offers. Using CUDA allows one to achieve very high-performance in time processing while also keeping the same performance in terms of accuracy.

Eklund et al. [8] presented a survey on the medical image processing on the GPU. This research delves into how graphics processing units (GPUs) are used in a wide range of applications. Since GPUs can dramatically accelerate parallel computing, are affordable as well as energy efficient, they can serve crucial for enabling practical use of computationally demanding algorithms used within the medical field (specifically with imaging). This review not only covers GPU acceleration on common image processing operations (such as filtering, interpolation, histogram estimation and distance transforms) but also goes over the most commonly used algorithms in medical imaging and how they are used toward individual scans.

Similarly, Shi et al. [17] presented a survey of GPU-based medical image computing techniques. The goal of this research was to provide a comprehensive reference source for the starters or researchers involved in GPU-based medical image processing. This work goes over the existing traditional applications within the three main areas of medical image processing: segmentation, registration, and visualization. As well as giving an overview of techniques used in application, we see a discussion of the advantages with GPU-based medical image visualization approaches.

Bankman [4] provided an introduction to segmentation. Within the medical field, imaging algorithms need to be able to handle variability. The data collected is from real people each with a unique condition. That is precisely what this piece of research addresses and reveals techniques used to combat data variation. Fuzzy clustering, the use of deformable models as well as volumetric data are discussed. Volumetric data deals specifically with 3D segmentation and material analysis which is useful for GPU-based imaging techniques.

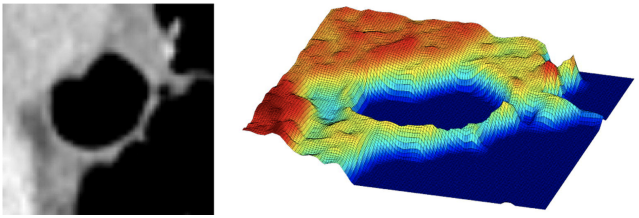


Figure 2: The watershed approach, where intensity is interpreted as height within the topological representation.

Kafieh et al. [9] presented a review of algorithms for sgmentation of optical coherence tomography from retina. This research focuses on the different types of algorithms that can be used for the segmentation of optical coherence tomography. Optical coherence tomography (OCT) is a technique to describe different information about the internal structures of an object and to image certain aspects of the biological tissue. This paper reads of the history of the algorithms used within the field and shows how the medical imaging field has grown over the years relating to the precision, abilities, and accuracy of their results.

Smistad et al. [18] provided a comprehensive review of how the GPU can solve problems with segmentation of anatomical structures. Since segmentation of imaging from CT, MRI and ultrasound is key for diagnostics, planning and guidance, more efficient solutions become required as time goes on. This paper goes over the essentials of GPU computing, segmentation algorithms, and a proposal of the future for GPU hardware manufacturers.

3 Image Segmentation Approaches

GPU's, although originally created for rendering graphics, have become popular for lots of high-performance computation due to the more recent programmability and low cost to performance. When compared with a CPU-based computing framework "...the difference in theoretical performance can differ by a factor ten in favor of the GPU" [8]. While both modern CPU and GPU can manage thousands of threads, the GPU running a suitable algorithm may progress over thousands of threads concurrently, while the CPU progresses over less than a dozen (generously). One of the fields of which capitalized upon this was the medical field within their medical image processing/segmentation. Since there is a strong nature of parallelism among GPU computation, performance often is tied to the implementation of parallel adapted algorithms.

This bears fortunate since image processing algorithms are often tremendously parallel by nature, which makes it easier to implement on a GPU [8]. Medical image processing runs on the backbone of a patient's available data, which is steadily increasing for each case, resulting in the need for fast algorithms that excel with accuracy as well as speed. There are many different types of algorithms involved medical image processing such as image registration, segmentation, denoising, filtering, interpolation, and reconstruction [18]. This research will focus mainly on the algorithms involved within image segmentation specifically. The first GPU-based medical image segmentation technique was an attempt to exploit the high performance of graphics cards for different numerical computations by formulating level set segmentation as a sequence of graphics operators of image blending [17]. Another technique, which used the then recent programmable shader technology, used the enhanced flexibility to enable 3D image segmentation using curvature regularization to favor smooth iso surfaces. Among other methods, including ones implemented with CUDA, we can generalize with two types of segmentation approaches: low-level and high-level. Low-level approaches, such as GPU-based implementations of watershed and region growing methods, require no statistical information about the types of objects within the image being segmented and directly manipulate the voxel/pixel information to then form connected regions of interest (ROI's). This is contrast to high-level segmentation frameworks, which do not directly manipulate the pixel/voxel information from the image. There has been work proposed in this area using geodesic active contours as well as contours with gradient vector flow [10] as well as CUDA powered approaches. Since CUDA was released, there has been high improved performance in terms of speedup with respect to the sequential version of various image segmentation algorithms using CUDA and CUDA-enabled GPUs [7]. Since the medical field needs real time, high dimension segmentation approaches, CUDA seems to be the go-to implementation going forward to deal with the very large data sets the medical imaging scans provide. CUDA does not operate proficiently without optimized algorithms though, so one needs to make the right decision with respect to choosing the most fitting algorithm to for the application.

3.1 Interacting Seeded Region Growing

The seeded region approach for image segmentation was designed to be robust, rapid and have no tuning parameters. Instead of the tuning parameters, the algorithm requires the input of a number of seeds (as either pixel or voxel regions) that control the formation of regions that the image will be segmented. A gray-scale volume image is usually a digital cubic grid $G = (V, E)$ where the vertices V are valued by a function $g : V \to [h_{\min}, \ldots, h_{\max}]$ with $V \subseteq \mathbb{Z}^3$ the domain of the image and $h_{\min}$ and $h_{\max}$ the minimum and the maximum

gray-value. A digital grid is a special kind of graph $G = (V, E)$ defined by a set V of vertices and a set $E \subseteq V \times V$ of pairs defining the connectivity. If there is a pair $e = (p, q) \in E$ we call p and q neighbors, or we say p and q are adjacent. The set of neighbors $N(p)$ of a vertex p is called the neighborhood of p. Usual choices are the 6-Connectivity, where each vertex has edges to its horizontal, vertical, front and back neighbors, or the 26-Connectivity,where a point is connected to all its direct neighbors. The vertices of a cubic digital grid are called voxels. Once the seeds are set, either using a GUI or automatically with prior knowledge, we use those seeds as start nodes to then grow our region of interest based on if neighboring pixels fit the predefined criteria. These criteria compare the current pixel to the seed or the already included pixels (once region has grown from seeds) using attributes such as intensity, gradient, or color. The region will grow as long as there are these neighboring pixels that satisfy such condition. The algorithm behaves comparably to a breadth first search algorithm.

As one could imagine, the number of threads grows as the border of the region of interest expands. On CPU-based computing frameworks, changing the number of threads usually is done by restarting the kernel, which makes it so all values from global memory need to be read again. We can, however, use the GPU by having one thread for each pixel in the entire image in each iteration. But doing this adds more computational work due to branch divergence as well as increasing memory usage [18]. Since the traditional CPU-based implementation has too high of a computational cost when images scale larger, CUDA and the GPU have been used to accelerate the seeded region growing approach. One notable approach, respective to this method, proposed to exploit sophisticated graphics hardware functionality (such as floating-point precision, render to texture, computational masking, and fragment programs) to allow the users to interactively paint growing seeds by drawing on the sectional views of the volume. In addition to visualization and interaction, the algorithm leverages the parallel processing capabilities of the GPU to run significantly faster than previous methods [16]. For this approach, it is crucial to draw as many seeded points as possible to make full use of the parallel performances of the GPU.

3.2 Watershed

The watershed approach to image segmentation is based on viewing an image as a 3D object. The height is determined from the intensity of the pixel/voxel. A path $\pi = (v_0, v_1, \ldots, v_l)$ on a grid G from voxel p to voxel q is a sequence of vertices where $v_0 = p, v_l = q$ and $(v_i, v_{i+1}) \in E$ with $i \in [0, \ldots, l)$. The length length (π) of a path π is defined as length$(\pi) = l$. The geodesic distance $d_G(p, q)$ between two voxels p and q is defined as the length of the shortest path $\pi_{\min}$ between p and q, with $\forall v \in \pi_{\min} : v \in G$. The

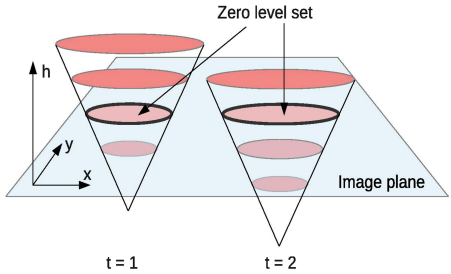


Figure 3: The level-set approach showing the movement through the image plane, which shows a circle that will grow over time.

geodesic distance between a vertex p and a subset of vertices Q is defined by $d_G(p, Q) = \min_{q \in Q} (d_G(p, q))$ [26].

Once the generated landscape is complete, there are three types of points, categorized by how a drop of water would behave at the specific point. These three categories include scenarios in which a drop of water would:

- stay at the point (local minima)
- move downward to into one local minimum
- move downward into > 1 local minimum

The points that are of type 2 are called watersheds or catchment basins and points of type 3 are called watershed lines or divide lines since they split into more than one local minimum. The distinction between types of points is important because the main purpose of this algorithm is to find watershed lines. This then begs the question; how does one find watershed lines? To answer this question, we must look to analogy based on the topological representation. Suppose there are holes in all the places where a point of type 1 (local minimum) occurs. Therefore, water would flow through these holes. The watersheds within the topological representation would then be flooded at a constant rate. So, when two watersheds are about to merge together, something called a dam (which is exactly the same to a dam you think of holding back a large body of water) is built between them, with an increasing height as the water level rises. This is continued until the height of the dam is the same as the highest represented point within the topological landscape. These dams that were created are the watershed lines.

As far as acceleration with parallelism on the GPU, we see an obstacle of constructing good multi-threaded algorithms. The watershed approach has an

inherent sequential nature. However, attempts have been successful in this regard by either transforming the topological landscape into a graph, subdividing the image, or flooding each local minimum in parallel. Overall, these speedups are reported to be only within the range of 2-7x faster. One notable attempt, by Kauffmann and Piche in 2008 [11], presented the watershed segmentation implementation using the cellular automation approach. They defined their algorithm (see Algorithm 1) by using a weighted graph, $G = (V, E, w)$, or by a valued graph, $G = (V, E, f)$, where respectively $w : E \to \Re$ is a weight function defined on the edges, and $f : V \to \Re$ is a valued function defined on the vertices. They used the Bellman-Ford algorithm to calculate a weighted cost of the shortest path from all pixels to each local minimum. Thus, the shortest path will then always lead downwards. By using this approach, they established a method that can process all of the pixels/voxels in an image in parallel using the same instruction, boasting speedups of 2.5x.

Algorithm 1 Watershed Automata evolution rule

for all minima: s_i with $i \in [1, k]$ **do**

$\lambda(s_i) \leftarrow 0$ and for all $v \neq s_i, \lambda(v) \leftarrow \infty$

$\text{label}(s_i) \leftarrow i$ and for all $v \neq s_i, \text{label}(v) \leftarrow 0$

for all $p \in V$ **do**

$U^t = \min_{q \in N_p} \{\lambda^t(q) + f(p)\}$

$\lambda^{t+1}(p) = \min[\lambda^t(p), U^t]$

$\text{label}^{t+1}(p) = \text{label}\{\min[\lambda^t(p), U^t]\}$

end for

end for

Wagner et al. [26] presented a parallel watershed-transformation algorithm that takes a gray-value gradient image $g : \Omega \to [h_{\min}, \ldots, h_{\max}]$ as input and computes a label image $l : \Omega \to \mathbb{N}$ which contains the segmentation result. For each gray-level h, starting at the global minimum $h_{\min}$, new basins are created according to local minima of the current level h. Already existing basins, are expanded if they have adjoining pixels of the gray-value h. The procedure stops when the maximum gray-value $h_{\max}$ is reached. This implementation was 5-7x faster than serial implementation on 3D images [16].

3.3 Level set Approaches/Methods

The level set approaches to segmentation are based on spreading a contour within the input images. We define a scalar function with positive values inside the current segmentation and having negative values outside, thus defining the segmentation boundary by the function's zero level set. This level set function is one dimension higher than the contour, which means that we can start with 2D images, define a boundary at the start and then propagate along the z axis, almost creating a 3D object out of slices, much like a 3D printer works. The level set function in 2D segmentation, the $z = \phi(x, y, t)$, is defined as a function which returns the height z from the position x,y in the image plane to the level set surface at time t. The contour is defined implicitly as the zero-level set, which is where the height from the plane to the surface is zero. At level-set zero, the image plane and the surface intersect. To propagate the contour in the x, y plane, the level set surface is moved in the z direction. How fast and in which direction a specific part of the contour moves, is determined by how the level set surface bends and curves. The closer the surface is to being parallel with the image plane, the faster it propagates. When the level set surface is orthogonal to the image plane, the contour does not propagate at all. Assuming that each point on the contour moves in a direction normal to the contour with speed F, the contour can be evolved using the following PDE:

$\frac{\partial \phi(x,y,t)}{\partial t} = F(x, y, I)|\nabla \phi(x, y, t)|$

The speed function F varies for different areas of the image I and can be designed to force the contour towards areas of interest and avoid other areas. In image segmentation, the speed function is usually determined by the intensity or gradient of the pixels, and the curvature of the level set function. A negative F makes the contour contract, while a positive F makes it expand.

The level set method starts by setting an initial contour on the object of interest. This is done either manually or automatically using prior knowledge. Next, the level set function is initialized to the signed distance transform of the initial contour. Finally, the contour is updated until convergence. Smistad et al. [18] proposed a parallel level sets algorithm as shown in Algorithm 2. The main issue with the level-set approach comes down to the extremely high computational cost when processing large images. A set of numerical simulations must be performed on every voxel of an image, so the limited bandwidth and limited number of execution cores on a CPU are not ideal.

3.4 Active Contours

Active contours, also known as snakes, were introduced by Kass et al. [10]. These contours move in an image while trying to minimize their energy, as

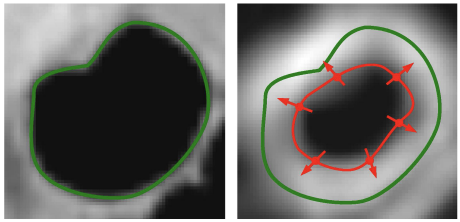


Figure 4: The active contour approach where the left image is the input, and the right image shows the gradient magnitude. The red line superimposed on the right image is the active contour, which is driven towards the high gradient parts of that image, corresponding to the edges in the original image. The green line superimposed on both images show the contour of the lumen.

shown in Fig 4. Active contour segmentation is very similar to that of the level set approach. The major difference between them is that the contour is represented explicitly by a large number of nodes, rather than a mathematical function. They are defined parametrically as $v(s) = [x(s), y(s)]$, where $x(s)$ and $y(s)$ are the coordinates for part s of the contour.

This method can be divided into two different, but still data parallel, operations. The first operation is calculating the external energy. The energy of a contour depends on the shape and is mathematically related to the tension and rigidity of the contour. The energy E of the contour is composed of an internal E_{int} and external energy E_{ext} : $E = \int_0^1 E_{\text{int}}(v,s) + E_{\text{ext}}(v(s))ds$ The internal energy depends on the shape of the contour and can, for example, be defined as:

$E_{\text{int}}(v,s) = \frac{1}{2}\left(\alpha\left|v'(s)\right|^2 + \beta\left|v''(s)\right|^2\right)$ where α and β are parameters that control the tension and rigidity of the contour.

The second is then evolving the contour itself. The contour can be driven towards interesting features in the image, by having an external energy with low values at the interesting features and high elsewhere. There are several different choices of external energy. A popular choice is the negative magnitude of the image gradient, i.e. $E_{\text{ext}}(\vec{x}) = -\left|\nabla\left[G_\sigma * I(\vec{x})\right]\right|^2$, where $G_\sigma *$ is convolution with a Gaussian lowpass filter. The convolution and gradient calculation can be executed in parallel for each pixel, and optimized using texture or shared memory.

A numerical solution to find a contour that minimize the energy E can be found by making the contour dynamic over time $v(s,t)$ $\alpha v''(s,t) - \beta v^{(4)}(s,t) - \nabla E_{\text{ext}} = 0$. The thread count is equal to the number of sample points on the contour, which is much lower than the number of pixels in the image. To evolve

the contour, each point s has to be extracted from the image using interpolation. Thus, active contours may benefit from using the texture memory, which can perform interpolation efficiently.

As shown by Xu and Prince [27], some different formulations of the external force field ∇E_{ext} may get stuck in local minima, especially if boundary concavities are present. They introduced a new external force field, gradient vector flow (GVF), which addressed this problem. The GVF field is defined as the vector field $\vec{V}$, that minimizes the energy function E : $E(\vec{V}) = \int \mu|\nabla\vec{V}(\vec{x})|^2 + \left|\vec{V}(\vec{x}) - \vec{V}_0(\vec{x})\right|^2 \left|\vec{V}_0(\vec{x})\right|^2 d\vec{x}$ where $\vec{V}_0$ is the initial vector field and μ is an application dependent constant. This approach differs from other choices of external energy, which are generally not iterative. GVF is thus more time consuming as many iterations are needed to reach convergence.

Algorithm 2 Parallel level sets

Initial segmentation and input image I Segmentation result S Initialize ϕ to signed distance transform from the initial segmentation

while a number of iterations or until convergence **do**

  for all voxels $\vec{x}$ in parallel **do**

    Calculate first order derivatives

    Calculate second order derivatives

    Calculate gradient $\nabla\phi(\vec{x})$

    Calculate curvature

    Calculate speed term $F(\vec{x}, I)$ $\phi'(\vec{x}) \leftarrow \phi(\vec{x}) + \Delta t F(\vec{x}, I)|\nabla\phi(\vec{x})|$

  end for

  $\phi = \phi'$

end while

for all voxels $\vec{x}$ in parallel **do** $\phi'(\vec{x}) \leq \phi(\vec{x})$ $S(\vec{x}) \leftarrow 1$

  $S(\vec{x}) \leftarrow 0$

end for

return

3.5 GPU Segmentation Implementation: NVIDIA AIAA

There are many tools developed for image segmentation in the medical field, and I would like to conclude my GPU-based image segmentation research by

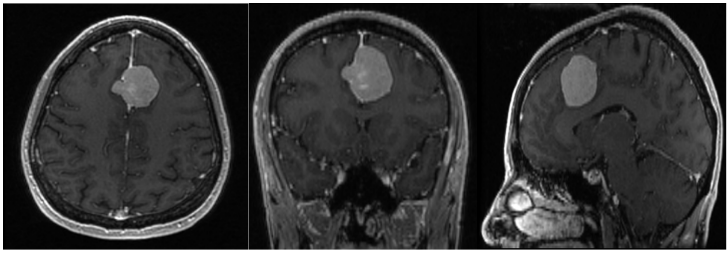


Figure 5: Three views from a contrast-enhanced brain MRI.

implementing a popular and free extension called "NVIDIA AI-Assisted Annotation (AIAA) for 3D Slicer" to demonstrate the capabilities of image segmentation based on the GPU. This extension uses artificial intelligence to train models based on human clinical data sets. Once a scan is initiated, the input scans are sent to a server running linux with an NVIDIA GPU for processing and returns a 3D segmentation result to slicer. For example, this may include a 3D tumor or mass of some kind.

3D Slicer is an open-source software platform for medical image informatics, image processing, and three-dimensional visualization. Mainly written in C++ and based on the NA-MIC kit, 3D Slicer relies on a variety of libraries: VTK, ITK, CTK, CMake, Qt and Python. 3D Slicer consists of both an application core, as well as having modules that offer specific functionality. The core implements the UI, data I/O, and visualization, while exposing developer interfaces for the use of extensions with new modules.

Using this extension, we will walk through an example of segmenting a brain tumor. We are provided a pre-trained ai model for segmenting tumors on contrast-enhanced brain MRI scans, we will be using this for our segmentation.

Step 1. The first step is to load the dataset into 3D Slicer, and once that is completed, we end up having three views from a contrast-enhanced brain MRI as shown in figure 5.

This importing process gives the user a scrollable three panel window displaying the scans. Each of these three views are fully interactable. If you are familiar with MRI scans, we can scroll through them to see all the layers through the scan. This means you can scroll from the very edge of the skull, all the way into the head to get to the meat of the mass we are trying to segment. This is our region of interest.

Step 2. Once this process is complete, we need to go to what Slicer has coined the Segment Editor. The segment editor is where the user adds segmentations to the imaging scans. Since we are using the NVIDIA AIAA extension,

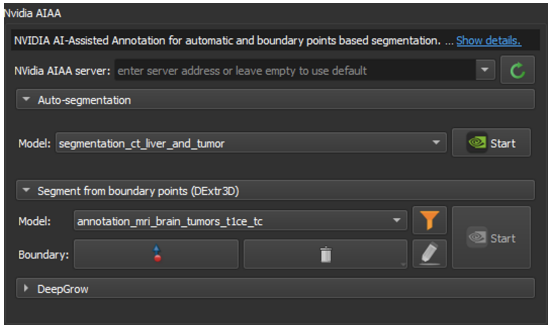


Figure 6: NVIDIA AIAA Slicer's UI.

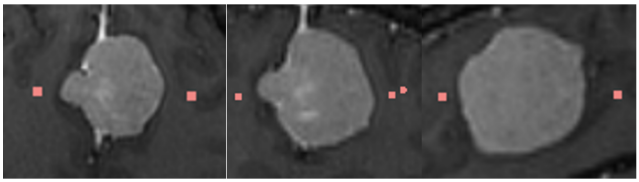


Figure 7: Setting two boundary points on the left and right side of the ROI.

we need to configure the segmentation accordingly. Once inside the segment editor, we add a new segment using the add button in Slicer's UI and add the NVIDIA AIAA effect. Once the effect is added to the segmentation, we get more options in the Slicer UI (see figure 6). We want to configure a segment from boundary points that uses the model "annotation_mri_brain_tumors_-t1ce_tc." This model is trained to segment tumors on contrast enhanced brain MRIs.

Step 3. In each image, we set two boundary points on the left and right side of the ROI, so the algorithm has a base of where to start segmenting, shown in figure 7 as pink squares when placed.

Step 4. After the boundary points have been set on each view of the scan, we are then ready to hit the now ungrayed out NVIDIA start button (when compared with figure 6) to send our data to their server for processing on their NVIDIA GPU powered server. After a few seconds, thanks to the power and capabilities of GPU computing, we get returned to Slicer a fully manipulatable 3D representation of the tumor, as shown in figure 8.

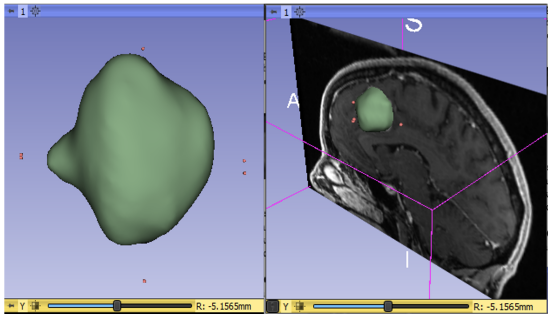


Figure 8: A fully manipulatable 3D representation of the tumor.

4 Conclusions

In terms of medical imaging, we've demonstrated how useful it is to use GPU-based processing. There are some applications where GPU-optimized algorithms fail to yield significant improvements, which is to be anticipated since the algorithms were designed with CPU architecture in mind. This is a small percentage of algorithms, as the great majority are parallel by nature and so thrive on a GPU-based computing platform. We've seen how great the gains are for specific algorithms designed for usage on the GPU in all of the trials included in this study, and the results speak for themselves. With speedups ranging from 2 to 13 times, we can conclude that GPU-based picture segmentation is the way to go at the moment and in the future. Since GPU technology has improved dramatically in the previous decade, we've seen a slew of new picture segmentation implementations emerge, resulting in nothing but better outcomes that will continue to develop and progress.

References

[1] 3D Slicer. `https://www.slicer.org/`, 2021. Accessed: 2021-01-06.

[2] NVIDIA AI-Assisted Annotation (AIAA). `https://docs.nvidia.com/clara/tlt-mi/aiaa/index.html`, 2021. Accessed: 2021-01-06.

[3] Toru Baji. Evolution of the GPU device widely used in ai and massive parallel processing. In *2018 IEEE 2nd Electron Devices Technology and Manufacturing Conference (EDTM)*, pages 7–9. IEEE, 2018.

[4] Isaac N Bankman. Introduction to segmentation, 2000.

[5] Sweta Bhattacharya, Praveen Kumar Reddy Maddikunta, Rajesh Kaluri, Saurabh Singh, Thippa Reddy Gadekallu, Mamoun Alazab, Usman Tariq, et al. A novel pca-firefly based xgboost classification model for intrusion detection in networks using GPU. *Electronics*, 9(2):219, 2020.

[6] Hao Chen, Ming Lu, Zhan Ma, Xu Zhang, Yiling Xu, Qiu Shen, and Wenjun Zhang. Learned resolution scaling powered gaming-as-a-service at scale. *IEEE Transactions on Multimedia*, 2020.

[7] Roberto Di Salvo and Carmelo Pino. Image and video processing on CUDA: state of the art and future directions. *MACMESE*, 11:60–66, 2011.

[8] Anders Eklund, Paul Dufort, Daniel Forsberg, and Stephen M LaConte. Medical image processing on the GPU–Past, present and future. *Medical image analysis*, 17(8):1073–1094, 2013.

[9] Raheleh Kafieh, Hossein Rabbani, and Saeed Kermani. A review of algorithms for segmentation of optical coherence tomography from retina. *Journal of medical signals and sensors*, 3(1):45, 2013.

[10] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[11] Claude Kauffmann and Nicolas Piche. Cellular automaton for ultra-fast watershed transform on GPU. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

[12] Chiyoung Lee, Se-Won Kim, and Chuck Yoo. VADI: GPU virtualization for an automotive platform. *IEEE Transactions on Industrial Informatics*, 12(1):277–290, 2015.

[13] John D Owens, Mike Houston, David Luebke, Simon Green, John E Stone, and James C Phillips. GPU computing. *Proceedings of the IEEE*, 96(5):879–899, 2008.

[14] Linda G Shapiro and G Stockman. Computer vision prentice hall. *Inc., New Jersey*, 2001.

[15] Neeraj Sharma and Lalit M Aggarwal. Automated medical image segmentation techniques. *Journal of medical physics/Association of Medical Physicists of India*, 35(1):3, 2010.

[16] Anthony Sherbondy, Michael Houston, and Sandy Napel. Fast volume segmentation with simultaneous visualization using programmable graphics hardware. In *IEEE Visualization, 2003. VIS 2003.*, pages 171–176. IEEE, 2003.

[17] Lin Shi, Wen Liu, Heye Zhang, Yongming Xie, and Defeng Wang. A survey of GPU-based medical image computing techniques. *Quantitative imaging in medicine and surgery*, 2(3):188, 2012.

[18] Erik Smistad, Thomas L Falch, Mohammadmehdi Bozorgi, Anne C Elster, and Frank Lindseth. Medical image segmentation on GPUs–a comprehensive review. *Medical image analysis*, 20(1):1–18, 2015.

[19] Abdel Aziz Taha and Allan Hanbury. Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool. *BMC medical imaging*, 15(1):1–28, 2015.

[20] Sajedul Talukder. Tools and techniques for malware detection and analysis. *arXiv preprint arXiv:2002.06819*, 2020.

[21] Sajedul Talukder and Bogdan Carbunar. AbuSniff: Automatic Detection and Defenses Against Abusive Facebook Friends. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[22] Sajedul Talukder, Iftekharul Islam Sakib, Faruk Hossen, Zahidur Rahim Talukder, and Shohrab Hossain. Attacks and defenses in mobile ip: Modeling with stochastic game petri net. In *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, pages 18–23, 2017.

[23] Sajedul Talukder, Md Iftekharul Islam Sakib, Zahidur Rahim Talukder, Upoma Das, Arnob Saha, and Nur Sultan Nazar Bayev. Usensewer: Ultrasonic sensor and gsm-arduino based automated sewerage management.

In *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, pages 12–17. IEEE, 2017.

[24] Sajedul Talukder and Zahidur Talukder. A survey on malware detection and analysis tools. *International Journal of Network Security & Its Applications*, 12(2), 2020.

[25] Sajedul Talukder, Shalisha Witherspoon, Kanishk Srivastava, and Ryan Thompson. Mobile technology in healthcare environment: Security vulnerabilities and countermeasures. *arXiv:1807.11086*, 2018.

[26] Björn Wagner, Paul Müller, and Gundolf Haase. A parallel watershed-transformation algorithm for the GPU. In *Workshop on Applications of Discrete Geometry and Mathematical Morphology*, pages 111–115, 2010.

[27] Chenyang Xu and Jerry L Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on image processing*, 7(3):359–369, 1998.

Opportunities for Using HBCU Culture to Teach Elementary Data Structures to Computing Students*

Gloria Washington[1], Marlon Mejias[2], Saurav Aryal[1], Todd Shurn[1] and Legand Burge III[1]
[1]Electrical and Computer Science Department
Howard University
Washington, D.C. 20059
{gloria.washington,tshurn,saurav.aryal,lburge}@howard.edu
[2]Department of Software and Information Systems
UNC Charlotte
Charlotte, NC 28223
mmejiase@uncc.edu

Abstract

Historically black colleges and universities (HBCUs) have long since produced high quality education through their unique culture, family-like environment, and access to Black leaders that actually look like the student body of the institution. Computing educators in introductory programming courses like CS 1 and CS 2 can leverage the unique culture within a specific HBCU to teach elementary data structures to undergraduate students. Through innovative curriculum, homework assignments, and informal learning opportunities that leverage the commonality of Greek probate shows performed on campus and access to alumni scientists and tech professionals; HBCU educators can help to engage their students and improve the performance of their students. This paper presents a study within Howard University CS 2 course that infused computing curriculum with HBCU culture to assist with explaining data structure topics likes arrays, lists, and sorting techniques.

Approximately 24 students were randomly chosen to interact with the HBCU-specific curriculum and were given homework assignments that supplemented their normal lecture. This work-in-progress results showed students exposed to curriculum performed better on the homework assignments than students without access to the curriculum. However, preliminary results show the results are not statistically significant and more evaluation needs to be done. Preliminary lessons extracted from this work may help non-HBCU institutions that have similar cultural activities and have rich alumni tech networks.

1 Introduction

HBCUs were founded to educate primarily African-American students in the United States and since 1837 have continued to provide quality education, culture, and social environments for students to thrive and create their own identities. These institutions have produced one third of all Black STEM PhD recipients[20], produced 19% of all STEM Bachelor's degrees, and provided clear pathways to the middle class[2, 15]. HBCU researchers and alumni describe these institutions as places providing "freedom to explore", "be yourself and develop yourself", and "unapologetic Black spaces"[2]. HBCUs are rich with social capital crucial to nurturing and promoting academic success[18]. Computer science professors can leverage this culture at HBCUs to instruct students on computer science skills.

This paper describes the use of cultural elements found mainly on HBCU campuses such as Greek step shows or probate shows that can be used to make lectures more culturally relevant, engaging, and provide students with informal learning opportunities outside of traditional homework assignments. Greek probate or step shows are African-inspired dances that Black Greek-lettered organizations use to introduce new members of a fraternity or sorority or introduce new students to the organizations. Examples of Black Greek-lettered organizations are Delta Sigma Theta or Omega Psi Phi. The primary research question this paper addresses is: does HBCU-specific curriculum improve the performance of HBCU students enrolled in a data structures course? Through a pilot study conducted at Howard University, we test this hypothesis and examine ways the culture-infused curriculum can be tailored for CS 1 and 2 courses.

Although HBCUs are not a monolith, this paper first provides a brief overview of the culture found on most HBCU campuses across America. Then we describe the use of socially and culturally relevant curriculum for teaching computing concepts across undergraduate education. Next, we describe the methodology to introduce HBCU students to the curriculum used in this study and its connection to the unique cultural elements at that HBCU. Then,

we describe the performance of the students that received the chance to interact with and learn from the curriculum. Finally, results of the study are described along with opportunities for future exploration of the concepts and techniques discussed in the paper at non-HBCU campus. Additionally discussed, are any challenges and caveats that should be considered if researchers may want to implement a similar program at their institutions.

2 Background

2.1 Culture and STEM Leadership Development at HBCUs

The first institution founded to educate African Americans in higher education with topics in agriculture, industrial engineering, home economics, and mechanics was Cheyney University in 1837 . Now, there are more than 100 historically black colleges and universities in the United States, with the bulk of these institutions residing in the Southeast part of the nation. Founded mainly to teach Black Americans skills relevant to farming and home management, these institutions in the 1950's became the premiere institutions to produce Black doctors, lawyers, teachers, and scientists that would go on to change United States' history[2, 20]. All HBCUs have a legacy of creating great African-American leaders. However, most Americans are more familiar with institutions like Howard University, Spelman College, Morehouse College, and Tuskegee University because of their media popularity, philanthropic efforts, famous alumni, and record of scientific inventions that revolutionized American farming[15].

Most Black students that have attended an HBCU have benefitted academically, professionally, and personally from enrollment at these types of institutions. Studies that have explored HBCUs ability to produce leaders talk about the close student involvement and interactions with faculty, abundant informal and formal mentorship experiences, access to an overrepresentation of Black leaders, and satisfaction with campus activities and culture (e.g. student clubs, organizations, and extracurricular activities)[8, 14, 17, 4]. Students that attend HBCUs also have greater intellectual development, higher career aspirations, and experience more satisfaction with their academic development[17, 18]. Many of these institutions also give "second chances" to students that might have low high school GPAs, low ACT/SAT scores, or little to no access to advanced placement classes in high school. Many of the students enrolled at HBCUs are students that a majority institution would consider academically underprepared and not likely to persist and graduate from college. However, many of these students go on to have successful careers and higher incomes than their counterparts that graduated from historically white institutions[7, 15].

HBCUs during the 1920's through the 1940s experienced a "Golden Age" of enrollment as Black Americans were just beginning to experience the fruits of their labor and move into the middle class[2]. Now with renewed energy surrounding HBCUs and enrollment numbers increasing, these institutions can continue to bring quality education and leadership development to a new group of interested students. HBCUs like Spelman, Morehouse, Morgan, Florida Agriculture and Mechanical University are involved in several partnerships with industry leaders like Amazon, Google, and Facebook to produce the next crop of computer scientists to work and innovate in Silicon Valley and in tech. Programs like the Howard West initiative that was the precursor to Google's Tech Exchange program helped bring light to the dire absence of Black tech professionals in Silicon Valley[9]. Industry leaders have long wanted to leverage the unique qualities of HBCUs to produce leaders by teaming with HBCU administrators to create introductory programming and applied computer sciences courses that would mix the HBCU classroom environment with the fail-fast, recover quick environment of tech. Many of these programs are having success getting their students in the door through internship experiences. Fulltime Black tech workers remains at about 16% for information technology (IT) companies and Big tech companies[5, 3, 19]. However, Silicon Valley numbers are much less with Black tech workers only representing 4.4% of the population.

HBCUs produce about 33% of the nation's Black computer scientists[8]. Tech companies that partner with HBCUs often assist with teaching students to become better software engineers. These companies may also benefit from infusing HBCU-positive culture into its employee resource groups or people development programs. Black tech professionals that have expressed work dissatisfaction mention the struggle to find mentors of color, racial bias and microaggressions, stereotypes and biases influence promotion decisions, and lack of sustained and consistent efforts by diversity and inclusion programs[5]. Many of these struggles and challenges experienced by the tech industry may be addressed by leveraging HBCU culture to gain easy access to mentors, interact with diverse leaders in tech, involve students in tech activities inside and outside of the classroom, and gain increased intellectual development of its employees.

2.2 Introductory Programming Courses

At most universities undergraduate students are taught programming through a series of introductory coding courses that provide the basics on computer science theory, elementary data structures, and application of these data structures for common real-world problems. Usually these courses follow a naming structure of computer science 1 (CS 1) and computer science 2 (CS 2) because they help students understand the basic syntax for a particular programming language and introduce the standard creation and manipulation of data struc-

tures for storing and processing programmatic data. CS 1 courses at most universities teach coding syntax, debugging, problem-solving, algorithmic creation, and object-oriented programming introductory experience. In this paper we focus more on the CS 2 course that teaches the design, implementation, and manipulation of data structures for effectively developing software. Course learning objectives commonly found in CS 2 courses refer to students understanding and identifying[13]:

- Basic software engineering principles
- Abstract data types or structures
- How data is represented at the application, logical, and implementation levels
- Sorted and unsorted lists, stacks, queues, and linked lists
- The difference between static and dynamic memory usage
- Recursions and their use in programmatic optimizations
- Common searching and sorting algorithms that operate on data structures

Most CS1 and CS 2 course are structured with both theory and lecture. However, some computing programs have found that use of flipped classroom[5] teaching approaches are better suited for students to learn coding. In flipped classroom approaches, students prepare for class by reading theory related to coding syntax, data structures, and concepts. During class, students implement the theories learnt in hands-on programming assignments, paired programming, or by answering short quizzes that test their knowledge of the materials that they read outside of class.

Along with this, computer education researchers are suggesting that students engage in more informal learning opportunities outside of the classroom like hackathons, programming challenges, and tech-related clubs[5]. These experiences help computing students by providing more opportunities to practice their computational thinking and problem-solving skills on problems relevant to their interests and hobbies. However, HBCU students because of socioeconomic factors are not always able to participate in these informal learning opportunities because of financial commitments that require them to get a job to supplement paying for their education. Because of this, they do not rack up the number of hours practicing and studying for their computing courses. Many researchers have expressed the need to create curriculum that integrates these informal learning activities with traditional lecture[5]. Techniques used by HBCU researchers are highlighted in the next section and show how some HBCUs are successful at mixing traditional lecture with informal learning activities outside of the classroom.

2.3 Socially and Culturally Relevant Computing Curriculum at HBCUs

Socially relevant computing refers to use of computation to solve problems that students are passionate about and have interest in[5]. Culturally relevant or culturally-responsive computing education refers to use of cultural elements to center computing problems around that students will solve or find inspiration in[13]. Use of socially relevant and culturally relevant curriculum has been mainly utilized in K-12 education to get students excited about pursuing computer science in undergraduate education[1]. However, use of culturally relevant and socially relevant curriculum in K-12 computing education is outside of the scope of this paper. Therefore this paper will focus on use of such curriculum within HBCU computing programs.

Socially and culturally relevant computing curriculum use students' lived experiences and cultural capital as pedagogical resources to engage students and aid their understanding of computing topics. Socially relevant computing emphasizes the use of computing to solve societal problems that students may find of interest[5]. Culturally relevant computing pedagogy relates theoretical concepts to students' prior knowledge, experiences, identity and perceptions to create meaningful learning experiences[14]. Culturally relevant pedagogical approaches happen both in the classroom learning environment, as formal curricula and outside the classroom as symbolic curriculum[13]. Symbolic curriculum implicitly associates the student's identity to content matter by representation in the physical learning environment. It uses media with diverse representation to communicate that diverse identities belong and achieve success in the field. Socially and culturally relevant computing affirms students' intersectional identities and affirms their cultural capital. These two approaches can lead to increased student interest, motivation, engagement and long term learning[13]. They also increase equity and inclusivity among underrepresented students because it allows them to leverage their unique intersectional identities to leverage theoretical concepts to contribute solutions to problems that relate to and benefit their communities[13]. Undergraduate students can develop higher-order thinking skills[11] needed for fully understanding abstract computer science concepts, data structures, and algorithmic techniques through personal experiences, situations, activities, or events surrounding their specific HBCU culture[10, 21].

Socially and culturally computing curriculum have been used extensively at historically black institutions. HBCUs like Spelman, Morehouse, and Morgan State University use it to get students to think concretely about abstract concepts like abstraction and elementary data structures that are normally taught in a CS 1 or CS 2 course. Spelman computer science instructors recently utilized the difficult subject of social justice to teach common data struc-

tures concepts like classes and objects [5]. In older studies, Howard University has incorporated culturally relevant pedagogical approaches in introductory CS classes as well as throughout the artifacts and media represented in the department[12]. Morgan State University has performed studies on inclusivity of women in tech to teach computing subjects[13]. FAMU has used the social and personal impacts of information security breaches to teach fundamentals of secure computing and information assurance to its students[16].

HBCUs have been successful at implementing curriculum that is relevant to its students, however many of the instructors at HBCUs have not published to the computer science education community about their techniques and how these techniques have helped them improve the performance of their students in introductory programming courses. In[6], the struggles that tenure-track faculty face at HBCUs is discussed. Most HBCUs instructors have high teaching loads and cannot devote time to publishing on the unique techniques they use to get students to retain and apply material. Additionally, HBCU instructors usually lack teaching assistants that help the professor grade and administer lessons to the students. With time spent instructing, creating curriculum, grading assignments, and programming laboratories, there is little time to publish.

3 Methodology

In this work we wanted to study if students performed better or worse in a CS 2 course taught at Howard University that used HBCU-specific socially and culturally relevant computing techniques to teach data structures using Python. HBCU-culturally specific examples included Black Greek step shows, presentation of fraternity/sorority members shows, and African step dances. To test this hypothesis, approximately 24 students were randomly chosen from a class of 56 students (24 female, and 32 males) to interact and use curriculum that had been modified to include 1) lectures over the similarity between Black Greek-lettered organizations' probate shows and data structures, 2) eight homework/lab assignments that included an implementation of the hash tables, stacks and queues, binary search trees, and sorting techniques, 3) students watched alumni videos that pre-recorded asking about their experiences in undergraduate and courses that impacted their current tech jobs, and 4) pre and post quizzes that asked specific questions about the data structure technique before and after reviewing the socially and culturally relevant material. The 32 students not randomly chosen to use the curriculum were assigned regular lecture, quizzes, and assignments and could not access the other culturally and socially specific curriculum. The performance of each group was recorded and compared. Additionally in the lectures, all students were told about tech

alumni that had graduated from their HBCU that have worked or are currently working at big tech firms. The tech alumni noted their personal experiences with their computing courses as well.

Figure 1: Black Greek-lettered organization probate and step show culture used to teach stacks, queues, push & pop functions, and sorting in the Howard curriculum.

4 Results

The Howard HBCU curriculum varied according to difficulty, with stacks and queues programming assignments designed to be slightly easier to implement than hash tables. Howard culture was infused into the lessons and homework assignments. The work presented in this paper is still being analyzed and another iteration of the curriculum will be taught in the Fall 2022 semester. However, preliminary analysis of student performance in the CS 2 class indicates that the HBCU-specific curriculum helped students perform on average

3% better than their peers. This slight increase is not statistically significant and may be explained by the difficulties, policies, and procedures that occurred during the global pandemic. This is discussed in the next section. Also, more quantitative analysis using analysis of variance (ANOVA) two-factor t-tests is being performed across the homework assignments examining factors including homework difficulty, gender of students, and computing topic. Qualitative analysis of feedback gathered from pre- and post-test questionnaires is also being conducted using thematic coding to identify topical themes indicated by the HBCU students.

5 Discussion

We understand that event programming and unique cultural locations are not found at every HBCU, so in this section alternative ideas for researchers to consider are provided.

1. **Inspiration from Cultural Programming.** Look for campus specific culturally elements that are normally associated with the Multicultural Student Union or the Black Student Union. These activities will take a little time to develop lecture materials for, however the students will appreciate the cultural relevance to their university environment.
2. **Mentorship Examples.** Reach out to diverse alumni that are currently working in tech. Record their experiences and incorporate some of their suggestions into lecture materials. Have the alum talk about examples they never thought they would get that were taught in their undergrad computing course. Students in this study, were exposed to student alum videos where alum talked about the course they wished they paid attention to and how it impacted their current jobs.
3. **Active Learning.** Look for places on campus that mimic physically the structure of a list, array, hash table, or queue. Examples include lines in the cafeteria, quad or Greek plots on campus, or groups of statues that students can physically perform the actions in the programs they are creating.

Some difficulties were experienced due to teaching remotely and interruptions related to the COVID-19 pandemic. The students in the CS 2 course had an average grade of 93%. However, compared with previous semesters' average final grade, the final grade of the course is not marginally higher or lower than what is produced during a normal semester. Additionally, during this semester due dates for assignments were stretched and students with difficulties due to COVID-19, could submit updates to their homework assignments; thereby improving their grades. This change in the structure of the

course and its activities due to COVID-19 will need to be studied to determine in what ways the average grade differed from previous semesters according to homework assignment topic and difficulty.

6 Conclusion

In this paper we described a small scale study that examined if socially and culturally relevant curriculum that used HBCU-specific cultural elements could improve the performance of computing students in introductory programming courses. A quasi experiment was conducted with students at an HBCU in a CS 2 course designed to teach elementary data structures using Python. The results of the student performance, although preliminary, seem promising to use the curriculum for improving the experience of the HBCU students. Institutions wishing to implement similar curriculum should leverage their campuses unique cultural characteristics or artifacts to provide new ways to teach students computing concepts.

References

[1] 2020 people of color in tech report: 2020. `https://www.trustradius.com/vendor-blog/people-of-color-in-tech-report`. Accessed: 2021-09-15.

[2] Independent lens: Tell them we are rising: The story of black colleges & universities. `https://www.kpbs.org/news/2018/feb/15/independent-lens-tell-them-we-are-rising-story-bla/`. Accessed: 2021-09-15.

[3] The state of ethnic minorities in u.s. tech: 2020: 2020. `https://www.computerworld.com/article/3574917/the-state-of-ethnic-minorities-in-us-tech-2020.html`. Accessed: 2021-09-15.

[4] Us department of education. historically black colleges and universities and higher education desegregation. us dep't of education office for civil rights washington dc., 1991.

[5] M. Buckley, J. Nordlinger, and D. Subramanian. Socially relevant computing. *ACM SIGCSE Bulletin*, 40(1):347–351, 2008.

[6] E. Dillon and K.L. Williams. Course content as a tool of inclusivity for black/african-american women in computing. *Journal of Computing Sciences in Colleges*, 36(3):151–160, 2020.

[7] P.M. Hardy, E.J. Kaganda, and M.S. Arguete. Below the surface: Hbcu performance, social mobility, and college ranking. *Journal of Black Studies*, 50(5), 2019.

[8] F.D. Henry. Persistence and retention strategies implemented at hbcus that support successful degree attainment of african american men. (2021).

[9] T. Hoyd. Google, howard expand computer science initiative into full-year program. *The Undefeated*, 2018.

[10] K. Jean-Pierre and M. Mejias. Toward smart content in adaptive learning systems: Potential, challenges, and solutions. *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, page 80, 2015.

[11] S. Masapanta-Carrión and J.Á. Velázquez-Iturbide. A systematic review of the use of bloom's taxonomy in computer science education. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, pages 441–446, 2018.

[12] M. Mejias, K. Jean-Pierre, L. Burge, and G. Washington. Culturally relevant cs pedagogy - theory and practice. *Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, pages 1–5, 2018.

[13] J. Morales-Chicas, M. Castillo, I. Bernal, P. Ramos, and B.L. Guzman. Computing with relevance and purpose: A review of culturally relevant education in computing. *International Journal of Multicultural Education*, pages 125–155, 2019.

[14] S.D. Museus, R.t. Palmer, R.J. Davis, and D.C. Maramba. Special issue: Racial and ethnic minority students' success in stem education. *ASHE Higher Education Report*, 36(6):1–140, 2011.

[15] R.A. Nathenson, A. Castro Samayoa, and M. Gasman. Moving upward and onward: Income mobility at historically black colleges and universities. 2019.

[16] J. Nias. Educational programming practices that inspire change: Social justice as situated in a computer programming course. *Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, 2021.

[17] C.L. Outcalt and T.E. Skewes-Cox. Involvement, interaction, and satisfaction: The human environment at hbcus. *The Review of Higher Education*, 25(3):331–347, 2002.

[18] R. Palmer and M. Gasman. "it takes a village to raise a child": The role of social capital in promoting academic success for african american men at a black college. *Journal of College Student Development*, 49(1):52–70, 2008.

[19] D. Tomaskovic-Devey and J. Han. Is silicon valley tech diversity possible now? *Center for Employment Equity| UMass Amherst*, 2018.

[20] R. Upton and C. Tanenbaum. The role of historically black colleges and universities as pathway providers: institutional pathways to the stem phd. *American Institutes for Research*, 2014.

[21] L. Vygotsky. Interaction between learning and development. *Readings on the development of children*, 23(3):34–41, 1978.

Introducing Computational Thinking to Pre-service Teachers*

Jiang Li, Paulette Shockey, Jennifer Cuddapah
Christy Graybeal, Marisel Torres-Crespo, Anthony Williams
Department of Computer Science and Information Technology
Department of Education
Hood College, Frederick, MD 21701
lij@hood.edu

Abstract

This paper describes a collaborative project that was conducted to promote K-8 Computer Science Education among in-service and pre-service teachers. More than 40 pre-service and in-service teachers participated in a learning experience designed to address the K-12 Computer Science Framework [2] and Maryland's K-12 Computer Science Standards. The collaboration was designed to facilitate participants' learning about and application of foundational principles of computer science and computational thinking into K-8 STEM curriculum and teaching. Participants explored hard/software platforms and used open-source sites such as Scratch, Code.org and Code Academy. Participants envisioned how activities apply to K-8 classrooms and worked in pairs or groups to design a problem-based project for students. Project evaluation included formative and summative assessments to examine changes in content and pedagogical knowledge.

1 Introduction

Computer Science (CS) builds students' logical and computational thinking skills as well as realistic problem-solving skills. Exposing K-12 students to computer science is extremely important for students' success in the future.

According to a study conducted by code.org and CSTA [1], in the past few years, more than 40 states have enacted one or more polices to make computer science education fundamental.

To promote the K-12 CS education, it is very important to prepare teachers. Systemic change, teacher engagement and development of teaching resources are required to bring CS to K-12 [3]. Recommendations have been made to create a successful CS education program, such as ensuring teachers are prepared and supported, create continuity and coherence around learning progressions, make participation equitable etc. [4]. The importance of preparing CS teachers and possible certification models were discussed by CSTA Teacher Certification Task Force [5].

Many successful programs have been developed to prepare CS teachers at high school level, while our project focus on pre-service teachers at K-8, which will help to build the pipeline of students who will explore CS further into high school.

Hood College's Computational Thinking Partnership (CTP) was a collaborative project involving the Departments of Computer Science and Information Technology (CS) and Education (EDUC) from Hood College, Frederick County Public Schools (FCPS), Frederick Community College (FCC), and Maryland Codes (a Code.org Regional Partnership). The collaboration was designed to facilitate participants' learning about and application of foundational principles of computer science and computational thinking into K-8 STEM curriculum and teaching. Participants were Hood and FCC pre-service and FCPS in-service teachers as well as Hood and FCC Computer Science students who were potentially interested in teaching.

The CTP project served as the initial step in a long-term plan to develop a Computer Science Education program as well as integrate CS principles into existing teacher education curriculum and specifically into the existing learning strategies and methodology courses. The CTP supported the designing and piloting of an initial professional learning experience that could then improve and transform in subsequent years.

2 Implementation

2.1 Background

Racially and ethnically diverse, Hood's undergraduate population is 62% White; 12% Black, not Hispanic; 9% Hispanic; 4% Asian; 5% Multi-Racial; less than 1% Native Hawaiian/Other Pacific Islander; and 5% Unknown or other. Averaging over the past 8 years, 30% of the incoming full-time freshman class are low-income students, and 22% are first-generation college students [7].

Hood's partner school district, Frederick County Public Schools (FCPS), comprises 67 schools including 10 high, 13 middle, and 37 elementary schools. The student body is diverse, matriculating from a mix of rural, suburban, and more urbanized communities. The racial/ethnic breakdown of the student body is 58.9% White, 17.1% Hispanic/ Latino, 12.5% Black, 5.5% Asian, 5.5% two or more races, 0.3% American Indian/Alaskan Native, and 0.2% Pacific Islander/Native Hawaiian. In addition, about 11,000 students (26% of the student body) receive Free and Reduced Price Meals, and 6% (about 2700 students) are English Language Learners (ELL)[6].

At Hood, all education students are placed each semester at an FCPS Professional Development School (PDS) for weekly $\frac{1}{2}$ day internship. The 100 day internship begins in the final fall semester with $1\frac{1}{2}$-3 days at a PDS followed by fulltime student teaching in the spring semester. Hood's PDSs are an integral aspect of the partnership agreement between Hood and FCPS and each placement gives students experience with different levels, curriculum, and demographics, to prepare them to teach in all environments. Additionally, Hood supports the FCPS teacher mentors with regular, differentiated professional development. New mentors attend a series of five workshops to help them develop in the role of mentor.

2.2 Goals and Objectives

The K-12 Computer Science Framework includes several recommendations for professional learning which informed the development of the CTP project. Recommendations about attending to novice anxiety regarding a lack of content understanding of CS, connecting to the grade level and subject area of the participating teachers, focusing on issues of access and equity, and managing the CS learning environment were reflected in the following goals and objectives of the proposed project.

Specifically, CTP project had the following goals. To facilitate the participants' learning about the principles and practices of computer science and computational thinking. To equip the participants to collaboratively develop computational thinking, problem-solving lesson plans and learning experiences for K-8 students. To launch CTP into a sustainable computer science learning experience.

2.3 Recruiting

The CTP project utilized multiple recruitment means and incentives. The project director actively worked with each partner and reached out to the potential participants in several ways. Recruitment strategies included: (1) website and Facebook posts, (2) instructor announcements in EDUC and CS

courses, (3) individual invitations during student's advising session, (4) posting flyers, (5) invitations to current Noyce grant recipients and their mentor teachers (FCPS), (6) information video, (7) dissemination through professional development opportunities announcement through FCPS, and (8) invitations to FCPS teachers who are mentors of Hood pre-service teachers.

Marketing and recruitment materials highlighted the importance of computer science in education and this CTP educational opportunity. Participants were offered a choice of 3-credit course, MSDE credit, or a stipend for participation. Any of these options incentivized participation and encouraged completion (i.e., retention). Additionally, participants received a technology kit to include kid-friendly robotics materials, and had the opportunity to receive mini-grants to implement the projects they designed in the CTP in their classrooms. The online Blackboard forum kept participants connected to one another and the faculty and consultants in between sessions.

2.4 Format and Content

Participants received professional development addressing the K-12 Computer Science Framework and Maryland's K-12 Computer Science Standards through a 37.5 hour educational experience occurring on 2 full-day sessions in late summer 2019 plus 4 monthly full-day follow-up sessions during the fall (September, October, November, December 2019).

Participants entered the project with no prior knowledge of computer science. Thus, we began by introducing participants to the core and crosscutting concepts of the K-12 Computer Science Framework: computing systems, networks and the internet, data and analysis, algorithms and programming, impacts of computing, abstraction, system relationships, human-computer interaction, privacy and security, and communication and coordination. Participants engaged in basic computer science activities first as students and then looked at the same activities as teachers. The participants were encouraged to envision how these activities could be implemented in the K-8 classroom, what challenges would be faced, how to support students as they learn the basics of computational thinking and computer science, and how to incorporate these ideas into the already full curriculum. Throughout the program participants learned about teaching resources and developed networks of support with whom they stay connected through a BlackBoard (Bb) mediated forum. Hood CS and EDUC faculty designed and delivered the professional development. Participants were given individualized and group support from these experienced professionals throughout the experience. Additionally, they received supplemental content guidance from some Hood CS majors, who are not pursuing teaching, selected to help on the content delivery-side of the project.

The K-12 Computer Science Framework is comprised of overarching Practices and Concepts which inform the curricular development of what occurred during the 6-day educational experience. Through a variety of hands-on, interactive and discussion opportunities, participants learned about the concepts and practices and then applied their learning to developing a project they can use with K-8 learners. Collaborative project development, particularly between pre-service and in-service teachers, was emphasized, and participants were encouraged to apply to receive additional funding for implementing and assessing their projects during the fall semester.

2.5 Sustainability

To obtain the sustainable goal, the following approaches were conducted: (1) Evaluating development and implementation of CTP through quantitative and qualitative analysis of surveys, observations, and documents; (2) Revising existing pre-service teacher preparation curriculum to integrate CS standards; (3) Maintaining collaboration between FCPS and the CS and EDUC Departments; and (4) Securing future funding for CS education.

3 Results

In August 2019, 22 participants completed the initial survey. In November 2019, the same 22 participants completed the end of semester survey.

18 participants identified as female, 4 as male. 14 participants identified as White or Caucasian, 4 as Black or African American, 4 as Hispanic or Latino, 1 as Asian or Asian American, and 2 as American Indian or Alaska Native. (Some participants identified as multiple racial and/or ethnic groups so the total is not 22.) Table 1 shows the age group of participants.

Table 1: Age group of Participants

Age Range	Number of Participants
Under 18	0
18-24	13
25-34	4
35-44	1
45-54	2
55-64	2
65+	0

16 participants were college students planning to become teachers. 1 was a college student unsure of career plans. 4 were teachers. 1 was a college professor.

3 participants planned to teach or currently taught pre-school or pre-kindergarten. 7 participants kindergarten or 1st grade. 9 participants 2nd or 3rd grades. 5 participants 4th or 5th grades. 7 participants 6th, 7th, or 8th grades. 6 participants 9th – 12th grades. 2 participants college. (Some participants taught or plan to teach multiple grade levels so the total is not 22.)

17 participants had no formal teaching experience. Each of the other participants had 4, 6, 8, 12, or 20 years of teaching experience.

Participants were asked how much formal Computer Science and/or Computational Thinking Education they have completed. 1 participant reported having taken a course or two in Computer Science and/or Computational Thinking and planning to complete a degree in Computer Science. 4 participants took a course or two but do not plan to complete a degree in Computer Science. 17 participants had no formal experience with Computer Science of Computational Thinking.

Table 2 shows the overall result of the survey. The values in the table indicate how strongly agree or disagree with the corresponding statements. (1 = Strongly disagree, 2 = Disagree, 3 = Neither agree nor disagree, 4 = Agree, 5 = Strongly agree)

Statement list:

- I have the **knowledge** I need to teach computational thinking effectively.
- I have the **skills** I need to teach computational thinking effectively.
- I have the **curricular tools and resources** I need to teach computational thinking effectively.
- I have a **social network** that enables me to teach computational thinking effectively.
- I can **interest** my students in computational thinking.
- I can effectively teach **all students** computational thinking.
- I can **assess** my students' learning and performance with regard to computational thinking.
- I am confident that I can use computational thinking **devices** in my classroom to teach the foundations of computer science.

4 Conclusion and Future Work

From the positive data shown in the previous section, our project was concluded as a success. Our participants gained domain knowledge and were

Table 2: Survey Result

	Before Workshops			After Workshops		
	Mean	**Mode**	**Range**	**Mean**	**Mode**	**Range**
Statement 1	2.36	2	1-4	4.27	4	3-5
Statement 2	2.5	2	1-5	4.18	4	2-5
Statement 3	2.29	2	1-4	4.36	4	2-5
Statement 4	2.68	2	1-4	4.59	5	4-5
Statement 5	3.54	4	2-5	4.73	5	4-5
Statement 6	2.95	3	1-4	4.32	4	2-5
Statement 7	2.91	3	1-5	4.27	4	3-5
Statement 8	3.09	3	2-5	4.64	5	3-5

equipped with curriculum content to teach CS in their classrooms. Our participants learned about the principles and practices of computer science and computational thinking and collaboratively developed computational thinking, problem-solving lesson plans and learning experiences for K-8 students. Future work includes transforming the CTP educational experience into a Computer Science Education course, to enhance sustainability of this project. We seek to sustain the curricular elements of the CTP project. We see this project as being one that can be offered in the teacher preparation program. We would like to use the CTP materials as part of the CS sequence for Praxis II preparation. Specific activities from the sessions can be infused into existing education classes.

This project was sponsored by Maryland Center for Computing Education (MCCE) Maryland Pre-service Computer Science Teacher Education Program.

References

[1] 2018 state of computer science education (2018). `https://advocacy.code.org`.

[2] K–12 computer science framework (2016). `http://www.k12cs.org`.

[3] Valerie Barr and Chris Stephenson. Bringing computational thinking to K-12: what is involved and what is the role of the computer science education community? *Acm Inroads*, 2(1):48–54, 2011.

[4] Paulo Blikstein and Sepi Hejazi Moghadam. Pre-college computer science education: A survey of the field. 2018.

[5] Judith Gal-Ezer and Chris Stephenson. Computer science teacher preparation is critical. *ACM Inroads*, 1(1):61–66, 2010.

[6] FCPS. (n.d.). Fcps fast facts. `https://www.fcps.org/about/fast-facts`.

[7] OIRA, Office of Institutional Research demographics, Hood College. Fall fact sheet: Student demographics. 2017.

Checkpoint Variations for Deep Q-Learning in Reconnaissance Blind Chess*

Kyle Blowitski and Timothy Highley
Department of Mathematics and Computer Science
La Salle University, Philadelphia, PA 19141
highley@lasalle.edu, kblowitski@protonmail.com

Abstract

Reconnaissance Blind Chess (RBC) is a chess variant where each player has only limited knowledge about the location of the opponent's pieces. Each player has a fixed allotment of time to think that must be budgeted over the course of the game. In this work, deep Q-learning has been applied to a bot that plays RBC in order to decide how much time to spend thinking and how broadly to search each turn. Two different techniques for checkpointing in deep Q-learning were utilized and compared. The first version played small batches of 10 matches and saved a checkpoint if it won at least 5 matches, while the second version played a single match at a time and only saved a checkpoint if it won. When played against each other, the second version won 58% of the matches. The bot finished in second place in 2020 Leaderboard Challenge for Reconnaissance Blind Chess.

1 Introduction

Reconnaissance Blind Chess is a chess variant where players are always aware of the location of their own pieces, but they have only partial knowledge of the location of their opponent's pieces [8]. Each turn consists of a sense action and a move action. For the sense action, the player chooses any square on the board. The player then learns the contents of that square and the eight

adjacent squares. The move action works like regular chess with a couple of exceptions. First, check is ignored. Second, a player may end up attempting to move through an enemy piece because they may not know its location. If that happens, the player captures the piece that is in the way. Whenever a capture occurs, the player performing the capture only learns where the capture occurred. They do not learn the identity of the captured piece. The game ends when a player wins by actually capturing the opposing king, or when a player loses by running out of time.

Highley, Funk, and Okin [4] developed a bot to play Reconnaissance Blind Chess that focuses on keeping track of the board state on a piece-by-piece basis (as opposed to a board-by-board basis). Their bot took second place in the NeurIPS 2019 Tournament [2], and this work builds on their original approach by introducing LaQ-Bot.

LaQ-Bot, like the original bot, uses the chess engine Stockfish to evaluate the strength of possible chess boards, and Stockfish evaluates boards in a unit called centipawns. A board score of 100 centipawns indicates that the player is ahead by approximately one pawn.

The primary difference between LaQ-Bot and the original bot is that LaQ-Bot uses deep Q-learning. LaQ-Bot took second place in the RBC 2020 Leaderboard Challenge, sponsored by Johns Hopkins University Applied Physics Laboratory. However, only minimal learning was completed before the challenge. This study addresses two questions. First, how helpful would additional learning time on the same hardware be? Second, which of two checkpointing approaches would work better?

2 Deep Q-learning

LaQ-Bot is an agent that plays RBC and utilizes deep Q-learning to make decisions on how much time to spend thinking during its turn and how many boards to analyze. Since playing many RBC games has a high time cost, different checkpointing techniques were employed with varying effects on the bots' performance after training.

Reinforcement learning, a technique inspired by behavioral psychology, guides how agents interact with their environments and how they take actions that maximize their expected returns [14]. The goal is to have the agent learn effective policies through "optimizing a cumulative future reward signal" [13].

Q-learning is a type of reinforcement learning that attempts to achieve the optimal policy in a Markov decision process. It is one of the most used applications of reinforcement learning [5] and works by using the Bellman optimality equation to iteratively update the Q-values for each state-action pair until it reaches the optimal policy. Because it uses these simple Q-functions, it acts as

the foundation for many other reinforcement learning algorithms [5]. However, in situations with large state spaces and action spaces, this approach is infeasible. In these situations, deep Q-learning can be employed, where a neural network is used to approximate the optimal policy.

Deep Q-learning is an approach which "combines Q-learning with a flexible deep neural network" [13]. Q-learning has difficulties with complex environments with very large numbers of state-action pairs [5], as calculating the optimal policy may be impossible [17] or take too much time. The use of a neural network helps get around those issues. With deep Q-learning, the heavy calculations can be done during the training stage while the application stage may only need seconds to generate an output [11]. Deep Q-networks typically consist of convolutional layers, which reduce the input to only the most important features, and fully connected layers, which classify the data. Hidden layers are layers that are in between input and output layers and use activation functions to get an output, with one of the most common being the ReLU activation function. Optimizers are often used to change attributes to reduce losses, and reward discounting is used to make the neural network value future rewards less and less. This is to prevent the bot from being too far-sighted after it has already learned a lot, and the discount factor, gamma (), is typically close to 1.0 to prevent the neural network from initially being too near-sighted.

In many situations, training is expensive in terms of time and required resources. To account for this, checkpointing can be utilized to carefully filter data so that the deep Q-network can use higher quality data to make up for the lower quantity of data, which could potentially expedite training. This approach was chosen because it also addresses the issue of resources, as it does not require the host machine to have any extra resources during training. The goal of careful checkpointing is to ensure that the "best possible trained model" is retained [3], and researchers will be able to tailor how large or small the batches they wish to run are to their project and their hardware limitations. Checkpointing allows the neural network to save what it has learned so it can pick up from where it left off. The model checkpoint observes the bot while it is training, and "If there are any positive changes..., the model checkpoint replaces the existing retained model with the new one" [3].

In Reconnaissance Blind Chess, a single match can last up to half an hour, and neural networks often require massive amounts of tests and data, so this exemplifies a situation where training is expensive and time is limited. In this paper, two different checkpointing techniques are employed. Deep Q-learning utilizes a convolutional neural network to approximate a Q-function for all action values for all states, and this approximation is used to derive an optimal policy [9]. In order to account for instabilities that arise from using a neural network, experience replay is used, which "randomizes over the data, thereby

removing correlations in the observation sequence and smoothing over changes in the data distribution" [7]. Additionally, a target network is used to calculate a target value that is periodically updated to "stabilize the learning process" [9].

Deep Q-learning has had practical applications in stock trading [1], workflow scheduling for IaaS clouds [15], optimizing energy consumption [18, 6], and protecting healthcare systems from Malware attacks [10]. Additionally, many researchers have made improvements to make deep Q-learning more efficient for varying circumstances, such as when actions do not directly affect the environment in any relevant way [16] or when there is a lot of missing data [12].

3 LaQ-Bot in Detail

LaQ-Bot is set up to use deep Q-learning to optimize how long it spends thinking and how many boards it should analyze each turn. The deep Q-network is coded in Python using PyTorch and consists of two fully connected layers, where one is given four input dimensions, feeds them into five neurons, and is activated using the ReLU activation function. The next layer consists of these five neurons and the action space, where it is given output dimensions of the first fully connected layer and it selects a single valid action from the action space. It uses the Adam optimizer to optimize the parameters of the network using a learning rate of 0.0001 and a mean squared error loss function.

The agent is set up so there is an online network and a target network. The online network is updated with gradient descent. The target network is used to handle the calculation of the target values. It is periodically updated with the weights of the online network. The agent uses a replay memory for sampling the agent's history and for training the network. It uses epsilon-greedy action selection as a solution to the exploration versus exploitation dilemma. The agent is given a gamma of 0.99, an initial epsilon of 1.0, an epsilon minimum when training of 0.1, an epsilon decrement of 1.5e-4, a batch size of 32, and a replace interval of 30 steps since a typical match plays for approximately 60 steps. With those parameters, it reaches the epsilon minimum after about 100 games. For learning, the agent waits until the specified batch size has been filled, and randomly samples the memory. It uses the online network to calculate the values of the actions the agent took in those states, and it uses the target network to calculate the values of the maximal actions for those states. This is done to move the agent's estimates of the action values closer to the maximal values, so it selects better actions that are closer to these maximal actions.

The goal was to improve the original RBC bot by using a neural network to optimize the amount of time it would spend thinking each turn. Deep Q-learning is attractive in this context because it learns while it plays.

To decide which action to choose, the neural network is given an observation of the current environment. The observation given to LaQ-Bot's neural network consists of the seconds it has left in the match, the current turn number, an evaluation of the board in centipawns, and a value called the uncertainty score, which indicates how certain the bot is that it knows where the opponent's pieces are.

In deep Q-learning, the neural network has a discrete number of actions to choose from, and this list of actions is called the action space. For LaQ-Bot, the action space consists of 30 items, which correspond to a combination of the time the bot should spend thinking for the turn and the number of boards the bot should analyze. The actions were set up so the bot would choose to use 1, 2, 4, 6, 8, 10, 15, 20, 25, or 30 seconds to think, and would choose to analyze a narrow, average, or broad number of boards. The narrow analysis always consists of 1 board for each time to use. This is chosen when the agent knows the location of all or nearly all of the opponent's pieces. The average analysis consists of 4 boards for 1, 2, 4, 6, and 8 seconds, and 10 boards for the rest. The broad analysis is chosen when the bot knows very little about the board state, and it consists of 10 boards for 1, 2, 4, 6, and 8 seconds, 20 boards for 10, 15, and 20 seconds, and 50 boards for 25 and 30 seconds. In general, the bot uses less time when time is running out, when the bot knows most of the board, and in the first few moves of the game. Determining exactly how much time to use under which circumstances is a goal of this study.

After every action taken, the neural network receives a reward, which it uses to evaluate whether it made a good decision. The reward given for each action is simply the centipawn evaluation of the board after the move has been taken. The bulk of the feedback, however, comes after a bot wins or loses. If the bot wins the match it gets a reward of +500,000 centipawns, while if it loses it gets a reward of -500,000, and these values are given regardless of if the bot won or lost by king capture or if it was from timing out.

In order to learn, the neural network stores transitions that consist of the previous observation, the action taken, the reward, the current observation, and whether the game is ongoing or if it is done. Checkpointing is employed to save what the neural network has learned so that it can pick up from where it left off. Two slightly different versions of LaQ-Bot were examined, which only differ in how they save checkpoints. The first version plays a batch of 10 games, and if it wins at least 5 out of those 10, it saves a checkpoint. The second version plays only one game at a time, and if it wins it saves a checkpoint.

4 Experiments

The experiments were performed on an Intel® Core™ i5-3320M CPU @ 2.60GHz processor, Intel® HD Graphics 4000 GPU, 16GB RAM, and Ubuntu 20.04

(64-bit) operating system. This is the same hardware that was used in the 2020 Leaderboard Challenge, and learning time was limited to two months of near-constant learning.

The two versions of LaQ-Bot were trained by playing 500 games each against the original RBC bot. Afterward, the two LaQ-Bots were each tested against the original RBC bot where they were allowed to continue learning, and then against another LaQ-Bot where the neural network always makes random decisions. Finally, the two versions faced each other.

For the first version of LaQ-Bot that plays a batch of 10 games and saves a checkpoint if it wins at least 5, it won 163 training games, giving it a training win-rate of 32.6%. From the 50 batches of 10 games, it won at least 5 matches 10 times, so it was able to learn from 100 games, where it had 52 wins and 48 losses. Subsequently, it played 100 matches against the original RBC bot with an epsilon of 0, where it won 35 games, giving it a win-rate of 35%. In those 10 batches of 10 games, it won at least 5 matches 3 times, giving it 30 games to learn from, where it had 16 wins and 14 losses. The first version of LaQ-Bot learned from 68 wins and 62 losses.

The second version of LaQ-Bot plays a single game at a time and saves a checkpoint if it wins. It won 165 training games, giving it a training win-rate of 33% and 165 wins to learn from. In its 100 games with an epsilon of 0 against the original RBC bot, it won 24 games, giving it a win-rate of 24% and an additional 24 wins to learn from. This gives this second version of LaQ-Bot a total of 189 wins to learn from.

Both versions of LaQ-Bot then played 100 matches against a LaQ-Bot with a random neural network. The first version won 61 of its matches, giving it a win-rate of 61%. The second version won 63 of its matches, giving it a win-rate of 63%.

Finally, the two versions of LaQ-Bot played 300 matches against each other. The first version of LaQ-Bot won 126 matches while the second version won 174 matches, giving them the respective win-rates of 42% and 58%.

5 Results and Conclusion

This study explored variants of checkpointing for deep Q-learning in the context of Reconnaissance Blind Chess. Specifically, deep Q-learning was used to answer the question of how many potential boards to examine and how long a bot should think on each turn, where each player has a fixed allotment of time for the game. The goal was to determine how to allocate thinking time in a 30-minute game on a given set of hardware. Training is expensive in this context. Because the allotment of time is the question at hand, training with a shorter game clock would produce answers regarding thinking time in the wrong context. Training could be faster by running multiple training games at

the same time, but because the goal is to determine the thinking time for a particular set of hardware, this approach only works if the hardware is replicated, which would be an additional cost.

The checkpointing variation that saves on each win was compared to the checkpointing variation that saves in batches of ten games where it only saves if it wins at least five of the ten games. The latter variation performed better than the former when facing each other, and it also performed better when they each faced a third party an equal number of times.

An early version of this RBC bot was entered in the 2020 Leaderboard Challenge sponsored by the Johns Hopkins University Applied Physics Laboratory, where it came in second place. The original bot that LaQ-Bot was based on also entered the competition, and LaQ-Bot had a better finish. However, in direct competition, LaQ-Bot did not consistently defeat the original bot. The original bot decides its thinking time and the breadth of its searches based on human-selected heuristics. With these mixed results, it is not conclusive that the training time was sufficient.

As a future direction for study, another checkpointing variation worth considering would be to save a checkpoint regardless of if it won or lost, which would help to determine if it is best to filter for higher quality games or have a higher quantity of games to learn from.

References

[1] Chakole, J., and Kurhekar, M. Trend following deep q-learning strategy for stock trading. *Expert Systems 37*, 4 (2020), e12514.

[2] Gardner, R. W., Lowman, C., Richardson, C., Llorens, A. J., Markowitz, J., Drenkow, N., Newman, A., Clark, G., Perrotta, G., Perrotta, R., et al. The first international competition in machine reconnaissance blind chess. In *NeurIPS 2019 Competition and Demonstration Track* (2020), PMLR, pp. 121–130.

[3] Gulzar, Y., Hamid, Y., Soomro, A. B., Alwan, A. A., and Journaux, L. A convolution neural network-based seed classification system. *Symmetry 12*, 12 (2020), 2018.

[4] Highley, T., Funk, B., and Okin, L. Dealing with uncertainty: a piecewisegrid agent for reconnaissance blind chess. *Journal of Computing Sciences in Colleges 35*, 8 (2020), 156–165.

[5] Jang, B., Kim, M., Harerimana, G., and Kim, J. W. Q-learning algorithms: A comprehensive classification and applications. *IEEE Access 7* (2019), 133653–133667.

[6] Liu, Y., Zhang, D., and Gooi, H. B. Optimization strategy based on deep reinforcement learning for home energy management. *CSEE Journal of Power and Energy Systems 6*, 3 (2020), 572–582.

[7] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature 518*, 7540 (2015), 529–533.

[8] Newman, A. J., Richardson, C. L., Kain, S. M., Stankiewicz, P. G., Guseman, P. R., Schreurs, B. A., and Dunne, J. A. Reconnaissance blind multi-chess: an experimentation platform for isr sensor fusion and resource management. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXV* (2016), vol. 9842, International Society for Optics and Photonics, p. 984209.

[9] Ohnishi, S., Uchibe, E., Yamaguchi, Y., Nakanishi, K., Yasui, Y., and Ishii, S. Constrained deep q-learning gradually approaching ordinary q-learning. *Frontiers in neurorobotics 13* (2019), 103.

[10] Shakeel, P. M., Baskar, S., Dhulipala, V. S., Mishra, S., and Jaber, M. M. Maintaining security and privacy in health care system using learning based deep-q-networks. *Journal of medical systems 42*, 10 (2018), 1–10.

[11] Shen, Y., Zhao, N., Xia, M., and Du, X. A deep q-learning network for ship stowage planning problem. *Polish Maritime Research* (2017).

[12] Tan, C., Han, R., Ye, R., and Chen, K. Adaptive learning recommendation strategy based on deep q-learning. *Applied psychological measurement 44*, 4 (2020), 251–266.

[13] Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence* (2016), vol. 30.

[14] Wang, J., Elfwing, S., and Uchibe, E. Modular deep reinforcement learning from reward and punishment for robot navigation. *Neural Networks 135* (2021), 115–126.

[15] Wang, Y., Liu, H., Zheng, W., Xia, Y., Li, Y., Chen, P., Guo, K., and Xie, H. Multi-objective workflow scheduling with deep-q-network-based multi-agent reinforcement learning. *IEEE access 7* (2019), 39974–39982.

[16] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning* (2016), PMLR, pp. 1995–2003.

[17] Yang, Y., Li, X., Li, H., Li, D., and Yuan, R. Deep q-network for optimal decision for top-coal caving. *Energies 13*, 7 (2020), 1618.

[18] Yu, K.-H., Chen, Y.-A., Jaimes, E., Wu, W.-C., Liao, K.-K., Liao, J.-C., Lu, K.-C., Sheu, W.-J., and Wang, C.-C. Optimization of thermal comfort, indoor quality, and energy-saving in campus classroom through deep q learning. *Case Studies in Thermal Engineering 24* (2021), 100842.

Resource Management with Java*

Yaodong Bi
Department of Computing Sciences
University of Scranton
Scranton, PA 18510
yaodong.bi@scranton.edu

Abstract

Resource management is a critical aspect of reliable and long-running applications. Releasing resources such as dynamically allocated memory, files and network connections is essential during normal execution and as critical in presence of errors or exceptions. Java has been extensively employed in a wide variety of applications and used in our computer science curricula, and it provides programming constructs in facilitating resource management. However, those constructs have not been adequately integrated in our computer science curricula. This paper is intended to raise the awareness of resource management in presence of errors and exceptions as well as normal operations and introduces two Java language constructs in managing different types of resources with examples.

1 Introduction

Resource management is a critical aspect of reliable and, especially, long running systems. System resources such as dynamically allocated memory, I/O streams, sockets, and database connections should be acquired when they are needed and released when they are no longer used. Furthermore, those resources should also be released in presence of errors or exceptions, otherwise the system leaks resources that often lead to poor performance and even system reboots. Computer science educators have not emphasized enough the importance of resource management and some have totally ignored it. The amount of system resources (big RAM, the large number of files allowed, for

example) in students' laptops and the small program size and short execution time of student assignments somehow inadvertently make student oblivious to the importance of resource management.

Java is a commonly used programming language in the industry as well as in our computer science curricula [1][2]. Java offers a set of language constructs that may be used to manage resources during normal execution and in presence of errors or exceptions. However, those constructs have not been integrated in our curricula as much as should have been. This paper is intended to introduce two main constructs to help instructors incorporate them in their courses.

Below shows a simple contrived example which writes the result of a division to a file and then closes the file (OutputStreams, actually).

```
public void leaking(int dividend, int divisor)
   throws IOException {
   DataOutputStream out = new DataOutputStream(
      new FileOutputStream("data.asc"));
   int result = dividend / divisor;
   out.writeInt(result);
   out.close();
}
```

The code seems to be innocuous as it does close the DataOutputStream when it is no longer needed. However, the danger of the code is that the close method of DataOutputSteam may not be called if an exception is raised. The exception could be a divide-by-zero (ArithmeticException in Java) when divisor is zero, or it could be an I/O exception (IOException in Java) thrown by the call to writeInt (when disk is full, for example). Since Java employs the termination model of exception handling [4], the method terminates at the statement at which the exception occurred and thus DataOutputStream's close would not be called, which causes a resource leak. (We don't need to worry about releasing FileOutputStream since the close method DataOutputStream inherited from its superclass FilterOutputStream calls the close method of its underlying output stream [5], i.e., FileOutputSteam in this case, as part of the decorator design pattern.)

The above example showed a resource that needs to be released (via its close method) back to the system when it is no longer needed by the application. There are other types of resources that need to be released so other threads in the same application can use. Such resources include Semaphore and Lock in Java. Let's look at the following code snippet for a thread-safe insertion method of a linked list:

```
Semaphore mutex = new Semaphore(1);
LinkedList items;
private void leakingInsert(Item item) {
   mutex.acquire();
   // critical section
```

```
    items.insert(item);
    mutex.release();
}
```

The code seems to be a de facto template for solutions of the critical section problem in the context of operating systems. Semaphore mutex is initialized as a binary semaphore and it is used to guarantee that the code between mutex.acquire and release is executed mutually exclusively with other critical sections (insert, delete, update for example) protected by the same mutex. However, if an exception occurs in the critical section, e.g., null pointer exception, the release method of Semaphore would not be called and mutex would not be unlocked, which likely leads to an application reboot since no other threads in the application would be able to enter their critical sections protected by mutex.

This paper will introduce two main Java language constructs, **try-with-resources** and **try-catch-finally**, which can be used to facilitate resource management in presence of errors and exceptions as well as in normal execution.

2 The try-with-resources Statement

The try-with-resources construct (or statement) was introduced in Java 7 and enhancements were added in Java 9, which is the version this paper is based on. The syntax of the try-with-resources construct is defined:

```
try (declaration of resources) {
    // use resources
}
```

The resources declared in the try opening expression must implement Java's AutoCloseable interface, which declares one single method void close(). When the try block completes either successfully or abruptly due to exceptions, the close method of each resource declared in the try opening expression is called. Java has retrofitted the interface into many classes of the standard Java SE runtime environment such as classes in java.io, java.net, and java.sql packages [7]. The construct may add catch blocks and a finally block as normal try-catch-finally construct.

For the leaking method in the Introduction section, it can be rewritten with the try-with-resources construct as shown below.

```
public void nonLeaking(int dividend, int divisor)
    throws IOException{
    try(DataOutputStream out =
          new DataOutputStream(new FileOutputStream("data.asc"))){
        int result = dividend / divisor;
        out.writeInt(result);
    }
```

```
}
```

With this revised implementation, the DataOutputStream is declared in the opening expression of the try block and the try-with-resources construct guarantees that the stream is closed when the try block completes even when an exception was thrown from the try block.

Below is a more realistic example used in an author's database class. It uses JDBC's PreparedStatement to insert a new product into the product table (simplified for brevity). The PreparedStatement should be released when the insertion is completed. The method returns true if the insertion was successful or false otherwise. When the insertion is successful, the close method of PreparedStatement is called before the insert method returns. If an SQLException (or any exception) was thrown, the close method is called before the exception is handled by the catch statement. The exception handler prints the stack trace for debugging purposes.

```
class ProductDao {
   private Connection conn;
   public boolean insert(Product product) {
      String sql = "INSERT INTO product VALUES(?, ?);";
      try (PreparedStatement ps = conn.prepareStatement(sql)) {
         // assign product Id to first ?
         ps.setInt(1, product.getId());
         // assign name to second ?
         ps.setString(2, product.getName());
         // send the sql statement to DBMS
         ps.executeUpdate();
         return true;
      } catch (SQLException e) {
         e.printStackTrace();
         return false;
      }
   }
}
```

Note that the scope of variable ps is the try block and any exception thrown from the creation of the resource is considered from the try block as well. In other words, variable ps is not accessible in the catch block nor outside the try statement. Any SQLException thrown from conn.prepareStatement would be caught by the catch block of the try statement. To make variable ps accessible outside the try block, it needs to be declared and created before the try statement (which is allowed since Java 9). For example,

```
public boolean insert(Product product) throws SQLException {
   String sql = "INSERT INTO product VALUES(?, ?);";
   PreparedStatement ps = conn.prepareStatement(sql);
   try (ps) {
      // original code
```

Please note that since conn.prepareStatement may throw SQLException and it is no longer in the scope of the try block, the insert method needs to specify SQLException in its signature.

2.1 Managing Multiple Resources with try-with-resources

Multiple resources may be declared in the opening expression of the construct. Since Java 9, one can list resource references (with no creation of the resources), declare references and create the resources, and mix the two in the opening expression. If multiple resources need to be created, the resources are created in the order in which they appear in the expression.

Recourses declared in the try opening expression are released in the reverse order of their appearance in the opening expression. The close method of each resource is called even when a previous call to a close method has thrown an exception. The underlying JVM also checks if each resource reference is null or not and would not call its close method if it is null. Thus, no NullPointerException would be thrown from releasing resources.

Below is an example of JDBC database access objects (DAOs), in which getAllProducts retrieves a list of products from the database using a SELECT statement. The PreparedStatement and ResultSet are resources, and they should be closed when the method completes. Using the try-with-resource construct, we declare and create PreparedStatement and ResultSet in the try opening expression. A PreparedStatement is created by conn.prepareStatement and a ResultSet is created when ps.executeQuery returns with a list of products. The getAllProducts method returns an empty ArrayList when an SQLException is thrown.

```
class ProductDao {
   private Connection conn;
   public ArrayList<Product> getAllProducts() {
      ArrayList<Product> result = new ArrayList<>();
      String sql = "SELECT * FROM product";
      try (PreparedStatement ps = conn.prepareStatement(sql);
           // send sql to DDMS
           ResultSet rs = ps.executeQuery()
          ) {
         while (rs.next()) {
            // map ResultSet to product
            Product product = resultSetToProduct(rs);
            result.add(product);
         }
      } catch (SQLException e) {
         e.printStackTrace();
      }
      return result;
   }
}
```

When the try block completes (successfully or abruptly), ResultSet's close (rs) is called first, followed by PreparedStatement's close (ps). If an exception was thrown from the try block, the calls to the close methods are made before the exception is handled or the catch block is executed. The code of resultSetToProduct is omitted here for simplicity.

2.2 Exception Handling with try-with-resources

The above JDBC examples showed that the resources are closed before the exception hander is executed. Then, what happens if a close method also throws an exception?

Let's first consider the case of a single resource. Statements (including those in the opening expression) of the try block may and may not throw exceptions and the close method of the resource may and may not throw an exception. Thus, there are four combinations. When there is no exception from any statement of the try block or the close method, the try block (and thus the try statement) completes with no error. If there is an exception (let's call it ExceptionTry) from a statement of the try block and an exception (call it ExceptionClose) from the close method, the exception from the try block is thrown as the main exception with the one from the close method as a suppressed exception (introduced in Java 7) of the main exception. If there is no exception from the try block, but an exception from the close method, the exception from the close method becomes the main exception. All those exceptions may be caught by a catch block of the try statement or propagated to the caller on the call stack. The following table summarizes the four cases.

Exception in Try Statements	Exception in close()	Final Exception from Try Block
None	None	No exception
ExceptionTry	None	ExceptionTry only
ExceptionTry	ExceptionClose	ExceptionTry with suppressed ExceptionClose
None	Exception Close	ExceptionClose only

When there is more than one resource listed in the try opening expression, the resources are closed in the reverse order of their appearance in the opening expression. Assume resources A, B, C are listed in the try opening expression in the given order, the following table extends the one above. When there is no exception from any statement of the try block and there are exceptions from the close methods (call them exceptions A, B, and C for resources A, B, and C, respectively), the first exception that was raised from the close methods is

the main exception and others become its suppressed exceptions.

Exception in Try	Exception in close()	Exception from Statement
None	None	No exception
ExceptionTry	C, B, A	ExceptionTry with suppressed C, B, A
None	C, B, A	C with suppressed B and A

Suppressed exceptions can be retrieved using the getSuppressed method declared in Java's Throwable interface, which is implemented by all exceptions in Java. The method returns an array of Throwable. The following pseudo-code shows an example of main and suppressed exceptions and how to retrieve suppressed exceptions. The example uses three resources (resA, resB, and resC), each of which throws, in its close method, an exception with a message of "Close Exception A[B,C]". The try block throws a RuntimeException.

```
try (Resource_A resA = new Resource_A();
     Resource_B resB = new Resource_B();
     Resource_C resC = new Resource_C();
    ) {
   // use the resA, resB, resC resources
   throw new RuntimeException("Exception in TRY");

} catch (Throwable t) {
   System.out.println("Catch: " + t.getMessage());
   Throwable[] suppressed = t.getSuppressed();
   for (Throwable s : suppressed) {
      System.out.println("Suppressed: " + s.getMessage());
   }
}
```

The output below shows that the RuntimeException thrown from the try block was caught by the catch clause as the main exception and the three resources were closed in the reverse order of their appearance in the opening expression and their exceptions were added to the main exception as suppressed exceptions. A complete copy of the test program can be found at `www.cs.scranton.edu/~bi/ccsc-e/NSuppresedExceptionsPaper.java`.

```
Catch: Exception in TRY
Suppressed: Close Exception C
Suppressed: Close Exception B
Suppressed: Close Exception A
```

2.3 Custom AutoCloseable Resources

Programmers can define their own resources to be used with try-with-resources by implementing the AutoCloseable interface. For example, if a temporary file is used in some computation and it should be deleted after the computation is completed, one may define a class, FileCleanUp, that implements the AutoCloseable interface:

```
class FileCleanUp implements AutoCloseable {
   String fileName;               // temporary file
   public FileCleanUp(String fileName) {
      this.fileName = fileName;
      createFile(filename); // method omitted
   }
   @Override
   public void close() {
      deleteFileIfExists(fileName); // method omitted
      System.out.println("File Deleted: " + fileName);
   }
}
```

The class may be used as shown below:

```
String fileName = "temp.txt";  // a temp file
try (FileCleanUp fcu = new FileCleanUp(fileName)) {
   System.out.println("Process File " + fileName);
   // computation using temp.txt
   // it may throw exception
}
catch (Throwable e) {
   System.out.println(e.getStackTrace());
}
```

The output of the program is shown below:

```
Process File temp.txt
File Deleted: temp.txt
```

3 The try-catch-finally Construct

While the try-with-resources construct provides an excellent way in managing resources that implement the AutoCloseable interface, the traditional try-catch-finally construct is still a necessary tool in managing resources that do not fit naturally with the closeable resources. Examples of such resources include Semaphore and Lock used in thread synchronization. Another type of such resources is objects (allocated via the new operator). When they are no longer needed, their references should be set to null to avoid object loitering [3].

The syntax of try-catch-finally is shown below. The catch block is optional, and more than one catch block may precede the finally block [6].

```
try {
   // try block
}
catch (ExceptionType a) {
   // handle the exception
}
finally {
   // code
}
```

The code in the finally block is always executed after the try block completes successfully or abruptly due to exceptions, even when there is a return statement in the try block. If the try block throws an exception and the exception is caught by a catch clause of the statement, the finally block is executed after the handler. If no handler is found in the construct, the finally block is executed before the exception is propagated to the caller in the call stack.

3.1 Semaphores in The Critical Section Problem

The leakingInsert example in the introduction section can be solved using the finally block. Below shows a revised version of the example, in which mutex.release is embedded in the finally block. Thus, mutex is always released when nonLeakingInsert completes, no matter successfully or abruptly with exception in the critical section.

```
Semaphore mutext;
LinkedList items;
public void nonLeakingInsert(Item item)
   throws InterruptedException {
   mutex.acquire();
   try {
      // critical section
      items.insert(item);
   }
   finally {
      mutex.release();
   }
}
```

It should be noted that the method specifies InterruptedException (which may be thrown by the acquire method of Semaphore) using the throws clause, not handled by the try statement. The reason for not handling it is that the nonLeakingInsert method is intended to be a library function and it does not know how the exception should be handled. Thus, the method should not hide the exception, but propagate it to the caller so it can decide how to handle the exception.

3.2 Managing Other Types of Resources

The above deleting temporary file example in the Custom AutoCloseable Resources section can be implemented using finally as shown below. This implementation is superior to the try-with-resources solution since it is simpler, no extra class (FileCleanUp) is needed, and the code is straightforward to read.

```
String fileName = "temp.txt";
try {
   System.out.println("Process File " + fileName);
   // computation using temp.txt
   // it may throw exception
} finally {
   deleteFileIfExists(fileName);
}
```

3.3 Avoiding Object Loitering

While Java does not exhibit the traditional memory leakage problem as in C due to Java's garbage collection capability, it may suffer a memory management problem called **object loitering** [3], which means that the garbage collector could not collect an object (no longer needed) because there are still unnecessary references associated with it.

The finally clause may be used to avoid object loitering in presence of exceptions. Here is an example, in which an image identified by the index to an ArrayList of large images is processed and saved to a file if possible, then it is removed from the ArrayList. If the image is not removed, it may not be garbage-collected since it would be strongly referenced in the ArrayList. To ensure the image is no longer loitering after the method even in presence of possible exceptions thrown from methods process and saveToFile, a try-catch-finally is employed and the call to remove the image is enclosed in the finally block.

```
public void processSaveRemoveLargeImage(ArrayList<Image> images,
   int index) {
   try {
      Image image = images.get(index);
      process(image);      // may throw exception
      saveToFile(image);  // may throw IOException
   } catch (Exception e) {
      e.printStackTrace();
      throw e;
   } finally {
      images.remove(index);
   }
}
```

4 try-with-resources vs. try-catch-finally

Before Java 7, the finally block was the only way in managing resources in presence of exceptions as well as in normal execution. The ProductDao example above may be rewritten using try-catch-finally as shown below. It is obvious the code is much longer and is not as clean as the try-with-resources solution.

```
class ProductDao {
   private Connection conn;
   public ArrayList<Product> getAllProducts() {
      ArrayList<Product> result = new ArrayList<>();
      String sql = "SELECT * FROM product";
      PreparedStatement ps;
      ResultSet rs;
      try {
         ps = conn.prepareStatement(sql);
         rs = ps.executeQuery();
         while (rs.next()) {
            Product product = resultSetToProduct(rs);
            result.add(product);
         }
      } catch (SQLException e) {
         e.printStackTrace();
      }
     finally {
        if (rs != null) rs.close();
        if (ps != null) ps.close();
     }
      return result;
   }
}
```

In addition, this solution has two differences from the implementation with try-with-resources. One is that if the close method of ResultSet(rs) throws an exception, the finally block would terminate and thus the close method of PreparedStatement(ps) would not be called, which would lead to the leak of PreparedStatement. Yes, one may use try-catch to enclose the call to rs.close, and thus the call to ps.close would be executed. However, this would prolong the code further. The other difference is related to exception masking. With try-catch-finally, an exception raised from the finally block will mask any exception raised from the try block. In contrast, with try-with-resources, the exception thrown from the try block becomes the main exception and exceptions from the close methods of the resources are attached to the main exception as suppressed exceptions.

Please note that, while this paper presents try-with-resources and try-finally as two separate constructs, the finally clause can be perfectly used together with try-with-resources in one statement. This would become convenient, if not necessary, in situations where a mix of closeable resources and resources that

do not implement AutoCloseable are used. Try-with-resources would manage the closeable resources and the finally clause to handle the others.

In summary, try-with-resources is an excellent construct for handling resources that are released back to the system and implement the AutoCloseable interface. Try-catch-finally may be employed to manage resources that do not fit naturally with the type of closeable resources.

5 Resource Management in CS Curriculum

Resource management may not need to be introduced in computer science curricula as a separate topic, rather it should be taught together with resource acquisition. Whenever, a resource is needed, the instructor should discuss how to acquire and how to release them, especially, in presence of errors and exceptions. With closeable resources, try-with-resources should be employed; with resources that are not closeable, try-finally should be used. Instructors may want to use those constructs in all their small and large examples.

Proper resource management should also be reenforced in student assignments and projects. Instructors may make resource management an explicit requirement in all student assignments and projects – making it a core part of programming.

6 Conclusions

Resource management is a critical aspect of reliable and long-running applications in presence of errors and exceptions as well as in normal execution. Java offers a set of language constructs that may be used to manage resources in both situations. However, those constructs have not been integrated into computer science curricula as much as should have been.

Java offers the try-with-resources and try-finally constructs that can be employed in managing resources. The try-with-resources construct is most suitable to managing resources that implement the AutoCloseable interface. Many of the resources provided by Java such as files, network connections, and database connections all implement this interface since Java 7. Resources such as dynamically allocated memory (objects created via the new operator), semaphores, and locks, that do not implement the AutoCloseable interface, may be managed with the try-finally construct, in which resources would be released in the finally clause.

This paper is not aimed to provide a comprehensive coverage on resource management in general or in Java. It is intended to raise the awareness of resource management in presence of errors and exceptions as well as in normal execution.

References

[1] The 10 most popular programming languages. https://www.northeastern.edu/graduate/blog/most-popular-programming-languages/. Retrieved on May 26, 2021.

[2] Computer programming languages. https://www.computerscience.org/resources/computer-programming-languages/. Retrieved on May 26, 2021.

[3] Yaodong Bi and John Beidler. Memory management in java. *Journal of Computing Sciences in Colleges*, 33(3):46–56, January 2018.

[4] Alan Burns and Andy Wellings. *Real-Time Systems and Programming Languages: Ada, Real-Time Java and C/Real-Time POSIX, 4th Ed.* Pearson, 2009.

[5] Oracle. Class filteroutputstream. https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/FilterOutputStream.html#close(). Retrieved on May 26, 2021.

[6] Oracle. The finally block. https://docs.oracle.com/javase/tutorial/essential/exceptions/finally.html. Retrieved on May 26, 2021.

[7] Julien Ponge. Better resource management with java se 7: Beyond syntactic sugar. https://www.oracle.com/technical-resources/articles/java/trywithresources.html. Retrieved on May 26, 2021.

379.bib

A Hands-On Arm64 Lab for Computer Organization*

Robert Montante
Mathematical and Digital Sciences
Bloomsburg University of Pennsylvania
Bloomsburg, PA 17815
rmontant@bloomu.edu

Abstract

This paper describes a set of lab activities that are used to add a hands-on component to an undergraduate course in Computer Organization. The activities are based on an interpreter and a simulator for the Arm64 instruction set architecture, such as is used in recent Raspberry Pi computers running 64-bit operating systems. The activities have been used in conjunction with a textbook that covers computer organization in the context of Aarch64 and similar "RISC" architectures. A web link is provided to the source code and lab writeups for the activities [?], [?].

1 Introduction

Courses that introduce computer organization generally deal with the subject matter from an abstract viewpoint, because the concepts are relatively complex and highly structured. Topics include characterization of performance, choices for instruction set architecture, and implementation of the Fetch-Execute Cycle. Engineering-oriented approaches blend into digital design topics and dwell on logical designs for ALUs, registers, and other circuit components, combining these into layouts for CPU datapaths. Computer science-oriented approaches focus more on the choices made for the machine instructions and their relationship to high-level languages. Various textbooks cover the field with more or less emphasis on physical versus conceptual aspects.

Digital-design approaches provide some opportunity for hands-on exercises in the form of circuit design and logic equations, and simulation or implementation on inexpensive FPGAs, but the instruction-set choices are given with little discussion. Computer science approaches are more concerned with the instruction-set choices, but at an introductory undergraduate level it is difficult to provide hands-on activities because the issues involved are fairly complex and detailed from the outset – the closest one can come to a "hello world" exercise might be a calculator implementation, but this does not easily generalize to a program-controlled machine.

The software package described in this paper is an attempt to bring some hands-on engagement to the computer-science approach. It starts with a simulation of the Arm64 instruction set, by processing the listing files from the gcc "as" assembler. It proceeds with a dissasembler/simulator that operates on statically-linked executable files. Students are provided with the C-language source code for the interpreter and the simulator, and assignments involve adding additional instructions to the small Arm64 subset that is implemented initially. Some simple assembly-language test programs are provided; students should be able to read and understand assembly language but need not be able to write assembly code.

The system being simulated is an ARMv8 processor running 64-bit Linux. The software was developed on and for a Raspberry Pi running 64-bit Debian Linux [**?**]. However the software is pure C, and can be (and has been) run on a Windows computer with a suitable C compiler. The example programs use Linux kernel syscalls for input and output, which the simulator implements using statements like "fgets()" and "fputs()".

Another hands-on approach to computer organization is provided by the Null and Lobur textbook with its Marie simulator [**?**]. A significant difference here is that Marie is purely a pedagogical architecture, whereas Arm64 is an actual commercial design that dominates the smartphone market and is finding increasing adoption in personal and server computing environments.

2 Discussion

The lab activities consist of two parts, an introductory interpreter that is intended to get students up to speed, and a subsequent simulator that offers opportunities for extended assignments. Each part is presented as one or more webpages, which contain source listings for the interpreter program and for the simulator program. The webpages also contain links to assembly programs which are the "test data" for the interpreter and simulator. The URLs for the webpages are provided in the references at the end.

2.1 First part - interpreter

The introductory part is a program that reads and parses the listing file that gcc's "as" assembler optionally produces. The listing file is parsed rather than the original source file, so that the assembler can resolve symbol references and calculate relative memory locations. Each instruction is interpreted as it is read and parsed, and the resulting effects on the processor registers are displayed.

Students are given this program in the form of ".c" source files and ".h" header files, along with a Makefile. The lab assignment asks the students to save and compile the program, and answer some questions about its coding and how it operates. The goal is to get students thinking about reading, loading, and interpreting Arm64 instructions, as well as bring everyone up to speed on building C programs into command-line executables.

Assembly listings are provided as input, along with the original assembly source files. Students who have access to an Arm64-based assembler are encouraged to build their own assembly listings for practice, although this isn't necessary. (The author's department expects students to purchase a Raspberry Pi for a prerequisite course, so most students are equipped to do the entire lab on their own Arm64 system.) The programs do a simple text output and a looping operation, thereby exercising a conditional test and a branch as well as a kernel syscall.

2.2 Second part - loader/simulator with disassembler

After students have "gotten their feet wet" with the listing interpreter, the second part of the lab is a new program that loads the assembled and linked machine code, then steps through and simulates the instructions. The overall structure of the program is broadly similar to the preliminary interpreter, but the details are completely different.

The simulator is also more complete than the interpreter, and handles more complex assembly programs. Besides the two programs used with the interpreter, a very basic program consisting only of a nop instruction is provided to demonstrate the basic fetch, decode, and execute steps. Other programs include a number-to-hexadecimal-string converter, a NUL-terminated-string printer, a number-to-decimal-string conversion routine, and recursive factorial and fibonacci programs.

As with the interpreter, students are provided with the source files, and questions about the code require them to look through and understand the code. The main() function of the program accepts options to display the loaded program memory, to actually execute the program once loaded, and to control the amount of output during execution. If machine instructions are displayed as assembly mnemonics when they're fetched and executed, the program is

effectively disassembling the executable.

Loading the binary machine instructions requires parsing the executable files, which are in the ELF format. The format is described in e.g. a Wikipedia article [**?**]. Such files are divided into sections that define how the executable uses the computer's memory – where in memory the actual machine instructions should be loaded and how much memory is made available for data storage. Every Arm64 machine instruction is a 32bit word so the instructions are stored into an array of 32-bit integers, with memory locations and addresses translated into appropriate array indexes. Following this step, a display of the memory array is available as a list of hexadecimal-formatted memory addresses and memory contents. The array can also be saved as a binary file, which can be opened and examined using a binary editor such as hexedit.

2.3 The fetch-decode-execute cycle

Once an executable file has been loaded into the (simulated) program memory, the simulator runs it by performing a classical fetch-decode-execute cycle – essentially an infinite loop. The loop is terminated either by encountering an error in the program, or by executing the "program exit" kernel syscall (or by user interruption).

Fetching each instruction is straightforward. The simulated PC register provides a memory address that is converted to an array index, and the 32-bit value stored in the array is returned.

The decode phase involves determining what the machine instruction does, based on the bits of the instruction. The meanings of the bits in each instruction are documented in ARM developer documentation [**?**]. The regularity of the Arm64 instruction format makes it fairly easy to decode instructions. Each instruction is tagged so that the execute phase can decide what to do for the instruction. For convenience, the tag is actually a form of the instruction's assembly mnemonic, thereby providing the "disassembly" behavior of the simulator program. An internal data structure contains the tag, register choices, operand sizes, and any addressing mode in use, for use in the execution phase.

The execution phase is similar to that in the interpreter. It is a large, multiple-clause if-else structure based on the instruction tags. All instructions used in the test executables will decode, along with other instructions that haven't been used; however, not all instructions are implemented with an execution clause. Any instruction tag that isn't executed generates an error message stating "Unknown instruction". As part of the lab, students must identify instructions that aren't implemented, and add code to perform them. This is easily demonstrated: if the instructions are implemented correctly then the test executables run properly and produce expected outputs.

3 Analysis

The questions about the source code could and should be expanded to explore the operation of the interpreter and the simulator in more depth. The simulator's memory-loading code is primarily implemented in the fillmem() function, which is quite extensive. It is presented essentially "as-is" with no explanation because it has to parse the ELF format. A course that goes into the system software in more depth could examine fillmem()'s operation. The author's approach and the textbook in use focus on the processor's datapath and the operation of the fetch-execute cycle, so the important things for the students to understand are the conversions from program-oriented memory addresses to the simulator's memory-array indexes.

The fillmem() function itself can probably be rewritten more cleanly and understandably. A version 2 of the program will do this, or an advanced course could assign the rewrite as a coding assignment.

A design goal for the lab was to have students add implementations of additional Arm64 instructions. As it stands the supplied Arm64 executables include a few instructions that are decoded but whose executions are not implemented. The lab activities require the students to determine which instructions don't work, and add implementations for them to the execute portion of the program. The if-else structure of the execute() makes it relatively easy to extend the function for additional instructions.

The current suite of test executables covers some significant instructions, enough to develop understanding of the way the simulator executes instructions while allowing for interesting programs. Implementing other instructions such as "adc" or "eori" would just be additional work without adding materially to the process of simulating instructions. Other instructions, such as floating-point or SIMD/NEON instructions, require adding the FP/vector register set to the CPU model. This could be the focus of an advanced lab assignment or even a final class project.

4 Conclusion and Further Work

The Arm64 ISA is well suited to modeling at a level that is accessible to undergraduates. Between them, the interpreter and simulator allow visualization of how assembly language works and what the processor actually does, which textbook readings and discussion questions don't provide. Students completed course evaluations at the end of the inaugural semester for the lab. They were very enthusiastic about the activities, especially compared to the homework assignments based on questions from the Patterson and Hennessy textbook [**?**].

The next logical step for the lab is to replace the"if-else" based simulation of the execution phase with with an approach that more closely emulates a hardware implementation of the datapath. This would reinforce the datapath development found in textbooks such as Patterson and Hennessy [**?**] and Harris and Harris [**?**] . The programs could also be adapted to similar ISAs such as the MIPS architecture described in the MIPS versions of Patterson and Hennessy or Harris and Harris, or perhaps the recent RISC-V architecture.

Development of a Virtualized Security Operations Center*

Robert de Céspedes III and George Dimitoglou
Department of Computer Science and IT
Hood College
Frederick, MD 21701
root@robdc.com, dimitoglou@hood.edu

Abstract

A centralized Security Operations Center (SOC) provides real-time network traffic and activity monitoring to detect, contain, remediate, and respond to attacks and security vulnerabilities. Developing monitoring infrastructures can be costly and resource-intensive. The motivation behind this work was to create a fully functional, yet low-cost SOC infrastructure that can be easily configured, deployed, and used for experimentation, research, and pedagogy. In this project we used virtualization technologies and leveraged existing open-source or free software to create data pipelines and workflows that integrated various different non-redundant tools. This resulted in a robust SOC platform that allowed the real-time collection, parsing, and analysis of network traffic. This platform also provided several visualization and data analysis facilities that enabled real-time monitoring and threat visualization.

1 Introduction

Organizational network infrastructures have grown dramatically in complexity and in the number of supported devices. Simultaneously, the average number of security breaches reported by organizations and the time for incident detection and remediation has steadily increased [9]. Network and endpoint hardening is no longer sufficient to protect networks; therefore, real-time threat detection and prevention are required.

A SOC provides the necessary "watchful eye´´ on network assets by identifying attacks as they happen while also collecting activity and traffic logs. The collected activity and traffic logs can be used to help in attack prevention, vulnerability remediation, and incident response. From an organizational perspective, a SOC provides situational awareness through reporting of real-time information, such as the number of monitored assets, along with threat, vulnerability, risk, incident, resilience, and compliance metrics that help assess the security posture of managed networks and systems. The organizational benefit of having a SOC is both high and immediate; however, the required infrastructure and personnel investment to establish and operate a SOC can also be significant.

In this paper, we describe the architecture, components, configuration, and deployment of a small SOC based on virtualized technologies. The primary objective for designing and building this SOC was to demonstrate the building of a modern network management and monitoring environment based exclusively on commodity hardware and open-source or free software tools. The secondary objective was to create a realistic educational infrastructure that can be used for teaching and scholarly research. The deployment of this proof-of-concept small SOC would provide an affordable form of meaningful network management, threat monitoring, and visualization capabilities according to common and best industry practices.

We focused the work on three core areas: (a) asset management, based on system information and event management (SIEM) software to capture, store, and analyze real-time event data; (b) network monitoring, establishing a distributed log transfer pipeline from the network's firewall and devices to a management system that converts raw data into useful information in real-time; and (c) physical monitoring, utilizing a closed-circuit television (CCTV) type system for continuous surveillance of hardware assets and the physical perimeter.

In this report, we provide a brief background on the key topics related to the establishment of centralized SOCs. We next describe the SOC's technical scope, network topology, architecture, and technology stack. A description of the deployed SOC is presented last. We end the report with a summary of the results and identify potential opportunities to further extend this work.

2 Background

The notion of having centralized security operations is not new, although the increasing volume, intensity, and impact of cyber attacks have underscored their importance and necessity. Enterprises, regardless of their industry, need to monitor their technology infrastructures. At-risk infrastructures vary from

those in typical business settings to environments that support the internet of things (IoT) [36, 25], supervisory control, data acquisition (SCADA) system infrastructures, and academic environments [11].

There is considerable interest and work on the technical and operational aspects of SOCs [35, 30], and the role and integration of SOCs within the organizational context of an enterprise [26]. Similar interests exist in academic settings with some institutions establishing SOCs to support educational and research activities [2, 4, 3].

As the building of SOCs become common practice, particularly for larger organizations, there is an effort to identify an architectural framework for a SOC [32, 13, 21, 17, 14] and establish best practices [20]. Towards this end, various deployment [28, 12], and analytical [21] models have been established and proposed. These models attempt to enhance the detection of common and sophisticated exploits along with defining the activities, processes, and organizational context of operation.

Key features of any SOC are log management, event visualization, and incident reporting. These three features are closely linked since the collected logs serve as the input for visualization subsequently used for incident reporting. Logs require storage, management, and often extensive processing (i.e., categorizing and prioritizing logged events) before becoming useful [23]. Once transferred to the visualization systems via different frameworks and methodologies [22], the log data is converted to meaningful graphical interpretations that allow SOC operators to quickly determine the health and security of any network asset, and by extension, the organization's overall security posture. As the virtualization systems are the SOC's interfaces and visual displays designed to be used by human operators [24], there is continuous interest in assessing their effectiveness. This is an intriguing area of exploration as it crosses several fields, including cognitive science, software engineering, and user interface design.

Much effort has also been made in identifying ways to assist SOC operators in handling the overwhelming data volumes and the high velocity of events that must be quickly analyzed and used for anomaly detection. Several studies and tools have attempted to use machine-learning [16, 27], event correlation [37], and deep learning [19] to automate the interpretation of network events and detection of anomalies.

Based on the body of work described above, we relied heavily on the areas that addressed the establishment, configuration, and deployment of the required hardware and software components to build an operational SOC. Advanced topics that related to the optimization of incident detection, operator training [15], and performance assessment [10] were not factored in this project but considered for future work.

3 Project Description

Every SOC is purpose-built: it is designed to support a specific environment within a certain organizational context and a specific information technology infrastructure. In this section, we describe the scope, architecture, and software and hardware building blocks of our proof-of-concept environment.

3.1 Scope

Our SOC was designed to function in an academic environment that can be assumed to support the activities of an academic department with multiple programs equivalent in size to a small business with under 50 employees. In our institutional context, this meant monitoring the infrastructure of several teaching and research-related computer servers and the devices associated with the work of faculty, staff, and a number of teaching and research assistants.

The network environment, technology stack, and services were assumed to be diverse, supporting endpoints with a mix of common operating systems (e.g., MS-Windows, OSX, and Linux), web servers, file-sharing servers, high bandwidth (i.e., streaming and telework) services, and casual network users. The SOC platform was expected to provide the means to log and visualize real-time normal and anomalous network activity, and then aggregate events while highlighting the security status for each network asset along with providing surveillance of the physical environment.

3.2 Network Topology

It is important that the SOC placement in the overall network architecture is determined based on security considerations and access to endpoints and networks to be monitored.

To ensure the security of the SOC, all network traffic outside the network perimeter flowed through a firewall. Every packet was logged at the firewall, and the logs were sent to the SOC tools for analysis. By configuring the network topology in this way, it was simple to build data monitoring (choke) points to ensure that all network traffic was captured. The traffic from both hard-wired devices and wireless devices was forced through their respective hubs and the firewall (Fig. 1).

3.3 SOC Architecture

To ensure the simplicity of the architecture and separation of concerns, the SOC was designed using distinct function-specific components to perform specific tasks. For the sake of generality, all the network devices (endpoints) were grouped since they all participate in monitoring in a similar way. The firewall

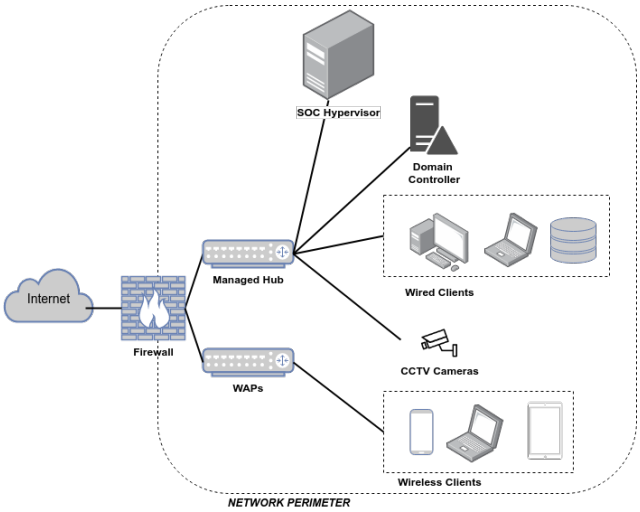


Figure 1: The network topology diagram illustrates the relationship between all incoming and outgoing traffic flowing through the firewall, managed hub, and wireless access points.

and managed hub devices were part of the overall network infrastructure. The SOC-specific components were as follows (Fig. 2):

1. Log Management Application. Served as a log aggregator, collecting all logged activity from endpoints and any network-based components (e.g., domain controller). This application also received logs from an intrusion detection system installed on the firewall. The data captured here was raw, so certain metrics had to be identified to allow the display of fewer, but more meaningful, data points. The most useful information present in the raw data was an active list of destination IP addresses, packets passed vs blocked, and a listing of all the interfaces on the firewall ranked by activity.
2. Visualization Application. This application provided significantly richer visualization options along with search facilities that enabled a more in-depth drill-down of collected data.
3. System Information and Event Management (SIEM) Application. This application conducted vulnerability scanning and network traffic analysis while also offering certain reporting capabilities for threat visualization. The advantage of the SIEM application came from its direct interaction with endpoints. Unlike the previous two applications that used raw log data, this agent required an install on each endpoint desired to be mon-

itored.

4. Physical Security Monitoring Application. The last aspect monitored in the SOC is physical security. Our design integrated the data feeds from a CCTV system, providing continuous footage of selected physical assets.

The implementation and deployment of these applications leveraged virtualization, allowing for the configuration and setup of dedicated virtual machines on a single hypervisor.

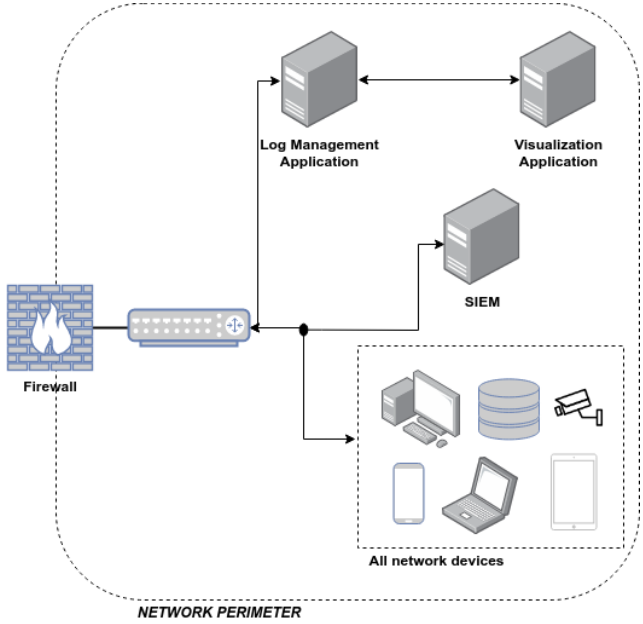


Figure 2: A diagram of how data flowed through the tools used in the SOC.

3.4 Methods: Technology Stack

One of the design goals behind the implementation of this SOC was the ability to use commodity hardware and open source or free tools. The use of virtualization, provided cost-saving opportunities, significantly reducing the required hardware investment if the SOC components were to be running on individual servers.

The firewall was configured with pfSense [31] and Snort [34] for intrusion detection. Microsoft's Hyper-V virtualization environment [15] was used to host the SOC's components as virtual machines. Graylog [1] was used for

log aggregation and management, Grafana [18] for visualization and advanced search capabilities, and AT&T's open-source tool Open Source Security Information Management (OSSIM) [29] was used as the SIEM solution. Wireshark [7], in concert with OSSIM, performed weekly vulnerability network scans, reported anomalies, or (depending on the severity) triggered alarms. Altogether, these tools gave a much deeper understanding of the traffic and network activities. Finally, ZoneMinder [8] was deployed as a camera system recording server to capture input from the CCTVs. Table 1 provides a mapping of the SOC components and the associated hardware and software parts that were used.

At the time of completion and deployment of the SOC, the total cost of all hardware and software did not exceed $1,600 USD. Throughout the installation, configuration, and deployment, best practices were consistently followed for securing (hardening) individual systems and tools.

SOC Component	Description (qty.)
Hardware	
Hardware Firewall Device	Qotom Q330G4 MiniPC, 4 NICs (1)
Host (hypervisor)	Dell R720 (1)
Monitors	Samsung 24"(4)
Storage server	Synology DiskStation DS218 (1)
CCTV Cameras	FosCam HD Indoor/Outdoor (9)
Software	
Operating System (hypervisor)	MS Windows Server 2016 Datacenter Edition; Hyper-V
Firewall	pfSense
Intrusion detection system	snort
Log Management Application	Graylog
Visualization Application	Grafana
SIEM Application	OSSIM l
Packet analyzer	Wireshark
Physical Security Monitoring Application	ZoneMinder

Table 1: A diagram of how data flowed through the tools used in the SOC.

4 Results: SOC Operation

After deployment, the SOC began to collect, process, analyze, and present data using the various tools listed in Table 1. A four-monitor set-up provided a summary view of the most important visualization components (Fig. 3).

During operation the overall functions of the SOC operation can be divided into two categories: (a) log analysis and (b) active SIEM observation and vulnerability scanning. Each category also offers its own set of configurable

data and threat visualization.

The log analysis portion of the SOC received logs from the firewall and the IDS and then parsed the data for anomalies and known vulnerabilities. The SIEM tool monitored network traffic using the packet analyzer and ran scheduled network-wide vulnerability scans. The data was shared with both the SIEM tool and the visualization application at all times. This first portion of the SOC was operationally characterized as a batch or passive, given that it is based on receiving, processing, analyzing, and displaying information received by voluminous logs.

The second part of the SOC was characterized as more interactive and active as the SIEM configuration engages in active vulnerability scanning and allows for the installation of an agent at each endpoint being monitored. Unlike the log analysis in which entire logs are shared for analysis, the role of the agent is more refined; it is designed to identify issues and immediately report them.

The final element was the integration of the CCTV streams that provided physical security and visualization monitoring. The video streams were routed and stored by the physical monitoring application.

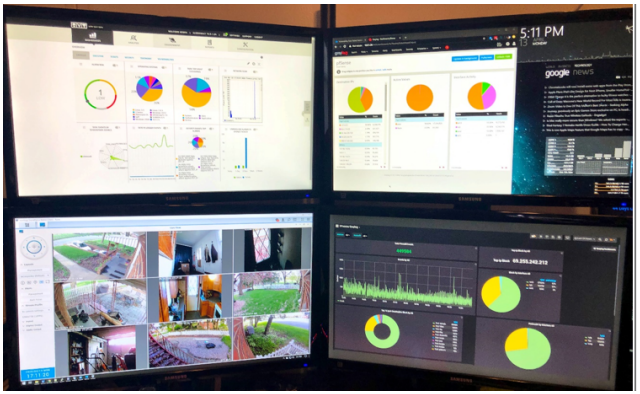


Figure 3: The culmination of all the configured tools monitoring the network environment. Top left: OSSIM's real-time dashboard. Top right: graphical data representation from Graylog. Bottom right: a deeper layer of network traffic analysis by Grafana, plotting firewall data coming through Graylog. Bottom left: live streaming feeds from CCTV devices monitoring physical security.

4.1 Conclusion and Future Work

Significant increases in cybersecurity attacks and devastating impacts of breaches have led many organizations to establish centralized security operation centers. These centers provide monitoring of incoming and outgoing network traffic along with the state of any device on the network. This type of monitoring is very useful in quickly detecting attacks and the spread of malicious software while being invaluable in reducing the time required for containment and remediation during incident response. The architecture and tools required to deploy a SOC vary and in large organizations the cost and required resources can rise quickly with scale.

The primary objective of this project was to deploy a proof-of-concept system that would provide real-time monitoring and visualization of network activities. Such a system could be easily deployed in an academic environment for teaching and research purposes with the added benefit that the system is robust enough to be used for actual security monitoring. Most existing environments are designed for students able to visit a physical facility. In this project, we set out to develop an inexpensive virtual SOC based on commodity hardware and existing open-source or free tools. Microsoft's Hyper-V virtualization environment was used to host the SOC's components as virtual machines in this implementation due to its availability, but it could be substituted for VirtualBox [5] or VMWare Workstation Player [6]. Altogether, these features reduce the cost barrier of deployment, enabling more widespread use, such as in rural areas, underserved communities, or organizations with limited space.

The success of the project was dependent on meeting the requirement of being low-cost. This was achieved through the flexibility and cost savings afforded by the virtualization of the SOC components. An unexpected benefit of virtualization was the realization of how easy it was to manage all of the SOC's components in a centralized way while having great flexibility to easily change configuration parameters, test, revert if necessary, or deploy new functionality in a matter of minutes.

The focus of this work was exclusively on the configuration and integration of the SOC components rather than its operational effectiveness. As future work, one potential interesting direction would be to assess the effectiveness of the SOC using an existing evaluation framework like SAIBERSOC [33] which allows the automatic generation and injection of synthetic attacks to evaluate various metrics of interest.

References

[1] [30] industry leading log management. Accessed: June 12, 2021. Available online from: www.graylog.org.

[2] International consortium of minority cybersecurity professionals (icmcp)/cyversity. Accessed: September 14, 2021. https://www.icmcp.org/index.php?option=com_content&view=article&id=54:educational-security-operations-center&catid=20:site-content&Itemid=147.

[3] Oregon research and teaching security operations center. Accessed: September 14, 2021. https://impactstudio.oregonstate.edu/oregon-research-and-teaching-security-operations-center.

[4] University of west florida - cyber operations and security training (c.o.a.s.t.) lab. Accessed: September 14, 2021. https://uwf.edu/centers/center-for-cybersecurity/about-the-center/facilities/.

[5] Virtualbox. Accessed: September 14, 2021. Available online from: www.virtualbox.org.

[6] Vmware workstation player. Accessed: September 14, 2021. Available online from: https://www.vmware.com/products/workstation-player.html.

[7] Wireshark go deep. Accessed: June 12, 2021. Available online from: www.wireshark.org.

[8] Zoneminder. Accessed: June 12, 2021. Available online from: www.zoneminder.com.

[9] *The Cost of Cybercrime*. Accenture and Ponemon Institute, New York, NY, USA, 2018.

[10] E. Agyepong, Y. Cherdantseva, P. Reinecke, and et al. Towards a framework for measuring the performance of a security operations center analyst. *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2020.

[11] L. Aijaz, B. Aslam, and U. Khalid. Security operations center - a need for an academic environment. *2015 World Symposium on Computer Networks and Information Security (WSCNIS)*, pages 1–7, 2015.

[12] W.P. Aung, H.H. Lwin, and K.K. Lin. Developing and analysis of cyber security models for security operation center in myanmar. *2020 IEEE Conference on Computer Applications (ICCA)*, 2020.

[13] R. Bidou, J. Bourgeois, and F. Spies. Towards a global security architecture for intrusion detection and reaction management. 2004. Berlin, Heidelberg: Springer Berlin Heidelberg.

[14] P. Bienias, G. Kołaczek, and A. Warzyński. Architecture of anomaly detection module for the security operations center. *2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2019.

[15] S.Y. Cho, J. Happa, and S. Creese. Capturing tacit knowledge in security operation centers. *IEEE Access*, 8:42021–42041, 2020.

[16] C. Feng, S. Wu, and N. Liu. A user-centric machine learning framework for cyber security operations center. *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2017.

[17] A.K. Ganame, J. Bourgeois, R. Bidou, and et al. A global security architecture for intrusion detection on computer networks. *2007 IEEE International Parallel and Distributed Processing Symposium*, 2007.

[18] Grafana. The open observability platform. Accessed: June 12, 2021. Available online from: `www.grafana.com`.

[19] N. Gupta, I. Traore, and P.M.F.d. Quinan. Automated event prioritization for security operation center using deep learning. *2019 IEEE International Conference on Big Data (Big Data)*, 2019.

[20] SANS Institute. Information security reading room: Common and best practices for security operations centers: Results of the 2019 soc survey. Available online from: `https://www.sans.org/media/analyst-program/common-practices-security-operations-centers-results-2019-soc-survey-39060.pdf`.

[21] S. Kowtha, L.A. Nolan, and R.A. Daley. Cyber security operations center characterization model and analysis. *2012 IEEE Conference on Technologies for Homeland Security (HST)*, 2012.

[22] T. Kwon, J. Song, S. Choi, and et al. Visnu: A novel visualization methodology of security events optimized for a centralized soc. *2018 13th Asia Joint Conference on Information Security (AsiaJCIS)*, 2018.

[23] A. Madani, S. Rezayi, and H. Gharaee. Log management comprehensive architecture in security operation center (soc). *2011 International Conference on Computational Aspects of Social Networks (CASoN)*, 2011.

[24] S. Mihindu and F. Khosrow-shahi. Collaborative visualisation embedded cost-efficient, virtualised cyber security operations centre. *2020 24th International Conference Information Visualisation (IV)*, 2020.

[25] N. Miloslavskaya. Security operations centers for information security incident management. *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 131–136, 2016.

[26] M. Mutemwa, J. Mtsweni, and L. Zimba. Integrating a security operations centre with an organization's existing procedures, policies and information technology systems. *2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, 2018.

[27] S. Oesch, R. Bridges, J. Smith, and et al. An assessment of the usability of machine learning based tools for the security operations center. *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, 2020.

[28] C. Onwubiko. Security operations centre: Situation awareness, threat intelligence and cybercrime. *2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security)*, 2017.

[29] OSSIM. The open source siem. Accessed: June 12, 2021. Available online from: `www.cybersecurity.att.com/products/ossim`.

[30] A. Perera, S. Rathnayaka, N.D. Perera, and et al. The next gen security operation center. *2021 6th International Conference for Convergence in Technology (I2CT)*, 2021.

[31] pfSense. World's most trusted open source firewall. Accessed: June 12, 2021. Available online from: `www.pfsense.org`.

[32] S.G. Radu. Comparative analysis of security operations centre architectures; proposals and architectural considerations for frameworks and operating models. *Innovative Security Solutions for Information Technology and Communications (Lecture Notes in Computer Sciences)*, 10006, 2016. Cham, Switzerland: Springer.

[33] M. Rosso, M. Campobasso, G. Gankhuyag, and et al. Saibersoc: Synthetic attack injection to benchmark and evaluate the performance of security operation centers. *Annual Computer Security Applications Conference*, pages 141–153, 2020.

[34] Snort. Network intrusion detection & prevention system. Accessed: June 12, 2021. Available online from: `www.snort.org`.

[35] M. Vielberth, F. Böhm, I. Fichtinger, and et al. Security operations center: A systematic study and open challenges. *IEEE Access*, 8:227756–227779, 2020.

[36] D. Weissman and A. Jayasumana. Integrating iot monitoring for security operation center. *2020 Global Internet of Things Summit (GIoTS)*, 2020.

[37] D. Zhang and D. Zhang. The analysis of event correlation in security operations center. *2011 Fourth International Conference on Intelligent Computation Technology and Automation*, 2011.

Integrating Cloud Computing into Computer Science Curriculum*

A pilot study with teaching introductory database courses

Aijuan Dong
Computer Science and Information Technology
Hood College, Frederick, MD 21701
dong@hood.edu

Abstract

In this study, we designed, implemented and assessed the integration of cloud computing education and public cloud services into introductory database courses. This is the pilot study for a curricular initiative that integrates cloud computing into core courses across computer science curriculum at a liberal arts college. Direct and indirect evidence show that it is feasible to integrate cloud computing basics into an existing course without impacting students' learning experience and curricular coverage and, given the current billing model, $50 Google cloud credits are enough for four hands-on course activities, but not enough to support students to continue their class projects after class. The paper also discusses cost involved, students' evaluation of Google Cloud Platform (GCP), students' perception of cloud computing, and difficulties encountered and lessons learned in the process.

1 Introduction

Cloud computing is the delivery of computing services—from applications to data storage and processing power—over the Internet ("the cloud") with a pay-as-you-go cost model. It revolutionizes the way that businesses are now acquiring, storing, and managing data. Increasing adoption from a variety of

domains makes it crucial for every CS and IT student to be equipped with knowledge, skills, and hands-on experience with this technology.

The widespread use of cloud services is beginning to percolate into the computer science curriculum. Based on Computer Science Curricula 2020[7], "Emerging curricula" areas include cloud computing, internet of things, edge computing, and among a few others. The report also points out "Employers frequently identify specific technologies or general knowledge areas (e.g. networking, cloud computing, systems analysis, and database)." As a result, more and more institutions are incorporating these contemporary topics into their CS curriculum to meet these guidelines and requirements.

1.1 Motivation

Most CS curriculum at our college is delivered with software and tools installed on institution servers—Each student is allocated an account for practice and projects, and these accounts are deactivated shortly after each semester ends. There is a big gap between the advances in cloud computing and its inclusion in our instructions. With the experience shared from[4][3], we argue that integrating cloud computing into core courses across our CS curriculum by means of public cloud services brings multiple benefits to our students and institution. It 1) provides our students with hands-on experience in deploying and managing an assortment of applications over the cloud, thus, improving their hireability; 2) enriches our already densely packed CS curriculum with the forefront of technology without adding a new core course; and 3) reduces management and maintenance responsibilities of and curriculum dependency on our local IT infrastructure.

1.2 Research Questions

Pedagogically, we argue dispersing cloud computing education into multiple core CS courses is a preferred model since the concepts and skills are introduced gradually. The ultimate goal of this project is to expose our students to quality cloud computing education by leveraging new technologies and teaching techniques. With that in mind, our first question is: Is it feasible to integrate the concepts and skills of cloud computing into an existing course without impacting student learning experience and existing curricular coverage?

Financial investment in a cloud-based approach has to be justified given the on-campus computing capabilities. Amazon and Google provide each student $100 or $50 usage credits per class via education grants, respectively. Our second research question is: Will these usage credits be enough for one class and, given the current billing model, is it feasible for students to host their projects on the cloud after class is over?

Our pilot run was conducted in spring 2021, during which we designed, implemented, and assessed the integration of cloud computing basics into our introductory database courses using Google Cloud Platform (GCP). This paper discusses the results and shares experience of this initial investigation.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 describes methods and resources, including the cloud computing platform and course components. Results and discussions are detailed in Section 4. Section 5 concludes this study and describes future work.

2 Related Work

A number of academic institutions have investigated ways to include cloud computing as part of their computer science curricula. Some institutions offer cloud computing as a stand-alone computer science course, either required or elective. [1][17][16][14]are a few examples. In addition, Coursera, an online course provider, lists a number of courses titled "Cloud Computing Specialization" offered by well-known Higher Education institutions and companies[2]. While this approach ensures the coverage of this topic, it presents some challenges to small liberal art institutions with scarce resources and an already densely packed curriculum.

In contrast to developing a standalone course dedicated to cloud computing, many other institutions integrate the topic of cloud computing with data science and Big Data analytics [6][13][10][11]. With these emerging topic courses, students are introduced to data analytics processes and tasks, algorithms, and tools. With this approach, the course focus is not cloud computing; the concepts, infrastructure, and economic foundation of cloud computing are mainly treated as background information and assisting technology.

Yet another approach adopted by some institutions is to integrate cloud computing topics with existing computer science courses, including high performance computing[12], networking and cybersecurity [18][15], databases[4] [9][5], and so on. With this approach, the instructional platform was switched to the Cloud and some instructors were also able to add new cloud-enabled components, such as query with BigQuery[4], enriching students' learning experience. However, the topics of cloud computing are covered together with curricular topics required in an existing class, which may not provide adequate depth and breadth of cloud computing, especially if it is only integrated in one or two existing courses among the whole CS curricula.

The above mentioned approaches, i.e., a dedicated cloud computing course, cloud-assisted courses, or cloud-enriched courses, have their own limitations and constraints. To offer cloud computing at a small liberal arts college like us, we plan to distribute cloud computing education into multiple core CS courses

so that the concepts and skills are introduced gradually and in multiple computer science areas, such as programming, database, networking, architecture, software engineering, and so on. The ultimate goal is to expose each and every student to quality cloud computing education by leveraging new technologies and teaching techniques. In this paper, we share the experience of integrating cloud computing into introductory database courses.

3 Methods

3.1 Cloud Platform

There are several industrial-grade public cloud service providers that offer free education credits to students from 2-year or 4-year institutions, including Google Cloud Platform (GCP), Amazon Web Services (AWS), Microsoft Azure, Oracle Cloud, etc. For this study, we picked Google Cloud Platform due to our collaboration with Google in its Applied Computing Series and participation in Google Faculty Institutes in 2017 and 2018.

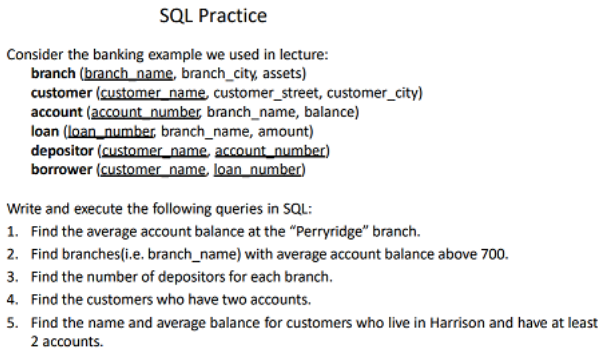
Google Cloud Platform (GCP) consists of a suite of cloud computing services and provides a variety of SQL and NoSQL database services[8]: a SQL database under Cloud SQL, which supports both MySQL and PostgreSQL; a full managed, mission-critical, relational database service under Cloud Banner; and two NoSQL databases, Firestore and BigTable. You can also set up other database systems on Compute Engine by using persistent disks.

Google Education Grant offers teaching and research credits to help university faculty and students explore tools and applications on Google Cloud. Faculty members apply for cloud credits for each class via a simple web application (`https://edu.google.com/programs/credits/teaching/?modal_active=none`). Upon approval, each student is awarded $50 worth of cloud credits per class while each faculty member or teaching assistant is awarded $100 per class. The credits can be used for most products in Google Cloud Platform.

3.2 Course Design

CS 329, Introduction to Database Management Systems is a required course for undergraduate students majoring in our BSCS program. Its graduate version (CS 530) is an elective for graduate students majoring in MSCS program. CS 329/530 is often taken by sophomores, juniors, or graduate students whose undergraduate major is not CS/IT related. This course covers the design and implementation of databases from a real world applications point of view. One of the course learning outcomes is to effectively construct simple and moderately advanced database queries using Structured Query Language/Data Ma-

nipulation Language (SQL/DML). There are two hands-on SQL labs designed for this outcome. Figure 1 shows SQL Lab 2.


SQL Practice

Consider the banking example we used in lecture:
branch (branch_name, branch_city, assets)
customer (customer_name, customer_street, customer_city)
account (account_number, branch_name, balance)
loan (loan_number, branch_name, amount)
depositor (customer_name, account_number)
borrower (customer_name, loan_number)

Write and execute the following queries in SQL:
1. Find the average account balance at the "Perryridge" branch.
2. Find branches(i.e. branch_name) with average account balance above 700.
3. Find the number of depositors for each branch.
4. Find the customers who have two accounts.
5. Find the name and average balance for customers who live in Harrison and have at least 2 accounts.

Figure 1: SQL Lab 2

There are two weeks allocated for this topic. Each week has two 75-minute sessions—the first session is lecture-based and the second one is lab-based, filled with instructor demos and student hands-on activities. During the lecture, we introduced cloud computing basics and SQL knowledge. Students learned not only SQL, but also what cloud computing is, what it supports, how its supported services are delivered, cloud economics, and GCP and its database services. During the lab time, the instructor first demonstrated how to use MySQL on GCP and then a local Linux server (The Pluto server) that is managed by the department administrator. In other words, students can use either GCP or a local server for the labs. This study setting was intentional since we wanted to observe students' general attitude toward this cloud platform adoption.

Please note besides the two SQL query labs, there are two other hands-on activities, each of which involves creating a database relation schema with less than ten tables. These activities are only involved in Section 4.2 Cost Study.

3.3 Assessment methods

To answer the two research questions posed in Section 1.2, we used both direct and indirect assessment methods to measure the impact on student learning experience and curricular coverage, and to evaluate the financial feasibility of using cloud services in and out of class.

Direct assessment: based on the course learning outcome, appropriate grading rubrics were developed. Then, student submissions were directly assessed to measure the degree to which students attain the learning outcomes and if

there is any difference in student performance between cloud platform approach and the local database server.

Indirect assessment: a carefully designed survey was developed to gather feedback on cost involved, impact on learning experience, and evaluation of the GCP with respect to usability, performance, reliability, etc. These questions are presented together with the student responses in Section 4.

4 Results and Discussions

The assessment results are grouped and discussed in the following areas: impact on student learning experience, cost study, evaluation of GCP, and open-ended questions on difficulties encountered and suggestions for improvement. The section ends with discussions of our research questions posed in Section 1.2.

4.1 Impact on Student Learning Experience

The impact on student learning was assessed in two ways: direct assessment via graded student work and indirect assessment via survey questions. Three survey questions were designed to assess students' perception of their interest in and skill level with SQL.

For the two SQL labs, about 63% of students conducted labs on GCP, while 37% of them did labs on the Pluto server. The Pluto server is a local Linux server with MariaDB installed; this server is managed by our department admin.

Table 1 shows the performance statistics for the two groups. The total points for the labs are 36. As can be seen, the statistics from two groups are very close. We conducted a two-tailed t-test with alpha = 0.05 on the mean values, the results showed no significant difference between the mean values for the two groups.

Table 1: Students' Performance in Graded Work

Platform	mean	min	max	median	standard deviation
GCP	30.92	25.0	36.0	32.0	3.26
Pluto	30.14	25.5	35.0	32.0	4.29

Three survey questions (Q2, Q3, Q4) were designed to assess the impact of GCP on student learning experience indirectly. These questions measure students' interest in the subject matter and their perception of their skill levels.

Q2. I found the topic of SQL interesting.

Q3. I would like to learn more about SQL and explore more in my future courses.
Q4. Rate your SQL knowledge and experience after this class.

Table 2 shows at least 90% of the students from both groups either "Strongly Agree" or "Agree" that the topic of SQL is interesting and they would like to learn more of the topic. It appears that a higher percentage of students in the GCP group are in the "Strongly Agree" category and slightly more students in the GCP group are in the "Neutral" category. Overall, students' high interest shows positive learning experience.

Table 2: Students' Interest in SQL

Question		Strongly Disagree(%)	Disagree(%)	Neutral(%)	Agree(%)	Strongly Agree(%)
Q2	GCP	0	0	8.3	8.0	83.7
	Pluto	0	0	0	71.5	28.5
Q3	GCP	0	0	16.7	8.3	75.0
	Pluto	0	0	0	57.1	42.9

Table 3 shows 83.4% of the students in the GCP group rated their skill level in SQL as either "Intermediate" or "Advanced," which is lower than that of the Pluto group. In addition, 16.6% of students in the GCP group rated their skill level as either "Clueless" or "Beginner," compared to 0% from the Pluto group. The survey response data from Table 3 appear to suggest students using the local environment, i.e., the Pluto server, are more confident about their grasp of the subject matter.

Table 3: Students' Perception of SQL Skill Levels

Question		Clueless (%)	Beginner (%)	Intermediate (%)	Advanced (%)	Total Guru(%)
Q4	GCP	0	0	8.3	8.0	83.7
	Pluto	0	0	0	71.5	28.5

Note all students used the Pluto server for a data structure/programming class before this introductory database class, but the database management system (MySQL or MariaDB) is new to all students. Students who conducted the labs on the Pluto server had a smaller learning curve with the lab environment.

4.2 Cost Study

Each student is awarded $50 cloud credits per class via Google Education Grants. As mentioned in Section 1.1, our plan is to integrate cloud computing into multiple core courses across our undergraduate CS curriculum. In other words, we expect only a few hands-on activities from each course using the cloud as an instructional platform. In this pilot study, we designed four hands-on activities as described in Section 3.2. At the end of these activities, we asked students to self-report how many cloud credits were left in their accounts.

Figure 2 showed about one third of students had less than $10 left, we do not know how many of them actually completely ran out of the credits. A little more than one third had more than $30 left. There are multiple factors contributing to the amount of cloud credits consumed. Responses from open-ended questions indicate some students constructed all SQL code outside the cloud and only started a MySQL instance from the GCP for testing the solutions. While other students struggled with remembering to shut down their database instances when they were done with their work.

Figure 2: Cost Study

4.3 Evaluation of GCP

Three survey questions (Q5, Q6, and Q7) were designed to obtain students' experience with and perception of the GCP used in this study.

Q5. The Google Cloud Platform is easy to set up and maintain.
Q6. The Google Cloud Platform is easily available and accessible.
Q7. The Google Cloud Platform is reliable and efficient.

Table 4 shows that about 37% of students rated the GCP platform as easy to set up and maintain, while 21% of students rated negatively on this aspect. In addition, about 32% of students rated the GCP platform reliable and efficient, while 16% of students rated negatively on this aspect. About 55% of students

rated the GCP platform easily available and accessible while only 6% rated negatively on this aspect. The comments from the open-ended question (Q9) revealed some difficulties students encountered. These comments are discussed in Section 4.5.

Table 4: Students' Evaluation of GCP

Question	Strongly Disagree(%)	Disagree (%)	Neutral (%)	Agree (%)	Strongly Agree(%)
Q5	10.5	10.5	42.1	31.6	5.3
Q6	5.6	0	38.9	44.4	11.1
Q7	5.3	10.5	52.5	21.1	10.5

4.4 Perception of Cloud Computing

In this study, we spent about one-hour lecture time on cloud computing basics. About three weeks were spent on instructor demos and student hands-on practices of SQL, including database relation schema implementation (DDL) and data manipulation and query (DML). One survey question (Q8) was designed to assess students' views and perception of cloud computing (Figure 4).

Figure 3 shows the majority of the students (about 80%) believe the integration of cloud computing is beneficial to their career. About 20% of the students were either not sure or negative toward this technology. Comments from the open-ended question (Q10) suggested some improvement actions. Additionally, when cloud computing is integrated in other core CS courses, these students will have more chances to learn and experience this technology. As mentioned in Section 1.1, we believe this scaffold integration of cloud computing into computer science curriculum is a better approach.

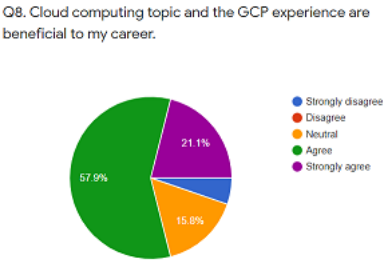


Figure 3: Students' Perception of Cloud Computing and GCP

4.5 Open-ended Questions on Difficulties and Suggestions

There are two open-ended questions (Q9 and Q10) in our survey. Question 9 asks "Difficulties encountered or concerns with GCP usage in our course." Major concerns are slow loading and cloud credits management, such issues as forgetting to turn off the service, or receiving constant bill updates, or getting anxious that the cloud credits would run out soon. A few students mentioned a deep learning curve and not intuitive user interface. These comments gave us some clues why 15-20% of the students rated the GCP experience negatively (Section 4.3).

Question 10 asks "Improvement suggestions of utilizing GCP in our BSCS curriculum." Student suggestions mainly focus on more knowledge of GCP, dedicated lecture time to walking through the environment, more in-class exercises, better on-ramp for beginners, and introducing the environment (for example, Google Cloud Shell Built-in Editor) earlier. Some students commented "it's great all the same to have had this experience in the class." "Definitely recommended, Google Cloud Platform it is very helpful." "I thought giving people the option of using the GCP or the local environment was good." Still a few students "don't like GCP" and preferred the local environment. One note from the instructor's perspective is that some students were familiar with the local Pluto server environment before this class, but new to GCP. There were no detailed hand-written instructions for using GCP other than demo videos. Additionally, due dates for SQL hands-on activities were firm. All these factors played into students' experience with the GCP and cloud computing.

4.6 Instructor's Feedback

Going back to the first research question posed in Section 1.2, we believe it is feasible to integrate cloud computing basics into our introductory database course without impacting our required curricular coverage negatively. Students' SQL lab grades reflect their thorough understanding of the topic and proficiency in constructing SQL queries. Their responses to survey questions (Section 4.1 Impact on Student Learning Experience) showed a relatively high percentage of students were interested in the topic, felt confident about it, and expressed interest in further study.

As for the second research question, $50 cloud credits are enough for the four hands-on activities designed for this class (Section 3.2 Course Design and Section 4.2 Cost Study). However, this amount is not enough to support students to continue their projects after class. In fact, we had to turn to the local environment for the group project. Some students commented more would be helpful. Cloud credits application via Google Education Grants is simple and fast. We were able to request more credit coupons in the middle of

the class.

5 CONCLUSIONS AND FUTURE WORK

This is our pilot study for a curricular initiative to integrate cloud computing education into core courses across the computer science curriculum at a small liberal arts education. Specifically, we designed, implemented, and assessed the integration of cloud computing basics into existing introductory database courses. Direct and indirect evidence showed student competency in subject matter and overall positive learning experience. The indirect survey responses also uncovered some issues and provided pointers for improvement.

With the experience gathered and lessons learned from this study, we will continue the integration of cloud computing into other core computer science courses. More advanced topics in cloud computing will be gradually and systematically penetrated into the curriculum. Additionally, the cloud provides an alternative instructional platform that expands and enriches our teaching and learning resources, an important factor for a computer science program at a small liberal arts college.

ACKNOWLEDGMENTS

This study was supported by Academic Innovation Grant from the Center for Teaching and Learning (CTL) at Hood College.

References

[1] Charles B. Border. Cloud computing in the curriculum: Fundamental and enabling technologies. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, page 147–152, New York, NY, USA, 2013.

[2] Coursera.com. `https://www.coursera.org/`.

[3] Karen C. Davis. Teaching database querying in the cloud. In *2019 IEEE Frontiers in Education Conference (FIE)*, pages 1–7, 2019.

[4] Debzani Deb, Muztaba Fuad, and Keith Irwin. A module-based approach to teaching big data and cloud computing topics at cs undergraduate level. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, page 2–8, New York, NY, USA, 2019.

[5] Aijuan Dong and Baoying Wang. Exploring instructional computing platforms for a small liberal arts college: A case study with nosql. In *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 332–337, 2016.

[6] Joshua Eckroth. Teaching future big data analysts: Curriculum and experience report. In *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 346–351, 2017.

[7] Association for Computing Machinery (ACM) and IEEE Computer Society: Joint Task Force on Computing Curricula. Computer science curricula. `https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf`.

[8] Google.com. About google cloud services. `https://cloud.google.com/docs/overview/cloud-platform-services`.

[9] Edward P. Holden, Jai W. Kang, Dianne P. Bills, and Mukhtar Ilyassov. Databases in the cloud: A work in progress. In *Proceedings of the 10th ACM Conference on SIG-Information Technology Education*, page 138–143, New York, NY, USA, 2009.

[10] Suzanne J. Matthews. Using phoenix++ mapreduce to introduce undergraduate students to parallel computing. *J. Comput. Sci. Coll.*, 32(6):165–174, June 2017.

[11] Ariel S. Rabkin, Charles Reiss, Randy Katz, and David Patterson. Experiences teaching mapreduce in the cloud. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, page 601–606, New York, NY, USA, 2012.

[12] Atanas Radenski. Integrating data-intensive cloud computing with multicores and clusters in an hpc course. In *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, page 69–74, New York, NY, USA, 2012.

[13] Bina Ramamurthy. A practical and sustainable model for learning and teaching data science. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, page 169–174, New York, NY, USA, 2016.

[14] M. Suhail Rehman, Jason Boles, Mohammad Hammoud, and Majd F. Sakr. A cloud computing course: From systems to services. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, page 338–343, New York, NY, USA, 2015.

[15] Khaled Salah. Harnessing the cloud for teaching cybersecurity. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, page 529–534, New York, NY, USA, 2014.

[16] Cornell University. Cs5412: Cloud computing. `http://www.cs.cornell.edu/courses/cs5412/2018sp/`.

[17] Berkeley University of California. Cs 294-170: Programming the cloud. `https://www2.eecs.berkeley.edu/Courses/CS294_1887/`.

[18] Chuan Yue, Weiying Zhu, Gregory Lynn Williams, and Edward Chow. Using amazon ec2 in computer and network security lab exercises: Design, results, and analysis. In *2012 ASEE Annual Conference and Exposition*, number 10.18260/1-2–22175, San Antonio, Texas, June 2012. `https://peer.asee.org/22175`.

A Successful Online Systems Class using Scaffolded Active Learning and Formative Assessment*

Michael O. Lam and Dee A. B. Weikle
Computer Science
James Madison University
Harrisonburg, VA 22807
{lam2mo, weikleda}@jmu.edu

Abstract

This last year has been a challenge for faculty transitioning from in-person instruction to online instruction. We approached the semester with serious concerns but discovered our hybrid course transitioned well to being completely online with a few key modifications: video versions of all lectures, Google slide implementations of in-class labs, course procedures that allowed students to choose their breakout room in Zoom, and randomization of question selection for midterm and final exams. In person, this course made thoughtful use of active learning and formative assessment to scaffold students into summative assessment. The online version does the same and, in our opinion, this contributed to the online success. This paper discusses the scaffolding formative assessments, how they build to the summative module tests and midterm and final exams as well as how we implemented active learning online. We also provide links to the formative labs and videos that made the active learning work.

1 Background and Related Work

Here we provide a discussion of formative assessment, primarily through selected work from computer science education literature because a comprehensive review is outside the scope of this paper.

Formative assessment is assessment designed to facilitate student learning. The assessment highlights student misconceptions for instructors so that they can tailor instruction and help students to self-assess and to correct misconceptions. This correction can take place in the classroom or in some cases by feedback to the student followed by an opportunity to retake the assessment. In contrast, summative assessment does not provide opportunities for students to resubmit work and is designed primarily for grading. Interested readers can see Black [1] and Dunn [6] respectively for a more in-depth literature review of formative assessment and a critique of the related research.

Nelson [9] provides an example formative assessment for tracing JavaScript programs along with a discussion of Kane's framework for assessment validity. It provides a detailed analysis of their formative assessment arguing for its validity using the four parts of Kane's framework: scoring, generalization, extrapolation and use. This article is particularly useful for its discussion of assessment design, explaining the importance of granularity in formative assessment such that questions clearly identify the specific skill being assessed and examples of confounds such as slips (misreading an operator) and guesses that might obscure the results of such an assessment. By carefully structuring their formative assessment they demonstrate the application of the validity argument and the process of designing such an assessment. Blaheta [2] describes transforming traditional pen and paper homework assignments in an Artificial Intelligence class into cooperative formative assessments by allowing students to submit homework in self-selected groups, receive comments, and resubmit for a final grade. Duarte [5] describes a blended learning environment where engineering students are given formative assessments in a learning management system (LMS) in addition to attending class face-to-face with the goal of improving learning outcomes and preventing premature withdrawal or unexpected failure at the end of the course. She saw performance correlation between the formative assessments in the LMS and final exam performance. Cauley and McMillan [4] discuss five best practices in structuring formative assessment: 1) provide learning targets, 2) give targeted feedback, 3) clearly attribute successes to effort, 4) encourage self-assessment, and 5) help students set attainable goals. Finally, Grover [7] provides an argument for formative assessment (specifically in the K-12 context but we believe the argument is applicable to higher education as well).

Our work is not as formal as that in Nelson, but is more of a combination of the approaches taken by Blaheta and Duarte. We make use of four types of formative assessment in the in-person version: 1) reading/conceptual quizzes in our LMS (Canvas), 2) instant-feedback questions during short lectures, 3) labs on paper that may include running or developing small programs, and 4) larger programming assignments in C where students are able to run provided

unit and integration tests to make sure their code is working properly prior to grading. In addition, we provide links to the labs via Google slides and companion videos that enabled this approach to work completely online. Section 2 describes the overall structure of the course and goes into detail about the different formative assessments.

2 Course Structure

Our course, Computer Systems I, is a basic computer architecture class combined with medium- to large-sized programming projects in C along with selected basic operating systems concepts. It serves as the foundation upon which our other computer systems courses build. Figure 1 is a high-level view of the topics covered. The course currently uses Computer Systems: A Programmer's Perspective by Randal E. Bryant and David R. O'Hallaron (CS:APP) [3]. The learning outcomes are as follows:

1. Explain the machine-level representation of data and code.
2. Summarize the architecture of a computer.
3. Explain how powerful, complex systems can be built from simple logic circuits.
4. Translate high-level code blocks into assembly and machine language
5. Write code to emulate the functionality of a computer.
6. Cultivate a sense of power and control over computer systems.
7. Gain an appreciation for the tools that facilitate software development.
8. Develop a sense of play when writing code.
9. Appreciate the principles and complexity of systems-level software.

Pre-pandemic, this course was a hybrid course designed for a 75-minute twice-per-week meeting format. Some assessments were completed online in Canvas, but there was a significant in-person experience. Before each in-person class period, students were assigned a reading from CS:APP and a Canvas quiz to complete online prior to coming to class. Each class started with a chance for students to ask questions about the quiz followed by a mini-lecture with instant-feedback questions. The second part of each class consisted of a lab worksheet that students worked on in groups. In addition, there were five programming projects, five Canvas module tests, and two in-class paper exams (a midterm and a final). The module tests were summative timed assessments associated with each module that students could take over a period of 2-3 days after a module had been fully covered in class. The modules correspond to the five high-level topics shown in Figure 1.

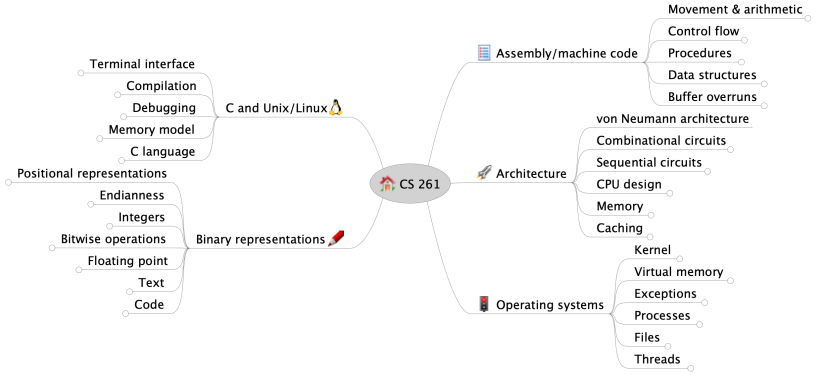


Figure 1: Computer Systems 1 Topics

2.1 Canvas Quizzes

The Canvas quizzes began as a way to make students accountable for the reading assignments but have evolved over time to include additional questions on basic concepts. These quizzes are formative because students see which questions they get wrong and are allowed to review the material, ask questions of each other or the instructor, and retake the quiz once. The quizzes make it clear what the instructors want students to know coming in to class. As class begins, the answers are available to students and they have the opportunity to ask questions about any remaining confusion from the quiz.

2.2 Instant-Feedback Questions

After students read about the content for the day and take the quiz, faculty lecture on the material in class, emphasizing concepts and giving examples. These lectures include another type of formative assessment – instant-feedback questions geared toward making sure students can remember key lecture concepts and do simple problems associated with those concepts. We used Socrative, an online Q&A platform that allows students to enter a code and answer questions associated with an ongoing lecture. Students are given credit for simply participating regardless of whether their answer is correct. The instructor can see the incorrect answers and can share the distribution of answers with the students. The instructor then gives the right answer, explaining how to get it, and why the other answers are wrong. This is another opportunity to correct student misconceptions in class.

2.3 In-class Paper Labs

Once the lecture is finished, the second part of the 75-minute class is dedicated to doing paper labs in groups of 2-3 students. These labs may ask students to run code on a lab machine or the student's personal computer. Each lab is a two-page series of questions designed to take students deeper into the material, scaffolding them into later exams or programming assignments. During this time, faculty monitor progress and answer student questions or point out mistakes as they crop up. Students must each turn in their own paper by the next day, but they are encouraged to work in groups both inside and outside of class. With labs due the next day, students also have an opportunity to work together more, go to open TA lab hours, or ask questions on a Q&A forum such as Piazza. Q&A responses are shared across all sections and monitored by both instructors. Questions can be answered by other students in the class or by the instructors and viewed by all students. In this forum, students can ask and answer questions showing up as anonymous to other students. With feedback during and outside of class, the labs provide another deeper formative assessment. Labs are then loosely graded with incorrect answers marked, but about half of the points awarded simply for turning in the assignment. Solutions are released to show students what was expected or possible correct answers. As with other course elements, discussion with faculty about any remaining confusion is encouraged, but students are not allowed to redo the labs after grading. A short description of each lab is included in Table 1.

2.4 Programming Assignments

Programming assignments for this course are described in detail by Weikle [11]. These assignments are designed to illustrate the underlying concepts while practicing C programming. Projects include tasks such as opening a binary file and reading in an object code header, loading object code regions into a memory array, decoding and disassembling machine code instructions from that code, and simulating the fetch-decode-execution cycle of machine code instructions. We use the Y86-64 instruction set architecture from CS:APP [3] with our own custom ELF-like object file format.

Programming assignments are released every 2-3 weeks and require students to apply the concepts after they have been introduced in readings, class, and labs. Students complete the programming assignments individually and are expected to collaborate only on concepts. In addition to the specification, students are given all the tests used to determine the functionality of their code, some of which are private (pre-compiled and stripped of symbols) while others are public (source code provided). These tests are organized into sets of tests for a particular grade level in a manner inspired by specification grading.

All tests for a particular grade level must be passed as well as all tests for any grade below that level to receive that particular grade. For example, to get a D a student would only need to pass the D level tests, but to get a C, they would have to pass all the D tests and all the C tests. Students are able to run the whole test suite at any time in development and receive automated feedback about their errors. This also enables them to look at the public tests for examples of how to test their code themselves. Finally, the bundling of tests into grade levels gives students a general guide to the most effective order of functionality implementation (i.e., the lower grade levels are the easier tasks to implement). In addition to the automated feedback and support, students can visit TA lab hours and instructor office hours as well as post questions to the previously mentioned Q&A forum.

3 Transitioning Online

The transition to a fully-online format kept the same routine with two significant modifications: 1) videos instead of lectures and 2) synchronous class in Zoom using Google slides for labs. We provided pre-recorded videos rather than lecture via Zoom because the videos can be downloaded and watched at any time even with a slow connection, be slowed down or sped up at the discretion of the student, and used for review or if a student missed class. One side effect of this change is that now the entire class period could be used for active learning using the labs, which we knew would be more difficult online. To maximize the effectiveness of the videos we limited their length to an average of 5-10 minutes and created playlists so students could break up their viewing sessions (this also simplifies searching for review content later). The instant-feedback questions were still provided in the videos with a prompt to pause the video and answer the question, followed by a description of the answer.

Active learning is key to keeping students engaged and maximizing learning. We found our method worked well to provide students support with technical content as well as helping them interact with other students to make friends and keep motivated. Prior to class, we set up a Google Doc with directions, announcements, and a set of breakout rooms (a Zoom feature that allows one call to have multiple isolated sub-calls; this feature is now also provided by several other videoconferencing providers) that students could sign into when they came to class. We also set up Google Slides with the lab questions as a static PDF background, adding text boxes or base diagrams for students to edit as their answers to the questions. [10]

When students entered the Zoom session (linked from Canvas), they also opened the folder for the day that contained the sign-up sheet and the empty lab template. Students signed up for a breakout room and then made their

own copy of the Google slide version of the lab. After an initial discussion with announcements, the instructor opened the breakout rooms and students worked together in these groups to complete the labs. Much like when students come to class in person and tend to sit with the same group of classmates each class period, the students in a particular breakout room were often the same, creating friendships and strong working relationships. Instructors could then monitor the Google slides in real time, making comments as students worked to encourage them or help correct their thinking. Students could also ask for an instructor to come into the breakout room to ask for help if they were stuck or unsure. If several groups were asking for help with the same issue, the instructor had the ability to bring everyone back to the main room, give a quick explanation of the issues allowing for questions, and then send everyone back to their breakout rooms.

Labs were due toward the end of the day. To turn in labs, each student had to download their own Google slide version as a PDF file and submit to Canvas. This allowed students to have more time if they needed and some would get together outside of class time to complete labs or to help each other with difficult concepts or sections. In most cases though, student could complete the majority of the lab in class. Labs were still loosely graded by the instructor with the majority of the credit going to turning in something. Solutions were released after the due date similar to in-person classes. The formative aspect of these assignments is that faculty can give feedback and guide students, noticing misconceptions and correcting them during class just as if we were in person.

The last modification we made in the online environment was the administration of the midterm and final exams. In person, these exams are held in the classroom and proctored by the instructor. Online, we made these exams in Canvas using primarily multiple choice, matching, and fill-in-the-blank questions. Our primary concern was to test similar skills online to what we have done in person to make sure that students are prepared for the next class in our systems curriculum. To this end, we used very similar questions or the same questions as our previous exams. Occasionally this required us to modify the question type slightly (e.g., change draw-a-diagram to multiple-choice with plausible distractors). A second concern was to prevent online cheating to the extent possible. Here we chose to provide an environment that made it convenient for students to be honest, realizing that it is impossible to prevent dishonestly completely in an online format and preferring to avoid disadvantaging honest students for whom more intrusive cheat detection would be an issue (e.g., internet connection quality or anxiety disorders). Our first mitigation strategy was to re-parameterize the questions from our four previous years of exams and use the Canvas feature which will randomly pick a question from a quiz bank for each question on an exam. When giving the exam, this makes it

highly unlikely that any two students will be given the same exam. In addition, we asked students to indicate they were upholding the honor code and required them to turn on their webcam video for a few minutes at the beginning of the exam session, greeting them by name as they did so. These latter techniques were motivated by the research that indicates institutional honor codes and better faculty-student connections reduce cheating. We felt that establishing contact right at the beginning of class reminded students of our connection in normal class periods and reminded them that we valued them doing their own work on the test. [8]

Our lab materials are available under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License: `https://bit.ly/lam-weikle-ccsc21`

4 Discussion

When initially contemplating teaching Computer Systems I online, we were concerned we would lose more students than in a typical in-person semester. However, our experience was that with this format our D/F/W rate was similar and grades on individual assignments were similar if not slightly higher. There are many reasons grades might have been slightly higher including: incremental improvements were made to the schedule and assignments, all exams were open book and open note online, videos could be reviewed when class was missed, withdrawals were allowed at the very end of the semester, and we may have graded more generously in the middle of the pandemic. However, we also received very positive feedback on student evaluations in both the survey comments section and via personal conversations. Anecdotally, we also saw new relationships formed between students, and the faculty felt connected to the majority of the students in class, similar to our prior experience in person.

We were particularly pleased with dedicating the entire class period to labs and active learning, primarily because more students were able to complete the labs during the class period but also because it reduced lecture prep time during the semester. We intend to continue using these videos and add to them for future in-person semesters so labs can make use of the entire class period. We also found that having students submit their labs electronically as PDFs significantly reduced the time spent grading because it is integrated with our LMS. We plan to continue this policy in person as well, observing that high-quality scanning apps are widely available for cell phones. Such a policy also makes it easier to accept late work on an ad-hoc basis.

5 Future Work

While our course was designed thinking about formative assessment and how each aspect of the course would build on other aspects, the transition online has encouraged us to be even more intentional about the formative aspect. We intend to review the Canvas quizzes and add some of the instant-feedback questions from the lectures and possibly more directed feedback for wrong answers. We also intend to reconsider the reading assignments in an attempt to reduce the amount of text required and focus on the most important concepts, in part because there is now a greater amount of time required to watch the videos before class. Finally, we intend to consider the work in Nelson [9] with the goal of making more rigorous arguments for the validity of our formative assessments in future semesters.

6 Conclusion

We were pleasantly surprised by how well the transition to online teaching went in our hybrid course built around formative assessments. Our primary contributions in this effort include video versions of all lectures, Google slide implementations of in-class labs, course procedures that allowed students to choose their breakout room in Zoom, and randomization of question selection for midterm and final exams. Informally, we found that student performance was equivalent or possibly even better than in previous semesters with tangible benefits to the instructors as well, and we plan to continue using (and improving) this course design in the future.

References

[1] Paul Black and Dylan Wiliam. Assessment and classroom learning. *Assessment in Education: Principles, Policy & Practice*, 5(1):7–74, 1998.

[2] Don Blaheta. Reinventing homework as cooperative, formative assessment. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, SIGCSE '14, page 301–306, New York, NY, USA, 2014. Association for Computing Machinery.

[3] Randal E. Bryant and David R. O'Hallaron. *Computer systems: A programmer's perspective*. Pearson, 2019.

[4] Kathleen M. Cauley and James H. McMillan. Formative assessment techniques to support student motivation and achievement. *The Clearing*

House: A Journal of Educational Strategies, Issues and Ideas, 83(1):1–6, 2010.

[5] Marina Duarte. Formative assessment in b-learning: Effectively monitoring students learning. In *Proceedings of the Second International Conference on Technological Ecosystems for Enhancing Multiculturality*, TEEM '14, page 497–501, New York, NY, USA, 2014. Association for Computing Machinery.

[6] K. Dunn and S. W. Mulvenon. A critical review of research on formative assessment: The limited scientific evidence of the impact of formative assessment in education. *Practical Assessment, Research and Evaluation*, 14:1–11, 2009.

[7] Shuchi Grover. *Toward A Framework for Formative Assessment of Conceptual Learning in K-12 Computer Science Classrooms*, page 31–37. Association for Computing Machinery, New York, NY, USA, 2021.

[8] Patricia A. Hutton. Understanding student cheating and what educators can do about it. *College Teaching*, 54(1):171–176, 2006.

[9] Greg L. Nelson, Andrew Hu, Benjamin Xie, and Amy J. Ko. Towards validity for a formative assessment for language-specific program tracing skills. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, Koli Calling '19, New York, NY, USA, 2019. Association for Computing Machinery.

[10] Jill Staake. Google slides 101: Tips and tricks every teacher needs to know. `https://www.weareteachers.com/google-slides/`, Dec 2020.

[11] Dee A. B. Weikle, Michael O. Lam, and Michael S. Kirkpatrick. Automating systems course unit and integration testing: Experience report. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE '19, page 565–570, New York, NY, USA, 2019. Association for Computing Machinery.

Table 1: Labs

Lab Number and Name	Description
01-intro	Introduction to a computer system
02-cmd_line	Linux command line and C compilation
03-c_intro	C memory model and pointers
04-arrays_strings	C arrays and strings
05-structs_io	C structs and I/O using fread and fgets
06-getopt_debug	Command line parameter parsing and debugging
07-binary_info	Binary/ hex representation and bitwise operations
08-integers	Integer encodings (unsigned, 2's compl.) and shifts
09-bin_arith, fp-intro	Binary arithmetic, intro to floating point
10-floating_point	Floating point representations
11-asm_intro	X86-64 assembly basics
12-asm_data	X86-64 data movement and arithmetic
13-asm_ctrlflow	X86-64 control flow
14-asm_proc	X86-64 procedures and runtime stack
15-asm_misc	X86-64 data structures and floating point
16-y86	Y86-64 simplified assembly (from CS:APP)
17-cmb_circuits	Combinational circuits
18-seq_circuits	Sequential circuits
19-arch_pipelining	Architecture and pipelining lab
20-y86_semantics	Y86 semantics lab w/ CPU pipeline stages
21-memory	Memory hierarchy and technologies
22-caching	Caching and set associativity
23-virtual_mem	Virtual memory and address translation
24-processes	Exceptions and processes
25-files	File systems and I/O
26-threads	Threads (preview of next systems course on concurrent computing)

How Colleges Can Better Prepare Technology Students for Internships Students Perception of Internship Programs*

Susan S. Conrad
School of Technology and Innovation
Marymount University
Arlington, VA 22207
sconrad@marymount.edu

Abstract

Internships have become the preeminent gateway to full employment in the technology sector. Students completing one or more internships greatly enhance their odds for getting multiple job offers with attractive salaries when compared to students lacking internship experiences. In a case study analyzing students' experiences after completing their internships, students identified three critical areas that would help them better succeed in their internship experience: Practical Application of Skills; On-going Career Development Skills and On-the-Job Best Practices. Students generally reported feeling prepared for the workforce technologically but identified the need to incorporate soft skills and confidence building strategies into their study curriculum. Students recommended colleges begin promoting the internship process as early as freshman year and provide a class to better prepare them. Recommendations include mandating a 3-credit internship preparation class early in the student's academic studies to help prepare students for the workforce by assisting students apply and succeed in an internship experience.

1 Introduction

The demand for technology educated students has never been greater. With employment growth projected to range between 11 - 15% for the next 8 years, the demand for workers with skills in cloud computing, big data and cybersecurity will sizzle. Preparing students to take on important roles is an essential function for post-secondary learning institutions across the globe with a broad range of academic curriculums and a menu of specialties for students to select. Internships have long been identified as a key factor in helping students grow skills, gain confidence and gain employment [14].

According to a 2020 student survey from the National Association of Colleges and Employers (NACE), 68 percent of students who interned were offered a full-time position upon graduation and the acceptance rate was 81.6 percent. Completing an internship increased job offers by 16%, and salaries for graduates with internships was 9 - 12% higher than compared to students without internship experience [18]. Students who had an internship reported greater confidence in their abilities and readiness for the workforce and a greater understanding of their perspective career choice [11].

However, despite these statistics, only 60% of students get internships while in college (Kapoor & Gardner-McCune, 2019). In addition, a 2018 study by McGraw-Hill revealed that 59% of graduating college students surveyed reported that they did not feel prepared for the workforce [9]. Other research revealed that 43% of graduates are underemployed [6]. This data demonstrating employment and salary gaps between students with internships and those without, supports the necessity for students to gain entry level work experience while attending college [4].

Given the importance of preparing students for success with internships, it is necessary to understand that academic preparation and internship readiness must include both technical and soft skill programs. This paper analyzes the results from a study of students who had participated in an internship program during Fall 2020 and Spring 2021 to learn what academic institutions can do to better help prepare students for their internship experience.

2 Background

A 2020 study revealed that two thirds of employers look for graduates with applicable work experience, and 37% reported that work experience was their number one criterion for hiring [15]. In particular, tech employers want new graduates with experience, not just because of their technical skills, but also because internship experiences help graduates know if they like the job. Thirty-four percent of students change their career direction after interning [6]. This

aligns with reports from 33% of internship employers stating that applicants did not have a satisfactory level of knowledge about their chosen career, resulting in high turnover and increased hiring costs [10].

Internships especially in the tech world have become the norm, with companies relying on recent interns to fill open employment positions. For example, in 2020 Facebook hired 80.2% of recent interns; Google hired 78.3%; IBM hired 81%; and PwC hired 90% [6]. With the growing shortage of tech worker, companies have created attractive internship programs to compete for top talent [13]. These companies offer structured internship programs, attractive salaries, social activities, housing and more. Technical aptitude, although important, was second to compatibility with the corporate culture. Ninety-one percent of manager surveyed reported that cultural fit is as important as skills and experience [18].

The importance of a student internship is not new. The literature is rich with articles spouting the benefits of promoting student internships. The data clearly shows that students who complete an internship before graduation have greater success finding employment and have starting salaries higher than students who do not complete an internship [17]. Many schools even require students to complete an internship prior to graduating [8].

The problem is that even though students have the necessary academic training, they often lack the workforce acumen and soft skills necessary to feel confident and perform their tasks. Complex problem solving, communication and "potential" topped the list of characteristics employers deemed necessary for a good fit with their organizations [1]. Another study conducted by LinkedIn (2020) stated that employers are willing to trade tech skills for soft skills such as creativity, persuasion, collaboration, adaptability, and emotional intelligence. Yet still another study identified "Understand Role/Structure in the Workplace and Have Realistic Work Expectations" as the most important skill for students and employees. This study also found that "listening skills" were rated significantly more important by managers than by students supporting the need for increased focus on communication skills. [3].

In addition to technology and soft skill preparation, lack of confidence is a real problem plaguing interns and new graduates. Lack of confidence can lead to "Imposter Syndrome" (ImP), a belief that the individual isn't qualified and doesn't deserve the job [2]. The impact of ImP on interns and new graduates can significantly impact performance, relationships and ultimately success [5]. In a study of tech workers conducted by Blind, an anonymous workplace social network of more than 4.1 million users, tech workers were asked if they experienced ImP. Approximately10,400 individuals responded with 58% reporting to experience or have experienced ImP. A 2020 study to identify the frequency of ImP in college students revealed that 57% of college students reported feelings

of imposter syndrome. [16]. Bringing ImP awareness to technology students was recommended as a first step in helping students cope with this syndrome [16].

Helping students gain relevant work experience is a focus of numerous university internship programs, but such programs often lack structure, resources or timeliness to be effective [7]. Focusing on technical skills without incorporating soft skills into the course offerings only semi-prepares students for internships [4]. Often, the internship requirements come at the end of a student's academic tenure, and the internship search becomes a last-minute rush for students to find any internship just to meet academic graduation requirements. Finding a happy medium for students to learn the necessary soft and technical skills suitable for a successful internship can be challenging in an already requirement-packed curriculum. Understanding how academic institutions can better prepare students for an internship is the focus of this paper.

3 Methodology

A case study utilizing mixed methods was conducted in a senior-level internship class to ask enrolled students about their internship experience. The students were majoring in one of the following three programs: Information Technology, Cybersecurity or Data Science. A population of 53 students were surveyed in which 74% responded. The data was collected at the end of fall 2020 and spring 2021 academic sessions. The purpose of the research is to understand what difficulties students encountered in their internship and identify gaps in curriculum to address these deficiencies. To collect this data, students were asked to complete an online survey using Google Forms. The Google form consisted of 23 questions of which 12 were quantitative and 11 were qualitative. Completing the survey was voluntary.

3.1 Data Analysis

Since this is a mixed methods study, multiple data analysis techniques were applied depending upon the data type. Quantitative data obtained through the questionnaire was analyzed through descriptive statistics and percentages. Qualitative data was coded using an inductive approach; dissecting the data to reveal themes, concepts, and relationships among the data. The categories were examined to identify connections between the data by applying open coding and axial coding methods.

4 Results

Students overwhelmingly reported satisfaction with their internship experience. The first few questions asked for demographic information and plans after graduation. Sixty-three percent of students reported that they would seek a job. Twenty percent said they were already working full-time and would continue with their current employer. Ten percent planned to attend graduate school; 3% planned to start a business; and the remainder were unsure of their plans. The survey then asked students to identify two or three things that were learned over the course of the internship. Most students reported that they improved their technical skills during their internship. Communication was the second most important skill students reported learning during the internship. One student commented, "This internship experience allowed me to recognize what I am good at and what I need to work on. I learned that communication is essential when working with people." Another student said, "I learned to better talk to people and how to ask question better on what my objectives were." Several students noted that collaborating with team members and asking question as important skills learned through their internship experience. This combined with professionalism such as customer service, meeting management and teamwork were also discussed. Developing remote working skills was also discussed. One student summarized this concept by saying: "I learned how to work with my supervisor remotely. I had to do a lot of tasks on my own since I wasn't in an office setting. This taught me how to be productive without being told." Figure 1 summarizes what skills students reported to improve upon during their internship experience.

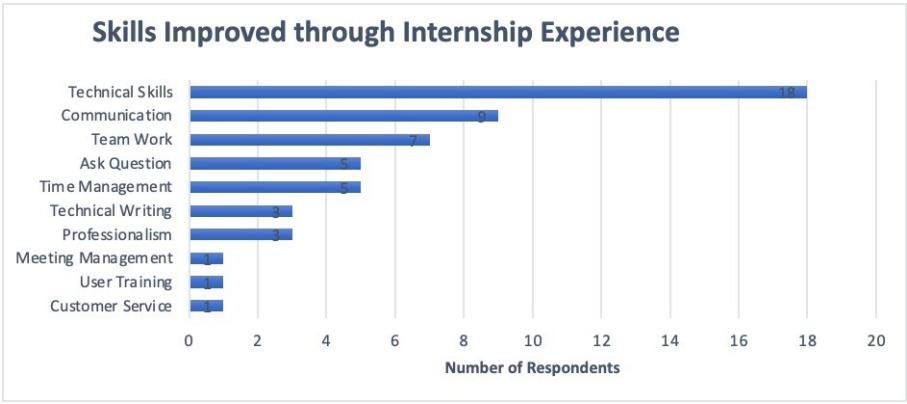


Figure 1: Student identification of skills improved due to the internship experience

Another question asked students to describe the qualities they liked about their supervisor. Helpfulness was the number two response, followed by communicative with prompt answers to questions. Students identified flexibility and concern for the individual student to be desirable managerial traits. One student summed up this sentiment with the following quote, "I admired that he really cared about my well-being and made sure that I was getting the best out of my internship." Students recognized the importance of communication and team building skills. As summarized by one student, my supervisor "Motivates the team. Provides assistance when needed and has great communication skills." Figure 2 summarizes qualities students liked about their supervisor.

Figure 2: Personal qualities students prefer in their supervisors

The next question asked students to describe what qualities about their supervisor that were difficult for them – in terms of doing their job. Responses to this question clearly identified communications as an area for improvement. Since 75% of the internships were virtual due to the pandemic, students had to improvise communications with emails, zoom meetings, slack conversations, and text messaging. These virtual communication methods gave way to miscommunications, delays in responses, misaligned expectations, and frustrations. Many students said they felt very isolated and missed working in an office. Other qualities students did not like in their supervisors include disorganization, forgetfulness, time zone differences and assigning tasks far be-

yond skill levels. One student summed up the internship experience saying: "I felt that it was very difficult to understand what the supervisor wanted at times and that policies and procedures were very loosely defined if at all, which made diagnosing issues very difficult as there was no standard procedure. It was a lot of 'do what I say but I'm not going to tell you how to do it.'" Communication styles and availability were mentioned by 77% of all respondents.

One question asked the students if they believed that their academic institution had prepared them for the workforce. Ninety percent of the respondents reported that their experience at their academic institution had technically prepared them for their internship. However, when asked the question as to what the university could do better to prepare them for the workforce, students identified three overarching gaps in their academic training: Practical Application; On-going Career Development; and On-the-job Best Practices.

Practical Application

- Connect classes with actual jobs: Students reported that they wanted to explicitly connect the concepts learned to real-world scenarios (Student 1, 11, 20, 22). "I think some of the more technical classes should be more hands-on with the learning and flesh out the more important concepts in a way that could be applicable to a real-world scenario." (Student 1).
- Hands-on experiences: Students wanted to test their skills in real test environments and expand the technical offerings (Students 1, 6, 14, 25, 26). "I think having a lab about specific field of study would help especially for networking. For example: My job in the internship was to configure routers, switches, hub etc. I have learned these things in books and classes, however I did not have a real time experience using these kinds of tools, so if the University can have a networking Lab, where students can create their own Local area network, I think that would definitely help." (Student 11).
- Simulate workplace activities: Students wanted opportunity to apply knowledge in simulated activities such as case studies and advanced project management scenarios. (Students 12, 19, 20).
- Exposure to different jobs and leaders in the IT field: Students reported that they wanted more education about the various specialties and job types affiliated with their specific major. (Student 6, 21, 23) "Sending the students outside the school to other organizations in order to let the students see how the employees are working" (Student 18). Ideas like bring a student to work was suggested. Another suggested asking alumni to share their experiences from their job and talk about their day-to-day responsibilities. (Students 10, 18, 21)
- Offer Workforce Readiness Class in Sophomore/Junior year: Several stu-

dents suggested that the content covered in the internship class be offered prior to working at the internship job (Students 6, 12, 17, 21, 30). "I also believe classes like this internship class should be made available at some capacity beginning sophomore year and up so students can get a head start or better direction in joining the workforce. I can honestly say a class like this earlier on in my college career would've bettered my focus and showed me the importance of making connections, looking, and applying for internship, interviews skills and the whole process." (Student 21).

- Inform and help prepare students for certificates: Students realized that earning relevant certificates and badges while in school would be advantageous when looking for a job (Student 4, 23). As noted by one student, "For Information Technology major students, I feel like helping/preparing students get more certificates before graduation would definitely benefit them as well as prepare them for the workforce."
- Focus less on grades and more on learning: Students expressed pressure to study for exams but not for comprehension (Students 3, 5, 24).

On-going Career Development

- Interviewing best practices and interviewing was an area identified by students for increased training. Students identified several key types of interviews which should be included in the curriculum: case interviews, virtual interviews, coding interviews and technology interviews (Students 2, 13, 15, 21, 22, 28).
- Networking with professionals, alumni, and past interns: Students wanted opportunities to learn and practice networking skills (Student 6, 8, 10, 13, 21). "I would say giving students the chance to attend a conference is a great way to create a connection or network that could be useful in our workplace" (Student 6).
- Identify Internship Leads: Students suggested that a current list of employers based upon previous internships be available for students to contact (Students 6, 16).

On-the-Job Best Practices

- Working remote best practices: Students expressed concern about how to best interact with managers and colleagues when working remote. Knowing how much communication was either too much or insufficient was a challenge (Student 7, 27).
- Professionalism: Students suggested a class on professionalism to include ethics and next steps once employed (Student 22, 29)
- Managing Anxiety and Imposter Syndrome: While only one student identified Imposter Syndrome and managing anxiety as an area for additional

training, the research shows that anxiety and imposter syndrome are real problems and especially for new graduates [5, 12, 2]. "I think many students like myself suffer from a bit of imposter's syndrome - the idea that we aren't *really* qualified for a given job - and struggle with anxiety when it comes to joining the workforce. Anything to help ease that what be helpful (Student 10).

5 Discussion and Recommendations

Results from this study clearly show that the student internship experience is very important for successfully developing a strong and ready workforce. The students reported that strong technical skills as well as strong communication skills were the two biggest areas if growth learned through the internship experience. These results suggest that academic curriculum provide robust hands-on application of technology in classroom environments, as well as create multi-disciplinary assignments where students combine technology training with communication deliverables emulating real-world tasks. For example, a coding project could also include a written memo where students create user documentation and a summary of the project. Connecting assignments to real-world job responsibilities provides students the opportunity to practice their skills in a safe space under the guidance of an instructor. Providing incentives and recognition for students who achieve industry certifications can serve as a motivator and help students align learning objectives with real-world outcomes.

Working virtually added an extra layer of complexity to the internship experience. Students reported feelings of isolation and uncertainty as to whether tasks were being completed correctly. Providing students with a framework to communicate status on projects in the classroom can serve as a model which students can use with their employers. Teaching numerous communication tools used in industry within the classroom can give students an arsenal of ways to reach out to managers and teammates.

Preparing and assisting students throughout the job search process was clearly identified as a take-away. Students want help to understand career paths within their discipline and want to begin the job preparation stage in the sophomore/junior year. Students want exposure to successful professionals, especially recent graduates. Tips on networking and exposure to top-level executives was also discussed. Students want to be prepared and practice interviewing with mentors.

The data also provided insight into what characteristics students want from their managers. Students reported to want a manager that is caring, helpful and communicative. These soft skills are an area for inclusion in academic settings. Case studies discussing manager/employee interactions can be a step

towards teaching students how to integrate these soft skills in the work environment. As a new employee, learning corporate culture norms, policies and procedures can be daunting, but by cultivating strong soft skills individuals can more successfully navigate the workforce terrain. In addition, by making students aware of what skills new employees look for in their manager, they can better remember those characteristics when they themselves become managers.

Helping students learn how to deal with stress, anxiety and imposter syndrome was identified as critical. As mentioned earlier in this paper, 58% of technology professionals have admitted to experiencing imposter syndrome at some point in their career. That feeling of self-doubt can be career-limiting if individuals do not develop strategies to cope with these emotional showstoppers. Incorporating guest speakers into classes that openly discuss these issues is the first step to helping students feel confident upon graduating.

5.1 Recommendations

Given the data and implications that have drawn, the following recommendations are made:

1. Begin discussions of internships in the freshman year. This would include information about types of internships (internal, external and coop), potential employers and exposure to technical career paths. Internships should be a mandatory university requirement for graduation.
2. Front load technology and communication classes so students will be able to perform basic technical tasks early in their academic career. Require students to complete writing and oral communication classes to help students learn how to communicate with supervisors and teammates. Requiring these courses early in the student's educational career will prepare them, making them attractive to employers. Other courses although important for a well-rounded education should be taken later in the student's study program.
3. Create a formal 3 credit internship class to be taken within the first two years of the student's academic career. The course would include lectures, speakers, workshops, and a weekly hands-on lab to practice skills learned. The class would have very clear objectives and outcomes examining all facets of workforce readiness. The topics could include networking with professionals, workforce problem-solving, communication with supervisors and teammates, workforce conflict-resolution, professionalism, workforce ethics, interviewing, resume development, remote working, career development, confidence building and emotional well-being.

6 Conclusion

Preparing student for the workforce is multi-faceted, incorporating social skills, confidence, leadership, professionalism, and technology into an ongoing process. Internships allow students to test the waters of different employers, job types, management styles and industries. Internships have become the pathway to jobs upon graduation and provide students the opportunity to develop and perform their best during the internship. The accelerated importance of the internship has moved the bar for universities to begin preparing students for their internship earlier in the student's academic career and focus career-readiness in their sophomore and junior years; not wait until months before graduation. In addition, schools need to provide resources to help students succeed in these internships. Resources include dedicated classes and mentors for guidance. Providing a class early in a student's academic career will help launch them into the workforce feeling prepared.

This research will be expanded to include data from future students in the internship process, focusing on the total person, not just technology training. In addition, the research will examine feedback from employers to identify what students lack in workforce readiness.

References

[1] Address skills gap by identifying skill adjacencies. HRNews (March 2021). Retrieved June 4, 2021 from http://www.proquest.com/career/docview/2497566431/abstract/185A7EF3ACB74289PQ/30.

[2] Imposter syndrome: Why it's so prevalent among it pros. TechGenix. Retrieved June 11, 2021 from https://techgenix.com/imposter-syndrome-it-industry/.

[3] Madeline St. Amour. Report: Which employability skills are students missing? Inside Higher Ed. Retrieved June 4, 2021 from https://www.insidehighered.com/quicktakes/2020/06/10/report-which-employability-skills-are-students-missing.

[4] Laura Ascione. Fewer than half of college students are ready for the workforce. eCampus News. Retrieved June 4, 2021 from https://www.ecampusnews.com/2018/08/03/fewer-than-half-of-college-students-are-ready-for-the-workforce/.

[5] CareerFoundry. 58% of tech employees experience imposter syndrome. here's how to overcome it. Medium. Retrieved June 11, 2021 from https://medium.com/wearefutureworks/58-of-tech-employees-experience-imposter-syndrome-heres-how-to-overcome-it-78172d8a2258.

[6] Chegg. Internships by the numbers. Chegg Internships.

[7] Jean Chu, Patricia Morreale, and Michael Press. Workforce and career readiness for computing and technology students. *J. Comput. Sci. Coll*, 35(8):76–86, 2020.

[8] Susan S. Conrad. Experiential learning: preparing students for the workforce through faculty mentorship and feedback in campus-based it projects. *J. Comput. Sci. Coll*, 36(3):142–150, 2020.

[9] McGraw Hill. 2018 future workforce survey analysis. (2020).

[10] Amanpreet Kapoor. Deconstructing successful and unsuccessful computer science undergraduate interns. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE '19, page 1297, New York, NY, USA, 2019. ACM.

[11] Amanpreet Kapoor and Christina Gardner-McCune. Understanding cs undergraduate students' professional development through the lens of internship experiences. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE '19, pages 852–858, New York, NY, USA, 2019. ACM.

[12] Shreya Malik. Imposter syndrome as experienced by a 24-year-old software engineer. Medium. Retrieved June 11, 2021 from `https://medium.com/@techwithshreya/imposter-syndrome-as-experienced-by-a-24-year-old-software-engineer-951ba798175d`.

[13] Zameena Mejia. 25,000 students voted on the no. 1 internship — and it isn't google or facebook. CNBC. Retrieved June 14, 2021 from `https://www.cnbc.com/2018/08/08/25000-students-voted-one-the-no-1-internship.html`.

[14] U.S. Bureau of Labor Statistics. Computer and information technology occupations: Occupational outlook handbook:? U.S. Bureau of Labor Statistics. Retrieved June 11, 2021 from `https://www.bls.gov/ooh/computer-and-information-technology/home.htm`.

[15] Rachel Pelta. Education vs experience: What do employers want more? FlexJobs. FlexJobs Job Search Tips and Blog. Retrieved June 14, 2021 from `https://www.flexjobs.com/blog/post/education-vs-experience/`.

[16] Adam Rosenstein, Aishma Raghu, and Leo Porter. Identifying the prevalence of the impostor phenomenon among computer science students. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, pages 30–36, New York, NY, USA, 2020. ACM.

[17] Staff Writers. What makes a recent college grad employable? BestColleges. BestColleges.com. Retrieved June 4, 2021 from `https://www.bestcolleges.com/blog/what-makes-a-college-graduate-employable/`.

[18] Arthur Zuckerman. 98 internship statistics: 2020/2021 data, trends & predictions. CompareCamp.com. Retrieved June 11, 2021 from `https://comparecamp.com/internship-statistics/`.

Enhancing Alternative Proof of Work for Cryptocurrencies Using Machine Learning*

Faculty Poster Abstract

Sajedul Talukder[1] *and Abdur R. Shahid*[2]
[1]*Southern Illinois University*
Carbondale, IL 62901
sajedul.talukder@siu.edu
[2]*Robert Morris University*
Moon Township, PA 15108
shahid@rmu.edu

Abstract

Blockchain consensus refers to the process of validating the network's status by guaranteeing that all blocks include the same information and have the correct values across all nodes. A cryptocurrency, often known as "crypto," is digital money that uses the Proof of Work (PoW) method to establish consensus. PoW is used by popular cryptocurrencies like Bitcoin to establish consensus, which is a time-consuming and resource-intensive procedure. Miners expend a lot of energy to perform a brute-force search for the desired hash value using a nonce that isn't helpful otherwise, in order to add a block of transactions to the blockchain. A nonce is a one-time-use value that, when appended to a block and hashed, ensures that the output of the block hash begins with the correct number of zeros. In this paper, we present machine learning-based methods that give a quicker means of establishing consensus and create a template for Proof of Work protocols that ensure similar security guarantees as Bitcoin. To forecast the nonce value from block headers properties, we utilize a mix of multi-layer perceptrons (MLP) and recurrent neural networks (RNN). Our approach minimizes the protocol's time and energy usage by quickly determining the correct nonce value while maintaining security and decentralization.

Analyzing the Optimal Level of Investment in Cybersecurity for Nonprofit Organizations(NPO) by Combining Gordon-Loeb and FAIR Models*

Faculty Poster Abstract

Natalia Ermicioi and Michelle (Xiang) Liu
Marymount University
Arlington, Virginia 22207
{natalia.ermicioi, michelle.liu}@marymount.edu

Abstract

Cybersecurity is rapidly becoming a global priority. Quantifying cybercrime losses and making sound cybersecurity investment decisions are crucial aspects for any organization. In contrast to common belief, nonprofit organizations (NPOs), like for-profit companies, participate in various data collection practices, making them vulnerable to cyber-attacks. This project represents the initial phase of analyzing the applicability of the Gordon-Loeb (GL) Model and FAIR Model to the nonprofit sector for allowing them to establish the optimal investment level in cybersecurity. The GL Model has been widely referenced in the academic and industry literature, and the FAIR Model is a well-known industry model that codifies and monetizes the risk. However, neither of the models has been previously applied to the nonprofit sector. The purpose of this project is to create a hypothetical scenario and example of how to use the insights from the GL Model and FAIR Risk Model to assess the information assets of an NPO and calculate the optimal level of cybersecurity investment. Investigating the costs associated with cybercrime and investment in cybersecurity in the nonprofit sector would broaden the body of theoretical learning in the economics of cybersecurity and significantly contribute to the industry.

Text-based Large-Scale Programming Assignment for Teaching Object-Oriented Analysis and Design*

Faculty Poster Abstract

Jessica Zeitz and Veena Ravishankar
Computer Science
University of Mary Washington
Fredericksburg, VA 22401
{jzeitz, vravisha}@umw.edu

We present a multi-part semester-long text-based project designed for our course within the computer science major, Object-Oriented Analysis and Design. The goal of this project is to provide students experience with analysis, design, and programming of object-oriented concepts and testing. The course is the second in our major sequence that uses the Java programming language aimed at strengthening the students' programming skills. The project is divided into three main parts: two individual and one final group project. The individual assignments give students a foundation for the larger scale group part and allow them to become stronger coders on their own. We found students improved their programming skills between the two individual assignments. All three projects build on the previous assignments teaching students incremental design. We provide a UML diagram for each individual project as scaffolding. This also provides students with an already well designed object-oriented program forcing them to code within the requirements. They use this to design their own functionality for the group project allowing them to practice object-oriented design themselves.

The context of all projects is a virtual tour around our institution, a relatable theme that provides familiarity to them. All students likely have a connection to the institution since they are enrolled. Those who matriculated before the pandemic have a sense of the campus. Those who did not can learn about campus hopefully increasing their connection despite the pandemic. Many students saw the connection to text-based games, but those who did not, were not hindered. Since the project was relatable, we found students added features to the final project that focused on other aspects of the institution. Themes included our institutions focus on diversity and inclusion and our mascot. During the group project, students honed their UMW design skills, programmed creatively, and interpersonal and teamwork skills.

An Applied Machine Learning Approach to Detecting Fraud in Mobile Transactions*

Faculty Poster Abstract

Faleh Alshameri, Alex Mbaziira , Khadijah Makhadmi
Dept. of Information Technology, Data Science, & Cybersecurity
School of Business and Technology
Marymount University
Arlington, Virginia 22207
{falshame, ambaziir, kjm92918}@marymount.edu

Abstract

There are increasing patterns of fraud in mobile due to increasing adoption and use of mobile technology for financial transactions. In this paper we demonstrate that it is possible to use machine learning to detect fraud in mobile transactions. We generate financial detection models using Naive Bayes, Support Vector Machines, kth Nearest Neighbor, and decision trees. The models were able to detect fraud with accuracies ranging from 50% to 96%.

Hybrid Student Service Learning through Creative Community Partnership*

Poster Abstract

Lily Liang[1], Briana Wellman[1], Uzma Amir[1],
Abdeladim Elhamdani[2], Jeffrey Enamorado[1], Jermel Watson[1]
[1]Department of Computer Science and Information Technology
Univ. of the District of Columbia, Washington, DC 20008
{lliang, briana.wellman, uzma.amir}@udc.edu
[2]Rockville Science Center, Rockville, MD 20850

Service learning connects students with their communities and provides meaning and affirmation to their career aspirations. It also helps them to develop their professional identities as well as leadership. However, the pandemic forced service learning to be reinvented. We share our experience of establishing a student service-learning mechanism that integrates both virtual and in-person activities, implemented via a creative partnership between the University of the District of Columbia (UDC) and Rockville Science Center (RSC). With this mechanism, we were able to leverage the resources of both partnering organizations and continue providing service-learning opportunities to our students, despite the constraints caused by the pandemic. Given the resources of each partner as well as the skill sets and personal interests of student volunteers, we created the following two distinctive projects: 1) a virtual workshop series on Artificial Intelligence for K-12 students; 2) a computer refurbish program that repairs, and upgrades used computers to be distributed to families in need. In the first project, RSC provides virtual workshops, hosting and advertising platforms to reach the community. UDC trains student volunteers to become workshop instructors. For the second project, RSC provides donated computers/parts, a venue for computer donation drop-off and pick-up, and serves as a distribution center for refurbished computers. During social distancing, UDC student volunteers refurbished the donated computers at home after picking them up from RSC and returned them for distribution. The volunteers attended RSC workshops for free as a benefit. These workshops provided them extra-curriculum enrichment. We not only successfully transformed our service-learning program to a hybrid model, but also expanded both our team and the scope of our services. This hybrid service learning model, though designed under social distancing, can be utilized post-pandemic and will continue to benefit our students and the community.

Survey of COVID Online Teaching Methodologies and How They Can Be Used in the Traditional Classroom*

Faculty Poster Abstract

Jennifer Polack
Computer Science
University of Mary Washington
Fredericksburg, VA 22401
polack@umw.edu

Abstract

In order to grasp how college students viewed online teaching during COVID and if any methods used by faculty during COVID were valuable when returning to an in-person classroom, this study conducted an online questionnaire survey of 120 undergraduate computer science students at a liberal arts institution whose primary teaching style is in-person. It aimed at evaluating five aspects, including synchronous vs. asynchronous, group work online vs. in-person, organization of course material, perceived knowledge attained and how students felt about what online teaching methodologies would work well in the future when returning to an in-person environment. What was the perception based on upperclassman and lowerclassman? Was any method of online teaching a positive? How did the organization of course material impact learning? What influences impacted students' opinions on online learning?

Teaching Advanced Data Structures and Algorithms Online During a Pandemic*

Faculty Poster Abstract

Olumide Kayode
Department of Computer Science and Information Technologies
Frostburg State University
Frostburg, MD 21532
ofkayode@frostburg.edu

Abstract

The COVID-19 pandemic has affected learning and course modality in an unprecedented way across various educational institutions. It has caused disruption to the method and way of teaching that greatly benefit students' learning as well as how instructors relate with their students. Most computer science courses that involve practical demonstrations in class for illustrating advanced concepts were affected and students may find it challenging to easily understand such concepts without adequate illustration. Remote learning, especially asynchronous online facilitates dissemination of course lectures and instructions to students who may be across different time zones and void of the need to be online at a specific time. Engaging students, encouraging interaction, and ensuring active participation are major challenges associated with asynchronous online courses. Additionally, students' evaluation and assessment will need to be adapted for such course modality. Toward ensuring improvement in the delivery of asynchronous online courses for advanced class, this poster presentation discusses and addresses the aforementioned challenges. Using advanced data structures and algorithms course as a case study, this work highlights the organization, learning activities, assessment techniques and measurable learning outcomes for an online course during the COVID-19 pandemic. It further discusses approaches implemented to address challenges associated with students' participation, communication, and application of learned concepts. Students' positive feedback gives us the assurance that it is an effective method for online learning.

Machine Learning Android-Based Malware Classification*

Faculty Poster Abstract

Micheline Al Harrack
Dept. of Information Technology, Data Science, & Cybersecurity
Marymount University
Arlington, Virginia 22207
micheline.al.harrack@marymount.edu

Abstract

Analyzing Android malware to improve classification, clustering and therefore detection continues to increasingly evolve as our interconnected society continues to grow and Android mobile market expands. In this research, I explore two publicly available malware collection datasets and apply a combination of Machine Learning algorithms for classifying and clustering each of them. Some classifiers are applied concurrently, while others exclusively then clusters are being tested on the two different datasets separately. Both sets have the same 215 attributes classified and clustered. The experiment results are presented for inference for best classifiers and clusters by comparing results of the proposed methods on the two datasets sharing the same 215 attributes albeit different in size of malware sample and benign software.

Applying Local Differential Privacy in Handwriting Recognition-based Systems*

Faculty Poster Abstract

Sajedul Talukder[1] *and Abdur R. Shahid*[2]
[1]*Southern Illinois University*
Carbondale, IL 62901
sajedul.talukder@siu.edu
[2]*Robert Morris University*
Moon Township, PA 15108
shahid@rmu.edu

Abstract

The sharing of handwritten text via applications, widely used in various consumer electronic devices and the Internet of Things (IoT) for handwriting recognition purposes, does not provide adequate privacy assurances. Privacy leakage can happen at any point of the recognition system, which includes 1) data collection, 2) model training, 3) output generation, and 4) model usage by the user. Breaching of a handwriting recognition-enabled application might result in the release of sensitive personal information on its writer, which can have significant ramifications. In this work, we present a method to distort a user's original handwriting before sharing it with a specific application by adding configurable statistical noise to achieve local differential privacy (LDP). We construct a shape-deformed test set by applying the method to the MNIST dataset and implement a visualization tool to illustrate the impact of the proposed approach on recognition to analyze and quantify its impact. Then, using the MNIST dataset, we train, assess, and compare convolutional neural networks (CNN), Naive Bayes, random forest, support vector machine, and decision tree classifiers. Our work shows that in local differential privacy settings, without adjusting the CNN for private pictures, it is possible to obtain approximately 80% accuracy, paving the way for more study in this area.

Evaluation of Privacy-Preserving Logistic Regression and Naive Bayes Classifiers in Healthcare*

Faculty Poster Abstract

Abdur R. Shahid[1] and Sajedul Talukder[2]
[1] Robert Morris University
Moon Township, PA 15108
shahid@rmu.edu
[2]Southern Illinois University
Carbondale, IL 62901
sajedul.talukder@siu.edu

Abstract

Machine Learning (ML) has had a substantial influence on the healthcare system during the last decade or so. Breast cancer and diabetes, two of the most prevalent and deadly diseases, are only two examples of a vast number of healthcare issues that have benefited significantly from machine learning. Early identification of malignant tumors and the reduction of the risk of maltreatment are now a reality thanks to ML. Similarly, several machine learning approaches for early diabetes prediction have been suggested and published in recent years. Although ML-based solutions appear to be rather appealing, the vulnerabilities associated with their design have yet to be completely explored. The ML model responsible for classification is regarded as a very valuable intellectual property in prediction-based healthcare applications that is subject to membership inference attacks, model inversion attacks, and training data leaks via the model's prediction. When this data leakage is combined with information regarding query distribution, a full reconstruction attack may be conducted. In this work, we show how to use differential privacy versions of two widely used and very effective machine learning algorithms, Logistic Regression and Naive Bayes, to classify breast cancer and predict diabetes. The privacy need and model accuracy trade-off are depicted using the prominent Wisconsin Diagnostic Breast Cancer (WDBC) dataset and the Pima Indians Diabetes dataset.

Privacy-Preserving Activity Recognition from Sensor Data*

Faculty Poster Abstract

Abdur R. Shahid[1] and Sajedul Talukder[2]
[1] Robert Morris University
Moon Township, PA 15108
shahid@rmu.edu
[2]Southern Illinois University
Carbondale, IL 62901
sajedul.talukder@siu.edu

Abstract

Wearable sensor-based Human Activity Recognition (HAR) has applications in a variety of fields, including healthcare, remote monitoring, behavior analysis, social networks, sports, and surveillance. The data for HAR is generated by a variety of sensors, including accelerometers, gyroscopes, and GPS. The sensor data characteristics are used to build a machine learning model to identify distinct actions. Recently, there has been a rise in research efforts aimed at developing an effective and robust HAR classifier. Despite this, a practical HAR system is still a long way off because of its inability to safeguard the privacy of human data needed to train the classifier against various machine learning model attacks. In this paper, we present our study on developing privacy-preserving machine learning models for HAR. We train two differentially private machine learning models, Logistic Regression and Naive Bayes, to classify different activities. We use data from the smartphone's accelerometer and gyroscope to create data on six actions (walking, walking upstairs, going downstairs, sitting, standing, and laying). To investigate the trade-off between the amount of privacy and the efficacy of the two classifiers, we train various privacy-preserving models of the two classifiers. We utilize accuracy, recall, and F1 score to compare the two models to conventional and commonly used benchmarking classifiers, Logistic Regression and Naive Bayes.

Prevalence of PII within Public Malware Sandbox Samples and Implications for Privacy and Threat Intelligence Sharing*

Student Paper Abstract

Aaron Weathersby
School of Information Technology and Innovation
Marymount University
Arlington, Virginia 22207
aew56369@marymount.edu

Abstract

The necessity of malware analysis and the democratization of previously complex tools to perform that analysis creates the potential for risk to individuals and organizations. The usage of Online Malware Scanners (OMS) to scan and identify documents that may be potentially malware create an opportunity for the inadvertent sharing of confidential information. This paper explores this subject through the examination of non-malicious PDF files uploaded to the OMS website Hybrid-Analysis. Data is gathered in the form of PDF files where regular expressions extract both human readable text and metadata to identify personally identifiable information (PII) and information that would otherwise be considered confidential. A quantitative analysis is performed attempting to infer applicability to the broader population of digital documents submitted to OMS. A research question is explored as to what is the prevalence of confidential information within a limited data set and what are its implications for threat intelligence sharing and confidentiality of the users who documents are being submitted. Ultimately this paper presents a statistically significant number of documents that contain at least a single indicator of confidentiality.

How Can Romania Leverage Education to Become a Regional Cybersecurity Leader?*

Student Paper Abstract

Daria Pop[1] *and Natalia Ermicioi*[2]
[1]*Georgetown University*
School of Continuing Studies
Washington, DC 20001
drp71@georgetown.edu
[2]*Marymount University*
Arlington, Virginia 22207
natalia.ermicioi@marymount.edu

Abstract

From activists and financially motivated threat actors to state-sponsored actors who conduct cyber espionage and large-scale attacks, the cyber realm now encompasses the technological, economic, and political landscape by leveraging powerful cyber weapons worldwide. This aspect has raised concerns for countries such as the United States and its allies, particularly the North Atlantic Treaty Organization (NATO) in their efforts to combat cyber threats. Romania, as a key NATO member due to its geostrategic position in the Black Sea region, has been working on developing a strong partnership with the U.S., as well as enhancing its cyber capabilities and resources. The purpose of this paper is to analyze Romania's collaboration in cyberspace with the U.S. and the country's efforts, from an educational perspective, to address certain challenges with regards to developing cyber skills needed in the work environment and build a competent cyber workforce to help Romania position itself as a regional NATO leader in cyberspace.

An Observational Assessment of CTI Standards for Blue Teams*

Student Paper Abstract

Jeremy McHugh, Dawn Childs, Jerry Jenkins
College of Business, Innovation, Leadership, and Technology
Marymount University
Arlington, Virginia 22207
{j0m53897, d0c56769, j0j32850}@marymount.edu

Abstract

Traditionally, cyber threat intelligence (CTI) has been considered a luxury and has not been treated as a necessity. This is likely because most organizations either do not have the resources to review threat feeds or do not have the proper infrastructure to apply threat intelligence to their security devices. However, with increasingly complex and sophisticated cyber threats and attacks and the ever-changing tool, tactics, techniques, and motivations of adversaries, cyber defenders must leverage available CTI. Cyber threat intelligence is an absolute requirement for effective blue teams. The volume of CTI data collected from a modern environment can be overwhelming. Further, when a large volume of CTI consisting of varying formats is received, it is extremely difficult to consume quickly. Our review found that comparatively, little research existed regarding this emerging and critical research domain. Organizations like the US-CERT occasionally publish detection signatures; however, those signatures are vendor or tool-specific and have little flexibility for organizations with different security devices. We searched for a universal format for threat feeds to increase the near-immediate capabilities and awareness of blue team cyber defenders. In the absence of such a discovery, we recommend a beneficial option for blue team members to remove barriers preventing them from taking faster action on threat intelligence obtained from multiple sources and suggest several possible future research opportunities.

DReAM: A Deepfake Recognition Assessment Model*

Student Paper Abstract

Justin Ubert, Sheryl Drake and Seth Moyers
Department of Cybersecurity
Marymount University
Arlington, Virginia 22207
{j0u40140, sid55148, ssm51455}@marymount.edu

Abstract

Deepfakes leverage a combination of machine learning and artificial intelligence to produce synthetic content, such as audio or visual files, with the goal to deceive some victim(s). There has been a slow uptick in the use of deepfakes in actual crimes: the voice of a company's CEO was used to trick a subordinate company into sending a wire transfer, and more recently, a mother created deepfakes of her daughter's cheerleader teammates in order to get them kicked off the team. The potential for serious harm and lasting impact from abuse of this technology is rising. Given the burgeoning deepfake technologies, use of deepfakes attacking political figures, and a recent Private Industry Notification by the FBI, we conduct a review of current deepfakes using the Diamond mode, showcasing how this model can be used to identify such emerging novel tactics. We then use the review findings to populate a template of a MITRE ATT&CK technique page.

Advantages of Python Programming Language in the World of Big Data*

Poster Poster Abstract

Meharban Arora
Marymount University
Arlington, Virginia 22207
msa27764@marymount.edu

Abstract

This paper introduces the readers to industry and society's adoption of technology, where the involvement of computer programs, coded with Python, have become a necessity. This paper elaborates on the increased usage of data and the rise of the "Big Data" industry. Upon that, there are details about Data science and Machine learning as a byproduct of the Big Data industry. Furthermore, the paper shifts the focus from "Big picture" data towards Python programming language. It specifies why this coding language is important and how it simplifies the data analysis process through the use of open-source Python libraries. The advantages of Python compared to the other similar languages such as C, and MATLAB are also included. Lastly, this project will also review several Python packages that can be downloaded from open-source online resources. There are explanations of the ways the Python packages can assist a Python programmer, novice or expert, analyze data functions in numerous different professional fields. Specifically, this paper will go over the Matplotlib package and one specific dataset example that can be used with it. Matplotlib example is included to show what functions a programmer can access when using Python based libraries. It showcases the ease of use of Python when visualizing data in a dataset.

Using Multiple Forensic Tools to Analyze the Data Acquisitions of Previously Used iOS & Android Devices*

Student Poster Abstract

Katrina Khanta
Marymount University
Arlington, Virginia 22207
k0k37140@marymount.edu

Abstract

Nowadays, most people carry their smartphone with them wherever they go, and it has become one of the primary necessities to not leave your home without. Because of the ever-growing rise in smartphone usage, we store a majority of our personal information into these handheld devices as they have evolved into an extension of ourselves. Additionally, it is critical to acknowledge how multiple mobile applications are connected to cloud storage and how many Internet of Things (IoT) devices require pairing to smartphones via Bluetooth function or access special services. The purpose of this study is to emphasize the significance of data remanence pertaining to mobile devices and how forensic data acquisitions from mobile phones prove as high value evidence in legal cases. We conducted an experimental methodology using various data acquisition tools, such as Magnet AXIOM, Oxygen Forensic Detective, Belkasoft X, and MSAB XRY to extract data from previously used smartphone devices purchased from an eCommerce website The study provides a comparison analysis between the data acquisition tools and the following smartphones: Apple iPhone 8, Samsung S9+, and Google Pixel 3, to determine which tools are more effective at extracting a specific range of deleted datasets.

US Policy on the Use of Force in Cyberspace*

Student Poster Abstract

Roncs Etame-Ese
Marymount University
Arlington, Virginia 22207
r0e09392@marymount.edu

Abstract

The United States' ability to defend and protect itself in cyberspace has evolved at an incredible pace. This evolution has been accomplished largely due to various frameworks such as the US National Cybersecurity Strategy (NCS), Joint Publication 3-12 and Presidential Policy Directive 21. The US government has made several moves over the last few years to secure its critical infrastructure against cyber threats that could disrupt or destroy lives. The goal of this strategy is to promote international cooperation against disruptive cyber-operations in a digital ecosystem built on certainty, transparency, and the law. An analysis of the US policy on the use of force in cyberspace reveals numerous challenges faced by policymakers and IT professionals, such as the legality of conducting offensive and defensive cyber-operations under current international law.

Cybersecurity Education for the Government Workforce: Lessons from Counterintelligence*

Student Poster Abstract

Vincent Pisani
Marymount University
Arlington, Virginia 22207
vfp15920@marymount.edu

Abstract

Cybersecurity has become an increasingly pressing issue for the United States Government (USG). Numerous approaches have been developed to meet this growing challenge in recent years. However, many of these solutions have focused on technical capabilities and incident response at the expense of human-centric solutions such as workforce education. While these developments are important, they have led to a defensive cybersecurity posture that reacts to human-centric vulnerabilities rather than preempts them. Examples of human-centric cybersecurity vulnerabilities include policy violations, loss of equipment, social engineering, and unauthorized system access. These four categories alone made up 38% of information security incidents inside of the USG in 2014 according to the US Government Accountability Office (GAO). This has been further compounded by a steady rise in information security incidents across the USG over the course of the 2010s. Any future whole-of-government approach to cybersecurity education will have to be informed by existing bodies of knowledge to allow for rapid implementation across the workforce. One potential area of interest to this discussion is counterintelligence due to its heavy reliance on human-centric vulnerabilities. Some USG cybersecurity practices, such as threat assessments and security clearances, originated in this discipline and have been deployed effectively.

An Examination of Consumer IoT Security & Privacy*

Student Poster Abstract

Darrell Andrews
Marymount University
Arlington, Virginia 22207
dandrews@marymount.edu

The widespread adoption of IoT-connected devices will ultimately demand that the average consumer participates in this world-connecting landscape. Initiatives must be enacted with respect to the security and privacy of the consuming public for the protection of our individual PII (Personally Identifiable Identification). Whether voluntarily or due to the overwhelming preponderance of IoT-ready devices that are continually being introduced into the marketplace, safeguards must be incorporated. As these technological capabilities arrive upon the consuming public, with lessening non-IoT related options available, such introductions could or perhaps will all but eliminate any freedom of voluntary participatory IoT use. If a home-based device or convenience is capable of being connected to the Internet, it will in all likelihood eventually be connected to the Internet. As such, what must the average consumer, IoT manufacturer(s), local, state, and federal governments do to at least minimally guarantee or ensure that consumer safeguards, mitigations and protections are firmly in place? In order to explore appropriate levels of assurances that our security, privacy rights (as commonly and generally understood in US, EU, and associated world-wide frameworks.), safety and protections against potential violations of our personal and interpersonal space survives, steps must be taken to safeguard our digital existences. As potential dangers will appear in multiple forms, but will not necessarily be limited to familiar forms and now known instances of cyber intrusions, the introduction of malware, ransomware, potential identity theft or obfuscation, and/or cyber-attacks should be acknowledged and expected to evolve. As the technology evolves, we should expect that the attacks and dangers will also evolve, as they have already begun to do so. If we are to be vigilant, efforts must be made to develop, enact and initiate effective, appropriate safeguards, protections, remedies or mitigations to hopefully guard against these inevitable IoT intrusions and violations.

An Analysis of the Rise of Data Breaches in the Healthcare Organizations During Covid-19*

Student Poster Abstract

Laura Vera
Marymount University
Arlington, Virginia 22207
lcv54996@marymount.edu

Abstract

Internet of Medical things, health information technology, telehealth, and cloud services have led the healthcare industry to a digital transformation. Digital health technologies have offered providers real opportunities to improve medical outcomes and efficiency while giving patients better access to treatments and more control over their health. However, the healthcare sector has become a target of security breaches. Although there are many types of incidents, the research aims to analyze the healthcare data breaches reported to the U.S. Department of Health and Human Services' Office for Civil Rights and what type of attacks are more prevalent in medium and large organizations. The study found that hacking/IT incidents are the most pervasive forms of attack behind healthcare data breaches, followed by unauthorized internal disclosures, being ransomware attacks the main cause of the breaches. The frequency of data breaches, number of records exposed, number of individuals affected, affected cover entities and financial losses due to breaches are increasing rapidly. Healthcare data is highly valuable. Healthcare data has become a major target of bad actors. The present study employs a time series analysis that aims to understand the many factors that leave healthcare organizations vulnerable.

Tool to Teach and Practice the GitHub Workflow*

Student Poster Abstract

Jacob Smith
Moravian University
Bethlehem, PA 18018
smithj09@moravian.edu

Abstract

The GitHub Workflow is an important but complicated process for students to learn. One area of common struggle is in the resolution of conflicts, and this skill typically requires two people to practice. The goal of this project is to create a hands-on learning environment where students can practice conflict resolution with computer automation as the second "person." We used the O'Reilly Katacoda platform, a service which allows you to create step-by-step tutorials and provides the student with a set of directions to follow in their own virtual machine. Throughout the scenario, we automate the role of the second person who creates the conflict on GitHub. To implement this system we had to solve a variety of problems such as security vulnerabilities and the long term management of auto-generated repos. We used foreground and background scripts to make calls to an API we created which creates the repo, creates the conflict, and deletes the repos used after a 24 hour time period.