The Journal of Computing Sciences in Colleges

Papers of the 27th Annual CCSC Midwestern Conference

September 25th-26th, 2020
Ivy Tech Community College
Fort Wayne, IN

Baochuan Lu, Editor
Southwest Baptist University

Saleh Alnaeli, Regional Editor
University of Wisconsin-Stout

Volume 36, Number 4 **October 2020**

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

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, School of Computing and Engineering, 5110 Rockhill Road, 546 Flarsheim Hall, University of Missouri - Kansas City, Kansas City, MO 64110.

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 "Crystal" Peng, Associate Editor, (816) 584-6884, crystal.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 2020 CCSC Midwestern Conference

Welcome to the 27th annual CCSC Midwestern Conference held September 25-26, 2020. While we had hoped to meet in-person, like many events, Covid-19 has altered our plans. We are excited to meet virtually!

We have an excellent program of speakers, papers, tutorials, panels, workshops, and vendor sessions. All paper submissions go through a double-blind review process. We had a strong set of paper submissions this year and accepted 10 of 13 for a 76.9% acceptance rate.

We are thrilled to have Dr. Gene Spafford from Purdue University as our keynote speaker. Dr. Spafford currently serves as Professor of Computer Science and Executive Director Emeritus at Purdue University CERIAS. Dr. Spafford is a leading expert in the field of cybersecurity and has earned numerous awards and honors. The title of his address is "It may be worse than you think".

Our banquet speaker is Dr. Peter Staritz, Assistant Professor of Physics Engineering at Taylor University. He worked for General Dynamics Robotic Systems and served as the acting Director of the Intelligent Robotics lab at Lockheed Martin Advanced Technology Labs. Dr. Staritz will be speaking on "Robotics for the warfighter".

Bill Siever is leading a pre-conference workshop titled "Micro:bit Magic: Engaging IOT Embedded for CS1/2 And K-12". We are again hosting a panel titled "What Students Need to Know About Industry," which has become a favorite for students and faculty alike. The Student Showcase and Programing Contest provide additional opportunities for student participation.

Google, zyBooks, Turing's Craft, and NSF are hosting sessions. We thank them for their support as CCSC National Partners. I encourage you to attend their sessions or talk with their representatives.

I would like to recognize and give special thanks to our Conference Committee and our Paper Reviewers. Without many volunteer efforts, this conference would not be possible. CCSC continues to provide a great venue to share ideas and learn new techniques. Thank you for being a part of CCSC Midwest!

Jeff Lehman
Huntington University
Conference Chair

2020 CCSC Midwestern Conference Steering Committee

Mary Jo Geise, Registrar (2019) ..... Univ. of Findlay, Findlay, OH
Saleh M. Alnaeli, Editor (2021) .. Univ. of Wisconsin-Stout, Menomonie, WI
Zaid Altahat, Webmaster (2020) Univ. of Wisconsin - Parkside, Kenosha, WI
Scott Anderson, Treasurer (2020) . Univ. of Southern Indiana, Evansville, IN
Sean Joyce, At-Large (2019) ..... Heidelberg Univ., Tiffin, OH
Kris Roberts, At-Large (2021) Ivy Tech Community College, Fort Wayne, IN
Cathy Bareiss, Regional Representative (2020) ..... Olivet Nazarene Univ., Bourbonnais, IL
Jeff Lehman, Conference Chair and Authors Co-Chair .... Huntington Univ., Huntington, IN
Deborah Hwang, Past Conference Chair .. Univ. of Evansville, Evansville, IN

Regional Board — 2020 CCSC Midwestern Region

Jeff Lehman, Conference Chair and Authors Co-Chair .... Huntington Univ., Huntington, IN
Grace Mirsky, Vice-Chair ..... Benedictine Univ., IL
Kris Roberts, Site Chair ..... Ivy Tech Community College, IL
Saleh Alnaeli, Authors Co-Chair . Univ. of Wisconsin-Stout, Menomonie, WI
Cyrus Grant, Nifty Tools and Assignments Dominican Univ., River Forest, IL
Robert Beasley, Papers ..... Franklin College, Franklin, IN
Deborah Hwang, Past Conference Chair .. Univ. of Evansville, Evansville, IN
Jonathan Geisler, Programming Contest Co-Chair . Taylor Univ., Upland, IN
Paul Talaga, Programming Contest Co-Chair ..... Univ. of Indianapolis, Indianapolis, IN
Md Haque, Programming Contest Co-Chair ..... Univ. of Indianapolis, Indianapolis, IN
David Largent, Publicity ..... Ball State Univ., Muncie, IN
Deborah Hwang, Co-Registrar ..... Univ. of Evansville, Evansville, IN
Stephen Brandle, Speakers Chair ..... Taylor Univ., Upland, IN
Scott Anderson, Treasurer and Speaker Co-Chair Univ. of Southern Indiana, Evansville, IN
Paul Gestwicki, Student Showcase ..... Ball State Univ., Muncie, IN
Shahsa Wu, Student Showcase Co-Chair ..Spring Arbor Univ., Spring Arbor, MI
Donna Ogle, Student Showcase Co-Chair ..... Rockford Univ., Rockford, IL

Kris Roberts, Two-year College Liaison Chair . Ivy Tech Community College, Fort Wayne, IN
Takako Soma, Vendors ..... Illinois College, Jacksonville, IL
Zaid Altahat, Webmaster ..... Univ. of Wisconsin - Parkside, Kenosha, WI
Jeff Lehman, Work-in-progress Chair ..... Univ. of Dayton, OH

Reviewers — 2020 CCSC Midwestern Conference

Alice Armstrong ..... Shippensburg Univ., Shippensburg, PA
Amber Settle ..... DePaul Univ., Chicago, IL
Baoqiang Yan ..... Missouri Western State Univ., Saint Joseph, MO
Brian Howard ..... DePauw Univ., Greencastle, IN
Bruce Maxim ..... Univ. of Michigan-Dearborn, Dearborn, MI
David Bunde ..... Knox College, Galesburg, IL
David Largent ..... Ball State Univ., Muncie, IN
Deborah Hwang ..... Univ. of Evansville, Evansville, IN
Eugene Wallingford ..... Univ. of Northern Iowa, Cedar Falls, IA
Henry Walker ..... Grinnell College, Grinnell, IA
James FitzSimmons ..... Wilmington College, Wilmington, OH
Jeffrey Lehman ..... Huntington Univ., Huntington, IN
Lawrence D'Antonio ..... Ramapo College, Mahwah, NJ
Mark Jones ... Univ. of Texas Health Science Cntr at Houston, Houston, TX
Mickey Hendon ..... Bloomsburg Univ., Bloomsburg, PA
Nicholas Rosasco ..... Valparaiso Univ., Valparaiso, IN
Paul Gestwicki ..... Ball State Univ., Muncie, IN
Stefan Brandle ..... Taylor Univ., Upland, IN
Takako Soma ..... Illinois College, Jacksonville, IL
Victor Norman ..... Calvin Univ., Grand Rapids, MI
Yingbing Yu ..... Austin Peay State Univ., Clarksville, TN
Zachary Kurmas ..... Grand Valley State Univ., Allendale, MI

Cybersecurity
"It May Be Worse Than You Think"*

Keynote Speech

Eugene H. Spafford
Computer Science, Purdue University
spaf@cerias.purdue.edu

We have been working in computer security for over 50 years — almost as long as we've been using shared computers. Despite all that experience — including several hard-won lessons — we continue to have major cybersecurity problems. You may not see how many there are, both because they are underreported and because they aren't as newsworthy as they once were. However, we are under constant attack — and we are often losing. In this talk, I'll explain some of the problems and decisions that keep us in an often-losing battle. We will also review a few things that could help improve the situation, if only they were adopted.

The speaker has been working in security for 40 years, in academia, industry, and government. That experience informs his views and this talk.

Speaker Bio

Eugene H. Spafford is one of the senior, most recognized leaders in the field of computing. His research and development work, including work with his students, underlies cyber security mechanisms in use on millions of systems in use today, including work in firewalls, intrusion detection, vulnerability scanners, integrity monitoring, forensics, and security architectures. Professor Spafford has been honored with every significant award in cyber security, including induction into the Cyber Security Hall of Fame; every major award at Purdue University for teaching; and many major awards for distinguished service to the computing community, including the CRA Distinguished Service Award. He is one of only two people to receive all three of the National Computer Security Award, be inducted into the Cyber Security Hall of Fame, and receive the Hal Tipton Award. He is also the only person ever to be named as a Fellow of the combination of the (ISC)2, ISSA (as a Distinguished Fellow), ACM, IEEE. American Association for the Advancement of Science (AAAS), and American Academy of Arts and Sciences (AAAS).

Robotics for the Warfighter*

Keynote Speech

Peter Staritz
Physics & Engineering
Taylor University, Upland, IN 46989
peter_staritz@taylor.edu

Dr. Startiz will talk about some of the programs he worked on in the past, what makes robotics hard, and why the advances in AI are going to rapidly knock down those problems.

Speaker Bio

Dr. Staritz has a BE in Mechanical Engineering from SUNY Stony Brook and a Ph.D. in Robotics from Carnegie Mellon University. He has been working in industry for over 15 years. At General Dynamics Robotic Systems, Dr. Staritz worked on a variety of technologies to support the individual warfighter. His early efforts looked into using drones to get Soldiers and Marines the information they needed in combat. He also led a team of engineers to develop a next generation LIDAR capable of building 3D maps for robotic systems.

At Lockheed Martin Advanced Technology Labs, Dr. Staritz was the Acting Director of the Intelligent Robotics lab. In that role he led LM's robotics group in the capture and execution of multiple robotics programs including the DARPA Collaborative Operations in Denied Environments (CODE) program for which he was the Principal Investigator.

In 2015 Dr. Staritz and his family served as missionaries in South Asia, serving a rural community there. Dr. Staritz was responsible for managing the construction of a large hospital. He organized the work, performed quality management and designed hospital systems. In addition, Dr. Staritz managed the emergency construction of two new wards on the old hospital to support a developing humanitarian crisis. Dr. Staritz is currently as assistant professor of Physics & Engineering at Taylor University.

Open Career Fairs for Online Computing Students in Ohio*

Kevin Buell
Arizona State University
kevin.buell@asu.edu

Abstract

Although online bachelor's degree programs in computing have become increasingly popular, the path from an online degree to a career in computing is not always clear. Online students benefit only partially from their institution's career services, particularly if students reside far from campus. For such remote students, events like career fairs and employer visits are generally not accessible. However, there are institutions that open their career fairs to the public and thereby serve local students in online degree programs from any institution. These open career fairs help fill the gap for remote online students in their transition to computing careers.

Using Ohio universities as a sample data set, we explore some characteristics of institutions that open their career fairs. We look at three specific traits—funding (public or private), location (urban or rural), and delivery (in-person or online). Contrary to our hypothesis that the most likely to open their career fairs would be rural public universities as well as universities offering at least one online program, we find evidence supporting different conclusions. Rural universities of all types are more likely to open their career fairs but also in-person institutions. Urban institutions as well as those offering online degree options are far less likely to open their career fairs. We discuss a few possible reasons for these findings as well as some implications for the online student experience of transitioning to a career in computing.

1 Background

Enrollment in online higher education programs has grown significantly in recent years [1, 4]. Unlike on-campus students who can take advantage of career services offered by most universities, online students may have access to fewer benefits, especially those who live far from their institution's physical campus. The path from an online bachelor's degree to a career in computing may not be as clear for these students.

Students living near their institution's physical campus interact directly with employers during recruiting events such as career fairs and information sessions. Career fairs are a particularly significant opportunity for students to meet with several employers in one place at one location. These events provide "one stop shopping" for students to secure employment after graduation or obtain an internship which may eventually lead to full-time, permanent employment. Career fairs also provide students an opportunity to improve their interviewing skills by discussing open positions with employers in person, without the need to be invited for an interview.

Of course, remote online students have some means of securing employment after graduation. They may apply directly to open positions, attend career fairs for experienced professionals, or pursue a variety of other networking and tech-centered community opportunities. However, since university career fairs are intended specifically for students (employers are specifically looking to recruit students at these events), they represent an important part of the career search experience that is often inaccessible for remote online students.

Fortunately, some institutions open their career fairs to the public. This allows remote online students of other institutions the opportunity to benefit from these events as long as they are reasonably close and accessible. Two important questions about these open career fairs are 1) why do some universities open their career fairs? and 2) which remote online students are able to benefit from them?

2 Related Work

Several works discuss broadly the different support services offered to online students [8, 10]. Career services, including career fairs, constitute one important component of these services. Work by Dey et al discusses how career services in education have evolved over several decades and identifies the current trend of "Connected Communities" as emphasizing experiential learning and enhanced collaboration with potential employers [5].

Characteristics of online learners are explored in several works [3, 7]. These confirm several anecdotal and intuitive assumptions. For instance, online learn-

ers tend to be older and are heavily involved with employment outside of schoolwork. These characteristics provide some advantages to online students for starting careers in computing (e.g. previous work experience) but at the same time can make balancing the demands of school, employment, and job search very challenging.

Work by Goodman et al explores expanding access to education and specifically looks at a well-known online program in computing [6]. They confirm that instead of reducing the in-person student population, online programs actually tend to increase the overall student population by allowing access to many who would otherwise not enroll. This confirms the urgency of studying the pipeline from online learning to employment, as it represents new opportunities for those who would not have these opportunities without online programs.

3 Hypothesis

Our hypothesis for this work is that certain characteristics of universities would incentivize them to open their career fairs. We originally thought that public institutions would be more inclined to open their career fairs as a service to the community, and especially rural public institutions since they often serve as a key conduit to career improvement where quality employment prospects are fewer than in urban areas. We also thought that institutions which themselves offer online degree programs would feel the need to open their career fairs as "good citizens" in the community of online programs. The idea here is that as more universities reciprocate by opening their career fairs, more of an institution's remote online students would benefit.

On the other hand, we thought that private universities would both lack the incentives of public accountability and indeed have a disincentive to opening their career fairs in order to justify the unique (and often costlier) services they provide. To test our hypotheses, we looked at universities in Ohio as a sample data set. Ohio has a relatively large number of universities with a healthy mix of public, private, urban, rural, in-person, and online programs.

4 Methodology

Of the over 80 institutions granting bachelor's degrees in Ohio, we considered the 43 that offered a computing related bachelor's degree (e.g. computer science, information systems, information technology, etc.) and either held or scheduled at the time of our data collection, a local career fair (e.g. job fair, career expo, etc.) in either 2019 or 2020. We categorized these institutions as either public or private, urban or rural, and in-person or online.

Table 1: Career Fairs Overall

	Open	Closed	Total
All	9 (21%)	34 (79%)	43

Table 2: Career Fairs by Funding

	Open	Closed	Total
Public	2 (12%)	15 (88%)	17
Private	7 (27%)	19 (73%)	26

Public and private are well-understood and refer to the institution's funding model. Urban or rural can be more subjective but we defined urban to be within the metro area of Cleveland, Columbus, Cincinnati, Dayton, Toledo, Akron, or Youngstown. All other institutions were considered rural, including locations which are in very far suburbs of larger cities and farther suburbs of smaller cities. Institutions were considered online if they offer at least one program (in any discipline) which can be completed entirely online. All other institutions were considered in-person. The percentage of online institutions in our data corresponds closely to the figures reported elsewhere [1].

We define an open career fair as one that permits members of the public to participate as job seekers. These are generally advertised online as "open to the public", "open to the community", or "everyone is welcome" whereas closed career fairs are generally advertised as "open to students and alumni". Note that we considered branch campuses as separate institutions as long as they separately met the criteria for inclusion (offer a computing degree at the branch campus and held their own local career fair).

A few interesting career fair arrangements are worth mentioning. First, a group of over 30 private universities (OFIC, or the Ohio Foundation of Independent Colleges) collaborate to provide a single career event called CareerFest held annually in Central Ohio [9]. This event is not local to most of the member institutions and is open only to students and graduates of member institutions (not to the public). If an institution only offers their students access to CareerFest but no other career fair, we did not include the institution in our list.

Note that this did not exclude any of the four OFIC institutions local to Columbus where the event is held. In fact, three of these (Capital, Otterbein, Ohio Dominican) collaborate to hold their own local career fair and we do count these in our list of institutions under consideration. Another group of three institutions (Kent State Stark, University of Mount Union, and Walsh Uni-

Table 3: Career Fairs by Location

	Open	Closed	Total
Urban	0 (0%)	17 (100%)	17
Rural	9 (35%)	17 (65%)	26

Table 4: Career Fairs by Delivery

	Open	Closed	Total
In-Person	4 (40%)	6 (60%)	10
Online	5 (15%)	28 (65%)	33

versity) collaborate in the Greater Canton Collegiate Job and Internship Fair, and we likewise include these in our list of institutions under consideration.

Some universities which do not currently offer a computing degree do open their career fairs to the public. Examples of these institutions include Ohio State Newark and Ohio State Lima. Although these are not in our list of institutions under consideration, it is interesting to note that they are outside of urban areas. Those not specifically interested in computing programs may wish to extend our work to include institutions not offering computing programs and they may find similar results to ours.

Our data set is admittedly small; however, even within this small set we can see a few interesting patterns emerge. We will highlight a few particularly interesting findings. In a future, more expansive work we would like to confirm these findings and add to our understanding of this subject. As expected, we did find that rural institutions were far more supportive of open career fairs than urban institutions. In fact, we found no urban institutions with open career fairs in our sample. This rural/urban divide in offering open career fairs was not surprising, but the magnitude of the difference was surprising.

Contrary to our hypothesis, funding model had a relatively small impact on whether a career fair was open or closed with private universities being slightly more open than public universities, the opposite of what we projected.

Also contrary to our hypothesis, we found that in-person institutions were much more likely to provide an open career fair than institutions offering online options. Had we found that no significant difference existed, we may have been able to dismiss it as uninteresting and possibly due to sample size issues; however, give these data we do feel compelled to look for possible explanations.

5 Conclusions

From the data we collected and analyzed, a picture of an institution incentivized to open its career fair to the public emerges. Currently rural and (surprisingly) in-person institutions are mostly likely to open career fairs. They often do so in conjunction with other educational and non-educational institutions in the area.

Marietta College is a good example of this phenomenon. A private, in-person institution located in southeast Ohio's Appalachian region just across the river from West Virginia, Marietta partners with several other institutions in the community to provide its career fair. Mid Ohio Valley Career Connect is hosted at Marietta and is also sponsored by Washington State Community College, Washington County Jobs & Family Services, and Building Bridges to Careers. The event is specifically organized to provide "one large and inclusive career fair" and to "connect students and adults of all ages to educational and workplace options" [2].

Rural universities are often unique in the community and likely find little reason to compete with other local institutions. Instead, they are incentivized to cooperate in order to attract a larger set of employers to their career events. The institutions are often community serving and may not be as interested in or likely to provide online programs. Indeed, it would not be surprising to see a growing number of rural institutions opening their career fairs to the public in the future.

On the other hand, some rural universities who compete for students at the national and international levels may be less incentivized to open their career fairs as they may feel it could dilute the quality of the candidates presented to employers and could be seen by students as a loss of unique benefits. Likewise, urban universities often compete with many other kinds of institutions in attracting employers to their career events, and they can likely afford to tightly control the experience presented to employers by restricting their career fairs to only students and alumni. Whether these reasons to close career fairs are valid is not clear, but it could be difficult to provide evidence to these institutions that would incentivize them to open their career fairs. Nonetheless, we have found some large, urban universities in other states that do open their career fairs either partially or fully to the public including Arizona State University and North Carolina State University.

For remote online students, the data on open career fairs are generally positive. Those in urban areas often have many alternatives for career events and networking opportunities. Although university career fairs are ideal for them, they can likely overcome the fact that most urban universities close their career fairs. For remote online students in rural areas, their alternatives are often more limited. However, institutions in their area are more likely to open their career fairs which is a much-needed opportunity for them.

Still, several gaps remain. In rural areas with no nearby institutions or without institutions providing a computing degree (and of course when a nearby institution elects not to open its career fair), a remote online student's options are more limited. In Ohio, there is reasonably good coverage of rural areas for open career fairs. However, northwestern Ohio is currently underserved and students in this area will find it more challenging to access career services. Although Ohio State University's branch campus in northwest Ohio (OSU Lima) does open its career fair, the campus does not provide a computing bachelor's degree and therefore it is less likely to attract employers looking for computing students.

What is entirely unclear is when or whether institutions offering online degrees will feel compelled to open their career fairs as a kind of "good citizen" in the community of online educational providers. Helping them see the link between opening their career fairs and the benefits their remote online students will receive as others reciprocate will likely be challenging.

References

[1] I Elaine Allen and Jeff Seaman. *Learning on demand: Online education in the United States, 2009*. Sloan Consortium, Newburyport, MA, 2010.

[2] Marietta College. Career connect. `https://www.marietta.edu/career-connect` Accessed Feb 16, 2020.

[3] Jozenia Torres Colorado and Jane Eberle. Student demographics and success in online learning environments. *Emporia State Research Studies*, 46(1):4–10, 2010.

[4] David J Deming, Claudia Goldin, Lawrence F Katz, and Noam Yuchtman. Can online learning bend the higher education cost curve? *American Economic Review*, 105(5):496–501, 2015.

[5] Farouk Dey and Christine Y Cruzvergara. Evolution of career services in higher education. *New directions for student services*, 2014(148):5–18, 2014.

[6] Joshua Goodman, Julia Melkers, and Amanda Pallais. Can online delivery increase access to education? *Journal of Labor Economics*, 37(1):1–34, 2019.

[7] Genevieve M Johnson. On-campus and fully-online university students: Comparing demographics, digital technology use and learning characteristics. *Journal of University Teaching & Learning Practice*, 12(1):4, 2015.

[8] Ruth Newberry. Building a foundation for success through student services for online learners. *Online Learning Journal*, 17(4), 2013.

[9] The Ohio Foundation of Independent Colleges. Members. `https://www.ofic.org/members` Accessed Feb 16, 2020.

[10] Barbara L Stewart, Carole E Goodson, Susan L Miertschin, Marcella L Norwood, and Shirley Ezell. Online student support services: A case based on quality frameworks. *Journal of Online Learning and Teaching*, 9(2):290, 2013.

Promoting a Sense of Learning Community in Asynchronous Online Courses*

Margaret Polk
Carroll University
Waukesha, WI 53186
mpolk@carrollu.edu

Abstract

Online degree programs present unique challenges with regards to student connectedness and sense of learning community. Yet, both are considered of critical import in the successful student retention and viability of a program. This study sought to address these challenges in a recent launch of an online degree program in software development. Given the literature on this matter, an essential component built into each course in the program, is the presence of weekly, mandatory, asynchronous discussion forums, which are meant to foster a student sense of learning community. After several course offerings in this new program, an assessment was conducted of the student posts in these discussion forums. Unfortunately, this assessment revealed a poor degree of student to student interaction. The need for a strategy to remedy this lack of student interaction became apparent. Any plan, however, was required to respect the asynchronous nature of the discussion forums. The strategy to remedy the lack of student interaction was introduced in two pilot courses and consisted of a group assignment, tactically introduced at a certain point in each course, in order to be of sufficient scope and magnitude. The introduction of the team assignment resulted in substantial improvement in student to student, interaction in later discussion forums in the pilot courses. Moreover, the courses in which the team assignment was introduced also yielded a greater percentage of subsequent student interaction, than did sister courses in which there was no team assignment, as evidenced by an analysis of the corresponding discussion forums.

1 Introduction

The purpose of this study is to measure the effectiveness of team assignments in encouraging student connectedness and a sense of learning community in an online, largely asynchronous, software development program at the author's previous institution. The assignments are not meant to address industry "best practice" in distributed teamwork; there is a course sequence later in the curriculum that adheres to such processes and practices. Instead the assignments are small-scale and well defined to introduce students to the challenges of working within a coding group, while more importantly encouraging student interaction.

In the process of designing this online program, it became apparent from the literature addressing discussion forums in primarily asynchronous courses [3, 4, 6, 11], that a student sense of connectedness and immersion within a learning community is an essential factor in student retention and academic viability. Whereas, student interaction is more natural to promote in a face-to-face instructional mode, it poses a significant challenge in online degree programs. To address this issue, discussion forums were integrated into all of the courses in the recently introduced online degree curriculum. An assessment of the effectiveness of these discussion forums identified some weaknesses in promoting a sense of learning community.

Each course in the new online software development program adheres to a common template distributed over an eight-week term, developed specifically for this program. The template dictates that in each of the first seven weeks of a course, students are required to view a series of asynchronous lectures, complete a homework assignment and a quiz, and participate throughout the week in two or three asynchronous discussion forums. The eighth week of each course consists of a Final Exam and one discussion forum.

Students view asynchronous lectures in virtual isolation, so students do not get the benefits of questions, comments and insights offered by other students. Nor do students acquire a sense of community, which is common as a result of such interaction. As a method to address this lack of interaction, students are required to post in discussion forums several times throughout the week, in order to foster ongoing engagement. These discussion forums are asynchronous and student posts are guided by instructor posed topics and questions.

While the frequency of student posts has been observed to be satisfactory, and the quality of student posts has been praiseworthy, the lack of interaction between students remains a concern. In fact, prior to this study, student to student interaction in the discussion forums was inadequate at best.

To address this inadequacy, we sought to identify initiatives that might remedy the situation. An obvious remedy would be to convert the asynchronous discussion forums to synchronous forums. However, the asynchronous aspect of

the lectures and discussions was considered essential. Essential, for the express scheduling benefit and convenience of the students in the program. Subsequently, in an effort to strengthen a sense of learning community, in balance with the pressure to implement asynchronous lectures and discussion forums, as the objective of this research, we introduced team assignments within some of the individual courses of the curriculum.

The inclusion of team assignments within software development courses is in line with soft skill expectations of our graduates. The importance of group work in computer science and software development courses is well-accepted. In fact, the ACM/IEEE Computer Science Curricula 2013 curriculum guidelines notes the importance of soft skills, such as teamwork competency, in preparing students for the workforce [2]. One of the ABET Criteria for Accrediting Computing programs lists, as one of the student outcomes, the ability to work effectively as a team member or team leader in activities appropriate to the program's discipline [1]. Such industry soft skill requirements are substantiated by wide-ranging industry trends that emphasize group development work, in software development activities, as is evidenced by the popularity of Agile and DevOps, for example.

The research subject of this paper was initiated to gauge the influence that team assignments might have on a student's perception of community within the program, and sense of connectedness to fellow students in the program, while also providing students with a necessary soft skill required for success later, when employed in the industry.

2 Related Work

In support of asynchronous methods of delivery, research has demonstrated that students prefer to use asynchronous, anonymous discussion boards over synchronous methods such as direct chat [7]. This same study reported that students with a greater number of posts tended to achieve higher grades. Similar results have been found in MOOCs environments, where students who contributed to discussion forums had a higher rate of successfully passing the course [8].

It is also encouraging that some reports have revealed that asynchronous online discussions improve students perceived learning [5, 9]. Other research has also found a correlation of convergent validity between the level of a student's perception of learning and the quality of the student's contribution in discussion forums [10].

The value of online discussion as the main vehicle of communication in an online learning community has been reiterated in studies, that extend to further emphasize the critical requirement that such online discussions foster

an actual sense of community [4, 6, 11]. The critical importance of creating a sense of community in online courses, is reported to reap the benefits of both a greater level of student satisfaction and achievement [6].

Moreover, another investigation of online forums differentiates the importance of interaction as opposed to frequency of discussion forum contribution. This research stresses that forums exhibiting a greater percentage of learner interactions demonstrate a stronger sense of community. It further states that orphaned contributions, on the other hand, identified as posts without an interactive thread, demonstrate a lower reported level of student sense of community [3].

Interestingly, related work reports that MOOC platform designers should consider building community-oriented features, such as small-group projects or facilitated discussions, for successful outcomes [12]. And, some also recommend assigning students tasks that require collaboration, but stress the importance of retaining individual assignments as well, to accommodate different personalities and learning styles [11].

3 Pedagogical Modification

A lack of student interaction was observed within the weekly discussion forums of the courses comprising our new online software development degree. This observation brought the degree of sense of learning community, within the online software development program, into question. Due to the awareness of this and the understanding of the importance of teamwork in software development, an idea was formulated to include a team assignment in a few courses, on a trial basis. It was determined that the size of a team was to be two or three students.

Two courses in the curriculum were chosen to accommodate the new teamwork assignment. The courses were selected based on their relative position within the curriculum. One of the courses is an early foundation course. In some cases, it is the student's first course, as a beginning Java programming course, in a 12-course curriculum. In other cases, it may be the student's second, or possibly third course, in the curriculum.

The other course in which a team assignment was introduced, is a later course in back-end web programming. It is for most of the students, a fourth or fifth course in the program. All students in this course have been in other courses with each other, although none of their previous courses included a team assignment.

The determination of which assignment to select in these courses, as the team assignment, required further consideration. It was argued that selecting an assignment too early in the course would yield a scope too small for an

actual team project. With this in mind, the assignment for the sixth of the eight-week term was selected, and its scope was significantly increased and enhanced to support a team assignment.

The team assignment, in both courses, was a programming project. The specifications of these coding projects were clearly detailed by the course instructors. Much deliberation went into the interactive requirements of the team assignments. Respectful of the asynchronous environment in which many of our students are required to work, student teams were provided with tools for online meetings, and discussion forums, and were encouraged to exchange telephone numbers. The students were told that they had to document all code that they added to the project. The student's initials were required as part of this comment. Furthermore, before making any code modifications, students were instructed to clear all changes in another student's code, with that other student, explicitly. Moreover, after notifying another student of a change to be made to that student's code, the student making the change was instructed to comment out the "old" code, before adding new code reflecting changes. All changes were required to be documented with program comment statements as well, and with student initials. These requirements were mandated to ensure communication among students in a team, as well as to facilitate the instructor grading of the team assignment.

Interestingly and importantly, the programs developed by the groups were successfully completed. A grading analysis of the comments indicated that all students participated in the coding of the solutions. It was also encouraging to note that students negotiated with each other to finalize successful coding solutions.

3.1 Data Collection Analysis

Data was collected from discussion forums in all courses offered to date in the online software development program, including those of the two courses with team assignments. The data was collected to determine the number of total student posts per week, and the quantity of those student posts that were found to exhibit student to student interaction. Furthermore, those posts that did indicate a student to student interaction, were further partitioned into two groups. The first partition identified those posts that exhibited a student reaching out to solicit follow-up from other students. The second partition identified those posts that were contributed in reply to another student's post.

The data was grouped into three categories: (1) all courses except those that included a team assignment, (2) the two courses that included a team assignment, and (3) all courses in total. The findings are described below.

- Considering all courses in which there was no team assignment, the aver-

age weekly posts that conveyed student to student interaction was computed to be 17.4%. The weekly breakdown of posts that conveyed student to student interactions is provided in Figure 1.

ALL COURSES EXCEPT TEAM ASSIGNMENT	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	
Student posts indicating a reaching out to others	0.0%	0.0%	2.6%	0.0%	0.0%	0.0%	0.0%	0.0%	
Student posts indicating a reply to others	6.1%	22.2%	23.1%	25.0%	11.8%	13.5%	20.0%	15.4%	
									Average
Student-to-student interaction posts	6.1%	22.2%	25.6%	25.0%	11.8%	13.5%	20.0%	15.4%	17.4%

Figure 1: Student Interaction Posts in Courses Without Team Assignment

- Considering those courses in which there was a team assignment, the average weekly posts that conveyed student to student interaction was computed to be 21.8%. The weekly breakdown of posts that conveyed student to student interactions is provided in Figure 2.

JUST TEAM ASSIGNMENT COURSES	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	
Number of Posts reach out	0.0%	6.1%	4.2%	8.3%	0.0%	0.0%	0.0%	0.0%	
Number of Posts reply to other	11.5%	12.1%	12.5%	12.5%	14.8%	26.7%	29.2%	36.4%	
									Average
Total student-to-student posts	11.5%	18.2%	16.7%	20.8%	14.8%	26.7%	29.2%	36.4%	21.8%

Figure 2: Student Interaction Posts in Courses with Team Assignment

- One comparison to determine what effect the team assignment may have had on student interaction within the discussion forums, is to consider the sixth week as a possible turning point in the level of student interaction in discussion forums. To consider this, a comparative analysis was conducted involving the posts made before the team assignment in the sixth week, to those posts contributed in the sixth week through the remaining weeks of the term. As Figure 3 illustrates, the student interaction in the first five weeks, before the introduction of the team assignment, was computed to occur in 16.4% of the discussion forum posts. With the introduction of the team assignment and in the weeks afterward, the average posts illustrating student interaction was computed to be 30.7%.
- But before making any conclusion, the "control" group should be considered. Therefore, Figure 4 illustrates the same student interaction post analysis, in those courses where there was no team assignment. Student interaction in the first five weeks, in those courses in which there was not a team assignment, was computed to occur in 18.1% of the discussion forum posts. In the last three weeks of the term, the average posts illustrating student interaction was computed to be 16.3%.
- Comparing the average percentage of discussion posts exhibiting student

JUST TEAM ASSIGNMENT COURSES	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	
Number of Posts reach out	0.0%	6.1%	4.2%	8.3%	0.0%	0.0%	0.0%	0.0%	
Number of Posts reply to other	11.5%	12.1%	12.5%	12.5%	14.8%	26.7%	29.2%	36.4%	**8 Week Average**
Total student-to-student posts	11.5%	18.2%	16.7%	20.8%	14.8%	26.7%	29.2%	36.4%	21.8%
		16.4%					30.7%		

Figure 3: Student Interaction Posts Pre-Team Assignment and Post-Team Assignment

ALL COURSES EXCEPT TEAM ASSIGNMENT	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	
Student posts indicating a reaching out to others	0.0%	0.0%	2.6%	0.0%	0.0%	0.0%	0.0%	0.0%	
Student posts indicating a reply to others	6.1%	22.2%	23.1%	25.0%	11.8%	13.5%	20.0%	15.4%	**8 Week Average**
Student-to-student interaction posts	6.1%	22.2%	25.6%	25.0%	11.8%	13.5%	20.0%	15.4%	17.4%
		18.1%					16.3%		

Figure 4: Student Interaction Posts in First Five Weeks and Last Three Weeks

interaction in the first five weeks in Figure 3 to those in Figure 4, indicates no significant difference. The computed mean percentages are 16.3% and 18.1% in Figure 3 and Figure 4, respectively, with a p-value of 0.27. However, comparing the sixth week through eighth week, after and during the introduction of the team assignment, yields some very interesting results. The average percentage of student discussion posts that exhibited student interaction in the sixth week through the eighth week in Figure 3 computes to 30.7%, whereas in Figure 4 the mean for the same period is 16.3%, and the computed p-value is 0.026.

- At the end of the courses in which a team assignment was completed, the students were asked to comment in their course evaluation on how much more they felt connected to their fellow students as a result of the group assignment. Seventy-five percent of the students indicated that they felt more connected. Twenty-five percent said that they felt no more strongly connected as a result of the team assignment.
- At the end of the courses in which a team assignment was completed, the students were asked to comment on how much more connected they felt to the degree program in general. Again, seventy-five percent reported that they felt more connected, whereas twenty-five percent reported that they felt no different than they had felt previously. We did not ask these students how connected they felt before the team assignment, as a differentiation for feeling no different.

4 Conclusion

Online instruction is an ever-growing strategic initiative of many institutions of higher education. A daunting issue of remote learning, as opposed to face-to-face instruction, is how to make asynchronous online courses effective in regard to promoting a sense of learning community and student connectedness. Both are considered necessary to ensure program viability and retention. For this reason, this research study chose to examine how to improve the degree of student connectedness, focusing on discussion forums.

In assessing the posts within the discussion forums, it became apparent that there was minimal student to student interaction. Working within the requirement that the discussion forums remain asynchronous, a more creative solution became necessary. The idea of a group assignment to promote student interaction emerged and was tested in two courses in the online degree.

Results were encouraging in that in the courses in which the group assignment had been introduced, subsequent discussion forums showed a student interaction measure of almost twice that which was measured prior to the group assignment. Also, subsequent discussion forums in these pilot study courses exhibited a mean percentage of student interaction that was almost twice that of sister courses in which there was no group assignment.

Finally, in a follow-up survey, incorporated into a course evaluation, seventy-five percent of the students in the courses in which a team assignment had been introduced, reported that they felt more connected to the other students in their courses, and to the degree program, in general.

References

[1] ABET, Inc. Accreditation board for engineering and technology, criteria for accrediting computing programs, 2020 – 2021, Dec 2019.

[2] ACM. Computer science curricula 2013. curriculum guidelines for undergraduate degree programs in computer science, 2013.

[3] Shane Dawson. Online forum discussion interactions as an indicator of student community. *Australasian Journal of Educational Technology*, 22(4), 2006.

[4] Zuheir Khlaif, Hamid Nadiruzzaman, and Kyungbin Kwon. Types of interaction in online discussion forums: A case study. *Journal of Educational Issues*, 3(1):155–169, 2017.

[5] Radu P Mihail, Beth Rubin, and Judy Goldsmith. Online discussions: Improving education in cs? In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 409–414, 2014.

[6] Robert L Moore. Importance of developing community in distance education courses. *TechTrends*, 58(2):20–24, 2014.

[7] Dip Nandi, Margaret Hamilton, James Harland, and Geoff Warburton. How active are students in online discussion forums? In *Proceedings of the Thirteenth Australasian Computing Education Conference-Volume 114*, pages 125–134, 2011.

[8] Alyssa Friend Wise and Yi Cui. Unpacking the relationship between discussion forum participation and learning in moocs: Content is key. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, pages 330–339, 2018.

[9] Dezhi Wu and Starr Roxanne Hiltz. Predicting learning from asynchronous online discussions. *Journal of Asynchronous Learning Networks*, 8(2):139–152, 2004.

[10] Erdi Okan Yılmaz and Halil Yurdugül. The perception of learning in asynchronous online discussions: A scale development study. *Procedia-Social and Behavioral Sciences*, 83:776–780, 2013.

[11] J Yuan and C Kim. Guidelines for facilitating the development of learning communities in online courses. *Journal of Computer Assisted Learning*, 30(3):220–232, 2014.

[12] Saijing Zheng, Mary Beth Rosson, Patrick C Shih, and John M Carroll. Understanding student motivation, behaviors and perceptions in moocs. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 1882–1895, 2015.

Examining the Use of the Virtual World to Enhance Group Learning in Hybrid Courses*

Imad Al Saeed
Computer Science Department
Saint Xavier University
Chicago, IL 60655
alsaeed@sxu.edu

Abstract

The purpose of this study is to examine the effectiveness of adopting a virtual world's environment within hybrid courses to enhance the group learning approach and improve students' scores. The study includes a target population of graduate students of both genders that are taking Mobile Applications as a required course in the summer 2017 and summer 2018 semesters at Saint Xavier University. The general findings indicate that adopting a virtual world's environment increases students' engagement, enhances students' group learning approach, improves the overall grades, and enhances students' retention rate.

1 Introduction

To provide students with a more flexible schedule, existing 16-weeks courses such as Software Engineering, Data Visualization, and Mobil Applications courses were re-designed and delivered in a hybrid format in the summer semester only as 8-week courses. Hybrid courses are a combination of traditional face-to-face and online activities [10]. Hybrid courses are the best alternative solution that provides the opportunity for face-to-face interaction and the benefit of reducing the classroom's seat time [10]. The Computer

Science department encourages faculty to develop the 8-weeks hybrid course version to cover the same materials as they cover in a 16-week course format for the same courses. The purpose of this research study is to examine the adoption of the Virtual World (VW) to enhance the teaching and learning processes of hybrid courses. It is good to mention that the Computer Science department at Saint Xavier University recently revamped its existing Master of Applied Computer Science (MACS) program to a fully online program to provide students with a more flexible schedule. Existing face-to-face 16-weeks courses such as Software Engineering, Data Mining, TCP/IP Architecture, and Protocol were re-designed and delivered in an online format as 14-week courses.

2 Related Work

Group projects are one of the most essential factors of the computer science curriculum [6]. In 2001, Redmond published a research study examining the value of group assignments for programming courses [6]. With group project assignments, students should communicate effectively with each other and participate in a group activity. In 2012, Centellas and Love published a research study focused on examining the effectiveness of a collaborative group-learning project for teaching a core competency in comparative politics. The finding indicated that the students who participated in the project scored significantly higher after the completion of the group project. This research study, however, did not examine how students were cooperating and how the collaborated media has been used [2]. Students sometimes faced some type of challenge to find the right media that can be used for group collaboration. Also, students may find difficulties finding the right time for all of them to meet, discuss, and participate in their group assignments. In 2011, Valente et, al. published a research study that examined the effects of a social network for group assignment strategies as an effective approach for online collaboration [8]. To succeed, hybrid courses must be improved in a way that can provide more interaction time for the students. Babb et al. [1] argued that "create a learning environment that supports the growth of a community of learners and shared knowledge ... [and] that fosters interaction, dialogue, and mentoring in an effort to produce learning outcomes similar to those in traditional courses" [1]. This research study responded to these studies by examining the adoption of an open-source virtual worlds (second life) to be used as an effective and cost-efficient tool used to increase the interaction time between instructor-students and students-students in hybrid courses. The next section provides a discussion of why Virtual World was chosen, followed by an overview of the general approach chosen to conduct this experiment.

3 Why Virtual World?

The Virtual Worlds are synthetic representations of reality that are focused on the experience that the users of these worlds have [7]. The Virtual World is an open-source online virtual environment that can be customized in different ways [9]. It is widely used for training and education purposes to facilitate trainees' learning activities [4]. The Virtual World is a playground for imagination and expands the boundaries of users' creativity in exploring, defining, creating, designing, modeling real environments, building, coding, document sharing and recording facilities, performing, and collaboration [3]. The Virtual Worlds can be a very effective and cost-efficient environment that can provide a new methodological framework that supports training purposes [5].

4 Approach:

A weekly discussion board (DB) assignment, group project (GP) assignment, and a quiz at a graduate-level were posted with the weekly modules in Canvas. Students are required to write 500-700 words that respond to the DB questions with their thought, ideas, and comments, and write 4-5 pages of original content about their selected GP, upload their software code, and taking a weekly quiz. The written GP is related to the focusing topic(s) and the reading assignments for that week. Students are required to do research, follow APA guidelines with their academic writing, provide intext citation, and provide high-quality paper. The study assumed that the students know how to write using APA styles and how to find peer viewed references. Below are two examples of GP assignments.

5 Example assignments:

5.1 GP Assignment1

Complete the following case programming projects. Use the same steps and techniques taught within the chapter. Submit the program you create to your instructor.

> **Requirements Document:**
> **Application Title:** Business Card App
> **Purpose:** In the Business Card App, your address, and other information are displayed.
> **Algorithms:** The opening screen displays a simple business card with your personal information. The first line should include your name. The second line should include your future dream job title. The third line

should include your address. The fourth line should include your phone number.

5.2 GP Assignment2

Complete the following case programming projects. Use the same steps and techniques taught within the chapter. Submit the program you create to your instructor.

Requirements Document:
Application Title: Your Contacts App—Address Book
Purpose: This large app provides business contact information in an address book. Create two screens for contacts for the app. You use the app to select a particular contact, and then display that person's information.
Algorithms:

1. The opening screen displays two names of contacts with the last name starting with the letter J. Each contact has a separate Button control below the name. Create your own layout.
2. The second screen displays the name, address, phone number, and picture of the contact. Create your own layout.

Conditions: Three Java classes and three XML layouts are needed.

6 Hypotheses:

This study focuses on the online GP assignments of the hybrid course. There are eight GP assignments involved in the course. The research study hypothesizes the following:

1. Students who are accepted to participate in the experimental study and attended the virtual word meeting score better than other students who are in a traditional hybrid course.
2. Students who are in a traditional hybrid classroom setting and do their assignments independently score lower than students who participating in the experimental study.

To address these hypotheses, measuring the learning is needed. For the GP assignments, students were divided into five groups of 4. For this study, students were required to conduct research, write 4-5 pages of original content, and submit the GP assignments as a group. The students focused on the quality and timing of their initial responses, their responses to their peers, and group paper assignments in both classrooms setting (with and without

VW). The research study assumed that the GP assignment instructions are written well, assess what they are intended to assess, and graded fairly. Also, the research study assumed that all students were following the school policy on academic integrity and were doing their own work. To eliminate any bias, two experienced faculty members with Mobile Application programming were employed to grade the student's discussion board assignments by using the following rubric in Table 1:

Table 1: Group Project (GP) Assignments Rubric.

Criteria	Pts
Task Requirements	30.0 pts
Demonstrate and application of knowledge	50.0 pts
Academic writing and format	20.0 pts

Each participant was assigned a randomized ID number to keep the results coordinated and anonymous. The two expert faculty members will not be able to identify the students when they grade the assignments. Participants have the right to change their mind and withdraw from the study at any time they want, even after signing the consent form and continue following the traditional hybrid class approach to complete their weekly class activities.

7 Logistics of Data Collection

The assignment rubric (see Table 1) used to assess the students' understanding of the subject matter through providing high quality written group project paper. Also, participants are required to respond to the survey's questionnaires. An online Web-based survey engine (survey monkey) was employed to distribute the questionnaire (as open-ended questions) to the students. Only contextual information relevant to the research goal is needed to be collected, and this was achieved by (1) a follow-up of specific questions after the general questions, (2) keeping the questionnaire short, and (3) presenting the questions in a logical order to facilitate completion. The score data was assembled and anonymized after the grades were in. No personally identifiable data was collected.

8 Logistics of Data Analysis

A qualitative method represented by open-ended questions was used to collect detailed information about the research problems and research goals. The

Nvivo studio will be used for data analysis. The records of this study were kept private. In any report of this study that might be published, the researcher did not include any information that will make it possible to identify students.

9 Experiment

The Mobile Applications course of the Summer 2018 semester was used as a pilot course in this experimental study. The study compared the students' grades collected from this study (with VW) with the students' grades collected from the previous semester (Summer 2017 semester) (without VW). All students who enrolled in the Summer 2018 semester were choosing to participate in the reach study and signed the consent form. The class meetings and lectures were conducted at the Virtual Worlds environment- Second Life. The researcher rented a land (the Private Region within Second Life) that runs on servers hosted specifically by the Linden Lab and built his classroom environment and paid the fee (see Figure 1). All access to the rented cloakroom environment requires a login token to establish the identity and permissions, including land access. Each student received his/her own code that cannot be shared with others. Each participant is required to create his/her own avatar with fake names to access the environment. No outside users can access, see, or listen to the classroom discussion.

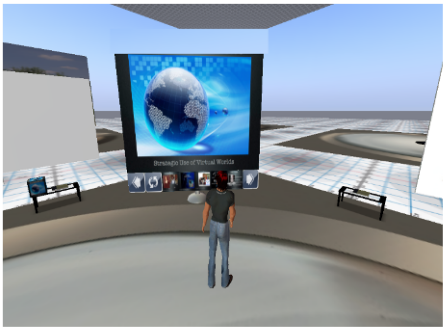


Figure 1. Classroom initial setup in second life.

Figure 2. Students private ground meeting in VW.

The researcher met with his students in Second life on Wednesdays of each week from 7:00 PM – 9:00 PM. The researcher discussed the focusing topic through lecturing, demonstrating, collaborating, classroom discussion, and debriefing. The researcher explained the GP assignment to the students and then posted them along with clear instructions on Canvas. Students that attended

the meeting can listen to the lecture, participate in the class discussion, ask questions, and participate with other classroom activities. The researcher set a specific room for each group to discuss their group assignment privately (see Figure 2). Students submitted their written responses on the Canvas course to be graded. The researcher collected group project written documents and sent them to the two expert faculty members for grading without students' identifications. After the final grades were posted of the course, the survey is sent to the participants to collect survey responses. All student names were replaced by random numeric keys to anonymize the data. The survey consisted of ten questionnaires that were submitted to all students via Survey Monkey.

10 Results:

The study includes eight GP assignments. Including all GP assignment were very important to examine the students' knowledge level and to eliminate any bias for choosing specific assignments than the others. In order to evaluate the hypotheses, the study considered the students who scored 80% and above as "Well Done" on the DB and GP project assignments, and "Poorly Done" for scoring 60% or below on the assignments. The table below showed the average student's grades for the two semesters graded by two expert semesters.

Table 2: Summer 2018 semester – GP assignment (with VW).

GP assignments	Week1	Week2	Week3	Week4	Week5	Week6	Week7	Week8
Instructor 1	75%	85%	90%	85%	84%	85%	90%	95%
Instructor 2	79%	81%	84%	82%	89%	89%	92%	96%

Table 3: Summer 2017 semester – GP assignment (without VW).

DB assignments	Week1	Week2	Week3	Week4	Week5	Week6	Week7	Week8
Instructor 1	72%	77%	71%	66%	75%	76%	79%	85%
Instructor 2	68%	78%	70%	70%	74%	69%	82%	82%

For statistical analysis, a repeated-measure ANOVA was used on the assignment score for students who completed all eight GP assignments. See the tables above for mean GP scores. The results showed that the overall score on the GP assignments for students who participated in the experiment was significantly different ($p<0.001$). This result is strongly supported by hypothesis 1. Also, the results showed that the overall scores on the GP assignments for students in a traditional hybrid course were significant ($P<0.001$). This result is strongly supported by hypothesis 2. As the statistical analysis showed, all students

who participated in the virtual world experiment submitted their assignments and score better than other students. On the other hand, qualitative analysis from collected students' responses to the open-ended questions showed that all students liked the online virtual meeting in Second Life. Students stated that the weekly online meeting provides them with a great opportunity to interact with their instructor and their peers, encouraged them to ask questions, and increase their engagement. For example, one student mentioned that "I am always too shy to ask questions. However, meeting in VW encouraged me to ask questions because my real identity was hidden." Another student mentioned that "I love the idea of the VW meeting and interaction. I do not like the idea of reviewing other students' posts and just reply to it." While one student indicated that "I am really surprised because my grades were improved by participating in this great experiment, ..., VW interaction saved a lot of time for me researching answers for the DB questions." These results considered as evidence that adopting a virtual world's environment increased students' engagement, improved the overall grades, and improved students' retention rate. It is important to mention here that the only issue that students faced during the experience is the learning curve of learning new tools. Some students faced some technical challenges such as Internet connection is not enough, the voice is not clear, and the quality of the microphone they used.

11 Conclusion

The purpose of this study was to examine the effectiveness of adopting a virtual world's environment within a hybrid course to enhance students' project assignments and improve their scores in online courses. The result reflected from this experiment provides evidence about adopting the VW enhanced students' interaction and engagement in the class, encouraged to ask questions, and improved their grades. The only drawback in this experience is the technical issue that students face for the first time when they tried to interact with the VW environments and the learning curve required to master it. The finding also indicated that the only potential drawbacks include learning time required and technological issues involved in using the online virtual environment. Use in Virtual Worlds could be the best solution for schools with the COVID-19 issue.

References

[1] Robert E Botsch and Carol S Botsch. Audiences and outcomes in online and traditional american government classes revisited. *PS: Political Science & Politics*, 45(3):493–500, 2012.

[2] Miguel Centellas and Gregory J Love. "we're off to replace the wizard": Lessons from a collaborative group project assignment. *PS: Political Science & Politics*, 45(3):506–512, 2012.

[3] Yung-Fang Chen, Genaro Rebolledo-Mendez, Fotis Liarokapis, Sara de Freitas, and Eleanor Parker. The use of virtual world platforms for supporting an emergency response training exercise. *Proc. of the 13th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational Serious Games*, 2008.

[4] Shailey Minocha and Christopher Leslie Hardy. Designing navigation and wayfinding in 3D virtual learning spaces. In *Proceedings of the 23rd Australian Computer-Human Interaction Conference*, pages 211–220, 2011.

[5] Youngblood P, Harter P, Srivastava S, and et al. Evaluation of virtual learning environments of emergency medicine and first responder training, 2005. Presentation at the Games for Health, Baltimore.

[6] Michael A Redmond. A computer program to aid assignment of student project groups. *ACM SIGCSE Bulletin*, 33(1):134–138, 2001.

[7] Sarah Smith-Robbins. Are virtual worlds (still) relevant in education? *eLearn*, 2011(12), 2011.

[8] Thomas W Valente, Beth R Hoffman, Annamara Ritt-Olson, Kara Lichtman, and C Anderson Johnson. Effects of a social-network method for group assignment strategies on peer-led tobacco prevention programs in schools. *American journal of public health*, 93(11):1837–1843, 2003.

[9] Maria Vargas-Vera. Reflections on the second life platform used in the development of a virtual university campus. *International Journal of Knowledge Society Research (IJKSR)*, 7(4):12–23, 2016.

[10] Barbara Ward. The best of both worlds: A hybrid statistics course. *Journal of Statistics Education*, 12(3), 2004.

KielceRB: A Highly Customizable Templating Engine for Course Documents

Zachary Kurmas
School of Computing
Grand Valley State University
Allendale, MI 49401
kurmasz@gvsu.edu

Abstract

We introduce KielceRB: a customizable templating engine for generating assignments, syllabi, web pages, and other course documents. KielceRB loads a hierarchy of key-value pairs from files at various file system levels. These values can then be inserted into web pages and other documents using Ruby's ERB templating engine. KielceRB simplifies the maintenance of course documents by moving data that changes from semester to semester into external data files where they can be easily identified and updated. By loading data from various file system levels, it is easy to share values among all documents for a particular course and/or semester. Data "values" can also be functions, which allow the values appearing in the documents to be composed and/or calculated.

1 Introduction

We introduce KielceRB, a highly customizable templating engine for assignments, syllabi, web pages, and other classroom documents. KielceRB loads a hierarchical set of key-value pairs from a series of data files, then inserts those values into course documents using Ruby's ERB templating engine. Placing frequently changing items like due dates and software version numbers in a separate data file makes updating a document from semester to semester easier and less error prone.

`KielceRB` also includes a simple deploy system for those instructors who prefer posting course documents on a traditional web site rather than using a Learning Management System (LMS) like Blackboard, Canvas, or Moodle.

1.1 Motivation

We designed `KielceRB` to address several inefficiencies in maintaining course documents. First, maintaining course documents can be error prone. Every semester, the instructor must search through each document and update data like due dates, semester names, and software versions. In our experience, it is easy to overlook updates – especially when a document has several dates (e.g., a syllabus, or an assignment with several deliverables). Moving these frequently-changing values into a separate document makes it easy to identify and update them all.

Second, making the updates can be tedious. Some values, such as the semester name, appear in every document. Similarly, there are values such as office hours and the withdraw deadline that appear in each course's syllabus. Moving these values to a separate document means they need only be updated once per semester, as opposed to once per semester per course.

We also designed `KielceRB` to simplify the process of pushing course documents to a web page. We have noticed that as the use of Learning Management Systems (such as Blackboard, Canvas, and Moodle) has increased over the years, it has become more difficult to casually browse the web and see how other instructors are approaching their courses. We believe this inhibits innovation; therefore, we make a point of posting our course materials where they can be easily found by other instructors and indexed by search engines. `KielceRB` supports this goal by providing scripts to (1) separate public content (e.g., the assignment description and any "starter code") from private content (e.g., solutions and notes) and (2) push that public content to a web server.

Finally, our design emphasizes *customizeability* over *configurability*. Many frameworks and tools make a point of being configurable: They offer many settings to allow users to adjust the tool to their own preferences. The challenge (as anybody who has spent time customizing a new IDE knows) is that it often takes a lot of time and effort to learn about all the options and how to use them. In the field of Computing education, many instructors choose to write their own tools and assignments rather than either (1) conform to the patterns established in an existing tool, or (2) take the time to learn how to effectively use all the configuration options.

Rather than providing a long list of configuration options that few instructors will take the time to learn, we decided to make `KielceRB` as simple as possible and encourage instructors to simply *customize* (i.e., re-write) it as they see fit. This approach certainly doesn't work for educators as a whole;

but, we believe it is a good match for how many Computing instructors prefer to work (especially those who like to code).

2 Templating Engine

At a high level, KielceRB loads a set of key-value pairs from a set of files, then uses Ruby's ERB templating engine to insert those values into a document. A data file can be as simple as

```
{
   name: 'Tom'
}
```

The most straightforward way to insert that value is to add ERB's <%= %> tags to a document: (KielceRB loads the data file into a global object named $d.)

```
  Hello, <%= $d.name %>. How are you today?
```

Generating the final product is as simple as running

```
kielce.rb inputFile.erb > outputFile.html
```

ERB is designed for the generation of HTML documents; but instructors can just as easily use ERB and/or KielceRB to help prepare any text document (e.g., LaTeX).

In addition to basic templating, KielceRB provides a few additional features to support its use for maintaining course documents:

2.1 Hierarchical data object

A KielceRB data file is Ruby code that returns a hash. As Figure 1 shows, the values of this hash can themselves be other hashes, allowing users to organize data hierarchically. This hash is converted into an object with properties instead of key/value pairs so that data can then be inserted into documents using the more convenient method call syntax:

```
Drop deadline: <%= $d.semester.dropDeadline %>
```

2.2 Hierarchy of data files

To facilitate the sharing of data among several courses and/or assignments, KielceRB merges the hashes from several files into a single hierarchy. Specifically, the tool will search all ancestor directories for files that begin with

```
{
  course: {
    prefix: 'CIS',
    number: '371',
    name: 'Web Application Programming',
  },

  semester: {
    year: 2020,
    term: 'Winter',
    dropDeadline: 'Friday, 6 March',
  },

  general: {
    myURL: 'https://www.myfavorite.edu/~smith'",
    honestyURL: 'https://www.myfavorite.edu/academic-honesty',
  },
}
```

Figure 1: Sample `KielceRB` input file

```
Courses
 + WebProgramming
   + Homework
     + WebServer
     + WebBrowser
     + CSS
 + ComputerArchitecture
   + Homework
     + Processes
     + Threads
     + Scheduling
```

Figure 2: Sample file structure for course documents

```
general: {
  piazza: lambda do |data|
   "https://piazza.com/gvsu/#{data.root.sem.piazzaName}/#{data.root.course.id}"
  end
},

course: {
  prefix 'CIS',
  id: -> (data) { "#{data.local.prefix}#{data.local.number}" },
},

sem: {
  year: 2020,
  term: 'Winter',
  dropDeadline: 'Friday, 6 March',

  fullName: -> (data) { "#{data.local.term} #{data.local.year}" },
  shortName: -> (data) { "#{data.local.term[0, 1]}#{data.local.year % 100}" },
  piazzaName: -> (data) { "#{data.local.term.downcase}#{data.local.year}" }
}
```

Figure 3: Semester data for moderately complex configuration

```
course:  {
  number: 371,
  refDir: 'Manuals',
  devManual: -> (data) {"#{data.local.refDir}/IntelDev_v#{data.args[0]}.pdf"},
  }
```

Figure 4: Data for one course

`kielce_data` and end with `.rb`. Thus, when using a typical file structure like the one shown in Figure 2, data shared among all course documents can be placed in `Courses/kielce_data_general.rb` (for example, items in the `general` and `semester` groups above); data shared among all documents for a specific course can go in a data file in that course's directory (e.g., `Courses/WebProgramming/kielce_data_wp.rb` and `Courses/ComputerArchitecture/kielce_data_ca.rb`); and, data for each specific assignment can go directly in that assignment's directory. (It is not necessary for each data file to have a unique name. We do that because it is easier to see at a glance which file we are currently editing when we are working on multiple files at once.)

2.3 Functions

The values can also be functions (i.e., Ruby lambdas), allowing us to write code to compute and/or compose values. For example, we use a `semester` data section similar to that shown in Figure 3. The `fullName` lambda combines the `term` and `year` values into the single string "Winter 2020" — allowing us to reference the "full name" of the semester as a single entity. The `shortName` lambda automatically generates the abbreviated string "W20". Thus, each semester, we need only update `year`, `term`, and `dropDeadline`; the other values get updated automatically.

Notice also that our semester data file also contains some course data – namely `prefix` and the `id` lambda. This is because the prefix is the same for all of our courses. Similarly, the rule for creating the course id (e.g., "CIS371") is identical for all courses. Therefore, we place these items in a file that is used by all courses. Similarly, the `piazza` lambda in the `general` section is also identical for all courses. Both lambdas rely on data that is unique for each course. In other words, we can establish "high-level" rules, but allow the specific data to be provided at a "lower" level.

The `data` parameter to the lambdas contains three objects:

- `root` refers to the root of the data hierarchy. Thus, the `fullName` lambda could have alternatively accessed the year as `data.root.semester.year`.
- `local` refers to the node in the data hierarchy that lambda is declared in. For example, in the `fullName` lambda, `data.local` refers to the `semester` block. (This option serves primarily as a shortcut for users who employ a deep hierarchy.)
- `args` refers to arguments passed to the lambda. Figure 4, demonstrates how users can generate the URL for a local copy of Volume 3 of the Intel Developer's Manual using syntax like `<%= $d.course.devManual(3) %>`. The actual parameter `3` is made available in the lambda as `data.args[0]`.

The examples above are all relatively simple, one-line functions; but, functions can be as long and complex as the user desires. For example, we have used longer functions to (1) format the course textbook (placing the title in italics, creating a link to the book's web site, etc.), and (2) generate a list of a courses's weekly labs.

2.4 Plug-ins

`KielceRB` also contains a rudimentary plug-in system, where users can place code that is too long and/or complex to reasonably be contained in a well-organized, maintainable data file. Specifically, `KielceRB` looks for directories

named `kielce_modules` and loads any Ruby modules found there. Those modules are then made available to the lambdas in the `kielce_data` files.

To demonstrate the usefulness of plug-ins, we wrote a plug-in that generates a timeline for a course based on data in an Excel spreadsheet. The plug-in code uses the `rubyXL` Ruby gem to read the Excel file, then generates two HTML tables: One large table containing a day-by-day listing of course topics, readings, references, and due dates. The second, smaller table just lists assignment deadlines and is placed on the course's main page. Web pages generated using this plug in can be seen here: `https://bit.ly/2YlolEF`

2.5 Miscellaneous

Finally, we mention a few minor, but helpful, features:

- `KielceRB` can import one file into another. This makes it easy, for example, to include a common CSS file on all pages, or add a common navigation bar to all pages.

- The tool adds a `link` method to the `String` class, which allows users to quickly generate a link from a string:

  ```
  <%= 'https://somereference.com/topic'.link() %>
  ```

  When we include a link in a course assignment, we often want the text of the link to be the URL itself so students can easily see and copy it, if desired. This short-cut allows us to avoid typing the URL twice.

3 Deploy Helper

We prefer to deploy assignments and other course documents to our university-provided web pages rather than an LMS. First, our university's LMS offers neither an API nor a command-line interface, so we must manually navigate through a slow, awkward, inefficient web iterface to deploy documents. As a result, an operation that should take less than a second takes tens of seconds. Second, we prefer to share our content with the educational community to help propagate new and innovative assignments and approaches to teaching. Although most LMS content can be made accessible to all, in our experience, traditional web pages are easier to search and navigate.

To facilitate the deployment of course documents, `KielceRB` includes a deploy helper that automatically copies documents into a local mirror of the teaching section of our web site. We then use `rsync` to push those changes to the actual web page.

The deploy helper uses the "convention over configuration" principle [5]. It assumes that our course documents are organized in a structure similar to that

```
function prepare() {
  kielce.rb -q css.html.erb > $target/index.html

  cp Images/* ${target}/Images
}
```

Figure 5: Typical `prepare_assignment` script

shown in Figure 2 (specifically, a `Courses` directory containing a subdirectory for each course). The teaching section of the web site is also assumed to maintain a parallel structure.

When run, the deploy helper does the following:

1. Creates a directory in the local mirror, if necessary. For example, if the helper is run from within `Courses/WebProgramming/Homework/CSS`, then a directory with the same name is created in the local mirror;

2. Runs a bash script named `prepare_assignment` that prepares the public documents and places them in the target directory within the local mirror; and

3. Creates a sym-link to the target directory.

Figure 5 shows a typical `prepare_assignment` script. This script must contain a function named `prepare`. The helper initializes `$target` to the target directory before calling `prepare`. In this example, the first step is to run `KielceRB` and redirect the output to the target directory. Next, other supporting documents (e.g., images) are copied to the target directory. The helper copies an explicit list of files to a separate target directory (as opposed to simply creating a sym-link to the current directory) because many assignment directories contain solutions or other files that are not intended to be public.

4 Customizable vs Configurable

Many complex software products, such as IDEs are *configurable*: There is a large configuration file with many options that allows the user to specify desired behavior. `KielceRB` takes a different approach to flexibility: The source code is sufficiently simple that the best way to make changes is to simply edit (i.e., "*customize*") it. The core code is only 350 lines long including comments. Adding a configuration system would make this code more difficult to follow and limit the types of changes that can easily be made.

The most complex parts of the templating system are:

- Merging the various `kielce_data` files, and

- Converting the resulting merged hash into an object.

We don't anticipate much demand for customization in this area. We do expect that different users may prefer different conventions for naming and finding the data files. In which case, the most flexible way to support that change is to encourage the users to simply replace those few lines of code. For example, `KielceRB` currently runs this line of code at each level of the directory hierarchy to find input files:

```
data = Dir.glob("#{dir}/KielceData/kielce_data*.rb") +
       Dir.glob("#{dir}/kielce_data*.rb")
```

We believe it makes far more sense to modify this line of code than to attempt to abstract all reasonable naming conventions and/or search algorithms.

5 Related Work

Templating engines are certainly nothing new, and are at least as old as the M4 macro processor introduced in 1977 by Kernighan and Ritchie [1]. Using a templating engine to generate web pages has been around since as least the introduction of PHP in 1995 [4]. The ERB templating engine that `KielceRB` uses is part of Ruby on Rails and has been around since 2005 [3].

`KielceRB`'s main contribution is not the templating engine, or the use of a series of data files in a hierarchy; but, rather a minimalist implementation that (1) reduces the steep learning curve and complex installation common in many web frameworks, and (2) makes the tool attractive to those computing instructors who, by their nature, tend to prefer implementing their own tools over conforming their workflow to existing tools.

`KielceRB`'s design is based on an earlier client-side JavaScript tool called simply `Kielce`. When using this client-side version, the browser would load data files from each level of the hierarchy, merge the files into a single data structure, then insert the data into the DOM. The original `Kielce` demonstrated the value of both (1) the hierarchical series of data files, and (2) allowing the data values to be functions; however, the client-side nature of the tool (i.e., inserting the values into documents using JavaScript running in the browser) proved to be slow, error-prone, and difficult to debug. Overall, it did not provide a good experience for either the instructors or the students [2].

6 Our Experience

We have been using `KielceRB` for two years. It has been a very positive experience: Updating our courses at the beginning of each semester takes less than 10 minutes. Before using `Kielce` it would take up to an hour — not including the time to fix oversights. Preparing new assignments is also very easy. In addition to simplifying the updating of assignments, having a templating engine has made it much easier to give each document a consistent look and feel (with both a common header and a common style sheet).

7 Availability

The software, as well as full documentation and additional sample uses, is available at `https://www.cis.gvsu.edu/~kurmasz/Software`

References

[1] Drian W. Kerninghan and Dennis Ritchie. The m4 macro processor. Technical report, Bell Laboratories, 1977.

[2] Zachary Kurmas. Kielce: Configurable html course documents. In *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '12, page 403, New York, NY, USA, 2012. Association for Computing Machinery.

[3] `https://weblog.rubyonrails.org/2005/12/13/rails-1-0-party-like-its-one-oh-oh/`.

[4] `https://groups.google.com/forum/#!msg/comp.infosystems.www.authoring.cgi/PyJ25gZ6z7A/M9FkTUVDfcwJ`.

[5] `https://rubyonrails.org/doctrine/`.

Why Cs Departments Should Consider Offering CUDA as a Standalone Course*

Imad Al Saeed
Computer Science Department
Saint Xavier University
Chicago, IL 60655
alsaeed@sxu.edu

Abstract

The purpose of this study was to encourage United States universities to offer CUDA courses as one of the main courses within the computer science department. This study included a target population of professors who were currently teaching CUDA courses, professors who used CUDA for a while but have discontinued using it in their courses, and professors who were teaching relevant courses such as Graphic Design, Game Design, Machine Learning, and Parallel Programming courses but haven't looked closely at CUDA to reasonably know whether it's a great technology for teaching the principles at 100 universities. The general findings indicated that CUDA should be taught either as a standalone course or as a tool to teach other relevant courses within all Computer Science department at United States Universities. The general findings also indicated few professors hadn't previously heard of (CUDA) as widespread use. It is now on their radar because of their participation in this study. This study provided a business plan that addressed all the resources needed to either create new CUDA courses or use it as an effective tool for other relevant Computer Science courses.

1 Introduction

In 2001, NVIDIA Corporation released a new Graphics Processing Unit (GPU) as a powerful modern many-core accelerator that has become increasingly available in a wide variety of systems such as high-performance computer systems,

traditional workstations, mobile devices, and embedded systems [3]. In 2007, the NVIDIA Corporation introduced the Compute Unified Device Architecture (CUDA), as the most widespread programming platform for GPU computing [7]. Thousands of software developers, researchers, and scientists took an advantage of GPU computing using the CUDA programming models in different areas including image and physics-based gameplay, equity risk analysis, weather prediction, video processing, computational biology and chemistry, etc. For example, the General Mills company used CUDA to create a CUDA-based system that assisted them in finding the optimal way to cook a frozen pizza in the microwave. The SeismicCity company also used CUDA to develop a CUDA-based depth imaging technology system to select new drilling locations to increase their chances of finding oil. Also, OptiTex Ltd used CUDA to simulate the look and movement of clothing on virtual models, allowing them to review, refine, and measure samples before the first piece of fabric are ever cut [4]. In addition, many other authorized NVIDIA partners provide training services and consultation related to CUDA programming such as ArrayFair, Acceleware Ltd, aneo, Dixar, eleks, etc. [5]. The widespread parallel computing hardware has increased the industrial demand for computer science graduates with CUDA programming skills [3]. This encouraged many universities to offer new courses for teaching this advanced technology. According to the NVIDIA company, CUDA technology is being taught at more than 300 universities worldwide including the University of Illinois, Stanford University, the University of Oxford, UC Davis, the University of Sheffield, etc. [6]. There are many problems that may prevent other universities from offering the CUDA course, some of which are financial problems, hiring new CUDA expert instructors, designing effective teaching material, and setting up new laboratories. The purpose of this research was to find suitable solutions for these problems and encourage other United States universities to offer CUDA courses.

2 Methodology

2.1 Hypothesis

For this study, the general null hypotheses were:

H1: CUDA cannot be offered as one of the main courses within the computer science department.

H2: The CUDA course cannot be used as an effective tool to teach the general principles of paralleled programming, Games design, and/or Machine learning.

2.2 Methods

A mixed methodology was employed to achieve research goals. The researcher used both quantitative and qualitative methods to provide the best understanding of the research problems [1]. The quantitative method was an online post-survey for professors who teach CUDA at 100 United States universities. To eliminate any bias, the research study also included professors who used CUDA for a while but have discontinued using it in their courses. The professors who were teaching relevant courses such as Graphic Design, Game Design, Machine Learning, and Parallel Programming courses but haven't looked closely at CUDA to reasonably know whether it's a great technology for teaching the principles. This method is especially effective when the population of interest is very large and/or dispersed across a large geographic area. The qualitative method was a post-interview that consisted of open-ended questions that were submitted to professors at 15 different universities in the United States. The researcher decided to go with this approach because the participants are more likely to be comfortable expressing their true beliefs in a private forum.

2.3 Instrument

To obtain quantitative data for the study, a survey instrument was developed and emailed to professors at 100 universities in the United States. The survey consisted of 10 questions, and each question required a Likert-type response (i.e. Strongly Agree, Agree, Neutral, Disagree, or Strongly Disagree). To obtain qualitative data for the study, an interview instrument was developed and emailed to professors at 15 universities. The interview consisted of 10 open-ended questions that typically began with words such as "What," "Is," or "Please tell me about..." They were not technical questions, but questions that implicitly ask for a response. The purpose of the post-surveys and post-interviews was to gather further details that help interpret and analyze the data.

2.4 Variables

There were three dependent variables in this study. The first was the professors' experience with CUDA architecture and programming. The second was whether offering CUDA as a standalone course or as a tool. The third was the number of professors who are capable of teaching CUDA courses. Also, there were two independent variables in this study. The first was the types of courses used CUDA as a tool. The second was the time.

2.5 Procedure

The data was collected over a one-month period. A survey monkey was employed to design a survey and interview questions. The researcher sent the survey link via email to each professor individually at 100 universities. Also, the interview questions were distributed via email to professors at 15 universities because it is challenging to gather answers to specific questions when the target population is distributed over a wide geographical area. Before completing the survey, each professor was required to read and sign an informed consent form approved by the institution's Institutional Review Board (IRB).

2.6 Logistics of Data Analysis

According to Sharp, the first step in data analysis is identifying recurring patterns or themes [8]. Quantitative and qualitative data were collected and analyzed separately. For quantitative data, a proportion testing technique was employed to determine whether the combined proportion of strongly agree and agree responses were significantly greater than the combined proportion of strongly disagree and disagree responses for a given question in the survey. All descriptive statistics of quantitative data ran into the SPSS software where the alpha level was set at $= 0.05$. On the other hand, qualitative data was collected from the participants' interviews and professionally transcribed, coded, analyzed, and interpreted using the DeDoose software.

3 Result

3.1 Post-Survey

From the 161 participants, eleven professors replied to the researcher via email noting that they have not yet started teaching CUDA technology, so they do not have the experience to report. Of the remaining 150 professors, 88.66% (133 professors) visited the survey website and responded to the survey questions. Of those 88.66% potential respondents who visited the survey, 5.26% (7 responses) were partially completed. Survey participants were required to answer all the non-demographic questions so that it would count as a completed survey. However, participants were not required to answer all the demographic data questions. Although the respondents were Computer Science professors, their identities were unknown to the researcher. The completed responses were examined, and the partially completed were discarded. The demographic information of the professors (45.6%, N = 126) was currently teaching, (32.4%, N = 126) reported that they were used CUDA as a tool for a while but have

discontinued using it in their courses, and (22%, N = 126) reported that they did not use CUDA at all.

3.2 Post-Interview

Of the 19 participants, six professors (31.57%) refused to be interviewed for personal reasons. 68.43% (13 professors) answered the interview questions including the demographic and open-ended questions. The demographic information of the professors was that 67.3% were currently teaching CUDA courses, 12.4% reported that they were used CUDA as a tool for a while but have discontinued using it in their courses for academics and financial reasons, and 20.3% reported that they did not use CUDA at all.

3.3 Hypothesis Testing

To reject the null hypothesis listed above, the combined proportion of the strongly agree and agree responses should be significantly greater than the proportion of combined strongly disagree and disagree responses. The results of Question 5 "Should CUDA be not be offered as one of the main courses within the computer science department" were 64.9% strongly agree and 13.5% agree = 78.4% total, 0% strongly disagree and 5.4% disagree = 5.4% total, and 16.2% were neutral. Since the result of Question 5 was significant, H1 was rejected. The results of Question 7 "Can CUDA be not be used as an effective tool to teach the general principles of paralleled programming, Games design, and/or Machine learning" were 51.4% strongly agree and 37.8% agree = 89.2% total, 0% strongly disagree and disagree, and 10.8% were neutral. Since the results of Question 7 were significant, H2 was rejected.

The researcher did not see any difference in the participants' assessment interview instrument answers. The interview results have validated the results reflected from the professors' post-survey because it gave approximately similar results. The interview participants also discussed the learning objectives and outcomes of their courses, and how they have met the learning objectives.

4 Discussion

The results showed that 78.4% of the participants from the online post-survey and 70.2% of the participants from the personal post-interview indicated that CUDA should be taught as one of the main courses within all computer science department at United States universities. A very limited number of the participants were neutral in their response and no one disagreed with that claim. In addition, most professors who do teach such courses recommend that CUDA should be taught at other universities who are not offering a new course for

CUDA. For example, 94.6% of the participants from the online post-survey and 100% of the participants from the personal post-interview supported this recommendation as well. 89.2% of the participants from the online post-survey and 71.5% of the participants from the personal post-interview indicated that their courses focused heavily on helping students understand the general principles of Parallel Programming, Machine learning, and the Games design and they currently using CUDA as an effective teaching and learning tool. In addition, those participants (97.3% from the online post-survey and 100% from the personal post-interview) were very interested in keeping using CUDA as an effective teaching and learning tool. But with the current increase enrollment that option is not possible. They indicated that to keep using CUDA as a tool, they must build more labs equipped with CUDA enable devices, and with the current budget crises, it is hard to do so. Exactly 97.3% of the participants from the online post-survey and 85.8

5 Suggested Strategic Business Plan

Based on the result reflected from the post-interview indicated that the proposed CUDA course may include free programs as well as professionally developed affordable programs. Schools may use web-based methods as one of their marketing and distribution strategy to offer an initial short class that can be free of charge (level A). To enroll in the CUDA courses, students should be familiar with the concepts of matrix manipulation algorithms and linear algebra, and functions/parameters/return-values, and the C/C++ programming language. Students can download the suitable CUDA development kit release for their hardware for free from the NVIDIA site and start coding with CUDA immediately. The programmers can select the release they want from new and improved CUDA libraries, CUDA driver, CUDA C runtime, development tools, new GPU Computing SDK code samples, etc. Students should check the recent production drivers appropriate for their hardware configuration. The formal instruction that leads to certification and/or degrees would be the premium service (Level B). CUDA courses aim to provide students with the knowledge and hands-on experience in developing and optimizing applications software on massively parallel graphics processing units (GPUs).

5.1 Level A – Introduction to CUDA Programming Course

This course might be offered online. The course may start with object-oriented programming with applications to engineering, reviewing C/C++ programming language concepts, and cover detailed information about the graphics card hardware (GPU) architecture. By the end of the course, students should

be able to understand the basic hardware/software architecture and how to write simple programs with CUDA and how to compile them on Linux and Windows.

5.2 Level B - Advance or Professional Course

This is an advanced or professional level course and might be offered on the ground. It is a lab-intensive that covers state-of-the-art parallel programming optimization methods. The course should discuss a popular programming interface for graphics processors, the CUDA programming tools for NVIDIA processors. The course should continue with a closer view of the internal architecture of graphics processors and how it impacts performance. By the end of the course, students should be able to implement applications and algorithms on graphics processors using CUDA programming. Students should be able to form interdisciplinary teams and optimizing a real-world compute-intensive problem in science or engineering.

5.3 CUDA course objectives and resources needed

The Computer Science Department should start to set up all necessary resources that are needed to teach the CUDA course including lab equipment, the CUDA development kits, textbooks with ISBNs, assignments, rubrics for grading, syllabi, class schedules, train existing instructors or hire expert CUDA instructors, etc. This preparation may require setting a high budget since the NVIDIA cards with GPA prices are high. To solve this financial problem, NVIDIA may offer a grant offering to set up the course labs. They can offer support to help set up labs by both donations and special price discounts.

5.4 Course objectives

By the end of the course, students should be able to:

- Apply knowledge of parallel programming techniques.
- Design a software system within realistic constraints.
- Identify, formulate, and solve complex algorithms using CUDA tools.
- Understand the GPUs, CUDA, Ocelot, OpenCL, parallel programming techniques.

The suggested CUDA course implementation strategy is as follow:

- **Textbook:** Required textbook might be "Programming Massively Parallel Processors: A Hands-on Approach. D. Kirk and W.-M. Hwu. Morgan-Kaufman." [2]

- **Labs:** Offers several labs focusing on GPU/CUDA programming skills, the software interface, and the basic architecture of the device, data layout and decomposition, and application of efficient parallel algorithms that utilize shared memory.
- **Final project:** The project focuses on open-ended research. The final project will culminate in a paper report, and a presentation/demo held at the final few class meetings.

6 Conclusions

The main goal of this research paper is to encourage offering a new course for teaching CUDA technology. Most professors who do teach such courses recommend that CUDA technology should be taught as one of the main courses within the computer science departments. The proposed CUDA course may include free programs as well as professionally developed affordable programs. Universities can take advantage of NVIDIA's grant offering to set up new labs for CUDA courses. Web-based methods might be the best choice for the marketing and distribution of the course information. In addition, a few companion opportunities, such as making a job network for CUDA programmers, a site for archiving and distributing CUDA applications and webinars, and online conferences.

References

[1] John W Creswell and J David Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2017.

[2] David B Kirk and W Hwu Wen-Mei. *Programming massively parallel processors: a hands-on approach*. Morgan kaufmann, 2016.

[3] Chris Lupo, Zoë J Wood, and Christine Victorino. Cross teaching parallelism and ray tracing: a project-based approach to teaching applied parallel computing. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 523–528, 2012.

[4] NVIDIA. CUDA in Action. `http://www.nvidia.com/object/cuda_in_action.html`, retrieved May 02, 2018.

[5] NVIDIA. CUDA Training Partners. `https://developer.nvidia.com/cuda_consultants`, retrieved May 23, 2018.

[6] NVIDIA. Existing University Courses. `developer.nvidia.com/educators/existing-courses`, retrieved June 01, 2018.

[7] Roberto Di Pietro, Flavio Lombardi, and Antonio Villani. Cuda leaks: a detailed hack for cuda and a (partial) fix. *ACM Transactions on Embedded Computing Systems (TECS)*, 15(1):1–25, 2016.

[8] Helen Sharp, Yvonne Rogers, and Jennifer Preece. *Interaction design: beyond human-computer interaction*. John Wiley & Sons, 2007.

A Scattered Neighborly Approach to Solving Nurikabe*

Paul Bass and Aise Zulal Sevkli
Mathematics and Computer Science
Denison University
Granville, OH 43023
{bass_p1,sevklia}@denison.edu

Abstract

Japanese pencil games such as Sudoku have been the subject of algorithm analysis in the past decades. In this paper, we propose a scattered neighborly approach to solving the Nurikabe Puzzle. We propose four neighborhood structures that address the violations of the puzzle, as well as a scatter-based board construction algorithm. Compared to the only existing heuristic algorithm for Nurikabe, our approach completes a similar number of puzzles, and has several computational advantages including easy parallelization.

1 Introduction

A Japanese pencil game, Nurikabe falls into the same category of games as Sudoku. Nurikabe is played on an n by m grid of squares, some of which initially contain numbers. The goal of Nurikabe is to create a board that does not violate any of the following rules:

1. Every numbered cell must occupy a white region (an 'island') formed of contiguous white squares. The island must be sized to the number contained in the cell - for instance, an island that contains the integer '3' must be of size three. A violation of this rule can be seen in Figure 2.

2. Black cells must form a contiguous wall around the individual islands. In other words, no black cell or set of black cells can be 'disconnected' from any other set of black cells. A violation of this rule can be seen in Figure 3.

3. The 'wall' of black squares cannot, at any point, form a 2 by 2 block. A violation of this rule can be seen in Figure 4.

The following board is a valid Nurikabe solution:

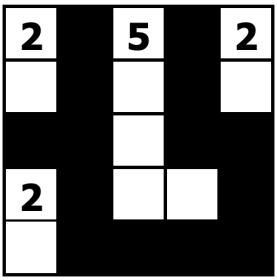


Figure 1: A valid Nurikabe solution

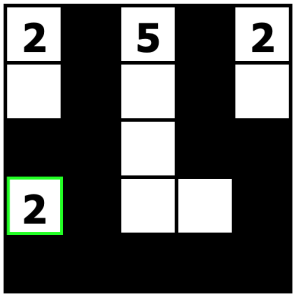


Figure 2: An example of a violation of rule 1

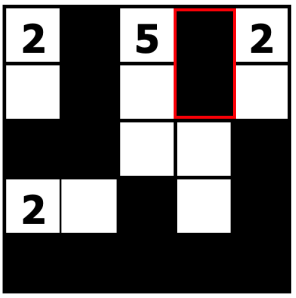


Figure 3: An example of a violation of rule 2 (outlined in red)

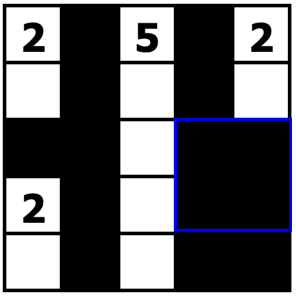


Figure 4: An example of a violation of rule 3 (outlined in blue) - this board also violates rule 2

Nurikabe is oft compared to Sudoku, a game on which extensive research has been conducted [1]. Comparatively, very little known is known about metaheuristic approaches to solving Nurikabe. Inspired by the results of Amos

et al. research into Nurikabe [2], we propose an alternative metaheuristic algorithm that can be used to solve Nurikabe puzzles.

Nurikabe is NP-complete as proved by Holzer et al [3]. Their work caused researchers to shift focus from exact algorithms to metaheuristic and approximation algorithms.

There exists, however, some work on direct solutions to Nurikabe [4]. Answer set programming has been used to create solutions to Nurikabe - however, due to Nurikabe's NP-complete status, such direct approaches become untenable for larger problem sizes.

In a recent paper, Amos et al. applied an ant colony metaheuristic (ACO) algorithm to Nurikabe [2]. Rather than beginning with an all white board and adding black cells to create white islands, their algorithm begins by setting all non integer cells as walls and creates islands by sending out 'ants' from each integer cell. Their algorithm was more effective than the logic based solver on smaller puzzles. Furthermore, the ant colony algorithm takes into account the game rules stated above, rather than (like the aforementioned answer set programming model) attempting to generate a violation-free board from scratch.

2 Proposed Algorithm: A Scattered Neighborly Approach

We propose a novel metaheuristic algorithm combining scatter search[5] and variable neighborhood search (VNS)[6]. At a high level, our algorithm generates a set of diverse solutions using a scatter search, selects the fittest subset of solutions, and searches the space around those solutions to find a correct, solved puzzle using a VNS made up of four neighborhoods.

2.1 Solution Representation

We encode each Nurikabe board using a n by m array of integers. We represent black squares using -1, white squares using 0, and integer squares using the corresponding positive integer.

Our algorithm can be, roughly, subdivided into two components: initial board generation, and neighborhood search.

2.2 Fitness Function

We use the following formula to determine a given board's fitness:

$$(w_1 * D) + (w_2 * W) + (w_3 * B)$$

D, W, and B are variables that, respectively, correspond to the following Nurikabe rule violations:

1. D - The sum total of the following formula for each island: |Island Integer − Island Size|, where 'island integer' denotes the integer found in the integer cell for a given island
2. W - The number of disjoint wall (black block) fragments
3. B - The number of 2x2 black blocks present on the board

Further, w_1 through w_3 determine how we weight each of these violations. These weights help us 'rank' how severe each violation is, helping us to answer questions like: if we fix a 2x2 block by increasing an island's size, causing said island to become larger than it ought to be, is our new board better or worse than our old one? We, through experimentation, determined that the following weights allow us to solve the most puzzles:

1. w_1: 1
2. w_2: 2
3. w_3: 5

2.3 Board Construction

We construct boards using a simple, but highly variable, generation algorithm. We first place the integers on the board, and initialize every other cell as a black square. We then create a pool of w white squares, where w is equal to the sum of the set of integers minus the number of total integers[1]

A weighted random distribution is used to distribute white cells among existing islands such that no two islands overlap. This 'no-overlap' criteria will never be violated - all islands will always remain distinct.

The weighted random distribution is more likely to add cells to islands with larger integers - in other words, an island containing 6 is twice as likely to receive a white square as an island that contains 3.

This process ensures that islands are generated semi-randomly, but also that islands that ought to be larger end up with more white squares than those that ought to be smaller. We then repeat this process, creating a diverse (due to the random nature of our generation algorithm) set of population members. After generation, we select a top percentage of the boards by fitness, and use those as our population pool for the subsequent neighborhood search.

[1]This is equal to the number of white squares that are present in a valid solution - in other words, boards are always initialized with the correct number of white cells

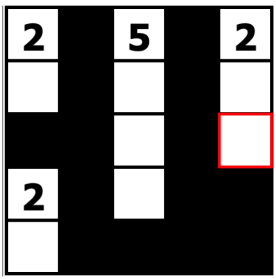
2.4 Neighborhood Structures

We propose four neighborhood structures. All neighborhoods must follow our 'no overlap' rule: white islands cannot overlap. So, any instance in which a white cell is selected and removed does not split an island, and any instance in which a black cell is removed or added neither splits a white island, nor merges two white islands.

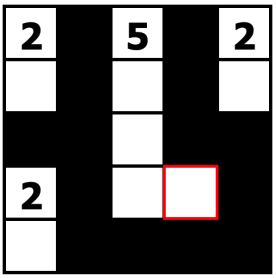
2.4.1 Surplus Island Swap

This neighborhood is intended to help decrease the number of island size violations. We first check which islands have too many white cells, adding all these islands to a set L. We then, using the same process, create a set of islands that are too small[2] (call it S).

We select an island I_L from L and an island I_S from S. We then remove a white cell from I_L without splitting it into two islands, and add a white cell to I_S without creating an overlap with another island. This function decreases the size of islands that are too large and increases the size of islands that are too small. An illustration of this process can be seen in figure (5).



(a) Before surplus swap operation



(b) After surplus swap operation

Figure 5: Surplus swap operation

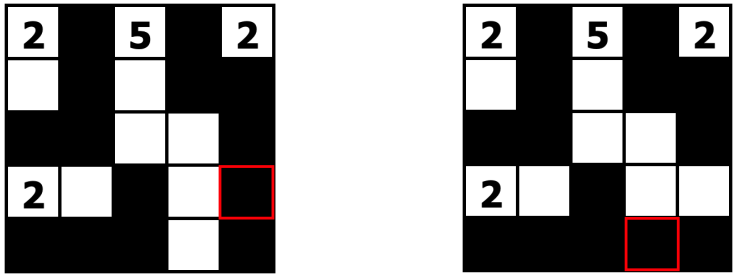
2.4.2 Unify Wall

This neighborhood is intended to help decrease the number of isolated black cells (or 'wall fragments'). We first make a list of wall fragments, then, using a selection weighted towards smaller fragments, select a random fragment. We then swap a random black square from the selected fragment with a random

[2]If there are islands that are too large then, per the pigeonhole principle, there are necessarily islands that are too small

white square abutting another wall fragment. An illustration of this process can be seen in figure (6).

This, over time, unifies the black blocks into a single monolithic, contiguous structure. Since the selection process is weighted towards choosing smaller fragments, we are more likely to remove blocks from smaller fragments and add them to larger fragments.

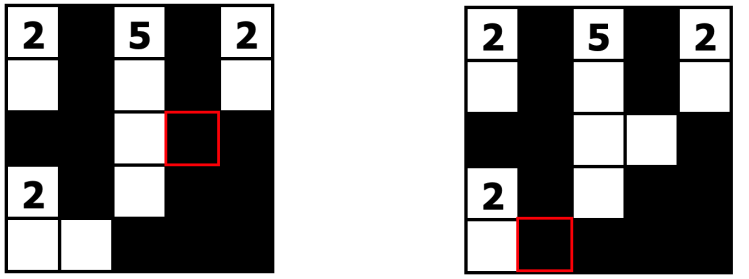


(a) Before unify wall operation (b) After unify wall operation

Figure 6: Unify Wall operation

2.4.3 Break Up 2x2 Black Blocks

This neighborhood, like unify wall, selects a random block that both makes up a 2x2 block and abuts an island. We then swap that black block with a white block from an island, unless said swap would create another 2x2 square. An illustration of this process can be seen in figure (7).

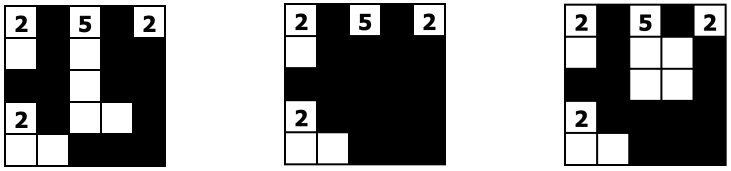


(a) Before break up operation (b) After break up operation

Figure 7: Break up 2x2 operation

2.4.4 Regenerate Island

This neighborhood is intended to help move our search out of 'ruts' in which none of the above neighborhoods can make moves without violating the overlap constraint. We select a random island, set its size to 0, and then regenerate it from scratch. This generates a new island equal in size to the original, but with a different shape, thereby giving other neighborhoods new movement options. An illustration of this process can be seen in figure (8).



(a) Before the regeneration operation (b) During the regeneration operation (c) After regenerating the island

Figure 8: Regeneration operation

2.5 A Scattered Neighborly Search

We combine our scatter search and VNS by, functionally, chaining them together. For each board provided by our scatter search generation and selection method, our VNS iterates a certain number of times. For each VNS iteration, we run each neighborhood search once, attempting to find the first best improvement for that neighborhood. If we can find an improvement over the current global best (call this improved board B_i), we do two things. We first update the current and best boards, terminating if the best board's fitness value reaches 0, indicating that it has been solved. We then iterate through every other board (call a given board B_o) that hasn't been searched yet. During this iteration there is a small chance that B_o will be replaced with B_i. Pseudocode for the algorithm can be found below.

2.6 Pseudocode

Algorithm 1 Scattered Neighborly Search

```
procedure ScatterVNS(board, popSize, fitPercent, probReplace, LSIters)
    popSize ← max(puzzle x-dimension, puzzle y-dimension)^3
    startingBoards[] ← Generate(popSize, board)
    ∀ startingBoard in startingBoards, calculateFitness(startingBoard)
    topBoards[] ← most fit fitPercent% of boards in startingBoards
    globalBest ← topBoards[0]
    for currBoard in topBoards do
        for LSIters many iterations do
            currBoard ← SurplusIsland(currBoard)
            currBoard ← UnifyWall(currBoard)
            currBoard ← BreakUp2x2(currBoard)
            currBoard ← RegenerateIsland(currBoard)
            if Fitness(currBoard) == 0 then
                globalBest ← currBoard
                Break
            if Fitness(currBoard) < Fitness(globalBest) then
                globalBest ← currBoard
                for Board in topBoards do
                    Replace(Board, globalBest, probReplace)
    Return globalBest
```

3 Experimental Results

3.1 Dataset

We use boards from Janko, the same puzzle repository used by Amos et al. as our algorithm's test dataset [7]. From this repository, we selected a set of 33 randomly selected puzzles ranging in size from 9 to 100 cells.

3.2 Parameter Setting

Our algorithm, as seen above, takes the following parameters:

1. fitPercent - This parameter determines what percentage of the initialized boards are selected for our VNS

2. Iterations - This parameter determines how many neighborhoods we run before moving on to the next candidate board

3. ProbReplace - This parameter determines how likely we are to replace a given board with a newly discovered global best

Experimentally, we determined that our program performs best with a fitPercent value of 0.2, an iterations value of 100, and a probReplace value of 0.05.

We ran and tested our algorithm using the same criteria as Amos et al running our algorithm 100 times. If any of the 100 runs found a solution, the puzzle was marked as solved. Our algorithm was tested on an Intel core i-7 8700k.

3.3 Results

Using the above criteria, every puzzle in our dataset was solved as shown in table 1. The 'num cells' column keeps track of how many cells a given puzzle has - this can be used as a proxy for puzzle difficulty. The 'tries to solve' column keeps track of how many tries it took to solve the puzzle in question. Finally, the 'hit ratio' column tracks robustness by tracking what fraction of attempts found a solution to the puzzle[3].

Table 1: Performance of Scattered Neighborly Search on 32 Puzzles

ID	Num. Cells	Time To Solve (sec.)	Tries to Solve	Hit Ratio
0	9	1.12	1	1.0
1	16	1.23	1	1.0
2	25	3.22	1	1.0
3	25	5.16	1	1.0
4	25	2.29	1	1.0
5	25	4.46	1	1.0
6	25	2.62	1	1.0
7	25	4.26	1	1.0
8	25	3.17	1	1.0
9	36	6.49	1	1.0
10	36	7.13	1	1.0
11	36	17.49	1	1.0
12	36	22.73	1	1.0
13	36	9.96	1	1.0
14	36	12.44	1	1.0
15	36	14.72	1	1.0
16	36	25.55	1	1.0

[3]A hit ratio is only provided for the first 19 puzzles, as (in the interest of time) the subsequent puzzles halted after finding a single solution

17	36	31.61	1	1.0
18	36	16.38	1	1.0
19	36	38.55	51	0.03
20	36	52.22	1	
21	36	193.29	1	
22	36	102.82	1	
23	49	34.15	1	
24	49	71.22	1	
25	49	23.15	1	
26	49	74.15	1	
27	49	91.15	21	
28	64	246.15	1	
29	64	102.21	70	
30	81	132.32	53	
31	100	201.11	3	
32	100	231.74	3	

Amos et al's paper does not contain individual puzzle completion times, so we compare our average solved puzzles with Amos et al's in table 2. Small puzzles are puzzles with less than 100 cells, mediums are those with 100-200 cells. We, like Amos et al, count a puzzle as solved if it was solved in under 60 seconds. For our tested data set, there are 31 smalls and 2 mediums, though we only compare smalls as the sample size for medium puzzles is too small.

Table 2: Comparison to ACO

Size	ACO Sol. Percent	ACO Hit Ratio	Our Sol. Percent	Our Hit Ratio
Small	99.2	0.857	80.6	0.95

Our program was able to solve every problem it was given using the same basic criteria as Amos et al's paper. However, Amos et al's algorithm (ACO) was only allowed to run for one minute before a run was counted as a failure. Were our algorithm subjected to the same constraint, it would have failed to solve several of the above puzzles. Our algorithm is, per our hit ratios, quite robust, moreso than Amos et al's'. Every hit ratio except one is 1.0, meaning that every attempt found a valid solution for the puzzle. Our algorithm struggled with one easy puzzle in particular (19), implying that there are some edge cases, or particular kinds of puzzles, it is not well suited to solving. As can be seen below in figures 9 and 10, our algorithm improves solution fitness very quickly:

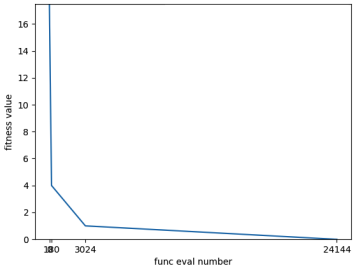


Figure 9: Convergence graph for puzzle 9

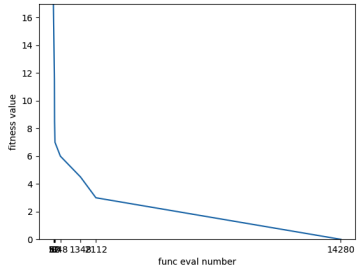


Figure 10: Convergence graph for puzzle 14

4 Conclusion and Further Work

Our algorithm, as demonstrated above is comparable to ACO's performance on easier puzzles, but both takes longer and solves fewer hard puzzles. However, our algorithm is more robust than Amos et al's algorithm per our hit ratio comparison in table 2.

Further, in our algorithm there is both minimal interaction between population members. Therefore, our implementation could be GPU accelerated or multi-threaded (with, say, a thread for each population member) far easier than Amos et al' algorithm. A preliminary attempt using two threads resulted in a 2x speedup, implying that the marginal benefit of additional threads would be quite high.

We also believe that, were we less concerned with runtime, our algorithm would perform better if we found a local optima for each board generated by our scatter search rather than giving up after a certain number of iterations. The above threading suggestion might give us some additional runtime leeway, and thus make this local optima search more practical.

Finally, a VNS is only as good as its neighborhoods, and a scatter search is only as good as its solution generation algorithm. Therefore, additional neighborhoods or improvements to our initial board generation would dramatically increase the efficiency of the algorithm.

References

[1] K. N. Das, S. Bhatia, and S. Puri, "A retrievable ga for solving sudoku puzzles", 2011.

[2] M. Amos, M. Crossley, and H. Lloyd, "Solving nurikabe with ant colony optimization (extended version)", Jun. 2019. DOI: 10.13140/RG.2.2.23813.19685.

[3] M. Holzer, A. Klein, and M. Kutrib, "On the np-completeness of the nurikabe pencil puzzle and variants thereof", in *In proceedings of the 3rd international conference on fun with algorithms*, 2008.

[4] M. Cayli, E. Kavlak, K. Ferhan, and E. Erdem, "Solving challenging grid puzzles with answer set programming", Dec. 2008.

[5] F. Glover, M. Laguna, and R. Marti, "Fundamentals of scatter search and path relinking", *Control and Cybernetics*, vol. 29, Jan. 2000.

[6] P. Hansen, N. Mladenović, J. Brimberg, and J. A. M. Pérez, "Variable neighborhood search", in *Handbook of Metaheuristics*, M. Gendreau and J.-Y. Potvin, Eds. Cham: Springer International Publishing, 2019, pp. 57–97, ISBN: 978-3-319-91086-4. DOI: 10.1007/978-3-319-91086-4_3. [Online]. Available: https://doi.org/10.1007/978-3-319-91086-4_3.

[7] O. Janko, *Home*. [Online]. Available: https://www.janko.at/Raetsel/Nurikabe/.

Optimizing Neural Network Architecture Through a Coarse Genetic Algorithm*

Logan Mallory and Aise Sevkli
Denison University
Granville, OH 43023
{mallor_l1, sevklia}@denison.edu

Abstract

The utilization of neural networks has increased dramatically over the past decade, and yet our understanding of the effects of different network characteristics on efficacy is limited. Network architecture plays a significant role in their loss, but the optimal topology is generally unknown and infeasible to calculate. This research implements a novel Genetic Algorithm to stochastically explore the search space and evolve towards a near-optimal network architecture while varying multiple parameters simultaneously. Using this meta-heuristic we are able to quickly and coarsely explore the search space to find an near-optimal network architecture for the Fashion-MNIST dataset. We demonstrate convergence from 10% to 82% validation accuracy within ten generations, identify influential algorithm parameters, and show the potential for test accuracy comparable to existing published results.

1 Introduction

1.1 Motivation

There has been explosive growth in interest and applications of neural networks over the past decade. Companies and researchers in almost every field utilize neural networks for forecasting, image recognition, natural language processing, fraud detection, self driving cars, and countless other topics. The attractiveness of neural networks stems from their ability to model non-linear relationships

between variables, and their surprising accuracy. The underlying architecture of neural networks can have a significant effect on this accuracy. However, the optimal structure of a neural network (i.e. number of hidden layers, nodes, activation functions, etc.) for a given task is not known ahead of time, and often never is, given the complex and infeasible optimization involved.

Further, existing methods for optimizing network architectures rely on extensive training and evaluation of each candidate network. This strategy is effective, but requires large amounts of time to run. This necessitates an implementation of a faster, more broad optimization algorithm to explore the relationship between the number of epochs each candidate is trained for, and the eventual near-optimal architecture. This research aims to apply the metaheuristic of Genetic Algorithms to address this problem and improve upon a naive neural network structure by evolving the number of layers, the number of nodes in each layer, and the activation functions of each layer.

1.2 Problem Definition

Multi-Layer-Perceptrons are a type of neural network in which connections exist between every node in sequential layers. This is in contrast to more state-of-the-art neural networks such as Resnet50, which allow for connections between individual nodes in any layer, representing the architecture as a graph structure (He et. al., 2015).

For a Multi-Layer-Perceptron neural network with L layers, $n_h^{[l]}$ hidden nodes and activation function $g^{[l]}$ in the l^{th} layer, there are $\sum_l^{L-1} n_h^{[l]} * n_h^{[l+1]}$ connections in the network. For each hidden node j in the l^{th} layer in the network, the activation value $a_j^{[l]}$ is determined by the activation function $g^{[l]}$ and the associated weighted connections. These terms [L, $n_h^{[l]}$, $g^{[l]}$] are the parameters we are aiming to evolve through iterative generations.

The networks will be trained and validated on the Fashion MNIST dataset, which has become a popular benchmarking dataset for new machine learning algorithms (Fashion MNIST Github, 2019). As the Genetic Algorithm implemented in this paper does not deal with network weights, the neural networks' weights will be optimized by Tensorflow using a stochastic gradient descent algorithm *Adam*. The fitness function for the genetic algorithm will be the validation of the neural network on a new, unseen set of test data, also referred to as the validation set.

While one would hope to compare the architecture of the resulting networks to the global optimum architecture, there are an enormous amount of possible network architectures, and this search space increases exponentially with the number of activation functions and maximum number of nodes. But by evolving the underlying architecture parameters of neural networks, we hope

to converge towards an architecture that has the highest validation accuracy, which implies that the neural network would also perform well on other new data (test accuracy).

1.3 Background

Representing a neural network architecture as a chromosome is complicated because a network architecture has a somewhat hierarchical structure. The layers in a network, and their respective number of nodes, determine how many connections there will be in the network, and equivalently how many weights. This structure can be represented as a string of numbers, where each number is the number of nodes in that layer. But when we consider that we are also optimizing the number of layers and the activation function for each layer, more bits of information have to be added for each layer, necessitating more complex genetic representations.

Previous research has classified genetic representations into two categories known as *Strong* encodings and *Weak* encodings (Hancock, 1997). *Strong* encodings are a direct mapping of connections and layers in the network, while *Weak* encodings are more similar to a set of rules defining the construction of phenotypes (Stanley et. al., 2002). It has also been noted that using *Weak* encoding can introduce an unknown bias into the network evolution, as the rules for growth are based on our current understanding of neural networks and neuroscience (Stanley et. al., 2002). There are several popular types of neural network encoding, most of which encode both the architecture and the node connections.

Binary Encoding A form of *Strong* encoding, Binary encoding represents neural networks as bit-maps, where each bit can represent a connection between two nodes in the network. Downsides however are that it has been shown to result in less efficient crossover methods while also placing a limit on the number of nodes allowed in the network (Dasgupta et. al., 1994).

Graph Encoding Encoding neural networks as graphs is another form of *Strong* encoding which represents network segments as sub-graphs which can then be interchanged during crossover. (Pujol et. al., 1998).

Non-mating Avoiding crossover altogether, Non-Mating is a *Strong* encoding method that store network genetic strings in a way that doesn't have to lend itself to crossover mechanics. This demonstrates that crossovers are not inherently necessary for effectively evolving neural networks (Yao and Liu, 1996).

Cellular Encoding A type of Weak encoding, it represents networks as graph structures, and sub-graphs as cells that have defined cellular division methods (Gruau, 1993). This method of encoding relates more closely to cellular division in nature, but requires a thorough understanding of both cellular division and neural network graph structures (Stanley et. al., 2002).

2 Proposed Algorithm

While drawing on past research into Genetic Algorithms, this proposed algorithm aims to be simpler and faster than previous implementations. Training for just one epoch enables a more coarse search of the feature space, and a specific architecture can be trained fully at the end to maximize its utility. This combination will hopefully enable further analysis of the specific architecture parameters that affect neural network accuracy, while also proving a useful tool for quickly exploring effective network architectures.

Specifically, we propose a simple genetic algorithm for evolving network topology which utilizes uniform crossover, repeated single point mutation, and elitist population selection to achieve efficient convergence towards an optimal architecture while minimizing population sizes and time to convergence.

2.1 Genetic Representation

The algorithm uses a type of *Strong* encoding to represent the networks as genetic strings, as the constraints on the network topology simplify the implementation of *Strong* encoding. There are a predefined number of input nodes n_{h_input}, output nodes n_{h_output}, maximum number of layers L_{max}, nodes per hidden layer n_{h_max}, and a list of activation functions g_{list} =[relu, sigmoid, linear, selu, softmax, tanh, softplus, softsign] (defined in Tensorflow 1.0).

Previous genetic representations for neural networks focused on defining the number of hidden layers, and the number of nodes in each hidden layer (Ritchie et al. 2003, Tian & Noor 2005). Our proposed genetic representation adds another dimension which captures the activation function of each hidden layer, which can play a significant role in the accuracy of the network as a whole.

It is important to distinguish here between the number of layers in a network, and the number of hidden layers. The number of layers will always be equal to the number of hidden layers plus two (input layer, output layer). L_{max} in this instance refers to the total number of layers, but it is important to note that the input and output layers can not be modified. For instance, if $L_{max} = 6$, $n_{h\_input} = 3$, $n_{h\_output} = 2$, $n_{h\_max} = 10$, and $g_{list} = [sigmoid, relu, tanh]$, the genetic string $[(2, 6), (1, 4), (3, 0), (2, 8)]$ represents a neural network seen in Fig. 1 with three hidden layers, with activation functions $g_{list[2]}$, $g_{list[1]}$, $g_{list[2]}$,

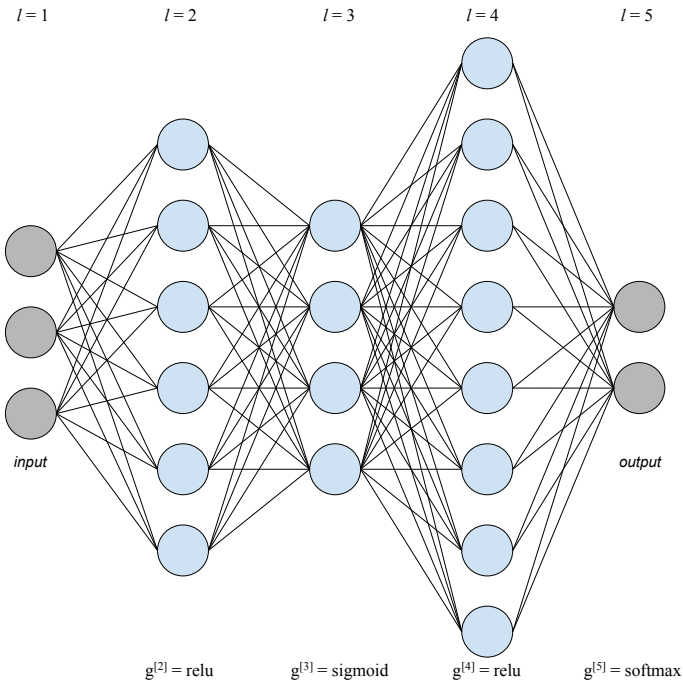


Figure 1: Neural Network with Genome [(2, 6), (1, 4), (3, 0), (2, 8)]

respectively. The layer with activation function three and zero nodes is left out of the network, as the absence of nodes means it has no effect on the network. Note that the tuples in the list are referred to as *operons*, which is a biological term for a unit of DNA encoding two or more linked genes.

2.1.1 Parameters

In running the algorithm, the parameters L_{max}, n_{h_max}, and g_{list} are set by the user, while the variables n_{h_input} and n_{h_output} are determined by the dimensions and attributes of the dataset being used. These parameters all control the phenotypes of the neural networks. The parameters for the Genetic Algorithm, including population size P_{size}, population mutation rate P_{mr}, population crossover rate P_{cr}, individual mutation rate I_{mr}, and individual activation function mutation rate I_{mr_af}, are user defined and affect the convergence of the algorithm overall.

2.2 Fitness Function

The dataset being used to train and evaluate the neural networks has to be randomly split into training data and validation data each new generation. A common pitfall of machine learning algorithms is a phenomenom known as "over-fitting", in which the algorithm (in this case the neural network) learns the weights that lead to the best accuracy on the data it was trained on, but fails to predict well for new, unseen data, because the weights were too specific to the training data. By randomly splitting the dataset each generation into a training and validation set, the fitness of the networks is evaluated as how well they perform on new, unseen data. This prevents the propagation of networks that over-fit the training data.

2.3 Gaussian Mutation

There are several popular methods for mutating continuous variables in Genetic Algorithms. Drawing on the success of Ding et. al. in using Gaussian Mutation to evolve a network to more accurately classify plants in the University of California Irvine Iris dataset, we chose to utilize Gaussian Mutation as the sole mutation method in our Genetic Algorithm (2011).

Gaussian Mutation introduces a normally distributed amount of mutation into the population. By centering the Gaussian distribution at 1, with a standard deviation of 0.2, roughly 68% of the mutations are between 80% and 120% of the original value. Of the remaining 32% of mutations, 27% are mutations of 60% and 140% of the original variable, and the remaining 5% are between 40% and 140%. The resulting normal distribution of mutations allows for easier understanding and analysis of the effect of mutations on convergence.

2.4 Uniform Crossover

As the maximum number of layers a network can have is defined by the parameter L_{max}, the genetic strings of each network are of the same length. Networks with fewer layers simply have a gene encoding for zero nodes in certain layers, as discussed earlier in Genetic Representation. This property of the genetic encoding allows for intuitive and efficient crossover between genetic strings. Drawing on the common 50/50 genetic split between parents seen in nature, Uniform Crossover creates two children, in which the first child has one random half of each parent, and the second child has the remaining two halves of the parents' genetic strings, as demonstrated in Fig. 2

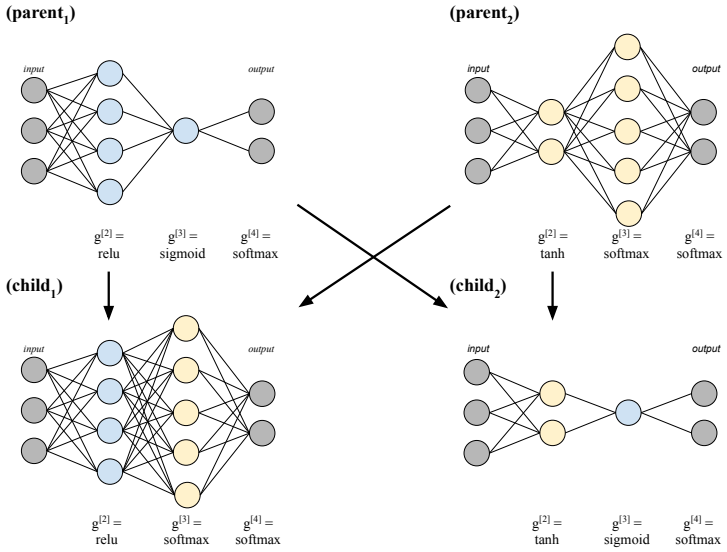


Figure 2: Example of Uniform Crossover - Two Parents, Two Children

3 Experimental Results

Table 1 gives a summary of the parameters used to run the genetic algorithm. To explore the effects of different algorithm parameters including population mutation rate P_{mr}, population crossover rate P_{cr}, and individual mutation rate I_{mr} the algorithm was run under three different settings (see Table. 1) referred to as $Sets$. Within each Set, the algorithm was run three times, and the results were averaged to produce the convergence results seen in Fig. 3. For all runs, the parameters L_{max}, P_{size}, I_{mr} and I_{af_mr} were kept constant after some initial exploration by the authors, but further research is needed to explore their influence on algorithm convergence.

Using a population mutation rate of 50% and a crossover rate of 100% in Set A, the Genetic Algorithm converges quickly towards the local optimum of 81% validation accuracy within ten generations (see Fig. 3. The average validation accuracy of each generation also increases consistently. Decreasing the population crossover rate to 50% in Set B yielded a mostly similar performance, but the average validation of the population did not increase at the same rate as Set A. Increasing both the population mutation rate and population crossover rate to 100% in Set C did appear to yield a faster convergence, but also result in a more unstable population, as the average and best validation accuracy fre-

Table 1: Algorithm Parameters

Set	L_{max}	n_{h_max}	P_{size}	P_{mr}	P_{cr}	I_{mr}	I_{af_mr}
A	18	512	10	0.5	1.0	0.5	0.25
B	18	512	10	0.5	0.5	0.5	0.25
C	18	512	10	1.0	1.0	0.5	0.25

Table 2: The Genetic Algorithm was run under three different settings to explore the influence of different parameters.

quently made large jumps higher and lower. Overall, the algorithm converges quickly, but plateaus around 82% validation accuracy.

Validation accuracy was measured within each generation, so the best and average metrics fluctuate from generation to generation. The global best validation accuracy is recorded but does not play a significant role in how the algorithm progresses.

4 Discussion

The consistently increasing validation accuracy of the networks in the Genetic Algorithm demonstrates that the population of candidate networks is not just randomly walking towards an optimal network architecture, but rather is benefiting from the genetic mechanisms of the Genetic Algorithm.

The best Multi-Layer-Perceptron test accuracy on *Fashion MNIST* is 87.7%, but it took roughly 15 minutes to train that network, likely training for multiple epochs (Fashion MNIST Benchmarks, 2019). The algorithm implemented in our research trained each network for only one epoch, but achieved 82% validation accuracy within 2 minutes. Using the best network architecture produced by the Genetic Algorithm and training the corresponding neural network for 50 epochs (5 minutes) resulted in a test accuracy of 88.8%. The fact that high validation accuracy architectures produced by the Genetic Algorithm are able to achieve higher test accuracy than existing published Multi-Layer-Perceptrons in a fraction of the time, suggests that although the validation accuracy of the networks in the algorithm are not as high as published results, they are good jumping off points to begin seriously training a network for more than a single epoch.

The variance in performance across Sets *A, B,* and *C* can be attributed partly to random chance, but mostly to the tuning of the Genetic Algorithm parameters P_{mr}, P_{cr}, and I_{mr}. As mentioned earlier, each Set was run three times and metrics were averaged to reduce the impact of randomness. Thus,

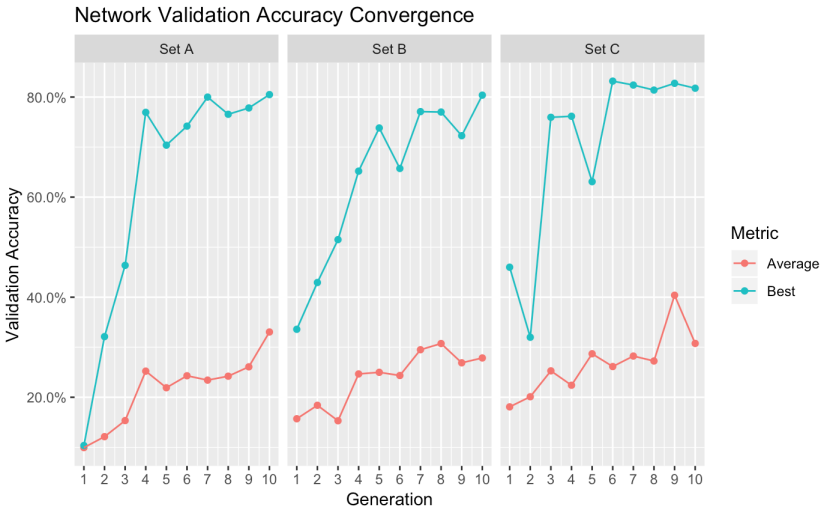


Figure 3: The best and average validation accuracy values of the GA population for 10 generations based on three different parameter setting named Set A, Set B and Set C.

the higher volatility seen in Set C's validation accuracy metrics is likely due to the fact that every generation, each candidate network in the population has 50% of their genes mutated; of which 75% will change the number of nodes in the layer, and the remaining 25% will change the activation function of the layer. And after this, all candidate networks breed and crossover their genes to produce a new generation of networks.

Drawing on current understandings of biological evolution, we would expect that populations with inherently high crossover rates and mutation rates will vary in *fitness* more than populations with less forced diversity. The question this raises is whether higher crossover and mutation rates will yield better validation accuracies more quickly because they are exploring the search space more wildly, or whether the higher parameters hinder convergence towards the optimal architecture.

Based on our results in Fig. 3, we see that Set C reaches 76% validation accuracy in three generations, while Sets A and B do not reach that threshold for at least four and seven generations, respectively. This suggests that in a constrained problem structure such as Multi-Layer-Perceptron Architectures, increased variability and diversity in populations lends itself to faster results by "bouncing" around the search space.

5 Conclusion

Existing methods for building neural network architectures rely on expert domain knowledge or complex optimization algorithms which can take days to run. Our research demonstrates that a simple Genetic Algorithm which modifies candidate networks' number of layers, number of nodes in each layer, and activation functions of each layer, can produce competitive results. In contrast to existing architecture optimization algorithms, our implementation trains each network for only one epoch, making a coarse exploration of the feature space. Our research modified the number of nodes in each layers by sampling from a Gaussian distribution, but other probability distributions may yield significantly different results. These results also highlight the potential for network accuracies comparable to other Multi-Layer-Perceptrons which were trained for much longer and many more epochs. Future research into the effect of different mutation and crossover algorithms is necessary to fill a gap in current literature.

References

[1] Fashion MNIST Benchmarks, `http://fashion-mnist.s3-website.eu-central-1.amazonaws.com`. Last accessed 8 Dec 2019

[2] Dasgupta, D. and McGregor, D., 1992, "Designing application-specific neural networks using the structured genetic algorithm", In Whitley, D. and Schaffer, J. D., editors, Proceedings of the International Conference on Combinations of Genetic Algorithms and Neural Networks, pages 87–96, IEEE Press, Piscataway, New Jersey.

[3] Ding, S., Su, C. & Yu, J. 2011, "An optimizing BP neural network algorithm based on genetic algorithm", Artif Intell Rev 36: 153.

[4] Fashion MNIST Github, `https://github.com/zalandoresearch/fashion-mnist`. Last accessed 8 Dec 2019

[5] Gruau, F., 1993, "Genetic synthesis of modular neural networks", In Forrest, S., editor, Proceedings of the Fifth International Conference on Genetic Algorithms, pages 318–325, Morgan Kaufmann, San Francisco, California.

[6] Hancock, P.J.B., 1997, "Genetic algorithms and permutation problems: a comparison of recombination operators for neural net structure specification", [Proceedings]

[7] He, K., Zhang, X., Ren, S., Sun, J. (2016). "Deep residual learning for image recognition". In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[8] Kenneth, O. Stanley and Miikkulainen, Risto, 2002, "Evolving Neural Networks through Augmenting Topologies", Evolutionary Computation, 10:2, 99-127

[9] Abbas Majdi, Morteza Beiki, 2010, "Evolving neural network using a genetic algorithm for predicting the deformation modulus of rock masses", International Journal of Rock Mechanics and Mining Sciences. Volume 47, Issue 2,, pg. 246-253

[10] Pujol, J. C. F. and Poli, R., 1998, "Evolving the topology and the weights of neural networks using a dual representation", Special Issue on Evolutionary Learning of the Applied Intelligence Journal, 8(1):73–84.

[11] Ritchie, M., White, B., Parker, J., Hahn, L., Moore, J., 2003, "Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases", BMC Bioinf., (28).

[12] Stanley, K., Miikuulainen, R., 2002, "Evolving Neural Networks Through Augmenting Topologies", Evolutionary Computation, 10(2):99-127.

[13] Suganuma, Masanori & Shirakawa, Shinichi & Nagao, Tomoharu, 2017, "A genetic programming approach to designing convolutional neural network architectures", pg. 497-504.

[14] Tian, L., and Noore, A., 2005, "Evolutionary neural network modeling for software cumulative failure time prediction", Reliability Engineering & System Safety, Volume 87, Issue 1, Pages 45-51

[15] Yao, X. and Liu, Y. 1996, "Towards designing artificial neural networks by evolution", Applied Mathematics and Computation, 91(1):83–90.

Algorithm 1 Evolve Network Architecture to Maximize Validation Accuracy

Initialize a population with P_{size} random networks
for $1 \rightarrow nGenerations$ **do**
    Split the training data into training and validation sets
    for Individual in Population **do**
        Train the Individual on the training data for one epoch
        Record fitness of Individual as the validation set accuracy
        Record the Individual's genome and validation accuracy to **Tracking** file
    for Individual in Population **do**
        if random() $< P_{mr}$ **then**
            for operon in Individual's genome **do**
                if random() $< I_{mr}$ **then**
                    if random() $< I_{af\_mr}$ **then**
                        Assign the operon a random activation function
                    else
                        Multiply the number of nodes by Gaussian[0,2)
    Sort the population in descending order based on fitness
    nChildren = $P_{cr} * P_{size}$
    for Each exclusive sequential Pair of parents **do**
        $child_1$, $child_2$ = uniform_crossover(Pair)
        Add children to new generation
        if length(new generation) $\geq$ nChildren **then**
            Break out of loop
    Add a random individual to the new generation for diversity
    Population = new generation
Return the best architecture in the **Tracking** file

Staging Human-computer Dialogs: An Application of the Futamura Projections*

Brandon M. Williams and Saverio Perugini†
†Department of Computer Science
University of Dayton
Dayton, Ohio 45469
BrandonWilliamsCS@gmail.com saverio@udayton.edu

Abstract

We demonstrate an application of the Futamura Projections to human-computer interaction, and particularly to staging human-computer dialogs. Specifically, by providing staging analogs to the classical Futamura Projections, we demonstrate that the Futamura Projections can be applied to the *staging* of human-computer dialogs in addition to the *execution* of programs.

1 Introduction

The *Futamura Projections* are a series of program signatures reported by [5] designed to create a program that generates compilers by repeated applications of a *partial evaluator* that iteratively abstracts away aspects of the program execution process. A partial evaluator transforms a program given any subset of its input to produce a version of the program that has been specialized to that input. We use the symbol `mix` from [7] to denote the partial evaluation operation because partial evaluation involves a `mix`ture of interpretation and code generation.

In this article, we introduce a model for staging human-computer dialogs based on The Futamura Projections. In other words, the Futamura Projections provide a way to generate programs that stage interactions between two participants engaged in a human-computer dialog for a variety of dialog representations. Table 1 is a legend mapping terms and symbols used in this article to their description. For a general introduction to the Futamura Projections, using the same diagramming conventions as this article, we refer the reader to [13].

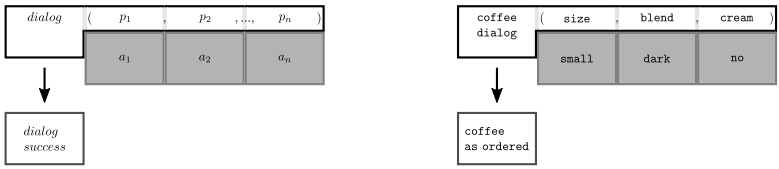
2 Staging Human-computer Dialogs

A *dialog*, for the purposes of this paper, is a series of interactions between a user and a computer system, which do not necessarily occur through a verbal modality. For instance, a user of an installation wizard for an application program participates in a human-computer dialog [4]. Additionally, dialogs in this paper are not specific to their responses, but represent an ordering of prompts and set of appropriate responses for a particular purpose of interaction. In this way, a dialog is analogous to a program: a definition of behavior with results that differ based on variable information provided as input. We can represent dialogs diagrammatically as seen in Fig. 1a or equationally as $[\![dialog]\!][a_1, a_2, ..., a_n] = [dialog\ success]$. For example, a dialog for ordering coffee may prompt a customer for the size and bean blend of the coffee, as well as whether or not to leave room for cream. Such an example instance is depicted in Fig. 1b and described equationally as $[\![\texttt{coffee dialog}]\!][\texttt{small}, \texttt{dark}, \texttt{no}] = [\texttt{coffee as ordered}]$.

Although dialogs are analogous to programs, a dialog is not a program and cannot execute on a computer. Instead, it must be *staged* with the assistance of a special program. A program that structures the interaction of a particular dialog is called a *stager* for that dialog. If a dialog is analogous to a program, a stager is analogous to a second program, semantically equivalent to the first,

Table 1: Legend of symbols and terms used in § 2 and §3.

Symbol	Description
dialog	A dialog between two parties.
p_n	A prompt for some input.
a_n	A response to a prompt.
dialog success	The result of a successful dialog.
DDSL	A domain-specific language designed for representing human-computer-dialogs.
DDSL compiler	A program that generates stagers when given a dialog specification.
dialog	An instance of a dialog specification DDSL.
stager	A program that stages a dialog.
DDSL interpreter	A program that stages a dialog given its specification and responses.
DDSL compiler generator	A program that creates DDSL compilers when given a DDSL interpreter.



(a) Dialog interaction.

(b) Instance of dialog interaction.

Figure 1: Dialog interaction and example instance.

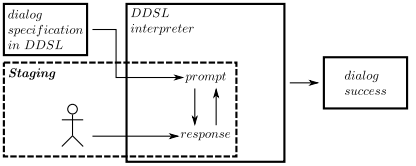
that has been implemented in a natively executable language. In other words, a stager is analogous to a compiled dialog. The details of staging human-computer dialogs, and particularly mixed-initiative dialogs [8], are given in [10].

2.1 Dialog DSL Interpretation

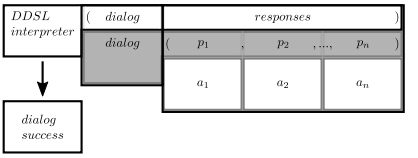
Just as high-level programming languages are suited to describing programs to humans, we can design a special domain-specific language (DSL) suited to describing dialogs to humans. As with programming languages, we can then write a dialog DSL (DDSL) interpreter that accepts a dialog specification in that language and stages it, a process detailed in Fig. 2a. Fig. 2b illustrates DDSL interpretation with our diagram conventions while the equational representation is $[\![DDSL\ interpreter]\!][dialog, responses] = [dialog\ success]$.

2.2 Dialog DSL Compilation

Compilers translate programs in a source language to equivalent programs in a target language. Analogously, we have created a program that takes dialog specifications in a DDSL and generates a stager for that dialog. We call this stager generator program a *DDSL compiler*. A DDSL compiler program is depicted in Fig. 3a and expressed equationally as $[\![DDSL\ compiler]\!][dialog] = [dialog\ stager]$. Although a dialog specification is not executable, Fig. 3b demonstrates the behavioral equivalence of the stager and the specification.

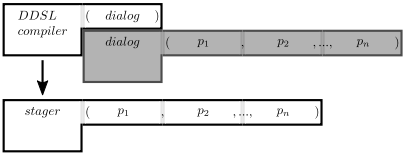


(a) The operation of a DDSL interpreter.

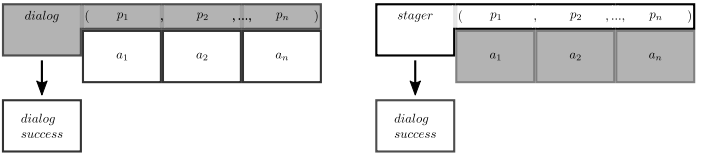


(b) The DDSL interpretation process.

Figure 2: Staging of dialogs with a DDSL interpreter.



(a) The DDSL compilation process.



(b) Comparison of DDSL compiler input and output.

Figure 3: Compilation of DDSL into a stager.

3 A Programming Model for Staging Dialogs

The Third Futamura Projection produces a program that, given an interpreter for *any* source programming language, can generate a compiler for that language. Furthermore, this interpreter need not be thought of exclusively in the traditional sense as a program interpreter. In particular, it can be *any* program that conforms to the interpretation signature by accepting two inputs, namely some source that describes a behavior and the input to that source, and enacting the behavior. Providing `mix` with a DDSL interpreter and dialog specifications yields programs meaningful to staging. Therefore, the Futamura Projections can be applied to the *staging* of human-computer dialogs in addition to the *execution* of programs, and this is the primary contribution of this paper. To facilitate the staging of dialogs, we provide staging analogs to the classical Futamura Projections.

3.1 First Futamura Staging Projection: Dialog DSL Compilation

Let us first look at the partial evaluation of a DDSL interpreter with a dialog specification as static input, as shown in Fig. 4a and expressed equationally as $[\![mix]\!][DDSL\ interpreter, dialog] = [stager]$. Here `mix` specializes the DDSL interpreter to the dialog, leaving the responses as dynamic input. The resulting program, shown in Fig. 4b and represented equationally as $[\![stager]\!][a_1, a_2, ..., a_n] = [dialog\ success]$, accepts the responses and completes the staging of the dialog. *In other words, the specialized program is a stager for the input dialog.* The partial evaluator has effectively generated a stager from a DDSL dialog specification. This transformation is analogous to the compilation of a source program into a target program in the First Futamura Projection pattern.

> **First Futamura Staging Projection**: A partial evaluator, with the aid of a DDSL interpreter, can generate stagers.

3.2 Second Futamura Staging Projection: DDSL Compiler Generation

We have applied the pattern of the First Futamura Projection to dialog staging to produce the First Futamura Staging Projection. Because the Second Futamura Projection is just a partial evaluation of the first, we can apply the pattern of the Second Futamura Projection to staging by partially evaluating the First Futamura Staging Projection. This Projection is shown in Fig. 5a and expressed as $[\![mix]\!][mix, DDSL\ interpreter] = [DDSL\ compiler]$, where `mix` itself is being partially evaluated with the DDSL interpreter as static input.

The resulting program, as with the result of the Second Futamura Projection, captures the behavior of the previous Projection. Fig. 5b (represented equationally as $[\![DDSL\ compiler]\!][dialog] = [dialog\ stager]$) demonstrates this by accepting a dialog specification and producing a stager for it. Notice that it has the same shape and labels as Fig. 4a. The diagrams are the same as those for the classical Futamura Projections, but with the source language interpreter replaced with a DDSL interpreter and the source and target programs replaced with a dialog specification and stager, respectively. Instead of abstracting the source program away from a `mix`-based program compilation process, we are abstracting the dialog specification away from a `mix`-based DDSL compilation process.

> **Second Futamura Staging Projection**: A partial evaluator, by making use of another instance of itself and a DDSL interpreter, can generate DDSL compilers.

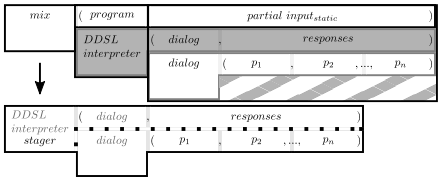
3.3 Third Futamura Staging Projection: Generation of DDSL Compiler Generators

Now we abstract the staging process one final degree by partially performing the Second Futamura Staging Projection while leaving the DDSL interpreter as dynamic input. Fig. 6a shows this third Projection, which is expressed equationally as $[\![mix]\!][mix, mix] = [DDSL\ compiler\ generator]$. Aside from the label that the result is a generator of *DDSL* compilers, there is nothing specific to dialogs or staging in this projection. *Because there is no mention of a generic input program, a dialog specification, or either variety of interpreter (i.e., program or staging), the Third Futamura Staging Projection is the same as the Third Futamura Projection.* However, when given different varieties of interpreter as input, the generator program serves different roles. The result of the Third Futamura Projection generates a traditional compiler when given a programming language interpreter, but in the context of the Futamura Staging Projections, it generates a DDSL compiler from a DDSL interpreter. The latter is depicted in Fig. 6b and expressed equationally as $[\![DDSL\ compiler\ generator]\!][DDSL\ interpreter] = [DDSL\ compiler]$.

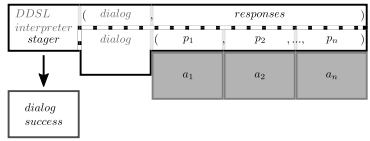
> **Third Futamura Staging Projection**: A partial evaluator, by making use of two additional instances of itself, can generate a program that generates DDSL compilers.

3.4 Summary: Futamura Staging Projections

Each human-computer dialog Staging Projection, akin to each Futamura Projection, abstracts away a previous process by partial evaluation. The First

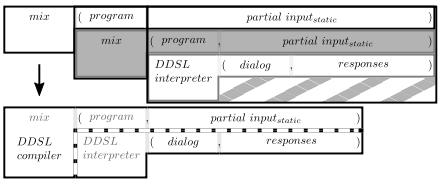


(a) The First Futamura Staging Projection.

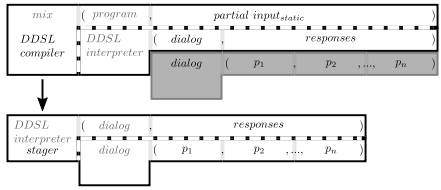


(b) The output of the First Futamura Staging Projection.

Figure 4: The First Futamura Staging Projection.

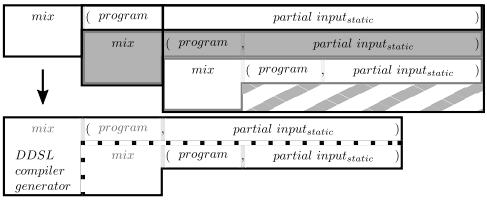


(a) The Second Futamura Staging Projection.

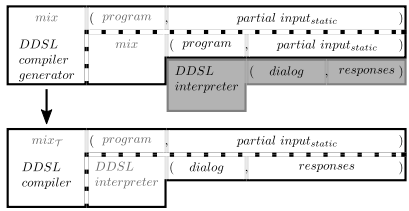


(b) The output of the Second Futamura Staging Projection.

Figure 5: The Second Futamura Staging Projection.



(a) The Third Futamura Staging Projection.



(b) The output of the Third Futamura Staging Projection.

Figure 6: The Third Futamura Staging Projection.

Table 2: Juxtaposition of related equations and diagrams from § 2 and § 3. **Legend:** O/P = Output.

Fig.	Equational Notation	Input Description	Output Description	O/P Fig.
1a	$[\![dialog]\!][a_1, a_2, ..., a_n] = [dialog\ success]$	A list of responses for the dialog.	The results of a successfully completed dialog.	N/A
2b	$[\![DDSL\ interpreter]\!][dialog, responses] = [dialog\ success]$	A dialog specification in the DDSL along with its responses.	The results of a successfully completed dialog.	N/A
3a	$[\![DDSL\ compiler]\!][dialog] = [dialog\ stager]$	A dialog specification in the DDSL.	A program that stages the dialog.	3b
4a	$[\![mix]\!][DDSL\ interpreter, dialog] = [stager]$	An interpreter for the DDSL and a dialog specification in the DDSL.	A program that stages the dialog.	4b
4b	$[\![stager]\!][a_1, a_2, ..., a_n] = [dialog\ success]$	A list of responses for the dialog.	The results of a successfully completed dialog.	N/A
5a	$[\![mix]\!][mix, DDSL\ interpreter] = [DDSL\ compiler]$	*mix* and an interpreter for the DDSL.	A DDSL compiler.	5b
5b	$[\![DDSL\ compiler]\!][dialog] = [dialog\ stager]$	A dialog specification in the DDSL.	A program that stages the dialog.	4b
6a	$[\![mix]\!][mix, mix] = [DDSL\ compiler\ generator]$	Two instances of *mix*.	A program that generates DDSL compiler	6b
6b	$[\![DDSL\ compiler\ generator]\!][DDSL\ interpreter] = [DDSL\ compiler]$	An interpreter for the DDSL.	A DDSL compiler.	5b

Table 3: Summary of Futamura Staging Projections. **Legend:** Prj = Projection, O/P = Output.

Prj	Description	Equational Notation	Fig.	O/P Fig.
1	`mix` can generate stagers.	$[\![mix]\!][DDSL\ interpreter, dialog] = [dialog\ stager]$	4a	4b
2	`mix` can generate DDSL compilers.	$[\![mix]\!][mix, DDSL\ interpreter] = [DDSL\ compiler]$	5a	5b
3	`mix` can generate a program that generates DDSL compilers.	$[\![mix]\!][mix, mix] = [DDSL\ compiler\ generator]$	6a	6b

Futamura Staging Projection treats the dialog responses as dynamic input. The Second Staging Projection furthers the abstraction, this time with the dialog specification. The Third Staging Projection allows the DSL used for expressing the dialog to vary by treating the DDSL interpreter as dynamic input. The Futamura Staging Projections extend the program-to-dialog analogy when each source-program-specific element is replaced by its dialog staging analog. In other words, replacing the source program with a dialog specification and the interpreter with a DDSL interpreter in turn replaces the target program with a stager and the compiler with a DDSL compiler. As mentioned earlier, the program compiler generator and the DDSL compiler generator (i.e., the results of third of each series of projections) are not just analogous, but identical. Table 2 juxtaposes the related equations and diagrams from both § 2 and § 3 in each row to make their relationships more explicit. Each row of Table 3 succinctly summarizes each Staging Projection by associating each side of its equational representation with the corresponding figure from § 3.

4 Related Research

Partial evaluation has been utilized to stage mixed-initiative dialogs [3, 10, 11]. The Futamura Staging Projections generalize this use across different DDSLs. Buck and Perugini have developed such a DDSL and a stager [1, 10]. Dialog

representations can be mined from detailed, high-level dialog specifications [9]. Research is also being conducted to make a staged dialog more conversational, which includes analysis of rich, dynamic data [12] as well as the improvement of mixed-initiative "personal assistants" [2]. The user side of the dialog interaction often needs to be simulated as well, either for testing or purposes of training in AI. One approach to such simulation is to use neural networks to build a realistic flow of user behaviors [6].

5 Conclusion

We applied the Futamura Projections to the staging of human-computer dialogs. Partial evaluation, through the Futamura Staging Projections, can be used to generate stagers, DDSL compilers, and programs that themselves generate DDSL compilers. Although the scope of the Futamura Projections has been largely limited to the programming languages research community, we are optimistic that this article has provided a programming model for using the Projections in staging and made a case for their role in building powerful programming abstractions in human-computer interaction.

References

[1] J.W. Buck and S. Perugini. A tool for staging mixed-initiative dialogs. In P.H. Phung, J. Shen, and M. Glass, editors, *Proceedings of the Twenty-seventh Modern Artificial Intelligence and Cognitive Science Conference (MAICS)*, pages 25–32, 2016.

[2] J.W. Buck, S. Perugini, and T.V. Nguyen. Natural language, mixed-initiative personal assistant agents. In *Proceedings of the 12$^{\text{th}}$ International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pages 82:1–82:8, New York, NY, 2018. ACM Press.

[3] R. Capra, M. Narayan, S. Perugini, N. Ramakrishnan, and M.A. Pérez-Quiñones. The staging transformation approach to mixing initiative. In G. Tecuci, editor, *Working Notes of the IJCAI 2003 Workshop on Mixed-Initiative Intelligent Systems*, pages 23–29, Menlo Park, CA, 2003. AAAI/MIT Press.

[4] A. Dix, J. Finlay, G.D. Abowd, and R. Beale. *Human-Computer Interaction*, chapter 16: Dialog Notations and Design. Prentice Hall, Harlow, England, third edition, 2010.

[5] Y. Futamura. Partial evaluation of computation process: An approach to a compiler-compiler. *Higher-Order and Symbolic Computation*, 12(4):381–391, 1999.

[6] I. Gur, D. Hakkani-Tür, G. Tür, and P. Shah. User modeling for task oriented dialogues. Technical Report arXiv: 1811.04369 [cs.CL], Cornell University Library: Computing Research Repository (CoRR), 2018. Available from `http://arxiv.org/abs/1811.04369`.

[7] N.D. Jones. An introduction to partial evaluation. *ACM Computing Surveys*, 28(3):480–503, 1996.

[8] D.G. Novick and S. Sutton. What is mixed-initiative interaction? In S. Haller and S. McRoy, editors, *AAAI Spring Symposium on Computational Models for Mixed Initiative Interaction*, number SS-97-04, pages 114–116. AAAI Press, Menlo Park, CA, 1997.

[9] S. Perugini. Mining mixed-initiative dialogs. In S.-F. Su, editor, *Proceedings of the IEEE International conference on Systems, Man, and Cybernetics (SMC)*, pages 2287–2294, Los Alamitos, CA, 2016. IEEE Computer Society Press.

[10] S. Perugini and J.W. Buck. A language-based model for specifying and staging mixed-initiative dialogs. In José Creissac Campos and A. Schmidt, editors, *Proceedings of the Eighth International ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS)*, pages 204–216, New York, NY, 2016. ACM Press.

[11] N. Ramakrishnan, R. Capra, and M.A. Pérez-Quiñones. Mixed-initiative interaction = mixed computation.

[12] M. Walker, A. Smither, S. Oraby, V. Harrison, and H. Shemtov. Exploring conversational language generation for rich content about hotels. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*. European Language Resources Association (ELRA), 2018.

[13] B.M. Williams and S. Perugini. Revisiting the Futamura projections: A diagrammatic approach. *Theoretical and Applied Informatics*, 28(4):15–32, 2016. DOI: `10.20904/284015`.

An Object-Oriented Media Computation Package in Python 3*

Paul Buis
Computer Science Department
Ball State University
Muncie, IN 47306
00pebuis@bsu.edu

Abstract

This paper demonstrates a translation of the library used in Guzdial's JES from Jython into Python 3 using the dot notation syntax that is favored in many object-oriented languages. It also demonstrates additional methods for image manipulation that allow the introduction of the functional programming paradigm at an introductory level in a natural seeming way.

Introduction

At Ball State University we have a single introductory programming course [8] that uses a contextualized [3] approach based on media computation [5] with sessions that include peer instruction [12] and sessions that include pair programming [9]. The design was chosen as the result of research showing that these three components work well together [13]. The course serves both majors [15] and non-majors [4] and uses the textbook Introduction to Computing and Programming in Python [7]. It's design is also is intended to increase the number of women [14] and other underrepresented groups majoring in Computer Science at Ball State.

The textbook uses a simple IDE paired with a library of Python functions called JES: The Jython Environment for Students which has been open sourced by its authors. Jython is a Python implementation written in Java that conforms to Python 2 syntax. The library of Python functions in JES invokes code written in Java that accompanied an earlier textbook [6] by the same authors that had similar examples coded in Java. The contract faculty who had been typically assigned to teach this course became vocal in their opposition to the

continued use of Jython in the course because Python 2 is obsolete, because the JES library doesn't used object- oriented dot syntax, and because they wanted to supplement the examples in the text with examples using some of the many CPython libraries for using Python 3 syntax. This opposition has made it difficult to staff the course.

I was initially puzzled by this opposition since it is a non-objective of the course to teach Python. We want to provide a programming experience that prepares students for the next course which deals with object-oriented programming in Java but had no other courses where the Computer Science Department expected Python to be used. Jython is as good as any other language for introducing students to branching, iteration, functions, and simple sequences like arrays or lists. My mind was changed when one of the contract faculty pointed out that they were using Python 3 in both our freshman level Discrete Structures course and our sophomore level Algorithms course. Historically, neither of these courses involved any actual coding or student programming courses. The new practice of showing students actual code and having them complete assignments where they demonstrated their understanding of the material by writing programs was having significant positive impact by engaging the students in courses they had previously considered boring and irrelevant.

However, the design of the course was tightly connected to the contextualization with media computing and other faculty were very fond of the textbook and its examples. Hence, I was motivated to create a work-alike library in pure Python 3 that would allow students and instructors to run the code from the textbook with minimal changes or alternatively provide the same functionality using object-oriented dot notation. The underlying media computation library in JES was written in Java and its object-oriented nature seemed obvious to me even wihout dot notation. So, I set out to use the same strategy as the textbook author: write object-oriented code and then provide wrapper functions that mimicked the JES library without dot notation.

The decision was made to use IPython [10] in a Jupyter [2, 11] environment and limit the package to only import Python packages that were preinstalled by default by the Anaconda [1] Python distribution in both Windows and MacOS environments. IPython is an interaction layer over the standard CPython replacing its purely textual interface with one that handles interaction with non-textual data such as sounds and images. Jupyter uses an IPython interpreter in a "kernel" on one side of an HTTP connection and a standard web browser on the other. Jupyter kernels (available for many programming languages other than Python as well) use web sockets over an HTTP connection. Students can use the Anaconda Python distribution to install the whole Jupyter system on their own machines.

Translation of Textbook Code to Dot Notation

Below is an example function defining a picture transformation to produce a sunset effect by reducing the intensity of the blue and green components of each pixel, making the image less bright and more reddish. This is similar essentially similar to one of the code examples from the textbook.

```
from MediaComp.pictures import *
pic = Picture.from_file("media/beach.jpg")
pic
```

```
def makeSunset(picture):
  copy = picture.copy()
  for j in range(0, picture.height):
    for i in range(0, picture.width):
      pixel = copy[i,j]
      pixel.blue = pixel.blue * 0.7
      pixel.green = pixel.green * 0.7
  return copy

make_sunset_pic = makeSunset(pic)
make_sunset_pic
```

Guzdial's JES library let the original Java code's setters and getters show through. The Jython makeSunset() provided by Guzdial to accompany JES looked like:

```
def makeSunset(picture):
  for p in getPixels(picture):
    value = getBlue(p)
    setBlue(p, value * 0.7)
    value = getGreen(p)
    setGreen(p, value * 0.7)
```

Since the color transformation of each pixel is independent of its location within the picture, this can be expressed a bit more simply as (note: this code also includes type annotations).

```
def makeSunset2(picture: Picture) -> Picture:
  copy: Picture = picture.copy()
  for pixel: Pixel in copy:
    pixel.blue = pixel.blue * 0.7
    pixel.green = pixel.green * 0.7
  return copy
```

If this refactoring of the code were expressed in terms of Guzdial's JES library, it would look like:

```
def makeSunset2(picture):
  for pixel: Pixel in picture:
    setBlue(p, getBlue(p) * 0.7)
    setGreen(p, getGreen(p) * 0.7)
```

However, the textbook code never nests function invocations, since Freshman-level students typically have a harder time understanding the sequencing involved.

Taking a Functional Programming Approach

The course for which this software was being developed has a mid-semester project where students create a collage by using a small number of picture files, performing what they consider to be artistically pleasing image transformations on them and then displaying a combined image. Both the resulting image and the code used to produce it are displayed for judging in separate categories. Observing student produced code, one sees data flow similar to that shown above. When performing multiple transformations on an image, the students usually produced ever more complex loop bodies. To an instructor responsible for teaching functional programming to upper division students, this was disturbing.

Taking a functional approach would mean finding analogs to the traditional map, filter, and reduce operations used in producing composable collection

operations. The kinds of operation illustrated in the previous example is clearly analogous to a map operation where the color of each pixel is mapped to a new color, resulting in code such as:

```
def sunset(color):
  return Color(color.red, color.green*0.7, color.blue*0.7)

sunset_pic = pic.map(sunset)
```

This abstracts away the loop construct, which perhaps was the learning objective in the introductory course. The result, however, is code that has lower code complexity by any reasonable metric.

A slightly less direct analogy to the traditional map operation performs a reordering of the pixels in the original picture by including the pixel coordinates in the data being transformed. In order to avoid introducing additional data types, it relies on the linguistic construct of tuples. This operation was named "remap" and the code below illustrates its use.

```
def sunset2(point_rgb):
  ((x, y), (r, g, b)) = point_rgb
  return ((x, y), (r, g * 0.7, b * 0.7))

sunset_pic2 = pic.remap(sunset2)
```

This example is a direct translation of the previous examples. To illustrate the reordering capabilities that the remap operation adds, see the following example which produces a left to right "flip" of the image.

```
def flip_left_right(point_rgb):
  ((x, y), rgb) = point_rgb
  return ((-x-1, y), rgb)

flipped_pic = pic.remap(flip_left_right)
```

The "trick" being used here is that in order to assure that the transformed coordinates are in the appropriate range, the remap operation performs a modulo operation them before placing the resulting RGB color in the appropriate location in the new picture.

To achieve the same result, without taking advantage of this trick, it would be tempting to hard code the width of the image into the subtraction. Of course, that would be horrible style. Instead, we can take advantage of functional programming and use a closure, as shown here:

```
def left_right_flip(width):
  def flip(point_rgb):
    ((x, y), rgb) = point_rgb
    return ((width-x, y), rgb)
  return flip

flipped_pic2 = pic.remap(left_right_flip(pic.width))
```

Compositional Pipelines

Rather than creating more complex image transformations with for complex transformation functions, the functional paradigm favors functional composition. So, for example, while we could combine the sunset and flip effects with a single function such as:

```
def flip_sunset(point_rgb):
  ((x, y), (r, g, b) = point_rgb
  return ((-x-1, y), (r, g * 0.7, b * 0.7))
```

We could also perform them as sequential operations with code such as:

```
sunset_pic2 = pic.remap(sunset2)
flipped_sunset_pic = sunset_pic2.remap(flip_left_right)
```

Or, rather than invoking the remap method twice and forming an intermediate image, one can compose the sunset2 and flip_left_right functions and apply the remap method once, a very "functional" approach:

```
flip_sunset = compose(flip_left_right, sunset2)
flipped_sunset_pic = sunset_pic2.remap(flip_sunset)
```

Conclusion

In this paper we have demonstrated a translation the library used in JES into Python 3 using the dot notation syntax that is favored in many object-oriented languages. We believe this translation will be useful in other introductory courses that use the media computation approach. It enables the same language variant to be used throughout a course that also uses standard Python libraries for parts of the course which do not focus on media computation. This paper has also demonstrated additional methods for image manipulation that allow the introduction of the functional programming paradigm at an introductory level in a way that seems natural.

See GitHub repository at `http://github.com/paulbuis/MediaComp/`

References

[1] Anaconda Python/R Distribution - Free Download. `https://www.anaconda.com/distribution/` [Accessed 1 April 2020].

[2] Project Jupyter | Home. `https://jupyter.org/` [Accessed 1 April 2020].

[3] Steve Cooper and Steve Cunningham. Teaching computer science in context. *Acm Inroads*, 1(1):5–8, 2010.

[4] Mark Guzdial. A media computation course for non-majors. In *Proceedings of the 8th annual conference on Innovation and technology in computer science education*, pages 104–108, 2003.

[5] Mark Guzdial. Exploring hypotheses about media computation. In *Proceedings of the ninth annual international ACM conference on International computing education research*, pages 19–26, 2013.

[6] Mark Guzdial and Barbara Ericson. *Introduction to computing & programming in Java: a multimedia approach*. Pearson Prentice Hall, 2007.

[7] Mark Guzdial and Barbara Ericson. *Introduction to computing and programming in Python*. Pearson, 2016.

[8] Timothy Huang and Amy Briggs. A unified approach to introductory computer science: can one size fit all? *ACM SIGCSE Bulletin*, 41(3):253–257, 2009.

[9] Nachiappan Nagappan, Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, and Suzanne Balik. Improving the CS1 experience with pair programming. *ACM SIGCSE Bulletin*, 35(1):359–362, 2003.

[10] Fernando Pérez and Brian E Granger. Ipython: a system for interactive scientific computing. *Computing in science & engineering*, 9(3):21–29, 2007.

[11] Jeffrey M Perkel. Why Jupyter is data scientists' computational notebook of choice. *Nature*, 563(7732):145–147, 2018.

[12] Leo Porter, Saturnino Garcia, John Glick, Andrew Matusiewicz, and Cynthia Taylor. Peer instruction in computer science at small liberal arts colleges. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, pages 129–134, 2013.

[13] Leo Porter and Beth Simon. Retaining nearly one-third more majors with a trio of instructional best practices in CS1. In *Proceeding of the 44th ACM technical symposium on Computer science education*, pages 165–170, 2013.

[14] Lauren Rich, Heather Perry, and Mark Guzdial. A CS1 course designed to address interests of women. *ACM SIGCSE Bulletin*, 36(1):190–194, 2004.

[15] Beth Simon, Päivi Kinnunen, Leo Porter, and Dov Zazkis. Experience report: CS1 for majors with media computation. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, pages 214–218, 2010.

Learn to Build Cross-Platform Mobile Apps Using the Ionic Framework*

Conference Tutorial

Victor Norman
Department of Computer Science
Calvin University
Grand Rapids, Mi 49546
vtn2@calvin.edu

Description

In this workshop, participants will learn about the Ionic framework (`https://ionicframework.com`) that can be used to quickly develop native apps for both Android and iOS using one code base, written in Javascript and Typescript. Participants will work through one or more step-by-step tutorials that lead them through the steps to program, test, and deploy an app. The instructor will provide an introduction to Ionic, Angular, and Typescript, as well as assist participant as they work through the tutorials. Working in pairs will be encouraged.

The Ionic Framework is a mature framework for building websites and mobile apps, using one single code base. Ionic apps are developed and tested on a standard computer, and then compiled and deployed to mobile devices (or a web server). During compilation Ionic UI components are replaced with native UI components for Android or iOS, so that the resulting app uses native components that the user is familiar with.

Ionic is paired with either Angular, React, or Vue. In this tutorial, we will work with Ionic/Angular to develop a simple quiz application. The application will demonstrate the organization of Ionic/Angular, how to create and route to pages, how to use a service to read the data, and, if time permits, how to pull the data from an online NoSQL Firebase database. (Or, students may learn how to write unit and/or end-to-end tests for their application, or practice styling the application more extensively.)

The intended audience is post-secondary educators who have some knowledge of web development (HTML, CSS, and Javascript), and are interested in teaching app/web development. Experience with Angular is not necessary.

Participants must be able to install and configure libraries and applications on their computers. Participants will have the richest experience if they have a Android or iOS phone they can connect to their computer (Android for a Windows 10 computer, Android or iOS for a MacOS computer).

The schedule for the workshop is:

Introductions	5 minutes
Ionic Framework Introduction	15 minutes
Software Installation / Verification	10 minutes
Building a Quiz App using provided step-by-step tutorial materials	60 minutes

Biography

Dr. Victor Norman is an associate professor of Computer Science at Calvin University. He teaches introductory programming, data structures, operating systems, networking, web development, and systems administration courses. During his sabbatical in 2018-2019, he worked in a team at a software consultancy firm building a large web application, and then worked in Germany building apps using Ionic.

Docker vs Vagrant – How I Use Docker and Vagrant Teach My Oracle Database Administration Class*

Conference Tutorial

Pak Kwan
Computer Science Department
Northern Kentucky University
Highland Heights, KY 41099
Kwanp1@nku.edu

Description

Docker is a set of platform as a service (PaaS) products that uses OS-level virtualization to deliver software in packages called containers [1]. Vagrant is an open-source software product for building and maintaining portable virtual software development environments. It tries to simplify the software configuration management of virtualizations in order to increase development productivity. Vagrant is written in the Ruby language, but its ecosystem supports development in a few languages [2]. In the past Spring, NKU has to close down because of COVID-19. In order to convert the database class, I have to move our VMWare base Lab into virtual environment that it can be ran on students' own pc or labtop. Docker and Vagrant based solution were considered. The workshop presenter has used Docker and Vagrant for courses in Linux server administration, Database administration, Network design ,Web server administration and Web application development. The workshop will have a quick introduction to Docker and Vagrant. Participants will learn how to install and Oracle database with both Docker and Vagrant technologies. The benefits of using Docker vs Vagrant on my Oracle administration class will be discussed. The presenter and his assistance(s) will be available to help the participants with the installation of the needed software for the tutorial prior to the start of the tutorial.

Requirements

A laptop loaded with Oracle VirtualBox. The presenter will provide the VirtualBox's appliance.

Materials

Participants will be provided with codes, oracle virtulbox's applicance, a list of references and resources.

Length of the Tutorial

90 minutes

Presenter Biography

Pak Kwan currently is working as an Enterprise Architect at GE Aviation and adjunct faculty at Northern Kentucky University, previously worked for Fifth Third Bank and IBM. Current research Interests: Massive data processing technologies (Big Data), Parallel Database, MapReduce/Hadoop, Machine learning and learning theories.

References

[1] `https://en.wikipedia.org/wiki/Docker_(software)`.

[2] `https://en.wikipedia.org/wiki/Vagrant_(software)`.

Using CS Materials, a System to Align Your Courses With National Standards*

Conference Workshop

Erik Saule[1], Kalpathi Subramanian[1], Jamie Payton[2]
[1]Computer Science
UNC Charlotte, Charlotte, NC 28223
{esaule, krs}@uncc.edu
[2]Computer & Information Sciences.
Temple University, Philadelphia, PA 19122
payton@temple.edu

This workshop provides instructors with a hands-on introduction to *CS Materials* [1], a software infrastructure for aligning a course with other courses in the institution's curriculum, as well with national curriculum guidelines. The CS Materials workshop will provide hands-on experience to instructors to use the system to classify their own courses against accepted curriculum guidelines. Workshop attendees will will learn to input learning materials, perform coverage analysis, align their course with other courses, and search for new materials for use in their classes. The system is available at `https://cs-materials.herokuapp.com/`.

References

[1] Alec Goncharow, Anna Boekelheide, Matthew Mcquaigue, David Burlinson, Erik Saule, Kalpathi Subramanian, and Jamie Payton. Classifying pedagogical material to improve adoption of parallel and distributed computing topics. In *Proc. of IPDPSW 2019*, May 2019.

Micro:bit Magic: Engaging IoT & Embedded for CS1/2 and K-12*

Conference Workshop

Bill Siever
Computer Science and Engineering
Washington University in St. Louis
St. Louis, MO 63119
bsiever@gmail.com

Are you interested in a fun way to introduce a variety of students to significant contemporary CS topics, like wireless networking, robotics, and the Internet of Things (IoT)? Do you want to do so using a platform that is cheap, has a low barrier to entry, but where learning can translate to the real world and where advanced students can pursue advanced topics? If so, the micro:bit is for you! The micro:bit is a platform developed by the British Broadcasting Corporation (BBC) to encourage children to pursue computing and electronics. Although designed for children, it's well suited to a variety of post-secondary applications. It includes a 32-bit processor, lights, buttons, an accelerometer, digital I/O, and wireless communication, making it ideal for wearables and robotics. It also supports a variety of programming languages, from a novice-friendly block-based language, to JavaScript, Python, and C++.

This workshop will introduce the micro:bit and focus on engaging, lightweight coverage of complex topics, including networking, wearables, and IoT. Participants will work through classroom-ready exercises suitable for K-12 workshops, recruiting events, CS1/2, and introductory IoT courses. The workshop will also include some subjects not commonly covered in existing micro:bit material, like integration with mobile apps, IoT applications, and wireless communication. Participants can purchase hardware to follow along with activities, but many activities can be completed in a browser without hardware.

CS Bootcamp for K-5 Educators*

Conference Workshop

Cathy Bareiss
Mathematical & Engineering Sciences
The Bethel University, Mishawaka, IN 46545
cathy.bareiss@betheluniversity.edu

Description

This workshop is designed to prepare elementary teachers to incorporate the Indiana state Computer Science standards into their curriculum. No previous knowledge of computer science is required. The workshop will include a review of the K-5 Computer Science standards, including terminology and context. Attendees will cycle through four 90-minute workshops balancing review of relevant standards with content and ideas for implementation particularly in these four areas:

- Digital citizenship (including online safety) in today' world,
- Data representation,
- Programming,
- Sorting and searching.

Outcomes

- Teachers will gain a familiarity with K-5 Computer Science standards, including relevant terminology and context.
- Teachers will consider aspects of safe digital citizenship, and how to integrate concepts of cyber safety in the context of classroom and 1-1 use of technology.
- Teachers will be introduced to activities for teaching K-5 Computer Science standards related to data representation, programming, and sorting and searching algorithms.
- Teachers will workshop ideas for the integration of activities and practices that address the Computer Science standards.

Nifty Assignment: Digital Signature Creation and Verification Using Kryptos*

Nifty Assignments

Imad Al Saeed
Computer Science Department
Saint Xavier University
Chicago, IL 60655
alsaeed@sxu.edu

This assignment is for a course on Cybersecurity for students with no prior knowledge with Cryptography. The course instructor explains the Hash Function Cryptography concepts as solutions for ensuring the integrity of the message being sent. The course instructor explains to the students that the Hash functions are not encryption functions. The essential reason is that the original data cannot be recovered from the hash value. This is one of the main requirements of the hash functions as the hash value cannot be inverted. Finally, the course instructor explains how the hash value used to verify the digital signature of the message has been exchanged. This assignment requires suing a special software called Kryptos. The instructor provides the students with the following link to download the Kryptos software: `http://sourceforge.net/projects/kryptosproject/files/Kryptos/Kryptos%202.0/`. Students should download Kryptos version (2.0) as the newer version may not work on their system. In this assignment, each student chooses a partner, communicate with him/her, and send the file that is signed by them to his/her partner. Upon receiving the signed file, the student's partner should verify the signature on the file using Kryptos software. This process should be done in three steps:

Step 1: Key generation

A student works with his/her partner to generate public and private keys using Kryptos software. A student shares public keys only with their partner.

Step 2: Signature creation

The students run the Notepad program, create a text file (call it yourlastnameMessage.txt), and write a message in plaintext to your partner. The students use the Kryptos software to create a signature in this text file. Students then save the signed file as (call it yourlastnameSignature). After that, the students open the original text file with the Notepad software, make changes to the original message, save the changes in a new file called (call it yourlastnameMessage2.txt), and use Kryptos to create a signature in this file. Finally, the students send the four files they created (yourlastnameMessage.txt, yourlastnameMessage2.txt, and yourlastnameSignature) via email to their partner.

Step 3: Signature verification

The partner should download the four files and save them in the appropriate folder. After that their partner should verify the signatures on the files that he/she received using Kryptos software.

Visualization and Analysis for C Code Security*

Nifty Assignments

Steve Carr[1] and Jean Mayo[2]
[1]Western Michigan University
steve.carr@wmich.edu
[2]Michigan Technological University
jmayo@mtu.edu

The C language is commonly used for low-level programs like operating systems. When a student does not understand how their C program executes, they can easily introduce vulnerabilities into their code. We have developed the SecureCvisual system to help students learn how to develop more secure and robust C programs through a deeper understanding of the execution of their programs.

A visualization system takes input from dynamic analysis of a C program. The analysis is performed using Pin. A sequence of events is generated that is processed by the visualization system. Each event corresponds to the execution of one line of source code. A student can step forward or backwards through the event queue.

A program address space visualization depicts the values of registers and the program address space as the program executes. Buffer overflows and other memory errors are easily seen. An integer representation window identifies the underlying representation of an integer variable and the effect on the representation and decimal value of the variable after a conversion. An Expression Evaluation window shows the coercions that take place around each binary operation. A sensitive data visualization teaches students how to protect data so that it never appears unencrypted on secondary storage.

The tool is convenient for lecture. Multiple levels of detail and different perspectives on an execution make the tool useful in a variety of courses.

Using Game-Based Learning to Improve Student Outcomes in Computer Science*

Work-in-Progress

John Kaufeld
Purdue University Fort Wayne
john.kaufeld@pfw.edu

Over the last 15 to 20 years, game-based learning has received growing attention from researchers in many academic areas, including computer science. This research project builds on that work by looking at ways to apply tabletop and/or computer-based games to the computer science learning process.

Research will consider both hard and soft skills related to success in studying computer science and then identify, test, and evaluate games that can engage students to develop and practice those skills.

For example, the ability to break a problem down into accessible chunks is fundamental to system analysis and programming, but there are many ways to approach the process. Both Ricochet Robots (Z-Man Games) and Screeps (Steam) give students a framework to practice breaking problems into component parts and testing alternate solutions. Ricochet Robots does that at an abstract level through an analog tabletop game, while Screeps does it at a practical level with javascript programming in a persistent online world.

This is not simply gamification of existing approaches, but rather finding new ways to use games to teach and reinforce fundamental computer science concepts.

The project's goals are to identify games that will:

- help students connect more quickly and deeply with the computer science content they cover in class
- understand the material at a deeper level so they feel more confident applying it
- be easy for faculty to introduce and use in their classes.

The first stage of the project will focus on freshmen taking the two introductory computer science major-track courses at Purdue University Fort Wayne.

Future research will expand to non-majors taking the general education computer science exploration course.

This research project's overarching goal is to improve retention among computer science students and, in the second phase, increase the number of non-major students who become computer science majors or minors.

References

[1] Chi-Cheng Chang, Chaoyun Liang, Pao-Nan Chou, and Guan-You Lin. Is game-based learning better in flow experience and various types of cognitive load than non-game-based learning? perspective from multimedia and media richness. *Computers in Human Behavior*, 71:218–227, 2017.

[2] Patrick Felicia. *Game-based learning: Challenges and opportunities*. Cambridge Scholars Publishing, 2014.

[3] Chris Harris and Patricia Harris. *Teaching Programming Concepts Through Play*. 2015.

[4] S Tobias, JD Fletcher, and AP Wind. Game-based learning, handbook of research on educational communications and technology. *Springer*, 2014.

Improving Teaching and Learning of Computer Programming Through SL*

Work-in-Progress

Imad Al Saeed
Computer Science Department
Saint Xavier University
Chicago, IL 60655
alsaeed@sxu.edu

A good computer programmer should have a skill of identifying a problem, developing an effective algorithm to solve it, and convert that algorithm to a program using syntax and semantics previously learned. Learning programming could be a difficult process for many students. The difficulty could push large number of students to withdraw from their introductory programming courses while trying to grasp the basic programing concepts. The emergence of virtual worlds such as Second Life brings new opportunities for enhancing a programing skill for many students. This approach spurred students' motivation and captured their interest in learning programming. This study analyses how programming skills could be developed within the use of virtual world. This will be achieved by presenting observations on the difficulties that students encountered in the development of projects and activities and discusses and use the approaches taken to increasing programming skill using second life.