February 2021

Volume 12 Issue 2

JoCsE

Journal Of Computational Science Education

Promoting the Use of Computational Science Through Education

ISSN 2153-4136 (online)

Volume 12, Issue 2 **February 2021**

JOCSE

Journal Of Computational Science Education

The Journal Of Computational Science Education (JOCSE), ISSN 2153-4136, published in online form, is a supported publication of the Shodor Education Foundation Incorporated. Materials accepted by JOCSE will be hosted on the JOCSE website and will be catalogued by the Computational Science Education Reference Desk (CSERD) for inclusion in the National Science Digital Library (NSDL).

Subscription: JOCSE is a freely available online peer-reviewed publication which can be accessed at `http://jocse.org`.

Contents

Introduction to Volume 12 Issue 2: Special Issue on HPC Training and Education

Nitin Sukhija
Guest Editor
Slippery Rock University of Pennsylvania
Slippery Rock, PA

FOREWORD

High performance computing is becoming central for empowering scientific progress in the most fundamental research in various science and engineering, as well as societal, domains. It is remarkable to observe that the recent rapid advancements in today's and future computing and software environments provide both challenges and opportunities for cyberinfrastructure facilitators, trainers, and educators to develop, deliver, support, and prepare a diverse community of students and professionals for careers that utilize high performance computing to advance discovery. This special issue focuses on original research papers submitted to the Second Workshop on HPC Education and Training for Emerging Technologies (HETET20), which was held in conjunction with the ISC20 Digital conference, June 25, 2020; the third Workshop on Strategies for Enhancing HPC Education and Training (SEHET20), which was held in conjunction with the PEARC20 conference, July 27, 2020; and the Seventh SC Workshop on Best Practices for HPC Training and Education (BPHTE20), which was held in conjunction with the SC20 conference, November 11, 2020.

This special issue begins with an article by Purwanto et al. that presents a non-degree computational training program, DeapSECURE, that provides significant high-performance computing (HPC) and big-data foundations for cybersecurity students. The authors detail major improvements of the DeapSECURE lesson modules by grouping them into the "compute-intensive" and "data-intensive" categories, more tightly integrating the modules to streamline the learning experience. The assessment results of the cohort group trained indicate the need for further adjustments to improve learning experience and outcome. Moreover, the piloted workshop showed great promise to address some challenges encountered through the second year project.

The article by Chakravorty et al. reports on two educational approaches that were implemented in the informal program hosted by Texas A&M High Performance Research Computing (HPRC) in the Spring, Summer, and Fall semesters of 2020. The two approaches employed were: 1) traditional in-person sessions taught by the staff that focused on offering a lot of information online and 2) a primer approach involving a peer-learning environment engaging learners via shorter pop-up courses in which participants chose the topic matter and students taught and moderated the training sessions. These were supplemented with YouTube videos and continued engagement over a community Slack workspace. The authors conclude by highlighting the data collected as part of this study, indicating that the Primer format could be a suitable pedagogical approach that enhances learner engagement while scaling back on staff time.

The article by Dey et al. describes the efforts taken at the Texas Advanced Computing Center to develop a successful academic and training curriculum with the goal of making virtual classrooms more engaging, and more collaborative, thus delivering a better educational experience. The authors report on the approach to teaching with multiple instructors and integrating aspects of gamification, open curriculum, casual classroom, and flipped classroom along with spending more class time focused on applying learned concepts versus lecturing on concepts, resulting in much-needed teacher-student interaction to create a positive learning environment.

The article by Bungo and Wong describes the NVIDIA Deep Learning Institute (DLI) kits that offer a complete course solution to lower the barrier of incorporating AI and GPU computing in the classroom. The authors discuss the DLI University Ambassador Program that enables qualified educators to teach DLI workshops at no cost across campuses and academic conferences to faculty, students, and researchers. The authors conclude by illustrating real examples of leading academics leveraging Teaching Kits and Ambassador workshops in the classroom.

The article by Weeden describes the XSEDE EMPOWER (Expert Mentoring Producing Opportunities for Work, Education, and Research) program coordinated by the Shodor Education Foundation for the Extreme Science and Engineering Discovery Environment (XSEDE). The author discusses the goal of the program, which is to engage a diverse group of undergraduate students in the work of XSEDE, matching them with faculty and staff mentors who have projects that make use of XSEDE services and resources or that otherwise prepare students to use these types of services and resources. The author concludes by discussing the impact of the program on advancing careers and conference participation of the underrepresented undergraduate students.

The article by Backhaus et al. presents the challenges faced by the Pawsey Supercomputing Centre in making transition to virtual training, including how to creatively motivate and engage learners, build our virtual training delivery skills, and build communities across Australia. The authors detail the self-guided learning, using Nimbus cloud and containers for improving the training content, ensuring alignment with learning objectives and learning outcomes, and incorporating best practices in (virtual) learner interaction and engagement. The authors conclude by discussing that there is no universal, one-size-fits-all learning solution to address virtual training challenges and there exist various solutions and platforms that need to be carefully selected for different groups of learners.

Chen summarizes the development of a set of HPC courses to meet the needs of multidisciplinary students at the National University of Defense Technology. The courses emphasize both vertical understanding of HPC systems (parallel computer architecture, operating system/resource management system, compilation, library optimization) and the understanding of multiple HPC application areas.

The article by Bautista and Sukhija describes a new approach at National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (LBNL) patterned after the apprenticeship program within the High Performance Computing domain. This approach requires an intern or apprentice to fulfill milestones during their internship or apprenticeship timeframe, with constant evaluation, feedback, mentorship, and hands-on work that allows candidates to demonstrate their growing skill that will eventually lead to winning a career position. The authors conclude by reporting the positive outcomes of the program such as recruitment of quality talent, improved retention, and encouragement to individuals to further their education.

The article by Ma et al. details the infrastructure and the outcomes of Ask.cyberinfrastructure.org, which is a collaborative, crowd-sourced Q&A site specifically curated for the research computing community. The authors discuss various technologies employed to build the site and the Locales, which allow institutions to create subcategories on Ask.CI where they can experiment with posting institution-specific content and use of the site as a component of their user support strategy. The authors report on lessons learned, plans to foster outreach efforts to reach out to other communities and mailing lists to expand Ask.CI's presence, and to invite any suggestions/recommendations from the community.

The article by Colbry presents a newly developed course at the Department of Computational Mathematics Science and Engineering (CMSE) at Michigan State University (MSU) for teaching parallel programming to undergraduates. The author describes the flipped classroom model and a "hands-on" approach used in the "Methods in parallel Programming" (CMSE 401) course for learning with multiple real-world examples from a wide range of science and engineering problems. The author concludes by discussing the feedback and challenges reported by the students and plans to improve the course.

The article by Peoples describes the learning outcomes that are focused on the transferable skills intended to be gained because of the assessment design. The author discusses assessments which were disseminated to a cohort of students on a Master of Science degree in Professional Software Development at Ulster University, United Kingdom. This Master's degree is a conversion degree into Information Technology for students from non-IT backgrounds. The author report that the creative assessment design helped to bridge these gaps by exposing students to state-of-the-art technology on an international basis, helping them to understand the software developments which are essential in their support at the back-end, and encouraging the application of knowledge in new way.

The article by Chen et al. presents the design of a parallel computing course offered at the College of Meteorologic Oceanography at the National University of Defense Technology in China. The authors discuss the design of the course, focusing on addressing the scalability challenges presented by non-computer science majors who lack a background in fundamental computer architecture, systems, and algorithms. The authors also present a set of assignments and projects that leverage the Tianhe-2A supercomputer for testing. The authors conclude by reporting in the result of the present pre- and post-questionnaires to explore the effectiveness of the class design.

The article by Hosseini and Lucas describes a quantitative methodology, POP, for the assessment of parallel codes at the Performance Optimisation and Productivity (POP) Centre of Excellence, funded by the EU under the Horizon 2020 Research and Innovation Programme. The authors detail the POP methodology as a scalable performance analysis methodology based on a set of hierarchical metrics, where each metric represents a common cause of inefficiency in parallel applications based on a set of hierarchical metrics, where the metrics at the bottom of the hierarchy represent common causes of poor performance. The authors conclude by illustrating the advantages of employing the POP methodology by describing two real-world examples that employ the POP methodology to help HPC users understand performance bottlenecks of their code.

DeapSECURE Computational Training for Cybersecurity Students: Improvements, Mid-Stage Evaluation, and Lessons Learned

Wirawan Purwanto
Old Dominion University
Norfolk, VA
wpurwant@odu.edu

Yuming He
Old Dominion University
Norfolk, VA
yhe004@odu.edu

Jewel Ossom
Old Dominion University
Norfolk, VA
josso001@odu.edu

Qiao Zhang
Old Dominion University
Norfolk, VA
qzhan002@odu.edu

Liuwan Zhu
Old Dominion University
Norfolk, VA
lzhu001@odu.edu

Karina Arcaute
Old Dominion University
Norfolk, VA
karcaute@odu.edu

Masha Sosonkina
Old Dominion University
Norfolk, VA
msosonki@odu.edu

Hongyi Wu
Old Dominion University
Norfolk, VA
h1wu@odu.edu

ABSTRACT

DeapSECURE is a non-degree computational training program that provides a solid high-performance computing (HPC) and big-data foundation for cybersecurity students. DeapSECURE consists of six modules covering a broad spectrum of topics such as HPC platforms, big-data analytics, machine learning, privacy-preserving methods, and parallel programming. In the second year of this program, to improve the learning experience, we implemented a number of changes, such as grouping modules into two broad categories, "big-data" and "HPC"; creating a single cybersecurity storyline across the modules; and introducing post-workshop (optional) "hackshops". Two major goals of these changes are, firstly, to effectively engage students to maintain high interest and attendance in such a non-degree program, and, secondly, to increase knowledge and skill acquisition. To assess the program, and in particular the changes made in the second year, we evaluated and compared the execution and outcomes of the training in Year 1 and Year 2. The assessment data shows that the implemented changes have partially achieved our goals, while simultaneously providing indications where we can further improve. The development of a fully on-line training mode is planned for the next year, along with a reproducibility pilot study to broaden the subject domain from cybersecurity to other areas, such as computations with sensitive data.

KEYWORDS

Parallel computing, Big data, Machine learning, Cybersecurity, Non-degree training, Hands-on

https://doi.org/10.22369/issn.2153-4136/12/2/1

1 INTRODUCTION

State-of-the-art cybersecurity research is increasingly reliant upon advanced computing, also known as advanced cyberinfrastructure (CI) to strengthen cyber systems against attacks. This includes research areas such as penetration testing, intelligent intrusion detection, run-time malware detection, and secure and privacy-preserving machine learning. DeapSECURE (Data-Enabled Advanced Training Program for Cybersecurity Research and Education) is a non-degree computational training program that provides solid foundations in high-performance computing (HPC) and big data for cybersecurity students. DeapSECURE aims to complement the degree programs in cybersecurity, considering the ever-increasing scale of cybersecurity challenges. The goals, approach, and philosophy of the training program have been elaborated in our previous publication [18].

DeapSECURE consists of six modules covering a broad spectrum of topics such as the HPC platform ("HPC"), big-data analytics (BD), machine learning (ML) including neural networks (NN), privacy-preserving methods (CRYPT), and parallel programming (PAR) [18]. Each module is delivered as a three-hour workshop, combining a presentation on current cybersecurity research topics and basic introduction to the CI methods. DeapSECURE emphasizes hands-on experience in CI tools and frameworks as *applied* to solving cybersecurity research problems. Currently, the six modules consider topics such as: spam/phishing analysis, mobile device security, encryption (privacy protection), and hardware security. We built the detailed content and activities for the modules and delivered them as six workshops during the 2018–2019 academic year and a week-long summer institute in June 2019.

This paper is focused on the changes made in the second academic year (2019–2020) of the DeapSECURE *workshop series* program in order to improve the learning experience. This paper is organized as follows: In Section 2, we recap our experience of the first year of the workshop series (2018–2019) as well as the lessons

learned. Then, we detail in Section 3 the improvements to the training program implemented in the second year. Section 4 covers the assessment results and lessons learned from the second year of the training. In Section 5, we briefly cover the pilot online workshop conducted in Summer 2020 as an online virtual event during the time when all the educational activities were held virtually nationwide. We briefly outline our future direction in Section 6, then conclude in Section 7.

2 FIRST-YEAR RECAP

The DeapSECURE's six lesson modules were delivered as a series of six workshops during the 2018–2019 academic year (three workshops per semester). They were all offered again as a summer institute in June 2019. In this paper, we will focus primarily on our workshop series experience. Each workshop began with a 30-minute cybersecurity research presentation by an Old Dominion University (ODU) faculty, followed by an introduction of a CI technique, such as big data or machine learning, featuring rather extensive hands-on activities on ODU's Turing HPC cluster. Because the workshops would run during the school semesters, we decided to limit the length of each workshop to three hours. This time duration would give an opportunity for students to do the exercises during the workshop, while preventing a long-drawn session, which may discourage participation.

Starting already in the first year of the training program, we have been employing pre- and post-workshop surveys, focus groups, as well as our own observation to constantly evaluate and improve our workshops. As initially reported in our previous paper [18], our training received positive response from the students in the first year. The majority of the survey respondents were satisfied or very satisfied with the workshops, and many would recommend the training to others. Students considered the hands-on activities as the most valuable aspects of the workshop. Students were exposed to technologies, methodologies, software tools, and computational resources far beyond their regular coursework.

While our training yielded many positive outcomes, we saw much room for improvement, as evidenced by the challenges we encountered then. The first notable issue was that the attendance of the workshops faltered towards the end of the semester, when the regular coursework put an increasing demand on students' attention and time. For example, in the Fall 2018, the first workshop was attended by more than 30 students, but the last one was attended by 24, a decrease of $\tilde{2}5\%$. In Spring 2019, the workshops were consistently attended by 11–12 students, considerably lower than half of the preceding semester's attendance.

Each workshop considered its own cybersecurity research topic [18], which means that a sizable fraction of the workshop time had to be devoted to introducing a new cybersecurity topic, thereby reducing the amount of time available for the hands-on activities. Indeed, it is a challenge to design a training program, such as DeapSECURE, which aims to provide a broad yet sufficient introduction to advanced computing topics under the tight time constraints of a workshop format. To overcome the limited length of the time available, we developed a written-lesson website for each training module [17]. These websites are available publicly and can be used by learners to further their learning after the workshops.

Another challenge that we have observed is that learners had difficulty in effectively applying high-level concepts taught from either the CI methods or cybersecurity during the hands-on activities. We have determined that this problem stemmed from the mismatch between a rather low-level command-line interface that is used to access supercomputing resources and students' habit of interfacing with computers via graphical interfaces and plug-and-play environments. Hence, the learners fell behind in the hands-on exercises. To solve this problem in the following years, we have decided to resort to more high-level tools, such as Jupyter notebooks, minimize the set of command-line tools used, and select workshop participants that already have some coding skills.

3 SECOND-YEAR IMPROVEMENTS

In the second year of this program, we implemented a number of changes with the goal to improve the learning experience. Among the most significant changes are (a) grouping modules broadly into the "big-data" (data-intensive) and "HPC" (compute-intensive) categories; (b) providing more continuity across several modules by creating a single cybersecurity storyline spanning them; and (c) introducing an optional post-module "hackshop" to enhance the hands-on experience. The changes are expected to facilitate maintaining students' interest and attendance across the entire year of this non-degree program. To take into account semester course load, we shifted the workshop schedule towards the beginning of the semesters by having a workshop approximately every other week with hackshops conducted in-between. As elaborated later in this paper, we also began training teaching assistants to contribute to the development of the lesson materials.

3.1 Revised Workshop Schedule

We reordered the modules taught in the workshops, recognizing that they fall roughly under two categories:

1. **The compute-intensive category** (the HPC, CRYPT, and PAR modules): The key question for this category is how to deal with the computational complexity of cybersecurity problems that take a long time to compute. A common theme in these three modules is the need to split the computational workload across many worker-processes on a modern HPC cluster to greatly reduce the time to solution. Further consideration for high performance will be part of the PAR module.
2. **The data-intensive category** (the BD, ML, and NN modules): The key issue for this category is how to leverage "big data" to detect and defend against cyber threats. Moden computing technologies have generated and made use of enormous amounts of data. From the perspective of cybersecurity, big data can be a two-edged sword. One the one hand, data are assets that are frequently targeted in cyber attacks such as data breaches, denial of service, and botnets. On the other hand, leveraging the state-of-the-art, data-intensive techniques such as machine learning and deep learning has become an indispensable skill for cybersecurity professionals to stay ahead the increasing level of malice and sophistication used to evade detection and defense measures. The three modules in this category aim to introduce these techniques to cybersecurity students.

Table 1: The revised DeapSECURE modules for the 2019–2020 workshop series.

Module	Research Presentation, Presenter, Affiliation	Workshop Hands-on	Hackshop Hands-on	Toolkits
HPC	High Performance Computing and Cybercrime: "An Ounce of Prevention Is Worth a Pound of Cure" (Roderick Graham, Sociology and Criminal Justice)	Determining country of origin of a large collection of spam emails	Making an IP address scanner using UNIX tools	UNIX shell (bash)
CRYPT	Security and Privacy of AI (Cong Wang, Computer Science)	AES and Pailier encryption and decryption	Brute-force AES encryption cracking	AES-Python [22], Python-paillier [5]
PAR	Introduction to Hardware Security and Physical Unclonable Function (PUF) Devices (Yiming Wen, Electrical and Computer Engineering)	Hands-on introduction of MPI for Python	Parallel homomorphic encryption of a bitmap data	mpi4py [6], Python-paillier
BD	QoS Assurance in Cloud Services (Xianrong Zheng, Information Technology & Decision Sciences)	Analytics on a large dataset of smartphone app activity using Pandas	Visualization and exploratory data analysis	Pandas [15], seaborn [7]
ML	Radio Frequency Signal Classification and Detection of Drones Based on Machine Learning (Michael Nilsen, Electrical and Computer Engineering)	Classification of smartphone apps based on system utilization data using classic ML methods	Exploration of various ML models to compare performance	scikit-learn [16]
NN	Virtual MAC Spoofing Detection through Deep Learning (Chunsheng Xin, Electrical and Computer Engineering)	Building neural networks to classify smartphone apps based on system utilization data	Tuning the networks for the best performance (hackshop was cancelled)	TensorFlow and KERAS [3]

We started the 2019–2020 workshop series with three workshops focusing on diverse aspects of parallel computing in the Fall semester, followed by the workshops on data-intensive computing in the Spring. Table 1 shows the updated sequence of CI and cybersecurity topics, as well as the hands-on activities, which we will elaborate in the upcoming section. Each row of the table shows the module name, the cybersecurity research presentation (along with the presenter and affiliated department at ODU), the hands-on activities chosen for the workshop and the hackshop, as well as the toolkits introduced. The overall flow of the training program is shown in Figure 1.

3.2 Rewriting the "Data-Intensive" Modules

In the present era where cyber attacks are proliferating and becoming increasingly sophisticated, the application of big data and machine learning techniques to derive timely, actionable intelligence from streams of data in real-time is rapidly becoming an indispensable need to increase cybersecurity posture [8, 13]. As we realize that the use of data-intensive techniques has gradually become a critical skill for cybersecurity students, researchers, and professionals to possess, we rewrote the three modules (BD, ML, and NN) in order to streamline the learning experience and maximize the learning outcome. Unlike the compute-intensive modules, techniques covered in the three data-intensive modules are closely related to one another and are frequently employed together in real-world applcations. The BD module covers the skill to handle large amounts of data as well as making sense of them using exploratory data analysis and visualization. The ML and NN modules build upon this foundation to introduce predictive techniques at increasing levels of accuracy. For this reason, we select a single cybersecurity use case to motivate the needs of BD, ML, and NN techniques. As the key points of these techniques gradually expand throughout the three modules, learners will see the entire pipeline by which the raw data are transformed into final insights and predictions, leveraging state-of-the-art techniques.

We choose the topic of malware detection in smartphones in our new data-intensive modules. This topic is a very relevant cybersecurity issue, which is also relatively easy to understand for anyone with little to no formal training in cybersecurity, as most students today have smartphones and use them extensively. In the near future, smart device users can expect a significant increase in malware and advancements in malware-related attacks, particularly on the mobile open-source platform as the user base is growing exponentially [4]. We make use of the publicly available sample of the "SherLock" Android smartphone dataset created by Mirsky et al. [14]. The SherLock dataset contains detailed information collected from smartphones used by volunteers over an extended period time. Using this dataset, Wassermann *et al.* [19] explored

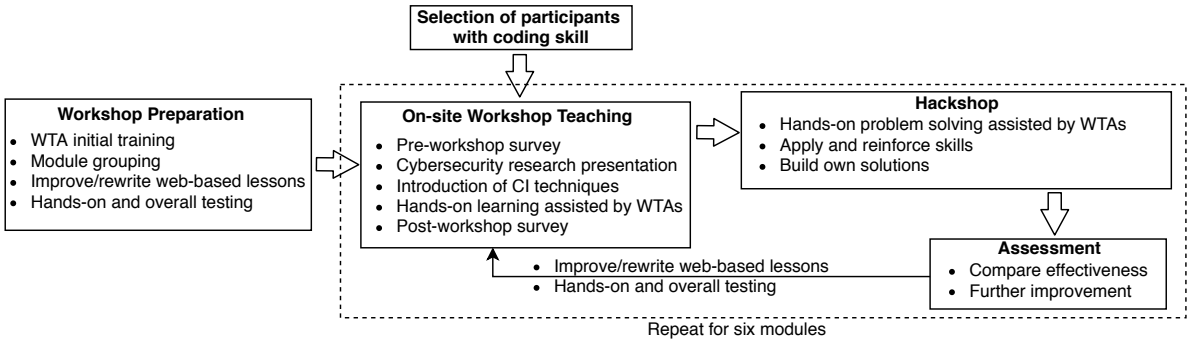


Figure 1: Overall process of the workshop series.

an approach to identify running applications and detect malware activity by analyzing this dataset. Based on their idea, we devised a simplified application classification task and split the entire analysis process into three parts in order to fit it into our rewritten lesson modules. Over the course of the workshop series, the approach of using a single cybersecurity topic would help conserve more time to use in teaching and/or hands-on activities.

For the BD module, we switched our choice of toolkit from PySpark [21] to Pandas [15]. Both are widely used tools that have their own use cases. While Spark is a scalable data processing platform capable of handling extremely large amounts of data (on the order of many terabytes and beyond), Pandas has a more gentle learning curve than PySpark for novice learners, and it is also more popular in the data science community. Although Pandas focuses more exclusively on tabular data, and its scalability is limited to a single computer's random access memory, it is nevertheless sufficient for the purposes of our training program. With this switch, Pandas becomes the base toolkit for all the three data-intensive modules.

3.3 "Hackshops"

To enhance the students' learning experience, we added a "hackshop" as a follow-on session to each workshop. A hackshop is a largely unstructured hands-on session, where learners will actively work on a pre-selected problem and come up with a solution in a small group setting, assisted by instructors and/or teaching assistants. The list of the problems we chose for the hackshops are also listed on Table 1. For the hackshops, we gave the learners some basic instructions and guidelines as well as the goal to achieve, then let them try to work it out on their own for the most part. A hackshop provides an additional opportunity for learners to "hack away" and to sharpen the skills they just learned in the workshop. Unlike the workshops, we made this activity optional to all the learners. Hackshop is a feature we experimented in the second year, as we observed in the first year's workshops that learners did not get sufficient time to freely explore the hands-on materials on their own. We set the hackshop to take place on the same three-hour time slot the week following the workshop.

3.4 Participant Recruitment and Selection

We opened a short enrollment window at the beginning of the Fall semester. We advertised the training through the University Announcements channel, as well as through targeted emails to students in cybersecurity, electrical and computer engineering, and modeling and simulation programs. The participants were expected to attend all six workshops (Fall and Spring); we incentivized this by offering a certificate of completion for those participating in at least five workshops. In the enrollment form, we collected their basic demographic information (gender, ethnicity, study area), as well as self-assessment of their computer competencies, such as programming languages (whether they know how to read and write and the level of complexity of the program written). We accepted participants that have basic programming skills (i.e. those who have at least written a short program—fewer than 100 lines in any language). We did so because the computational techniques require some experience of programming to apply them. As a result of this selection, the Fall workshops were attended by significantly fewer participants than our expected number of around 20. We therefore reopened enrollment at the beginning of the Spring semester, where we also promoted the training program to the HPC user community at ODU. This resulted in a large initial spike of attendees in the Spring (around 30), which dropped to 10 in the last workshop.

3.5 Lesson Developers' Training

Once the basic contents of each module were developed after the first year, it became necessary to refine and prepare them for the continuity of development in a plug-and-play fashion. To ensure continuity of the lesson maintenance, development, and improvement, we trained four workshop teaching assistants (WTAs), who are co-authors of this paper, to become content developers. This effort is seeding a community of contributors for this training program, which will be needed when the training project moves toward an open, community-driven development lifecycle in the near future.

Before the Spring workshop series, a PI held weekly meetings for several months to train the WTAs. The training began with an introduction to pedagogy, lesson development methodology, and tools such as Git/Gitlab, Jupyter notebooks [12], and Jekyll [11]. These initial training sessions prepared the WTAs for a smooth collaborative development process with the PIs to update and/or rework the modules.

The three data-intensive modules (BD, ML, NN) were rewritten collaboratively by the WTAs and the PIs immediately following

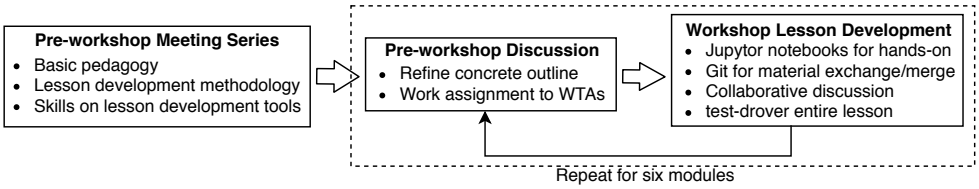


Figure 2: Overall process of the lesson developers' training.

the initial training. First, the team applied the reverse instructional design approach [20] to identify the core concepts needed to achieve the objectives of a lesson. These core concepts were weaved into the lesson outline and the hands-on activities. Each WTA was assigned to build specific parts of the written lesson and/or the hands-on activities by utilizing the knowledge learned from the training. Jupyter notebooks were extensively used to draft and refine the hands-on activities, and a private Gitlab repository was used to exchange and merge lesson materials under development. The WTAs also test-drove the entire lessons, ensuring that the involved steps/operations were clearly understood and making the necessary adjustments. These exercises proved especially valuable to prepare the WTAs to lead breakout sessions in the online delivery mode, because each WTA could separately lead the help session that was tuned by them to suit their own teaching style and preferences. The process of WTA training is shown in Figure 2.

4 ASSESSMENTS AND LESSONS LEARNED

Training assessments were conducted both in the first (Y1) and second (Y2) years of the program. They included online demographic data collection, pre- (PRE) and post-workshop (POST) knowledge questions, and post-workshop opinion questions to evaluate the content and format of the workshops. Figure 3 shows the participants' profiles for Y1 and Y2, including the demographics that show the diversity of participants in race and gender, student classification, and major. In both years, we had a similar total number of unique people participating in at least one of our workshops (44 in Y1 and 43 in Y2). In both years, we had a diverse mix of people in terms of their ethnicity, gender, academic classification, and major. In Y2, more cybersecurity students were drawn into this training, which indicates a positive increase in their interest to what we teach in this program.

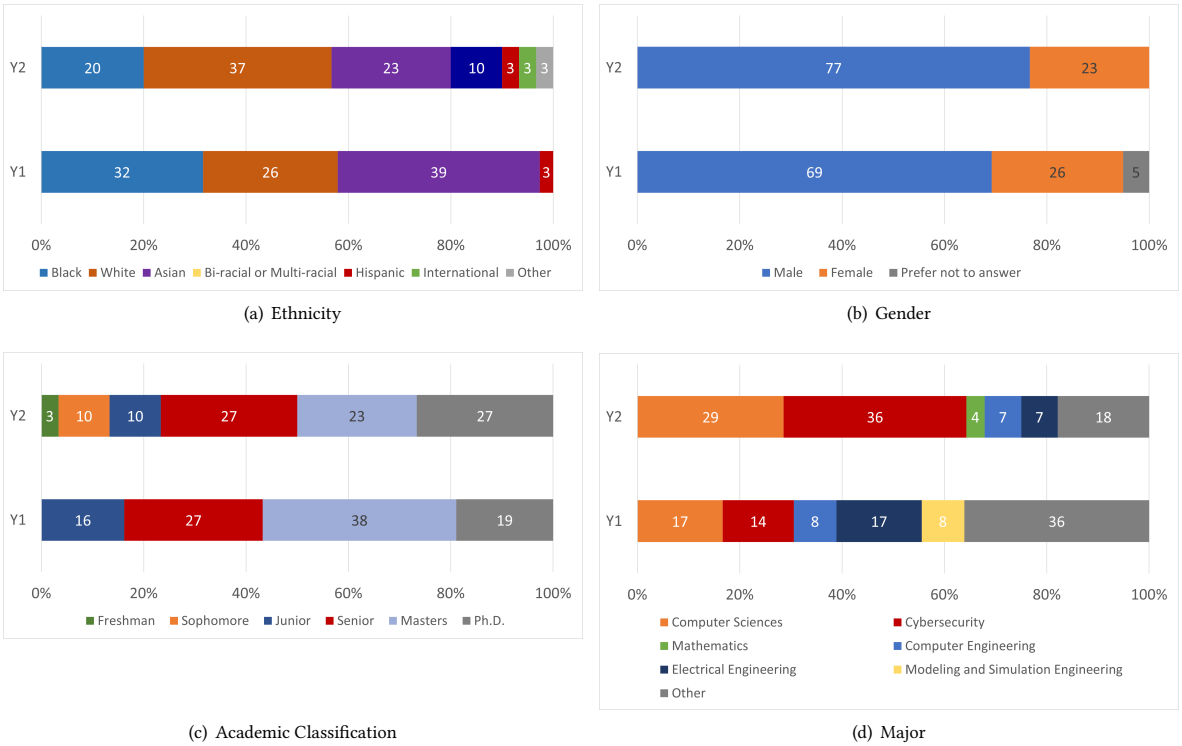
In Y2's enrollment form, we added three new questions to shed light on students' familiarity with UNIX, Python, and C/C++. We asked students to self-evaluate their familiarity with these basic tools: not familiar, novice, intermediate, or expert. In the Fall semester, a large majority indicated that they were not familiar or were novice (about 80%, 90%, and 62% for UNIX, Python, and C/C++). Since UNIX and Python form a critical base for the training, in the first two workshops, we added a brief introduction to these tools. In the Spring semester, we had a better mix of competence, where there were significantly fewer of those who claimed to be unlearned or novice (about 39%, 51%, and 62% for UNIX, Python, and C/C++).

The questionnaires in both years were very similar, which enabled us to compare the effectiveness of our mid-project changes. However, the focus of the evaluation during Y1 was to obtain formative information to improve the workshops as they were being delivered. The post-workshop opinion questionnaire for Y1 was very comprehensive, with 15 questions, including rating of specific components (content, organization, pace, etc.) and open-ended questions to gather qualitative information from participants on what needed improvement and what they found to be most and least valuable from each workshop. For Y2, the rating and opinion questionnaire was shortened to five questions. There were no radical differences in answers to the opinion rating and open-ended questions between the two years. All the workshops in each year were rated as good or extremely good by more than 80% of participants. In Y2, two out of six workshops were rated as "neither good nor bad" by one person; in Y1, the opinions on the very first workshop differed greatly, which we took into account right away and remedied in all the subsequent workshops. (See our description on the necessary adjustments in [18]). Overall, the students received the training program very well in both Y1 and Y2; many of them indicated the hands-on exercises and new knowledge as the most valuable takeaways of the workshops.

Two important metrics that we strive to improve by implementing the changes in the second year are (1) attendance retention and (2) knowledge acquisition. We will consider both quantitative measures (such as number of participants, knowledge assessment results) as well as qualitative and anedoctal feedback to evaluate the impact of our effort in Y2. While the quantitative measures shed light on the areas we need to further improve, we still receive many encouraging feedback from our own observation of, and direct interaction with, the participants.

4.1 Attendance Retention

Figure 4 shows the number of attendees for every workshop we held in Y1 and Y2. Our target is to have 20–25 participants on average per workshop. In Y1, due to the late start of the project in the semester, we held two workshops in the Fall and four workshops in the Spring semester. The first workshop in Fall 2018 was attended by more than 30 students; by the end of Spring 2019, the workshops were consistently attended by 11–13 students, representing $\tilde{3}0\%$ of the original number of participants in the first workshop. As we mentioned earlier, in Y2 we started with a lower number of participants, because we required participants to have basic computer programming experience. A second enrollment in the Spring led to another spike in attendance (30), which leveled to 13 at the end. Figure 5 shows a measure of attendees' retention by counting the number of participants who attended any $N = 1, 2, \ldots 6$ number



(a) Ethnicity

(b) Gender

(c) Academic Classification

(d) Major

Figure 3: Demographic distribution of the workshop participants, comparing the first year ("Y1", 2018–2019) and the second year ("Y2", 2019–2020).

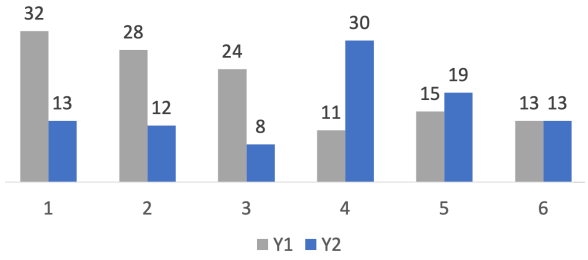


Figure 4: Number of participants attending each workshop.

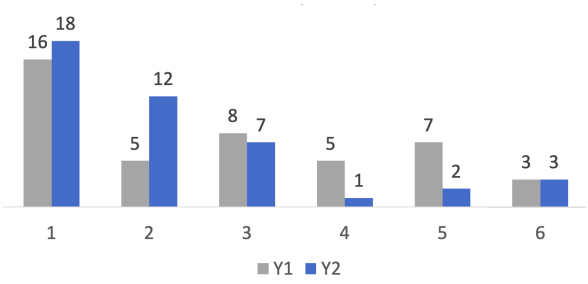


Figure 5: Number of participants that attended any N workshops (N shown on the horizontal axis).

of workshops. The result is also mixed, where Y2 shows better participation for $N = 2$, while Y1 shows better participation in 4 or 5 workshops.

From this we learn that for a non-credit workshop series, attendance tends to spike only on the first workshop; later on participants who remain would do so because they are truly interested in the topic of the workshop. It is worth commenting that while we were not able to achieve our targeted number of 20, from our interactions with the learners, those who remained were very engaged and interested in the materials. The numbers 10–15 may very well be the natural size of the cohort for our local community. Given that our lesson modules are divided into two categories, it seems reasonable that we would open the enrollment twice a year, one for each category, thereby allowing students to pick three modules that better align with their interests.

4.2 Knowledge Assessment

A second metric of interest is whether there is an improvement in the knowledge acquisition as the result of the content changes implemented this year (reorder of the module sequence, rewrite of the data-intensive modules, change in tools). To compare the knowledge acquired by participants, we selected two workshops in each year on the same topics (BD and CRYPT). In each workshop, we asked 5–8 questions on the fundamentals of the CI topics at the beginning of the workshop and at the end to assess the impact of the workshop on the participants' understanding about the topic. These

are high-level questions such as, "What is considered the primary goal of looking at big data/large data sets?" and, "Exploratory Data Analytics is ...(mark all that apply)" (for BD module); "What is the homomorphic encryption?" and, "Without the key, you cannot recover the message from a ciphertext. Which statement is true?" (for CRYPT module).

In Y2, the CRYPT module was offered in the Fall, and a short introduction to Python was added due to the fact that the majority of the learners were very new or not familiar with Python. The analysis of the knowledge questions in aggregate for the CRYPT workshops shows that the knowledge acquisition was better in Y1 than Y2. It is likely that participants missed some of the key knowledge due to the inadequate amount of time to cover the less familiar topic of encryption.

In the BD workshop, which was offered in Y2 in the Spring, comparing the PRE- and POST-knowledge responses shows an improvement in four out of five questions, as compared with Y1, where only two out of five questions show an improvement. The BD module was reworked this year, and this improvement may indicate that our improved lesson and delivery resulted in better understanding of the topic.

These numerical results need to be taken with a grain of salt. The sample size, i.e. the number of responses, was very small in these surveys. For the CRYPT workshops, the sample size is 5 and 6, for Y1 and Y2 respectively, whereas for the BD workshops, they are 6 and 11. Hence, an analysis of knowledge acquisition will have to be done at the respondent level to draw deeper conclusions.

In general, the mixed results suggest that we need to adapt further our materials to better fit into the 3-hour workshop duration. For example, in Fall 2019, much time was spent to introduce UNIX shell in the HPC module and basic Python syntax in the CRYPT module. As a result, more pertinent topics (such as job scheduler, parallel processing, and Paillier encryption), were short-handed, and may have lead to weaker results in the POST test after the CRYPT module. In Spring 2020, we adhered better to relative time constraints during workshops, the downside of which was a perception of rushing though the material, as expressed during the focus-group interviews conducted post-workshops. We continue our search to strike a right balance of topic coverage within a workshop. Our current solution is to carefully select topics to cover in depth during a workshop, while leaving the remaining ones for interested learners to pursue on their own using, e.g., our web-based lesson materials and/or Jupyter notebooks.

4.3 Hackshops

In Y2, we provided the new "hackshop" session, which provided a much higher level of interactivity and engagement of the learners with the materials, as well as with TAs and instructors. According to the statistics, over 55% participants came to the hackshop, and we are happy to see five learners from the Y1 workshop series coming back in our new hackshops at least once. They gave us positive feedback on how the workshop synergistically helped them in their coursework. We consider this a promising seed towards building a local community of practice for computational techniques in cybersecurity.

Based on our observation, participants who came to hackshops were able to engage with the hands-on tasks with great interest. In this respect, the hackshops accomplished their purpose. However, the desired goals in these hackshops (e.g., cracking a secret message) were not achieved, partly due to the gap between participants' programming competence and the required skills to complete these goals. We learned that participants may need more scaffolding, i.e. more guidance and stepping stones, to solve the challenge questions within a three-hour timeframe.

5 PILOT ONLINE WORKSHOP

DeapSECURE workshops were originally designed for in-person workshops, although the sessions were recorded with an intention to build an online version of the training in the future for scalability. The COVID-19 pandemic hit shortly after we finished our last workshop in Spring 2020, which provided us a strong impetus to convert our training to a fully online (remote) format. We decided to try out one pilot online workshop using the BD module in the summer 2020 in lieu of a Summer Institute. This conversion required a thorough redesign of the workshop format to suit the online delivery and learning experience. The planning and redesign process took a substantial amount of time (about three months). A great challenge with the online format was the lack of interpersonal interactions and the inability to directly assist learners on their own computers. Another significant challenge was the limited screen real-estate available for the hands-on format. To help learners overcome these challenges, we developed three Jupyter notebooks which closely mirror the progression of the hands-on activities in the web-based lesson module. The key points as well as incomplete code snippets from the web-based lesson were incorporated concisely in the notebooks, thereby removing the need to open two browser tabs to follow the instructor. Participants accessed the Jupyter environment on ODU's Wahab HPC cluster via the newly deployed Open OnDemand [9] web-based interface. This proved to alleviate most of the technical difficulties encountered in the past workshop series.

The pilot workshop consisted of three one-hour sessions with 15-minute breaks in between. About ten participants joined the workshop via the Zoom platform. Each session started off with a brief explanation of the basic concepts as well as hands-on demonstration using the Jupyter notebooks, followed by a hands-on exercise held within smaller groups in Zoom breakout rooms led by WTAs. To maintain participants' level of interest, we conducted a 5-minute interactive yet competitive quiz session using the Kahoot! platform [1] at the end of each session. The results of the quiz provided feedback by measuring the learning success of the session. The Slack [2] platform was used for nonverbal communications (chats) during the workshop, which we leveraged to maintain contact with learners after the workshop. Slack messages are persistent, thus previously answered questions and addressed challenges can be recalled in the future. Overall, based on the informal feedback from participants, the pilot workshop was successful. A detailed assessment of this event is outside the scope of this paper.

6 SUMMARY AND FUTURE DIRECTION

In summary, we performed major improvement of the DeapSECURE lesson modules by grouping them into the "compute-intensive" and "data-intensive" categories, more tightly integrating the modules to streamline the learning experience. The current version of the web-based lesson materials can be accessed from our main website [17]. We added "hackshop" into our training schedule to increase participants' engagement with the hands-on materials. We trained a cohort of workshop teaching assistants to be contributors to further development and refinement of the lesson materials. The assessment results indicate the need for further adjustments to improve learning experience and outcome. The pilot workshop showed great promise to address some challenges we encountered through the second year project. We believe that the improvements we implemented in the second year will put us in a good position to offer the entire portfolio of DeapSECURE modules online and provide learners with the best online learning experience.

The online pilot workshop in Summer 2020 has shown that online training is not only feasible but even more effective in reaching out to trainees who otherwise could not be part of the program. In the next project year, the development of a fully online training format utilizing all the six modules is planned. Efforts will be made to ensure the online training is engaging and effective. The training modules will be streamlined for online delivery. Lectures will be completed in a large group format while labs and games will be completed in small groups facilitated through Zoom breakout room. Effort is underway to ensure that the training materials (lessons and hands-on) can be ported to other institutions and HPC sites. The PIs will also reach cybersecurity as well as CI professional communities throughout the U.S. to promote the adoption of DeapSECURE in other parts of the country. Once the preparation for fully online workshops have been completed, this training can be offered across universities the Commonwealth of Virginia on "ACCORD", a shared cyberinfrastructure currently being built for computation of protected data as well as training and education [10]. The online workshops will be fully assessed along with a reproducibility pilot study to broaden the subject domain from cybersecurity to another area, such as computations with sensitive data.

ACKNOWLEDGMENTS

The development of DeapSECURE training program is supported by NSF CyberTraining grant #1829771. The workshops utilized Turing and Wahab HPC clusters provided by ODU Research Computing Services, part of Information Technology Services. Wahab cluster was acquired in part using NSF Major Research Instrumentation grant #1828593. We thank Issakar Doude, who assisted the lesson development throughout the first year and part of the second year. We also thank the ODU Distance Learning for their support in recording the workshop sessions.

REFERENCES

[1] 2020. Kahoot! Game-based Learning Platform. https://kahoot.com
[2] 2020. Slack. https://slack.com
[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.
[4] Abdullahi Arabo and Bernardi Pranggono. 2013. Mobile malware and smart device security: Trends, challenges and solutions. In *2013 19th international conference on control systems and computer science*. IEEE, 526–531.
[5] CSIRO's Data61. 2013. Python Paillier Library. https://github.com/data61/python-paillier
[6] Lisandro D. Dalcin, Rodrigo R. Paz, Pablo A. Kler, and Alejandro Cosimo. 2011. Parallel distributed computing using Python. *Advances in Water Resources* 34, 9 (2011), 1124 – 1139. https://doi.org/10.1016/j.advwatres.2011.04.013 New Computational Methods and Software Tools.
[7] Michael Waskom et al. 2017. Seaborn: statistical data visualization. https://doi.org/10.5281/zenodo.592845
[8] B Geluvaraj, P.M. Satwik, and T.A. Ashok Kumar. 2019. The Future of Cybersecurity: Major Role of Artificial Intelligence, Machine Learning, and Deep Learning in Cyberspace. In *International Conference on Computer Networks and Communication Technologies (Lecture Notes on Data Engineering and Communications Technology)*, S. Smys, R. Bestak, J.Z. Chen, and I. Kotuliak (Eds.), Vol. 15. Springer, Singapore, 739–747.
[9] David E. Hudak, Douglas Johnson, Jeremy Nicklas, Eric Franz, Brian McMichael, and Basil Gohar. 2016. Open OnDemand: Transforming Computational Science Through Omnidisciplinary Software Cyberinfrastructure. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale (XSEDE16)*. ACM, New York, NY, USA, Article 43, 7 pages. https://doi.org/10.1145/2949550.2949644
[10] Ron Hutchins, Scott Midkiff, Masha Sosonkina, Thomas Cheatham, Deborah Crawford, and ACCORD Team. [n. d.]. The Virginia ACCORD Project. https://www.va-accord.org/
[11] The jekyll team. 2018. Jekyll—static site generator. https://jekyllrb.com
[12] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing, and Jupyter Development Team. 2016. Jupyter Notebooks—a publishing format for reproducible computational workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, Fernando Loizides and Birgit Schmidt (Eds.). IOS Press, Amsterdam, 87–90. https://doi.org/10.3233/978-1-61499-649-1-87
[13] Tariq Mahmood and Uzma Afzal. 2013. Security Analytics: Big Data Analytics for Cybersecurity: A Review of Trends, Techniques and Tools. In *2013 2nd National Conference on Information Assurance (NCIA)*. IEEE, 129–134. https://doi.org/10.1109/NCIA.2013.6725337
[14] Yisroel Mirsky, Asaf Shabtai, Lior Rokach, Bracha Shapira, and Yuval Elovici. 2016. SherLock vs Moriarty: A Smartphone Dataset for Cybersecurity Research. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security (AISec '16)*. ACM, 1–12. https://doi.org/10.1145/2996758.2996764
[15] The pandas development team. 2020. pandas-dev/pandas: Pandas. https://doi.org/10.5281/zenodo.3630805
[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.
[17] Wirawan Purwanto, Issakar Doude, Yuming He, Jewel Ossom, Qiao Zhang, Liwuan Zhu, Masha Sosonkina, and Hongyi Wu. 2020. DeapSECURE Lesson Modules. https://deapsecure.gitlab.io/lessons/
[18] Wirawan Purwanto, Hongyi Wu, Masha Sosonkina, and Karina Arcaute. 2019. DeapSECURE: Empowering Students for Data- and Compute-Intensive Research in Cybersecurity through Training. In *Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning) (PEARC '19)*. ACM, New York, NY, USA, Article 81, 8 pages. https://doi.org/10.1145/3332186.3332247
[19] Sarah Wassermann and Pedro Casas. 2018. BIGMOMAL: Big Data Analytics for Mobile Malware Detection. In *Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity (WTMC '18)*. 33–39. https://doi.org/10.1145/3229598.3229600
[20] Grant P. Wiggins and Jay McTighe. 2008. *Understanding by Design* (2nd ed.). Association for Supervision and Curriculum Development, Alexandria, VA.
[21] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. 2016. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* 59, 11 (Oct. 2016), 56âĂŞ65. https://doi.org/10.1145/2934664
[22] Bo Zhu. 2015. A pure Python implementation of AES. https://github.com/bozhu/AES-Python.git

Exploring Remote Learning Methods for User Training in Research Computing

Dhruva K. Chakravorty[1]
chakravorty@tamu.edu

Lisa M. Perez[1]
perez@tamu.edu

Honggao Liu[1]
honggao@tamu.edu

Braden Yosko[1]
byosko@tamu.edu

Keith Jackson[1]
kjackson@tamu.edu

Dylan Rodriguez[1]
dylan@tamu.edu

Stuti H. Trivedi[1]
stutitrivedi1373@tamu.edu

Levi Jordan[1]
ljordan56@tamu.edu

Shaina Le[1]
sle1019@tamu.edu

ABSTRACT
The COVID-19 national health crisis forced a sudden and drastic move to online delivery of instruction across the nation. This almost instantaneous transition from a predominantly traditional "in-person" instruction model to a predominantly online model has forced programs to rethink instructional approaches. Before COVID-19 and mandatory social distancing, online training in research computing (RC) was typically limited to "live-streaming" informal in-person training sessions. These sessions were augmented with hands-on exercises on live notebooks for remote participants, with almost no assessment of student learning. Unlike select instances that focused on an international audience, local training curricula were designed with the in-person attendee in mind. Sustained training for RC became more important since when several other avenues of research were diminished. Here we report on two educational approaches that were implemented in the informal program hosted by Texas A&M High Performance Research Computing (HPRC) in the Spring, Summer, and Fall semesters of 2020. These sessions were offered over Zoom, with the instructor assisted by moderators using the chat features. The first approach duplicated our traditional in-person sessions in an online setting. These sessions were taught by staff, and the focus was on offering a lot of information. A second approach focused on engaging learners via shorter pop-up courses in which participants chose the topic matter. This approach implemented a peer-learning environment, in which students taught and moderated the training sessions. These sessions were supplemented with YouTube videos and continued engagement over a community Slack workspace. An analysis of these approaches is presented.

CCS CONCEPTS
•**CS**→Computer Science; •**Cybertraining**→training on using cyberinfrastructure; •**HPC**→high performance computing

[1] High Performance Research Computing, Texas A&M University, College Station, TX

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/2

Keywords
Online education, COVID-19, YouTube education, Cybertraining

1. INTRODUCTION
The COVID-19 national health crisis forced a sudden and drastic move to online delivery of instruction across the nation. This almost instantaneous transition from a predominantly traditional "in-person" instruction model to a predominantly online model has forced programs to rethink instructional approaches. Unlike select instances, such as the Petascale Institute, that have traditionally focused on a geographically-distributed audience, local campus computing training curricula were primarily designed with the in-person learner in mind. Prior to the changes brought by COVID 19-related national social distancing norms, online training in research computing (RC) was typically limited to "live-streaming" informal in-person training sessions. For example, training and educational sessions offered by Texas A&M HPRC **[**1**]** primarily focused on the "in-person" participants, with tracking, support, and strong assessments. The online experience was augmented with hands-on exercises on live notebooks for remote participants, with limited assessments of efficacy and student learning.

The impact of these adopted social norms affected research computing as well. In the Spring months of 2020, with a view toward combatting the spread of COVID-19, several institutions staggered, limited, or closed research facilities that required in-person interactions. While researchers were asked to practice social distancing at some institutions, at others they were encouraged to stay off campuses. Unable to perform physical experiments, computationally-curious, albeit untrained, researchers flocked to campus RC sites. For example, at Texas A&M HPRC, we saw a significant increase in both new users and the number of job submissions on our clusters. This influx of new researchers offered opportunities to experiment with sustainable and scalable training approaches for researchers new to RC.

2. ONLINE TRAINING AND EDUCATION
Much like other campus research computing efforts, Texas A&M HPRC has offered a series of training, outreach, and educational efforts that supports our researcher community [2–6]. Our user training program has been operational for several years, with thousands of participants signing up for events. At its heart are two-and-a-half-hour sessions, called the short course program, that are built along the idea of active-learning approaches [7–10]. Prior to March 2020, these sessions were offered both in-person and over live remote (Zoom/WebEx) modalities. These sessions were augmented with day-long workshops that were traditionally

focused on in-person attendees. Both the short courses and workshops largely relied on our production environments — Jupyter notebooks, virtual machines, and command line interface (CLI). While instructional aids (slide decks and Jupyter notebooks) were available on our website, video recordings of the courses were not available. These courses have been included in several formal curricular efforts at Texas A&M. A detailed report on them has been presented elsewhere [2–6].

In response to social distancing policies recommended due to the COVID pandemic, we realized that we had to change our approach toward user training. We adopted two distinct approaches toward user training. For the Summer and Fall 2020 semesters, Texas A&M HPRC chose to offer both versions of its informal learning program in an online modality. These programs continue to evolve as we experiment with pedagogical approaches to strengthen our curricula. Here we report on the progress, strengths, weaknesses, and opportunities to improve on these approaches. At the outset, these programs were offered over Zoom, with the instructor assisted by moderators using the chat features.

2.1 Short Courses

The first approach, called shortcourses, closely mirrored our traditional in-person focused sessions, albeit in an online-only setting. By design, these courses are detailed, information-intensive, and built as information resources that can be revisited by participating students. These are typically taught by experienced HPRC staff, Texas A&M faculty, or scientists. Curricular materials are available for download from Github or the HPRC website. These courses are two-and-a-half hours long and are tiered with other short courses. To further establish a learning structure, these courses are often combined with complementary offerings, such as workshops or user group meetings. In the now online format, the course instructor was supported by other HPRC scientists over Zoom chat. The goal here is to offer a deeper introductory dive into computing. These courses are built on a tiered instruction model where the topics covered during the courses build on each other. As such, a learner can participate in courses throughout a semester and develop a comprehensive understanding of RC software and tools. To enable effective delivery, we developed a document describing expectations from presenters and participants on Zoom. To collect participant feedback on our courses, we migrated our surveys to Google Forms.

2.2 Primers

Toward the end of the Spring 2020 semester, we realized that our now online short course program was probably competing with other online commitments for a learner's time. We were also concerned that the short course program took a considerable amount of staff time away from responding to our growing user-needs. We also realized that, traditionally, new users often belonged to research groups that had roots in RC. In this scenario, we could rely on existing computing expertise within the new user's group to bring him/her/them up to speed. Due to the COVID-19 crisis, we had a new set of researchers join research computing. These computationally-curious researchers belonged to research groups (or facilities) that didn't provide the scaffolding that our short course program relied on. Since these researchers came from varied backgrounds, we also didn't have a pre-existing framework that informed us what and how these researchers wanted to learn.

Despite the curricular strengths of our short course program, we felt the need for a new pedagogical approach that taught the new generation of users while focusing on learner engagement [11]. Admiring the success of short videos on social media platforms such as TikTok and YouTube, we realized that online informal education could be offered as bursts of information rather than relying on a structured tiered learning approach. In a related vein, platforms such as Discord have successfully coupled "live streaming" with "live chat" to engage the audience. Here, the presenter performs a task and converses using the video feed, while the audience participates in a "live chat" where they react or add to the presenter's actions. During this time, we also noticed that users were requesting information via our Helpdesk ticketing system that could be scaled out via informational videos. These requests were typically handled by our experienced student technician group that includes members from current and previous Super Computing Student Cluster competition teams.

3. EXPERIMENTAL DESIGN

Driven by the need to innovate, and inspired by the opportunities in our operations group, we developed a second approach that focused on high learner engagement by offering information on demand. This program, called Primers, relied on 60-minute courses and moved away from the focus on a semester-long learning experience. The Primer courses were intended to provide a burst of information for learners in an online-learning-friendly format. For the Primers, we first identified core competencies that RC researchers need to know. These core competencies were identified via discussions with HPRC staff, consultation with groups working in this area, HPRC user tickers, our "Introduction to HPRC" short course, and its corresponding assessments.

As part of this design, we took a cue from pop-up courses and crowd sourced when and how often these topics should be taught. Towards this, the registration form allowed participants to vote on the courses that would be offered, the sub-topics to be covered during the course, and suggestions on what should be taught. As a rule, we required that a minimum of five learners had to register for a Primer course for it to be offered. The program was geared to offer quick information and get a user to actively work on the problem. Unlike our short course program, it had no explicit tiered or prolonged learning structure. As such, we anticipated learners signing up for one-off courses, with the learning limited to a single semester. Building on the depth of expertise in our student technician program, we implemented a peer-learning environment in which two experienced undergraduate or graduate students taught and moderated the training sessions.

Instructional materials for the Primers were prepared by Texas A&M HPRC staff and students. While one student technician presented the material and guided the class through the hands-on sessions, the second student technician posted comments on Zoom chat and added additional information. Scaffolding was offered via materials like Jupyter notebooks [12]. Each 60-minute session was followed by a 15-minute informal "Open Mic" session during which, participants could chat or talk about any topic related to RC. To ensure success, we endeavored to build a support structure along the live courses. To capture these discussions and foster closer collaborations among researchers, participants were invited to use the NSF CC* Cyberteam SWEETER Slack workspace [1]. In addition to offering course-related resources, such as slide decks and Jupyter Notebooks, these sessions were recorded and offered as YouTube videos. These recordings are available free-of-charge via the Texas A&M HPRC YouTube channel. Closed captioning was included on each video, and the videos met Texas A&M's requirements for the Americans with Disabilities Act.

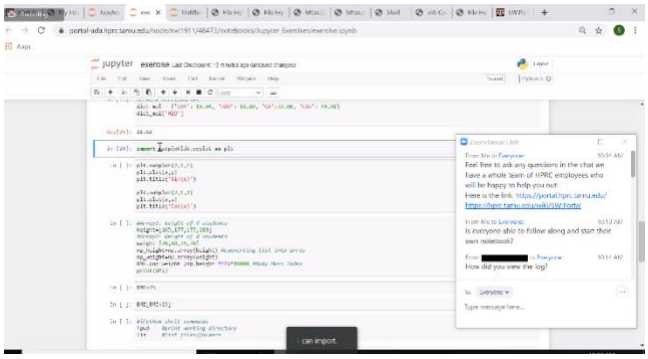


Figure 1. A screenshot of the teaching interface. Here the peer-instructor is working on a Jupyter Notebook while the peer-moderator encourages and supports a parallel discussion in the Zoom chat window.

4. RESULTS AND FUTURE WORK

Here, we briefly describe the results from our online short course and Primer programs. A complete list of our training activities is available on the HPRC website.

4.1 Short Courses Offered

In Spring 2020, we completed our planned bouquet of Spring 2020 short courses in an online-only format using the Zoom platform. This was followed by an online series of short courses on Quantum Mechanics offered in Summer 2020. In Fall 2020, the shortcourses returned to our offering. The move to an online-only platform did not impact the number of participants registering for our short courses. Registration and participation in the Spring 2020 short courses mimicked that of previous semesters, when the courses were taught in the hybrid in-person and online format. Since all interactions were now via Zoom, we noticed that the interactions between the instructor and the attendees were much more limited. This was a marked change from the in-person interactions between the instructor and the participants, and it has been ascribed to variety of factors, ranging from technology limitations, poor internet connections, participant hesitation to speak out in front of a larger audience, a reluctance to enter questions into chat forums, competing online distractions, and a lack of engagement with instructor or course materials.

4.2 Primer Courses Offered

The Primer courses were launched in late Spring 2020. The Primers were advertised and managed using our regular broadcast email, and registration and content was managed via our website and Google Forms. To our pleasant surprise, and perhaps an indicator of the rising demand for research computing, all course offerings were selected, and we rapidly reached the minimum threshold of five learners for each Primer course. Primers were offered on introductory topics related to Linux, CLI, Cluster Usage, scheduler usage (SLURM and IBM Spectrum Scale LSF), using the OpenOnDemand Portal, Data Management Practices, and using Jupyter notebooks. A listing of all Primer courses offered in 2020, and the number of students registered per course are presented in Table 1.

For the purposes of brevity and maintaing clarity, Primer courses are grouped in terms of Operating Systems (Linux), Technology (Jupyter Notebooks and Data Management Practices), Schedulers (LSF and SLURM), and Clusters (Ada and Terra) in this manuscript. The portal refers to Texas A&M HPRC's implementation of the OpenOnDemand portal developed by Ohio Supercomputer Center. In all, 15 Primers were offered.

Table 1. List of Primer courses, and the number of registered attendees for each session, from Spring 2020 to Fall 2020. The primers are listed in the order in which they were presented.

Semester	Courses	Registered
Spring 2020	Introduction to Linux w/ MobaXterm	126
	Introduction to the Ada Cluster	98
	LSF: Job Scheduling	44
	Introduction to the Terra Cluster	92
	SLURM: Job Scheduling	27
	Data Management Practices	96
Summer 2020	Introduction to HPRC – Clusters, Duo, VPN	63
	Jupyter Notebooks on the Portal	59
	Introduction to Linux w/ MobaXterm	40
	Introduction to Linux w/ Portal	39
	Introduction to the Ada Cluster	42
	LSF Job Scheduler	44
	Introduction to the Terra Cluster	43
	SLURM Job Scheduler	69
	Data Management Practices	91
Fall 2020	Introduction to Linux w/ Portal	71
	Introduction to the Terra Cluster	54
	Introduction to the Ada Cluster	64
	Data Management Practices	70
	SLURM: Job Scheduling	53
	LSF: Job Scheduling	55
	Jupyter Notebooks on the Portal	67

On average, about 55 participants registered for each Primer course. Due to the unique registration format, registered participant counts include those who showed interest in the topic and didn't have a preference for the day on which the course was offered. It is noteworthy that since new graduate student enrollment is typically highest in Fall semester, we see typically see a drop-off in participation in our Introductory short courses in the Spring semester. The registration numbers for Spring reflect enthusiasm for both computing and the new learning format at that time. In response to the continued demand for quick online programs, the Primers were offered a second time in Summer of 2020.

Summer attendance in the Primers series was encouraged by the summer research learning programs such as the Online Research Experiences for Undergraduates program at Texas A&M. A slight drop in registrations was observed. In Fall 2020, we continued to work in an online-only setting. As such, the Primers program ran in parallel with the Short course program. Figure 2 shows that a greater number of learners registered for the Primers in the Spring, Summer, and Fall semesters of 2020, as compared to those registered for similarly-themed Introductory short courses that were offered in the hybrid in-person and online format in Fall 2019. Perhaps a testament to the success of this online-only format is the continuing participation in Fall 2020.

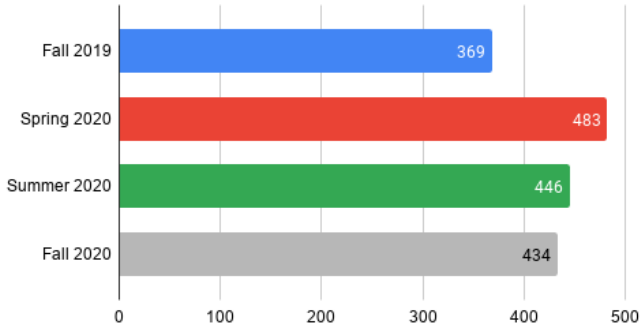


Figure 2. Total number of participants registered for Primer courses in Spring (red), Summer (green), and Fall (gray) semesters of 2020. For comparison, we show the number of students who participate in the 2.5-hour-long course in Fall 2019 (blue).

4.3 Participation Trends

Primers maintained student interest in all the major categories, as shown in Figure 3. Consistent with the class of new researchers using our facilities, we saw increased participation in courses related to using the campus clusters and interactive technologies like Jupyter notebooks. Polling data collected during the registration process found that nearly all participants voted for all the topics. As such, beyond telling us that the pre-selected topics were of interest, crowd-sourcing did not provide clear guidance on what sub-topics to teach. Our SWEETER Slack workspace offers a rich collaborative space that connects over 470 researchers from several countries. It includes several public and private channels related to research computing and software usage. We also find that while several learners joined the SWEETER slack, most discussions took place on private topic-specific channels rather than on a public channel. As the courses progressed we learned that while the interactive sessions were scheduled for 10 minutes, they may carry on for up to 30 minutes after a primer course. As such, we assume that the Primers filled a significant knowledge gap for researchers new to research computing clusters environments.

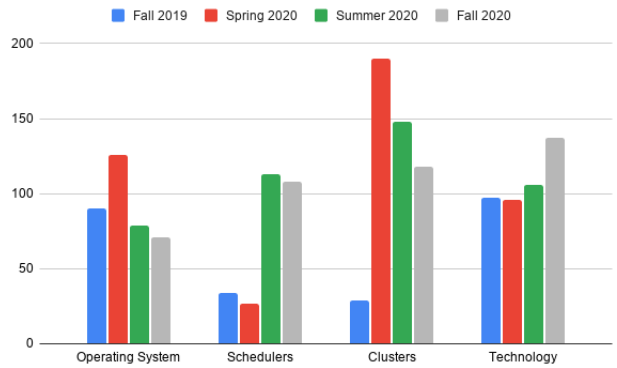


Figure 3. Participation in Primers in the Spring, Summer, and Fall semsters of 2020. Participation in the 2.5-hour version of short courses in Fall 2019 is shown as a benchmark for when longer sessions were offered on these topics.

Texas A&M HPRC supports users from several fields of science. Figure 4 shows the participation of learners from various colleges representing different fields of science. We find that the courses maintained the cross-disciplinary appeal that was observed in our short courses program in Fall 2019. Increased participation from Engineering disciplines was observed. This possibly ties into the increased use of computing in engineering research, students wanting to learn new research skills during the downtime brought about by the COVID-19-implied norm, and possibly because learners were now able to tune into an online course, rather than travel to a classroom on the other end of campus.

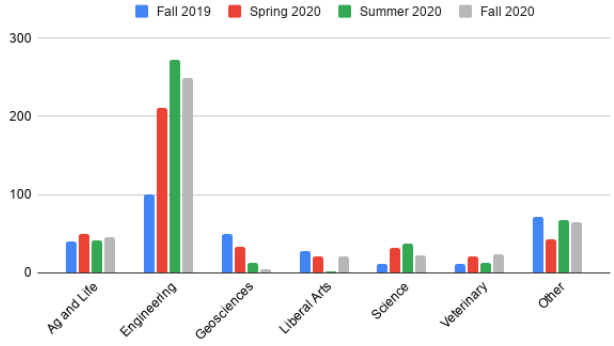


Figure 4. Participants per college for Spring, Summer, and Fall semesters of 2020. For comparative purposes, we include data from Fall 2019.

4.4 Learner Persistence

Tracking how students approach the topics offered by the program and learner persistence are key considerations for improvements in future iterations. For each of these live-streamed Primer courses, persistence was tracked along two lines of enquiry. First we observed how long a participant remained on during a course, and next we saw how many Primers courses were attend by a learner.

Here, we report on our findings for the Primers offered in the 2020 calendar year. As described above, our original target participation for our courses was five participants per course. In order to track persistence, i.e. what percentage of students complete the session, across a Primer course, we observed how long a participant remained on the Zoom session. The data from the calendar year is shown in Figure 5. Here we find a slight drop-off in the first 15 minutes. The majority of learners (greater than 60%) complete the hour-long exercises and stay for the Open Mic session. We hypothesize that the early drop rate could be reflective of various factors. Learners could have realized that they have either signed up for the wrong class, that the class materials and course recording are available for later viewing, that the materials do not meet with their expectations, or perhaps they have unstable Internet connectivity. We find that as we got into the Summer and Fall semesters, more students remained until the conclusion of the course. We surmise this is because learners are becoming more familiar with the platform and adjusting their expectations. Noting that this metric may be an indicator of the popularity of the Open Mic session that happens after the Primer, we point out that participation in these sessions varies depending on the topic and the audience on a given day.

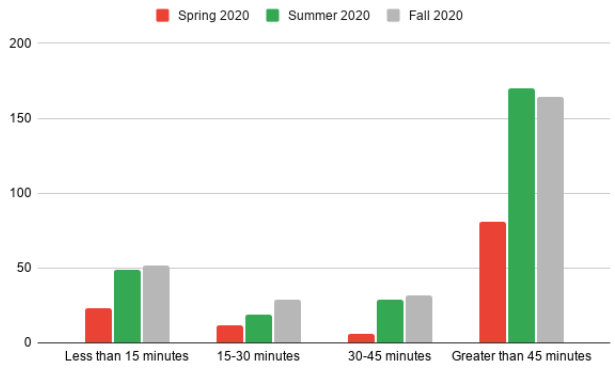


Figure 5. Learner persistence in each Primer session. Number of minutes spent by Zoom attendees (Y-axis) in Spring, Summer, and Fall Semesters of 2020.

Learner participation in the program was tracked across each semester, and across multiple semesters. As described previously, the Primers are geared to give relatively quick bursts of information and are not tiered for a longer or sustained learning effort. As shown in Figure 6 (a), we find that consistent with our intended goals, 45% of learners attended a primer on a given topic, and 43% of learners continued to participate in two or more classes. Figure 6 (b) shows the distribution of learning across semesters. We find that consistent with the goals of the program, the overwhelming number of learners attend Primers in a single semester. A small percentage of learners availed of the primers across two semesters.

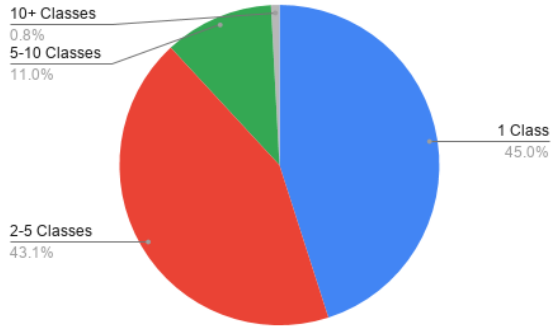


Figure 6 (a). Total courses registered per learner across the Fall, Summer, and Spring 2020 semesters.

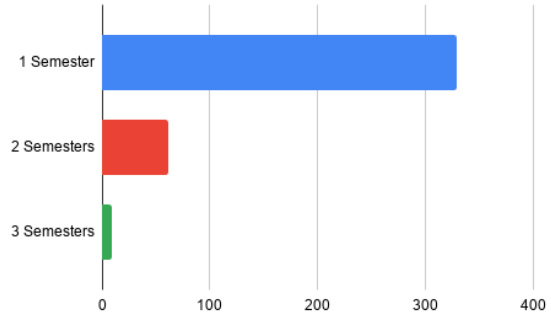


Figure 6 (b). Number of learners registered for multiple semesters across the Fall, Summer, and Spring 2020 semesters.

4.5 Staged Curricular Materials

We have continued to stage our teaching materials and exercises on online platforms. Our website [1] hosts a collection of our training materials. These materials are updated by the instructors each time the Primers are offered. For the Fall 2020 semester, we find that the Primer course slide decks and notebooks for the Primers were downloaded 517 times by individuals and ~30 times by bot services. Details of downloads per course and thematic areas are shown in Table 2, and the breakdown across thematic areas is shown in Figure 7. Consistent with Primer registration, we find that cluster usage dominated among these categories.

Table 2. List of Primer and short videos offered on the Texas A&M YouTube channel and associated views.

Type	Courses	Views
Introductory Videos (5 minutes or less)	What is Texas A&M HPRC?	174
	Applying for Accounts	141
	Cluster Access using SSH	113
	Accessing Cluster from Windows	33
	File Management on Clusters	99
	Managing Allocations	122
	Modules System	56
	Submitting a Job using LSF	162
	Submitting a Job using SLURM	23
	Submitting a Job File using Tamubatch	100
Primers (45 to 60 minutes)	Introduction to HPRC – Clusters, Duo, VPN	77
	Jupyter Notebooks on the Portal	65
	Introduction to Linux	13
	Introduction to Linux on a portal	96
	Using the Ada Cluster	173
	LSF Job Scheduler	34
	Using the Terra Cluster	93
	SLURM Job Scheduler	64
	Data Management Practices	34

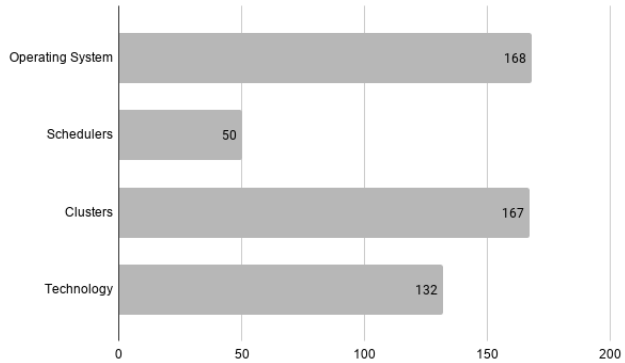


Figure 7. Distribution of downloaded primer course materials by themes for Fall 2020.

4.6 Efficacy of Online Videos

Recordings of the hour-long Primer sessions are offered on the Texas A&M HPRC YouTube Channel. The channel hosts 34 instructional videos in short (five-minute), medium (45-to-60-minute), and long (two-hour) durations that have amassed over 2,500 views. All videos are indexed (bookmarked) and checked for the accuracy of the closed captioning. Since its launch in late April 2020, the channel has gained over 139 subscribers as of November 2020. Complementing the Primer videos are short (less than 5-minute) videos on topics such as how to access the HPRC clusters. An analysis of the videos shows that learners are more likely to gravitate toward shorter videos as opposed to more detailed videos. A detailed breakdown of viewership statistics is presented in Table 3. Viewership and subscription data were collected at the time of writing this manuscript to show the differential impact of vlength versus usage.

Table 3. List of course material downloaded by individuals for HPRC Primers offered in Fall 2020.

Courses	Learner Downloads
Introduction to Linux	168
Introduction to the Ada Cluster	83
Introduction to the Terra Cluster	52
Data Management Practices	62
Introduction to HPRC	32
Jupyter Notebooks on OOD	70
LSF Job Scheduler	10
SLURM Job Scheduler	40

In keeping with the philosophy of Open Science, all materials are available free-of-charge for use and adoption to the larger research computing community. The encouraging viewership of YouTube videos by the Research community, while heartening, revealed that a significant portion of our viewership came from outside the United States. Approximately a third of our viewers used the closed captioning service on these videos. Figure 8 shows the viewership trends for the shorter 5-minute videos versus the longer Primer-recorded (1-hour) videos on YouTube. The total viewership minutes per category, calculated by multiplying the total viewership of a video by the duration of the video, remains approximately the same in each category. As such, one may hypothesize that while shorter videos are more likely to reach out to a broader audience, the longer one-hour videos serve an important purpose by helping learners who are interested in a slightly deeper dive into the topic. We once again note that courses on cluster usage get the most viewership.

5. CONCLUSIONS

The data collected as part of this study show that the Primer format could be a suitable pedagogical approach that enhances learner engagement, makes the materials more relatable, and leverages peer-learning and peer-led-discussion approaches while scaling back on staff time. The courses in conjunction with the online communities, pre-staged materials, and online videos showed increased participation from learners and were a better fit for an online-only educational platform. It is heartening to note that despite these viewership of materials on YouTube and availability of course materials on our website, the Primers consistently engaged new learners, and participation remained high in the Summer and Fall semesters of 2020.

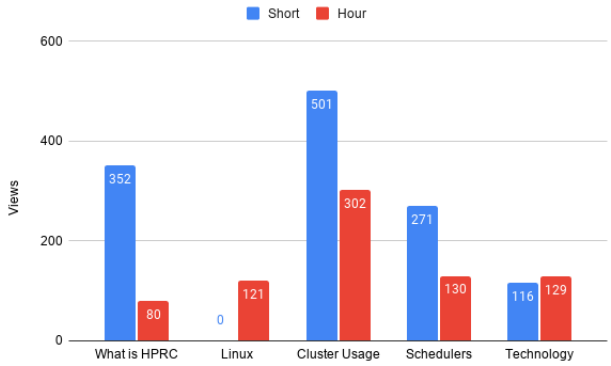


Figure 8. Distribution of short video (<5 minute) versus longer (1-hour) videos in topic areas.

6. CHALLENGES AND FUTURE WORK

While the Primer course format is better suited for an exclusively-online instruction-dependent world, challenges remain. Getting participants to complete evaluations that gauge the effectiveness of our program is a challenge. While we traditionally had over 70% of participants respond to surveys, we saw responses drop to 10% upon switching to Google Forms. We have since switched to polling participants over Zoom and observe upwards of 50% participation. We note that Zoom is a limited medium compared to the richness of Google Forms. Our questions today are limited to:

1. Did you attend this course for research, personal, and/or class needs?
2. Did the course meet your objectives?
3. Would you like future courses to be more generalized, specialized, or both?

Moving to online-only usage of resources encouraged us to explore mechanisms to improve and scale our training operations. The last couple of semesters have shown us the strengths of adopting an online-only approach. As a nation, we have collectively observed that training over online resources has its own share of questions related to access, inclusivity, equity, and diversity [6]. While we celebrate the expanded reach enabled by offering training over the Internet, we sadly realize that students with limited access to technology and reliable Internet connectivity are in danger of being left behind. Today, HPRC is experimenting with a new online pedagogical approach, called the “technology labs” [1]. These labs are geared toward placing the participants in a real-world scenario on entry. At the time of writing this manuscript, it is hard not to acknowledge that we stand at the crux of a “twindemic” that could well progress the remote-only settings to the Summer of 2021 or beyond. Indeed, at Texas A&M University, the Spring 2021 semester has been adjusted. Based on the usage characteristics, we plan to offer these courses in an online setting into the foreseeable future.

7. SUPPORTING INFORMATION

All training materials used in this study are available to the community via the Texas A&M HPRC website at https://www.hprc.tamu.edu/training. Videos and course recordings may be accessed via the Texas A&M HPRC channel on YouTube. The community is invited to join the SWEETER Slack workspace at https://hprc.tamu.edu/sweeter. Surveys and review exercises that will be developed as part of this longitudinal study may be requested from the author. Please send us feedback about your adoption experience via an email to help@hprc.tamu.edu.

8. ACKNOWLEDGEMENTS

The authors thank the staff at Texas A&M HPRC for assisting with the research related to this study. We gratefully acknowledge support from the National Science Foundation (NSF). We thank the NSF for award #1649062, "NSF Workshop: Broadening Participation in Chemical and Biological Computing at the Early Undergraduate Level," award #1730695, "NSF CyberTraining: CIP: CiSE-ProS: Cyberinfrastructure Security Education for Professionals and Students," award #2019136, "NSF CC* BRICCs: Building Research Innovation at Community Colleges," and award # 1925764, "NSF CC* SWEETER: South West Expertise in Expanding Training Education and Research."

9. REFERENCES

[1] Texas A&M High Performance Research Computing website. URL: https://hprc.tamu.edu

[2] D. K. Chakravorty, M. Pennings, H. Liu, Z. Wei, D. M. Rodriguez, Levi T. Jordan, D. F. McMullen, N. Ghaffari, and S. D. Le. "Effectively Extending Computational Training Using Informal Means at Larger Institutions," Journal of Computational Science Education 2018, 40–47 DOI: 10.22369/issn.2153-4136/10/1/7

[3] D. K. Chakravorty, M. Pennings, H. Liu, Z. Wei, D. M. Rodriguez, L. T. Jordan, D.F. McMullen, N. Ghaffari, S. D. Le, D. Rodriquez, C. Buchanan, and N. Gober. "Evaluating Active Learning Approaches for Teaching Intermediate Programming at an Early Undergraduate Level," Journal of Computational Science Education 2018, 61–66 DOI 10.22369/issn.2153-4136/10/1/10

[4] D. K. Chakravorty, D. F. McMullen, N. Gober, J. H. Seo, M. Bruner, and A. Payne. "Using Virtual Reality to Enforce Principles of Cybersecurity," Journal of Computational Science Education 2018, 81–87 DOI 10.22369/issn.2153-4136/10/1/13

[5] D. K. Chakravorty, M. Pennings, H. Liu, X. Thomas, D. M. Rodriguez, and L.M. Perez. "Incorporating Complexity in Computing Camps for High School Students – A Report on the Summer Computing Academy Program at Texas A&M University," Journal of Computational Science Education 2020, 11, 1 12–20 DOI https://doi.org/10.22369/issn.2153-4136/11/1/3

[6] D. K. Chakravorty, M. T. Pham "Evaluating the Effectiveness of an Online Learning Platform in Transitioning from High Performance Computing to a Commercial Cloud Computing Environment," Journal of Computational Science Education 2020, 11, 1, 93–99.

[7] Diverse Education News: Impact on Disadvantaged students https://diverseeducation.com/article/189597/

[8] K. Saichaie, D. C. Brooks, P. Long, R. Smith, R. Holeton, C. Meyers, A. Finkelstein, S. Dugdale, "7 Things You Should Know About Research on Active Learning Classrooms," in ELI 7 Things You Should Know, Educause Learning Initiative (ELI), 2017.

[9] P. Baepler, J. D. Walker, D. C. Brooks, K. Saichaie, C. I. Petersen, B. A. Cohen, "A Guide to Teaching in the Active Learning Classroom: History, Research, and Practice," Stylus Publishing, ISBN-13: 978-1620363003, 2016.

[10] Student-Centered Active Learning Environment with Upside-down Pedagogies: http://scaleup.ncsu.edu/

[11] M. Prince, "Does active learning work? a review of the research," Journal of engineering education, vol. 93, no. 3, pp. 223–231, 2004.

[12] J. Parsons and L. Taylor, "Improving student engagement," Current issues in education, vol. 14, no. 1, 2011.

[13] Larkin, M. (2002). Using scaffolded instruction to optimize learning. http://www.vtaide.com/png/ERIC/Scaffolding.htm

A. REPRODUCIBILTY INDEX

All training materials are available via our website at https://hprc.tamu.edu. Videos are available free-of-cost via the Texas A&M HPRC channel on YouTube. Surveys, analytics for Slack and YouTube, and review exercises that will be developed as part of this longitudinal study may be requested from the author. Please send us feedback about your adoption experience, questions, and requests to join our training Slack (SWEETER Slack) via an email to help@hprc.tamu.edu.

Transitioning Education and Training to a Virtual World, Lessons Learned

S. Charlie Dey[1]
charlie@tacc.utexas.edu

Victor Eijkhout[1]
eijkhout@tacc.utexas.edu

Lars Koesterke[1]
lars@tacc.utexas.edu

Je'aime Powell[1]
jpowell@tacc.utexas.edu

Susan Lindsey[1]
slindey@tacc.utexas.edu

Rosalia Gomez[1]
rosie@tacc.utexas.edu

Brandi Kuritz[1]
bkuritz@tacc.utexas.edu

Joshua Freeze[1]
jfreeze@tacc.utexas.edu

ABSTRACT

Interaction is the key to making education more engaging. Effective interaction is difficult enough to achieve in a live classroom, and it is extremely challenging in a virtual environment. To keep the degree of instruction and learning at the levels our students have come to expect, additional efforts are required to focus efforts on other facets to motivate learning, whether the learning is relative to students in our academic courses, student internship programs, Summer Institute Series, or NSF/TACC's Frontera Fellowship Program. We focus our efforts in lecturing less and interacting more.

Interaction now comes in the form of:

- taking a more casual approach to teaching
- gamifying the classroom
- giving students more choices regarding the path the curriculum follows
- constantly relating the educational material to the students' current and future projects
- flipping the lessons where the students apply concepts in class
- integrating peer programming groups
- taking advantage of all the technology options at our disposal

We have refocused our efforts on interacting with students using alternative means. As a result, we have built a successful academic and training curriculum, making our virtual classrooms more engaging and more collaborative, thus delivering a better educational experience.

This paper will detail those efforts, what worked well, what aspects needed adjusting, how those adjustments were implemented, and how those efforts were received by our students.

[1] Texas Advanced Computing Center (TACC), Austin, TX

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/3

1. RATIONALE

TACC has a wide array of training and educational offerings, aimed at everyone from IT professionals to research scientists to graduate and undergraduate students to high school and elementary school students. Our approach to training and education is very similar no matter the audience, to build a sense of community.

Teacher-student interaction is important. The more interaction there is, the stronger the learning experience can be. To create a positive learning environment, capable of meeting all of the educational needs, teachers must build a positive relationship with their students. Positive teacher-student interaction can be defined by shared acceptance, understanding, engagement, trust, respect, care, and cooperation. In a face-to-face classroom, this is a much simpler task. Trying to build a community with students online, can be more of a challenge.

For this reason, TACC took a step back on our traditional approach, and through an iterative process, reimagined the classroom while still providing representation, recognition, understanding, intimacy, expectation, respect, care, and cooperation to bring the aforementioned community together online. By taking a more casual approach to teaching with multiple instructors and then integrating aspects of gamification, loosening the curriculum, applying lessons to current events, spending more class time focused on applying learned concepts versus lecturing on concepts, breaking the class into groups to make learning more intimate, and using all available resources and technologies into our classes, we were able build the needed teacher-student interaction to create a positive learning environment.

2. DISCUSSION

2.1 Casual Classroom

When standing in front of a classroom, you have an automatic lead position where all students' eyes are upon you. It is easy to tell if students are paying attention or not, to read the classroom on how well they are understanding the material, when you should ask leading questions, when another example is in order, or when the students have reached their saturation point. This is not an easy task in the virtual classroom.

The virtual classroom requires a stronger relationship with students. By taking a casual or more informal approach to teaching, students feel more comfortable to ask and answer questions and ask for clarification. TACC implements this technique by utilizing multiple instructors in the classroom. The instructors work off one another. As one instructor focuses on the context of the lecture, the

other instructors add to or highlight important concepts. This also allows the secondary instructors to conduct discussions and polls during the lecture to get a better read on how the class is absorbing the information, then bringing any items needing an immediate or more involved response to the attention of the primary instructor.

The secondary instructor also helps highlight any errors the primary instructor may make during any programming demonstrations. Keeping the comments and critiques light and in jest accomplishes three things. The errors in the code are brought to the students' attention and therefore more easily recognized in their own code. Students realize that everyone at every level makes mistakes. Lastly, students are more willing to open up, unmute their mics, and point out errors that the professor might have made, thereby making it so students are more willing to share their code and screens with the rest of the class and not be disconcerted about their error.

With a more casual approach to leading a class, having multiple instructors with secondary instructors leading discussions and polling the class during lectures, light conversations between the instructors, and pointing out errors during live coding examples, students are able to form a bond with the teachers and are more adept to learn and participate during class.

2.2 Gamifying the Classroom

Gamification is defined as the use of activities and external rewards to encourage motivation in non-game contexts. It is designed to increase a person's experience and engagement with a course, goal, or system. It helps bring a level of competitiveness and active participation to class, motivating students to learn. Gamification motivates people by making the learning process more enjoyable and engages the student more with a course.

Gamifying tasks are implemented in our courses and hackathons in particular. Sometimes the tasks are directly related to the learning process, and sometimes they are meant as an icebreaker to encourage students to open up. Some gamification tasks come as badges that students collect as they progress through the material and coding tasks. Other times, badges can be given to students who answer questions, ask questions, or share their screens and code with the rest of the class in order to get help debugging efforts. Students are also encouraged to hand out virtual badges to their peers, if a peer helps answer a question asked over chat.

Bringing healthy competition to the classroom can have positive effects, especially if it is a competition that helps move the class forward as a group instead of a competition between peers. This brings about a shared experience with mutual respect and cooperation.

2.3 Open Curriculum In The Classroom

Open curricula, where the student is open to take relevant courses in differing order with a faculty mentor, are normally associated with directing a field of study. This idea has been applied to our educational activities. Though each course and training activity conducted at TACC has set curriculum goals that need to be met, the route we take to achieve these goals can be fluid.

The experience of "open curriculum" classes allows students some freedom in how the course is directed and gives students some ownership regarding the course material. This helps create a culture of learning in which students display motivation, innovation, and self-direction. An open curriculum promotes independent thinking and creative problem-solving. We implement this with different types of projects and exercises during class time. Depending on how the students react to the different challenges presented, the next lesson can be modified to focus more or less on how the concept was received.

2.4 Relating Material to Current Events

Students need a personal connection to the material, bridging the new information with previously-acquired knowledge, or directly applying new knowledge with current world events. One of the keys to effective teaching is keeping the course material and projects relevant. It keeps the learning experience engaging. After key concepts are covered, the projects proposed for students to investigate are ones relative to current world topics, such as disease propagation, climate change, gerrymandering, and traffic patterns in urban settings.

By keeping projects directly tied to the world around them, students stay motivated, learning and retaining more of the material. Creative and critical thinkers work for work's own sake. They are driven by the desire to understand how the current world is progressing and where the world is heading. By being able to apply what they learn directly to real world issues, students achieve a better understanding of the role computational and data science plays in day-to-day issues.

2.5 Flipping the Classroom

Flipping the classroom is a response to the idea that class time can be used to engage students in learning through focused techniques, rather than through delivering lectures alone. By blending normal lectures with more student-centered learning strategies, instructors have more opportunities to deal with mixed levels of student comprehension, attend to any student difficulties, and differentiate learning preferences during in-class time. The amount of flipping varies from course to course, but for the majority of our training offerings, our courses integrate 20–30 minutes of hands-on activities through in-class exercises and less time lecturing. This allows us to turn the class into an active learning environment.

After key concepts of the course material are learned, the focus is turned to applying those concepts through in class challenges or group exercises. As student reactions to the challenges are observed, instructors can change the direction of the class based on these observations. This allows instructors to verify students are able to understand and apply their learned knowledge before moving to the next concept. These concepts are then built upon each other leading to a final project.

The flipped classroom environment allows students to better relate the material to previous lessons and apply the material to the future lessons. It also allows the instructors to make sure that the students have a good grasp of the content and are capable of working towards the major projects. This better engages the students and the instructors for a more productive classroom experience. The flipped classroom puts more control into students' hands regarding their own learning processes.

2.6 Peer Groups and Paired Programming

Another teaching approach implemented in our educational efforts is the use of peer groups and pair programming. Pair programming allows students to learn from one another and reduces the risk of going down an irrelevant path in trying to solve a problem. When properly implemented, this method allows students to learn from their peers. A student who may not be grasping a certain concept may achieve a better understanding when paired with a student who does. Due to the group environment, it also reduces the amount of effort required by the instructors to make sure all students are at an equal level.

Taking advantage of technologies available, students are broken into pairs and assigned to a virtual space to communicate and collaborate within. Instructors will occasionally enter these spaces to see how students are progressing and answer any questions that may have arisen.

Another group activity in peered programming that TACC implements is "programming out loud." This technique enables students to clearly understand and articulate the complexities the coding tasks may require. They follow the instructor's lead to build an understanding of the coding steps involved to solve the exercise and then replicate the technique in their groups. The goal of this strategy, and all of our strategies, is to better engage students and instructors.

2.7 Technology Options

Classroom interaction is very important. To achieve this properly, every technical resource that enhances that interaction should be implemented. Technology can be used to enhance the dynamic between students and instructors. A combination of different technologies may be needed to sufficiently support a positive dynamic.

At TACC, we've used a combination of collaborative tools to enable students and instructors to better communicate. Zoom is used to conduct our training events, with breakout rooms for peer and paired programming sessions. Slack is utilized for students to engage with other students and instructors outside of normal classroom hours. Git and Repl.it are used for collaborative coding, while SSH is used to communicate with our classroom servers. This combination of tools allows students to be better connected to each other as well as the instructors, all in an effort to build a cohesive online, virtual community.

3. LESSONS LEARNED, CONCLUSIONS

When the class went virtual, a new approach to the classroom was very necessary. Progressive approaches to managing learning needed to be implemented in a short amount of time to best secure an accepting, understanding, and engaging environment where trust, respect, care, and cooperation exist. The approach we took at TACC included:

- taking a more casual approach to teaching
- gamifying the classroom
- giving students more choices regarding the path the curriculum follows
- constantly relating the educational material to the students current and future projects
- flipping the lessons where the students apply concepts in class
- integrating peer programming groups
- taking better advantage of all the technology options at our disposal

Unfortunately, there were some drawbacks during implementation, and it took multiple iterations to balance out the approach. Too casual of a classroom environment, and students lost some discipline and felt less pressure to get work in on time. We had to make sure that instructors maintained proper discipline through grading the material and holding students to due dates with some leniency. Gamification had a slight negative consequence with certain students who felt upstaged by others. To counteract this, we made sure that instructors kept lines of communication open with students who felt upstaged and made sure all students had an opportunity to interact in class. An open curriculum without a proper introduction to the concepts can slow down the course. Spending more time on key concepts actually can speed up the class because there are less interruptions on repeating core material. Occasionally the material can be dry, and making the material relevant is not always an option, but the concepts still need to be covered, and there is not much that can be done, aside from following up a dry lecture with some interactive lessons. Regarding flipping the classroom and peer programming groups, new and inexperienced programmers might require more of a traditional lecture from the instructor to better understand some concepts. We have found that by involving instructors within the peer programming groups, we can have "micro" lessons and still have the benefits of a flipped classroom and peer/paired programming.

Once a balance was attained, we were able to move our classes forward. We built an environment that encouraged interaction between students and instructors, leading to a stronger learning experience. This has led to a successful virtual training and education program.

4. REFERENCES

[1] Brame, Cynthia. “Flipping the Classroom.” Vanderbilt University, Vanderbilt University, 26 Mar. 2020, cft.vanderbilt.edu/guides-sub-pages/flipping-the-classroom/.

[2] Cox, Janelle. “10 Ways to Keep Your Class Interesting.” ThoughtCo, www.thoughtco.com/ways-to-keep-your-class-interesting-4061719.

[3] Erenli, Kai. “The Impact of Gamification — Recommending Education Scenarios.” International Journal of Emerging Technologies in Learning (IJET), Publisher: International Journal of Emerging Technology in Learning, Kassel, Germany, 31 Jan. 2013, www.learntechlib.org/p/45224/.

[4] Wilson, L.A and S. C. Dey, "Computational Science Education Focused on Future Domain Scientists," 2016 Workshop on Education for High-Performance Computing (EduHPC), Salt Lake City, UT, 2016, pp. 19–24, doi: 10.1109/EduHPC.2016.008.

[5] “How Scientists Use Slack.” Nature News, Nature Publishing Group, www.nature.com/news/how-scientists-use-slack-1.21228.

[6] “Motivation in Education: What It Takes to Motivate Our Kids.” PositivePsychology.com, 1 Sept. 2020, positivepsychology.com/motivation-education/

Bringing GPU Accelerated Computing and Deep Learning to the Classroom

Joseph Bungo
NVIDIA Corporation
Austin, Texas
jbungo@nvidia.com

Daniel Wong
University of California, Riverside
Riverside, California
danwong@ucr.edu

ABSTRACT

The call for accelerated computing and data science skills is soaring, and classrooms are on the front lines of feeding the demand. The NVIDIA Deep Learning Institute (DLI) offers hands-on training in AI, accelerated computing, and accelerated data science. Developers, data scientists, educators, researchers, and students can get practical experience powered by GPUs in the cloud. DLI Teaching Kits are complete course solutions that lower the barrier of incorporating AI and GPU computing in the classroom. The DLI University Ambassador Program enables qualified educators to teach DLI workshops, at no cost, across campuses and academic conferences to faculty, students, and researchers. DLI workshops offer student certification that demonstrates subject matter competency and supports career growth. Join NVIDIA's higher education leadership and leading adopters from academia to learn how to get involved in these programs.

By attending this talk, you will learn:

- How educators can access Teaching Kits with curriculum materials in accelerated computing, Deep Learning, and robotics.
- How to access free online training, certification, and cloud access to GPUs for teachers and students.
- An overview of the NVIDIA DLI and University Ambassador Program.
- How the Ambassador Program fits into larger programs that support teaching.
- Real examples of leading academics leveraging Teaching Kits and Ambassador workshops in the classroom.

Keywords

Hands-on learning, Training, HPC education, Deep learning, Machine learning, Artificial intelligence, GPU, Data science, Parallel computing, Accelerated computing

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/4

XSEDE EMPOWER: Engaging Undergraduates in the Work of Advanced Digital Services and Resources

Aaron Weeden
Shodor Education Foundation
Durham, NC
aweeden@shodor.org

ABSTRACT

To address the need for a diverse and capable workforce in advanced digital services and resources, the Shodor Education Foundation has been coordinating an undergraduate student program for the Extreme Science and Engineering Discovery Environment (XSEDE). The name of the program is EMPOWER (Expert Mentoring Producing Opportunities for Work, Education, and Research). The goal of the program is to engage a diverse group of undergraduate students in the work of XSEDE, matching them with faculty and staff mentors who have projects that make use of XSEDE services and resources or that otherwise prepare students to use these types of services and resources. Mentors have coordinated projects in computational science and engineering research in many fields of study as well as systems and user support. Students work for a semester, quarter, or summer at a time and can participate for up to a year supported by stipends from the program, at different levels depending on experience. The program has run for 11 iterations from summer 2017 through fall 2020. The 111 total student participants have been 28% female and 31% underrepresented minority, and they have been selected from a pool of 272 total student applicants who have been 31% female and 30% underrepresented minority. We are pleased that the selection process does not favor against women and minorities but would also like to see these proportions increase. At least one fourth of the students have presented their work in articles or at conferences, and multiple credit the program with moving them towards graduate study or otherwise advancing them in their careers.

Keywords

Undergraduate student programs, Advanced digital services and resources, HPC, Computational science, Data science, Internships

1. INTRODUCTION

To address the need for a diverse and capable workforce in advanced digital services and resources, the Shodor Education Foundation [8] has been coordinating an undergraduate student program since summer 2017 for the Extreme Science and Engineering Discovery Environment (XSEDE) [10]. The name of the program is EMPOWER (Expert Mentoring Producing Opportunities for Work, Education, and Research). The goal of the program is to engage a diverse group of undergraduate students in the work of XSEDE, matching them with faculty and staff mentors who have projects that make use of XSEDE services and resources or that otherwise prepare students to use these types of services and resources. Mentors have coordinated projects in computational science and engineering research in many fields of study as well as systems and user support. Students work for a semester, quarter, or summer at a time and can participate for up to a year supported by stipends from the program. Students participate at one of three different levels depending on their existing experience: Learners are trained on new skills and knowledge, Apprentices apply their skills and knowledge to supervised tasks, and Interns work more independently.

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/5

2. METHODS

2.1 Recruiting Mentors and Students

We recruit mentors and students by promoting the program in XSEDE newsletters [12]; engaging with the Campus Champions community [3], who participate themselves and/or spread the word on their own campuses; and with help from the XSEDE Broadening Participation team [1], who engage with historically underrepresented faculty and students through conference and campus visits.

2.2 Receiving and Reviewing Applications

Shodor maintains an in-house application website [9] and database, originally developed to coordinate workshop registrations for the National Computational Science Institute [7]. Student and mentor application forms for EMPOWER have been developed based on those of the Blue Waters Student Internship Program [4]. The mentor form requests the mentor's affiliation/role with XSEDE, the project title and summary, student job description, use of XSEDE resources, contribution to the community, start and end dates, location, participation level, training plan, number of students the mentor can support, student names (if already identified), and additional student prerequisites and qualifications. The student application form requests GPA; subject areas studied; subject interests; relevant courses and grades; relevant work and internship experiences; career goals; interests in contributing to XSEDE; and experiences with mathematics, computing, application software, programming languages, Unix/Linux, parallel computing, visualization, data science, and machine learning.

When students apply, they can optionally indicate a gender (Male or Female) and/or an ethnicity (African-American, American Indian or Alaskan, Asian or Pacific Islander, Hispanic, Caucasian, or Middle Eastern). For our analysis, we consider the following ethnicities to be underrepresented minorities: African-American, American Indian or Alaskan, and Hispanic.

As applications come in, the program coordinator reviews the submissions, looking for issues that may need to be addressed, such as start and end dates that do not line up with the dates of the

program, or mentors indicating preferred students who have not themselves submitted applications (or vice versa for students and their preferred mentors). The coordinator corresponds with mentors to identify which student applications should be matched with which mentor applications during the review process.

With the goal of avoiding biases during the review process, an anonymous review is conducted. The coordinator prepares a Portable Document Format (PDF) file for each mentor/student application pair in which names, pronouns, genders, ethnicities, institutions, locations, and URLs are removed. The process of preparing these PDF files is partially automated and partially manual. A script written in the PHP programming language automatically searches for gender pronouns and replaces them with an indicator that a gender pronoun has been removed. The script also allows for other search strings to be specified by the coordinator manually (i.e. those indicating names, pronouns, genders, ethnicities, institutions, locations, and URLs). These search strings are automatically replaced with indicators that the string has been removed. The coordinator reads each application looking for search strings to include and types them into a spreadsheet. The spreadsheet is set up to construct a URL that is used to run the PHP script. A separate Bash script requests each URL and downloads the resulting output into PDF files. The result is a collection of anonymized PDF files that are given to reviewers.

The coordinator recruits three reviewers for each PDF file from the community: primarily from the Campus Champions, the XSEDE Community Engagement and Enrichment staff, and former mentor participants of EMPOWER. Reviewers score each PDF using a Google Form that presents ten prompts on a 1–5 scale: 1) the project contributes to the work of XSEDE, 2) the proposed level of participation (Learner, Apprentice, or Intern) is appropriate, 3) the training plan is appropriate, 4) the project is suitably scoped given the start and end dates, 5) the mentor is likely to do a good job supporting the student, 6) the student is new to the XSEDE community, 7) the project is a good fit for the student's expressed interests and skill level, 8) the student is likely to do a good job in the project, 9) the student's participation will advance them in their career path as a user or facilitator of advanced digital services and resources, and 10) the match of mentor, student, and position should be selected for the program.

For a given PDF and reviewer, a weighted score is calculated based on the reviewer's scores for each prompt using the formula below.

$$4*(P10) + 3*(P1+P5+P8+P9) + 2*(P2+P3+P4+P7) + P6$$

A final score for each PDF is calculated using an iterative model that takes into account the weighted scores and reviewer leniencies, based on the Differential Model described in [5]. PDF files are ranked by this final score, reviewer comments and other considerations (such as previous participation) are taken into account, and final selections are made by the coordinator.

2.3 Training Students

The student participants of the summer 2017 and summer 2018 programs were invited to participate in the Petascale Institute [11], a two-week training event conducted by Shodor for the undergraduate students of the Blue Waters Student Internship Program. Students from EMPOWER who were interested and available attended this training, where they learned how to apply parallel and distributed computing concepts to computational simulation and modeling using the Blue Waters supercomputer [2] as the example architecture. Funding for the Petascale Institute ended in 2018.

Outside of the Petascale Institute, Shodor has so far not conducted training for the EMPOWER students. Instead, the mentor participants in the program provide training to the students whom they are mentoring. The modes of training have varied and include formal courses, informal lectures and one-on-one tutoring, and self-learning using online resources.

2.4 Reviewing Student Work

EMPOWER students complete monthly progress reports in which they describe accomplishments, issues, and what they plan to accomplish before the next report. In reading these reports, the EMPOWER coordinator looks for mentions of publications or presentations that have been produced or prepared by the students, as well as other highlights of significant accomplishments. These highlights are reported to XSEDE and the National Science Foundation as evidence of impact of the program. Students receive their stipend payments once they have completed all of their required monthly reports.

The EMPOWER coordinator also occasionally conducts optional small surveys of the students and mentors to obtain specific information, such as the number of hours per week that mentors are putting into mentoring their students or comments about the impact of the program on students' career paths.

3. RESULTS

There have been 11 iterations of the EMPOWER program from summer 2017 through fall 2020 (one each per fall, spring, and summer). The 111 total student participants have been 28% female and 31% underrepresented minority, and they have been selected from a pool of 272 total student applicants who have been 31% female and 30% underrepresented minority.

At least one fourth of the students have published articles or presented at conferences about their EMPOWER work. We have also heard anecdotally from students about the impact of the program on their career paths, including highlights such as being selected as a college's valedictorian, securing cooperative education, receiving offers from Research Experience for Undergraduate programs, deciding what to study in graduate school, impressing job recruiters, improving public speaking skills, leading and tutoring new student researchers, and networking and meeting new collaborators.

In an optional survey, mentors reported putting in an average of 2–10 hours of mentoring per week per student.

4. DISCUSSION

The program has had essentially equal proportions of female student applicants and female student participants, as well as essentially equal proportions of underrepresented minority student applicants and underrepresented minority student participants, which suggests the selection process does not favor against female or minority students. We would like to increase the proportions of female and minority applicants and participants. We will change the application form to use more standard and inclusive categories. For example, we will update the optional "gender" prompt in our account creation form to provide a "Non-binary" option as well as a "Prefer to Self-describe" option. We will also use the race and ethnicity categories from [6] as a starting point for updating our form's "ethnicity" prompt, as well as taking into account inclusivity research about the categories. We would also like to do a more thorough study of the impact of the program on student career paths and do other analyses of our data. The EMPOWER program will run for five more iterations through summer 2022.

5. ACKNOWLEDGMENTS

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562.

Robert M. Panoff set the vision of EMPOWER as executive director of Shodor. Jennifer K. Houchins helped coordinate the program in its first few iterations. The XSEDE Campus Champions, External Relations, and Broadening Participation teams have been immensely helpful in recruiting for the program and reviewing applications. We are grateful to everyone who has participated in the program as a mentor, student, or application reviewer.

6. REFERENCES

[1] Linda Akli. 2018. XSEDE: Tackling Diversity and Inclusion in Advanced Computing. *Computing in Science & Engineering* 20, 3 (May–Jun. 2018), 71–72. DOI: https://doi.org/10.1109/MCSE.2018.03202635.

[2] Brett Bode, Michelle Butler, Thom Dunning, Torsten Hoeer, William Kramer, William Gropp, and Hwu Wen-Mei. 2013. The Blue Waters Super-System for Super-Science. *Contemporary High Performance Computing: From Petascale toward Exascale*, 339–366. CRC Press.

[3] Marisa Brazil, Dana Brunson, Aaron Culich, Lizanne DeStefano, Douglas M. Jennewein, Tiffany Jolley, Timothy Middelkoop, Henry J. Neeman, Lorna I. Rivera, Jack A. Smith, and Julie A. Wernert. 2019. Campus Champions: Building and sustaining a thriving community of practice around research computing and data. In *PEARC '19: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning)* 78 (July 2019), 1–7. DOI: https://doi.org/10.1145/3332186.3332200

[4] Patricia Jacobs, Phillip List, Mobeen Ludin, Aaron Weeden, and Robert M. Panoff. 2014. The Blue Waters Student Internship Program: Promoting Competence and Confidence for Next Generation Researchers in High-Performance Computing. *2014 Workshop on Education for High Performance Computing* (Nov. 2014), 49–55. DOI: https://doi.org/10.1109/EduHPC.2014.6

[5] Hady W. Lauw, Ee-Peng Lim, and Ke Wang. 2007. Summarizing Review Scores of "Unequal" Reviewers. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, April 26–28, 2007, Minneapolis, MN, 539–544. DOI: https://doi.org/10.1137/1.9781611972771.58

[6] United States Office of Management and Budget (OMB). 1997. *Revisions to the Standards for the Classification of Federal Data on Race and Ethnicity*. Notice 62 FR 58782. 9 pages. Document number 97-28653. Retrieved from https://www.federalregister.gov/documents/1997/10/30/97-28653/revisions-to-the-standards-for-the-classification-of-federal-data-on-race-and-ethnicity/.

[7] Shodor. 2020. The National Computational Science Institute. Retrieved from http://computationalscience.org/.

[8] Shodor. 2020. The Shodor Education Foundation, Inc. Retrieved from http://www.shodor.org/.

[9] Shodor. 2020. XSEDE EMPOWER Program. Retrieved from http://computationalscience.org/xsede-empower/.

[10] John Towns, Timothy Cockerill, Maytal Dahan, Ian Foster, Kelly Gaither, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D. Peterson, Ralph Roskies, J. Ray Scott, and Nancy Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Computing in Science & Engineering* 16, 5 (Sept.–Oct. 2014), 62–74. DOI: https://doi.org/10.1109/MCSE.2014.80

[11] Aaron Weeden, Colleen Heinemann, Skylar Thompson, Cameron Foss, Mobeen Ludin, and Jennifer K. Houchins. 2019. The Blue Waters Petascale Institute: Longitudinal Impact and Assessment-Driven Development of an Intensive, Hands-on Curriculum for Teaching Applications in HPC. In *PEARC '19: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning)* 85 (July 2019), 1–8. DOI: https://doi.org/10.1145/3332186.3337771

[12] XSEDE. 2020. Impact. Retrieved from https://www.xsede.org/impact/.

Pawsey Training Goes Remote: Experiences and Best Practices

Ann Backhaus[1]
ann.backhaus@pawsey.org.au

Sarah Beecroft[1]
sarah.beecroft@pawsey.org.au

Lachlan Campbell[1]
lachlan.campbell@pawsey.org.au

Maciej Cytowski[1]
maciej.cytowski@pawsey.org.au

Marco De La Pierre[1]
marco.delapierre@pawsey.org.au

Luke Edwards[1]
luke.edwards@pawsey.org.au

Pascal Elahi[1]
pascal.elahi@pawsey.org.au

Alexis Espinosa Gayosso[1]
alexis.espinosa@pawsey.org.au

Yathu Sivarajah[1]
yathunanthan.sivarajah@pawsey.org.au

ABSTRACT

The Pawsey Supercomputing Centre training has evolved over the past decade, but never as rapidly as during the COVID-19 pandemic. The imperative to quickly move all training online — to reach learners facing travel restrictions and physical distancing requirements — has expedited our shift online. We had planned to increase our online offerings, but not at this pace or to this extent.

In this paper, we discuss the challenges we faced in making this transition, including how to creatively motivate and engage learners, build our virtual training delivery skills, and build communities across Australia. We share our experience in using different learning methods, tools, and techniques to address specific educational and training purposes. We share trials and successes we have had along the way.

Our guiding premise is that there is no universal learning solution. Instead, we purposefully select various solutions and platforms for different groups of learners.

Keywords

Online training, Virtual training, Remote training, HPC training, Engagement, Interactivity, Containers, Visualization, Australia

1. INTRODUCTION

Pawsey offers Australian researchers a diverse range of training. We provide basic computer science concepts, through to introductory and intermediate supercomputing, cloud and visualization, to parallel programming courses, GPU hackathons, and customized training for specific scientific domains and/or groups. Basic UNIX/Linux skills were also taught to prepare attendees for the hands-on training activities.

Pawsey has offered in-person training for over 12 years, at universities and research institutions across Australia. We have also offered virtual training ad-hoc as well as hybrid training (a combination of face-to-face and virtual). While we have dabbled in different delivery modalities, our go-to approach has been in-person.

The advent of the COVID-19 pandemic dramatically changed Pawsey training. In January and February 2020, before COVID-19 restrictions in Australia, we had a fully (over)booked schedule of in-person user training, teacher professional development, student hands-on activities and on-site events, Internship Program events, and community outreach. However, in March and April 2020, physical distancing and Australian border restrictions and closures meant that all Pawsey in-person activities ceased. Pawsey staff were asked to work remotely.

Working off-site increased the complexity of creating new, online training, as we use a hands-on "whiteboarding" approach to training design and development.

Pawsey faced two main challenges when moving training online:

- Re-purposing training content. The existing two-day "roadshow," which included hundreds of PowerPoint slides and dozens of Carpentry-esque episodes, required re-purposing.
- Re-focusing Pawsey's trainers. In parallel with content re-creation, Pawsey trainers needed to develop or refine their virtual training skills, such as online engagement and community building. This was a non-trivial task, considering training is an add-on to the staff's main role of working with researchers on code optimization and Pawsey resource uptake.

Pawsey set out on its training change journey.

2. RE-PURPOSING TRAINING

Pawsey staff saw the requirement to re-purpose training content into a virtual format as an opportunity to improve the content, ensuring alignment with learning objectives and learning outcomes, and incorporating best practices in (virtual) learner interaction and engagement. In this section, we briefly describe how this process was implemented for an example "core" course, *Introduction to Nimbus*, later renamed *Using the Nimbus Research Cloud*.

[1] Pawsey Supercomputing Centre, Kensington WA, Australia

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/6

2.1 Introduction to Nimbus: In-Person

To start, we listed the episodes of the existing, in-person *Introduction to Nimbus* training (Table 1) to assess the flow of content at a high level. We also noted additional supporting materials, such as online documentation.

Table 1. *Introduction to Nimbus* in-person training episodes.

***Introduction to Nimbus* training (3 hours)**
Let's talk about cloud computing
Nimbus: Cloud computing at Pawsey
How does Nimbus work for users?
First steps: making keypairs
Simplified security and networking
Launching an instance
Attaching volume storage
Maintaining your instance
Using snapshots to save time

Next, a new staff member attended the in-person training, reviewed the open code resources, and scanned the documentation to see if they could successfully launch an instance. They could not. They noted confusion and contradiction about the overall flow (what to do first, second, third), asked questions about security and jargon, etc. This feedback aligned with another new team member's feedback, received informally, earlier.

2.2 Using the Nimbus Research Cloud: Virtual

The reviewer's input on the existing, in-person content opened productive discussion and debate about what should and should not be in an introductory Nimbus Cloud training.

After several iterations, the team arrived at a newly designed series of high-level steps based on the flow of a new user's tasks (Apply > Set Up > Use > Manage > Develop > Optimize > Retire) and accompanying, detailed flowcharts for each step. Figure 1 shows a working draft of the sample flowchart for the Set Up step.

From the flowcharts, the team outlined those topics that were core versus those that were advanced or non-essential to getting an instance up and running. Only the essential topics would be included in the foundational training.

Table 2 shows the revised, online episodes for the new, two-part *Using the Nimbus Research Cloud* training as well as the modularized video recordings derived from the "live" virtual conducts.

Table 2. Nimbus virtual training & modularized recordings.

Virtual conduct (3 hours)	**Short modularized recordings**
Management	Introduction to Nimbus What is a Nimbus instance?
Authentication	Log into the Nimbus dashboard Review the Nimbus dashboard Create keypairs to access the Nimbus instance
Instance Creation	Create a Nimbus instance
Instance Access	Access your Nimbus instance
Storage	Set up your instance storage
Data	Transfer data to/from your instance
Software	Manage instance software

Short, modularized (edited) recordings are available for viewing after a virtual training conduct. The entirety of the workshop is usually not viewable, as the large file size and lengthy duration make them difficult to readily access and use by learners.

2.3 Design Considerations & Approach to Moving Online

During the reconstruction, consideration was given to feedback areas, including too few learner activities (beside hands-on coding), too quickly progressing from basic to intermediate concepts, and unexplained use of jargon. These latter comments are common manifestations of "expert blind spots" or "expert awareness gaps" [1]. Having forgotten what it is like to be a novice learner — unfamiliar with the language or concepts of the topic — experts may inadvertently overlook explanations and teach at what seems to be "breakneck speed" to a new learner.

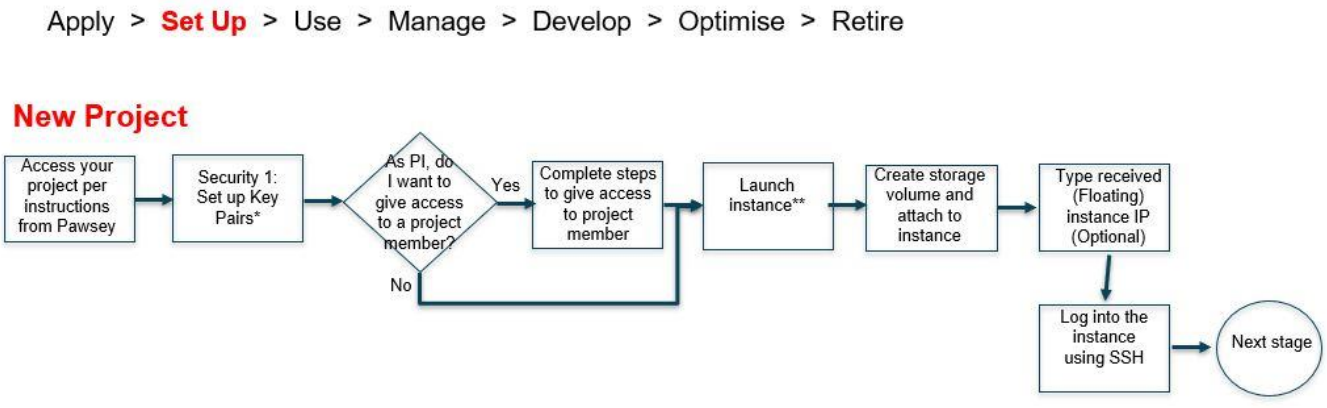


Figure 1. Early flowchart for the Set Up step for Pawsey's *Using the Nimbus Research Cloud* training.

A brief, but important, mention must be made here about Learning Outcomes (LOs) and backward design. LOs sit at the core of Pawsey training. Everything — training activities, discussions, coding, etc. — must contribute to and support the trainee in reaching LOs.

When designing a training course, we start with LOs, then work backwards to the activities, discussions, etc. In this way, we follow the backward design approach: LOs are formulated first and then course design follows, to determine how to assess (or validate / confirm learning) and how to teach (activities to use, etc.) [2].

During reconstruction, we rewrote LOs for each episode. Anything extraneous to those LOs were removed from content. New content was added as needed, with significant consideration and thought given to making the online content interactive and engaging [3], [4] and [5].

3. SUPPORTING SELF-GUIDED LEARNING

In July and August, Pawsey launched its suite of core online training to replace its national roadshow. For 90 minutes, each Monday morning, learners were offered free, significantly revised trainings on basic concepts in supercomputing, data, cloud, and visualization.

Pawsey offers these new virtual trainings in several ways: open enrolment, institutional requests, and domain requests. After a virtual conduct, we release videos of the events in short (5–15 minute), topical "chunks". This modularization enables learners to readily find a specific topic or watch an entire series of recordings (See Table 2).

Learners can access all training content (PowerPoint slides, code samples (GitHub), video recordings, etc.) from the Pawsey Training Portal [6].

4. CREATING NEW INTERMEDIATE AND ADVANCED CONTENT: CONTAINERS

With an online repository of core training accessible 24/7, Pawsey staff can turn their attention to the creation of intermediate and advanced training. Topics include such areas as effective use of compute infrastructure (parallel and accelerated computing) architecture, reproducible science (containers), and advanced visualization. Focusing on intermediate and advanced training was very difficult pre-COVID, when much of the trainers' time was spent traveling for in-person conducts of core trainings.

One advanced topic that Pawsey has focused on since moving entirely online is Containers. The *Using Containers in X* is a multi-day, webinar-workshop series focused on addressing specific needs for the targeted scientific domain.

This series has provided staff with rich opportunities to partner with domains, experiment with various online teaching techniques, and trial different peer-to-peer and community building approaches. At the time of writing, Pawsey has partnered with three domains to create tailored Containers training: computational fluid dynamics, bioinformatics, and radio astronomy. For each of these we offer core ("generic") webinars followed by domain-specific ("bespoke") workshops.

4.1 Creating a Baseline of Knowledge: Generic Container Webinars

To ensure that all learners have the requisite baseline to participate meaningfully in the hands-on workshops, we offer a three-part webinar series. In these 90–120-minute sessions, Pawsey trainers introduce concepts using illustration, discussion, and coding demonstrations. Individuals can practice simultaneously or while watching recordings or by using step-by-step instructions.

Table 3 shows the generic topics covered in the *Using Containers* webinar series.

Table 3. Core topics for container training (webinars).

Using Containers (Generic Webinars)
Introduction, Running applications in containers
Building containers, Setting up graphical applications
High performance containers (MPI, GPUs, I/O intensive)

During core training, learners can ask questions using the designated communications channel. We find that each domain has a "standard" or usual means of communicating, which we leverage. Through this reuse, we try to eliminate "noise" or learning distraction. When the group is large, we may use multiple communication channels, again to eliminate learner distraction. For example, we may use Zoom for learner-instructor communication and Slack for instructor-facilitator communication.

The webinar series is a prerequisite to the workshop series [7].

4.2 Building Expertise: Bespoke, Hands-on Container Workshops

The multi-day workshop series that follows the webinars makes up the tailored part of the training. In 2–3 three-hour sessions, learners are guided along a path from learning demonstrations to guided coding/application to extrapolated coding/application to mini-hack or BYO (Bring Your Own) code or pipelines. A forum discussion wraps up the series, providing an opportunity to summarize learning outcomes and share reflections and feedback.

While the overall workshop program has similarities across domains, in detail it is bespoke. The learners' journey — its goals, its starting and ending points, and its focus along the way — comes out of a close collaboration between Pawsey trainers, domain experts and facilitators, and learners themselves.

For example, in the Containers pre-workshop Expression of Interest, we collected key inputs. Potential attendees specified their profiles/roles/locations/institutions, levels of (self-reported) expertise in Containers, frequency of Container use, and "hot topics" of interest. They could enter "blue sky" topics; however, we also guided topic selection through a list pre-vetted by domain experts and facilitators. This latter approach ensured that we had on-hand facilitators skilled in the topics being offered.

Table 4 shows the listing of topics that came out of the Bioinformatics Expression of Interest.

Table 4. Hot topics for container training (workshops).

Using Containers (Bespoke Workshops)
Containerizing a full workflow
Designing and building container images
Collaborative and reproducible research with containers and workflow tools
Transitioning from Docker to Singularity
Using a virtual file system to encapsulate large numbers of I/O files
Setting up and configuring your own Singularity installation
Simplifying your user experience by "hiding" container syntax
Using RStudio, JupyterHub or other web platforms via containers

To construct an appropriate set of topics and groupings for the workshop, we asked the 16 facilitators to self-assess their expertise per hot topic. We then grouped attendees with the appropriate facilitators.

Table 5 shows the complexity of the groupings per session. These sessions reflect not only the matchings but also our training partner's goals for learning outcomes and community building.

Table 5. Virtual workshop details by grouping, focus, goals.

	Session 1	Session 2	Session 3
Grouping content	Learners and facilitators by institution (sub-grouping of skill level)	Learners by topic of interest mapped to facilitators with expertise (sub-grouping of skill level)	Learners by hot topics; facilitators pick topic per expertise and bring case study(s)
Focus	Detailed scripts and code / samples	Training content more guided, less prescriptive	Own pipelines, projects; problem solving
Goals	Significant guidance and scaffolding; early "wins" (for confidence); reinforce local community	Engage, motivate; community of shared interest; learners apply previous learning in a guided setting	Facilitator freedom to choose interaction; expand learner knowledge; wider community

We expected our biggest challenge to this live virtual training would be the complex, facilitator-attendee matching, but we were wrong. Laptop setup posed the largest challenge. Preparing attendees' institutional laptops to enable access to our cloud or super computing resources caused significant distraction at course startup and in one instance became a barrier to participation. When in-person, trainers can troubleshoot setup by quickly glancing at attendee screens. Virtually, and especially when class sizes are 50 or more, troubleshooting becomes significantly more difficult and time consuming. Our initial attempts at real-time problem solving were chaotic. We trialed several techniques and decided on the use of numerous operating system-specific breakout rooms, which we opened 30 minutes before the training started.

Finally, we debated the value of recording the workshop sessions for public viewing because of the individualized nature of hands-on workshop experiences. We decided to make workshop recordings available and let learners view if desired. The three-part, full workshop recordings are available on the Australian BioCommons YouTube channel [8]. Recorded, modularized workshop topics are available on the Pawsey YouTube channel [9].

5. DELIVERING INTERACTIVE HANDS-ON VISUALISATION TRAINING

Like other Pawsey training, visualization training presents challenges when moving online because of the preference to use hands-on exercises to best learn and practice new skills. The Pawsey Visualization Team found that the key to delivery of interactive, virtual hands-on Vis training was to redesign and re-plan the training from scratch. The result? Web-based remote visualization training.

By transforming the training to a web-based visualization focus, technological challenges were lessened, and training prerequisites minimized. The only requirement was for attendees to have a laptop/desktop with a web browser installed to perform web-based remote visualization.

We modularized the three-hour, in-person training into smaller topics across three days (one 90-minute session per day). We simplified the training slides with detailed screenshots explaining each step. Also, we published the training slides online for the students to access. This approach to slideware not only gave the students a copy of the original slides with high resolution images, but also enabled them to go through the hands-on parts at their own pace.

During the interactive hands-on parts, students were divided into smaller groups and moved to breakout rooms with facilitators. This gave the opportunity to enable two-way communication when needed, such as for questions and troubleshooting.

Planned breaks and scheduled large group Q&A sessions helped us to keep on track — despite the full learning agenda — and finish on time.

6. COMMUNICATING WITH AND SUPPORTING TRAINERS AND TRAINEES

Clear communication and support are some of the most important aspects of training and education activities for trainees and trainers. In contrast to the non-verbal cues so readily available in face-to-face training, communication and interaction in virtual training must be explicit and purposeful.

Good communication between instructor and student is key. Devoid of in-person cues (e.g., body language), trainers online rely on videos, which many trainees turn off. For this reason, we include frequent and varied checkpoints, such as virtual polls, voice Q&A, and hands-on activities for large and small groups.

We also rigorously practice "talk out loud" training and thinking. No on-screen activity, such as coding, is done without speaking about it, even when — especially when — we have a coding "glitch". Glitches provide "learning moments" rich in impact; they are opportunities for students to watch and listen as an expert "unpacks" an issue. Such moments can build immediacy with the instructor and engagement with the content.

Pawsey trainers rarely train alone. Co-training or including facilitators, or helpers, allows the presenting instructor to focus on delivery and engagement, not on the technology or the "chat." Facilitators also make breakout rooms possible. These small, intimate working groups are key to foster attendee participation in the conversation. Attendees are encouraged to ask questions. The platform's chat function is mostly used, but Slack and Google Docs are also used. For example, in the *Using Containers in Bioinformatics* training, trainees filled 18 pages of an online document with robust discussion.

7. TALKING THE TALK, WALKING THE WALK

Pawsey is not only developing a new suite of virtual training targeted externally — at Australian teachers, students and Pawsey users — but also sourcing and co-designing virtual training for Pawsey staff.

The Pawsey Supercomputing Centre is in the process of refreshing supercomputing, data, storage, and networking equipment. New supercomputing systems bring new opportunities as well as challenges for both users and Pawsey staff. To prepare staff for the new infrastructure, Pawsey staff are working to a rigorous learning schedule to address identified skill gaps and to build requisite skills for the new systems.

As with the Pawsey user training, all internal staff training sessions are being conducted virtually, and we are learning numerous techniques and online learning practices, by being students ourselves.

8. RESULTS: TRAINING DELIVERY AND SCALABILITY

At time of publication, we have conducted one full round of our virtual core trainings (plus one-off requests), and we are designing new intermediate and advanced online trainings.

In Table 6 and 7, we compare attendance numbers (attendee reach) for in-person and online training for two sets of conducts: a single conduct of a two-day, in-person roadshow versus a single conduct of the replacement suite of online training, **and** the reach, to date, of the Container trainings.

Table 6. Sample single conduct attendance: in-person vs virtual.

Single Conduct – 2 days In-person roadshow		Single Conduct (10.5 hours) New Virtual Training Suite*	
Live	**Views**	**Live**	**Views**
20	NA**	66	448

*Note that at the time of writing, *Using Nimbus Research Cloud – Part 2* was not published on the Pawsey YouTube.
**Not Applicable

Table 7. Consolidated conduct attendance: Sample virtual training.

3 Conducts New Virtual Container Training	
Live	**Views**
473	1,890

Tables 6 and 7 present training reach only. They do not consider associated costs, such as travel costs / lost "opportunity costs" / trainer travel fatigue for in-person training or development costs for virtual training.

The virtual numbers report a level of scalability unattainable through in-person roadshows or through our previous method of in-person advanced training.

Previously Pawsey had run an advanced webinar on GPU programming. Attendance numbers were strong (85), when compared to in-person attendance figures. However, when Pawsey collaborates with one or more partners, we reach far wider and deeper than a "solo" event.

That our numbers show an increase in attendee reach is not revolutionary, when comparing in-person to online training. However, our increased bandwidth to focus on scaling and sustainability is new. We now have time to work with partners similarly incented to upskill learners in essential and advanced skills in super compute, cloud compute, data, and/or visualization, and we can continue to move our training program forward because we have "reserves," that is, trainers are not experiencing travel-induced trainer fatigue.

The "what's next" conversations in the online training space — too long postponed — are happening in earnest, and we are able to design training programs with best practice online experiences in mind — continually adding onto and refining our virtual toolset.

The opportunities being offered up through teaching and learning online are enormous.

9. CONCLUSIONS

We found that moving online wholesale is both challenging and rewarding. What it is not — is merely shifting materials to a new medium. Going online requires from content developers and trainers to employ a fresh perspective and a willingness to try techniques — and try yet more techniques when the earlier ones do not work as expected.

The positive outcomes to our move online are many — some intended and others unintentional. Attendance is more inclusive. Before, geographical barriers prevented individuals from attending Pawsey's two-day, in-person training, held only in capital cities nationally. Now, location-specific barriers are removed.

Pawsey is reaching out more broadly and actively to find partners with which to collaborate. Finding domain, institutional, and other partners enables us to tailor our technical training to include partner-relevant examples and to focus on "hot topics" of interest to the group. These approaches increase the relevance and impact of Pawsey training, and build cloud, super compute, data and visualisation skills in our user base, and beyond.

We have found that there is no universal, one-size-fits-all learning solution. Rather, there are various solutions and platforms that need to be carefully selected for different groups of learners.

10. ACKNOWLEDGMENTS

We would like to acknowledge the Whadjuk people of the Noongar nation as the traditional custodians of this country, where the Pawsey Supercomputing Centre is located and where we live and work. We pay our respects to Noongar elders past, present, and emerging.

We would like to acknowledge all past and present Pawsey Supercomputing Centre staff who actively contributed to the development and running of our training and outreach activities over the years and to our new colleagues who are already actively contributing to designing and delivering Pawsey virtual training.

We would also like to thank all collaborating institutions especially Software Carpentry, Australian BioCommons, Pawsey Partner institutions as well as all participants of Pawsey's training and education programs.

11. REFERENCES

[1] Koch, C. and Wilson, G. (Ed.). 2016. Expertise and Instruction: Expert Awareness Gap. In *Software Carpentry: Instructor Training*. Version 2016.06, May 2016. https://carpentries.github.io/instructor-training/, 10.5281/zenodo.57571.

[2] Capps, M., Dunaway, S., et al. 2020. Backward Design: A Handy Tool for Remote Teaching. In *William & Mary Law School's Conference for Excellence in Online Teaching Legal Research & Writing*.18 June 2020. https://scholarship.law.wm.edu/excellence_online_teaching/zoomsessions/june18/1/

[3] Tobolowsky, B. F. (Ed.). 2014. Paths to learning: Teaching for engagement in college. National Resource Center for The First-Year Experience. University of South Carolina. 2014.

[4] Leslie, H.J. 2020. Facilitation fundamentals: redesigning an online course using adult learning principles and trifecta of student engagement framework. *Journal of Research in Innovative Teaching & Learning*. 18 May 2020. Emerald Publishing Limited. DOI=https://doi.org/10.1108/JRIT-09-2019-0068

[5] Dixson, M.D. 2010. Creating effective student engagement in online courses: What do students find engaging? *Journal of the Scholarship of Teaching and Learning*. 10, 2 (June 2010), 1–13. https://scholarworks.iu.edu/journals/index.php/josotl/issue/view/159

[6] Pawsey Supercomputing Centre. 2020. Training Portal. https://pawseysc.github.io/training.html

[7] Pawsey Supercomputing Centre. 2020. Containers Training Series. https://pawseysc.github.io/containers.html

[8] Australian BioCommons 2020 Training YouTube Channel. Australian BioCommons YouTube channel

[9] Pawsey Supercomputing Centre. 2020. YouTube Channel. https://www.youtube.com/c/PawseySupercomputingCentre/

High-Performance Computing Course Development for Cultivating the Generalized System-level Comprehensive Capability

Juan Chen
College of Computer
National University of Defense Technology
Changsha, Hunan Province, China
juanchen@nudt.edu.cn

ABSTRACT

Supercomputers are moving towards exascale computing [1], high-performance computer systems are becoming larger and larger, and the scale and complexity of high-performance computing (HPC) [2] applications are also increasing rapidly, which puts forward high requirements for cultivation of HPC majors and HPC course development [3]. HPC majors are required to be able to solve practical problems in a specific field of high-performance computing, which may be a problem for system design or a problem for a specific HPC application field. Regardless of the type of problem, the complexity and difficulty of the problem are often very high because HPC is interdisciplinary. The development of HPC courses to meet these kinds of talent cultivation needs must emphasize the cultivation of students' *Generalized System-level Comprehensive Capabilities*, so that students can master the key elements in the limited course knowledge learning process.

System-level Comprehensive Capability refers to the ability to use the knowledge and ability of the computer system to solve practical problems. The ACM/IEEE Joint Computer Science Curricula 2013 (CS2013) [4] also involves *System-level Perspective*. *System-level Comprehensive Capability* is considered to be a crucial factor to improve students' system development ability and professional ability. This is especially important for students majoring in high-performance computing. Furthermore, due to the HPC field's interdisciplinary and high complexity characteristics, *System-level Comprehensive Capability* is not enough for HPC majors, and students need to have *Generalized System-level Comprehensive Capabilities*. A knowledge system at the computer system level "vertically" (from bottom to top: parallel computer architecture, operating system/resource management system, compilation, library optimization, etc.) is no longer enough; multiple high-performance computing application areas should also be "horizontally" involved. *Generalized System-level Comprehensive Capabilities* vertically and horizontally can meet the needs of different types of high-performance computing talents.

How to cultivate the *Generalized System-level Comprehensive Capabilities* of HPC majors? National University of Defense Technology (NUDT) of China has faced some challenges [5] in building a series of high-performance computing courses, adopted some measures, and gained some experience [5–7]. NUDT has developed the Tianhe-2 supercomputer, which ranked No. 1 in the TOP500 list[1] six times from June 2013 to November 2015. These achievements are inseparable from the training of high-performance computing talents and HPC curriculum development. NUDT has offered a series of high-performance computing courses from freshman to postgraduate for a long time. The courses cover a wide range, are difficult and practical, and pay great attention to the cultivation of students' *Generalized System-level Comprehensive Capabilities*. The following main means are adopted: i) Hierarchical capability model construction is used to guide the establishment of curriculum system and curriculum setting; ii) Real practice platforms and real cases from frontier scientific challenges are used to construct step-by-step practice cases; iii) A teaching mechanism that integrates scientific research and teaching content is adopted. In the curriculum setting, the emphasis is placed on basic mathematics, general science courses, high-performance computing professional courses, and basic courses for specific HPC application fields. Regarding the content of the curriculum, it is based on the principle of breaking the boundaries of disciplines and specialties, establishing the relevance of the knowledge system and frontier scientific issues, and designing the whole process of teaching content with the direct facing of basic scientific issues and frontier scientific research issues. In terms of course implementation methods, there are various forms, including small-class teaching, seminar-based teaching, case-based teaching, and flipped classrooms, etc. In the past ten years, the curriculum construction at NUDT has achieved remarkable results. We have cultivated *Generalized System-level Comprehensive Capabilities* of high-performance computing majors very well and greatly helped the development of our high-performance computing research.

KEYWORDS

High-performance computing, High-performance computing curricula, Generalized system-level comprehensive capability

https://doi.org/10.22369/issn.2153-4136/12/2/7

[1] www.top500.org.

ACKNOWLEDGMENTS

This work is supported by the 2019 Hunan Province Higher Education Teaching Reform Research Foundation of China (titled with "Teaching Practice of Training High-Performance Computing Talents Relying on High-level Scientific Research"), and the 2019 Hunan Province Postgraduate Outstanding Professional Case Foundation of China (titled with "High-Performance Computing Series Case Library").

REFERENCES

[1] Daniel A Reed and Jack Dongarra. Exascale computing and big data. *Communications of the ACM*, 58(7):56–68, 2015.

[2] NetApp, Inc. What Is High-Performance Computing?, 2019. https://www.netapp.com/us/info/what-is-high-performance-computing.aspx.

[3] Rajendra K. Raj, Carol J. Romanowski, John Impagliazzo, Sherif G. Aly, Brett A. Becker, Juan Chen, Sheikh Ghafoor, Nasser Giacaman, Steven I. Gordon, Cruz Izu, Shahram Rahimi, Michael P. Robson, and Neena Thota. High performance computing education: Current challenges and future directions. In *ITiCSE-WGR'20: Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, pages 51–74, New York, NY, USA, 2020. ACM.

[4] Joint task force on computing curricula, association for computing machinery (acm) and ieee computer society. computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science. Technical report, New York, NY, USA, 2013. 999133.

[5] Juan Chen, John Impagliazzo, and Li Shen. High-performance computing and engineering educational development and practice. In *Proceedings of the 50th Frontiers in Education 2020 (FIE2020)*, pages 1–8, Uppsala, Sweden, 2020. IEEE.

[6] Juan Chen, Li Shen, Jianping Yin, and Chunyuan Zhang. Design of practical experiences to improve student understanding of efficiency and scalability issues in high performance computing (poster). In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE2018)*, pages 1090–1090, New York, NY, USA, 2018. ACM.

[7] Juan Chen, Yingjun Cao, linlin Du, Youwen Ouyang, and Li Shen. Improve student performance using moderated two-stage projects. In *Proceedings of ACM Global Computing Education Conference (CompEd2019)*, pages 201–207, New York, NY, USA, 2019. ACM.

Employing Directed Internship and Apprenticeship for Fostering HPC Training and Education

Elizabeth Bautista
NERSC
Lawrence Berkeley National Laboratory
Berkeley, CA
ejbautista@lbl.gov

Nitin Sukhija
Department of Computer Science
Slippery Rock University of Pennsylvania
Slippery Rock, PA
nitin.sukhija@sru.edu

ABSTRACT

Positions within High Performance Computing are difficult to fill, especially that of Site Reliability Engineer within an operational area. At the National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (LBNL), the Operations team manage the HPC computational facility with a complex cooling ecosystem and also serve as the wide area network operations center. Therefore, this position requires skill sets in four specific areas: system administration, storage administration, facility management, and wide area networking. These skills are not taught in their entirety in any educational program; therefore, a new graduate will require extensive training before they can become proficient in all areas. The proximity to Silicon Valley adds another challenge in finding qualified candidates. NERSC has implemented a new approach patterned after the apprenticeship program in the trades. This program requires an intern or apprentice to fulfill milestones during their internship or apprenticeship timeframe, with constant evaluation, feedback, mentorship, and hands-on work that allow candidates to demonstrate their growing skill that will eventually lead to winning a career position.

KEYWORDS

Site reliability engineer, HPC education, Training, Apprenticeship, Internship

1 INTRODUCTION

According to a 2008 analysis from the Public Policy Institute of California, it is projected that by 2025, the number of college graduates will not meet the projected demand of the workforce [2] [1]. The analysis states that in recent decades the economic growth took place in a time-frame where there was significant growth in the number of workers with a college education. However, the analysis projects limitations due to a slower growth in the supply of college-educated workers in coming decades [3].

As part of succession planning, the National Energy Research Scientific Computing Center (NERSC) Operations team decided they needed to find a different way of recruiting talent and implemented a directed approach to both their internship and apprenticeship programs [8]. Not only do these programs allow them to identify and recruit more qualified and committed candidates, but they also help in retaining them for a career position.

This paper documents the process of the directed internship and apprenticeship programs in this format. Section 2 provides the background of how unsuccessful recruiting and retention processes lead to this decision. Section 3 explains the difference between the internship and apprenticeship programs. Section 4 explains the logistics such as how the program works, what are the essential parts, and what makes it work. Section 5 provides positive outcomes, and Section 6 provides final thoughts.

2 BACKGROUND

When NERSC moved from Livermore to Berkeley in 1995, the 24x7 Operations Team was a group of 9 technicians who used written manuals from the systems and storage groups. They followed a specific set of directions then engaged the person on call, who would eventually solve the problem. Because these positions had a standard operations procedure (SOP), it was not difficult to recruit for talent, as long as they can follow directions. The team were onsite 24x7 across three eight-hour shifts. As technicians, they were eligible for overtime. This was not a deterrent for recruiting and retaining talent.

With an incoming new manager in 2011, the team decided they wanted to grow professionally in order to have more control over their area. Through professional development, the team's classification was changed to Site Reliability Engineer, and they became salaried staff, much higher in range than the previous technician classification. The team had upgraded their daily work such that they were now managing systems, storage environments and the wide area network.

With the impending move to a new and state-of-the-art building in 2015, they were poised to manage not only a more sophisticated water cooled system but also a building ecosystem that could support their path to exascale. Managing operations in this type of environment requires staff who must, in addition to system administration, understand power, infrastructure and cooling requirements demanded by the ecosystem. Such complexity and scale provide unique challenges, including usage fluctuations and providing high availability and high utilization for users who need to have their jobs run in spite of failures or cooling requirements involving both air and water. The additional skill set now also required knowledge of mechanical and electrical engineering. Because of the new classifications, the prior pipeline stopped providing qualified candidates. Rather, candidates who applied to the position would be knowledgeable in one or two areas, and the team would need to

https://doi.org/10.22369/issn.2153-4136/12/2/8

train for the additional two areas. The current position now directly competes with highly sought-after skillsets in Silicon Valley, which makes it much more challenging to retain candidates. Within a year or two, the new hire would eventually leave, noting that they were unhappy with the off-shift and had found a position elsewhere working a standard 9–5, Monday through Friday. Retention became especially challenging during the holidays, where the staff continued to be onsite while all others have the paid time off. This short retention required the team to constantly have to restart training a new person, if they can recruit an adequate candidate.

According to the 2010's high-tech employment report by the Public Policy Institute of California, 12.6% of all employments in the computer industry are with Silicon Valley companies, which accounts for fifteen times the national average. Moreover, other cities only employ 3% or less in the computer industry [5] [4] [6].

Realizing that the higher classification meant they were competing much more with Silicon Valley companies who tend to recruit them out of NERSC, even if their training is not complete, the team decided on a new approach; they would grow their workforce. In this way, they could also determine the commitment of the individual as they were learning the skill [11].

3 DEFINITIONS

3.1 Directed Internship

An internship is the position of a student or trainee who works in an organization, sometimes without pay, in order to gain work experience or satisfy requirements for a qualification or to earn credits.

In this case, interns are still in school and usually worked twenty hours during the school term and full-time when they are out of school, such as in between term breaks or the summer. They can either be directly hired and paid, or if they are earning credits for the internship, this is arranged through their school and NERSC's affiliate program.

Upon discussion with the intern, they are provided with a specific project to work on that allows them hands-on practice of a skill that they already studied in school. The directed internship provides them the opportunity to work on a real-world project that is implemented at the workplace.

Interns can work for additional school terms and in the summer to learn additional skills until, if they choose, they graduate. At this point, they can be prepared to compete for a career position or choose to enter the apprenticeship program [9] to finetune their skill.

3.2 Directed Apprenticeship

An apprenticeship is an effective work-based learning strategy that creates pathways to career advancement and higher wages through hands-on experience. The program can provide access to successful career on-ramps for targeted worker populations, such as disadvantaged youth, veterans, and women in non-traditional fields.

Apprentices are usually close to or at the end of their educational program and are ready to commit to a full-time training program. They are hired for a one-year paid term with the understanding that they have milestones to fulfill. Successful completion of these milestones gives them a very high degree of leverage in competing for a career position in twelve months [7].

4 LOGISTICS

4.1 Assumptions of the Program Participants

The expectation of the intern or apprentice is that they are working in a realm of adult education and preparation. They should be capable adults who have the ability to express their needs, their problems, and their interests, and they should be able to make the type of decisions that adults normally make. They should have the ability to choose, when appropriately informed, the situation or environment for experiential learning. For example, an apprentice is given a task to complete within a timeframe. After two weeks and much research, they find that they need a book. They should be able to make the decision to either purchase the book and ask if there is a reimbursement process or ask their supervisor to purchase the book for them. Further, if there is a class they need to take that is a one-week seminar, they should also be able to ask this of their supervisor.

Since we are in the business of educating adults, this includes assumptions and attitude of trust. Interns and apprentices are capable of discerning effective and/or appropriate behavior in others, especially when given the encouragement to reflect upon their observations. For example, an intern should be able to tell their supervisor that the person assigned to work with them on their technical project does not explain things in a way they can understand. They also need to trust the supervisor that they can come to a mutually beneficial agreement or solution.

It has been our experience that participants who are more successful in the program are those who are mature, sometimes second-career individuals with more work experience, even if it is not in the computer science field. They are more thoughtful, are serious about their education and training, and have an idea of their career path. The advantage of this program is that they are constantly being mentored and managed to work with actual staff who perform the job for which they will eventually compete, most of the time, in a one-on-one approach.

4.2 The Directed Approach

The goal of the program is to successfully train and expose participants daily to the job toward which they will eventually compete. As such, training is hands-on based on what they learned in school. For example, if they took a class in Python, they will be assigned to write a Python program. If they took a class in system administration, they will assist in managing the HPC systems. They work side-by-side with the individual, and this provides daily insight into their working environment. After some training, the participant will eventually, under direct supervision, perform the role to diagnose and triage problems. Then they are debriefed on their performance and provided feedback on constant improvement.

This type of learning promotes the participant's sense of responsibility and ownership toward the experience. As they strengthen their skill, they continue to perform with less supervision on one skill as they practice a new skill until they complete a milestone. They continue to complete milestones until they are ready to compete for a career position.

During this time, they perform the daily operational tasks to manage a data center, they are given one to two projects that they complete, and they are given a larger project that they must present to the management before implementation. Each skill is tightly practiced and developed, even the public speaking and presentation techniques.

4.3 What Makes It Work?

Certain processes need to occur in order to make this program work.

4.3.1 Administrative

Work with Human Resources to create a position description for both the apprentice and intern. The intern position should highlight that the participant will learn a skill through working on and completing the project.

The apprentice position should list required skills that will allow the participant to practice what they learned in school. Further, the position description should be the entry-level position for the career they will compete for at the end of the program.

It should be the goal of the organization to minimize the idea that apprentices and interns are free help, i.e. irrelevant. Do not provide work that is easy with little training or supervision, repetitive work, or work that the participant already knows how to perform. The help they provide should be almost at the level of your current staff.

4.3.2 Mentors, Trainers and Attitudes

It is essential to identify staff who like to train and work with a person one-on-one as part of their day. They should consistently have a positive attitude toward the participants and their job.

The mentor or trainer should understand that participants will solve problems any way they can, so encourage them to use their strongest skills. Treat them like they are fresh eyes, and allow them to provide you feedback of how to make a process more efficient.

"Everyone learns how to survive with minimal training, unless a teacher, "systems manager" can design a strategy for ensuring that staff cope with their deficiencies as well as utilize their strengths."[10].

It is also important to identify a supervisor who can technically evaluate progress and set milestones, and they need to work closely with the mentor or trainer. This person needs to have a clear understanding of the role of the trainer and the role of the participant. They monitor progress and, if needed, troubleshoot an interaction or a teaching issue and implement corrective actions. For example, provide more challenging training if the participant seems to learn quicker than usual. They need to be highly accessible to both parties to be a soundboard, role play, provide advice on paths or solutions, or even to assist in navigating a course of action. Finally, this person can serve as a model for the aspiring professional.

Lastly, there needs to be an individual that understands the hiring process of the organization. They can help the manager navigate the hiring process, help evaluate readiness from a "paper" standpoint. For example, they can assist in evaluating the participant's resume and determine if the candidate is ready, at least on paper.

4.3.3 Why does it work?

The participant gains a variety of experiences and sees what is being done in the job they want to acquire.

Having different roles separated provides a higher quality of supervision and exposes the participant to many more staff in the organization.

The psychological benefits are numerous, including the following:

- The idea that it is a workplace, not campus, and provides a "real world" feel.
- Exposure to different roles, providing a feeling that they can find a niche in the organization.
- Many more networking opportunities and role models.
- Coming from a school environment, it provides a clean slate, especially for a second-career individual.
- They develop self-confidence and an improved self-image. (You have no idea how many of my apprentices call themselves Site Reliability Engineers).
- They perform operational work that needs time management and work management, and they find their place in the overall workplace.

As part of recruitment, we need to explain an apprenticeship or even a directed internship; therefore, we engage more with the community and community schools. A participant who comes from a particular community feels like they represent the community and will continue the engagement when they are hired.

As an economic benefit to the organization, apprenticeships are less expensive than directly hiring and not retaining, and because we are engaging the community, there is potential for developing a workforce pipeline.

5 POSITIVE OUTCOMES

Below are statistics of the directed apprenticeship program from January 2015 through December 2019.

- NERSC Operations has had 25 apprentices.
- Of these, SIX have been fully retained in a career position.
- THREE have completed the program and accepted a position elsewhere. This is considered a win.
- FOUR opted to continue toward a higher education, which we also consider a win.
- THREE are currently in the program. (As of January 2021, 1 was hired, 2 went on to a graduate program.)
- NINE did not complete the program.

6 CONCLUSION AND FURTHER WORK

The positive outcomes of the program show that NERSC Operations is able to recruit quality talent, retain more individuals, or encourage individuals to further their education. Even those who decided to find a position elsewhere were able to find a job at a higher pay level than they would have without the apprenticeship. Overall, it has had a positive impact, not only for NERSC Operations, but also for the individuals themselves.

In terms of fostering HPC education and training, the program itself exposes participants to a subject rarely taught in school. The more they work in this niche, the more they become familiar with the "topics" they need to learn, practice, and eventually grow into.

Participants take this knowledge and apply it once they become career individuals and seek out education and training opportunities toward their professional development accordingly. The program works due to the constant evaluation to meet milestones.

Our assessment of success is determined by the retention of the apprentice as a hired individual. Because of this program, NERSC Operations is now fully staffed.

This program is generic enough that it can be used in any situation: for example, to diversify your workforce. The pandemic that started in 2019 has provided us the challenge of staff not being able to work side-by-side with their mentors. However, we leveraged technology to be able to continue the program.

ACKNOWLEDGMENTS

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under Contract No. DEAC02-05CH11231.

REFERENCES

[1] [n. d.]. California 2025: Taking on the Future. https://www.ppic.org/content/pubs/report/R_605MB2R.pdf. Accessed: 2021-01-14.
[2] [n. d.]. California's Need for Skilled Workers. https://www.ppic.org/publication/californias-need-for-skilled-workers/. Accessed: 2021-01-14.
[3] [n. d.]. Can Apprenticeships Train the Workforce of the Future? States Hope So. https://www.governing.com/topics/mgmt/gov-work-study-student-debt-apprenticeships.html. Accessed: 2021-01-14.
[4] [n. d.]. Can California Import Enough College Graduates to Meet Workforce Needs? https://www.ppic.org/content/pubs/cacounts/CC_507HJCC.pdf. Accessed: 2021-01-14.
[5] [n. d.]. High-Tech Employment In California. https://www.ppic.org/content/pubs/jtf/JTF_HighTechEmpJTF.pdf. Accessed: 2021-01-14.
[6] [n. d.]. High-Tech Employment in California. https://www.ppic.org/content/pubs/jtf/JTF_HighTechEmpJTF.pdf. Accessed: 2021-01-14.
[7] [n. d.]. Why Apprenticeship Works. https://www.ajactraining.org/why-apprenticeship-works/. Accessed: 2021-01-14.
[8] Liliane Bonnal, Sylvie Mendes, and Catherine Sofer. 2002. School-to-work transition: apprenticeship versus vocational school in France. *International Journal of Manpower* (2002).
[9] Barbara LeGrand Brandt, James A Farmer Jr, and Annette Buckmaster. 1993. Cognitive apprenticeship approach to helping adults learn. *New directions for adult and continuing education* 1993, 59 (1993), 69–78.
[10] Edgar S Cahn. 1980. Clinical Legal Education from a Systems Perspective. *Clev. St. L. Rev.* 29 (1980), 451.
[11] Lisa M Lynch. 2007. *Training and the private sector: international comparisons*. University of Chicago Press.

Ask.Cyberinfrastructure.org: Creating a Platform for Self-Service Learning and Collaboration in the Rapidly Changing Environment of Research Computing

Julie Ma
MGHPCC
Holyoke, MA
jma@mghpcc.org

Torey Battelle
Colorado School of Mines
Boulder, CO
battelle@mines.edu

Katia Bulekova
Boston University
Boston, MA
ktrn@bu.edu

Aaron Culich
UC Berkeley
Berkeley, CA
aculich@berkeley.edu

John Goodhue
MGHPCC
Holyoke, MA
jtgoodhue@mghpcc.org

Jacob Pessin
Boston University
Boston, MA
jpessin@bu.edu

Vanessa Sochat
Stanford University
Stanford, CA
vsochat@stanford.edu

Dana Brunson
Internet2
Ann Arbor, MI
dbrunson@internet2.edu

Tom Cheatham
University of Utah
Salt Lake City, UT
tec3@utah.edu

Sia Najafi
Worcester Polytechnic Institute
Worcester, MA
snajafi@wpi.edu

Chris Hill
Massachusetts Institute of Technology
Cambridge, MA
cnh@mit.edu

Adrian Del Maestro
University of Vermont
Burlington, VT
adrian.delmaestro@uvm.edu

Bruce Segee
University of Maine
Orono, ME
segee@umaine.edu

Ralph Zottola
University of Alabama
Birmingham, AL
rzottola@uab.edu

Scott Valcourt
University of New Hampshire
Durham, NH
sav@cs.unh.edu

Zoe Braiterman
OWASP
New York, NY
Zoe.braiterman@owasp.org

Raminder Singh
Harvard University
Boston, MA
R_singh@g.harvard.edu

Robert Thoelen
Pratt & Whitney
Rthoelen@ieee.org

Jack Smith
West Virginia Research
Jack.smith@wvr.org

ABSTRACT

Ask.CI [3], the Q&A site for Research Computing, was launched at PEARC18 with the goal of aggregating answers to a broad spectrum of questions that are commonly asked by the research computing community. As researchers, facilitators, staff, students, and others ask and answer questions on Ask.CI, they create a shared knowledge base for the larger community.

For smaller institutions, the knowledge base provided by Ask.CI provides a wealth of knowledge that was previously not readily available to scientists and educators in an easily searchable Q&A format. For larger institutions, this self-service model frees up time for facilitators and cyberinfrastructure engineers to focus on more advanced subject matter. Recognizing that answers evolve rapidly with new technology and discovery, Ask.CI has built in voting mechanisms that utilize crowdsourcing to ensure that information stays up to date.

Establishing a Q&A site of this nature requires some tenacity. In partnership with the Campus Champions, Ask.CI has gained traction and continues to engage the broader community to establish the platform as a powerful tool for research computing. Since launch, Ask.CI has attracted over 250,000 page views (currently averaging nearly 5,000 per week), more than 400

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/9

contributors, hundreds of topics, and a broad audience that spans the US and parts of Europe and Asia.

Ask.CI has shown steady growth in both contributions and audience since it was launched in 2018 and is still evolving. In the past year, we introduced *Locales*, which allow institutions to create subcategories on Ask.CI where they can experiment with posting institution-specific content and use of the site as a component of their user support strategy.

CCS CONCEPTS

• General and reference → Document types → General literature • Human-centered computing → Human computer interaction (HCI) → Interaction paradigms → Web-based interaction

Keywords

Q&A, Research computing, Self-service learning

1. BACKGROUND

Ask.cyberinfrastructure.org is a collaborative, crowd-sourced Q&A site specifically curated for the research computing community. The project began in September 2017 with the vision of constructing a resource that allows the research computing community to more quickly find answers to commonly asked questions by way of a shared, public knowledge base, resulting in better/faster science results.

The goal of the project is to aggregate answers to a broad spectrum of questions that are commonly asked by researchers and educators as they utilize advanced computing and data resources. The result is a self-service knowledge base for domain researchers, facilitators, cyberinfrastructure (CI) engineers, and others who comprise the research computing community. The hope is that this site will become the go-to platform for sharing frequently asked questions, comparing solutions, and building on previous work pertaining to research computing. Making this knowledge readily available in the public domain will free up time for facilitators and CI engineers to focus on more advanced subject matter, thereby elevating the research computing practice. Simultaneously, the platform allows users to apply an andragogic approach to explore the information on the site at their own pace.

As the project launched, it rapidly drew the enthusiastic support of the XSEDE Campus Champions leadership team. We have collaborated since December 2017 to build the site and launched it together in July 2018 at PEARC18. A naming contest at the conference yielded “ask.cyberinfrastructure.org”, which we then nicknamed “Ask.CI” (available at https://ask.ci/). The site has been well-received, with nearly 5,000 page views per week, over 400 registered contributors, and hundreds of topics.

Establishing a Q&A site of this nature requires some tenacity. We have gained some traction and hope to continue to engage the broader community to firmly establish this platform as a tool for the global research computing community. Ask.CI has further inspired thinking and awareness about the importance of the subject matter. Throughout the development process, we have been thinking frequently about what defines "research computing" in relation to other computing disciplines. The hope is that not only will Ask.CI become a great resource for the community, but that it will also provide public testimony of the importance of research computing and how it exists in relation to enterprise IT, computer science, and domain research.

1.1 StackExchange and Discourse

As we investigated possible technologies upon which to build the site, there was consensus about using a platform that supports a voting mechanism that enables crowd-sourced monitoring and pushes the best answers to the top, with moderation tools to manage spam/trolling. This led us to StackExchange [2], the gold standard for Q&A platforms. StackExchange is the platform behind Stack Overflow and many other widely used Q&A sites. While it is not easy to establish a StackExchange site, it offers several advantages, including search engine visibility, built-in backup and maintenance, security and resistance to spam and trolling, voting, and a clear question-and-answer syntax that yields definitive answers. Launching a StackExchange site involves a rigorous, four-phase process, including a restart if any phase exceeds a specified time limit. In our first iteration, which closed in May of 2018, we reached the second phase and attracted a working group of volunteers who developed questions and answers to post on the site in preparation for when it became functional. Since our StackExchange site was not yet functional, we curated these on another platform called Discourse [1], an open source Internet forum, and we developed a methodology for culling nuggets of information from ad hoc user questions and adapting them for use by a general audience. While Discourse is not as ubiquitous as StackExchange, it is a very flexible platform with a low startup threshold.

When our StackExchange effort was closed, we decided to formalize the Discourse content, add a voting mechanism to mimic Stack Exchange functionality, and launch our Q&A site on the Discourse platform. Subsequently, we discovered several benefits of using the Discourse platform over StackExchange. These include: 1) having flexibility in the question format to include discussion topics as well as Q&A and 2) having the ability to set up categories, which we are using to create institution-specific “locales,” described below.

1.2 Broadening Participation

The idea for Ask.CI originated with the NSF-sponsored Northeast Cyberteam Program, which aims to make research computing more accessible to small/medium sized institutions in northern New England. Ask.CI became an integral part of the strategy. By creating a shared, public knowledge base composed of content which is often found behind the firewall at large institutions, Ask.CI enables researchers at smaller institutions to become more self-sufficient, reducing the need for Research Computing Facilitation support.

2. Ask.CI 2020

Since launching at the 2018 Conference on the Practice & Experience in Advanced Research Computing (PEARC18), we have nurtured Ask.CI into an active, growing site managed by a dedicated group of volunteer site moderators who meet weekly via Zoom. The primary purpose of the meeting is to actively curate the site and discuss outreach activities. Active curation includes reviewing new posts, checking for unanswered topics, considering new subject matter areas to cover, and planning weekly marketing activity. While Ask.CI has shown steady growth in both contributions and audience, finding methods to continue to grow audience participation is an ongoing focus of our attention, as the expert research computing knowledge that we seek to gather is widely distributed among the community. We have employed several methods to do this, described below.

2.1 Events, Meetings and Conference Calls

We actively seek opportunities to talk about Ask.CI with audiences at venues where the research computing community has congregated. Since launch at PEARC18, we have conducted Birds of a Feather (BoF) sessions at all subsequent PEARC and Super Computing (SC) conferences. We have also presented on the CI Brown Bag, XSEDE Campus Champions, Campus Research Computing Consortium (CaRCC), and EDUCAUSE Research Computing (RCD) calls, and at regional gatherings whenever possible.

2.2 Question of the Week

Each week at our site moderators' Meeting, we review new content that has been posted on the site and content that has not been answered. If we find a topic that has been unanswered for over a week, we will likely mark it a question of the week (QoW). QoWs are emailed to the Campus Champions mailing list and tweeted to the Ask.CI twitterverse. A goal for this year is to expand the recipient list beyond the Campus Champions, to other groups in the Research Computing ecosystem that might be able to answer the questions.

2.3 Friday Factoid and Sunday Science

In addition to the QoW, we occasionally tweet Friday Factoids, interesting tidbits of relevant material about the research computing world, particularly if they are timely with calendar or current events. We also post Sunday Science stories, which are more domain-specific deeper dives. All three of these methods function as reminders that attract the community back to the site, in part to see if new topics have been posted, or simply as a reminder that Ask.CI exists and that new content is always welcome.

3. INFRASTRUCTURE

3.1 Q&A and Discussion Zone

One of the most stringent requirements of building a site using StackExchange is that questions must be written in a manner that there can be a well-defined, best answer. This is necessary so that the voting mechanism, which is required to ensure that content stays up to date, can work correctly.

In research computing, where work is frequently in unexplored territory, sometimes it is not clear if there is a best answer, or how this can be evaluated. Often, a dynamic and relevant discussion among subject matter experts can help to formulate an answer or a response to a particular situation. For this reason, we created the Discussion Zone, where discourse in response to a question can take place. We distinguish topics that go in the discussion zone from those that have a clear-cut best answer, which are categorized as Q&A. At present, for the most part, we have not found a need to further categorize questions by subject matter or domain, and in fact believe it to be useful to have topics on wide-ranging subject matter in a single category, as this will encourage cross-pollination of solutions among different domains.

3.2 Tags

To facilitate searching for answers that pertain to particular subject matter on the site, questions are tagged with labels that identify subject matter and other characteristics of the content. Users can also search by tag to obtain a listing of all topics tagged with this particular label. This mechanism is used both to delineate content, and also to aggregate it when appropriate. There are currently 216 active tags on the site, many of which are only used a few times. There are also tags to indicate the audience for a particular topic. Following the CaRCC model of audience delineation, these are researcher, system, and data, representing topics of interest to researcher-facing, systems-facing, and data-facing facilitators.

3.3 Voting

Faced with ever-changing technology advances, one of the key strategies for keeping content up to date on the site is a voting mechanism modeled after the voting function on StackExchange sites. Users can vote for the "best" answer to a given question, and they can also vote on "best" questions. The software then re-arranges topics so that topics and questions with the highest number of votes appear first. Unlike the StackExchange model, our system only allows "up" votes, which creates a more convivial environment for participants. The intended result is that over time when new content is posted on a topic that renders other answers obsolete, voting will ensure that the most relevant answer appears first.

4. EXPANSION VIA LOCALES

In spring 2019, one of the Ask.CI moderators observed that there could be significant benefits to having institution-specific content on the site. We introduced a program that allows institutions to create subcategories on Ask.CI, dubbed "locales," where they can experiment with posting institution-specific FAQs and using the site as a component of their user support strategy. The intent is that by sending users to Ask.CI via the institution-specific sandbox, it will encourage them to start down a path of self-service learning, simultaneously encouraging user-to-user collaboration both within the institution's own user community and across the research computing community as a whole. One of the key benefits of having this exploration occur under the umbrella of Ask.CI is the simplicity of migrating content from the main site to a locale and vice versa.

We piloted the locale concept from April to November 2019 with six institutions and formally announced the Locales program at SC19. As of April 2020, there are 13 locales in service: Brown University, Colorado School of Mines, the Computing against Covid-19 Project, Harvard University, MGHPCC, the Northeast Cyberteam, Northeastern University, Ohio Supercomputer Center, the ResearchSOC Cybersecurity community of practice, Stanford University, Tufts University, University of Maine, and Yale University. Other institutions are in the process of starting theirs up: MIT, University of Alabama, University of Missouri, University of New Hampshire, University of Vermont and several other institutions. The CaRCC consortium and US-RSE have expressed interest in building locales later in 2020. Locale moderators join our Ask.CI site moderators' call once a month to check in and exchange ideas. We are also developing a toolkit to facilitate integration with existing support platforms (websites and ticketing systems) that are often behind institutional firewalls.

5. EVALUATION AND METRICS

A standing item on the site moderators' weekly meeting agenda is to review certain statistics to monitor the health of the site, including page views (daily, monthly, and aggregate), users, and return visits. We also periodically count the number of unique institutions represented, the topics by audience type (researcher-facing, systems-facing, data-facing, end-user), and the total number of locales.

6. LESSONS LEARNED AND FUTURE PLANS

Ask.CI is entering its third year of existence and is growing at a steady rate. We have learned a great deal about the effort required to establish and maintain a site of this nature. A few key observations are noted below.

6.1 Creating a Workflow Shift Takes Time

The process of creating a public record of content that is typically disclosed on a private mailing list or behind a firewall requires a shift in mindset and habits. Although most people understand the value of creating this knowledge base, it is an extra step, and not all individuals with the knowledge are readily able to take the time to post topics or answers on a regular basis. In the upcoming year, we hope to introduce a program which creates incentives to contribute, by recognizing the effort and the value of the information already posted.

6.2 Discourse Flexibility Allows Creativity in Outreach Not Possible with StackExchange

As described in Section 1, we began this effort thinking that we would build this site on the StackExchange platform. While we were initially disappointed that our first effort was terminated, we have found the flexibility of being able to support a discussion zone and locales has been a happy outcome of shifting to the Discourse platform. This year, we hope to put mechanisms in place to address one of the key benefits that StackExchange platforms enjoy, which is the high Google ranking that they inherit from StackExchange. The inherited ranking can put content from other sites earlier in search results than Ask.CI content, even if the Ask.CI content is more relevant to the topic being searched.

The methods that we have in mind for this year would put Ask.CI on a more equal footing with sites that currently have this advantage.

6.3 Convivial Weekly Meetings Have Yielded a Dedicated Group of Moderators and Allow for Ebb and Flow of Individual Workloads

The weekly site moderators' meetings comprise a group of seven volunteers who have been the backbone of Ask.CI. The site would not have matured to the level that it has reached without their tireless efforts. In a given week, the number of people at the meeting will vary, so it has been very beneficial to have a large enough group to be able to make forward progress each week.

6.4 Outreach is Key

The Question of the Week, Friday Factoid, and Sunday Science communications have had a noticeable impact on our page view statistics each week. We hope to reach out to other communities and mailing lists this year to expand our presence and welcome any suggestions/recommendations from the community.

6.5 Cross-Posting to Specialized Communities of Experts for Specific Topics

We have identified certain specialized boards, both in the community and at vendors, that have been great sources of one-off content when the need has arisen. Finding a systematic way to keep those groups engaged with our site will have long term positive results.

6.6 Locales Have Great Potential When Institution/Community of Interest is Ready

Although we are in early stages with many of our locale partners, we have had very promising initial results. We hope to capitalize on this concept with further outreach to the community to find other institutions that are ready to create a locale. In addition, building the integration tools mentioned above will facilitate the onboarding of other institutions and create a more seamless experience for the end users.

6.7 Lessons Learned and Future Plans

We welcome and encourage feedback and participation on the Ask.CI platform. Participation can take many forms, from simply reading content on the site and posting a reply occasionally, to full participation on the site through a locale.

7. ACKNOWLEDGEMENTS

This work has been partially supported by the National Science Foundation under grant award ACI-1659377. The authors also thank the many the Ask.CI site moderators and Locale moderators and volunteers whose efforts to build and maintain the site are beginning to yield significant results, and the hundreds of contributors who have generously shared knowledge and experience on Ask.CI.

8. REFERENCES

[1] "Discourse — Civilized Discussion." 2020. Discourse — Civilized Discussion. Discourse. Accessed June 16, 2020. https://www.discourse.org/.

[2] "Stack Exchange." 2020. Accessed June 16, 2020. https://stackexchange.com/.

[3] "Ask.Cyberinfrastructure." 2020. Ask.Cyberinfrastructure. Accessed June 16, 2020. https://ask.cyberinfrastructure.org/.

[4] Goodhue, J., Ma, J., Del Maestro, A., Najafi, S., Segee, B., Valcourt, S., Zottola, R. Northeast Cyberteam:Workforce Development for Research Computing at Small and Mid-sized Institutions. Proceedings of the Conference on Practice and Experience in Advanced Research Computing (PEARC20), Portland, OR, July 26–30, 2020, doi:10.1145/3311790.3396662.

[5] Goodhue, John, et al. "Northeast Cyberteam Program – A Workforce Development Strategy for Research Computing." The Journal of Computational Science Education, vol. 11, no. 1, 2020, pp. 8–11., doi:10.22369/issn.2153-4136/11/1/2.

The Design of a Practical Flipped Classroom Model for Teaching Parallel Programming to Undergraduates

Dirk Colbry
Michigan State University
East Lansing, MI
colbrydi@msu.edu

ABSTRACT

This paper presents a newly developed course for teaching parallel programming to undergraduates. This course uses a flipped classroom model and a "hands-on" approach to learning with multiple real-world examples from a wide range of science and engineering problems. The intention of this course is to prepare students from a variety of STEM backgrounds to be able to take on supportive roles in research labs while they are still undergraduates. To this end, students are taught common programming paradigms such as benchmarking, shared memory parallelization (OpenMP), accelerators (CUDA), and shared network parallelization (MPI). Students are also trained in practical skills including the Linux command line, workflow/file management, installing software, discovering and using shared module systems (LDMOD), and effectively submitting and monitoring jobs using a scheduler (SLURM).

Keywords

Computational science, Flipped classroom, Parallel programming.

1. INTRODUCTION

Established in 2015, the Department of Computational Mathematics Science and Engineering (CMSE) at Michigan State University (MSU) represents a new discipline at the intersection between methods (math and computer science), domain applications (science and engineering) and computation (programming and large-scale computing). CMSE's mission is to advance the use of computational methods in all areas of scientific research and engineering within the university [1]. This includes the training of undergraduate and graduate students from a wide variety of STEM (science, technology, engineering, math) and non-STEM majors in how to best utilize computation as they become experts in their own fields. Our first two introductory courses (CMSE 201 and 202) teach students programming, computational modeling techniques [2], and tools for computational modeling (similar to and motivated by software carpentry [3]). Our latest course, which is the focus of this paper, is "Methods in Parallel Programming" (CMSE 401). This course is intended for advanced students who would like to speed-up their research and utilize advanced computational hardware.

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/10

By the end of CMSE 401, students will be able to:

- Give examples of major science and engineering domains that use parallel programming and of the common types of algorithms that need large scale computing (e.g. the seven dwarfs of HPC).
- Demonstrate the ability to access, navigate, and use a variety of advanced computing systems with remote Linux connections (ssh, module systems, BASH, text editing, file systems, software install and building, environment variables, schedulers, etc.).
- Analyze software by conducting profile and benchmark studies with different parameters and options. Explain the bottlenecks and scaling of the code and present results to peers with predictions of times and scaling.
- Summarize the fundamentals of parallel programming concepts, including strong and weak scaling, Amdahl's Law, communication overhead, locks, and racing conditions.
- Explain differences between major parallel hardware and software paradigms. Compare and contrast the different approaches and be able to choose appropriate tools for a given problem.
- Develop and evaluate parallel codes using a variety of paradigms, including pleasantly parallel, shared memory parallelization (e.g. OpenMP), accelerator (e.g. GPUs and FPGAs), shared network parallelization (e.g. MPI, Hadoop, and Charm++), and parallel libraries (e.g. cupy, numba, mkl, fftw and blas).

The remainder of this paper discusses the major components of the design of CMSE 401, gives selected examples, and provides some limited analysis of the material though student feedback.

2. COURSE DESIGN

This course uses a "flipped classroom" model, where students spend class time doing hands-on practice activities with instructors and classmates, while traditional lectures are replaced with time outside of class reading and watching videos. When done correctly, this model of teaching is believed to provide a richer learning environment for students [4].

2.1 Jupyter Notebooks

All of the course materials are provided to the students using a Git repository and Jupyter notebooks [5]. The use of Jupyter notebooks may be confusing, since Jupyter notebooks are traditionally linked to Python, which is not a traditional language when considering computational performance and parallelization. However, Jupyter notebooks are rich and efficient communication tools that combine the benefits of a multimedia webpage, LaTeX, and executable example code. We develop Jupyter notebooks as a kind of

interactive textbook that, when used properly, is an effective way to organize course content and communicate with students. Although we do spend some time doing Python examples (all the students are familiar with Python from prerequisite courses), most of CMSE 401 is taught using the C family of languages (C, C++, and CUDA), which are also familiar to students from another C++ prerequisite.

One particularly useful Jupyter feature is the `%%writeout` "magic" command that allows the contents of a Jupyter cell to be written out to a file. This feature, in combination with the ability to execute bash commands using the "`!`" prefix, allows a Jupyter notebook to provide example code in any language, compile the code and run it all from within the notebook. In this way, students can have fully "literate" programming [6] with explanations right next to example code.

2.2 Course Hardware Resources

Students in CMSE 401 have access to a Jupyterhub server [7], the university's High Performance Computing Center (HPCC) [8], and multiple XSEDE resources though a teaching allocation [9]. This variety of hardware was chosen to expose students to different interfaces and help them generalize their understanding of computing hardware, in the hopes that they will develop a strong foundation of understanding and be able to figure out how to utilize new resources as they are developed in the future.

In addition to traditional hardware, during the first two weeks of the course, students were introduced to two different "portable" clusters. The first was a 7-node Raspberry Pi-based system, based on the Tiny Titan (https://tinytitan.github.io/), and the second was six (6) MacBook Pros connected using a small, off-the-shelf routing hub. Each of the laptops was installed with BCCD [10] inside of a virtual desktop. Students explored both systems during class as a hands-on learning activity focused on how to connect computers in a commodity cluster format. After building this commodity cluster in class and running examples, students also toured the campus HPC facilities. These hands-on lessons and in-person tours were motivating and helped students get excited about the topics they would be studying in CMSE 401.

We also experimented with a Jupyterhub server equipped with GPGPUs and CUDA support. Since CUDA would not work on many of the students' computers, this CUDA-enabled Jupyterhub server turned out to be a useful asset when introducing students to the language.

2.3 Assignments and Assessments

In the spring of 2019, CMSE 401 met three times a week for 70 minutes. Before each class, students completed a pre-class assignment, consisting of reading, videos, and practice problems. During class, the instructor reviewed questions that came up during the pre-class activities, and then students worked individually, in pairs, and in groups on example problems. Students also worked individually on more open-ended and in-depth problems in the form of homework assignments, which were due approximately every two weeks. Three times during the semester, students were given timed exams (two midterms and one final) to help assess their learning. Finally, at the end of the semester, students presented work on individual projects relating to topics taught in class. The remainder of this section describes these activities and assignments in more detail.

2.3.1 Pre-Class Assignments

These assignments are given to students in the week prior to class and include reading, multiple short videos (5–15 minutes each), example code, and practice questions. Students are expected to go through the materials before class, so that they are ready to participate in the in-class activities. These pre-class assignments are not graded; instead, students fill out a survey at the end of each pre-class assignment with questions similar to the following:

- Approximately how long (in minutes) did this assignment take for you to complete?
- What questions do you have, if any, about any of the topics discussed in this assignment after working through the Jupyter notebook?
- Do you have any further questions or comments about this material, or anything else that's going on in class?
- Based on what you've learned in the pre-class activities, what are you hoping to learn more about in class?

These questions are designed to get an idea of where students are struggling, so the instructor can address issues during class.

2.3.2 In-Class Assignments

Before class, instructors review all questions from the pre-class assignment survey, group them by topic, and develop a mini-lecture to help structure the class time most effectively. These mini-lectures vary in length depending on the issues students highlighted from the pre-class assignment. While the instructor has in-class activities planned, it is more important to address student questions and make sure they understand the pre-class assignments than to "get through" the day's materials.

After the mini-lecture, students work through the in-class notebooks. Students are expected to help each other out and work ahead on different questions if they get stuck on one particular problem. The goal here is to train students so that they are able to find solutions themselves, with instructors available to give suggestions and encouragement in order to avoid frustration. Instructors focus on helping students understand concepts and jargon; instead of solving problems for the students, instructors walk them through a variety of problem-solving techniques and suggest terms and phrases that they could use to search for helpful solutions on the Internet.

2.3.3 Homework Assignments

Homework assignments are designed to let students explore. Although many of them start out very similarly to in-class assignments, the idea for homework is to push students and get them solving multiple problems end-to-end. Students need to figure out how to download data, write code (including submission scripts), submit jobs to schedulers, interpret results, and visualize/share their results with their peers. A key component of the CMSE 401 homework assignments is a "creative component" that allows students to do something different and creative. Examples include a contest to see who can get the fastest code, trying out a new dataset, or exploring a software package. Again, the learning goals focus on exploration and problem solving in the context of large-scale computing in order to help students develop both familiarity with specific tools and creative problem-solving skills. We hope this approach also makes CMSE 401 more fun for students.

2.3.4 Exams

There are two midterms and a final exam in this course. Given the highly interactive and collaborative nature of the course, these exams provide an opportunity to individually assess student knowledge and skills. In all other assignments, students are expected to work together and support each other's learning, but that approach can make it difficult for instructors to identify areas where individual students are struggling. Timed exams, where students work alone, provide an assessment of individual knowledge and progress.

Of course, excellent students who have a deep understanding of the material may not perform well on timed exams — just as some students are excellent at taking tests but may not be able to perform as well in less structured scenarios. Exams in CMSE 401 are primarily seen as learning tools and try to reflect real-world scenarios. Thus, all exams are open-network: no one programs in a vacuum, and we are assessing students' ability to find answers and develop solutions using all of the resources that would be available to them in a real-life setting. Exams include four (4) problems, each with five (5) component questions. Although the questions relate to each other, we try to write them in such a way that they can be answered correctly even if previous answers are wrong. Students' informal feedback suggests that the exams are famously challenging — yet also rewarding. Even students struggling in the course have proven able to demonstrate their knowledge through these exams, and, although these exams are primarily used as a summative assessment tool, instructors are able to formatively assess progress and adjust course content and individual student learning goals. The exam grades are just one factor in students' overall learning, and thus are a relatively small percentage of students' final grades.

2.3.5 Student Projects

At the end of the semester, students present unique projects that demonstrate some aspect of what they learned over the semester. At a minimum, projects are expected to contain some sort of benchmark timing comparison. However, instructors are very flexible and encourage projects that relate directly to "real-world" problems that students are encountering in their work or other classes. For example, working with an existing faculty to download, install, and run a code on the HPC is considered an excellent project for CMSE 401. Another good project is to download a parallel library or language, get it working on the HPC, and do a benchmark comparison between some of its features (e.g., Tensorflow was quite popular). Students may not necessarily do much parallel programming in their projects; instead, we focus on the more common issue of workflow management and performance measurements, as these are the tools that researchers need to utilize advanced computing systems. Some example titles of student projects include:

- Ising Model Optimization
- Numerical Relativity with Numba
- MPI Poission Equation with MPI4Py
- OSCAR (Operational Research in Scala)
- Utilizing TensorFlow for Machine Learning in Biomedical Imaging
- Parallel Optimization of Sabermetric Quantifier
- Optimizing Garfield++ For Use in Simulating a Nuclear Detector
- Parallel Optimization in FLASH
- A Charm++ Parallel Stock Market Simulator
- Breast MRI Classification using TensorFlow
- Classifying Dog and Cat Images Using TensorFlow
- Penalization of TDCI

Student projects have multiple milestones through the semester, and students present progress to their peers. Although each student works on their project individually, time is given both in-class and out of class for students to share their work, and collaborative feedback and peer review are highly encouraged.

3. COURSE SCHEDULE AND TOPICS COVERED

The semester is divided into approximately 15 weeks, and the overall course covers the following major topic areas:

Major Topic 1 — Benchmarking and compilers

Major Topic 2 — Tools of the trade (remote systems, software installs and schedulers)

Major Topic 3 — Shared memory parallelization

Major Topic 4 — Accelerators

Major Topic 5 — Shared network parallelization

In practice, rather than being a linear progression of content, these topics are woven together throughout the semester. For example, in the first few weeks of class, students are exposed to a mini cluster (Raspberry Pi and laptop BCCD cluster) and are running a variety of parallel examples (shared memory, shared network, and GPUs). When they see these topics again later in the semester, the previous exposure has prepared them to jump in and program them on their own. A more detailed list of individual modules follows:

1. How a cluster is born — basic introduction to clusters, big-iron, little-iron and accelerators
2. Languages and Compilers — Benchmarking of both interpreted (Python) and compiled languages (C/C++), code optimization (compiler flags), introduce/review Big-O notation, and practice benchmarking.
3. Command line scripting (BASH), and accessing remote systems (SSH and SCP)
4. Schedulers — unique components of a shared system (schedulers and module system) and writing single core and pleasantly parallel examples to the scheduler (SLURM)
5. Shared Memory Parallelization — students are introduced to shared memory parallelization (OpenMP) and shown/encouraged to work on personal laptops
6. Shared Memory Parallelization — more about loops and programming options; goal is to become familiar with the variety of OpenMP capabilities and not necessarily become masters
7. Accelerators — introduction to accelerator coding (CUDA) and comparisons with shared memory programming, submitting jobs to a scheduler
8. More Accelerators — learning the basics of CUDA and writing their first program
9. More Accelerators — discuss the good and bad about CUDA, understanding thread blocks and tiling — where does it work and where does it fall apart?
10. Shared Network Parallelization — understanding network throughput and latency, benchmarking MPI code on different numbers of cores and nodes

11. Shared Network Parallelization — writing their first MPI program, debugging MPI code and improving performance
12. Hybrid Systems

While this is a rough outline of the topics, plenty of room was included in the 15-week schedule to allow instructors to adapt the pacing for more or less difficult topics, respond to student feedback, and give plenty of time for students to work on homework assignments and projects.

4. EXAMPLE

Whenever possible, instructors try to ground classroom examples using real-world scientific and engineering problems as motivation. Throughout the semester, students are shown how what they are doing connects directly to on-going research. This means we try to avoid spending too much time on "toy" examples such as sorting, calculating pi, or making games (although these examples can be useful). For the interested reader, samples of classroom materials have already been drafted and can be downloaded from the following git repository:

https://github.com/colbrydi/CMSE401_Examples.git

These examples include Jupyter notebooks that contain the following:

- A pre-class assignment that includes videos on using the command line and ssh keys
- An in-class assignment on CUDA programming on a GPU enabled node running Jupyter
- Shared Memory Parallelization example homework
- An example project template
- An example exam

These examples demonstrate the style and pedagogical approach of CMSE 401. The course is being offered a second time during the spring of 2021, and all of the course materials will be available as an Open Education Resource (OER) by the summer of 2021 at the course website (http://cmse.msu.edu/cmse401). Instructors interested in the instructor materials are encouraged to reach out to the author, as we are happy to provide additional instructor notes and answers.

5. STUDENT FEEDBACK

Although no formal evaluation of the materials was conducted for this paper, all university courses are evaluated using a 21-question survey, which 12 of the students completed. The students are able to choose a rating (from the following) for each question.

1 = (S) — Superior: exceptionally good

2 = (AA) — Above Average: better than the typical

3 = (AV) — Average: typical of courses or instructor

4 = (BA) — Below Average: not as good as the typical

5 = (I) — Inferior: exceptionally poor course or instructor

Please note that this course evaluation tool is known to be fairly biased and is being reworked by the university. The author also acknowledges that the results presented do not include a controlled reference point. However, the data do provide some context. A selected summary of the results can be reviewed in **Error! Reference source not found.**.

Table 1. Summary of student feedback grouped by type.

Composite Factors	Mean	Std
Instructor Involvement (Questions 1–4)	1.10	0.23
Student Interest (Questions 5–8)	1.73	0.38
Student-instructor Interaction (Questions 9–12)	1.31	0.51
Course Demands (Questions 13–16)	1.79	0.71
Course Organization (Questions 17–20)	1.55	0.56

Table 2 shows a sample of feedback questions given to the students. Based on this feedback and some informal polling, students reported that the course was challenging which is reflected in their end of semester survey evaluations. Specifically students found the course to be highly enjoyable (Question 21) while also being intellectually challenging (Question 6). Probably the biggest informal complaint was the difficulty and length of the homework (Question 14).

Table 2. Selected questions that reflect student feedback to the content and format of the course.

#	Question	Mean	Std
3	The Instructor's concern with whether the students learned the material	1.17	0.39
4	Your Interest in learning the course material	1.17	0.39
5	Your general attentiveness in class	1.83	0.39
6	The course as an intellectual challenge	2.25	0.75
7	Improvements in your competence in this area due to this course	1.42	0.67
10	The Student's Opportunity to ask questions	1.42	0.67
12	The appropriateness of the amount of material the instructor attempted to cover	1.33	0.65
13	The appropriateness of the pace at which the instructor attempted to cover the material	1.75	0.97
14	The contribution of homework assignments to your understanding of the course material relative to the amount of time required	2.08	1.00
15	The appropriateness of the difficulty of assigned reading topics	1.67	0.78
17	The course Organization	1.42	0.67
20	The adequacy of the outlined direction of the course	1.33	0.49
21	Your general enjoyment of the course	1.17	0.39

Overall, the instructors are also very satisfied with the course and plan to make significant improvements when it is taught again in the Spring of 2021.

6. ACKNOWLEDGEMENTS

Many of the details for this course were conceived during Shodor's 2018 Community Building for Parallel Computing Curriculum Development workshop. Shodor has taken an amazing lead in developing real world motivated examples in education (http://www.shodor.org/). I also want to specifically thank David Joiner for providing the example 1D wave equation code for the first benchmarking homework; this was a great first project and really helped start the discussion for compilers options and vectorization.

7. REFERENCES

[1] D. Colbry, M. Murillo, A. Alessio, and A. Christlieb, "Computational Mathematics, Science and Engineering (CMSE): Establishing an Academic Department Dedicated to Scientific Computation as a Discipline," *JOCSE*, vol. 11, no. 1, pp. 68–72, Jan. 2020, doi: 10.22369/issn.2153-4136/11/1/11.

[2] D. Silvia, B. O'Shea, and B. Danielak, "A Learner-Centered Approach to Teaching Computational Modeling, Data Analysis, and Programming," in *Computational Science – ICCS 2019*, Cham, 2019, pp. 374–388.

[3] G. Wilson, "Software carpentry: getting scientists to write better code by making them more productive," *Computing in Science & Engineering*, vol. 8, no. 6, pp. 66–69, 2006.

[4] L. Abeysekera and P. Dawson, "Motivation and cognitive load in the flipped classroom: definition, rationale and a call for research," *Higher Education Research & Development*, vol. 34, no. 1, pp. 1–14, Jan. 2015, doi: 10.1080/07294360.2014.934336.

[5] T. Kluyver *et al.*, "Jupyter Notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, 2016, pp. 87–90.

[6] D. E. Knuth, "Literate Programming," *Comput J*, vol. 27, no. 2, pp. 97–111, Jan. 1984, doi: 10.1093/comjnl/27.2.97.

[7] "JupyterHub," *GitHub*. https://github.com/jupyterhub (accessed Jun. 09, 2020).

[8] D. Colbry, W. Punch, and W. Bauer, "The Institute for Cyber-Enabled Research: Regional Organization to Promote Computation in Science," San Diego, California, USA, Jul. 2013.

[9] J. Towns *et al.*, "XSEDE: Accelerating Scientific Discovery," *Computing in Science Engineering*, vol. 16, no. 5, pp. 62–74, Sep. 2014, doi: 10.1109/MCSE.2014.80.

[10] B. Lu, "Use bootable Linux CD (BCCD) to teach cluster and parallel computing concepts: conference workshop," *J. Comput. Sci. Coll.*, vol. 24, no. 5, p. 142, May 2009.

Creative Assessment Design on a Master of Science Degree in Professional Software Development

Cathryn Peoples
Ulster University
United Kingdom
c.peoples@ulster.ac.uk

ABSTRACT

A Master of Science (MSc) conversion degree is one which retrains students in a new subject area within a fast-tracked period of time. This type of programme opens new opportunities to students beyond those gained through their originally-chosen degree. Students entering a conversion degree do so, in a number of cases, to improve career options, which might mean moving from an initially-chosen path to gain skills in a field that they now consider to be more attractive. With a core goal of improving future employability prospects, specific requirements are therefore placed on the learning outcomes achieved from the course content and delivery. In this paper, the learning outcomes are focused on the transferable skills intended to be gained as a result of the assessment design, disseminated to a cohort of students on a Master of Science (MSc) degree in Professional Software Development at Ulster University, United Kingdom. The coursework submissions are explored to demonstrate how module learning has been applied, in a creative way, to facilitate the assessment requirements.

Keywords

Conversion degree, Java, Master of Science (MSc).

1. INTRODUCTION

A Master of Science (MSc) conversion degree is one which retrains students in a new subject area within a fast-tracked period of time. A subject which would be taught within a three-year period in an undergraduate degree is instead taught during one intensive year. This type of programme opens new opportunities to students beyond those gained through their originally chosen degree. For a number, it is critical that what is learnt during the programme improves their employability potential in a field which is new to them. Students entering a conversion degree do so, in a number of cases, to improve career options, which might mean moving from an initially-chosen path to gain skills in a field that they now consider to be more attractive. The conversion degree can help them to gain the required knowledge and skillsets to do so.

With a core goal of improving future employability prospects, specific requirements are therefore placed on the learning outcomes achieved from the course content and delivery. In this paper, the learning outcomes are focused on the transferable skills intended to be gained as a result of the assessment design. Assessments presented in this paper were disseminated to a cohort of students on a Master of Science degree in Professional Software Development at Ulster University, United Kingdom. This is a conversion degree into Information Technology for students from non-IT backgrounds.

To understand the specific reasons that students had become part of the degree programme, and to avoid assuming that it was to improve their employability options, a survey was disseminated at the beginning of the academic year. This was done with the core objective of tailoring the teaching approach to meet their needs. When asked about their reasons for completing the degree, the majority of responses were focused around the fact that students were studying software development with the goal of employment in this field (Table 1). Going deeper into the reasons that students wanted to work in this field, they acknowledged it was due to their passion for technology, and because they identified the IT industry as one with a more certain chance of employment than others. When designing the teaching approach, it was therefore considered to be important to support students moving into this industry and to improve their prospect of doing so.

Transferable skill development was encouraged through the design of the module assessment, with students being assessed on their ability to apply knowledge gained during the teaching period while inherently developing transferable skills in doing so. To achieve this, assessments were shaped around the state-of-the-art in technology. The objective was to select news stories which are reported internationally and which would hopefully appeal to student interest, an approach in line with the belief that, *"… the concept of 'student engagement' is predicated on the belief that learning improves when students are inquisitive, interested or inspired, …*" [1]. Furthermore, "*When a topic connects to what students like to do, engagement deepens as they willingly spend time thinking*" [2]. In line with the objective of inherently gaining transferable skills, it was hoped that selecting popular news stories would give students an opportunity to develop their ability to discuss technical concepts comfortably, to become critical in the selection and application of their knowledge to solve real-life problems, and to appreciate the context of their learning in relation to the wider field.

It is acknowledged in [4] that soft skills such as those described above are particularly important, in recognition of the fact that, *"In an increasingly global, technological economy, they say, it isn't enough to be academically strong. Young people must also be able to work comfortably with people from other cultures, solve problems creatively, write and speak well, think in a multidisciplinary way, and evaluate information critically*" [4].

DOI: https://doi.org/10.22369/issn.2153-4136/12/2/11

Table 1. Student reasons for completing the degree.

Why are you studying for this degree?
"*Interested in computers, hope will lead to employment*"
"*Always wanted to learn coding, love technology, lack of programming knowledge held me back*"
"*To get a job in software development*"
"*Want career as a programmer or developer, as software development has good prospects and I enjoy logic and following patterns*"
"*Interested in programming language, could use skills from masters and linguistic knowledge to develop voice-activated software*"
"*To broaden skills for strong professional career in future*"
"*Interested in computers, but did PE teaching, employment limited*"

It is therefore in an attempt to bridge the gap between gaining the necessary knowledge and making the students employable that the assessments presented in this paper have been designed. The objective of this paper is to present a selection of the assessments which were designed to support student employability after completing the MSc conversion degree.

The remainder of this paper continues as follows. In Section 2, a literature review is presented, which considers how assessments can be designed to maximize student engagement with them, how to apply creativity in assessment design, and how to mark creative assessments consistently. This is followed in Section 3 with presentation of the creative coursework specifications which are the focus of this paper, together with a selection of the solutions in Section 4 to demonstrate how students harnessed their programming skills to fulfil the assessment requirements. Finally, the paper concludes in Section 5.

2. LITERATURE REVIEW

The assessments which are presented in this paper are used to examine a student's grasp of the entire module content and to apply their knowledge to a software development problem in a creative way; they are therefore summative assessments. As defined by the Council for the Curriculum, Examinations and Assessment [5]:

- "*Summative assessment usually takes place after pupils have completed units of work or modules at the end of each term and/or year.*
- *The information it gives indicates progress and achievement usually in grade-related or numerical terms.*
- *It's the more formal summing-up of a pupil's progress.*
- *The information can then be … used for certification as part of a formal examination course*."

Summative assessment is important due to the role it plays in determining a student's understanding of the module content at the end of the teaching period. It is therefore using this teaching material that it can be determined if module learning objectives have been achieved on a per student basis. There are high stakes associated with summative assessments, as it is not possible to revise submissions or receive feedback for improving future work, as is the case with the alternative assessment type, formative. It is therefore important that summative assessments are designed in a way which will maximize the opportunity that students can perform to the best of their ability.

King and English (2015) report that students respond most effectively to assessments which use real world scenarios, with problems contextualized in a way which students can understand [7]. This concurs with an opinion of Kearney and Perkins (2014), in that, real-world problems "*better engage them in their coursework and better prepare them for the world outside the classroom*" as opposed to "*research projects that do not have significance outside of the classroom*" [26]. This, essentially, describes the concept of authentic assessment. "*Authentic assessment is based on students' abilities to perform meaningful tasks they may have to do in the 'real world.' In other words, this form of assessment determines students' learning in a manner that goes beyond multiple choice tests and quizzes*" [27]. These findings therefore validate the effectiveness of the design approach applied to the assessments presented in this paper.

The conclusions reached by King and English (2015) are based on a study for which students were "recruited" as Optical Engineers and asked to build an optical instrument which could be used to spy on people. They concluded from this study that the assessment was appropriate for engineering students, given that it enabled ability to structure the stages of design, construction, and redesign in the development. This was in support of the fact that, "*meaningful STEM-integration is possible when students have the prior knowledge to apply to a well-structured engineering design task*" [7]. It is agreed in this paper that problem-solving ability is more likely when students are providing solutions to problems which they can contextualize, through either viewing them and/or having first-hand experience of the problems involved. It is with this understanding that the assessments presented in this paper have been designed. While it may be unlikely that students can have first-hand experience of the assessment problems presented in this paper, each domain in the assessment was chosen for the reason that the software might be one which they use in their day-to-day lives, e.g. Facebook, or because it is one highly likely to be of interest to anyone involved in technology, e.g. the emotion engine, Pepper the robot. Students were asked to use their technical knowledge to create similar systems, a form of situated cognition which helps them to recognize the placement of their abilities within the wider field and to understand the ways in which popular technologies are created in reality. This was done in recognition of, "*the need to draw explicit connections among topics for retention of learning*" [8]. Furthermore, for students who are new to the IT field, given the requirement for entry onto the degree programme that students have no prior IT education, it was hoped that using state-of-the–art technologies would help them to, "*keep pace with the rapid change and recent development in this era of globalization, …*" [9].

The approach of selecting technologies which students use in their day-to-day lives as the focus of each assessment was an action taken to "*facilitate creativity in which learners are motivated to discover things by themselves*" [9]. This was based on the fact that, "*Intrinsic motivators include fascination with the subject, a sense of its relevance to life and the world, a sense of accomplishment in mastering it, and a sense of calling to it*" [24].

The assessment specifications were presented in detail, and there was limited flexibility in what should be achieved. This is in spite of the fact that, "*There have been calls in the literature for changes to assessment practices in higher education, to increase flexibility and give learners more control over the assessment process*" [10]. Students were not restricted, however, in how they could achieve it, with marks awarded for the creative ways in which their technical knowledge was harnessed. As, "*Research has shown that creativity leads to intellectual development and brain growth, when*

creativity is nurtured …" [9], it was an objective to ensure that students focused their research efforts on how they achieved the solution, as opposed to what they were providing a solution to.

"*Creativity is the application of knowledge and skills in new ways to achieve a goal*" [11]. In having a creatively-designed assessment, it was hoped that this would encourage innovation in the solution presented by the student. This is in line with Kampylis and Berki (2014), who state that, "*Creative thinking is defined as the thinking that enables students to apply their imagination to generating ideas, questions and hypotheses, experimenting with alternatives* …" [12]. Asking students to build a replica system of one they are familiar with using programming techniques gained during the module teaching is one way to allow their creativity to be demonstrated. It is recognized, however, that, "*certain approaches to education may possibly foster greater creativity than others*" [11]. This statement is made specifically in relation to children of school age, with Montessori education using self-directed creativity and collaborative play [14], and Reggio Emilia education focusing on collaboration between children learning from their environment [15]. It is believed that the assessment presented in this paper has similarities to the Reggio Emilia approach, given that the assessment is based on a technology from the wider environment in which students operate. Furthermore, two of the principles of Montessori are to understand the systems of which the world exists and to support the imagination. Again, by basing the developments on software systems which are available from the wider environment, and by asking students to apply their knowledge from the modules which they have been taught, it is believed that both principles are met, helping to verify the suitability of the assessment design.

However, "*There is a lot of risk aversion in relation to assessment design. Staff fear being too creative in case their assessment is too challenging* …" [13]. Furthermore, it is recognized that marking creative work is challenging, given the desire to mark it quickly and the need to mark it consistently. Jackson (2005) of the Higher Education Academy notes that, "*Of all the aspects of creativity the one that poses the greatest challenge to teachers is how to assess/evaluate it*," identifying that some teachers just do not know how to assess such work [16]. This is significant, with the author going on to explain that, "*evaluation is critical to the very idea of creativity*" [16].

"*Students as well as academic staff … often ask the question as to how one marks creative writing. Indeed, they often wonder if it is even possible? Surely, they say, this is a subjective response, a matter of taste?*" [17]. Brookhart (2013), however, proclaims that, "*We can assess creativity …", and demonstrates how with a "Rubric for Creativity*" [18]. This is essentially based on evaluating work according to 1) how the ideas are combined together, with the highest levels of creativity being demonstrated when the, "*Ideas are combined in original and surprising ways* …", and 2) what is communicated, with creativity indicated with "*an original contribution that includes identifying a previously unknown problem, issue, or purpose*" [18]. Undoubtedly, there will likely always be some element of subjectivity when assessing creative work; however, a rubric provides the basis for a standardized approach to achieving this.

Computer software is one approach to assessment which can be electronically and automatically marked [19]. Automated marking of software programmes, however, is at odds with the concept of a creativity focus presented in this paper: Acknowledged by Brookhart, "… *with a broad concept as creativity, there's no single formula that will always work*" [18]. While Hill and Turner (2014) write about "*Code Originality*" [19], this is concerned about similarity between student work as opposed to an original design through its creativity. In [20], an automated system is proposed to assess software programs. This essentially tests its ability to compile, given the entry of input values chosen by the instructor. Therefore, to assess the creativity of a software program, it is unlikely to be possible to exploit automated marking, where, using Brookhart's rubric [18], the creativity is assessed based on the way in which the knowledge is put together.

It is believed that the assessments presented in this paper follow a transformative approach to learning. According to [21], transformative learning is described as occurring in situations where, "… *opportunities [are created] for critical thinking through providing content that introduces new ideas."* It was the objective that this opportunity was presented to students using state-of-the-art technologies which students were required to mimic in their software solutions. "*Transformation then happens in a community as students bounce ideas off one another*" [21]. It was the intention that this would be possible given that all students were set the same task. As part of transformative learning, it is also necessary for the instructor to, "*provide the opportunity for students to act on their new found beliefs*" [21]; it was hoped that this would be achieved through the overall assessment selection.

Practical programming solutions, such as the output required for the assessments presented in this paper, need to be designed in such a way that the software meets a specific target and achieves a certain goal. In addition to this practical level of functionality, submissions are also assessed according to how the module knowledge has been used. This can be contrasted with a research-based task, on the other hand, for which there can be an open and variable outcome, the case for which simply needs to be argued. Problem solving skills help students to work out how to reach the end goal, with critical thinking helping them to select the relevant elements from their learning and creative ability to apply them in a meaningful way. These are important qualities in support of employability: "*Merely having knowledge or information is not enough. To be effective in the workplace …, students must be able to solve problems to make effective decisions; they must be able to think critically*" [25].

The assessments which are presented in this paper have been designed in a manner which would support the International Baccalaureate (IB) Learner profile [6]. IB education is an international education programme delivered to students in school who are aged between three and nineteen years old, which is considered by some to be, "very well-respected by universities" [23]. One objective of the IB programme is to develop student skillsets such that they are internationally minded. It was hoped that this would be achieved in these assessments through the focus on international news stories in the field of technology, with recognition that technical capabilities vary widely across the world. Another IB programme objective is to develop thinkers, able to make decisions in relation to complex problems. It was hoped that this would be achieved in these assessments through empowering students with the necessary knowledge to solve a problem and giving them an interesting domain in which to apply them.

3. CREATIVE SOFTWARE ENGINEERING ASSESSMENT SPECIFICATIONS

At the beginning of the MSc degree programme, students are initially exposed to two six-week modules running one after the other on general Java software development skills, alongside firstly

a six-week module on Computer Hardware and then a six-week module on Operating Systems. In Semester 2, students progress to six-week modules on Data Structures and Databases run in parallel with one another, followed by two six-week modules also run in parallel on Concurrent Systems and Mobile Devices and Applications. The assessment specifications for Data Structures and Concurrent Systems are presented in the remainder of this section.

3.1 Concurrent Systems

Objectives of a module on Concurrent Systems include identifying the need for concurrent systems, providing an understanding of the issues and requirements to be addressed when designing and developing such systems, and providing opportunities to develop practical systems illustrating aspects of concurrent systems.

Two assignments for the Concurrent Systems module were based on Pepper, a humanoid robot [3]. Pepper is an emotion engine designed to make people happy. He does this through delivering jokes based on the emotions sensed from humans. Another assignment on Concurrent Systems was based on the creation of a system using Java to represent operation of the Android operating system. Both assignments were based around the development of systems where concurrent operation is ultimately the focus.

3.1.1 Concurrent Systems: Simulate Pepper Operation

The first assignment was to implement a program using Java that would simulate operation of Pepper. It was required to be a multi-threaded solution, with each thread representing sensed data being fed into the operating system for processing from Pepper's ears, eyes, and hands.

More specific system requirements were also defined in the specification in relation to each thread: Threads should be created to represent sensed data from Pepper's ears, each of which requires 20 bytes of RAM per second. Similarly, Pepper's eyes require 30 bytes of RAM per second, and Pepper's hands require 40 bytes of RAM per second. The total system capacity is 1,000 bytes of RAM. The OS needs 300 bytes of RAM to run, and the supporting activities, including drivers, required by the operating system take up 200 bytes of RAM. After loading the OS so that it is ready to accept workload, there are subsequently 500 bytes remaining for application and other system activity. The CPU processes workload at a rate of 200 bytes per second.

This scenario is essentially the Producer-Consumer problem, with a requirement for multi-process synchronization. The queue into which sensed data arrives is a fixed-size buffer, with a restricted amount of space to support application and system workload. The producers generate the sensed data, passing it to the ports into the operating system via the buffer, which is shared with a consumer. At the same time that the producer is producing workload, the consumer is consuming the data, removing it from the buffer one piece at a time. The robot's engines can be considered to be the consumer, processing jobs and enforcing decisions from the system. The challenge is to ensure that the producer will not try to add data into the buffer if it's full, and that the consumer will not try to remove data from an empty buffer. To avoid these occurrences, the producer either goes to sleep or discards data if the buffer is full. The next time the consumer removes an item from the buffer, it notifies the producer, who starts to fill the buffer again. In the same way, the consumer can go to sleep if it finds the buffer to be empty. The next time the producer puts data into the buffer, it wakes the sleeping consumer.

Each thread runs for a period of time dependent on the sensed motion duration, or the number of bits being stored to disk and the CPU's processing capability. RAM availability influences the operating system's ability to support threads simultaneously.

Marks were awarded in this assessment for achievement of the required functionality (35%), technical quality of the program code (35%), dealing correctly with multiple threads (10%), adherence to good programming practices (10%), and clarity of the instruction sheet/booklet (10%).

3.1.2 Concurrent Systems: Pepper as a Client-server System

For the second Pepper-based assignment, students were organized into pairs and were required to implement a program using Java to simulate a connection between Pepper as the client and a remote database as the server. A server was required to hold tailored responses to be delivered by Pepper based on the "sensed" emotions of humans interacting with the robot. One student in the pair was required to be responsible for the client program and one student for the server program. As Pepper is an emotion engine, the database was required to return jokes to a user when a sad emotion is sensed. As a restriction, a joke was allowed to be returned once only within a session. The user should also have the capability to select a genre of a joke. The "database" on the server side of the system could be held within arrays. The system was required to use TCP sockets at the client and server sides of the network to support communication.

Marks were awarded for achievement of the required functionality (35%), technical quality of code (35%), dealing correctly with TCP socket programming (15%), communication issues (robustness of software, error handling) (5%), and adherence to good programming practices (5%).

3.1.3 Concurrent Systems: Android OS

In another Concurrent Systems assignment, students were required to implement a program using Java to simulate a multi-threaded Android operating system. The system was required to support simultaneous application threads, including a thread to start a BubbleWitch2 session lasting 10 seconds and requiring 100 bytes of RAM per second, and a thread to start a 20-second Spotify stream requiring 250 bytes of RAM per second. A system and management thread was also incorporated, requiring 50 bytes of RAM per second and to execute for a random duration of time once invoked. Controlling execution of the system for the purpose of demonstrating its operation, students were required to implement a thread to install a new security update of 2KB, which requires 150 bytes of RAM per second while installing. Overall capacity within the system is 1,000 bytes of RAM; the OS needs 300 bytes of RAM to run, and the drivers consume 200 bytes of RAM. After loading the OS so that it is ready to accept workload, there are 500 bytes remaining for application and other system activity. The CPU processes workload at a rate of 200 bytes per second.

Marks were awarded for technical quality of the implementation (35%), achievement of the functional requirements (35%), dealing correctly with multiple threads and robustness of the software (10%), and structure and presentation of the program (20%).

3.2 Data Structures

An objective of a module on Data Structures includes to provide students with skills in using and implementing abstract data types. The development of a social networking website, similar to Facebook, was a coursework assignment which lent itself easily to

a module on Data Structures. Students were asked to develop a social networking website using as many different data structures as possible. These would typically be used to retain data associated with each user account. Students were also asked to implement algorithms at the back-end of the system to search this stored data, so, for example, identifying people who might be their friends, or providing a reminder of a friend's upcoming birthday. Their design would be assessed in terms of the efficiency of the operation of the site, a feature which would be influenced by the most efficient data structures, sorting and searching techniques taught during the module. Certain data structures are more appropriate for certain types of information than for others, and it was the student's responsibility to select the most appropriate structure and to justify their choices.

It was therefore an assumption of the system implementation that a repository of information would be retained to support demonstration of the full system functionality, including options on pages which a user may "like" and a number of users who hold accounts with the system which a user may add as a friend.

Appropriate structure selection was important in relation to the efficiency and reliability of its operation, where efficiency is measured by the latency to execute a request and volume of memory processed when executing an operation, and reliability is measured in terms of the effectiveness of recommendations made by the system for individual users.

Conditions were also required to be applied in the scenarios implemented, such as, for example, the requirement that adding a new account would require checking that the user does not already have an account in the system — this would make demands on a searching technique implemented — or that deleting a user account would require deleting the user as a friend of other system users — a feature also requiring efficient searching techniques.

Marks were awarded for achievement of the functional aspects (30%), technical quality of the implementation (40%), effectiveness of the design choices (15%), and originality of the design (15%).

4. SOFTWARE ENGINEERED SOLUTIONS

In Section 4, a selection of the programme code solutions are presented. These are used to demonstrate the ways in which students harnessed the technical concepts learnt during each modules' teaching to create the software solution. The solutions presented are specific to one specific student for each module.

4.1 Concurrent Systems: Pepper Operation Simulation

In a software solution for the first Concurrent System assignment, a student created a Producer and Consumer class, which would communicate with each other via a shared buffer to achieve simultaneous movement of Pepper's limbs.

4.1.1 Initializing the Pepper Programme

To begin program execution, a buffer is initialized with 500 bytes of capacity (queue) (which fulfils the requirement set out in the assessment specification that the original 1,000 bytes available is also consumed by the operating system and the drivers which it needs). A random period of time (length) defines the length of time which the program should execute; a requirement of the coursework is that the program runs through all operations to demonstrate concurrent execution of the robot, and not to require external input to trigger events. After the processor and system and management threads are started, the moveable elements of the robot are invoked, including Pepper's eyes, ears, and hands.

```
public class Pepper {
   public static void main(String[] args) {
      PepperBuffer queue = new PepperBuffer (500);
      Random length = new Random();
      new PepperCPU(queue).start();
      new PepperSenses("System & Management", 50,
      queue, length.nextInt(20), 10).start();
      new PepperSenses("eye 1", 20, queue,
         length.nextInt(20), 10).start();
      new PepperSenses("eye 2", 20, queue,
         length.nextInt(20), 10).start();
      new PepperSenses("ear 1", 30, queue,
         length.nextInt(20), 10).start();
      new PepperSenses("ear 2", 30, queue,
         length.nextInt(20), 10).start();
      new PepperSenses("hand 1", 40, queue,
         length.nextInt(20), 10).start();
      new PepperSenses("hand 2", 40, queue,
         length.nextInt(20), 10).start();
   }// main
}//class
```

At this stage, the system producer and consumer classes are required to both push and pull workload to and from the shared buffer.

4.1.2 Pepper Producer Class

The Pepper Producer class achieves the functionality of generating workload, in the sense of movements from each of Pepper's 'body' parts. These are added into the shared buffer for the processing. In a live Pepper deployment, each sense consumed from the shared queue would result in an aspect of Pepper moving.

The concurrent threads of Pepper's system are represented in this solution using the PepperSenses class.

```
public class PepperSenses extends Thread {
   private int amountOfRam, lengthOfTime;
   private PepperBuffer queue;
   public PepperSenses(String sense,
      int amountOfRam, PepperBuffer queue,
      int lengthOfTime, int Priority) {
      this.setName(sense);
      this.amountOfRAM = amountOfRAM;
      this.queue = queue;
      this.lengthOfTime = lengthOfTime;
      this.setPriority(priority);
   }
   public void run() {
      for (int seconds = lengthOfTime; seconds >
      0; seconds--) {
      if (((int) (Math.random() * 2) +1) == 1) {
         this.pause();
      }
      else {
         this.actionOccurred();
      }
      try {
         sleep(1000);
      } catch (InterruptedException e) {}
      }
   }
}
```

Once the program is initiated (using run()), a pause() is invoked at random to force the program to wait for a period of time to simulate delay between activity associated with each of Pepper's senses. actionOccurred() is invoked during intervals outside the pause periods, which forces workload to be added into the shared buffer queue:

```
public void actionOccurred() {
   queue.put(this);
}
```

This workload is queued for the CPU, as the consumer, to extract and process. This is possible due to creation of the PepperBuffer object within PepperSenses, and initialization of the availableRAM value:

```
public PepperBuffer(int totalRAM) {
   if (totalRAM <= 0)
      throw new IlegalArgumentException("Size
      is illegal");
   this.totalRAM = totalRAM;
   this.availableRAM = totalRAM;
}
```

The put() method within PepperSenses simulates producer functionality. The wait() method is invoked until there is space in the buffer to allow the job to be placed there; notifyAll() is then invoked to communicate to the consumer that there is workload available to be consumed:

```
public synchronized void put(PepperSenses sense) {
   int ram = sense.getAmountOfRAM();
   while(noSpace(ram)) {
      try {
         wait();
      } catch(InterruptedException ex) {
      }
   }
   buffer.add(sense);
   availableRAM -= ram;
   notifyAll();
}
```

The noSpace() method checks if the buffer is full, preventing new jobs from being added, in which case the producer will wait:

```
public synchronized Boolean noSpace(int ram) {
   return (availableRAM < ram);
}
```

4.1.3 Pepper Consumer Class

The consumer removes workload from the buffer as each sensed event becomes available and the consumer is notified of its arrival.

```
public class PepperCPU extends Thread {
 public void run() {
  while(true) {
    while(cpuAvailable >= 0) {
     PepperSense sense = queue.get();
      setCpuAvailableRemove(sense.getAmountOfRAM());
      sense.sleep();
    }
    try {
      this.sleep(1000);
   }catch (InterruptedException e) {
      Thread.currentThread().interrupt();
   }
   setCpuAvailable();
 }
}
```

The consumer first checks if the buffer is empty, which will be the case when there are no jobs to remove. In this event, the consumer will wait until workload has been added to the queue. When workload is present in the queue, on the other hand, jobs are removed by invoking the get() method:

```
public synchronized PepperSenses get() {
   PepperSenses sense;
   while(isEmpty()) {
      try {
         wait();
      } catch(InterruptedException ex) {
      }
   }
   sense = buffer.remove(0);
   availableRAM += sense.getAmountOfRAM();
   notifyAll();
   return sense;
}
```

The consumer then removes the workload and invokes notifyAll() to inform the producer that there is increased space in the buffer to accept new workload. The availableRAM value is also updated to reflect the amount of RAM now available in the queue in response to the robot movement having been dequeued.

The concept of Pepper provided a suitable context to support the development of a concurrent system, fulfilling two of the module objectives to "identify the need for concurrent systems" and to "provide opportunities to develop simple practical systems illustrating specific aspects of concurrent systems." Furthermore, creation of the solution required that students had an appreciation of the main components of a concurrent system, such as the need to have a shared buffer, and one producer and one consumer to use it. This helped to fulfil the learning objective of the module, to "provide an understanding of the issues and requirements to be addressed when designing and developing such systems." Additionally, having this awareness required that the fourth learning objective had been fulfilled to "introduce the underlying principles of concurrent systems." Organizing the development around the concept of Pepper helped students to appreciate the wider context within which their learning exists.

4.2 Concurrent Systems: Pepper as a Client-Server System

In the solution, the client and server were required to connect to the same port in order to communicate, continuously listening on the same socket for communications between each other.

In the solution presented, the client is created and initialized using the PepperClient class:

```
public class PepperClient {
   public static void main(String args[]) {
      Scanner keyboard = new Scanner(System.in);
      String serverName = "193.61.167.145",
      username="", response;
      Socket serverSocket;
      int serverPort = 3829;
      InputStream isFromServer;
      OutputStream osToServer;
      DataInputStream disFromServer;
      DataOutputStream dosToServer;
      try {
         serverSocket = new Socket(serverName,
         serverPort);
         isFromServer =
         serverSocket.getInputStream();
         osToServer = serverSocket.getOutputStream();
```

```
            disFromServer = new
            DataOutputStream(osToServer);
         } catch(Exception e) {}
      }
   }
```

PepperClient is associated with port 3829. Data streams are established to facilitate the communications, in directions both to the server (OutputSteam osToServer) and from the server (DataInputStream disFromServer). Sockets are additionally created to support each data stream.

Data streams and sockets are also established on the server side in parallel:

```
public class PepperServer {
   public static void main(String[] args) throws
      IOException {
         InputStream is;
         OutputStream os;
         DataInputStream disFromClient;
         DataOutputStream dosToClient;
         Socket clientSocket;
         ServerSocket listenSocket;
         int clientInt;
         int serverPort = 3829;
         listenSocket = new ServerSocket(serverPort);
         clientSocket = listenSocket.accept();
         is = clientSocket.getInputStream();
         os = clientSocket.getOutputStream();
         disFromClient = new DataInputStream(is);
         dosToClient = new DataOutputStream(os);
         boolean live = true;
   }
}
```

To facilitate the end-to-end communication, the server also listens on port 3829. After the definition of these classes, the client and the server are in a position to interact with one another. This requires capability to communicate the mood of the user who is interacting with Pepper. It also requires that the database of jokes is created for return to the user in the event that their mood is one of sadness.

Due to the fact that this system is a simulation of a robot, it was necessary to explicitly articulate the simulated user's mood. In a live system, this information might be autonomously collected using the sensors for Pepper's eyes to identify their facial characteristics, or Pepper's ear sensors to detect what the user is saying. In the simulated system, this functionality is achieved by the user entering their emotion using a keyboard, in response to a prompt delivered by PepperClient:

```
public static int askForEmotion(String name) {
   Scanner keyboard = new Scanner(System.in);
   String feeling;
   System.out.print("Tell me " + name + " do you feel
   sad (Y/N: ");
   feeling = keyboard.nextLine();
   if (feeling.equalsIgnoreCase("N")) {
      System.out.println("\nHow do you feel " + name
      + "?" +
      "\n1. Happy" +
      "\n2. Angry" +
      "\n3. Hungry" +
      "\n4. Scared" +
      "\n5. I want you to leave me alone");
      return (keyboard.nextLine().charAt(0)-48);
   }
   else {
      return 6;
   }
}
```

PepperClient asks the user to enter an integer which indicates their mood, in the instance that they are not feeling sad. In the instance that the user reports that, in fact, they are feeling sad, this will be communicated to PepperServer. Jokes are then returned to the user from PepperServer.

```
while(live) {
   emotion = askForEmotion(userName);
      switch(emotion) {
         …
         case 6:
         jokeCategory = jokeGenre(username);
         String reply;
         int noOfReplys;
         dosToServer.writeInt(jokeCategory);
         noOfReplys = disFromServer.readInt();
         for(int index=0; index < noOfReplys;
            index++) {
            reply = disFromServer.readUTF();
            System.out.println(reply);
         }
      break;
      default:
      System.out.println("I didn't understand");
      }
   }
}
```

In the case of selections 1, 2, 3, 4, or 5, Pepper will take the action of exiting the user from the session by closing the socket, or will return a statement to the user depending on them being happy or hungry, essentially any case where they are not sad. In the case that the user is sad, Pepper will return a joke to the user from the joke database. Pepper attempts to return a joke which is personalized for the user, by considering their preferred genre of joke. The user therefore has a choice of selecting a knock-knock joke, a one-liner joke, a chicken-crossing-the-road joke, a computer joke, or a pun.

Pepper similarly provided an appropriate opportunity for students to exploit their learning of Java socket programming in a remote client-server setup. This provided an opportunity for students to demonstrate their "understanding of the issues and requirements to be addressed when designing and developing such systems," by appreciating how a robot in fact interacts with a remote server when responses are selected for return. Again, the "need for the concurrent system" is highlighted in this situation, where it is essential that interactions are delivered in the correct order in the support of a meaningful "conversation."

4.3 Concurrent Systems: Android OS

An OperatingSystem class is created in one implementation, which creates and initializes the shared buffer that retains workload added by the producer and removed by the consumer.

```
class OperatingSystem {
   private int contents;
   private int buffer = 1000, driver = 200,
   operSRun = 300, consumptionRate = 200,
   workload = buffer- (driver+operSRun);
   private long startTime;
   private long endTime, waitTime;
   private int waitCount = 1;

   public synchronized int get() {
      while(workload <= (driver + operSRun)) {
         try {
            wait();
         } catch (InterruptedException e) {}
      }
```

```
      workload = workload – consumptionRate;
      notifyAll();
      return consumptionRate;
      }
      public synchronized void put (int amount) {
         while (workload >= buffer) {
            startTime = System.currentTimeMillis();
            try {
               wait();
            } catch (InterruptedException e) {}
            endTime = System.currentTimeMillis();
            waitTime = (endTime – startTime);
            averageRequestTime(waitCount, waitTime);
         }
         contents = amount;
         workload = workload + amount;
         notifyAll();
      }
   }
```

The OperatingSystem class initializes attributes used to support the scenario, including the buffer size. It also contains the necessary put() and get() methods, which are responsible for allowing workload to be added and removed to and from the queue. When workload is inserted into the queue (put(int amount)), the system first checks that the size of the queue, if the workload were to be added, does not exceed the maximum possible size. In the case that the action would result in a queue of unfeasible size, the wait() method is invoked, meaning that the thread will wait until it has been notified that workload has been removed and that there is now space in the queue. Once the workload can be added, the queue size will be updated, and the consumer thread will be notified that there is workload available for processing, using notifyAll(). The newly added workload will be removed.

Application classes are also defined to support execution of system threads. These threads are responsible for adding workload to the shared queue. As one example, a class Spotify simulates activity for 20 seconds and requires 250 bytes of RAM per second:

```
   private class Spotify extends Thread {
      private OperatingSystem operatingSystem;
      private int number, loops;
      private int long startTime;
      private long endTime;

      public Spotify(OperatingSystem os, int number,
      int duration) {
         operatingSystem = os;
         this.number = number;
         this.loops = duration;
      }
      public void run() {
         startTime = System.currentTimeMillis();
         for (int i = 0; i < loops; i++) {
            operatingSystem.put(250);
            try {
               sleep(1000);
            } catch (InterruptedException e) {}
         }
         endTime = System.currentTimeMillis();
      }
   }
```

Each application thread extends the Java Thread class. The constructor for the thread sets up the number of the thread and the duration of the thread; the duration is used to ensure that the thread executes for the necessary interval of time. Enforcing that the thread executes sleep(1000) results in a one second delay between operatingSystem.put(250) being invoked with each loop iteration, therefore helping to simulate the thread running for a period of time.

The invocation of each application thread simultaneously is controlled using the System class.

```
   public class System {
      private static int count;
      private static long time = 0;
      private static long average;

      public static void main(String[] args) throws
      InterruptedException {
         long startTime;
         long endTime;

      OperatingSystem os = new OperatingSystem();
      BubbleWitch2 bWitch2 = new BubbleWitch2(os, 1,
         10);
      Spotify Spotify = new Spotify(os, 2, 20);
      SystemAndManagement sysAndManagement = new
         SystemAndManagement(os, 3);
      CPU processor = new CPU(os, 1);
      SecurityUpdate securityUpdate = new
         SecurityUpdate(os, 4, bWitch2, Spotify,
         sysAndManagement, processor, 15, 2000);
      startTime = System.currentTimeMillis();

      bWitch2.start();
      Spotify.start();
      sysAndManagement().start();
      securityUpdate.start();
      processor.start();
      securityUpdate.setPriority(1);
      securityUpdate.join();
      processor.stopRunning();
      average = time / count;

      endTime = System.currentTimeMillis();
      }
      public static void averageRequestTime(int
         waitCount, long waitTime) {
         count += waitCount;
         time += waitTime;
      }
   }
```

The BubbleWitch2 and Spotify threads are initialized using their constructors, with information which includes the OperatingSystem with which they are associated, the thread number, and the duration of time which they are required to run. The systemAndManagement thread does not indicate a duration, as it is required to execute throughout the lifetime of the application. Similarly, the processor thread will also be available continuously.

When the SecurityUpdate thread is created, the other application threads are passed through the constructor so they can be joined with SecurityUpdate. Java's .join() method supports one thread waiting while other threads complete execution. Invocation of securityUpdate.join() will result in the securityUpdate thread being paused when another thread with a higher priority is in an executable state. The securityUpdate thread is given a priority of 1. This is the lowest priority which may be assigned to a thread, and enforces that this thread is executed with minimum priority.

```
   public class SecurityUpdate extends Thread {
      private OperatingSystem operatingSystem;

      public SecurityUpdate(OperatingSystem os, int
         number, BubbleWitch2 bWitch2,
```

```
      Spotify Spotify,
      SystemAndManagemetn systemAndManagement, CPU
      processor, int desired, int max) throws
      InterruptedException {
         operatingSystem = os;
         this.number = number;
         this.loops = desired;
         this.max = max;

         bWitch2.setPriority(10);
         Spotify.setPriority(10);
         systemAndManagement.setPriority(10);
         processor.setPriority(10);

         bWitch2.join();
         spotify.join();
         systemAndManagement.join();
         processor.join();
   }

   public void run() {
      …
   }
}
```

The priorities assigned to application threads enable processor capacity to be assigned to each application as it becomes available, depending on the duration of the thread. Invocation of the .join() method in association with each application thread enforces that the securityUpdate thread will be the last to execute.

Use of the Android operating system allowed students to appreciate the role which concurrent systems play in their day-to-day lives. The module learning objective to "provide an understanding of the issues and requirements to be addressed when designing and developing such systems" was highlighted in this assignment, with the need to consume residual memory to support an operating system and additional drivers and then support the application threads simultaneously around that.

4.4 Data Structures

There were two primary aspects of the Data Structures assessment, firstly in terms of the structures used to hold data, and secondly in terms of the algorithms used to organize and search the data structures. Assessments for this module are therefore considered from these perspectives in the following sections.

4.4.1 Setting Up Data Structures

A stack is used to retain the news feed for the social network using the Java Stack class: Following the Facebook approach, the news feed presents the most recent news item, the most recently added element to the stack, first. This is possible due to the fact that the class operates on a last-in first-out approach (LIFO). Extracting the most recent news update can therefore be achieved by "popping" the top item from the stack. New items of news are added by "pushing" them onto the stack, and are continuously pushed down the stack as new items are added.

```
class Stack {
   private int maxSize;
   private User[] stackArray;
   private int top;

   public void push(User j) {
      stackArray[++top] = j;
   }
   public User pop() {
      return stackArray[top--];
   }
   public Boolean isEmpty() {
      return (top == -1);
   }
   public Boolean isFull() {
      return (top == maxSize – 1);
   }
}
```

When items are pushed on or popped from the stack, a counter is maintained (top), allowing the size of the stack to be captured.

As another example of a data structure implemented, an array is used to capture the personal details of users of the system:

```
allUsers[j] = new User(firstName, lastName,
dateOfBirth, homeLocation, emailAdd, password,
employer, school);
```

The array effectively stores several items of the same type.

4.4.2 Organising and Searching Data Structures

A user is added as an object into a sorted list of system members:

```
public void addUser(String firstName, String
   lastName, Date dateOfBirth,
   String homeLocation,
   String emailAdd, String password,
   String employer, String school) {
      for (int j = 0; j < totalUsers; j++) {
      if (allUsers[j].getEmailAdd().
       compareTo(emailAdd) > 0)
      break;
      }

      for (int k = totalUsers; k > j; k--) {
         allUsers[k] = allUsers[k – 1];
      }
      allUsers[j] = new User(firstName, lastName,
      dateOfBirth, homeLocation, emailAdd,
      password, employer, school, currentStatus,
      statusTime);
      totalUsers++;
   }
}
```

The correct position in the array is identified by searching through email addresses, which are sorted into alphabetical order using .compareTo(emailAdd). This compares the email address being added with the email address at the position in the array currently being searched. If the result of the comparison is a positive integer, the email address lexicographically follows the argument string, and it should be added at this position. Items currently in the array beyond this point are shifted down by one position to make space for the new item being added.

New friends are added into a friend list, again sorted according to email address and using an insertion sort:

```
public void emailSort() {
   int in, out;
   for (out = 1; out > totalUsers; out++) {
      User temp = allUsers[out];
      in = out;
      while (in > 0 &&
      allUsers[in – 1].getEmailAdd().
      compareTo(temp.getEmailAdd()) > 0) {
         allUsers[in] = allUsers[in – 1];
         --in;
      }
      allUsers[in] = temp;
   }
}
```

An insertion sort on email address is used to organize the data associated with an individual account so that it is organized in the most efficient way when it comes to search the data. The insertion sort considers each list element from left to right, comparing each one by one. It places the data element in its correct location within the sorted list. The process is repeated until there are no unsorted elements remaining. The algorithm therefore operates by comparing the current email address with the email address which precedes it.

Capability was integrated to support account deletion. This requires that the account is also removed from their friend' lists.

```
public void delete(int userIndex) {
   int indexInFriendLists;
   for (int I = 0; I < totalUsers; i++) {
      indexInFriendLists =
      allUsers[i].friendFind(
      allUsers[userIndex].getEmailAdd());
      if (indexInFriendsLists >= 0) {
        for (int startPosition =
        indexInFriendLists; startPosition <
        allUsers[i].getFriendCount();
        startPosition++) {
          allUsers[i].setFriendsLists(
          startPosition, allUsers[i].
          getFriendsList()[startPosition + 1]);
        }
        allUsers[i].setFriendCount(-1);
      }
   }
   for (int startPosition = userIndex;
   startPosition < totalUsers; startPosition++) {
      allUsers[startPosition] =
      allUsers[startPosition + 1];
      totalUsers--;
   }
}
```

Deleting requires that the friendsList array for each account holder is also searched, and the person who is deleting their account is also removed from their list of friends.

A binary search is applied to find friends within a user's friend list, with the assumption that a friend of a friend is a plausible option for a friend recommendation.

```
public int friendFindEmail(String email,
int lowerBound, int upperBound) {
   int curIn;
   curIn = (lowerBound + upperBound) / 2;

   if (lowerBound > upperBound)
      return -1;
   else if
   (friendsList[curIn].
   getEmailAdd().compareTo(email) == 0)
      return curIn;
   else {
      if (friendsList[curIn].getEmailAdd().
      compareTo(email) < 0)
         return friendFindEmail(email, curIn + 1,
         upperBound);
      else
         return friendFindEmail(email, lowerBound,
         curIn – 1);
   }
}
```

The binary search compares the target with the middle list element. If the values are not equal, the half where the target cannot reside is eliminated, and search continues in the remaining half until successful.

A method is incorporated to search for friends which have a birthday in the current month or in the next month:

```
public void displayBirthdays(int currentUserIndex) {
   Date now = new Date();
   Stack birthdayStack = new Stack(totalUsers);

   for (int i = 0; i <
   allUsers[currentUserIndex].getFriendCount();
   i++) {
      if (allUsers[currentUserIndex].
      getFriendsList()[i].getDateOfBirth().
      getMonth() == now.getMonth() + 1) {
         birthdayStack.push(allUsers[
         currentUseIndex].getFriendsList()[i]);
      }
   }

   if (!birthdayStack.isEmpty()) {
      while (!birthdayStack.isEmpty()) {
         User temp = birthdayStack.pop();
         System.out.println("Birthdays!");
         System.out.println(temp.getFirstName() +
         " " + temp.getLastName());
      }
   }
}
```

A number of friends are associated with each user. The system therefore captures the number of friends and searches through their birthdays to find if any have a birthday in the current month, with the objective of providing the user with a reminder. These identified contacts are subsequently "pushed" onto a stack for temporary storage during the user's session, such that they may be output for information. If the stack is not empty, a user is informed that birthdays of their friends according to their contact list are coming up; the relevant users are popped from the temporary stack.

As with simulation of the Android operating system as a concurrently operating environment with which students have first-hand and day-to-day experience, the social media Facebook-type platform similarly has such relevance to students. Students could appreciate, for example, how functionality is provided by the newsfeed feature; harnessing the use of abstract data types, as a learning objective of the Data Structures module, allowed students to appreciate the software development operating in the backend to support popular social media environments.

5. CONCLUSIONS

A conversion degree provides new opportunities, particularly in relation to employability, by training students in areas for which they were not previously academically accredited. When students were asked about their expectations for the study year, both positive (Table 2) and negative (Table 3) angles were presented.

Student ambitions were clear (Table 2), with expectations including gaining new skills and knowledge, and ideally a job. Coursework cropped up as a weight about which students had negative expectations (Table 3). When considered in relation to their desire for improved employability options, it was therefore important to support students in their ambitions post-degree and remove their coursework fear. It is believed that the creative assessment design helped to bridge these gaps, by exposing students to state-of-the-art technology on an international basis, helping them to understand

the software developments which are essential in their support at the back-end and encouraging the application of knowledge in new ways. In doing so, the creative assessment designs provided a mechanism to facilitate student innovation in their output.

Table 2. Student expectations of the year (positive).

What are you looking forward to this year?
"*Challenge of learning something completely new*"
"*To meet like-minded individuals to create something together to leave with business ideas*"
"*Gaining a masters and a job*"
"*Learning practical programming skills, how computers work, how software programs are created*"
"*To be challenged and learn new skills*"
"*Getting a job*"
"*Gaining experience in software programming*"
"*Challenges that the course will bring*"

Table 3. Student expectations of the year (negative).

What are you not looking forward to this year?
"*Travelling to and from university*"
"*Travelling, exams, formal aspects of course, struggle to sit down and write about a task*"
"*Hard to learn a new discipline at such a fast pace*"
"*Using mathematical formula*"
"*Coursework*"
"*Falling behind, not understanding something*"
"*Exams*"
"*Parts of the lecture which are not understood, mainly due to the terminology*"
"*Written assignment*"

6. REFERENCES

[1] Great Schools Partnership, "*Student Engagement*," The Glossary of Education Reform; Available: https://www.edglossary.org/.

[2] J. McCarthy, "*Learner Interests Matters: Strategies for Empowering Student Choice*," Edutopic, Aug. 2014; Available: https://www.edutopia.org/blog/differentiated-instruction-learner-interest-matters-john-mccarthy.

[3] SoftBank Robotics, "*Who is Pepper*?" Online; Available: https://www.softbankrobotics.com/emea/en/robots/pepper.

[4] C. Gewertz, "*'Soft Skills' in Big Demand*," Education Week, Jun. 2007; Available: https://www.edweek.org/ew/articles/2007/06/12/40soft.h26.html.

[5] Council for the Curriculum, Examinsation and Assessment, "*What is Summative Assessment?*" Online; Available: http://ccea.org.uk/curriculum/assess_progress/types_assessment/summative.

[6] International Baccalaurete, "*The IB Learner Profile*," Online; Available: https://www.ibo.org/benefits/learner-profile/.

[7] D. King and L. D. English, "*Engineering Design in the Primary School: Applying STEM Concepts for Build an Optical Instrument*," International Journal of Science Education, Dec. 2016, pp. 2762–2794; DoI: 10.1080.09500693.2016.1262567.

[8] S. Karen and S. Gregg, "Alternative Assessment – Can Real-world Skills be Tested? Policy Briefs," National Library of Australia, 1993.

[9] S. S. Alfuhaigi, "*School Environment and Creativity Development: A Review of Literature*," Journal of Educational and Instructional Studies, Vol. 5, Iss. 2, May 2015, pp. 33–37.

[10] B. Irwin and S. Hepplestone, "*Examining Increased Flexibility in Assessment Formats*," Assessment & Evaluation in Higher Education, 2012, pp. 773–785.

[11] A. Craft, "An Analysis of Research and Literature on Creativity in Education: Report prepared for the Qualifications and Curriculum Authority," Qualifications and Curriculum Authority, Mar. 2001.

[12] P. Kampylis and E. Berki, "*Nurturing Creative Thinking*," International Academy of Education, UNESCO, 2014.

[13] JISC, "*Transforming Assessment and Feedback with Technology*," Online; Available: https://www.jisc.ac.uk/full-guide/transforming-assessment-and-feedback.

[14] Montessori Northwest, "*What is Montessori Education*?" n.d. Online; Available: https://montessori-nw.org/what-is-montessori-education/.

[15] An Everyday Story, "*What is the Reggio Emilia Approach*?" Online; Available: http://www.aneverydaystory.com/beginners-guide-to-reggio-emilia/main-principles/.

[16] N. Jackson, "Assessing Students' Creativity: Synthesis of Higher Education Teacher Views," The Higher Education Academy, Jun. 2005.

[17] Warwick University, "*Marking Creative Writing*" Online; Available: https://warwick.ac.uk/fac/arts/english/currentstudents/undergraduate/modules/fulllist/second/en232/marking_creative_writing/.

[18] S. M. Brookhart, "*Assessing Creativity*," Educational Leadership, Vol. 70, No. 5, Feb. 2013; Available: http://www.ascd.org/publications/educational-leadership/feb13/vol70/num05/Assessing-Creativity.aspx.

[19] G. Hill and S. J. Turner, "*Electronic Online Marking of Software Assignments*," Progress in IS: Software Engineering Education fro a Global E-service Economy, 2014, pp. 41–48.

[20] A. Venables and L. Haywood, "*Programming Students need Instant Feedback!*," in Proceedings of 5th Australasian Computing Education Conference, 2001.

[21] Learning Theories, "*Transformative Learning Theory (Mezirow)*," Online; Available: https://www.learning-theories.com/transformative-learning-theory-mezirow.html.

[22] J. Bosch and P. Molin, "*Software Architecture Design: Evaluation and Transformation*," Proceedings of IEEE Conf. and Workshop on Engineering of Computer-based Systems, Mar. 1999.

[23] BBC, "*International Baccalaureate*," Online; Available: http://www.bbc.co.uk/schools/parents/international_baccalaureate/.

[24] Vanderbilt, "*Motivating Students*," Online; Available: https://cft.vanderbilt.edu/guides-sub-pages/motivating-students/.

[25] L. Gueldenzoph Snyder and M. J. Synder, "*Teaching Critical Thinking and Problem Solving Skills*," The Delta Pi Epsilon Journal, Vol. L, No. 2, 2008, pp. 90–101.

[26] S. P. Kearney and T. Perkins, "Engaging Students through Assessment: The Success and Limitations of the ASPAL (Authentic Self and Peer Assessment for Learning) Model," ResearchOnline@ND, 2014.

[27] DePaul University, "*Authentic Assessment "Assessing by Doing*"," Online; Available: https://offices.depaul.edu/teaching-learning-and-assessment/assessment/assessing-learning/Documents/AuthenticAssessmentInformationSheet.pdf.

What Influences Students' Understanding of Scalability Issues in Parallel Computing?

Juan Chen
College of Computer
National University of Defense Technology
Changsha, Hunan Province, China
juanchen@nudt.edu.cn

Brett A. Becker
School of Computer Science
University College Dublin
Dublin, Ireland
brett.becker@ucd.ie

Youwen Ouyang
Department of Computer Science and Information Systems
California State University, San Marcos
San Marcos, CA, US
ouyang@csusm.edu

Li Shen
College of Computer
National University of Defense Technology
Changsha, Hunan Province, China
lishen@nudt.edu.cn

ABSTRACT

Graduates with high performance computing (HPC) skills are more in demand than ever before, most recently fueled by the rise of artificial intelligence and big data technologies. However, students often find it challenging to grasp key HPC issues such as parallel scalability. The increased demand for processing large-scale scientific computing data makes more essential the importance of mastering parallelism, with scalability often being a crucial factor. This is even more challenging when non-computing majors require HPC skills. This paper presents the design of a parallel computing course offered to atmospheric science majors. It discusses how the design addressed challenges presented by non-computer science majors who lack a background in fundamental computer architecture, systems, and algorithms. The content of the course focuses on the concepts and methods of parallelization, testing, and the analysis of scalability. Considering all students have to confront many (non-HPC) scalability issues in the real world, and there may be similarities between real-world scalability and parallel computing scalability, the course design explores this similarity in an effort to improve students' understanding of scalability issues in parallel computing. The authors present a set of assignments and projects that leverage the Tianhe-2A supercomputer, ranked #6 in the TOP500 list of supercomputers, for testing. We present pre- and post-questionnaires to explore the effectiveness of the class design and find an 11.7% improvement in correct answers and a decrease of 36.8% in obvious, but wrong, answers. The authors also find that students are in favor of this approach.

KEYWORDS

Atmospheric science majors, Computing non-majors, Scalability, Undergraduate education

https://doi.org/10.22369/issn.2153-4136/12/2/12

1 INTRODUCTION

Many modern scientific developments depend on large scale data processing and the exploitation of parallelism on supercomputer systems. Examples include computational simulations for scientific and engineering applications in atmosphere, earth, and environmental realms [1], in addition to commercial applications such as data mining and transaction processing.

Integrating parallel computing earlier into the undergraduate curriculum has been under exploration since the 1990s. In 1994, Donald Johnson et al. [2] proposed teaching parallel computing to freshmen by integrating parallel computation into a data structures course. The Curriculum Development and Education Resources (CDER) center for parallel computing education proposed a detailed curriculum and promoted progress in parallel computing undergraduate courses [3]. Nonetheless, introducing parallel computing into the early years of a bachelors curriculum remains challenging.

Reflecting the growing importance of parallel computing in undergraduate curricula, CS2013 (the ACM/IEEE Joint Computer Science Curricula) stated, "Previous curricular volumes had parallelism topics distributed across disparate KAs [knowledge areas] as electives. Given the vastly increased importance of parallel and distributed computing, it seemed crucial to identify essential concepts in this area and to promote those topics to the core" [4, p 29]. CS2013 introduced a new KA in parallel and distributed computing which explicitly names scalability [5] as a core-tier2 topic.

Understanding scalability issues is key for learning parallel computing. This paper introduces the design of a compulsory parallel computing course at the College of Meteorologic Oceanography at National University of Defense Technology in China. Atmospheric scientists need parallel computing to solve design issues for the parallelization of optimal interpolation algorithms and atmospheric data analysis [6]. This course is intended to provide a broad overview of the topics in parallel computing, as a lead-in for more advanced classes that follow it. These non-computing majors need to leverage HPC to make optimal use of their applications and to solve problems on different scales. Proper solutions resulting in satisfactory speedup are necessary but difficult to design. To start, these applications need to consider the physical architecture to and fully exploit hardware parallelism. The increase in large-scale

scientific computing data aggravates the difficulty of exploiting parallelism.

1.1 Challenges of Understanding Scalability

Scalability is the mechanism by which a parallel system's speedup changes with the number of available processors. Amdahl's law dictates the achievable speedup and efficiency — specifically what happens to efficiency when both the number of processors and the problem size increase. The scalability of a parallel algorithm on a parallel architecture is a measure of the algorithm's ability to effectively utilize an increasing number of processors. Scalability analysis is helpful in selecting the best algorithm/architecture combination for a given problem under different constraints on the growth of the problem size and the number of processors [7, 8].

Scalability is divided into hardware scalability and software scalability, which refers to the ability of system to deliver greater computational power when the amount of resources is increased. Hardware scalability refers to the capacity of the whole system, which theoretically can be proportionally increased by adding more hardware. Software scalability refers to parallelization efficiency - the ratio between the actual speedup and the ideal speedup over a given number of processors [9].

The scalability of a system can take many forms, including speed, efficiency, size, applications, generation, and heterogeneity [10, p 63]. In terms of speed, a scalable system is capable of increasing its speed in proportion to the increase in the number of processors. In terms of efficiency, a scalable parallel system means its efficiency can be kept fixed as the number of processors is increased, provided that the problem size is also increased. Scalability testing can be performed at the hardware or software levels. Parameters used for scalability testing differ from one application to another. Different forms of scalability were also mentioned in [10, p 66], "In his vision on the scalability of parallel systems, Gordon Bell indicated that in order for a parallel system to survive, it has to satisfy five requirements. There are size scalability, generation scalability, space scalability, compatibility, and competitiveness." Three of these survivability requirements are the forms of scalability. Here size scalability measures the maximum number of processors a system can accommodate. Generation scalability refers to the ability of a system to scale up by using next-generation components.

HPC application scalability is inherently complicated as the performance of modern HPC systems approach exascale. Exascale computing refers to computing systems capable of at least a billion billion calculations per second. It is becomming even more complex for HPC applications to fully exploit hardware parallelism, due to many factors. In addition, many applications have poor scalability regardless of the underlying hardware. See [11] for more details.

Scalability modeling and evaluation for real problems are often abstract. Programs are often designed and tested for smaller data sets on fewer processors, but the real problems are much larger and need more hardware, in recent times scaling up to millions of cores. Performance and correctness of programs based on scaled-down systems is difficult to establish [12, p 208], but it remains a cost efficient and practical means of testing. Based on such tests, performance extrapolation is not intuitive.

1.2 Research Goals

This work has three research goals:

- RG1 Explore the effects of understanding or misunderstanding parallel computing scalability on students' performance.
- RG2 Explore the relationship between real-world scalability and parallel computing scalability from the perspective of understanding and learning.
- RG3 Explore students' questions valuable to the understanding of parallel computing scalability.

2 BACKGROUND

2.1 Parallel Computing Course and Scalability

Many modern instructors agree that parallel computing topics should be covered in first- or second-year undergraduate classes [2, 13–15]. Additionally, in Section 1, the authors discussed the fact that parallelism is also a growing trend to which CS2013 has responded. The primary reasoning is that a solid understanding of parallel computing concepts will tremendously improve students' ability to write software that is able to effectively utilize the underlying parallel hardware architecture. For example, Yousun Ko et al. [14] found if parallel computing concepts were introduced as a senior-level undergraduate or graduate elective, students had difficulty transitioning from sequential to parallel thinking. Lori Pollock et al. [16] also thought parallel programming required a very different thought process from traditional sequential programming, as the programmer must think differently, such as performing tasks in parallel, organizing information communication, and balancing workload between parallel processes. Making such a switch from sequential thinking to parallel thinking was a big step for many students. CS2013 recommends parallel computing could be their freshmen or sophomores courses.

Some challenges in parallel computing courses are closely related to scalability. For example, Yousun Ko et al. [14] presented a challenge problem for understanding parallelism. They chose an analogy from cooking to introduce task, data, and pipeline parallelism. They also used the concrete example to illustrate task parallelism and data parallelism. Another challenge they presented is about parallel program performance evaluation. They observed the inevitable question was, "Why is my parallel program slower than the sequential version?" They answered this question by introducing the definition of speedup, scalability, and efficiency, followed by Amdahl's law and performance bottleneck analysis. Besides the above two challenges, Yousun Ko et al. [14] presented three other course modules and challenge problems. They used the decomposition approach of knowledge to structure the course as five course modules, among which each module teaches one fundamental concept of parallel programming. All parallel programming concepts were introduced with the help of worked-out programming examples.

Besides the challenges of switching from sequential thinking to parallel thinking mentioned above, Pollock et al. [16] presented the practical challenges for inexperienced programmers: i) lack of stable and useful debugging tools; ii) the need to analyze why their program is not performing well in parallel and how to improve its performance. They used cooperative learning to meet the practical

challenges, as well as to provide a more real-world context to the course. In the experience of teaching HPC [17], H. Neeman et al. used analogies to explain some concepts to capture the fundamental underlying principles without going deep into technical details.

2.2 Real-World Scalability and Cognitive Ability

There are many scalability issues in real life. For instance, compound interest concerns how investments scale with time, and population growth concerns how the population of reproducing organisms scales with time. Fittingly, most HPC and parallel computing concepts also come down to time — after all, that is why these domains exist — to do more in less time. However, it is well known that many people have trouble truly comprehending the growth of a fund due to compound interest, or the growth of populations with time. It seems scalability is linked to cognition.

In 2005, Frederick introduced the cognitive reflection test (CRT), which researchers have cited nearly 3,500 times. The CRT is a simple three-item measure of one type of cognitive ability. Specifically, Frederick found that CRT scores were predictive of the kinds of choices that prominently feature in tests of decision-making theories. The CRT questions are [18, p 27]:

Q1 A bat and a ball cost $1.10 in total. The bat costs $1.00 more than the ball. How much does the ball cost?

Q2 If it takes 5 machines 5 minutes to make 5 widgets, how long would it take 100 machines to make 100 widgets?

Q3 In a lake, there is a patch of lily pads. Every day, the patch doubles in size. If it takes 48 days for the patch to cover the entire lake, how long would it take for the patch to cover half of the lake?

Interestingly, questions Q2 and Q3 deal with scalability, and Q2 also concerns explicit parallelism.

The CRT questions are crafted very carefully, and for a reason. Each question has an incorrect "intuitive" answer. Frederick provides substantial evidence that these incorrect yet intuitive answers are indeed intuitive. Two pieces of evidence are 1) the incorrect intuitive answers, such as 24 days (half of 48 days) for Q3[1], dominate in trials with large populations; and 2) respondents answer with the correct response much more often with analogous problems that invite more computation (i.e. don't have obvious intuitive answers). For example, respondents miss Q1 much more often than they miss this analogous problem that essentially forces calculation due to the lack of an intuitive answer:

- A banana and a bagel cost 37 cents. The banana costs 13 cents more than the bagel. How much does the bagel cost?

We will come back to the idea of questions on real-life scalability concepts and "intuitive answers" in Section 4.2.

3 STUDY DESIGN

The study was carried out in College of Meteorologic Oceanography in Spring 2019. Fourteen sophomores enrolled in the "Parallel Computing" course, and all students participated in the study. All of them had no prior experience with systematic computer architectures, operating systems, and algorithms. The prerequisite of the course is C programming.

3.1 Course Design

Figure 1 shows the structure of the course, including 12 lecture classes, 6 laboratory classes, assignments, projects, and pre-/post-questionnaires. The length of each class period is 90 minutes with a 10-minute middle break. Formative assignments were released once or twice each week and required to be finished before Sunday night 23:00 pm. Projects were released on Thursday night laboratory class between Weeks 4 and 7. Each project lasted one week until the next Wednesday night.

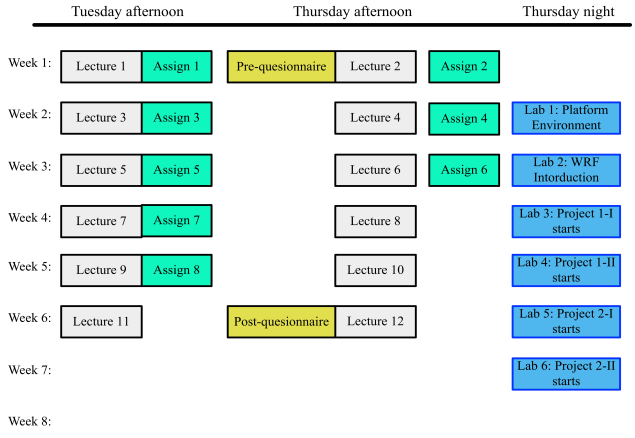


Figure 1: Weekly and daily structure of the course.

The course learning objectives for the lecture part are shown in Table 1. One of the authors is the lead instructor of this course. In order to teach scalability, the authors split the effort from the following eight aspects, which are abbreviated D1–D8 in Table 4.

Table 1: Parallel computing course content topics.

Topic	Content	Lecture
1	Overview of parallel computing: Flynn's taxonomy, parallel hardware and software, interconnection network, etc.	1–2
2	Basic principles of parallel computing: task partition, parallel task scheduling, principles of parallel algorithm design, performance metrics, concepts of speedup/efficiency/scalable applications, etc.	3–4
3	Distributed-memory programming with MPI, collective communication, performance evaluation of MPI programs, scalability, etc.	5–7
4	Shared-memory programming with OpenMP, data dependences, loop scheduling, cache coherence, etc.	8–9
5	Applications: Jacobi methods and computation-communication overlap, numerical weather forecast model WRF, numerical climate forecast model CAM, etc.	10–12

1) Decompose the scalability topic into some detailed notions. According to core topics of parallel computing provided by NSF/IEEE-TCPP Curriculum Initiative [3], the authors decompose the scalability topic into three-type 19 notions as Table 2 shows. The three

[1]The correct answer is 47 days. If the patch doubles in size every day, one day before the final day, 1/2 of the pond must be covered.

types are architecture, programming, and algorithms. Each notion has its learning outcome. The scalability topic is too abstract to teach. However, this decomposition can help make clear what scalability stands for in the parallel computing world, such as what detailed contents of scalability people should teach students and what learning outcome students should have.

2) Design the assignments. Design the fundamental assignments as well as literature reading tasks. The course includes eight formative assignments and a literature reading task throughout an 8-week period. Each of these was graded based on functionality and documentation. All eight formative assignments are about the answer to some basic questions, followed by fundamental parallel programming exercises, which are suitable to all different majors (See Assignments 1–8 in Table 3 for more details). The literature reading task is special for atmospheric science majors in order to deepen their understanding of weather research, the forecasting model, and its parallel solution method. The students need to read at least one paper from the following three papers, which are titled "Development of a Next-generation Regional Weather Research and Forecast Model", "Precipitation Simulations Using Weather Research and Forecasting (WRF) as a Nested Regional Climate Model," and "Sensitivity of WRF Forecasts for South Florida to Initial Conditions."

3) Design the projects. Two application projects are special for Atmospheric Science majors in our class. These two projects are both about numerical weather forecast simulation on a WRF model. WRF model simulation is fundamental to most atmospheric science majors in their professional study and research. WRF is short for the Weather Research and Forecasting model, which is a mesoscale numerical weather prediction system designed for both atmospheric research and operational forecasting applications [19]. The WRF model features two dynamical cores, a data assimilation system, and software architecture supporting parallel computation and system extensibility. The model serves a wide range of meteorological applications across scales from tens of meters to thousands of kilometers. Before the 2019 spring semester, students did not use WRF until the senior year. It was a challenge for sophomores to finish the two projects. In order to reduce the difficulty of studies, the authors divided Project 1 into two phases: Project 1-I and Project 1-II. Project 1-II is a moderately incremental release based on Project 1-I. The same is true for Project 2. See Table 3 for more details of the two projects. Before releasing the two projects, the authors have two laboratory classes to introduce the Tianhe-2A system environment and basic usage, followed by the WRF model background (Lab 1 and Lab 2 in Figure 1). The Tianhe-2A supercomputer is ranked #6 in the TOP500 list of supercomputers[2].

4) Correlate scalability topic & notions to assignments and projects. The authors connect all the scalability notions to eight assignments and four projects in terms of the contents of assignments and projects as well as each learning outcome shown in Table 2. This correlation is helpful to grade the understanding of scalability. The scalability grade for each assignment and project can be given based on students' learning outcomes. The scalability grade of each student is counted by the average scalability scores of all assignments and projects.

Table 2: Decomposition of scalability topics and correlation of scalability notions, learning outcomes, and assignments & projects. The rightmost column 'Learning Outcome' is taken from Tables 1-3 in Reference [3]

Topics/Notions	Assign	Project	Learning Outcome
Architecture			
SMP → Buses	A1	-	Limited bandwidth and latency, scalability issues
NUMA → Directory-based CC-NUMA	-	-	Be aware that bus-based sharing does not scale, and directories offer an alternative
Message passing (no shared memory)	-	-	Shared memory architecture breaks down when scaled due to physical limitations (latency, bandwidth) and results in message passing architectures
Latency	-	P1, P2	Know the concept, implications for scaling, impact on work/communication ratio to achieve speedup
Bandwidth	-	P1, P2	Know the concept, how it limits sharing, and considerations of data movement cost
Cache organization	-	-	Know the cache hierarchies, shared caches (as opposed to private caches) result in coherency and performance issues for software
Programming			
Shared memory	A8	-	Be able to write correct thread-based programs (protecting shared data) and understand how to obtain speed up
Synchronization	-	-	Be able to write shared memory programs with critical regions, producer-consumer communication, and get speedup; know the notions of mechanisms for concurrency
Performance metrics	A2, A6, A7, A8	P1, P2	Know the basic definitions of performance metrics (speedup, efficiency, work, cost), Amdahl's law; know the notions of scalability
Speedup	A2, A6, A7, A8	P1, P2	Understand how to compute speedup, and what it means
Efficiency	A2, A6, A7	P1, P2	Understand how to compute efficiency, and why it matters
Amdahl's law	A2, A6, A7	P1, P2	Know that speedup is limited by the sequential portion of a parallel program, if problem size is kept fixed
Gustafson's law	A2	-	Understand the idea of weak scaling, where problem size increases as the number of processes/threads increases
Isoefficiency	-	P1, P2	Understand the idea of how quickly to increase problem size with number of processes/threads to keep efficiency the same
Algorithm			
Speedup	A3, A6, A7	-	Recognize the use of parallelism either to solve a given problem instance faster or to solve larger instance in the same time (strong and weak scaling)
Scalability in algorithms and architectures	A6, A7	P1, P2	Comprehend via several examples that having access more processors does not guarantee faster execution — the notion of inherent sequentiality
Model-based notions	A8	P1, P2	Recognize that architectural features can influence amenability to parallel cost reduction and the amount of reduction achievable
Matrix computations	A5, A6, A7	-	Understand the mapping and load balancing problems on various platforms for significant concrete instances of computational challenges that are discussed at a higher level elsewhere
Matrix product	A6, A7	-	Observe a sample "real" parallel algorithm

[2]www.top500.org. TOP500 List, November 2020

Table 3: Descriptions of eight assignments and two projects.

Assignments & Projects	Tasks
Assignments 1–3	Search for information: TOP500, parallelism of pipeline, parallelism of vector operations, speedup formulas, Foster's methodology.
Assignments 4–7	MPI programs: trapezoidal rule MPI parallelization; matrix-vector multiplication MPI parallelization with row partitioning; matrix-vector multiplication MPI parallelization with column partitioning.
Assignment 8	OpenMP programs: Odd-even transposition sort OpenMP parallelization.
Project 1-I	Install WRF software and library, and run an ideal case with different amount of computing nodes.
Project 1-II	Install WRF software and library, and run two more ideal cases with different amount of computing nodes.
Project 2-I	Compile real model em_real and run a real model data. WRF preprocessing system WPS need to be installed before running. Download actual observational data from the official website and do the test with different parallel scales.
Project 2-II	Reset the model domain by modifying the model grid parameters (resolution, range, and grid location) in model input file 'namelist.input', recompile the real model em_real and run real model data. Repeat the steps of Project 2-I.

5) Design incentive mechanism to encourage questioning in class. Students were encouraged to question in class by adding the number of questions into final grades. On average, more than ten questions per class were proposed, which greatly increased participation in class activities and inspired students' learning enthusiasm. After class, TAs collected all the questions and counted the grades, which were published to students every other week.

6) Do experimental instruction. Four teaching assistants (TAs) participated in the class activities, especially being involved in experimental instruction in laboratory classes. Before projects began, one TA presented a talk to introduce the background, demands, and expected outcomes. TAs finished all the experimental steps and prepared a detailed experiment instruction manual before students started projects. In laboratory classes, students were divided into four groups and each TA gave individual tutoring to one group of students having difficulties. Students were required to write scientific reports describing their experiments, including objective, platform & environment, steps, results & analyses, questions, and experiences. Students struggled with writing these experimental reports and the analysis of parallel program performance and scalability. This appeared to be busy work to the non-CS majors. For some common questions, the teacher asked one student to present their initial results firstly and then organize a discussion to analyze reasons. Finally, students reached an agreement and designed one more experiment to validate their assumptions. A discussion usually lasted about 15 minutes. The authors twice had such discussions.

7) Design of the evaluation and grading mechanism. This course had no final examination. Student performance was scored by assignment scores, in-class questioning, attendance, project scores, and literature reading scores. Section 4.1 gives the detailed evaluation method.

8) Design pre-/post-questionnaires and feedback questionnaire. Before the parallel computing course began, a pre-questionnaire was administered to test the understanding of students' real-world scalability. These six questions all draw from real life examples of scalability — See Q1–Q3 in Section 2.2 for examples. Each question has three categories of answers: an incorrect yet intuitive answer, an "obviously" wrong answer, and a correct answer. The authors did not expect (or want) the students to simply calculate the right answer — any of these questions can be easily calculated given enough time. Instead, we were trying to test their "intuition" of what answer "seems" correct. In other words, we wanted to measure students' real-world intuitions that they have gained through experience. Students were told to try to capture their "gut feeling" — their intuition — when answering the questionnaire. Accordingly, students were given three minutes (30 seconds per question). At the end of the course, a post-questionnaire, the same as the pre-questionnaire, was administered to students. We then analyzed these scores using a paired statistical significance test called the *Scalability Understanding Paired Test* and correlated the questionnaire results with final course grades. Students were not formally assessed on their questionnaire responses/scores — in other words, the questionnaire scores did not factor into final grades.

At the end of the course, students were encouraged to complete a feedback questionnaire inquiring about their understanding of scalability issues, including a self-evaluation of the learning outcomes of the 19 scalability notions, misunderstandings and correctness experiences with scalability issues, their biggest impression of the course, their confidence toward parallel computing, and other experiences with course activities.

3.2 Correlation of Course Design and Research Goals

What the authors did for teaching scalability in Section 3.1 is based on our three research goals. There are some correlation between them, as Table 4 shows, by ticking √. In Table 4, D1–D8 stand for eight aspects of our course design for teaching scalability in Section 3.1.

Table 4: Correlation of course design and research goals.

RG	D1	D2	D3	D4	D5	D6	D7	D8
RG1	√	√	√	√			√	√
RG2	√						√	√
RG3		√	√		√	√		√

4 RESULTS

4.1 RG1: Scalability Understanding and Performance

We measure student performance using their assignment scores, in-class questioning, attendance, project scores, and literature reading scores. The scores for eight assignments, in-class questioning scores, and attendance comprise 30% of the final course mark. The scores for Project 1-I, Project 1-II, Project 2-I, and Project 2-II comprise 50% of the final mark, and literature reading scores comprise 20% of the final mark. The final course grades for all students are shown by the blue solid line in Figure 2.

According to Table 2, the authors measured student scalability grades for each assignment and each project. This scalability grade for each assignment and project is divided into three levels of achievement: sophisticated (90 points), competent (70 points) and

not yet competent (50 points). Then, we calculated the average scalability grade for all assignments and projects, which constitutes the scalability grade of each student for learning parallel computing (See the red dashed line in Figure 2). A chi-squared test of goodness-of-fit was performed to determine whether the scalability grade of a student for learning parallel computing is linearly correlated to the final course grade. The scalability grade of a student and the final course grade were positively correlated, $r(14) = 0.66, p = 0.010$. This is evidence that one can reasonably expect a better course performance from students with a better understanding of parallel computing scalability.

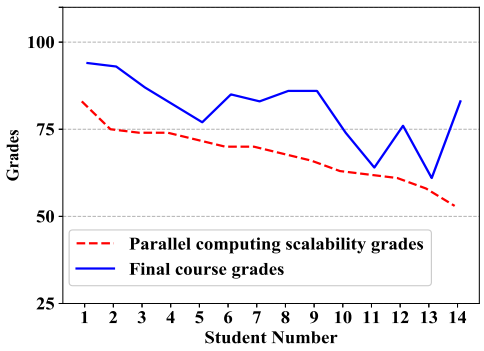


Figure 2: The impact of understanding or misunderstanding parallel computing scalability on students' performance.

At the end of the course, a multiple-choice feedback questionnaire was sent to all 14 students, and all of 14 students responded. The feedback questionnaire was designed to examine students' experience with learning parallel computing scalability. Questions in the feedback questionnaire were three multiple-choice, two single-choice, and three subjective questions. As can be seen, 78% students thought parallel computing scalability studies were very beneficial for understanding key concepts in parallel computing and improving their parallel programming ability. The remaining students thought such help was okay. Students self-evaluate their understanding of parallel computing scalability with five stars. The proportions of five, four, three, and two stars in students' self-evaluation are 21%, 28%, 35%, and 14%, respectively.

In the three multiple-choice quesitons involving architecture, programming, and algorithms, students picked out some items (rows) from Table 2 for which they thought they have achieved the learning outcomes. According to students' self-evaluation, the top five items the students achieved are *Programming*-Speedup (14/14), Efficiency (13/14), Amdahl's law (12/14), Gustafson's law (11/14) and *Algorithm*-Speedup (11/14). The worst five items are *Programming*-Isoefficiency (1/14), *Architecture*-NUMA→Directory-based CC-NUMA (3/14), Cache organization (4/14), *Programming*-Shared memory (4/14), and Synchronization (4/14). This is evidence that our decomposition of scalability topics is really beneficial to teach students to understand scalability issues. It also shows that more assignments and projects can help students better understand scalability notions. In both of the two projects as well as several fundamental programming assignments, students need to use speedup, efficiency, and Amdahl's law concepts to calculate and evaluate the results. Repetitive calculations correct some misunderstandings and deepen their understanding to scalability. On the contrary, for those knowledge notions lacking exercises, it is hard to expect students to have a good learning performance. For example, the NUMA concept, cache organization, shared memory, and synchronization are not directly relevant to assignments or projects. The learning performance of understanding these notions is worse than that of understanding speedup, efficiency, Amdahl's law, and Gustafson's law.

Students needed to give at least one experience of misunderstanding parallel computing scalability. We list all the feedback as follows, merging some same or similar feedback.

- Students A1, A2 thought that the running time was inversely proportional to the number of processes. But in the actual cases, they observed that running time sometimes was constrained by bandwidth.
- Students B1, B2 could not understand why the increasing computing power or number of processes sometimes could not bring about speedup improvement.
- Students C1, C2, C3, C4 thought using the more processor cores must result in the faster speedup, the higher efficiency, and the stronger scalability, but by experimental results they found more processor cores were not sure to bring about higher performance, and sometimes parallel time would be reduced only when the number of processor cores reaches a certain value.
- Student D thought scalability was only related to the application problem itself, but they later found scalability was also closely related to the parallel algorithm.
- Student E thought matrix-vector parallelism in row partition had the same effect with that in column partition, but by communication analysis and experimental results, they found different matrix partitions would bring about different amounts of collective communication and also big performance differences.
- Student F thought the scalability only represented the running time of a program and the shorter running time meant better scalability.
- Students G1, G2, G3 thought there was no relationship between speed, the amount of input data, and the number of processes/threads, which prevented them from understanding speed and scalability.

They were asked to explain if learning more knowledge of scalability and overcoming the misunderstandings of scalability were useful to their performance improvement of learning parallel computing or not. All the answers are YES.

4.2 RG2: Real-World vs. Parallel Computing Scalability

We released a pre-questionnaire and a post-questionnaire to test students' real-world scalability cognitive ability. The correctness ratio increased by 11.7% from the pre-questionnaire to the post-questionnaire. The number of obvious but wrong answers was

reduced by 36.8%. This proves that students' understanding of scalability improved. The Pearson correlation coefficient between pre-questionnaire grades and post-questionnaire grades was 0.73.

We compare two questionnaire results and parallel computing scalability grades in Figure 3. The Pearson correlation coefficient between pre-questionnaire grades and parallel computing scalability grades is 0.0079. This proves that real-world scalability grades and parallel computing scalability grades are correlated with very low strength. The Pearson correlation coefficient between post-questionnaire grades and parallel computing scalability grades is increased to -0.34. It shows that the authors could expect the correlation strength between real-world scalability and parallel computing scalability can be increased by learning parallel computing.

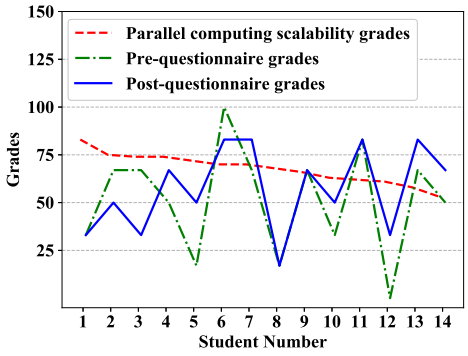


Figure 3: The relationship between real-world scalability and parallel computing scalability.

4.3 RG3: Students' Questions Valuable to the Understanding to Scalability

Students' questions mainly came from in-class questions, laboratory class questions, and project questions. Each class lecture averaged ten questions. These questions were commonly about topics such as architecture concepts and programming problems. Major questions related to scalability issues were as follows: How to calculate the efficiency? What is weak scaling? What is strong scaling? How to identify the scalability of one program? What is the difference between Amdahl's law and Gustafson's law? How to understand the four steps of Foster's parallelization method and how to apply the Foster method to solve a real application? How to calculate the number of collective communications for different partitioning approaches?

Lab class questions are mainly about programming problems, algorithm implementation, and actual operations on the Tianhe-2A supercomputer, such as how to assign jobs to the expected amount of computer nodes? How to map processes to computer nodes and processor cores? Below are some choice questions that show the value of understanding parallel computing scalability.

Q1 The impact of different matrix partitioning methods on parallel performance in a matrix-vector multiplication program.
Question: For a matrix-vector multiplication program, which is better, partitioning the matrix by column or by row? Why?
Teaching Solution: Analyze two matrix partitioning methods (by row and by column) and compare their numbers of collective communications. Run programs and compare execution time under two partitioning methods. Students will find partitioning the matrix by column would produce more collective communications compared to partitioning the matrix by row and then result in parallel performance reduction.

Q2 The impact of memory bandwidth on speedup and efficiency.
Question: There was a case in Project 1-II: a program with 8 processes achieved higher speedup than with 4 processes. But the program running with 16 processes did not achieve expected speedup like with 8 processes. Why? Then the speedup with 32 processes were again higher than that with 16 processes. Why?
Teaching Solution: For a shared-memory architecture on one compute node of the Tianhe-2A supercomputer we count the number of physical cores used for each row and consider the impact of limited memory bandwidth on speedup and efficiency. We doubt this speedup jump is related to the number of assigned cores on one compute node. It is possible to do more experiments to verify our assumption, where different numbers of compute nodes and processor cores are assigned.

Q3 The impact of architecture and node allocation strategy on parallel execution time.
Question: Students found an 8-process program running on one compute node was slower than running on two compute nodes. Why does the fixed amount of processes have a different execution time? How do different node allocations influence parallel execution time?
Teaching Solution: Illustrate the concept of memory bandwidth and list the facts that influence memory bandwidth. The limited memory bandwidth on a shared-memory architecture sometimes has an influence on parallel execution time. Analyze why different assignments of processes to compute nodes will bring about different actual memory bandwidth on one node. Suggest an experiment to verify the assumption: parallel execution time of a memory-intensive program will be greatly influenced by actual memory bandwidth.

5 CONCLUSIONS

This study focused on what influenced students' understanding of parallel computing scalability issues. The authors found understanding or misunderstanding parallel computing scalability has a correlation with students' performance. We explored the relationship between real-world scalability and parallel computing scalability, and we found real-world scalability and parallel computing scalability were correlated with low strength. There is no evidence to prove real-world concepts and experiences will greatly influence the learning of parallel computing concepts. However, the authors could expect the correlation strength between real-world scalability and parallel computing scalability can be increased by learning parallel computing. We need more research and analyses about the two types of scalability in the future. Finally, the authors showed some examples of student questions that are valuable to the understanding of parallel computing scalability. According to pre- and post-questionnaires, the effectiveness of our parallel computing course resulted in an 11.7% improvement in correct answers and a

decrease of 36.8% in obvious but wrong answers. Most students are in favor of the approach used.

ACKNOWLEDGMENTS

The authors would like to thank Dr. John Impagliazzo (Hofstra University) for providing insight to improve the paper and the anonymous referees for their valuable comments and helpful suggestions. This work is supported by the 2017 Hunan Province Degree and Graduate Education Teaching Reform Research Foundation of China under Grant No. JG2017B004, the 2019 Hunan Province Higher Education Teaching Reform Research Foundation of China (titled with Teaching Practice of Training High-Performance Computing Talents Relying on High-level Scientific Research), and the 2019 Hunan Province Postgraduate Outstanding Professional Case Foundation of China (titled with High-Performance Computing Series Case Library).

REFERENCES

[1] Blaise Barney. Introduction to parallel computing. https://computing.llnl.gov/tutorials/parallel_comp/, 2019.

[2] Donald Johnson, David Kotz, and Fillia Makedon. Teaching parallel computing to freshmen. 1994. *Conference on Parallel Computing for Undergraduates*, 1994.

[3] Nsf/ieee-tcpp curriculum working group. nsf/ieee-tcpp curriculum initiative on parallel and distributed computing - core topics for undergraduates. technical report, ieee-tcpp. http://tcpp.cs.gsu.edu/curriculum/?q=system/files/NSF-TCPP-curriculum-version1.pdf, 2012.

[4] Joint task force on computing curricula, association for computing machinery (acm) and ieee computer society. computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science. Technical report, New York, NY, USA, 2013. 999133.

[5] André B. Bondi. Characteristics of scalability and their impact on performance. In *Proceedings of the 2nd International Workshop on Software and Performance*, WOSP '00, pages 195–203, New York, NY, USA, 2000. ACM.

[6] G. von Laszewski. An interactive parallel programming environment applied in atmospheric science. In *Proceedings of the 6th Workshop on the Use of Parallel Processors in Meteorology, G.-R. Hoffman and N. Kreitz, Eds., European Centre for Medium Weather Forecast. Reading, UK: World Scientific*, pages 311–325, 1996.

[7] V.P. Kumar and A. Gupta. Analyzing scalability of parallel algorithms and architectures. *Journal of Parallel and Distributed Computing*, 22(3):379–391, Sep 1994.

[8] J. Y. Shi, M. Taifi, A. Pradeep, A. Khreishah, and V. Antony. Program scalability analysis for hpc cloud: Applying amdahl's law to nas benchmarks. In *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*, pages 1215–1225, Nov 2012.

[9] Xin Li. Scalability: strong and weak scaling. https://www.kth.se/blogs/pdc/2018/11/scalability-strong-and-weak-scaling/, 2018.

[10] Hesham El-Rewini and Mostafa Abd-El-Barr. Advanced computer architecture and parallel processing. John Wiley & Sons. ISBN 978-0-471-47839-3., 2005.

[11] JJ Dongarra. *On the Future of High Performance Computing: How to Think for Peta and Exascale Computing*. Hong Kong University of Science and Technology Hong Kong, 2012.

[12] Ananth Grama, Anshul Gupta, George Karypis, and Vipin Kumar. *Introduction to Parallel Computing (2nd Edition)*. Pearson Education Limited, 2003.

[13] E. Saule. Experiences on teaching parallel and distributed computing for undergraduates. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 361–368, May 2018.

[14] Yousun Ko, Bernd Burgstaller, and Bernhard Scholz. Parallel from the beginning: The case for multicore programming in the computer science undergraduate curriculum. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 415–420, New York, NY, USA, 2013. ACM.

[15] D. J. John. Integration of parallel computation into introductory computer science. In *Proceedings of the Twenty-third SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '92, pages 281–285, New York, NY, USA, 1992. ACM.

[16] Lori Pollock and Mike Jochen. Making parallel programming accessible to inexperienced programmers through cooperative learning. In *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '01, pages 224–228, New York, NY, USA, 2001. ACM.

[17] H. Neeman, J. Mullen, L. Lee, and G. K. Newman. Supercomputing in plain english: Teaching high performance computing to inexperienced programmers. In *Proceedings of the 3rd LCI International Conference on Linux Clusters: The HPC Revolution*, 2002.

[18] Shane Frederick. Cognitive reflection and decision making. *Journal of Economic perspectives*, 19(4):25–42, 2005.

[19] Weather research and forecasting model. national center for atmospheric. https://ncar.ucar.edu/what-we-offer/models/weather-research-and-forecasting-model-wrf, 2020.

Promoting HPC Best Practices with the POP Methodology

Fouzhan Hosseini
The Numerical Algorithms Group
Manchester, UK
fouzhan.hosseini@nag.co.uk

Craig Lucas
The Numerical Algorithms Group
Manchester, UK
craig.lucas@nag.co.uk

ABSTRACT

The performance of HPC applications depends on a wide range of factors, including algorithms, programming models, library and language implementations, and hardware. To make the problem even more complicated, many applications inherit different layers of legacy code, written and optimized for a different era of computing technologies. Due to this complexity, the task of understanding performance bottlenecks of HPC applications and making improvements often ends up being a daunting trial-and-error process. Problematically, this process often starts without having a quantitative understanding of the actual behavior of the HPC code.

The Performance Optimisation and Productivity (POP) Centre of Excellence, funded by the EU under the Horizon 2020 Research and Innovation Programme, attempts to establish a quantitative methodology for the assessment of parallel codes. This methodology is based on a set of hierarchical metrics, where the metrics at the bottom of the hierarchy represent common causes of poor performance. These metrics provide a standard, objective way to characterize different aspects of the performance of parallel codes and therefore provide the necessary foundation for establishing a more systematic approach for performance optimization of HPC applications. In consequence, the POP methodology facilitates training new HPC performance analysts. In this paper, we will illustrate these advantages by describing two real-world examples where we used the POP methodology to help HPC users understand performance bottlenecks of their code.

KEYWORDS

Parallel performance analysis, HPC performance optimization, POP metrics

1 INTRODUCTION

High-Performance Computing (HPC) is an essential tool for science and industry. It is used to solve diverse problems such as weather forecasting, material design, drug discovery, climate modeling and predictions, etc. Most HPC facilities represent a major capital investment and run at a high level of utilization. Improving the efficiency of application software running on these facilities means less time to solution and more capacity to solve larger, more challenging problems.

Today's HPC facilities allow using hundreds to hundreds of thousands of cores to perform extensive calculations and process large amounts of data. Efficient use of these facilities has been proven to be extremely challenging. While HPC applications are often designed with performance in mind, many suffer from poor performance; here, unexpected behavior is more likely to happen, and an excellent design choice for a given problem size and a particular hardware might lead to poor performance for a different configuration.

There has been significant research and engineering effort to build tools that support performance optimization of HPC applications by collecting and analyzing their runtime behavior; for examples, see [5]. However, profiling or tracing HPC applications often results in large amounts of performance data that are difficult to interpret beyond simple observations. There is often a lack of a quantitative understanding of the actual behavior of parallel applications, and the performance optimization task, in turn, becomes a trial-and-error and ad-hoc process.

Performance Optimisation and Productivity (POP) is a Centre of Excellence (CoE) in HPC funded by the EU under the Horizon 2020 Research and Innovation Programme. The mission of POP [6] is to provide performance optimization and productivity services for HPC applications in all domains. POP attempts to achieve this while establishing general principles and systematic methods for parallel performance optimization.

POP has defined a scalable performance analysis methodology based on a set of hierarchical metrics [7], where each metric represents a common cause of inefficiency in parallel applications. These metrics are calculated using basic runtime statistics and provide a standard, objective way to characterize different aspects of performance of parallel codes, such as communication overhead and load imbalance. These metrics allow comprehensive comparison of the performance of a parallel application across different platforms or with different configurations (e.g. different numbers of threads/processes). This allows for a better understanding of program efficiency, quick diagnosis of performance problems, and identification of target kernels for code refactoring.

In Section 2 of this paper, we provide an overview of POP metrics for Message Passing Interface (MPI) applications and their calculation. Section 3 and 4 review performance assessments of two parallel applications using POP methodology. These examples are chosen from different domains: molecular dynamics simulation, Section 3, and computational fluid dynamics, Section 4. Section 5 concludes the paper.

https://doi.org/10.22369/issn.2153-4136/12/2/13

2 POP METRICS FOR PARALLEL PERFORMANCE ANALYSIS

In this section, we review the MPI performance metrics used and promoted by the POP CoE [7]. These metrics measure relative impact of parallel inefficiency factors on overall performance and provide a quantitative understanding of parallel application behavior.

The hierarchy of POP metrics for pure MPI applications, in fact applications written using any message-passing model, is shown in Fig. 1. The *Global Efficiency* at the top of the hierarchy indicates how well a parallel application scales. At this level, inefficiencies are typically due to two factors:

(1) overhead imposed by parallelism, represented with *Parallel Efficiency*, and
(2) poor scaling of computation with increasing number of processors, represented with *Computational scaling*.

The *Global Efficiency* is defined as product of the *Parallel Efficiency* and the *Computational scaling*. Going further down the hierarchy, both of these metrics are defined as the product of their own sub-metrics.

For MPI applications, the *Parallel Efficiency* reports inefficiencies due to either uneven distribution of computational work or overhead of data communication and synchronization between processes. These are measured with the *Load Balance* and the *Communication Efficiency*, respectively.

These two metrics are calculated using basic statistics from a program execution, including the total runtime and the computation time per process. Here, the computation time refers to the time that useful instructions are being executed, e.g. it excludes the CPU time in the MPI library. The *Load Balance* is defined as the ratio of the average computation time of all processes to the maximum computation time across all processes. The *Communication Efficiency* is defined as the ratio of maximum computation time to the total runtime.

The *Communication Efficiency* includes two metrics: *Transfer Efficiency* and *Serialization Efficiency*. The former indicates performance loss due to actual data transfer time. The latter reveals communication inefficiencies due to idle time within communication, i.e. when no data is transferred. This happens when processes wait at communication or synchronization points for other processes to arrive. To calculate these two metrics, we need to calculate the total runtime of the application on a system with an ideal communication network, i.e. what the runtime would be if data transfer were instantaneous. The *Transfer efficiency* is the ratio of the runtime on an ideal network to the runtime on the real system, and the *Serialization Efficiency* is the ratio of the maximum computation time to the total runtime on an ideal network.

Going up in the metrics hierarchy, the *Computational Scaling* shows how well the computation load scales with increased parallelism. It is calculated with respect to a reference execution case using total computation time, i.e. the time spent executing useful instructions summed over all processes. For example, when analyzing strong scaling behavior, it is calculated as the ratio of the total computation time for a reference case such as one processor (or one node) to the total computation time as number of processors (or nodes) is increased.

Multiple issues can lead to a poor *Computational Scaling* value, and they can be investigated using hardware performance counter data via interfaces such as PAPI counters [4]. In the POP hierarchy of metrics, the *Computational Scaling* is composed of three metrics:

- *Instruction Scaling*: compares the total number of instructions executed for different numbers of threads/processes. Decreasing values of this metric indicate that total computation load increases with employing more processes.
- *Instruction Per Cycle (IPC) Scaling*: compares how many instructions per cycle are executed for different numbers of threads/processes. Decreasing values indicate that rate of computation has slowed down. Decreasing cache hit rate and exhaustion of memory bandwidth are typical causes.
- *Frequency Scaling*: compares the processor frequency for different numbers of threads/processes. Decreasing values indicate that with increasing load, some cores operate with lower frequency.

Basic runtime statistics which are needed to calculate the POP metrics can be collected using almost any performance analysis tool. However, automatic calculation of the POP metrics is supported in the tools developed by Barcelona Supercomputer Center (BSC) [2] and the Jülich Supercomputing Centre (JSC) [9]. The former family includes Extrae for collecting performance data, Dimemas for simulating behavior of MPI applications under different network conditions, and Paraver and Basic Analysis for post-mortem trace analysis, including calculation of the POP metrics. The latter includes Scalasca and Cube for parallel performance analysis; Scalasca uses Score-p [10] for instrumenting parallel applications and collecting performance data.

By definition, the POP efficiency metrics can take values between 0 and 1, with higher numbers representing better performance. As a rule of thumb, values above 0.8 are considered acceptable, whereas lower values indicate performance issues that need to be explored in detail.

3 EXAMPLE 1 - MOLECULAR DYNAMICS SIMULATION

In this section, we describe the use of the POP metrics in assessing parallel performance of a molecular dynamics simulation (MDS) code. We call this code E1-MDS. E1-MDS uses MPI and consists of a legacy core written in Fortran with a layer of modern C++ on top. We did not have access to the source code. Performance data was collected by code developers using Extrae [2], running the application on their in-house server machine with a dual Intel Xeon Gold 6248 CPU (40 cores per socket).

Extrae uses instrumentation mechanisms[1] to collect performance data at known application points (e.g at MPI function calls) and collects trace data of the application runtime behavior. All performance data can be gathered in one file for post-mortem analysis. We were given trace data for the application running on 2, 10, 20, 30, and 40 cores, solving the same problem. Given these trace files, we used Basic Analysis [2] to calculate the POP metrics.

[1] Sampling mechanisms are supported as well.

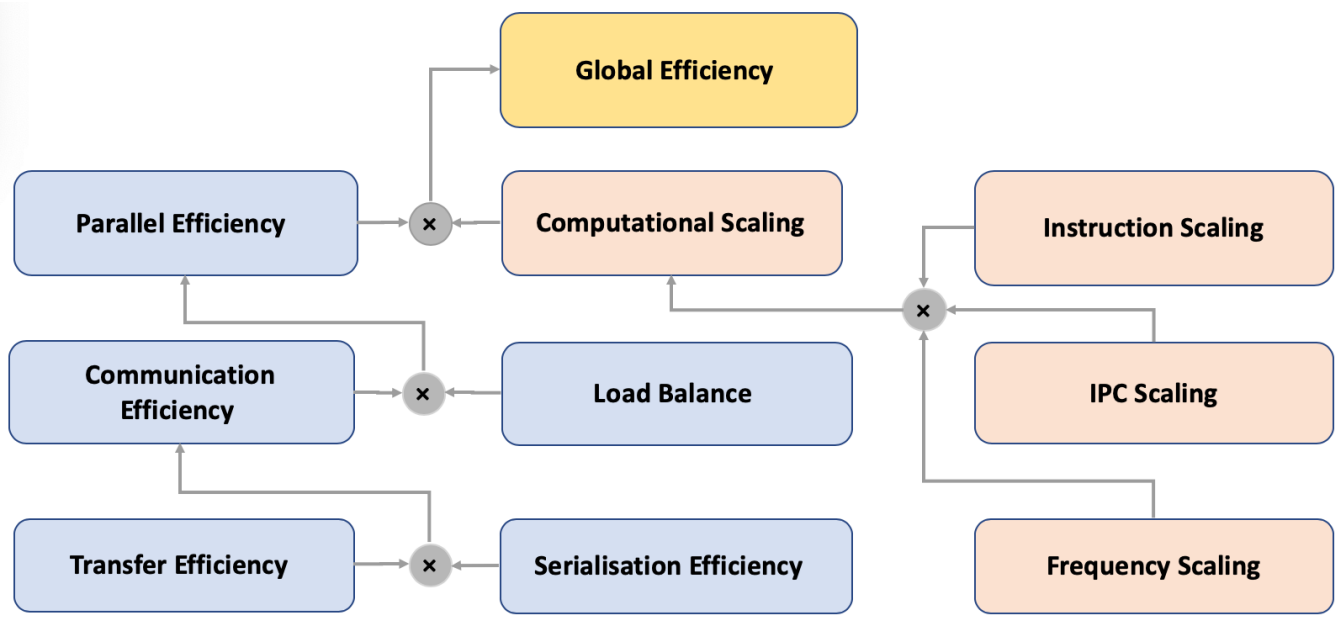


Figure 1: POP MPI Parallel Efficiency Metrics.

E1-MDS has three stages: initialization, main body of the simulation, and finalization. The time spend in initialization and finalization phases is negligible in comparison with the main body. Therefore, the performance assessment was focused on the second stage, i.e. the main body was the Region of Interest (ROI) for this assessment.

Figure 2 and 3, respectively, show the scalability plot for E1-MDS and the POP metrics calculated for ROI using 2, 10, 20, 30, and 40 cores. As shown in Fig. 2, the speedup drops below 80% of the ideal, i.e. linear speedup, on 10 cores, and it does not scale well beyond that. This is also evident in the *Global Efficiency* metric; it drops to 73% on 10 cores and gets as low as 36% on 40 cores.

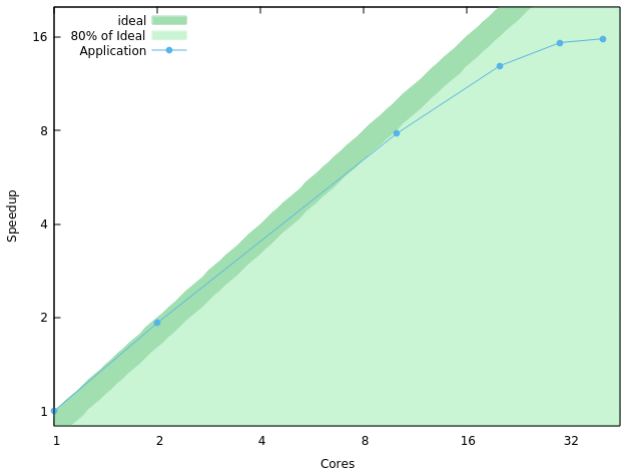


Figure 2: E1-MDS Scalability plot.

Figure 3 shows that the POP metrics decrease as the number of cores is increased. The values of these metrics reveal which factors contribute more in the loss of performance. The *Computational Scaling* drops below 80% on 20 cores, with the *Instruction Scaling* being the fastest dropping factor. The *Parallel efficiency* also drops below 80% on 30 and 40 cores, with the load imbalance being the major contributing factor. Therefore, according to the POP metrics, poor instruction scaling and load imbalance are the two main factors that limit scalability of the application.

Number of cores	2	10	20	30	40
Global Efficiency	0.95	0.73	0.60	0.47	0.36
Parallel Efficiency	0.95	0.89	0.81	0.75	0.68
Load balance	0.95	0.92	0.85	0.81	0.80
Communication Efficiency	0.99	0.97	0.95	0.92	0.85
Serialisation	0.99	0.99	0.99	0.98	0.94
Transfer Efficiency	0.99	0.98	0.96	0.94	0.91
Computational Scaling	1.00	0.82	0.74	0.63	0.53
Instruction Scaling	1.00	0.87	0.83	0.79	0.76
IPC Scaling	1.00	0.99	0.95	0.90	0.83
Frequency Scaling	1.00	0.95	0.94	0.88	0.83

Figure 3: E1-MDS, the POP metrics for ROI: poor instruction scaling and load imbalance limit scalability of the application.

The findings of the POP metrics were confirmed with further analysis of the trace data using Paraver [2]. The next step was to report our findings to the developers of E1-MDS. They could rather

quickly put their fingers on regions of the code that caused load imbalance. This, however, was not the case for the *Instruction Scaling*. It took some digging in the algorithms and the code to confirm that poor *Instruction Scaling* was due to duplicated computation. This is one of the strengths of the POP metrics. They can provide an insight into a code of which developers are ignorant.

This example showed how the POP metrics can help us to quickly diagnose the causes of poor parallel performance. This allows for a better understanding of program efficiency and the identification of target kernels for code refactoring. In case of E1-MDS, algorithmic changes are needed to make the code scalable on higher numbers of cores; however, using hybrid parallelism, i.e. OpenMP + MPI, can be a quick way to get better performance on the existing hardware with minimum code refactoring. Running the code with fewer MPI processes and using OpenMP to exploit extra free cores will improve instruction scaling and load imbalance.

4 EXAMPLE 2 - COMPUTATIONAL FLUID DYNAMICS

Our second example is a computational fluid dynamics code. It is an incompressible flow solver, and we refer to it as E2-CFD. E2-CFD uses MPI for parallelism, it is written in modern C++, and it depends on a couple of libraries for numerical computation. We had access to the source code. Performance data was collected using Scalasca [9], running E2-CFD on MareNostrum-IV [3] using 1, 2, 4, 8 and 16 nodes, where each node has 48 cores. Scalasca supports calculation of the POP metrics.

E2-CFD scales well on a couple hundred cores, and the speedup drops below 80% of ideal on 768 cores (16 nodes). The POP metrics for ROI are shown in Fig. 4. As can be seen, the *Global Efficiency* only drops below 80% on 768 cores with the *Communication Efficiency* and especially the *serialization* being the major contributing factors. The *IPC Scaling* improves on higher number of cores, likely due to better cache access. The *Instruction Scaling* also drops by about 8% on 768 cores but it is still above 90% and in the acceptable range. In short, the POP metrics suggest that for code optimization we need to find the regions of the code that cause low *Serialization Efficiency*. Serialization typically happens due to at least one process arriving early/late at a synchronization point.

Number of cores	48	96	192	384	768
Global Efficiency	0.91	0.93	0.97	0.86	0.69
Parallel Efficiency	0.91	0.87	0.87	0.75	0.58
Load balance	0.99	0.98	0.97	0.94	0.91
Communication Efficiency	0.92	0.89	0.89	0.8	0.64
Serialisation	0.93	0.91	0.93	0.87	0.73
Transfer efficiency	0.98	0.98	0.96	0.92	0.88
Computational Scaling	1.00	1.07	1.11	1.15	1.19
Instruction Scaling	1.00	0.99	0.97	0.95	0.92
IPC Scaling	1.00	1.08	1.16	1.27	1.44
Frequency Scaling	1.00	1.00	0.99	0.96	0.90

Figure 4: E2-CFD, the POP metrics for ROI: serialization is the main factor that limits scalability

To identify causes of poor *Serialization Efficiency*, we used delay cost analysis [1], which is available in Scalasca. The delay cost metric highlights the root causes of serialization by attributing processes' waiting time to the routines causing it [8].

This further analysis identified that low *Serialization Efficiency* was mainly related to a library function call, and it was caused by regions of computational load imbalance between MPI synchronization points and growing waiting time, especially in MPI collective calls.

In this example, POP metrics provide a quick insight on the causes of parallel performance loss. While we used other tools for further analysis and to locate problematic regions of code, the choice of this tool was guided by the POP metrics.

5 CONCLUSION

Attempts to optimize performance of HPC applications start with collecting performance data. This could result in large amounts of performance data that are difficult to interpret beyond simple observations. The problem is often a lack of a quantitative understanding of the actual behavior of HPC applications. To address this, POP CoE [6] has defined a set of hierarchical metrics [7], where each metric represents a common cause of inefficiency in parallel applications.

In this paper, we described the use of the POP methodology with two real-word examples. In both cases, POP metrics quickly and correctly highlighted causes of parallel inefficiency and provided the knowledge necessary to decide the best course of action to improve efficiency of the parallel applications. Both examples are production codes used in their respective communities. They belong to different domains of science and technology and run on different scales. This is the other advantage of the POP metrics; they work across domains and scales. The POP metrics establish a systematic and efficient approach for parallel performance evaluation, help HPC users to better understand performance bottlenecks of their codes, and facilitate training new HPC performance analysts.

REFERENCES

[1] D. BOHME, M. GEIMER, L. ARNOLD, F. Voigtlaender, and F. Wolf. 2016. Identifying the root causes of wait states in large-scale parallel applications. *ACM Trans. On Parallel Computing* 3, 2 (2016).

[2] BSC-tools [n. d.]. Perforamnce Parallel Tools Developed at BSC. https://tools.bsc.es.

[3] MareNostrum [n. d.]. https://www.bsc.es/marenostrum/marenostrum.

[4] PAPI [n. d.]. Performance Application Programming Interface. http://icl.cs.utk.edu/papi/.

[5] POP [n. d.]. Parallel Performance Tools. https://pop-coe.eu/partners/tools.

[6] POP [n. d.]. The POP CoE. https://pop-coe.eu/.

[7] POP-Metrics [n. d.]. POP Standard Metrics for Parallel Performance Analysis. https://pop-coe.eu/node/69.

[8] Scalasca [n. d.]. Performance properties. https://apps.fz-juelich.de/scalasca/releases/scalasca/2.5/help/scalasca_patterns-2.5.html#delay

[9] Scalasca [n. d.]. A Software Tool for Performance Optimization of Parallel Programs. https://www.scalasca.org.

[10] Score-p [n. d.]. Scalable Performance Measurement Infrastructure for Parallel Codes. https://www.vi-hps.org/projects/score-p/.

February 2021

Volume 12 Issue 2

ISSN 2153-4136 (online)