Introducing Novices to Scientific Parallel Computing

Stephen Lien Harrell
Purdue University
sharrell@purdue.edu

Betsy Hillery
Purdue University
eahillery@purdue.edu

Xiao Zhu
Purdue University
zhu472@purdue.edu

ABSTRACT

HPC and Scientific Computing are integral tools for sustaining the growth of scientific research. Additionally, educating future domain scientists and research-focused IT staff about the use of computation to support research is as important as capital expenditures on new resources. The aim of this paper it to describe the parallel computing portion of Purdue University's HPC seminar series which is used as a tool to introduce students from many non-traditional disciplines to scientific, parallel and high-performance computing.

KEYWORDS

JOCSE submissions, Undergraduate, Parallel Computing, Training, HPC

1 INTRODUCTION

Scientific computing supports a wide range of disciplines to enable new and exciting research topics and to create new opportunities for multidisciplinary collaborations, which are vital for cutting-edge research [13]. High performance computing (HPC) permits exploration of complex phenomena that cannot be observed or replicated by experiment. Recently, data-intensive science has emerged as, considered by many, the fourth paradigm of scientific discoveries [8]. Universities, research organizations, businesses and government entities are working to create the best possible environment for research and innovation to ensure the long-term success of computational scientific research. Educating future domain scientists and research-focused IT staff about the use of computation to support research is as important as the supercomputers themselves. A recent report by the NSF Cyberlearning Workforce Development (CLWD) Task Force states that "computational science must be introduced into the K-20 curriculum in ways that build deep understanding and stimulate further exploration. At the undergraduate level, interdisciplinary computational approaches have essential roles both as separate content areas and incorporated into existing math and science (including social and behavioral sciences) curriculum. These interdisciplinary computational approaches, including computer science, have to be presented as more than just programming" [14]. Similarly, NITRD's High End Computing Interagency Working Group suggests an approach that includes "Development of the next generation workforce in undergraduate and graduate university programs through collaborative curriculum development to establish base skills" [9]. Additionally, the necessary skills needed are varied depending on the specific research topics and typically require many fields of knowledge to be covered[12]. In this paper we will discuss Purdue Research Computing's approach to teaching novices (often in scientific undergraduate programs) and how to use parallel and data-intensive computing through a variety of lectures and exercises. By doing that, we aim to give undergraduate students an opportunity to explore the field of HPC and big data in a non-traditional computer science course setting and build a basic foundation of computational and data skills for their further education and research activities.

1.1 Inspiring Undergraduates

As part of Purdue University's Sesquicentennial anniversary campaign, students are asked "What if". What if we could control the brain for better health? What if we return to Pluto? What if the world ran on 100 percent renewable energy? At Purdue, most undergraduates are likely familiar with these lines from this "what if series". However, they may not realize that many researchers will rely on advanced computing and data solutions to enable them to answer these complex questions.

Through computational science, we inspire students to change the world. In this class, we pay special attention to hot topics, such as climate change and artificial intelligence, in order to give students an extra push to spend both extra time with their homework and exercises as well as consider graduate work and/or staff roles within advanced computing technologies.

1.2 Prior Work at Purdue

Purdue's Research Computing has had a history of mentoring, training, and educating students in HPC. Although we are not alone in these actions [3][4], our staff have had great success mentoring undergraduate students in Systems-Facing as well as Research-Facing roles [5]. The Student Cluster Competition [7] has also been a useful tool to inspire students to consider HPC as a career. In the classroom we have been developing techniques to explore the breadth of HPC [6] and broaden participation from under-served demographics [10].

2 HIGH PERFORMANCE COMPUTING SEMINAR

The primary goal of the class is to have undergraduates recognize that computing is an important creative vehicle for scientific discovery on a myriad subjects, ranging from physics to social studies. Aligning to this goal, we employed an integrated and informational approach in teaching computing for this course. Specifically, we integrated parallel computing instruction with different scientific domains. To do this we adapted a combination of lectures from domain science faculty and created labs where students led the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

© 2020 Journal of Computational Science Education
https://doi.org/10.22369/issn.2153-4136/11/1/14

Figure 1: Entire Class with Instructors

discussions on the tools and assignment. As the course designers, we began with these learning outcomes in our mind:

- A good understanding of scientific workflow
- Familiarity of building and using scientific applications
- Basics of parallel computing, such as difference between multi-node parallelism and node level parallelism
- Overview of state-of-art computing architectures (e.g. accelerators)
- Performance characteristics (strong and weak scaling) and their connection with the architecture choices
- Bottlenecks in HPC (e.g. communication and data movement) and strategies to minimize them

2.1 Approach

Our approach to this curriculum was twofold, we engaged students with hands-on exercises using a real-world scientific application and regularly lectured on more general parallel computing topics in the class. For their first assignment, we asked the students to follow a typical workflow of a weather forecast experiment and reproduce the results. Specifically, convert numerical weather prediction data from the National Weather Service into a full input grid for WRF, run WRF, and interpret the output results. During the lecture, the students were taught parallel computing concepts.

2.2 Broadening Participation

In order to communicate the availability of this newly created class, the instructor team was deliberate when it came to effectively advertising it campus wide and how we would engage the larger undergraduate community on campus. During the initial recruitments, the HPC Seminar leaders made a number of visits to clubs on campus, specifically clubs such as Women in Computer Science, Women in Engineering and the Women in HPC. Along with these recruiting opportunities, direct communication was sent to student peers of the previous all-female Student Cluster Challenge team [10] in an effort to further attract additional participants.

One goal of this experience was to create interest in the class from non-traditional computer science students âĂŞ specifically by taking advantage of our current Research Computing employees teaching across the university. Additionally, the faculty sponsor of the class made direct contact with the Data Mine [10], a large-scale living learning community for undergraduates from all majors, focused on Data Science for All, in an effort to recruit students that are not typically Computer Science students.

2.3 Syllabus

2.3.1 Course Description. This course introduces undergraduates to advanced topics in High Performance Computing clusters, operating systems, and the cluster batch-operating systems. Topics covered in this course focus on aspects of the design, implementation, and use of high performance computing systems at the system level. No previous experience with operating systems or programming is required.

2.3.2 Course Objectives. Students will be able to effectively communicate general High Performance Computing (HPC) concepts and knowledgeable on how scientific applications run on HPC resources. The specific learning objectives for this course are:

(1) Students will effectively communicate how to build and compile scientific applications
(2) Students will effectively understand the basics of the Linux Shell
(3) Students will effectively communicate how scientific related topics relate to high performance computing

2.4 Lectures

2.4.1 Introduction to the Linux Shell. Review the basic command line interface. Students receive a solid foundation in how to use the terminal and how to get a computer to do useful work. Some materials were adapted from Software Carpentry lessons. [2]

2.4.2 Compiling and running HPL. Introduce HPL as a measure of a computer system's floating point computing power, thus providing data for the Top500 list to rank against supercomputers worldwide. Students will understand and practice the usage of a benchmark program on a cutting edge HPC system.

2.4.3 Installing Scientific Applications. In this lecture students are familiarized with one of the most difficult tasks in the HPC world, installing new scientific software packages. Students are introduced to a few common used tools for managing the build process of packages, such as Automake and CMake.

2.4.4 History of Weather and Computing - Guest Speaker. A historical review of the connection between numerical weather prediction and high performance computing and how the advancement of computing technology changes research in the field. The lecture was given by Associate Professor Mike Baldwin from the Department of Earth, Atmospheric, and Planetary Sciences at Purdue.

2.4.5 Weather Prediction - Guest Speaker. Professor Baldwin presented a few weather case studies that require some of the most powerful supercomputers in the world. He also showcased his Purdue football game day weather forecast.

2.4.6 Cluster Design. Present the various factors in designing a computing cluster, such as performance, availability, scalability, cost, and the range of applications. Additionally, the impacts that applications can have on those factors.

2.4.7 Big Data - Guest Speaker. Introduce the basic ideas of big data to analyze and systematically extract information. Students were giving a tutorial on how to use R to manipulate data sets too large or complex to be dealt with by traditional data-processing tools. The lecture was given by Associate Professor Mark Ward of Purdue's Statistics Department.

2.4.8 Science Writing and Presenting. Students are provided an overview on effective scientific communication. Both presentation and scientific writing was covered. Some fundamental tips and techniques for effectively writing and presenting scientific information are given.

2.4.9 Computational Fluid Dynamics - Guest Speaker. Purdue Mechanical Engineering Professor Carlo Scalo illustrated the examples of using HPC and numerical analysis in solving problems that involve fluid flows. His primary example of fluid flows was aircraft design.

2.4.10 Molecular Dynamics - Guest Speaker. Purdue Material Scientist Alejandro Strachan presented predictive atomistic and molecular simulation to describe materials from first principles and their application to problems of technological importance. His presentation included shape memory and high-energy density materials. Dr. Schachan also demonstrated interactive simulation tools on nanoHUB.org [11], a community-contributed resource for nanotechnology.

2.4.11 Astrophysics - Guest Speaker. Research Computing Staff Matthew Route shared experiences in running a variety of current parallel codes in astrophysics. He presented examples on both large-scale simulations and big-data analysis of observational data sets.

2.4.12 Introduction to Python. Students learned the fundamentals of the Python programming language, along with some of the programming best practices. A few examples include using Python data types and variables to represent and store data, and using conditionals and loops to control the flow of your programs.

2.4.13 Introduction to Juypter Hub and R Studio. An introduction to two popular interactive and flexible computational environments for data analysis and graphics.

2.4.14 Final Presentation. Students presented in group about what they learned from this class and their suggestions for improvement. Excerpts from these presentations are available in section 4.1.

2.5 Assignments

2.5.1 Student Biographies. To get to know the students, we had each student responsible for writing biographies about themselves highlighting their area of study and special interests

2.5.2 Fundamentals of Compiling Applications. To warm the students up for the major assignments, each student had to learn how to compile HPL using Spack. Then each student did the same exercise compiling HPL by hand. The idea is to have them understand how to read logs and dependencies of compiling applications.

2.5.3 Compiling and Running WRF and OpenFoam. Students compiled and ran WRF and OpenFoam and did visualizations of their findings. These two applications make up the core assignments for the class and are described in detail in section 3.

2.5.4 Guest Lecture Prompt Responses. After each guest lecture the students received writing prompts about the science being discussed, how do we apply the lecture to the High Performance Community and how would they apply the science to solving a problem using a cluster.

2.5.5 Final Team Project. The students at the of the semester were asked to put together a power point presentation. They needed to answer 5 fundamental questions

- Describe your experience with building and compiling applications. Name the applications that you have built, what they were used for and what struggles you had building them.
- Discuss your understanding of Linux, what your experience was before attending the class and any new things you have learned.
- Describe how scientific related topics relate to high performance computing. Outline at least one presentation that you attended and how it relates to the class.
- Discuss your opinion on the scientist presentations that were held in class. Were they valuable? Why or Why not? Which presentations if any did you like? Why or Why not?
- Discuss your opinion of the pace and difficulty of the class. Be specific and describe the track you involved in.

They then presented their findings to the class and invited guests.

3 TEACHING SCIENTIFIC APPLICATIONS

To teach parallel computing principles, we chose WRF, widely used on various HPC systems, as an illustrative scientific application. WRF is related to the weather modeling history, and its background taught by Dr. Baldwin for this class. Additionally, WRF is a ubiquitous scientific code with layman-relatable input and output data. We designed the lecture and hands-on WRF excersizes to achieve the learning outcomes from Section 2. All assignments and hands-on exercises were performed using Purdue's teaching and learning cluster, Scholar [1].

Keeping the students' learning outcomes in mind, the second application chosen was OpenFOAM, a popular open source software for the solution of continuum mechanics problems, most prominently with computational fluid dynamics. The goal of the assignment was to have students further hone their skills learned in the class to solve a practical problem independently. In contrast to the WRF exercise, we decided not to reserve any class time for questions (discussions among students in person or on Slack are encouraged).

3.1 WRF

First, students learn the basics of parallel computation and MPI. Secondly, they learn the advantages and disadvantages of different parallel paradigms such as distributed memory parallelism vs. shared memory parallelism. In practice, students also learn the intricacy of properly setting up a parallel computing environment. Finally, students gain a good knowledge of the performance characteristics and how this will be impacted by the choice of hardware architectures. They are also asked to perform scaling studies (strong vs weak). Which illustrates the bottlenecks in parallel computation, such as network and I/O related overhead.

While WRF is a very common application among meterologists, when aimed at undergraduates it has some detriments. First, it can be very hard to build for a novice. For instance, at compile time you must know the differences between node-level and multi-node parallelism and the important frameworks for each. Additionally, some features are are hidden behind compile-time options in a non-obvious way. For instance the choice of NETCDF 3 vs 4 has consequences that may require one to recompile if the data set does not match. This is all but obvious for weather scientists, however, it requires a well-grounded understanding of Linux user-space environments as well as parallelism in standard HPC clusters today. Finally, the workflow for WRF is more complex than some other scientific applications, requiring the use of multiple executables from multiple packages in order to do a full weather simulation, which is well documented in the language of meteorologists, but not undergraduates.

3.2 OpenFoam

This assignment was given to the students after the Dr. Scalo's lecture on computational fluid dynamics. By this point the students were more comfortable with software dependencies and how to use tools like Spack, which was covered in a previous tutorial. Also, they had a basic idea of fluid dynamics and a few important terms after the guest lectures. Both would tremendously help them find a viable solution for the assignment.

The individual assignment for the student was to take the software OpenFOAM and manually compile the software or use Spack. Once the students had the application compiled, the next step was to run the simulations and visualize what they had obtained. Each student was provided an initial input file, but once they were comfortable visualizing the assignment, the next step was to change the input file to see the effects. The students used Paraview, which they also needed to familiarize themselves with to visualize the results. In this assignment, students were asked to solve a problem of simplified dam break in two dimensions, where a transient flow of water separated by a sharp interface. Knowing how to see the effect of water breaking over a dam, students were more engaged than in the WRF exercise, discussed in Section 4.1.1. We found that most students were able to successfully present demo simulation to the class.

Figure 2: Students Presenting about Learning Outcomes

4 OUTCOMES

4.1 Student Evaluation of the Class

In the final project, students were given prompts regarding the learning objectives of the class and if they were met, students presented their responses seen in Figure 2. Focusing on three of the prompts, the responses below are illustrative of the general outcomes of the class, the following information was conveyed:

4.1.1 Describe your experience with building and compiling applications. Students found that the directions for installing Spack and OpenFoam were generally straightforward and easy to follow, however, in practice it was very difficult to compile and run correctly. Additionally, the logic behind the steps was not explained thoroughly, so it was difficult to troubleshoot errors in the homework assignments. Through the exercises the understanding of Linux increased dramatically and although the students found the method of learning difficult, this learning outcome was met. Many students finished the class knowing how to navigate the shell and how it interacts with the applications.

However, there were also frustrations with this format. Such as the speakers didn't understand the class objective was to learn about HPC and it's different implementations, not just whatever the invited speaker science was. It would have been better if the speakers focused more on how they use computers and HPC to do their research and less on the details of what their research is for.

4.1.2 Discuss your opinion of the pace and difficulty of the class. One student said that having to learn the science and parallel computing at the same time was too much. Another said that the class in general was very fast paced.

4.1.3 Discuss your understanding of Linux, what your experience was before attending the class. Students stated they believed it was very important to have a base understanding of Linux before attending the class, as there was not enough of "getting to know Linux" done in class.

4.2 Lessons Learned

While considerable success was achieved, such as compiling WRF and getting correct simulation results, it was difficult for the students to fully appreciate the entire process due to their lack of meteorological knowledge. Additionally, the pace of the class had to slow down to accommodate the majority of the students, and consequently, instructors had no time for covering the visualization of the simulation results. This resulted in dampening the students' interest in the topic and thus the students became largely dependent on instructors for the finishing assignments for this portion of the class.

A better strategy would have been to have the students visualize an existing WRF output file, allowing the students to become familiar with meteorological visualizations before attempting the somewhat daunting exercise of compiling and running WRF for the first time. Although it does not follow the actual sequence of weather modeling, we propose that this would better keep students' attention and keep them motivated throughout the more esoteric work during the hands-on section. More importantly, such experience highlights the key difference in instructional design between a graduate course and an undergraduate one.

A second key lesson learned was that the Linux skills required for this type of class were a serious impediment to students interest and learning. A majority of the students started with little to no experience in Linux, even with a class period dedicated to command line basics, their understanding was insufficient. This put a lot of burden on the students to learn how to compile an application without knowing how to navigate the environment. In future classes, it is recommended that either Linux knowledge becomes a requirement for the class or a more significant amount time is dedicated to this topic.

5 FUTURE WORK

For the next HPC seminar we run we plan to vet prior Linux experience and split the first few classes between experienced Linux users and novices. This will allow novice users to get a more complete understanding of basic Linux skills. The other major change we will implement is switching visualization to be first when teach scientific applications, this approach allows the students to visually see the science in action.

ACKNOWLEDGMENTS

We'd like to acknowledge Dr. Mark Ward, Dr. Mike Baldwin, Dr. Carlo Scalo, Dr. Alejandro Strachan and Dr. Matt Route for sharing their knowledge with our students. We'd also like to thank Christopher Phillips for his keen editorial skills.

REFERENCES

[1] M. E. Baldwin, X. Zhu, P. M. Smith, Stephen Lien Harrell, R. Skeel, and A. Maji. 2016. Scholar: A Campus HPC Resource to Enable Computational Literacy. In *2016 Workshop on Education for High-Performance Computing (EduHPC)*. 25–31. https://doi.org/10.1109/EduHPC.2016.009

[2] Software Carpentry. [n. d.]. https://swcarpentry.github.io/shell-novice/

[3] Dirk Colbry. 2014. iCER Interns: Engaging Undergraduates in High Performance Computing. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment (XSEDE '14)*. ACM, New York, NY, USA, Article 71, 5 pages. https://doi.org/10.1145/2616498.2616573

[4] Andrew Fitz Gibbon, David A Joiner, Henry Neeman, Charles Peck, and Skylar Thompson. 2010. Teaching high performance computing to undergraduate faculty and undergraduate students. In *Proceedings of the 2010 TeraGrid Conference*. 1–7.

[5] Stephen Lien Harrell, Marisa Brazil, Alex Younts, Daniel T. Dietz, Preston Smith, Erik Gough, Xiao Zhu, and Gladys K. Andino. 2018. Mentoring Undergraduates into Cyber-Facilitator Roles. In *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*. ACM, New York, NY, USA, Article 70, 7 pages. https://doi.org/10.1145/3219104.3219138

[6] Stephen lien Harrell, Benjamin Cotton, Michael Baldwin, and Andrew Howard. 2013. Developing a Scientific Computing Cluster Course for the Undergraduate Curriculum. In *Summit for Educators in System Administration 2013*. Washington D.C. http://funnelfiasco.com/research/sesa13.pdf

[7] Stephen Lien Harrell, Hai Ah Nam, Verónica G. Vergara Larrea, Kurt Keville, and Dan Kamalic. 2015. Student Cluster Competition: A Multi-disciplinary Undergraduate HPC Educational Tool. In *Proceedings of the Workshop on Education for High-Performance Computing (EduHPC '15)*. ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/2831425.2831428

[8] Tony Hey. 2012. The Fourth Paradigm–Data-Intensive Scientific Discovery. In *E-Science and Information Management: Third International Symposium on Information Management in a Changing World, IMCW 2012, Ankara, Turkey, September 19-21, 2012. Proceedings*, Vol. 317. Springer, 1.

[9] High End Computing Interagency Working Group. [n. d.]. *Education and Workforce Development in the High End Computing Community*. Technical Report. NITRD.

[10] Elizabett Hillery, Mark Daniel Ward, Jenna Rickus, Alex Younts, Preston Smith, and Eric Adams. 2019. Undergraduate Data Science and Diversity at Purdue University. In *Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning) (PEARC '19)*. Association for Computing Machinery, New York, NY, USA, Article Article 88, 6 pages. https://doi.org/10.1145/3332186.3332202

[11] Gerhard Klimeck, Michael McLennan, Sean P Brophy, George B Adams III, and Mark S Lundstrom. 2008. nanohub. org: Advancing education and research in nanotechnology. *Computing in Science & Engineering* 10, 5 (2008), 17.

[12] S. Lathrop. 2016. A Call to Action to Prepare the High-Performance Computing Workforce. *Computing in Science Engineering* 18, 6 (Nov 2016), 80–83. https://doi.org/10.1109/MCSE.2016.101

[13] National Academy of Sciences, National Academy of Engineering, and Institute of Medicine. 2005. *Facilitating Interdisciplinary Research*. The National Academies Press, Washington, DC. https://doi.org/10.17226/11153

[14] Task Force on Cyberlearning and Workforce Development. 2011. *A Report of the National Science Foundation Advisory Committee for Cyberinfrastructure*. Technical Report. National Science Foundation.