J. Parallel Distrib. Comput. ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Unifying computing resources and access interface to support parallel and distributed computing education

Linh B. Ngo [a,b,*], Ashwin Trikuta Srinath [a], Jeffrey Denton [a], Marcin Ziolkowski [a]

[a] Clemson Computing and Information Technology, Clemson University, Clemson, SC, United States
[b] School of Computing, Clemson University, Clemson, SC, United States

HIGHLIGHTS

- Design of a framework combining various computing resources and browser-based access interfaces.
- Description of learning modules for a CS course leveraging this framework.
- Reports on students' skill improvement and feedback on lecture contents and course project.

ARTICLE INFO

Article history:
Received 31 July 2017
Received in revised form 9 February 2018
Accepted 26 February 2018
Available online xxxx

Keywords:
Computing resources
MPI
MapReduce
Spark
Python
Distributed systems
Parallel computing education

ABSTRACT

This article presents how various on-site and remote computing resources are combined into a framework to support teaching parallel and distributed computing (PDC) at the undergraduate level. The combination of these resources enables the delivery of PDC programming, system, and architectural concepts via a browser-based common interface (JupyterHub) and a single programming environment (Python and its supported libraries). This also allows lecturers and students to focus more on the principles of PDC and less on the technicalities of native languages for different platforms. We describe how this framework can support a comprehensive set of PDC course modules, including lectures, assignments, and projects, for a full semester junior-level class. Adoption of this framework in various teaching environments at Clemson University has received positive feedback from both instructors and participants.

Published by Elsevier Inc.

1. Introduction

Parallel and distributed computing (PDC) has become integral to various aspects of IT across all academic disciplines and industrial areas. It is critical that college graduates are properly introduced to PDC core concepts and technologies for their future careers. The existing body of PDC knowledge spans across four different areas: Data Structures and Algorithms, Software Design, Software Environments, and Hardware [8]. A traditional Beowulf-based computing cluster [40], available through either on-site or public resources, can adequately provide a classroom computing environment for this knowledge, as shown in Fig. 1.

However, there remains a number of issues in working within this environment. First, interactions with computing clusters are typically done through a Linux-based command line interface (CLI). Terminal tools to support these interactions are available by default on Linux and Mac, but not on Windows. A portion of the class time must be spent on ensuring that all students can access the computing infrastructures. This necessitates that instructions and technical support for Linux and Mac terminals, and terminal emulators for Windows (e.g., PuttY or SSH Secure Shell) need to be prepared. The second issue arises from the need to include various new technologies in the curriculum as PDC moves beyond the traditional high performance computing concepts and into areas of data-intensive computing, big data analytics, and large-scale streaming systems. This leads to an increase in the number of computing tools and platforms that need to be taught together with the corresponding PDC concepts. To deliver this expanding set of new PDC concepts and enable students to have adequate hands-on practice within the limited class time, instructors will need to face in-class technological issues that will most certainly arise from working with these various tools and platforms. Furthermore, while most of these new platforms support Java, a commonly taught language within the core curriculum, the code complexity

* Corresponding author at: Clemson Computing and Information Technology, Clemson University, Clemson, SC, United States.
E-mail addresses: lngo@clemson.edu (L.B. Ngo), atrikut@clemson.edu (A.T. Srinath), denton@clemson.edu (J. Denton), zziolko@clemson.edu (M. Ziolkowski).

https://doi.org/10.1016/j.jpdc.2018.02.020
0743-7315/Published by Elsevier Inc.

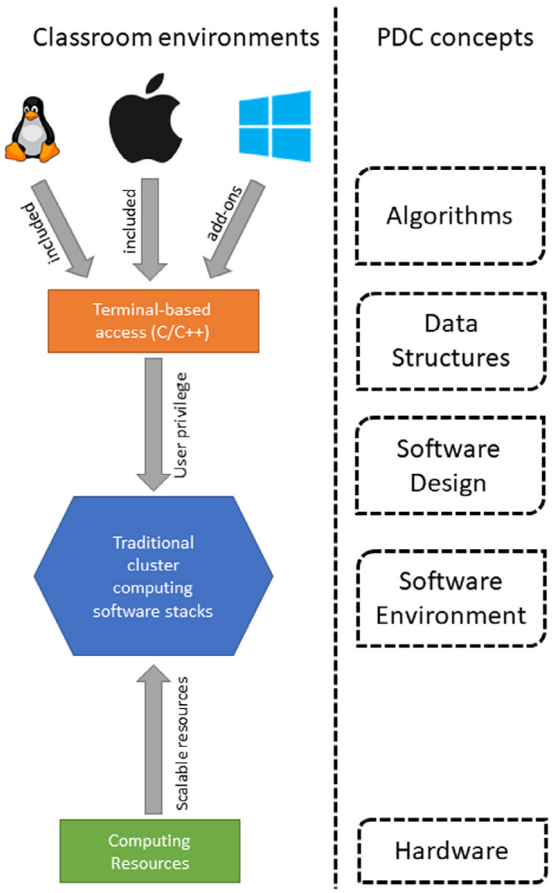

Fig. 1. A traditional classroom computing environment for teaching PDC concepts.

is significant. For example, the well-known Hadoop MapReduce WordCount program [1] requires fifty five lines of Java code. Out of these lines, only ten are directly related to the MapReduce programming portions, while the remaining forty five include library imports, class and function declarations, and standard job configurations. In both formal and informal educational settings, explanation of these lines present significant timing overheads during the hands-on practice segments of the lectures. They distract learners from programming paradigms and workflow designs, which are more important aspects of the learning process. In many cases, they require instructors to spend valuable class time for trouble shooting and technical support.

In this paper, we describe our approach at Clemson University in addressing these issues by unifying on-site and remote computing resources, access platforms, and programming tools into a common PDC educational framework. This framework is shown in Fig. 2. To support the traditional PDC concepts, we utilize Clemson University's research cluster, the Palmetto Supercomputer (Palmetto). For advanced PDC concepts, educational modules are developed using CloudLab, a public research computing testbed [37]. To combine these two resources, we deploy and integrate JupyterHub as the front-end for Palmetto. JupyterHub allows users to spawn Jupyter server, a platform that provides convenient access to computing environments for high performance computing, big data, and data intensive computing infrastructures. With Jupyter's ability to support a diverse set of programming languages, a browser-based terminal, and a text editor, students can be introduced to PDC through a standardized browser-based interface across different operating systems. The browser-based terminal also allows students to seamlessly interact with CloudLab from Palmetto.

The default language for JupyterHub Notebooks is Python. The availability of various community-supported Python libraries for PDC allows instructors to teach PDC concepts from different platforms without having students learn the various native languages of these platforms. As a result, students can spend more time understanding PDC concepts and less time on syntax correctness of specific tools and languages. This facilitates the instructional delivery of the most common PDC areas such as high performance computing, data-intensive computing, and in-memory distributed computing, which were originally designed for different computing platforms using different languages. With the popularity of Python and the variety of Python libraries/APIs that support parallel and distributed programming, there exists a number of tutorials and teaching materials for PDC using Python. Examples include the exhaustive tutorial for Python and MPI by the creators of mpi4py [12] or the framework to teach MapReduce programming via a web browser [17]. Our framework will further extend the ability to utilize Python to teach PDC by developing Python-based learning modules on a common and consistent interface, the Jupyter notebooks.

The unified educational framework for PDC presented in this article will enable important aspects of teaching PDC [21] such as showing speedup, real time results, visual results, interactivity, active learning, reproducibility, and accessibility. All resulting educational modules are available online. The remainder of this paper is organized as follows. Section 2 describes Palmetto and CloudLab, the on-site and remote computing resources, as well as the design and deployment of JupyterHub on Palmetto. Section 3 contains three topics. First, We discuss previous work that used Python in PDC education. Second, we describe course modules that use Jupyter and Python to teach high performance computing, data-intensive computing, and in-memory distributed computing concepts. Third, we show how the course modules can combine CloudLab, Jupyter, and Python in teaching advanced topics such as distributed system architectures, schedulers, and distributed file systems. Section 4 describes user evaluation of this unified educational computing framework from participants in various workshops and students from a class that teaches high performance and data-intensive computing. The evaluation includes performance observations and feedback from students and instructors who use the platform. Section 5 concludes the paper and discusses future work.

2. Unifying computing resources and access platforms

It is typical for PDC to be primarily taught as a single course within the entire undergraduate CS curriculum while having some PDC concepts embedded in other courses [9,42]. At Clemson University, PDC education is delivered through two settings, formal academic courses and informal half/full-day training workshops.

The PDC academic course is CPSC 3620, a computer science course on the topic of distributed and cluster computing. This course is taught in fifty-minute classes three times a week. While this is a required junior-level course, most students in the class wait until the first or second semester of their senior year before taking the course. The enrollment of the class ranges between 40 and 45 students. The course is intended to present students with a broad overview of PDC. The topics covered include distributed file systems, the message-passing programming paradigm, scheduling on a cluster of computers, big data and data-intensive computing, the map-reduce programming paradigm, in-memory distributed computing, cloud computing, message-oriented middleware, and distributed stream processing [28].

Please cite this article in press as: L.B. Ngo, et al., Unifying computing resources and access interface to support parallel and distributed computing education, J. Parallel Distrib. Comput. (2018), https://doi.org/10.1016/j.jpdc.2018.02.020.