Coding in the Cloud - Capturing Programming Behaviors at Scale

Dan Connors and Kyle Dunn
Department of Electrical Engineering
University of Colorado Denver
Denver, Colorado 80217–3364
Email: \{dan.connors, kyle.dunn\}@ucdenver.edu

Abstract—Access to computing resources, specifically high-performance parallel systems, is still a fundamental barrier to providing the necessary learning environment for many students in the Parallel and Distributed Computing (PDC) courses. Many top-ranked universities may have ample computing facilities for both research and education, however, other programs are not able to provide competitive infrastructure and therefore must develop different strategies. While virtual machine environments on student-owned personal computers are effective for providing a consistent low-cost and uniform platform to students, there are deficiencies in their use for specific areas of the PDC curriculum, such as performance analysis, tuning, and optimizations techniques. Moreover, the virtualized bring-your-own-system model eliminates a core and critical opportunity emerging in cloud environments, that development in the cloud can be centrally analyzed by teachers to assess learning objectives, evaluate course outcomes, and provide dynamic feedback to instructors. Forms of cloud coding in which programmers develop and run on remote servers using web-based interfaces have significant potential in broadening teaching efforts in PDC.

The talk will present the experience with providing a large class of upper-level undergraduate and graduate students with access to a high-performance cloud coding server that supports IPython Notebooks. In addition to the principles of accessing the system from any web browser, the developed framework allows instructors to observe, replay, and catalog the student programming process. The system has been used to help integrate multi-threading and GPU programming.

I. INTRODUCTION

Development in cloud environments has many advantages, starting with the ability to allow developers to use a range of devices in numerous locations to complete the same task. Thus, “Cloud coding” or “coding in the cloud” loosely define the process of leveraging the cloud as a development environment for programming projects. While cloud development is the focus of business productivity, such environments are not standard in the education realm. There are many opportunities to extend the cloud to the computer science curriculum area of Parallel and Distributed Computing (PDC). In many ways, cloud environments can cast as an equalizing element across different tier institutions, providing all students with equal opportunities to reach their full potential.

Cloud coding can be tracked alongside the emergence of Massively Open Online Courses (MOOCs), that aim to increase and broaden the participation of higher education. There are a number of MOOC systems [6] such as Udacity, EdX, Coursera that have steadily growing numbers of participants that range from full-time students to hobbyists. By using the scaling nature of the internet, MOOCs evolve the use traditional course materials and digitally extend the reach of lecture videos, readings, assignments, discussion forums, and even teaching assistants (or coaches) to create a new method of distance learning and hybrid learning. One goal of many of the MOOC courses is to reach as many participants as possible. While traditional liberal arts courses may not stress the digital computing resources at scale, a course such Scott Rixner’s at Rice University “An Introduction to Interactive Programming in Python” [7] requires participants to have access to Python framework. Rixner’s CodeSkulptor framework, accessible at www.CodeSkulptor.org, is an outstanding component of cloud coding as it allows students to learn Python programming and develop interactive programs without specific software support except a web browser and an internet connection. The critical question is how to provide the same level of infrastructure and accessibility to learners in PDC and computer science.

Virtualization systems as in bring-your-own-device [8] are one low-cost opportunity to provide a uniform environment to learners. Such systems are effective at providing a uniform and extensive environment for computer science teaching objectives. However, the virtualization model eliminates for the large part a core advantage of the MOOC platforms: observing and understanding the learning process. As learners digitally interact in MOOC platforms, there is significant advantage to understanding and analyzing learning behaviors. Simple aspects of correlating the habits video watching such as number of re-watched lectures, viewing speeds, and the content specifically presented, are rich with feedback to educators on how the presentation of course concepts are working. Such tracking is not easily realizable in the bring-your-own-device model, but is possible in cloud coding environments.

Since 2007, Hwu teamed up with NVIDIA to create a course called Programming Massively Parallel Processors, taught on Coursera as “Heterogeneous Parallel Programming” [5]. Thousands of students throughout the world have followed the course, and more importantly the course provides a custom cloud coding environment called WebGPU [2] that supports a web-interface for participants to access GPU nodes at the University of Illinois and the National Center for Supercomputing Applications (NCSA). WebGPU provides a code development interface, logging, and debugging for CUDA execution of GPU systems. There are numerous benefits to the WebGPU framework that other institutions could leverage,
however the development of the framework represents a substantial cost to many smaller education programs.

There are other custom cloud coding environments that support general computing frameworks. The following are some of the current cloud coding systems providing free or subscription models of Python and Linux environments:

- http://www.pythonanywhere.com
- http://www.wakari.io
- http://cloud.sagemath.com
- http://live.sympy.org
- http://c9.io

II. PYBox - CLOUD CODING CASE STUDY

One web-based framework that supports running a remote server accessible over web-browser systems is the IPython Notebook [1]. The IPython Notebook is a web-based interactive computational environment that in addition to providing a well-structured code development environment, also provides a framework for observing and recording and results of code execution, linking text such as comments, equation generators for mathematics, plots and other rich media formatting options. As the IPython environment is interactive, code cells, or small sections of code can be tested in-place without requiring the entire program example to be changed. The cloud coding advantage is that the IPython Notebook Viewer renders the code as a web page and users can read and interact with a remote system without having to install anything on their device. Figure 1 shows an example IPython Notebook with code and execution results.

![IPython Notebook example from www.ipython.org](image)

**Fig. 1.** IPython Notebook example from www.ipython.org

The Blackbox project [4], launched in 2013, collects large amounts of data about student programming activities in the BlueJ IDE. The Blackbox project includes data of Java source code as well as information about compilations, debugger usage and other compiler interactions. Based on that concept of tracking novice programmer behaviors and giving access to servers, we have constructed an IPython Notebook-based framework called PyBox. Figure 2 illustrates the framework built to allow students to interface to the computer vision OpenCV [3] library, a large image database, and NVIDIA GPU hardware. In this environment, the cloud coding solution is critical as virtualization support cannot overcome the limitations of student-owned resources not having GPU hardware.

![PyBox Notebook framework](image)

**Fig. 2.** The PyBox Notebook framework.

In addition to increasing accessibility to resources for students, the PyBox framework is designed to catalog certain programmer activities to help understand learning outcomes. Currently the activities tracked are: code development checkpoints, code executions, and lecture note execution tracking. The goal is to analyze the coding behaviors to determine the various pathways that students travel to deriving programming solutions. While deploying the PyBox framework in Spring 2014, we collected some initial usage results and plan to present at the EduHPC workshop.

III. CONCLUSIONS AND FUTURE WORK

The current goal is to integrate the PyBox framework into other core curriculum courses starting with the introduction to programming as well as open the use to other institutions. Cloud coding resources can provide critical benefits to smaller programs. The foundation of cloud coding creates an outreach opportunity to reach any level school, community center, or village with computers connected to the internet. The goal is to demonstrate the IPython framework, explore some of the multi-threading and GPU support, and introduce researchers to the PyBox framework for efforts in analyzing programming behaviors.

REFERENCES