Incorporating PDC Topics throughout the Undergraduate Computer Science Curriculum

Anu G. Bourgeois, Sushil Prasad, Ying Zhu, Jaman Bhola, and Yi Pan
Department of Computer Science
Georgia State University
Atlanta, GA USA
{abourgeois, sprasad, yzhu, jbhola, ypan}@cs.gsu.edu

Abstract – As part of the NSF/TCPP early adopter program, we incorporated various parallel and
distributed computing topics into a number of our courses in Fall 2011. The courses selected spanned
lower-level as well as higher-level undergraduate computer science classes. Depending upon the level
of the class, students were either introduced to a wide range of topics and expected to gain an understanding
of the concepts or were expected to apply the concepts within the framework of the particular class. We
selected classes that are required for all undergraduate computer science students, as well as a few senior
elective courses. We found that students enjoyed the extensions to the coursework and were able to
understand these topics and recognize its importance.

BACKGROUND
The requirements for an undergraduate degree in computer science at Georgia State University consist of
a series of required courses that all students must take as well a set of electives to choose from. The
required classes range from introductory classes to a set of senior-level classes that provide a foundation
for computer science topics.

Besides the required classes, our curriculum is broken into seven different concentrations that
students choose from. The set of elective classes, all of which are senior-level classes, are categorized to
fit within one or more concentrations. Depending upon the concentration selected, students must take a
number of classes within their chosen concentration to develop an in-depth understanding of that area and
then select two electives outside of their concentration to provide a breadth of understanding in computer
science topics.

While we regularly offer an elective class on parallel computing, there are many students that do
not get a chance to take this course. Besides this one class, there are classes in which the instructor may
choose to present PDC concepts in the class. However this is instructor specific as it is not currently built
into the curriculum. As a result, a number of students could potentially complete their degree without any
exposure to concepts of parallel and distributed computing. To prevent this from continuing, our faculty
participated in the early adopter program to find ways to revise our curriculum to impact all of our
students.

TARGETED COURSES AND PDC TOPICS
The faculty selected certain classes in which to introduce various PDC topics. The classes included lower-
level and higher-level classes, as well as required and elective courses. In this way we could ensure that a
majority of our current 400-plus undergraduate students would be exposed to these concepts. Once in
place, this will ensure all of our future students will understand and be able to apply PDC concepts
appropriately. The following classes were chosen:

CSc 3210 – Computer Organization and Programming (Anu Bourgeois)
CSc 3410 – Data Structures (Jaman Bhola)
CSc 4520 – Design and Analysis of Algorithms (Sushil Prasad)
CSc 4820 – Computer Graphics Algorithms (Ying Zhu)
CSc 4998 – Cloud Computing (Xiong Xue)
Below we describe the topics covered in each of the targeted classes. Evaluation of students’ understanding was administered through exams in each of the classes.

**CSc 3210** – This class is taught using the SPARC machine. A particular focus was given to how pipelining is used within the instruction cycle. Students were expected to trace through code to understand how multiple instructions are processed simultaneously as well as understand the impact on program flow. They were exposed to concepts of speed up for general cases of pipeline technology rather than just specifically on how it applies to the SPARC machine and its 4-stage pipeline.

**CSc 3410** – In this class, students learn basic concepts and analysis of data representation and associated algorithms. It teaches students how to approach various problems and that selecting efficient data structures is key to designing efficient algorithms. As this is a foundational course, we introduced the how’s and why’s of parallel computing and its importance. Students were also taught the concepts related to threading and applied this in a project to scale a vector. They were required to do this with and without the use of threading and were asked to compare the execution times. Their results clearly showed that the later was much faster than the former and that they were able to have even faster results with the more threads used.

**CSc 4520** – This class presents an overview of algorithmic concepts. Students are presented with algorithms on searching, sorting, selection, and graph problems including minimum spanning trees and shortest path problems, to name a few. Typically the students were presented these problems using approaches such as divide-and-conquer or greedy algorithms in a sequential fashion. Rather than solely present sequential approaches to the problems, the instructor described how to approach each new topic in a parallel fashion. Students also learned about the PRAM and studied basic problems using the PRAM as a computational model.

**CSc 4820** – In this course, the instructor discussed parallel processing architecture on GPUs, particularly the 3D pipeline and data parallelism. Students practiced GPU shader programming with OpenGL Shading Language. Related topics such as CUDA, OpenCL, and general purpose computing on GPUs were also discussed. By the end of the semester, students were familiar with the parallel processing techniques in the context of 3D computer graphics.

**CSc 4998** – This course introduces the fundamental principles, technology and current development of cloud computing. Topics include cloud computing architecture, green computing, smart computing, performance analysis, as well as sharing resource schemes and fault tolerant schemes. Students were asked to work in groups to write a research paper. Sample topics include an alternate scheduling paradigm for the Hadoop Framework (schedulers in distributed computing) and secure virtualization in cloud computing.

**FUTURE PLANS**

Our future plans include identifying additional PDC concepts to disseminate in our required classes (please refer to Appendix for a list of classes). Key will be to introduce an overview of the concepts and the basics of thread-level programming in our introductory courses and to build upon these at greater depths throughout the subsequent required courses. This will enable us to foster the importance of these concepts early on in the curriculum and develop the ability for students to apply their understanding towards efficient design and solutions. This will also enable us to elevate the PDC elective class with increasing rigor and breadth.