Incorporating PDC Modules into Computer Science Courses at Jackson State University

Ali Abu El Humos, Sungbum Hong, Jacqueline Jackson, Xuejun Liang, Tzusheng Pei and Bernard Aldrich
Department of Computer Science
Jackson State University
Jackson, MS 39217, USA
{ali.a.humos, sungbum.hong, jacqueline.m.jackson, xuejun.liang, tzusheng.pei}@jsums.edu,
bernard.m.aldrich@students.jsums.edu

Abstract— **The Computer Science Department at Jackson State University (JSU) is updating its curriculum according to the new ABET guidelines. As part of this effort, the computer science faculty members have integrated modules of the NSF/IEEE-TCPP Curriculum Initiative on PDC (Parallel and Distributed Computing) into department-wide core and elective courses offered on fall 2014. These courses are: csc 119 Object Oriented Programming (core), csc 216 Computer Architecture and Organization (core), csc 312 Advanced Computer Architecture (elective), csc 325 Operating Systems (core), csc 350 Organization of Programming Languages (core) and csc 425 Parallel Computing (elective). The inclusion of the PDC modules was gradual and light weighted in the low level courses and more aggressive in the high level courses. Csc 119 Object Oriented Programming provided students with an early introduction to Java Threads: how to create and use. In csc 216 Computer Architecture and Organization students learned about GPUs and were asked to write simple problems using CUDA. Csc 312 Advanced Computer Architecture covered Instruction level and Processor level Parallelism. For csc 325 Operating Systems, mutual exclusion problems and Parallel Computing and Algorithms were introduced. In csc 350 Organization of Programming Languages, students learned about the implementation of threads in Java. Csc 425 Parallel Computing is an advanced study of parallel computing hardware and software issues. Assessment results showed that student perception of PDC concepts was satisfactory with some weakness in writing parallel code. However, students were very excited and motivated to learn about PDC. We were also able to share our experience with the Computer Engineering Department at JSU. New PDC modules will be integrated into some of their courses next fall and spring semesters. Our findings were made available on the Center for Parallel and Distributed Computing Curriculum Development and Educational Resources (CDER) website. In this paper, we will describe our experience of incorporating PDC modules into the aforementioned computer science courses at JSU.**

Keywords-PDC modules; Computer Science; Jackson State University.

I. INTRODUCTION

Jackson State University (JSU) has a student population where over 90% are from under-represented groups. This NSF/IEEE-TCPP Curriculum Initiative [1] award therefore had a direct impact on minorities, specifically in the Computer Science field. Not only did this award help integrate PDC topics into the computer science curriculum at JSU, but also it better prepared our graduates for their future careers where PDC knowledge is a must to know.

The computer science new curriculum at JSU is a 125 credit-hour program in which there are 57 in computer science. Some advanced computer architecture topics are moved to elective courses. If some students choose not to take these elective courses, this may prevent them from learning about important PDC topics such as pipelining, superscalar architectures, multiprocessors and multi-core processors and GPU programming. Consequently, it is crucial to revisit the contents of every offered computer science course in order to smoothly integrate PDC topics into these courses.

II. EARLY ADOPTING COURSES

In order to make sure that our students will be exposed to various PDC topics, five computer science professors participating in the Early Adopting Program updated their courses with different PDC modules. These courses were:

CSC 119 Object Oriented Programming: This course covers problem solving methods and algorithm development; definition of language syntax and semantics of JAVA; and developing the ability to design, code, debug, document and successfully execute programs. This course builds upon the topics of CSC 118 (Fundamentals of Computer Science) and covers inheritance, polymorphism, interfaces, exception handling, streams and file input/output, recursion, dynamic data structures (linked lists, stacks, queues, hash tables, graphs, trees) and associated algorithms. Lecture topics were reinforced weekly with laboratory assignments. To support parallel computing, Java has the Thread class and the Runnable interface, and it also provides rich primitives with the java.util.concurrent packages, which include the fork/join framework. Students explored these features in Java for parallel computing.

CSC 216 Computer Architecture and Organization: This course covers the basic concepts of computer architecture which includes machine level representations of data, computer arithmetic, instruction set architecture and assembly language, datapath and control, memory system, and bus architectures and I/O devices. A new PDC module was added to this course to introduce students to multi-core processors and GPU hardware. Also, throughout the course, parallelism at different levels was discussed.

CSC 312 Advanced Computer Architecture: This course is becoming an elective course under the new curriculum. It covers various advanced topics of the PDC curriculum such as instruction level parallelism: pipelining and superscalar architectures, processor level parallelism: array processors, multi-processor and multi-computer systems. Techniques to reduce instruction pipeline stalls and set associative caches are analyzed. Quantitative approaches of computer performance are emphasized. A new PDC module was added to this course to introduce students to benchmarks and how they can be used to compare the performance of various parallel systems.

CSC 325 Operating Systems: This course introduces the major concepts of process communication and synchronization, protection, performance measurement, and causes and evaluations of the problems associated with mutual exclusions and process synchronization among concurrent processes. It also introduces and analyzes various operating systems in terms of processor, memory, device, information, and distributed systems management. A PDC module was incorporated into this course to extend process synchronization issues to parallel programming concepts. With this module, the course provided students with parallel thread programming opportunities.

CSC 350 Organization of Programming Languages: This course is a study of the organization and specification of programming languages. It covers several issues in language design, including typing regimens, data structure models, control structure models, abstraction, virtual machines, language translation, interpreters, compiler design, lexical analysis, parsing, symbol tables, declaration and storage management, code generation and optimization techniques. In this course, after a brief review of the features in Java for supporting parallel computing (taught in CSC 119), parallel programming assignments were given for gaining hands-on experience. Generic concepts in parallel computing were also introduced.

CSC 425 Parallel Computing: This is a newly developed elective. It is a study of the hardware and software issues in parallel computing. It is a theoretical and practical survey of parallel processing, including a discussion of parallel architectures, parallel programming languages, and parallel algorithms. Students will write programs for multiple parallel platforms in a higher-level parallel language. From this course, the students will learn how to write parallel programs on three different parallel architectures: i) shared memory model- thread programming; ii) Cluster- Message Passing Computing; and iii) Multicore- GPU Programming. Due to low enrollment, this course was not offered in Fall 2014 and will be offered again in Fall 2015.

III. EVALUATION

In order to get initial feedback from students, we conducted a survey early in the fall semester to ask undergraduate students (Seniors, Juniors, Sophomores and Transfer students from local community colleges) about their knowledge and interest of PDC. Table I is a summary of the survey questions and results where a score of 4 is excellent and a score of 1 is poor. The survey results reflect the dire need to emphasize PDC topics across the computer science curriculum. It also shows that our students are motivated to learn PDC topics. These facts are also reflected in the following student comments reported in the survey:

Table I. Survey Questions and Results

Question	**Average Score**
Please, rate your current knowledge of PDC.	1.4
Please, rate the breadth of PDC topics covered in the computer science curriculum at JSU	1.4
Please, rate the depth of PDC topics covered in the computer science curriculum at JSU.	1.5
Please, rate your overall learning experience of PDC at the computer science department at JSU.	1.5
Will you be interested in pursuing a career that requires PDC knowledge?	2.5
Will you be interested in registering for Advanced PDC classes if offered during fall or spring semesters?	3.0

- “I do not have knowledge of PDC. It would be helpful if we learned about this topic in our classes. This could help with strengthening our programming skills. Activities or projects during class would be helpful”.
- “I hope that there would be mentoring sessions with faculty to help students address the issue of being able to learn on our own different languages and principles of Computer Science”.
- “One could provide real world Applications of PDC. If it is implemented in classes, make sure that you provide real world application as well as theory. The combination of both solidifies its importance and builds interest”.
- “I think it is very relevant, but at the same time, I do not know anything about it. I just know that our curriculum is already filled with classes, and it is tough trying to manage those classes”.
- “The topic should be touched in classes at every level. For a topic like this, the students should start learning about it as freshmen so that they can increase understanding of it over time”.
- “PDC careers should be discussed more and more. More hands on activities”.
- “We really did not talk about PDC in my undergraduate program. I would love to see it offered for students”.

At the end of the fall semester, we assessed our student perception of PDC topics. It is included in the Faculty Course Assessment Report (FCAR) submitted every semester to the department required for ABET program assessment.

Table II. Assessment Results

Course Name	[EX, EF, M, U] Vector	Weighted Average
CSC 119 Object Oriented Programming	[3, 0, 0, 1]	2.87
CSC 216 Computer Architecture and Organization	[3, 0, 1, 1]	3.00
CSC 312 Advanced Computer Architecture	[2, 4, 3, 1]	2.70
CSC 325 Operating Systems	[8, 5, 1, 24]	1.92
CSC 350 Organization of Programming Languages	[2, 1, 1, 0]	3.25

Table III. PDC Topics used for Assessment

Course Name	PDC Topics
CSC 119 Object Oriented Programming	Concurrent activities in Java, Java thread.
CSC 216 Computer Architecture and Organization	GPU Architecture and Computing, Simple CUDA programs.
CSC 312 Advanced Computer Architecture	Flynn's Taxonomy, Instruction Level and Processor Level Parallelism, Benchmarks.
CSC 325 Operating Systems	Algorithms for mutual exclusion, Parallel Computing and Algorithms.
CSC 350 Organization of Programming Languages	Understanding fundamental concepts in threads, Reading programs with threads.

For each assessment tool, every instructor calculated the percentage score of the students and categorized them into the following categories based on the percentage scores: Excellent (EX – score $\geqslant$ 75%), Efficient (EF – score $\geqslant$ 50% and < 75%), Minimal (M – score $\geqslant$ 25% and < 50%) and Unsatisfactory (U – score < 25%). The average score (ranges from 1 to 4) for each Course Outcome is the weighted average of the values for its [EX, EF, M, U] vector, with the weights being 4 for EX, 3 for EF, 2 for M and 1 for U. Table II shows the assessment results for the five classes. Table III summarizes the PDC topics used for assessment in these courses. For specific questions and assignments, please check the CDER website for these courses [2].

In general, students' perception of PDC topics was good except for those covered in the Operating Systems class which has a large student enrollment since both computer science and engineering students take this class together. All instructors indicated that most students were very interested and motivated to learn PDC topics. Some instructors were short on time to cover more PDC topics, and will figure out some ways to accommodate more time to cover these topics in future offerings of these courses.

IV. CONCLUSIONS AND FUTURE WORK

PDC modules were implemented in the aforementioned courses. A presentation about PDC education for ACM computer science students was organized early in the fall semester. Students shared their views on how to integrate PDC into the computer science curriculum and had constructive feedback from students and faculty. Two graduate students were motivated about PDC education and attended the EduHPC 14 workshop, where we presented some of our early experience of PDC education at JSU [3]. Assessment data was collected at the end of the fall semester. The data showed that students were comfortably able to learn PDC concepts and were motivated by these topics to pursue a career or do research in the area of PDC. To support our students with adequate PDC resources, a Tesla K40C, GPU hardware granted by NVIDIA Inc. [4] has been added to the Distributed Computing Laboratory. The GPU contains 2880 CUDA cores with 12 Giga Bytes of memory on the Tesla K40 and 400 GB HDD and another 192 CUDA cores on a Quadro 2000 GPU card. Combined, they provide adequate power to support simulations requiring high-power computing capacity. Matlab [5] Parallel Computing Tool box will soon be available for our students. Internal university funding is also sought to continue this project in the coming semesters.

REFERENCES

[1] "NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing – Core Topics for Undergraduates." [Online]. Available at: http://www.cs.gsu.edu/~tcpp/curriculum/.

[2] http://www.cs.gsu.edu/~tcpp/curriculum/?q=node/21183.

[3] A. Abu El Humos, S. Hong, J. Jackson, X. Liang and T. Pei, "NSF/TCPP Early Adopter Experience at Jackson State University Department of Computer Science". Workshop on Education for High-Performance Computing EduHPC-14, in conjunction with SC-14: The International Conference for High Performance Computing, Networking, Storage, and Analysis, New Orleans, LA, November 16-21, 2014.

[4] http://www.nvidia.com.

[5] http://www.mathworks.com.