NSF/TCPP Early Adopter Fall-13 Proposal for Department wide PDC Adoption

Sheikh Ghafoor, Mike Rogers {sghafoor,mrogers@tntech.edu}
Department of Computer Science
Tennessee Technological University, Cookeville, TN 38505

Abstract

Category of Proposal: Department wide
List of Course: CSC 1200 - Principle of Computing (CS0), CSC -2100 - Introduction to Problem solving and Computer Programming(CS1), CSC2110 - Data Structures and Algorithm (CS2), CSC 4100 - Operating Systems, CSC 4320 - Computer Architecture, CSC 4760 - Parallel Programming.

The widespread deployments of multicore and GPU based systems in recent years have changed the computing landscape. However, most undergraduate computer science (CS) program does not teach multicore and parallel computing concepts, and CS undergraduates are typically exclusively trained to think and program sequentially. This proposal plans to addresses the rapidly widening gap between highly parallel computer architectures and the sequential programming approach taught in traditional CS courses by systematically integrating parallel computing into current undergraduate curricula in the Computer Science Department at Tennessee Technological University. Specifically the project plans to integrate elements of the TCPP curriculum into multiple undergraduate CS classes over multiple semesters starting from fall 2013.

The overall goal of this project is to enable CS under graduate students to be prepared for their future careers in light of the technological shifts towards parallelism through multicores, GPUs, and corresponding software environments by gradual incorporation of topics of TCPP curriculum in undergraduate CS classes. Specifically this project plans to: 1) Redesign selected undergraduate classes. 2) Implement the redesigned classes gradually over multiple semesters. 3) Evaluate and revise the implementation, and 4) gradually integrate complete the TCPP curriculum in undergraduate CS curriculum.

Department Back Ground

Tennessee Technological University is a medium size public four-year engineering college with approximately twelve thousand students. The computer science department has nine full time and several adjunct and part time faculty members. There are roughly three hundred and fifty majors in CS department. The department offers a traditional computer science bachelor's degree focusing on software engineering that is ABET accredited, and offers a bachelor's degree in information technology. In addition, the department offers a master's degree in computer science, and a PhD degree through the College of Engineering.
Our curriculum covers a broad range of computer science courses, as do other universities, but it also has some distinguishing features. Our introductory sequence is a three course programming classes consisting of *Introduction to Problem solving and Computer Programming*, *Data Structures and Algorithms*, and *Object Oriented Programming and Design*. We require a one-hour, hands-on lab for each programming course in the sequence. We teach these introductory courses every semester. In the fall 2013 semester we are adding a required *principle of computing* course. We also include *Design of Algorithms* as a required course. Required upper division course are typically offered once each year. Our required upper division courses for the traditional computer science degree include *Operating Systems*, *Networking*, *Computer Architecture*, *Database Systems*, and a two-semester software engineering series. The department is introducing *Parallel Programming* as an upper division computer science elective.

Project Plan

To achieve the goal of the project we plan to introduce PDC topics in the TCPP curriculum into multiple relevant CS courses over multiple semesters. In four years we plan to gradually implement the complete TCPP curriculum.

We will redesign the existing classes to incorporate PDC concepts and topics. Our redesign effort will have three components: 1) Change the approach of teaching existing traditional topics to encompass the opportunities for "thinking in parallel". 2) Adopt relevant PDC modules/materials that has been already developed by others educators/researchers in similar effort and revise them as needed. We plan to use resources developed other early adopters and NSF funded projects. 3) Develop new appropriate and relevant PDC modules/materials as deemed necessary.

These three components of our redesign effort can be used to both create new courses and update existing courses. Recently the CS department has started a discussion of integrating parallelism in our CS undergraduate curriculum. As a first step in achieving this goal, the CS department is offering a new senior level elective stand-alone parallel programing course in fall 2013 that we will continue to offer in subsequent semesters.

The integration of PDC topics in existing courses will be an iterative process. We will redesign a class and will teach it the following semester. We will conduct an evaluation of the redesign and implementation. Then, based on the evaluation, we will modify the redesign and teach it again. We plan to adopt this process for all relevant courses over multiple semesters. The following table shows the project implementation plan:

Activity	**Semester**
Redesign CSC 1200, CSC2100, CSC 4100	Fall 2013
Implement CSC 4760	Fall 2013
Implement CSC1200, CSC 2100, CSC 4100	Spring 2014
Evaluation of implementations	Spring 2013
Redesign of CSC 2110	Spring 2013
Revisit the redesign CSC 1200, CSC2100, CSC 4100 based on evaluation	Summer 2014
Implement CSC 1200, CSC 2100, CSC 2110	Fall 2014
Evaluation of implementations	Fall 2014
Redesign CSC 4320	Fall 2014
Implementation of CSC 4200, 4320, CSC 1200, CSC 2100, CSC 2110, CSC 4100	Spring 2015
Evaluation of implementation	Spring 2015
Redesign of other CS courses (Networks, Database, upper/lower division algorithm, software engineering and continue to teach redesigned courses	Beyond Spring 2015

We plan to share PDC modules, course templates developed as part of this project in courseware website, and CS in parallel website. We have a plan to develop a project web site where we will publish all PDC materials developed as part of this project. Through that web site we will also share the project evaluation data, reports, and publications.

Course Details

The courses that we intend to revise to incorporate the TCPP curriculum are *Principles of Computing*, *Introduction to Problem Solving and Computer Programming*, *Data Structures and Algorithms*, *Operating Systems*, and *Computer Architecture*. We also include a description of our new *Parallel Programming* course below.

CSC 1200 – Principles of Computing

This course gives a broad overview of computer science and focuses on introducing the basic terms and principle of computer science. Topics include computing as a creative activity, abstraction, data and information, algorithms, programming, the Internet, and global impacts of computing. Given the introductory nature of this course, it provides us with ample opportunity to introduce some of the basic concepts and terms of PDC, such as concurrency, parallelism, and distributed computing. We intend to add case studies to show how students use the basic concepts of distributed computing and parallelism in their every-day lives via web browsing, smartphone apps, etc.

CSC 2100 – Introduction to Problem solving and Computer Programming

This is the first course in our introductory programming sequence with 3 credit hours of lecture and 1 credit hour of lab. This course introduces students to digital computers, the procedural approach to programming using C++, principles of good program design, problem solving techniques and the process of devising a computer solution from a problem statement. This course covers many topics beginning with describing computers and how they work and ending with structures and pointers in C++. This course assumes students have little to no background in

computer systems. Because of a high failure rate, the department is introducing a required Principles of Computing (CS0) class. Some of the introductory material of this course will move to CS0 clearing space for PDC topics. Some the PDC topics that will be covered in this class are concurrency and parallelism, decomposition, data and task parallelism, divide-and-conquer, speedup, scalability. Some of these concepts will be repetition from CS0. More than one thread based programming assignments will be given to students.

CSC 2110 – Data Structures/Algorithms
Topics of this course include introduction to software engineering, abstract data types, object oriented programming, lists, stacks and queues, recursion, binary search trees and heaps, sorting algorithms, and searching algorithms. This course is the second course in our programming sequence. It consists of 3 credit hours of lecture and 1 credit hour of lab. Some the PDC topics such as decomposition, data and task parallelism, divide-and-conquer, speedup, scalability which have been introduced in CS1 will be repeated in this course. This core course is an ideal course to introduce algorithmic PDC terminology, and to add parallel components to existing topics. For example, we will present material that describes how parallelism can be applied to inherently recursive data structures, and we can introduce language extensions, libraries, and compiler directives to help write parallel code.

CSC 4100 – Operating Systems
This is a classical operating system course covering concurrency, non-determinism, processes, threads, scheduling, synchronization, memory management, locality of reference, file systems, I/O, protection and security. The course is pragmatic, attempting to merge theory and principle with practice. Therefore, the course programming projects consists of implementing a small bootable OS on the Intel architecture. When the operating system is completed, it has the following features: memory organization, preemptive multitasking, and synchronization via mutexes and semaphores. For the last assignment, students not only implement mutexes and semaphores, but apply them by implementing a producer/consumer application. Though the course topics and assignments introduce and practice the concepts of concurrency, the focus of the topics and assignments is on single core environments. As a TCPP early adopter, we will integrate material about multi-core environments into both the course topics and the assignments. We will focus on SMP architectures because that architecture is most prevalent for general-purpose operating systems. Examples of included material are operating system organization, scheduling, and implementing synchronization primitives in an SMP environment, and case studies for popular operating systems.

CSC 4760 – Parallel Programming
The department offers several parallel and distributed computing courses at the MS and PhD level. As a first step to introduce parallelism in undergraduate courses, this course is going to be offered for the first time for undergraduates in fall 2013. This is broad course covering the foundations of parallel computing, including the parallel computer architectures, principles of parallel algorithm design, programming models for shared and distributed memory systems, along with numerical and non-numerical algorithms for parallel systems. The course will include material on emerging multicore hardware, shared-memory programming tools (Intel Thread Building Blocks, Cilk etc.), and programming models for GPGPUs (CUDA). A key aim of the course is to give students hands-on knowledge of the fundamentals of parallel programming by writing efficient parallel programs in some of the programming models that they learn in class. The course includes many short lab assignments using OpenMP, Cilk plus, CUDA, and MPI. The course covers many of the TCPP topics in all four Areas. As the integration of PDC topics in earlier CS classes happens over time this class will evolve accordingly, as some of the materials would have been covered in multiple classes at different level of abstraction and details.

CSC 4320 – Computer Architecture
This is a classical computer architecture course covering the von Neumann architecture, the instruction cycle and interrupts, bus design, memory hierarchy, i/o methods and DMA, instruction formats and addressing modes, pipeline operation and branch prediction, and RISC architectures. We plan to integrate multicore and multithreaded material into existing topics, as well as add some new PDC topics that are not currently taught in the course. Examples of topics that we intend to integrate into the course include Flynn's taxonomy, data and task parallelism, multicore and cluster architectures, stream-based architectures and GPU computing, and performance issues, such as latency and bandwidth issues.

Evaluation Plan

Evaluation will be an important part of the project. The goals of our evaluation will be to measure: a) how the redesigned classes impact student ability to think effectively using parallel concepts and gained knowledge in PDC topics. B) Efforts required by the faculty to implement the redesigned class. The pre and post surveys will also be designed to gather feedback from faculty to aid in the redesign effort.

Evaluation methods: For each redesigned class there will be student's pre/post survey to measure the knowledge gain in PDC concepts and topics. The pre and post surveys are subjective measurements. For objective measurements of knowledge transfer, assessment of individual courses will be done through lab assignments, exams quizzes etc. Both PIs of this proposed project are currently involved (as PI and senior personnel) in another similar NSF funded TUES project (**SecKnitKit: Integrating Security into Traditional Computer Science Courses)** which aims to develop instructional material and a support system for non-security faculty to integrate security modules into traditional required computer science courses. The project has successfully incorporated security topics in four upper division CS courses (Computer Networks, Database Management, Software Engineering, and Operating Systems). As part of that project we have developed (with external evaluators) evaluation instruments (pre/post surveys for both students and faculty implementers) to measure the impact of the Project. The proposed project has lot of similarities with SecKnitKit project: in the proposed project we are trying incorporate PDC concept and topics into existing CS courses. For the proposed project we plan to heavily leverage the already developed and validated instruments from the SecKnitKit Project. We will modify the instruments to incorporate PDC topics and concepts instead of security topics. The evaluation data will be analyzed and will be used for the redesign. We plan to share our evaluation results with the community through publications and presentation in conferences and meetings such as SIGCSE, EduPar etc and through project web site.

Budget

We are requesting for US $ 2500, which will be spent for faculty support for redesign and implementation of their respective courses. The Computer Science Department currently has access to a high end 48 core SMP machine with multiple tesla GPU boards and an old 32 node Beowulf style cluster. These machines are primarily used by graduate students and faculty researchers. A low end multiple processor machine with a GPU would be very useful for teaching the undergraduate PDC curriculum. One such machine is Little Fe (http://littlefe.net/buildout), which is 4 node (8 core) Beowulf style portable computational cluster with GPU chipset on each node. Little Fe costs about US$2800. The Department of Computer Science agreed to match the TCPP grant (if successful) up to US$3000 towards the procurement of hardware for teaching PDC.

Faculty Bio

Sheikh Ghafoor is an Associate Professor in the Department of Computer Science at Tennessee Technological University. He joined the department in 2008. His primary research includes Parallel, Distributed Computing, and High Performance Computing. His current research is in autonomic resource management for high performance computing environment, programming model for parallel adaptive applications, and fault tolerant computing. Dr. Ghafoor is also very interested, and actively engaged in research in the area of computer science and engineering education. Dr. Ghafoor has been principal investigator on grants from NSF and DOE. Dr. Ghafoor regularly teaches CS1, CS2, Computer Networks class in undergraduate level and parallel and distributed computing related classes in graduate level. He will teach the undergraduate parallel programming class in fall 2013.

Mike Rogers is an Associate Professor in the department on Computer Science at Tennessee Technological University. He joined the department in 2002. He is an active researcher, and his interests include parallel and distributed computing and web based protocols. He regularly teaches *Introduction to Problem Solving*, *Operating Systems, Database Systems*, and well as various distributed computing graduate level courses.

Sushil Prashad
NSF/TCPP Curriculum Working Group

Subject: Support for NSF/TCPP Early Adopter Fall-13 Proposal for Department wide PDC Adoption

Dear TCPP,

As the Chairman of the Computer Science (CS) Department at Tennessee Technological University, I give my full support and recommendation for the above mentioned proposal by co-PIs, Dr. Sheikh Ghafoor and Dr. Mike Rogers. The CS department fully supports the effort to integrate Parallel and Distributed Computing (PDC) topics in relevant undergraduate classes.

The widespread deployment of multicore and GPU based systems such as computers and smartphones over the last decade has drastically changed the software and hardware landscape. Given this circumstance, our department is discussing ways to incorporate PDC concepts and topics throughout our undergraduate curriculum. I welcome and support the effort by Drs. Rogers and Ghafoor. If the proposal is successful the CS department will match the grant up to US $3000 towards the procurement of hardware for teaching PDC at the undergraduate level. I give my complete support and commitment to contribute to the success of this project.

Regards,

Douglas Talbert
Chairman
Dept. of Computer Science
Tennessee Tech University
Cookeville, TN 38506
dtalbert@tntech.edu