NSF/TCPP Early Adopter: Introducing Parallel and Distributed Computing to Minority Undergraduates

Lei Huang and Yonggao Yang
Department of Computer Science
Prairie View A&M University, Texas, USA 77446
Email: {lhuang,yoyong}@pvamu.edu

Abstract—This paper describes our ongoing work on introducing parallel and distributed computing concepts to minority undergraduates. We report our experience and feedback received in introducing these concepts in Spring 2012, and present our future plan to further integrate these concepts into additional Computer Science courses.

I. INTRODUCTION

Since Spring 2012, the Computer Science Department at Prairie View A&M University started updating the course inventory and program curriculum to include more emerging technologies to undergraduate students. We have adopted the parallel computing concepts in three existing Computer Science core courses, and are currently working on creating a new course “Introduction to Parallel and Distributed Programming” in the undergraduate curriculum.

Prairie View A&M University (PVAMU) is the second oldest public institution of higher learning in Texas, and is one of Historical Black College and Universities (HBCU). With an established reputation for producing engineers, nurses, and educators, PVAMU offers baccalaureate degrees in 50 academic majors, 37 master degrees and four doctoral degree programs through nine colleges and schools. The university is accredited by the Southern Association of Colleges and Schools as a comprehensive public institution of higher education. The total Fall 2009 enrollment was 8,577 students with 90% of the student population consisting of African-Americans. The university has established a reputation as one of the nation’s top 10 producers of African-American engineers. The College of Engineering (CoE) at PVAMU consists of six departments and seven programs accredited by the Accreditation Board for Engineering and Technology (ABET).

Computer Science department is under the college of engineering with faculty from diverse backgrounds. Within a minority-serving university, the department conducts abundant research to improve the STEM graduation and retention rate. It is crucial for the department to continue to provide the state-of-the-art knowledge and excellent research matriculate opportunities to students and expose them to the latest technology development. The department is currently in the process of updating course inventory and program curriculums for the CS undergraduate, and graduate programs (MSCS and MSCIS). The purpose of this update is to remove the redundant and obsolete courses, and add new courses covering the emerging technologies that will allow our graduates stand out from the high competitive job market. Parallel Computing is one of courses we are looking for to add to the Computer Science curriculum, and we believe that the course will play an important role to meet the industry’s requirements for Computer Science graduates.

II. THE STATUS OF TEACHING PARALLEL AND DISTRIBUTED COMPUTING

In Spring 2012, we have revised syllabi of Computer Science II (COMP 1224), Algorithm Analysis (COMP 3053), and Operating System (COMP 3063) by integrating TCPP concepts. TCPP lectures have been integrated into the related chapters in the three courses. We have collected the student survey data to be evaluate the student’s feedback. The new course “Introduction to Parallel Computing” has been submitted to the curriculum committee and the teaching materials are under development.

A. Adopt TCPP topics in Computer Science II (COMP 1224 Core Course)

“Computer Science II” (COMP 1224) is a 4 credit hours class with 3 hours for teaching and 1 hour for computer laboratory. It is continuation of CS1 “Computer Science I” with continued emphasis on program development techniques, array based lists, pointers, basic linked lists, classes, abstraction, data hiding, polymorphism, inheritance, stacks and queues. There are 26 students in Spring 2012.

TCPP Integration: We have introduced the basic parallelism concepts to students in this class, including threads, data sharing, synchronization, and thinking problem solving in parallel. In particular, we have introduced OpenMP to students and it is much easier for them to understand it. Figure 1 shows an OpenMP example to calculate $\pi$. It stimulates students’ interests when they found the performance can be doubled or 4 times in their laptops by adding a single line of OpenMP directive (line #11).

Feedback: From the survey and test results, we have found that it is really a good idea to introduce the parallel computing concepts to students at the early stage. Students have shown enthusiasm in studying parallel programming skills after they found the significant difference in performance, compared with sequential code. OpenMP is much easier to understand and can be used shortly after introducing the syntax. However, multithreading is confusing to most of them.
static long num_steps = 1000000000;
doUBLE step;
int main ()
{
  int i;
double x, pi, sum = 0.0;
doUBLE dif;
time_t start, end;
time(&start);
step = 1.0/(DOUBLE) num_steps;
#pragma omp parallel for private(x) reduction(+:sum)
for (i=0;i<= num_steps; i++)
{
  x = (i+0.5)*step;
  sum = sum + 4.0/(1.0+x*x);
}
pi = step * sum;
time(&end);
dif = (DOUBLE) diff time (end,start);
cout << "The value of PI is " << pi << endl;
printf("It took %.2lf seconds.", dif);
return 0;
}

Fig. 1. OpenMP demo to show the performance difference of calculating π

B. Adopt TCPP topics in Algorithm Analysis (COMP 3053 Core Course)

“Algorithm Analysis” (COMP 3053) This course is a very popular course in computer science curriculum. Students in this class found the interests of programming and developed problem solving skills. There are 16 students in Spring 2012.

TCPP Integration: we have added parallel algorithms including parallel sort and parallel search. We let students to follow an algorithm and then let them think how to go parallel if he/she works in a team. We then compare the parallel and sequential algorithms, introduces speedup concept, discusses if/how these classical sequential algorithms can go parallel, and debates whether an algorithm is efficient or not in parallel architectures. The goal of these adjustments will let students take the parallelism into consideration when design an algorithm in solving a problem.

Feedback: Compared with CS II students, the introduction of parallel programming becomes more challenging since more students in this class get used to sequential programming. The comparisons of sequential and parallel algorithms help a lot for them to understand the parallel concepts. Students are more interested and clear to understand parallel algorithms when they work in a team to mimic the parallel process. More practices of using multithreading and MPI are needed.

C. Adopt TCPP topics in Operating System (COMP 3063 Core Course)

“Operating System” (COMP 3063) is one of core courses in Computer Science too. We have revised the syllabus to emphasize the parallel concepts, multithreading scheduling and load balance. There are 10 students in Spring 2012.

TCPP Integration: Besides introducing the concepts such as multicore architectures, multithreading and synchronizations, we have focused on introducing the threads scheduling and load balancing under multicore architectures. OpenMP is introduced too to let them have a clear idea on how to write a parallel program.

Feedback: The introduction of thread scheduling and load balancing help students understand better how OS manage tasks. Distributed computing has not been introduced yet. A class project for thread scheduling would help students understand better the modern architectures and load balancing strategies.

III. What’s Next

A. New Course Plan

We are currently working on the plan to introduce a new course “Introduction to Parallel and Distributed Computing” in the curriculum of Computer Science in 2012. The change will not only introduce parallel computing to undergraduates, but also have positive impacts to attract more students in studying Computer Science in MSCS and MSCIS programs. The course will be a 3 credit hours course offered to junior and senior undergraduates. It will introduce the parallel architectures, the concepts of parallelism and applications, the parallel programming models, and parallel programming practices. The parallel and distributed computing concept is also proposed to be embedded to the following computer programming related core courses in the current CS curriculum.

B. Other Courses Integration Plan

In this semester, we first make adjustments for the existing three courses, and we will gradually introduce these topics in other classes, including programming languages, computer architectures, and network. The department will conduct surveys to get feedbacks from the students to understand the difficulties and adjust them accordingly. We also plan to evaluate the impacts of these curriculum adjustments by tracking students’ career and conducting surveys with their employers after they graduated.

IV. Conclusion

Based on our experience in the Spring 2012, the TCPP introduction to undergraduates is not only feasible, but also is promising. We have observed the interests of students in learning the new concepts and found out the difficulties of teaching them. From our feedback, we found it is necessary to introduce parallel concepts at a very early stage of teaching how to program. The concepts should be over and over strengthened in most computer science courses, if not all. Moreover, a dedicated parallel computing course should be added into the undergraduate curriculum for junior to senior students. The experience is very useful and we will continue to integrate the knowledge to other classes. We are also sharing the teaching materials to other faculty members.

ACKNOWLEDGMENT

We gratefully thank to the NSF/TCPP Core Curriculum Early Adopter Group to support the curriculum changes at PVAMU, and provide the traveling funds to present our work at the EduPar-12 conference.