Identifying the Challenges and Strategies of Teaching Parallel Programming to Computational Science Undergraduates: Experiences and Statistics

Bhaskar Chaudhury
Group in Computational Science and HPC, DA-IICT, Gandhinagar, India - 382007
e-mail: bhaskar_chaudhury@daiict.ac.in

***Abstract* – This article presents our experiences with teaching HPC/PDC topics to undergraduate students and my thoughts on course design philosophy, meaningful evaluations and course evolution. A brief summary of challenges faced, lessons learned, tools developed and success stories supported by data and statistics are also presented.**

Our three years of experience with teaching a High Performance Computing (HPC) core course (CS301, 3 hrs. of lecture & 3 hrs. of lab per week, total ~ 15 weeks) to third year Computational Science undergraduate students at DAIICT, India made us realize that imparting effective Parallel and Distributed Computing (PDC) education within a limited time-frame (e.g. one core course or an elective course) is a challenging task because it involves teaching both the software as well as the hardware aspects of HPC. Students must understand that theory of parallel algorithm design is based on abstract concepts of time and memory which may ignore real life constraints for simplicity, and therefore not take into account non-deterministic and hardware factors. It is important for a student to realize that computing systems are in general non-terminating and non-deterministic, whereas the behavior of algorithms is terminating, deterministic and platform-independent. The main challenge is to make sure that after successfully completing an HPC/PDC focused course, a student understands the practicalities of doing HPC to achieve the maximum possible performance out of a particular system for a particular problem [1].

The CS301 course has been designed by following the recommendations of NSF/IEEE-TCPP Curriculum Initiative on PDC [2], and keeping in mind the technical background of our students and program objectives. The course covers important PDC topics such as modern processors, parallel architecture, basics of code optimization, design of parallel algorithms, shared & distributed memory based parallel programming and performance measurements/ analysis [1]. These topics supported by examples/ case-studies (computational science focused) are imparted in a particular order which helps students to make gradual transition from serial computing to parallel computing and thereby acquire the ability to "think in parallel". Students implement the case-studies discussed during the lectures as assignments in the lab using OpenMP in the first three months and finally using MPI in the later part of the course [1]. The course evaluation is based on 2 mid-semester exams (60 %), and 6 lab assignments & a compulsory 1-month project (40%). The heavy focus on practical programming strengthens students' design and implementation skills.

Course Evaluation by Students: In 2015 only one general course evaluation survey was carried out and we realized that we require more course specific surveys to specifically understand the challenges faced by the students and evolve the course content/ assignments. From 2016 onward, we started carrying out 2 more course surveys – some important statistics of these surveys are presented in the Appendix (Table – 1, 2, 3). Students primarily faced challenges in co-relating all the deterministic and non-deterministic factors which affect performance, difficulty in data collection for performance analysis, self-evaluation while completing the assignments, time constraints in preparing lab reports etc. Based on the survey results of 2016, the course delivery in 2017 was changed by reshuffling the order of lectures/ modules and focusing more on difficult concepts so that students can easily correlate the theory and the lab part.

Tool development: To address the important challenges, an online open-access web-based platform (www.letshpc.org) is being developed to help streamline the process of analyzing parallel programs, self-learning, teaching, evaluation, discussions, understanding numerous deterministic/ non-deterministic factors of both the software and the hardware etc. [1]. Scripts are provided for automatically collecting performance related data, which can then be analyzed using the platform's tools. The platform also allows students to prepare a standard lab/project report aiding the instructor in uniform evaluation. This platform was extensively used by the students in 2017 to complete their course assignments/ projects and subsequently a comprehensive evaluation of this tool was carried out. Results of survey and important features of this tool will be presented in the poster.

Many students (after completing the CS301 course) getting involved in complex HPC based research projects in their fourth year is a very encouraging trend; several of these students won parallel programming challenges at HiPC-2015, 2016, presented research papers at HiPC-2016, 2017, and SC17 conference, and also contributed to journal publications.

[1] B. Chaudhury et al. "Let's HPC: A web-based platform to aid Parallel, Distributed and High Performance Computing Education," Journal of Parallel & Distributed Computing, March 2018 (press) (doi.org/10.1016/j.jpdc.2018.03.001).

[2] S. K. Prasad et al. "NSF/IEEE-TCPP Curriculum Initiative on PDC - Core Topics for Undergraduates, Version I (2012). www.cs.gsu.edu/~tcpp/curriculum

Appendix

Table-1

Sr #	**Questions** (responses collected on a 10 point scale)	2016 Survey 1		2017 Survey 1		2017 Survey 2	
		P1	**P2**	**P1**	**P2**	**P1**	**P2**
1	How familiar are you about High Performance Computing concepts	2.48	6.89	1.98	2.1	7.78	84.9
2	How much do you consider the impact of the architecture of the computer on the performance of your programs	7.53	89.65	4.83	41.7	8.36	88.7
3	How much do you consider the impact of the memory hierarchy related factors of the computer on the performance of your programs	NA	NA	4.8	41.8	8.26	86.8
4	How important should the theory part in this course be?	6.0	53.4	6.85	77	7.46	79.2
5	How important should the lab part in this course be?	8.1	89.5	8.45	89.6	8.48	86.9
6	How difficult do you think it is to understand and analyze the factors affecting the performance of your program?	NA	NA	7.78	91.8	7.48	81.1
7	How difficult do you think it is to measure the factors affecting the performance of your program?	NA	NA	7.91	93.7	7.51	83.1
8	How good an idea do you have about the project topic that you are going to do as part of your course project	3.8	24.13	3.65	16.7	NA	NA
9	How good an idea have you got about HPC from the course project that you did?	NA	NA	NA	NA	8.07	88.8
10	Did you find the course interesting?	7.95	87.9	7.0	79.1	8.15	90.6
11	How relevant do you think HPC is in the industry?	8.18	94.82	7.98	81.3	8.13	88.7

Table-2

	Questions (can select more than one answer)	2017 Survey 1	2017 Survey 2
1	*From the lectures, what do you think is parallel programming?*		
(a)	Breaking down the problem into parallel executable chunks	70.8	71.7
(b)	Converting a serial execution into a parallel execution	39.6	39.6
(c)	Changing the algorithm fundamentally for parallel execution	56.3	69.8
2	*From the lectures, which of these broad areas do you think are a part of HPC?*		
(a)	Parallel algorithm design	95.8	96.2
(b)	Compiler design	33.3	26.4
(c)	Storage systems	39.6	45.3
(d)	Network design	12.5	26.4
(e)	Load balancing, scheduling and resource management	83.3	90.6

Table 1, Table 2 and Table 3: Course evaluation survey statistics for course CS301. 2016 autumn semester – 58 students (22% female; 78% male) and 2017 autumn semester: 58 students (14% female; 86% male).
Table-1 - **P1**: weighted average of responses (10-point scale) and **P2**: percentage of favorable responses (≥6). Survey-1 was carried out after two weeks of lecture and Survey-2 at the end of the course. Table-2 – Percentage of students choosing that particular answer.

Table-3

Have you heard of the following terms prior to taking this course?	2016 Survey 1	2017 Survey 1
i. Parallel Programming and HPC	86.2	58.3
ii. Concurrency and Multitasking	83.1	68.8
iii. Latency and Throughput	84.4	50
iv. Granularity and profiling	1.72	0
v. Locality (Spatial and Temporal)	25.8	72.9
vi. Multicore	91.3	68.8
vii. Shared & Distributed Memory	72.4	35.4
viii. Multithreading	96.5	70.8
ix. Speedup and Efficiency	44.8	14.6
x. OpenMP and MPI	3.44	0

Table 3: Percentage of students replying "yes"

Acknowledgements: The author would like to acknowledge the supports received through NSF/TCPP CDER Center Early Adopter Awards Program (Fall 2014). The work has been carried out using the HPC resources at DA-IICT. Thanks to the CS301 course TAs (Y. Keswani, A. Varma, A. Mankodi, R. Pinge and O. Damle)