— Curriculum Initiative on Parallel and Distributed Computing at the University of Western Ontario —

Marc Moreno Maza, Mark Daley, Mike Katchabaw and Hanan Lutfiyya

Overview and Background

This poster describes the integration of high-performance computing courses in the undergraduate curriculum of the computer science department at the **U**niversity of **W**estern **O**ntario. The department is home to over 100 undergraduate students and 120 graduate students. The research interests of the 28 Faculty members cover a large range of topics in HPC and their industrial partners include IBM, Intel, Microsoft, Maplesoft. UWO is also home of SHARCNET, a consortium of Canadian institutions who share a network of high performance computers.

As many other computer science departments, we are implementing the necessary transformations of our curriculum that the democratization of High Performance Computing (HPC) requires.

Integration Plans

Algorithms are often taught with algebraic complexity as the main complexity measure and with serial running time as the main performance counter. On modern computers minimizing the number of arithmetic operations is no longer the primary factor affecting performance. Effective utilization of the underlying architecture can be much more important. Therefore, many traditional courses need to evolve and integrate the different and related aspects of HPC, among them, parallel processing, distributed computing, computer architecture, models of computation, code optimization, performance analysis, etc. This transformation of our curriculum will certainly go through several iterations.

We describe below the three courses (CS 2101, CS 3101, CS 4402) of our new *Minor in High Performance*, which will be offered for the first time this academic year.

CS 2101 – Foundations of Programming for High Performance Computing

Following the notations of the *NSF/TCPP Curriculum Initiative on Parallel and Distributed Computing - Core Topics for Undergraduate*, this course is a one-term course in between CS1 and CS2. We assume that students have already taken an introductory CS course. Nevertheless, CS 2101 starts with a 4-week overview of C-UNIX during which students are introduced to performance issues, in particular memory traffic.

The next 6 weeks will be dedicated to an introduction to multithreaded programming, using `Cilk`. Indeed, this language is an extension of C (with essentially three additional keywords) and its running system comprises a dynamic scheduler. This will allow students to focus on exposing parallelism in their programs. However, notions from scheduling (dependencies, task graphs, work and span, work stealing principle) together with performance metrics (speedup, efficiency, scalability) are presented to the students. Matrix multiplication, matrix transposition, counting sort and merge sort will illustrate the topics of the course. The last two weeks, students are taught how SPMD programs are written and how they execute.

CS 3101 – Theory and Practice of High Performance Computing

This course is built on the principle that, in parallelism, the models of computation that have a practical value are those which bear a relationship to what happens in hardware. Thus this course will interleave the following topics from computer architecture and algorithms.

Architecture topics cover data parallel models, control parallelism models, issues with data traffic and performance analysis while algorithm topics include models and complexity, paradigms (divide & conquer, series-parallel composition) patterns (broadcast, MapReduce, stencil computation) concurrent data structures, synchronization and scheduling.

CS 4402 – Distributed and Parallel Systems

This course is dedicated to the programming aspects of HPC. It comprises an in depth study of three concurrency platforms that illustrate the main stream programming paradigms in distributed/parallel computing:

- CilkPlus for fork-join multithreaded programming targeting multicores
- CUDA for data parallelism on GPUs
- MPI for message passing distributed computing.

The course also gives an overview of other popular concurrency platforms such as PThreads, OpenMP and OpenCL.

Evaluation Plans

For our Minor in High Performance Computing, we present our evaluation plans, at both the student level and the instructor level.

1. First of all, and in addition to the traditional instructor evaluations and course surveys, we value greatly the spontaneous feedback letters from the students. For its Winter 2011 edition the instructor of CS 4402 received several such letters.
2. Secondly, for a course like CS 4402, where the final exam is replaced by individual course projects, the student-instructor interaction during the project development provides a way to measure the course impact.
3. For courses with several programming assignments (CS 2101, CS 4402) one can also easily analyze the students' progress (for instance in terms of code performance) by running appropriate software tools.
4. Evaluating the impact of CS 3101 (for which the portion of programming exercises in the assignments will be limited) will be done differently. Indeed, the first edition of CS 3101 will be taught during the Winter 2013. Thus, until that term, CS 4402 integrates part of the CS 3101 materials. After that, we hope to see that the students will be much more at ease at programming CilkPlus, CUDA or MPI, when they have acquired the appropriate background knowledge in architecture and algorithms during a previous course, namely CS 3101.
5. We are also looking forward to see the impact of these courses on non-CS students. We hope that the number of such students will increase regularly and that our Department could become a *Service Department* for teaching HPC on our campus.