Lessons Learned after the Introduction of Parallel and Distributed Computing Concepts into ECE Undergraduate Curricula at UTN-Bahía Blanca Argentina

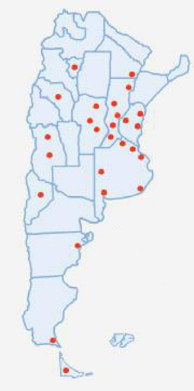
Javier Iparraguirre
Guillermo R. Friedrich
Ricardo J. Coppo

Universidad Tecnológica Nacional, Facultad Regional Bahía Blanca
11 de Abril 461, Bahía Blanca, Argentina
j.iparraguirre@computer.org
http://www.frbb.utn.edu.ar/hpc/

May 15, 2012

UTN in Argentina

- 29 branches across the country.
- Focus on engineering programs.
- Around 75000 students total.



Our Location: Bahía Blanca

- 700 Km south from Buenos Aires.

Motivation and Timeline

- Around 2003 microprocessor manufacturers hit the *power wall*. We saw the birth of the multi-core processor.
- Near 2008 NVIDIA released Compute Unified Device Architecture (CUDA) and Khronos Group proposed Open Computing Language (OpenCL).
- We decided to teach Parallel and Distributed Computing (PDC) concepts to undergraduate Electrical and Computer Engineering (ECE) students.
- In 2011 we began to teach a course called **Parallel Processing**.
- During 2011 we received an award from the NSF/IEEE TCCP 2011 Early Adopters Program [1].

Parallel Processing Course - Prerequisites

- In Argentina most engineering programs have a 5 year extension.
- Only advanced students are eligible for the course.
- Before they enroll students have to pass:
  - A course on C programming language and structured program design.
  - A course on C++ and OOP (Object Oriented Programming).
  - Two courses on computer architecture (basic and advanced topics).

Course Contents

- One semester (16 week schedule).
- Every week contains a two hour theoretical lecture and a two hour practical activity.
- Focus on practical work.
- The course is divided in four stages.
  1. Week 1 to 5: Theoretical concepts
  2. Week 6 to 8: Message Passing Interface (MPI)
  3. Week 9 to 12: Open Multi-Processing (OpenMP) and POSIX Threads (Pthreads)
  4. Week 12 to 15: Heterogeneous computing platforms and OpenCL

Course Schedule (1 of 2)

Week	Concept
1	Introduction to parallel processing
2	Parallel processing architectures
3	Principles of parallel processing algorithms
4	Communication and synchronization models
5	Analytical model of parallel programs
6	MPI
7	MPI
8	MPI

Course Schedule (2 of 2)

Week	Concept
9	Pthreads
10	Pthreads
11	OpenMP
12	OpenMP
13	OpenCL
14	OpenCL
15	OpenCL
16	Frequently used algorithms

Resources

1. A computer cluster running ROCKS.
2. Sun SPARC Enterprise T5120 Server.
3. A server with an OpenCL-capable GPU installed.

Pedagogic Methodology

- Our course is very practical.
- Students must complete three projects.
- First project is related to MPI.
- The second challenge consists on programming Pthreads and OpenMP.
- The third one explores OpenCL.
- All participants must do the same activity.

Projects

- In each programming assignment students had access to a serial and a parallel implementation of the code.
- They are required to understand the programs provided.
- They must improve the performance of the parallel code.
- In 2011 the programming projects were based on a 2D particle system simulation.
- Most of the material was based on CS267 course at Berkeley University [2].

Evaluation Methodology

- The course grading is a mix of multiple aspects:
  - Results obtained.
  - Quality of the presentations.
  - Class participation.
  - Creativity of the solutions achieved.
- Students are forced to direct attention to multiple aspects such as speech, posture, and gestures besides the programming task.

Lessons Learned

- After finishing the course we asked the students to give feedback.
- We arrive to a 6 items list.

Lesson 1: Practical Work is Essential

- Students have no difficulties to understand PDC theoretical concepts.
- They *see* the reason to study them when they must use them so solve a problem.
- A typical example is mutual exclusion.
  - It is simple to understand what a critical section is.
  - Students have a hard time when it comes to solve a problem that involves multiple threads that have some degree of dependency.

Lesson 2: Small Assignments Every Week

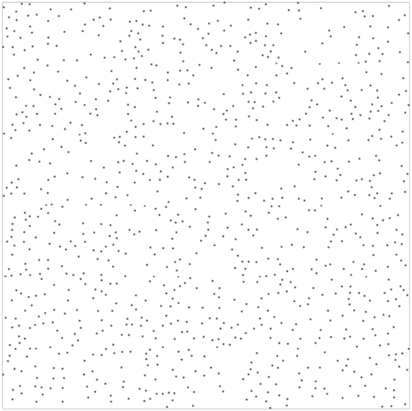
- We observed that students spent most of their time dealing with the technology side of the problems.
- By the time they dominated the technology and the requirements they were close to deadline.
- Weekly practical workshops help to avoid this problem.

Lesson 3: Shared Memory Concepts should be Taught First

- The first assignment in our parallel processing class involved MPI.
- We realized that this is not the best way to initiate.
- Shared memory practices are easier to understand and should be taught first.
- OpenMP is a good choice as an initial experience.

Lesson 4: Result Visualization is Important

- In our programming projects we provided an application to visualize the results.
- It helps to teach students that parallel programming can be tricky when it comes to maintain the data consistency.



Lesson 5: PDC Concepts should be Distributed Among the University Curricula

- It is impossible to include all the required PDC concepts in a one semester course.
- It is necessary to spread part of the basic material among multiple courses.
- we found very helpful the NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing.
- We are evaluating our complete curricula.

Lesson 6: Use Smartphones and Tablets as Target Platforms

- The latest smartphones in the market include multi-core CPUs and a GPU
- They are the perfect low-cost heterogeneous platform to introduce PDC concepts.
- They attract the attention of young engineering students.

Updated Course Schedule (1 of 2)

Week	Concept
1	Introduction to parallel processing
2	Parallel processing architectures
3	Principles of parallel processing algorithms
4	Communication and synchronization models
5	Analytical model of parallel programs
6	OpenMP
7	OpenMP
8	Pthreads
9	Pthreads

Updated Course Schedule (2 of 2)

Week	Concept
10	OpenCL
11	OpenCL
12	OpenCL
13	MPI
14	MPI
15	MPI
16	Frequently used algorithms

Future Work and Conclusions

- Our first experience teaching Parallel Processing at UTN has been very rewarding.
- We are working on modifications in the parallel processing course and in the University curricula.
- Students were highly motivated by the practical "hands-on" nature of the course.
- Most of the presentations of their programming tasks were well done.
- Some have triggered an interest in future research work on more advanced topics.

Thank you!

- Questions?
- E-mail: `j.iparraguirre@computer.org`
- Website: `http://www.frbb.utn.edu.ar/hpc/`

References

1 S.K. Prasad, A. Chtchelkanova, S. Das, F. Dehne, M. Gouda, A. Gupta, J. Jaja, K. Kant, A. La Salle, R. LeBlanc, et al. NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing: core topics for undergraduates. In Proceedings of the 42nd ACM technical symposium on Computer science education, pages 617–618. ACM, 2011.

2 V. Volkov. Berkeley University CS267 Particle Simulation Code, 2011 (accessed February 28, 2012). `http://www.cs.berkeley.edu/~volkov/cs267.sp09/hw2/`