Teaching Parallel and Distributed Systems Programming for 4th year Computer Science and Other Discipline Students

Marcelo Arroyo
Universidad Nacional de Río Cuarto
Argentina

2013
<table>
<thead>
<tr>
<th></th>
<th>Chalenges</th>
<th>Approach</th>
<th>Techniques and tools</th>
<th>Results</th>
</tr>
</thead>
</table>

**Table of contents**

1. Chalenges
2. Approach
3. Techniques and tools
4. Results
Challenges

Teaching PDC
- Teaching parallel and distributed systems programming to:
  1. Undergraduate computer science students
  2. People from other disciplines (mathematics, engineering, physics, biology, ...)

Problems
- Heterogeneity of knowledge
- Different expectations
Students background

Computer science students
- 4th year students
- Good background on:
  - Programming language paradigms
  - Data structures and algorithms design
  - Concurrency topics

Other students
- Basic skills on (imperative) programming
- Limited knowledge on data structures and algorithms
- Almost no background on concurrency
## Approach

### Top-down approach

- Using high-level parallel patterns and skeletons
  1. Introduce classical problems (matrix multiplication, sorting, searching, . . .)
  2. Solve applying patterns
  3. Do performance analysis
  4. Study/analyse/extend/improve pattern internals (threads, MPI, . . .)

### Advantages

- Reduce the gap with sequential programming
- Allow students to focus on problem solving
- Students are enthusiastic from the beginning
## Challenges

<table>
<thead>
<tr>
<th>Approach</th>
<th>Techniques and tools</th>
<th>Results</th>
</tr>
</thead>
</table>

## Tools

### Parallel patterns
- Object-oriented parallel patterns
- C++ parallel skeletons (templates meta-programming)
  - SkeTo, VecCL, pdt, ...
- Targets on threads, openmp, MPI, CUDA and OpenCL
- Patterns hide implementation targets
- Patterns presented graphically
- Each pattern has its corresponding (high level) contract

### Students differentiation
- Computer science students are encouraged to develop new patterns (or implement new targets)
- Other students focus on problem solving and performance improvements
The experience

Results

- We can teach both: computer science and other discipline students
- All students can build complex parallel programs from start
- Knowledge is given in increasing way: from high-level designs to low-level implementation details
- Parallel patterns and skeletons help to reduce the gap with sequential programming.
- Students can compare different implementations with same patterns (on different targets)