Parallelism (PAR)

Course presentation

Eduard Ayguadé, Jordi Garcia, Josep R. Herrero,
Daniel Jiménez, Julián Morillo, Nacho Navarro,
Jordi Tubella and Gladys Utrera

Computer Architecture Department
Universitat Politécnica de Catalunya

2014/15-Fall

General objective I

Design, implement, compile and execute parallel programs

- Specific objectives
  - Identify and describe the different levels of parallelism in a computer
  - Create a task or data decomposition strategy to parallelize a serial application
  - Implement, compile and execute the parallelization strategy (mainly OpenMP)
    - Understand what others programming models (MPI, OmpSs, CUDA, ...) do offer
  - Use synchronization techniques to avoid race conditions, deadlocks, etc.

General objective II

Create simple performance models, measure and optimize parallel programs

- Specific objectives
  - Create simple models based on the decomposition strategy
  - Measure the performance of an implementation using instrumentation and visualization tools
  - Detect bottleneck factors based on the performance measures: granularity, load balance, task interaction, etc.
  - Apply optimizations to solve bottleneck problems

Chronological Syllabus (2T/P and 2L per week)

1. Introduction and motivation
   - Parallelism and concurrency
2. Understanding parallel performance
   - Amdahl's law, speedup, scalability, overheads, performance models, ...
3. Introduction to parallel arquitectures
4. Parallel programming patterns I
   - Task decomposition
5. Parallel programming patterns II
   - Data decomposition

Course Methodology: Theory

- Theory/Problems (T/P): 2 hours/week
  - Presentation of concepts and examples using slides
  - Two in-term exams:
    - First in-term exam: October 22nd (12:30–14:00)
    - Second in-term exam: December 17th (12:30–14:00)
    - They will be done without computer, calculator, documentation, ...

Course Methodology: Laboratory Activities

- Laboratory (L): 2 hours/week
  - 4 laboratory assignments (Lab0–3), to be published at **FIB Raco**
- Development context:
  - Groups of two students
  - Remote access to a multiprocessor server machine at DAC
    - Architecture: 4 nodes with 12 cores each
    - Operating system: Ubuntu Linux
  - Programming language: C using OpenMP extensions

Course Methodology: Laboratory Activities

- Guided laboratory sessions
  - Initial sessions (Lab0): Compilation, execution, performance prediction and analysis tools, understanding overheads and OpenMP mini-tutorial
  - Parallelization sessions (Lab1–3): three small applications using OpenMP
- Deliverables will be submitted via the **FIB Raco**
  - `pdf` documents, C codes, scripts, etc.
  - **Deadline**: Just before starting next laboratory session

Bibliography

- Mainly covered by 2 books
  - Mattson, T. and Sanders, B. and Massingill, B., `Patterns for Parallel Programming`, Addison Wesley Software Patterns Series, 2004.
  - Grama, A. and Karypis, G. and Kumar, V. and Gupta, A., `Introduction to Parallel Computing`, Addison-Wesley, 2003.
- Additional documentation published through the **FIB Raco**
  - Slides, collection of exercises and collection of solved in-term/final exams for the T/P sessions
  - Description of L sessions
  - Links to manuals and quick reference guides for the programming models

Evaluation

- We evaluate through the following elements:
  - AC = (1^{st} in-term exam · 0.3) + (2^{nd} in-term exam · 0.7)
  - Lab = Evaluation of the laboratory sessions
    - Lab0 (20%), Lab1 (20%), Lab2 (30%) and Lab3 (30%)
    - Monitoring of the laboratory sessions by the professor
  - Final Exam (EF): **mandatory** if $AC < 5.0$, optional otherwise
- If $AC \geq 5.0$
  $Final\ Mark = 0.7 \cdot AC + 0.3 \cdot Lab$
- If student does the final exam, independently of AC, then
  $Final\ Mark = 0.7 \cdot max(EF, 0.25 \cdot AC + 0.75 \cdot EF) + 0.3 \cdot Lab$

Evaluation

- The Third Language generic competence will be evaluated through
  - Report for Lab3 fully written in English, using an appropriate format
  - Previous Lab1–2 will include short writing activities that do not contribute to the competence grade
- Rubrics and evaluation criteria known in advance
- Grading: A, B, C, D or NA

Parallelism (PAR)

Course presentation

Eduard Ayguadé, Jordi Garcia, Josep R. Herrero,
Daniel Jiménez, Julián Morillo, Nacho Navarro,
Jordi Tubella and Gladys Utrera

Computer Architecture Department
Universitat Politécnica de Catalunya

2014/15-Fall