Course Overview

Miguel Areias

Computer Science Department
Faculty of Sciences
University of Porto

Parallel Computing 2018/2019

Contacts

- Lecturer (1st part) - Miguel Areias
  - Room: 1.03
  - Email: `miguel-areias@dcc.fc.up.pt`
  - Web: `www.dcc.fc.up.pt/~miguel-areias/teaching/1819/cp/`
- Lecturer (2nd part) - Eduardo Marques
  - Room: 1.72
  - Email: `edrdo@dcc.fc.up.pt`
  - Web: `...`

Structure

- **Teaching Language**: English.
  Most of the slides are already in English. If you want the Portuguese versions, please check the previous versions of the course.
- **Theoretical - Practical Classes**:
  Theoretical classes: discuss the program topics.
  Practical classes: solve some exercises.
  Support classes: assist in the projects.
- **Timetable**:
  Monday 17:00-18:30 - Room 1.42 (S3)
  Monday 18:30-20:00 - Room 1.55 (Lab1)

Grading Components

- **Components of the Evaluation**

  2 projects assignments (TP_1 and TP_2) with individual presentations.
  Groups with 1/2 elements. Without minimum grade.
  Final exam (FE). With a minimum grade of 40%.

- **Final classification (CF)**:

  $CF = 3 * TP_1 + 3 * TP_2 + 14 * FE$

- **Timetable**:

  TP_1: show - 8 October delivery/presentation - 8/15 November
  TP_2: show - 19 November delivery/presentation - 17/19 of December
  FE: 1st and 2nd exam seasons

Main Goals

- Introduce the students to advanced concepts on the theory and practice of computational models for parallel and distributed memory architectures.
- Hands-on experience on programming in:
  - distributed memory architectures (MPI);
  - shared memory architectures using processes and threads (Pthreads and OpenMP).

Syllabus

- **Introduction and foundations:**
  Parallel programming, concurrency and parallelism, and Flynn's taxonomy. Foster's programming methodology. Major parallel programming models and paradigms.

Syllabus

- **Introduction and foundations:**
  Parallel programming, concurrency and parallelism, and Flynn's taxonomy. Foster's programming methodology. Major parallel programming models and paradigms.
- **Programming for distributed memory architectures using MPI**:
  MPI specification, explicit message passing, communication protocols, derived types and data packing, collective communication, communicators and topologies.

Syllabus

- **Introduction and foundations:**
  Parallel programming, concurrency and parallelism, and Flynn's taxonomy. Foster's programming methodology. Major parallel programming models and paradigms.
- **Programming for distributed memory architectures using MPI**:
  MPI specification, explicit message passing, communication protocols, derived types and data packing, collective communication, communicators and topologies.
- **Programming for shared memory architectures with processes**:
  Processes, shared memory segments, shared memory through file mapping, spinlocks and semaphores.

Syllabus

- **Programming for shared memory architectures with threads**: Pthreads specification, multithreaded programming, mutexes, conditional variables and keys.

Syllabus

- **Programming for shared memory architectures with threads**: Pthreads specification, multithreaded programming, mutexes, conditional variables and keys.
- **Programming for shared memory architectures with OpenMP**: OpenMP specification, compilation directives, work-sharing constructors, basic constructors, synchronization constructors, basic functions, locking functions, environment variables, removing data dependencies, performance.

Syllabus

- **Programming for shared memory architectures with threads**: Pthreads specification, multithreaded programming, mutexes, conditional variables and keys.
- **Programming for shared memory architectures with OpenMP**: OpenMP specification, compilation directives, work-sharing constructors, basic constructors, synchronization constructors, basic functions, locking functions, environment variables, removing data dependencies, performance.
- **Hybrid Programming with MPI, OpenMP and Pthreads**: Combining MPI with OpenMP and using MPI in a multithreaded environment.

Syllabus

- **Memory Cache and Multi-Processor Architectures:**
  The importance of *caches* in a multi-processor architecture: space and temporal locality, synchronization with memory barriers and with mutual exclusion.

Syllabus

- **Memory Cache and Multi-Processor Architectures:**
  The importance of *caches* in a multi-processor architecture: space and temporal locality, synchronization with memory barriers and with mutual exclusion.
- **Performance metrics:**
  Speedup measures, efficiency, redundancy, usability and quality of a parallel application. Amdahl law. Gustafson-Barsis law. Karp-Flatt metrics.

Syllabus

- **Memory Cache and Multi-Processor Architectures:**
  The importance of *caches* in a multi-processor architecture: space and temporal locality, synchronization with memory barriers and with mutual exclusion.
- **Performance metrics:**
  Speedup measures, efficiency, redundancy, usability and quality of a parallel application. Amdahl law. Gustafson-Barsis law. Karp-Flatt metrics.
- **Parallel algorithms:**
  Scheduling and load balancing strategies. Parallel algorithms for sorting, search, monte-carlo simulation and matrix multiplication.

Main Bibliography

- **Parallel Programming with MPI**
  P. Pacheco. Morgan Kaufmann.
- **Parallel Programming in C with MPI and OPenMP**
  Michael J. Quinn. McGraw-Hill.
- **Parallel Programming for Multicore and Cluster Systems**
  Thomas Rauber and Gudula Rünger.
- **Advanced Linux Programming**
  M. Mitchell, J. Oldham and A. Samuel. New Riders.

Main Bibliography

- **Pthreads Programming**
  B. Nichols, D. Buttlar and J.P. Farrell. O'Reilly.
- **Parallel Programming in OpenMP**
  R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald and R. Menon. Morgan Kaufmann.
- **Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers**
  B. Wilkinson, M. Allen. Prentice Hall.