COURSE GUIDE: Numerical and Data-Intensive Computing (COMP)

Academic year:	2017-2018
Program:	Master's program in Research and Innovation in Information and Communications Technologies (i^2-ICT)
Center:	Escuela Politécnica Superior
University:	Universidad Autónoma de Madrid
Last modified:	2017/06/13
Status:	Approved 2013/05/29

1. ASIGNATURA / COURSE (ID)

Cálculo intensivo y manejo de datos a gran escala
Numerical and Data-Intensive Computing (COMP)

1.1. Programa / program

Máster Universitario en Investigación e Innovación en Tecnologías de la Información y las Comunicaciones (i^2-TIC)

Master in Research and Innovation in Information and Communications Technologies (i^2-ICT) [Officially certified]

1.2. Course code

32416

1.3. Course areas

Computer Science and Artificial Intelligence

1.4. Tipo de asignatura / Course type

Obligatoria [itinerario: todos los itinerarios]
Core [itinerary: all itineraries]

1.5. Semester

First semester

1.6. Credits

6 ETCS

1.7. Language of instruction

The lecture notes are in English. The lectures are mostly in Spanish. Some of the lectures and seminars can be in English.

1.8. Recommendations / Related subjects

Knowledge of the C programming language, Data Bases, probability and statistics at an introductory level is useful to follow the course.

Related subjects are:

- Procesamiento de información temporal [Temporal Information Processing]
- Aprendizaje Automático: teoría y aplicaciones [Machine Learning: Theory and Applications]
- Métodos bayesianos aplicados [Applied Bayesian Methods]
- Aceleración de algoritmos en sistemas heterogéneos [Algorithm Acceleration in Heterogeneous Systems]
- Procesamiento de señales biomédicas y sus aplicaciones [Biomedical Signal Processing and its Applications]
- Procesamiento de audio y voz para biometría y seguridad [Speech and Audio Processing for Biometrics and Security]
- Técnicas de análisis de secuencias vídeo para videovigilancia [Techniques of Analysis of Video Sequences for Surveillance]

1.9. Lecturers

Add @uam.es to all email addresses below.

Lectures and labs:

Dr. Carlos Santa Cruz Fernández (Coordinator)
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Office: B-343
Tel.: +34 914972337
e-mail: carlos.santacruz
Web: http://www.eps.uam.es/~santacru

Dr. Miguel Ángel García García
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Office: C-242
Tel.: +34 914976208
e-mail: miguelangel.garcia
Web: http://www.eps.uam.es/~mgarcia/

Dr. Estrella Pulido Cañabate
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Office: B-413
Tel.: +34 914972289
e-mail: estrella.pulido
Web: http://www.eps.uam.es/~epulido/

Dr. Gonzalo Martínez Muñoz
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Office: B-422
Tel.: +34 914977528
e-mail: gonzalo.martinez
Web: http://www.eps.uam.es/~gonzalo/

1.10. Objetivos de la asignatura / Course objectives

Esta asignatura está divida en tres partes. La primera parte corresponde a una introducción a las técnicas de programación eficiente de las arquitecturas paralelas de memoria compartida, incluyendo los procesadores multi-núcleo y los multiprocesadores fuertemente acoplados. En concreto, se describen herramientas de análisis de rendimiento (*profilers*), se identifican los principales factores que afectan a la eficiencia de las arquitecturas paralelas, y se estudian técnicas de paralelización de bucles y programas secuenciales en arquitecturas de memoria compartida mediante OpenMP. Finalmente, se realiza una introducción y análisis comparativo de los entornos de programación distribuida mediante OpenMPI, Hadoop y Spark.

En esta asignatura se presenta además una introducción al sistema Spark de procesamiento distribuido sobre un cluster de nodos. Este sistema además de permitir la alta disponibilidad permite el procesamiento de grandes volúmenes de datos en un cluster de máquinas

Las competencias básicas que el estudiante adquiere en esta asignatura son:

- C1. Capacidad para resolver problemas utilizando el paradigma de computación en paralelo de Apache Spark.
- C2. Capacidad para crear soluciones en Apache Spark eficientes.
- C3. Capacidad para crear soluciones en Apache Spark que utilicen: datos estructurados, métodos de aprendizaje automático y/o fuentes de datos de streaming.

La competencia de tecnología específica que el estudiante adquiere en esta asignatura es:

- E5 - Capacidad de diseño, implementación, despliegue y depuración de aplicaciones intensivas en datos y/o computación, orientadas a conjuntos de datos de escala masiva, incluyendo la capacidad de analizar y optimizar aplicaciones en arquitecturas multicore, sistemas paralelos y cloud computing

Las cualificaciones ubicadas en el nivel de competencias transversales que el estudiante adquirirá en esta asignatura son:

- TR1 Capacidad para actualizar conocimientos habilidades y destrezas de forma autónoma, realizando un análisis crítico, análisis y síntesis de ideas nuevas y complejas abarcando niveles más integradores y pluridisciplinares.

- TR4 Capacidad para transmitir de un modo claro y sin ambigüedades a un público especializado o no, resultados procedentes de la investigación científica y tecnológica o del ámbito de la innovación mas avanzada, así como los fundamentos mas relevantes sobre los que se sustentan. Capacidad para argumentar y justificar lógicamente dichas decisiones de un modo claro y sin ambigüedades, sin dejar de considerar puntos de vista alternativos o complementarios.

This subject is divided into three parts. The first part corresponds to an introduction to efficient programming techniques for shared memory parallel architectures, including multi-core processors and tightly-coupled multiprocessors. In particular, it aims at describing performance analysis tools (profilers), identifying the main factors that affect the efficiency of parallel architectures, and studying parallelization techniques for loops and sequential programs on shared-memory parallel architectures through OpenMP. Finally, a comparative analysys of OpenMPI, Hadoop y Spark is performed.

In this subject, an introduction to the spark is presented. This application is a distributed computing system over a cluster of nodes. This architecture allows managing and handling a large amount of data in a scalable and resilience architecture.

At the end of each unit, the student should be able to:

UNIT BY UNIT SPECIFIC OBJECTIVES

PART I	
UNIT 1.- Introduction	
1.1.	Know the different families of parallel architectures and choose the one that best suits a specific application scope.
1.2.	Measure the performance of parallel algorithms in terms of speedup and efficiency.
1.3.	Understand the basic concepts behind parallel programming, including tasks, processes and synchronization mechanisms.
1.4.	Use performance analysis tools (profilers) to analyze the efficiency of sequential algorithms and identify portions susceptible to be accelerated through parallelization.
1.5.	Implement simple parallel algorithms on shared-memory parallel architectures using C and OpenMP.
UNIT 2.- Loop parallelization on shared-memory parallel architectures	
2.1.	Understand the different types of parallel loops and choose the ones that best suit a specific problem.
2.2.	Describe the dependencies between iterations of a set of sequential loops through a dependency graph.
2.3.	Parallelize a set of sequential loops from their corresponding dependency graph.
2.4.	Apply code transformations in order to optimize the parallelization of a set of sequential loops.
UNIT 3.- Parallelization in distributed architecture and distributed memory	
3.1.	Understand the different architectures for distributed memory environments
3.2.	Understand the main development environment for distributed systems
PART II: Spark	
4.1.	Understand the spark architecture
4.2.	Understand the program flow of a spark program
4.3.	Understand main input and output sources
PART III: Spark SQL	
5.1.	Understand DataFrame concepts
5.2.	Learn how to use the DataFrame API in SparkSQL
PART IV	
UNIT 6.- Introduction	
6.1.	Understand the floating-point data representation in modern processors
6.2.	Understand that the round-off error is due to the fact that arithmetic among numbers is not exact
6.3.	Understand that the errors due to the algorithm used are independent of the hardware
6.4.	Understand that errors can be successively magnified due to unstable algorithms

UNIT 7.- Solution of Linear Algebraic Equations	
7.1.	Write a Multiple Linear Regression problem in matrix form
7.2.	Solve some practical examples

1.11. Course contents

PART I

1. Introduction
1.1. Parallel architectures: motivation
1.2. Shared-memory parallel architectures
1.2.1. Multi-core processors and tightly-coupled multiprocessors
1.2.2. OpenMP
1.3. Distributed-memory parallel architectures: loosely-coupled multiprocessors
1.4. Grid computing and cloud computing
1.5. Basic concepts
1.5.1. Tasks and processes
1.5.2. Semaphores and barriers
1.5.3. Performance analysis tools (profilers)

2. Loop parallelization on shared-memory parallel architectures
2.1. Parallel loops
2.2. Scheduling of parallel loops
2.3. Analysis of dependencies between iterations
2.3.1. True dependencies
2.3.2. Anti-dependencies
2.3.3. Output dependencies
2.3.4. Dependencies in nested loops
2.3.5. Dependency graphs by levels
2.4. Generation of parallel code
2.4.1. Strongly connected components
2.4.2. Acyclic condensation
2.4.3. Barrier-free arcs
2.4.4. Generation of clusters and segments
2.4.5. Code generation
2.4.5.1. Code generation for serial segments
2.4.5.2. Code generation for parallel segments
2.4.6. Example problems
2.5. Transformations to support parallelization
2.5.1. Loop normalization
2.5.2. Scalar substitution
2.5.3. Scalar expansion
2.5.4. Variable copying
2.5.5. Loop interchange

3. General process for parallelization of sequential programs
3.1. Decomposition
3.2. Assignment
3.2.1. Load balancing
3.2.2. Communication reduction
3.2.3. Overhead reduction

3.3. Orchestration
3.4. Mapping

PART II

4. Spark Fundamentals
   4.1. Introduction: architecture and organisation
   4.2. Data in Spark: Resilient Distributed Datasets (RDDs)
   4.3. Program flow in Spark
   4.4. Data input and output
   4.5. Transformations
   4.6. Persistence
   4.7. Actions
   4.8. Shared variables: broadcast and accumulators

PART III

5. Spark SQL
   5.1. Introduction to DataFrames
   5.2. Data sources: Json, JDBC/ODBC, Parquet, etc.
   5.3. DataFrames API
   5.4. UDFs and Window Functions

PART IV

6. **Introduction**
   6.1. Floating-Point Representation
   6.2. Roundoff Error
   6.3. Truncation Error
   6.4. Stability

7. **Solution of Linear Algebraic Equations**
   7.1. LU Decomposition
   7.2. Cholesky decomposition
   7.3. General Linear Regression Model in matrix term
   7.4. Practical applications

1.12. Course bibliography

1. "Parallel Computer Architecture: A Hardware/Software Approach" D. Culler, J.P. Singh, A. Gupta. *Ed. Morgan Kaufmann, 1998.*
2. "Supercompilers for parallel and vector computers" H. Zima, B. Chapman *Ed. ACM Press, 1991.*
3. “Optimizing Compilers for Modern Architectures: A Dependence-based Approach”. Allen, K. Kennedy *Ed. Morgan Kaufmann, 2001.*
4. "Computer Architecture: A Quantitative Approach" (5a. ed.) J.L. Hennessy, D.A. Patterson *Ed. Morgan Kaufmann, 2011.*
5. "Advanced Computer Architecture: Parallelism, Scalability, Programmability" K. Hwang. *Ed. McGraw-Hill, 1992.*
6. “Numerical Recipes in C: The Art of Scientific Computing”, W. H. Teulosky, A. A. Vetterling, W. T. Flannery, B. P., Cambridge University Press, 1992
7. “Numerical Mathematics. Theory and Computer Applications” C. E. Froberg.. Addison-Wesley, Reading, Massachusetts, 1985.
8. “Scientific Computing: An Introductory Survey” M. T. Heath., 2nd. ed. McGraw-Hill, New York, 2001.
9. “Análisis numérico con aplicaciones” C. F. Gerald and P.O. Wheatley., 6a ed.Prentice Hall, México, 2000.
10. Spark Cookbook. Rishi Yadav. Packt Publishing. 2015.
11. Spark in action. Peter Zecevic, Marko Bonaci. Manning Publications. 2017.

1.13. Coursework and evaluation

The course involves lectures, weekly assignments, lab assignments, a seminar presentation and one exam.

In the ordinary exam period, it is necessary to have a pass grade (≥ 5) in the exam to pass the course. In the extraordinary exam period, it is necessary to have a pass grade (≥ 5) in the report on a related research topic to pass the course.

- In the ordinary exam period, the evaluation will be made according to the following scheme
  - 50 % Lab assignments
  - 50 % Exam [end of term]

  The grades of the individual parts are kept for the extraordinary exam period.

- In case of a fail grade in the ordinary exam period, in the extraordinary exam period, the student has the opportunity to
  - Turn in all the lab assignments with corrections
  - Turn in a report on a research topic about numerical and data-intensive computing.

  The grade will be determined by
  - 50 % Lab assignments [only if the lab assignments are turned in]
  - 50 % Report on a related research topic [only if the report is turned in]