Universitat Autònoma de Barcelona

2021/2022

Advanced Computer Architectures

Code: 102778
ECTS Credits: 6

Degree	Type	Year	Semester
2502441 Computer Engineering	OB	3	1
2502441 Computer Engineering	OT	4	1

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities.

Contact

Name: Juan Carlos Moure Lopez
Email: JuanCarlos.Moure@uab.cat

Use of Languages

Principal working language: spanish (spa)
Some groups entirely in English: No
Some groups entirely in Catalan: No
Some groups entirely in Spanish: No

Teachers

Miquel Àngel Senar Rosell

Prerequisites

It is very, very advisable to have studied and passed the subjects of Fundamentals of Computer Science, Fundamentals of Computers, Computer Structure, Methodology of Computer Programming and Architecture

Objectives and Contextualisation

The fundamental objective of the subject is that students acquire the capabilities of analysis, use and evaluation of advanced computer architectures, for the development of applications and computer services.

The basic concepts that will be described in the theoretical sessions are: the multi-core processor, the shared memory multi-processor system, the computer accelerators, performance evaluation, and the programming languages and models of these parallel systems. The most specific objectives are the following:

1. Understand the keys to obtain good performance from multi-core and multi-processor computers, and from computer accelerators (GPUs)

2. Identify the opportunities for parallelization in an algorithm or application, at different levels (instructions, iterations of a loop, calls to functions ...), and express it formally

3. Design the data structures of an application and the algorithms that access these data to favor the good performance of the memory hierarchy

4. Analyze, design and implement parallel algorithms with abstract computation patterns (reduction, transformation ...) under programming paradigms based on shared variables and synchronization; and with current parallel languages, such as OpenMP and OpenACC.

5. Verify the functionality and evaluate the performance of parallel applications, identifying bottlenecks with respect to performance

6. From performance analysis, select the right computer to an application and / or perform code optimizations that further improve its performance

Competences

Computer Engineering

- Acquire personal work habits.
- Communication.
- Conceive and develop centralised or distributed computer systems or architectures by integrating hardware, software and networks.
- Have the capacity to analyse and evaluate computer architectures, including parallel and distributed platforms, and develop and optimise software for the same.
- Have the capacity to analyse, evaluate, select and configure hardware platforms for the development and execution of computer applications and services.
- Have the right personal attitude.

Learning Outcomes

1. Analyse the available models, paradigms and parallel programming languages to determine which best fit the needs of an application.
2. Analyse the characteristics of applications that require high performance.
3. Communicate efficiently, orally or in writing, knowledge, results and skills, both in the professional environment and before non-expert audiences.
4. Configure hardware platforms and programming environments for the development and running of high performance applications.
5. Determine the most suitable platforms for each type of application.
6. Develop parallel applications based on the existing paradigms.
7. Evaluate and predict the performance of different high performance computer platforms for running applications.
8. Evaluate the benefits of high performance architectures, as well as the functionality of their applications.
9. Evaluate the functionality and performance of developed parallel/distributed applications.
10. Maintain a proactive and dynamic attitude with regard to the development of ones own professional career, personal growth and continuing education. Be able to cope with adversities.
11. Prevent and solve problems.
12. Propose computer solutions based on high performance systems that integrate both the architecture of the systems hardware components and interconnection between the same, and the design of the necessary software.

Content

1. Instruction-Level Parallelism

- Dynamic reordering of the execution of instructions
- Data dependencies between instructions in a loop

2. Data-Level Parallelism

- Analysis of the data parallelism of an algorithm
- SIMD (vectorization) and MIMD (multi-thread) parallelism
- Vector Architectures and SIMD instructions
- GPUs and compute accelerators

3. Parallel computer architecture

- Structure and architecture of multi-thread and multi-core processors, and multi-processor nodes
- Memory hierarchy (NUMA) and support for cache data coherence
- H/W support for synchronization and for communication between threads

4. Parallel Algorithms

- Model of Parallelism with shared variables and synchronization. Parallel computing patterns
- Data parallelism (SIMD): reduction and transformation operations on multi-dimensional data structures
- Task Parallelism (MIMD): master/worker, divide & conquer, pipeline
- Non-deterministic programming and synchronization without locks
- OpenMP and OpenACC directives

5. Performance evaluation of Parallel Applications

- Analysis of complexity, parallelism and locality
- Measurement of the execution time, IPC, bandwidth and arithmetic intensity
- Total work and critical path of parallel execution
- Speedup and efficiency of parallel execution
- Overhead of synchronization and communications

Methodology

Theory classes: the knowledge of the subject will be explained. The basic concepts will be described and illustrated with practical examples. The most important learning problems will be highlighted and will show how to complete and deepen the contents. Practical cases will be discussed and the teacher will detect the most common comprehension and reasoning problems and solve them for all students.

Classes of Problems in Laboratory: cooperative activities of resolution of problems and of simple practical cases will be made, that will serve as support to the theory. After a previous individual work, the students will make a group sharing and solve their doubts. The teacher will detect the most common comprehension and reasoning problems and solve them in groups or for all the students. The student's analysis and synthesis capacity, critical reasoning and problem-solving ability are promoted. During the session, students should inform the teacher about their progress and the problems that may be encountered, and they will deliver a final document with the results and reflections on their activities.

Practical case of Performance Engineering: throughout the course, students must design, plan, perform, present and defend orally a practical work, indicating the objectives, development, results obtained with understandable graphics, the most notable incidents and, finally, the most relevant conclusions. Students are expected to have a proactive and dynamic attitude, the autonomous search for knowledge that is necessary, and ambition with the objectives of the work.

Transversal Competences:

- T02.04 - Prevent and solve problems. We work in the classes of problems in the laboratory and evaluate with the problems that the students have to solve in group.
- T04.01 - Communicate efficiently, orally or in writing, knowledge, results and skills, both in the professional environment and before non-expert audiences. Work is done in the tutorials and it is evaluated in the deliveries of works and in the presentation of the practical work, and supposes 15% of the final grade.
- T06.01 - Maintain a proactive and dynamic attitude with regard to the development of one's own professional career, personal growth and continuing education. Be able to cope with adversities. It is encouraged and evaluated in practice sessions and in the preparation of the Performance Engineering assignment, where the teacher interacts with students and observes and helps correct their attitudes; accounts for 25% of the grade of these activities.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Classroom Tutorials	6	0.24	1, 8, 4, 12
Problems on LAB	24	0.96	1, 2, 7, 9, 8, 4, 6, 5, 10, 11
Theory	24	0.96	1, 2, 8, 4, 5, 12
Type: Supervised			
Preparation and Presentation of Practical Work	5	0.2	3
Preparation of the work to be done in the laboratory	15	0.6	7, 9, 6, 10
Type: Autonomous			
Autonomous study	24	0.96	1, 8, 4, 5, 12
Prepare LAB sessions	30	1.2	1, 2, 6
Problem solving	20	0.8	2, 9, 8, 5, 11

Assessment

The dates for continuous evaluation and submission of works will be published on the Moodle platform of the UAB (Virtual Campus) and may be subject to programming changes for reasons of adaptation to possible incidents. Always be informed on the Moodle platform about these changes, as it is understood that it is the usual platform for exchanging information between teacher and students.

Evaluation Activities
Activity A: solving practical problems in the laboratory; it is done as a group; it represents 20% of the final grade; it is necessary to obtain a minimum score of 4 points out of 10; and it cannot be recovered.
Activity B: delivery of a report of activity A; it is done individually; assumes 15% of the final grade; it is not necessary to obtain a minimum grade; and it cannot be recovered.
Activity C: proposal and resolution of a practical case of Performance Engineering; it is done as a group; it represents 30% of the final grade; it is necessary to obtain a minimum score of 5 points out of 10; and it can be recovered.
Activity D: oral defense of activity C; it is done individually; assumes 5% of the final grade; it is not necessary to obtain a minimum grade; and it can be recovered.
Activity E: delivery of a report of activity C; it is done as a group; assumes 5% of the final grade; it is not necessary to obtain a minimum grade; and it cannot be recovered.
Activity F: attendance and active participation in class; it is done individually; assumes 5% of the final grade; it is not necessary to obtain a minimum grade; and it cannot be recovered.
Activity G: written examination of theory and problems; it is done individually and without notes; assumes 20% of the final grade; it is necessary to obtain a minimum score of 5 points out of 10; and it can be recovered.

Apart from the assessment activities described above, other voluntary activities may be proposed throughout the course that may contribute to the final grade for the course.

Program of Evaluation Activities
Activities A, B, C and F are carried out throughout the course. Activities D and E are carried out at the end of the course. Activity G is performed on the dates specified by the coordination.

Recovery Process
The student can present himself to the recovery whenever he has submitted to the activities B, D and G.

According to the coordination of the Degree and the direction of the School of Engineering, only the activities C, D and G can be recovered, they represent 55% of the final grade.

Review procedure
All evaluation activities can be reviewed in a place, date and time that will be published beforehand. Claims may be made on the activity grade, which will be evaluated by the faculty responsible for the subject. If the student does not show up for this revision, this activity will not be reviewed later.

Qualifications
An honor grade (MH) will be awarded to students who reach a final grade equal to or greater than 9.00, up to a total of 5% of the students enrolled, prioritizing the part of the grade that corresponds to individual assessments (activities B, D, F and G).
If the student does not reach the minimum grade in any of the evaluation activities, the final numerical grade will be the minimum between 4.5 points and the weighted sum of the grades of all the activities.
A student will be considered non-assessable (NA) if he / she has not submitted to any of activities B, D and G.

Irregularities by students, copy and plagiarism
Without prejudice to other disciplinary measures deemed appropriate, will be scored with a zero the irregularities committed by a student that may lead to a variation of the rating of an act of evaluation. Therefore, copying, plagiarism, cheating, letting copy, etc. in any of the evaluation activities will involve suspending it with a zero. The evaluation activities qualified in this way and by this procedure will not be recoverable. If it is necessary to pass any of these assessment activities to pass the subject, then this subject will be suspended directly, with no opportunity to recover it in the same course. In the case of not passing the subject due to the fact that the student has committed irregularities in an evaluation act, the numerical note of the record will be the lower value between 3.0 and the weighted average of the marks (and therefore the approved by compensation will not be possible).

Evaluation of repeating students
There is no special treatment for repeating students.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Assistance and active participation in class	5%	0	0	3, 10
Delivery of a performance engineering case report	5%	0	0	3
Delivery of a report on the resolution of practical problems in the laboratory	15%	0	0	3
Individual written examination of theory and problems	20%	1.9	0.08	1, 2, 9, 3, 5, 11, 12
Oral defense of Performance Engineering work	5%	0.1	0	3
Proposal and resolution of a practical case of Performance Engineering	30%	0	0	1, 2, 7, 9, 8, 4, 6, 5, 10, 11, 12
Resolution of practical problems in the laboratory	20%	0	0	2, 7, 9, 8, 4, 6, 5, 10, 11, 12

Bibliography

Computer Architecture: A Quantitative Approach. 6th Edition
(https://cataleg.uab.cat/iii/encore/record/C__Rb2034006)
John Hennessy, David Patterson, Morgan Kaufmann (Elsevier) 2019 (Cap. 4 i 5)

Structured Parallel Programming: Patterns for efficient computation
M. McCool, J. Reinders, A. Robison, Elsevier, 2012

Parallel Programming for Multicore and Cluster Systems
T. Rauber, G. Rünger, Springer (Elsevier), 2010

Programming Massively Parallel Processors: A Hands-on Approach
D. Kirk, & W.M. Hwu, Morgan Kaufmann (Elsevier), 2010

Software

None