Course guide
200231 - AIC - Algorithmics and Complexity

Last modified: 05/06/2025

Unit in charge: School of Mathematics and Statistics
Teaching unit: 723 - CS - Department of Computer Science.

Degree: BACHELOR'S DEGREE IN MATHEMATICS (Syllabus 2009). (Optional subject).

Academic year: 2025 **ECTS Credits:** 6.0 **Languages:** English

LECTURER

Coordinating lecturer: MARIA JOSE SERNA IGLESIAS

Others: Primer quadrimestre:
MARIA JOSE SERNA IGLESIAS - A

PRIOR SKILLS

This is an advanced course in algorithmics and complexity. Students are expected to have prior knowledge, at the second year level, of algorithmic techniques, programming and mathematical methods, particularly discrete mathematics and probability.

REQUIREMENTS

The students are expected to have some knowledge of the basic algorithmic techniques, divide and conquer, greedy, linear programming and dynamic programing. Also they are expected to have a mathematical maturity at the level of second year in the FME.

DEGREE COMPETENCES TO WHICH THE SUBJECT CONTRIBUTES

Specific:
3. CE-2. Solve problems in Mathematics, through basic calculation skills, taking in account tools availability and the constraints of time and resources.
5. Ability to solve problems from academic, technical, financial and social fields through mathematical methods.
GM-CE1. CE-1. Propose, analyze, validate and interpret simple models of real situations, using the mathematical tools most appropriate to the goals to be achieved.
GM-CE3. CE-3. Have the knowledge of specific programming languages and software.

Generical:
GM-CB1. CB-1. Demonstrate knowledge and understanding in Mathematics that is founded upon and extends that typically associated with Bachelor's level, and that provides a basis for originality in developing and applying ideas, often within a research context.
GM-CB2. CB-2. Know how to apply their mathematical knowledge and understanding, and problem solving abilities in new or unfamiliar environments within broader or multidisciplinary contexts related to Mathematics.
8. CG-3. Have the ability to define new mathematical objects in terms of others already know and ability to use these objects in different contexts.
GM-CB3. CB-3. Have the ability to integrate knowledge and handle complexity, and formulate judgements with incomplete or limited information, but that include reflecting on social and ethical responsibilities linked to the application of their knowledge and judgements.
9. CG-4. Translate into mathematical terms problems stated in non-mathematical language, and take advantage of this translation to solve them.

TEACHING METHODOLOGY

Two hours of theory class and two hours of presentation and discussion of problems by the students.
Students are expected to dedicate a certain number of hours per week to solving the problems proposed in class.

LEARNING OBJECTIVES OF THE SUBJECT

Provide a solid algorithmic basis to address the resolution of computational problems, both in a future professional job in industry or in the academy in the field of discrete mathematics or theoretical computer science.

Revise the basic techniques and data structures used to solve algorithmic problems: divide and conquer, voracious, dynamic programming, heaps, hashing, linear programming. Introduce new topics such as computational complexity, random techniques, approximate algorithms and parameterization.

STUDY LOAD

Type	Hours	Percentage
Hours small group	30,0	20.00
Hours large group	30,0	20.00
Self study	90,0	60.00

Total learning time: 150 h

CONTENTS

Introduction

Description:
Asymptotic notation, algorithms cost analysis. Review and consolidation of the algorithmic technique.

Specific objectives:
Examples of applications of the basic algorithmic techniques.

Related activities:
Problem solving.

Full-or-part-time: 13h
Theory classes: 4h
Practical classes: 4h
Self study : 5h

Computational complexity

Description:
Decidability and indecidability. The classes P, NP and NP-complete. Reductions. Examples of NP-complete problems .

Specific objectives:
Turing Machines, the Halting problem, The word problem. The classes P, NP and EXP. Clique, SAT and variants, Independenset, Vertex cover.

Full-or-part-time: 32h
Theory classes: 6h
Practical classes: 6h
Self study : 20h

Randomized algorithms. Modular arithmetic and primality

Description:
Introduction to randomized algorithms. Primality testing and applications.

Specific objectives:
Examples of random algorithms. Modular arithmetic, MCD, Random generation of prime numbers, Random algorithm for primality testing, Cryptography and RSA.

Related activities:
Problem solving.

Full-or-part-time: 23h
Theory classes: 4h
Practical classes: 4h
Self study : 15h

Approximation algorithms

Description:
Introduction to basic algorithmic techniques in approximation algorithms. Complexity classes and approximability limits.

Specific objectives:
Load balancing, Max cut, Knapsack, Traveling salesman problem. Integer and linear Programming, relax and round technique. Duality and approximation.

Related activities:
Problem solving.

Full-or-part-time: 41h
Theory classes: 8h
Practical classes: 8h
Self study : 25h

Parameterization

Description:
Introduction to the basic techniques to design parameterized algorithms. Parameterized complexity.

Specific objectives:
Parameters and complexity. Bounded search algorithms and kernelization. Graph parameters, treewidth.

Related activities:
Problem solving.

Full-or-part-time: 41h
Theory classes: 8h
Practical classes: 8h
Self study : 25h

GRADING SYSTEM

Two partial exams (P1, P2)
A final exam covering all the course (F).
Problem solving and presentation, participation (C)
The exam mark (E) is E = F, if you opt for doing the final exam, or (P1+P2)/2 otherwise.
Course mark: E*0.80+C*0.2

EXAMINATION RULES.

During the exams you will not be able to access any support material.

BIBLIOGRAPHY

Basic:

- Cormen, Thomas H., Leiserson, Charles Eric, Rivest, Ronald L., Stein, Clifford. Introduction to algorithms [on line]. 4th ed. Cambridge: MIT Press, cop. 2022 [Consultation: 24/05/2024]. Available on: https://web-p-ebscohost-com.recursos.biblioteca.upc.edu/ehost/ebookviewer/ebook?sid=5c0f1538-dfb6-47bd-8ecb-78fb365150f3%40redis&vid=0&format=EK. ISBN 0262367505.
- Sipser, Michael. Introduction to the theory of computation. 2nd ed. Boston: Thomson Course Technology, cop. 2006. ISBN 0534950973.
- Kleinberg, Jon; Tardos, Éva. Algorithm design. Boston: Pearson, 2014. ISBN 9781292023946.

Complementary:

- Moore, Cristopher; Mertens, Stephan. The Nature of computation. New York: Oxford University Press, cop. 2011. ISBN 9780199233212.
- Cygan, Marek; Saurabh, Saket; Pilipczuk, Marcin; Pilipczuk, Michal; Marx, Dániel; Lokshtanov, Daniel; Kowalik, Lukasz; Fomin, Fedor V. Parameterized algorithms. New York: Springer, 2015. ISBN 9783319212746.
- Vazirani, Vijay V. Approximation algorithms. Berlin: Springer, 2001. ISBN 9783540653677.
- Mitzenmacher, Michael; Upfal, Eli. Probability and computing: randomized algorithms and probabilistic analysis. Cambridge: Cambridge University Press, cop. 2005. ISBN 0521835402.