270213 - AP3 - Algorithmics and Programming III

Coordinating unit: 270 - FIB - Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science

Academic year: 2018

Degree: BACHELOR’S DEGREE IN DATA SCIENCE AND ENGINEERING (Syllabus 2017). (Teaching unit Compulsory)

ECTS credits: 6  
Teaching languages: Catalan

Prior skills

- Familiarity with the basic techniques of programming and the C++ programming language: iterations, alternatives, recursive functions, parameter passing, pointers, references, dynamic memory, classes, objects, methods, ...
- Knowledge of basic algorithmic concepts: efficiency of algorithms, asymptotic notation, graphs, graph traversal, data structures (lists, search trees, hash, heaps, ...) 
- Basic knowledge of discrete mathematics, linear algebra and calculus
- Basic knowledge of probability theory and statistics

Degree competences to which the subject contributes

Specific:
CE2. To be able to program solutions to engineering problems: Design efficient algorithmic solutions to a given computational problem, implement them in the form of a robust, structured and maintainable program, and check the validity of the solution.

General:
CG1. To design computer systems that integrate data of provenances and very diverse forms, create with them mathematical models, reason on these models and act accordingly, learning from experience.
CG2. Choose and apply the most appropriate methods and techniques to a problem defined by data that represents a challenge for its volume, speed, variety or heterogeneity, including computer, mathematical, statistical and signal processing methods.
CG5. To be able to draw on fundamental knowledge and sound work methodologies acquired during the studies to adapt to the new technological scenarios of the future.

Transversal:
CT4. Teamwork. Be able to work as a member of an interdisciplinary team, either as a member or conducting management tasks, with the aim of contributing to develop projects with pragmatism and a sense of responsibility, taking commitments taking into account available resources.
270213 - AP3 - Algorithmics and Programming III

Teaching methodology

The syllabus is explained in a practical way, through the presentation of many examples.

Theory lectures introduce all of the required concepts and techniques, which are put into practice in the problem and lab lectures by means of a collection of problems and exercises in an automatic judge.

The two hours of theory classes are taught weekly. The two hours of lab classes are taught every other week. The two hours of problem classes are taught every other week.

The project integrates the contents and competences of all the course.

The course uses the C++ programming language.

Learning objectives of the subject

1. Be aware of the limits of computation: to understand the implications of the question "P=NP?", understand the statement of Cook-Levin's Theorem, recognize and identify several classic NP-complete problems.
2. To know, explain, design, analyze, compare and implement exhaustive search algorithms using the backtracking technique.
3. To learn the dynamic programming scheme, identify when it can be applied and how, and be familiar with some fundamental dynamic programming algorithms.
4. To learn the scheme of greedy algorithms, identify when it can be applied and how, learn the most usual techniques for proving their correctness and be familiar with some fundamental greedy algorithms.
5. To learn approximation algorithms that efficiently compute approximate solutions (close to the optimal ones) for tractable optimization problems. To learn their limitations or problems that cannot be approximated in polynomial time.
6. To learn algorithms based on local search for solving untractable problems efficiently. To learn a variety of metaheuristics of different nature and to be able to identify when and how they can be applied on concrete computationally hard problems.
7. To complete and modify implementations of several algorithms for solving problems of average difficulty in the C++ programming language.
8. Identify and propose solutions to possible problems of efficiency in programs written in the C++ programming language.
9. To develop projects of average size as as member of a team, learning how to divide a project into smaller parts, to distribute them amongst its members and act with responsibility in a coordinated way for the successful accomplishment of the assigned tasks.

Study load

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
<th>Hours large group: 36h</th>
<th>24.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours small group: 24h</td>
<td></td>
<td>16.00%</td>
</tr>
<tr>
<td>Guided activities: 6h</td>
<td></td>
<td>4.00%</td>
</tr>
<tr>
<td>Self study: 84h</td>
<td></td>
<td>56.00%</td>
</tr>
</tbody>
</table>
## Content

### Tractability: classes of problems P and NP

**Degree competences to which the content contributes:**

**Description:**
Classes P and NP, Cook-Levin's Theorem, reductions, NP-completeness.

### Exhaustive search

**Degree competences to which the content contributes:**

**Description:**

### Dynamic programming

**Degree competences to which the content contributes:**

**Description:**

### Greedy algorithms

**Degree competences to which the content contributes:**

**Description:**
Theoretical foundations: general scheme of greedy algorithms. Examples: task scheduling, subset sum, Huffman codes, algorithms of Bellman-Ford and Johnson for shortest paths, algorithms of Kruskal and Prim for minimum spanning trees.

### Approximation algorithms

**Degree competences to which the content contributes:**

**Description:**
Theoretical foundations: theoretical notion of approximability, non-approximable problems. Examples: vertex cover, partition, travelling salesman, knapsack, Max 3-SAT.

### Metaheuristics

**Degree competences to which the content contributes:**
Description:
Constructive procedures. Local search. Metaheuristics: GRASP, Simulated Annealing, Tabu Search, Genetic algorithms, Ant colony, Path Relinking, etc. Applications to computationally complex problems.
## Planning of activities

<table>
<thead>
<tr>
<th>Block</th>
<th>Hours</th>
<th>Theory classes: 6h</th>
<th>Practical classes: 4h</th>
<th>Laboratory classes: 0h</th>
<th>Guided activities: 0h</th>
<th>Self study: 8h</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tractability</strong></td>
<td><strong>18h</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Exhaustive Search</strong></td>
<td><strong>16h</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Dynamic Programming</strong></td>
<td><strong>16h</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Greedy Algorithms</strong></td>
<td><strong>16h</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Specific objectives:

- **Tractability:** 1
- **Exhaustive Search:** 3, 8, 9
- **Dynamic Programming:** 4, 8, 9
- **Greedy Algorithms:** 5, 8, 9
### Approximation Algorithms

- **Specific objectives:** 6

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 4h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 4h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 8h</td>
</tr>
</tbody>
</table>

### Metaheuristics

- **Specific objectives:** 7, 8

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 4h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 6h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 8h</td>
</tr>
</tbody>
</table>

### Consolidation

- **Specific objectives:** 1, 3, 4, 5, 6, 7

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 4h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 8h</td>
</tr>
</tbody>
</table>

### Mid term exam

- **Specific objectives:** 1, 3, 4

<table>
<thead>
<tr>
<th>Hours</th>
<th>Guided activities: 3h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

### Project

- **Specific objectives:** 3, 4, 5, 6, 7, 8, 9, 10

<table>
<thead>
<tr>
<th>Hours</th>
<th>Guided activities: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Self study: 28h</td>
</tr>
</tbody>
</table>
## Final exam

<table>
<thead>
<tr>
<th>Hours: 3h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 3h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
1, 3, 4, 5, 6, 7

## Lab exam

<table>
<thead>
<tr>
<th>Hours: 3h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 3h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
3, 4, 5, 6, 7, 8, 9

## Qualification system

NPar = grade mid term exam  
NFT = grade final theory exam  
NFL = grade final lab exam  
NPro = grade project

FINAL GRADE = max( 30% Npar + 30% NFT + 20% NFL + 20% NPro, 60% NFT + 20% NFL + 20% NPro)

The grade of the reevaluation exam, if there is any, replaces the grade of the theory final exam (NFT). The grades of mid term, project and lab (NPar, NFL, NPro) are preserved.
Bibliography

Basic:


Complementary:


Others resources:

Hyperlink

https://jutge.org/