270027 - AA - Advanced Algorithmics

Coordinating unit: 270 - FIB - Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2017
Degree: BACHELOR'S DEGREE IN INFORMATICS ENGINEERING (Syllabus 2010). (Teaching unit Optional)
ECTS credits: 6

Teaching languages: Catalan

Teaching staff

Coordinator: Maria Jose Serna Iglesias (mjserna@cs.upc.edu)
Others: Josep Diaz Cort (diaz@cs.upc.edu)
- Maria Del Carme Alvarez Faura (alvarez@cs.upc.edu)

Prior skills

Knowledge of algorithms and related concepts: efficiency of algorithms, asymptotic notation, greedy algorithms, dynamic programming, ...

Basic knowledges of the theory of computation: automata, grammars, Turing machines, decidibilitat, complexity.

Critical capacity.

Mathematical maturity.

Requirements

- Prerequisite A

Degree competences to which the subject contributes

Specific:

CCO1.1. To evaluate the computational complexity of a problem, know the algorithmic strategies which can solve it and recommend, develop and implement the solution which guarantees the best performance according to the established requirements.

CCO2.5. To implement information retrieval software.

CCO2.6. To design and implement graphic, virtual reality, augmented reality and video-games applications.

General:

G7. AUTONOMOUS LEARNING: to detect deficiencies in the own knowledge and overcome them through critical reflection and choosing the best actuation to extend this knowledge. Capacity for learning new methods and technologies, and versatility to adapt oneself to new situations.
270027 - AA - Advanced Algorithmics

Teaching methodology

The theoretical contents of the course is taught in theory classes.

In the classes of problems students solve problems with the help of the professor and also present some of their solutions on the board.

Students have to deliver three written submissions presenting the solution of problems assigned by the professor. Depending on the difficulty of the problems, solutions may be presented in groups of 2 or 3 people. In some cases, it will be necessary to use methods that complement the ones seen in theory class and it will require some bibliographic research. In these cases the students will be asked to present solutions in public during the problem sessions.

Learning objectives of the subject

1. To know the fundamental concepts of Computational Problem and Algorithm. To be able to analyze the computational resources like Time and Space, which are required by an algorithm.
2. To know how to classify the complexity of a computational problem. To be able to distinguish between tractable problems and intractable problems. Knowing the techniques of reducibility and completeness to analyze the computational difficulty of a problem.
3. To know some classical intractable problems and the classes that are identified by these problems: NP, PSPACE, EXP i NEXP.
4. To know Random Algorithms to solve intractable problems. In particular, to know two varieties of random algorithms: Monte Carlo algorithms which compute in polynomial time a solution that it may be not correct with low probability and Las Vegas algorithms which always compute a correct solution and guarantee polynomial time with high probability.
5. To know Approximation Algorithms to compute efficiently approximate solutions (solutions close to the optimum) for optimization intractable problems. Knowing their limitations or problems that can not be approximated in polynomial time.
6. To know Local Search Algorithms to solve efficiently intractable problems. To know the paradigm inspired by the evolution that aims to select an optimum solution from a reduced set of solutions or neighborhood, in polynomial time.
7. To know Fixed Parameter Algorithms that allow to solve in polynomial time certain restrictions of intractable problems. To know how to identify specific parameters of a problem so that when they are fixed then the problem can be solved efficiently.

Study load

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
<th>Theory classes: 45h</th>
<th>30.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 15h</td>
<td>10.00%</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 0h</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 6h</td>
<td>4.00%</td>
</tr>
<tr>
<td></td>
<td>Self study: 84h</td>
<td>56.00%</td>
</tr>
</tbody>
</table>

Study load:

The theoretical contents of the course is taught in theory classes.

In the classes of problems students solve problems with the help of the professor and also present some of their solutions on the board.

Students have to deliver three written submissions presenting the solution of problems assigned by the professor. Depending on the difficulty of the problems, solutions may be presented in groups of 2 or 3 people. In some cases, it will be necessary to use methods that complement the ones seen in theory class and it will require some bibliographic research. In these cases the students will be asked to present solutions in public during the problem sessions.

Learning objectives of the subject

1. To know the fundamental concepts of Computational Problem and Algorithm. To be able to analyze the computational resources like Time and Space, which are required by an algorithm.
2. To know how to classify the complexity of a computational problem. To be able to distinguish between tractable problems and intractable problems. Knowing the techniques of reducibility and completeness to analyze the computational difficulty of a problem.
3. To know some classical intractable problems and the classes that are identified by these problems: NP, PSPACE, EXP i NEXP.
4. To know Random Algorithms to solve intractable problems. In particular, to know two varieties of random algorithms: Monte Carlo algorithms which compute in polynomial time a solution that it may be not correct with low probability and Las Vegas algorithms which always compute a correct solution and guarantee polynomial time with high probability.
5. To know Approximation Algorithms to compute efficiently approximate solutions (solutions close to the optimum) for optimization intractable problems. Knowing their limitations or problems that can not be approximated in polynomial time.
6. To know Local Search Algorithms to solve efficiently intractable problems. To know the paradigm inspired by the evolution that aims to select an optimum solution from a reduced set of solutions or neighborhood, in polynomial time.
7. To know Fixed Parameter Algorithms that allow to solve in polynomial time certain restrictions of intractable problems. To know how to identify specific parameters of a problem so that when they are fixed then the problem can be solved efficiently.
<table>
<thead>
<tr>
<th>Content</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Problems and Algorithms</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
</tr>
</tbody>
</table>
| **Description:** | Computational problems.  
Complexity of algorithms: Time and Space.  
Complexity of problems.  
Tractable problems: Accessibility, Shortest paths.  
Intractable problems: Traveling Salesman Problem, Knapsack. |
| **Intractable Problems** |  |
| **Degree competences to which the content contributes:** |  |
| **Description:** | Reducibility and Completeness.  
NP-complete problems: Satisfiability, Subgraphs, Colorability, Tours, Partitions, Scheduling.  
PSPACE, EXP, and NEXP problems: Quantified boolean formulae, games, tiling, equivalence of regular expressions. |
| **Random Algorithms** |  |
| **Degree competences to which the content contributes:** |  |
| **Description:** | Monte-Carlo Algorithms.  
Las Vegas Algorithms.  
Generation of random numbers.  
Factorization (Heuristic Pollard Rho).  
Cryptography (RSA) |
| **Approximation Algorithms** |  |
| **Degree competences to which the content contributes:** |  |
| **Description:** | Optimization Problems and Approximability.  
Vertex Cover.  
Traveling Salesman Problem.  
Knapsack.  
Random Approximation Algorithm for MAX 3-SAT.  
Approximability Hierarchy. |
| **Local Search Algorithms** |  |
| **Degree competences to which the content contributes:** |  |
Fixed Parameter Algorithms

Degree competences to which the content contributes:
Description:
Parameterized problems: Vertex cover, Max Sat, Knapsack.
Algorithmic Methods: Data reduction, Bounded-depth Search trees, Dynamic Programming.

Algorithms and Internet: Modelling Internet

Degree competences to which the content contributes:
Description:
Basic definitions: Game, strategy, cost and payoff, selfish players.
Nash Equilibrium, Social Cost, Price of Stability and Price of Anarchy
Introduction to Network Formation Games. Understanding the behavior of Internet: A game equilibrium.
### Planning of activities

<table>
<thead>
<tr>
<th>Delivery of problems: Solutions to Intractable problems (I)</th>
<th>Hours: 3h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 3h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
1, 2, 5, 6

<table>
<thead>
<tr>
<th>Final Exam</th>
<th>Hours: 11h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Guided activities: 3h</td>
</tr>
<tr>
<td></td>
<td>Self study: 8h</td>
</tr>
</tbody>
</table>

**Description:**
Written exam.

**Specific objectives:**
1, 2, 4, 5, 6, 7, 8, 9

<table>
<thead>
<tr>
<th>Delivery of problems: Solutions to Intractable problems (II)</th>
<th>Hours: 3h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 3h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
1, 2, 4, 5, 6, 7, 8, 9

<table>
<thead>
<tr>
<th>Delivery of problems: Intractability</th>
<th>Hours: 4h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 4h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
1, 2, 4

<table>
<thead>
<tr>
<th>Learning the topic “Problems and Algorithms”</th>
<th>Hours: 8h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 3h</td>
</tr>
<tr>
<td></td>
<td>Practical classes: 1h</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: 4h</td>
</tr>
</tbody>
</table>

**Description:**
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.

**Specific objectives:**
1
## Learning the topic "Intractable Problems"

**Hours:** 28h  
Theory classes: 9h  
Practical classes: 3h  
Laboratory classes: 0h  
Guided activities: 1h  
Self study: 15h

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.

**Specific objectives:**  
2, 4

## Learning the topic "Random Algorithms"

**Hours:** 18h  
Theory classes: 6h  
Practical classes: 2h  
Laboratory classes: 0h  
Guided activities: 0h  
Self study: 10h

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.

**Specific objectives:**  
5

## Learning the topic "Approximation Algorithms"

**Hours:** 20h 30m  
Theory classes: 7h  
Practical classes: 3h  
Laboratory classes: 0h  
Guided activities: 0h 30m  
Self study: 10h

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.

**Specific objectives:**  
6
### Learning the topic "Local Search Algorithms"

**Hours:** 18h 30m  
- Theory classes: 7h  
- Practical classes: 2h  
- Laboratory classes: 0h  
- Guided activities: 0h 30m  
- Self study: 9h  

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.  

**Specific objectives:** 7

### Learning the topic "Fixed Parameter Algorithms"

**Hours:** 18h 30m  
- Theory classes: 7h  
- Practical classes: 2h  
- Laboratory classes: 0h  
- Guided activities: 0h 30m  
- Self study: 9h  

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.  

**Specific objectives:** 8

### Learning the topic "Algorithmics and Game Theory: Modelling Internet"

**Hours:** 17h 30m  
- Theory classes: 6h  
- Practical classes: 2h  
- Laboratory classes: 0h  
- Guided activities: 0h 30m  
- Self study: 9h  

**Description:**  
Students attend the theory classes, try to understand this subject and solve problems, asking professor for help in the class of problems. Furthermore the students can also be asked to present one of the assigned problems to the blackboard.  

**Specific objectives:** 9
270027 - AA - Advanced Algorithmics

Qualification system

There are two types of evaluation activities: Delivery of problems and Final Exam.

Delivery of problems:
This part consist of solving lists of problems that have been assigned to each student (or each small group work) as indicated in the plan. In the class of problems, the students can discuss their doubts together jointly with the professor, but it is considered as a personal and autonomous work (or groups) that must be completed during their time of study. In general the solution will require to apply the acquired knowledge, to choose the appropriate method in each case and also to do some bibliographic research. The students deliver their written solutions and present them in public if it is deemed appropriate (when the solutions extend the knowledge introduced in the current issue and especially when the work is in group). The self-learning will be evaluated by this work.

The note $P$ of the delivery of problems is the average grade of all deliveries.

Exam:
There will be a final exam to evaluate whether the student knows the more important topics studied during all the course.

The final grade for the course is calculated from the note of the problems $P$ and the final exam grade $E$ as follows:

Final Mark = 0.3 $P + 0.7 E$

The evaluation of competence G7.3 will be carried out individually for each student based on public presentations and written solutions to the assigned problems.

The assessment of competence G7.3 does not affect the evaluation of the course.

Bibliography

Basic:

Complementary: