UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

COURSE SYLLABUS

Design and Analysis of Algorithms

2627-3-E3101Q113

Aims

Students will acquire knowledge of the main techniques for the design and analysis of algorithms and the ability to identify the most appropriate algorithmic techniques to efficiently solve specific computational problems.

Knowledge and understanding

This course provides basic knowledge and understanding on:

- Dynamic Programming (DP) algorithmic technique for solving combinatorial optimization problems
- paradigmatic algorithms for combinatorial optimization problems over sequences, sets, and graphs (Longest Common Subsequence, Weighted Interval Scheduling, ..., and their variants) based on the dynamic programming technique, optimal substructure property
- Greedy algorithmic tecnique for solving combinatorial optimization problems
- Disjoint-set data structure and computational complexity of the related algorithms on the basis of the representation of the data structure and the way of implementing its operations.
- Algorithms for computing minimum spanning trees (Kruskal and Prim algorithms)
- Algorithms for computing shortest path on graphs (Floyd-Warshall and Dijkstra algorithms)
- direct-address tables and Hash tables
- the classes P, NP and NP-complet problems
- reducibility between problems

Ability to apply knowledge and understanding

Ability to solve combinatorial optimization problems (and their decision versions) over sequences, sets, and graphs by the dynamic Programming technique, being able to:

- define the sub-problems, the related coefficients (variables) data structure for storing them,
- reformulate the problem in recursive terms providing the base case and recursive step of the recurrence equations

- provide the optimal value of the problem on the basis of the values of all coefficients
- design an efficient bottom-up algorithm for computing the values of all coefficients and the optimal value
- design an efficient algorithm for reconstructing a solution to the problem (subsequence, subset or paths of optimal value)

Ability to understand whether and when the greedy technique can be successfully employed for solving a combinatorial optimization problem.

Abilty to use disjoint-set data structure as far as graph algorithms are concerned.

Ability to build the instance of a problem starting from the instance of the problem from which the former has been reduce.

Making judgements

Ability to identify the most suitable algorithmic technique and data structures for efficiently solving specific computational problems.

Communication skills

Ability to explain in a clear and rigorous way, the theoretical contents, the algorithmic techniques and the algorithms discussed, including the proofs.

Learning Skills

Ability to independently search for and learn new algorithms and data structures to solve computational problems. Tackle new computational problems.

Contents

The course will introduce the main algorithmic techniques (dynamic programming, greedy), with particular attention to the efficiency of the algorithms, with the main analysis methods. The main algorithms for several combinatorial optimization problems, especially over sets, sequences, and graphs will be presented, including minimum spanning trees construction and shortest path problems.

Detailed program

1. Mathematical tools (review)

- Growth of functions, asymptotic notations
- Execution time of iterative algorithms
- Recursion and recursive algorithms
- Recurrence equations and Execution times of recursive algorithms

2. Algorithmic Techniques: Dynamic Programming (DP)

- Introductory examples
- Main features - Recursion and optimal substructure property
- Implementation with matrices

- Combinatorial optimization problems over sequences, sets and graphs. Decision Variants. Optimal substructure property
- Resolution by: definition of coefficients (variables) associated with the sub-problems and related data structure for storing them, formulation of recurrence equations (base case and recursive step) for computing the values of all coefficients, identification of the optimal value on the basis of the values of all coefficients, bottom-up algorithm for computing the values of all coefficients and the optimal value, algorithm for reconstructing a solution to the problem (subsequence, subset or paths of optimal value).

3. Algorithmic Techniques: Greedy method

- Introductory examples
- Similarities and differences with the dynamic programming technique
- Independence systems and Matroids
- Optimization (maximum/minimum) problem associated with a weighted independence system and related greedy algorithm
- Rado Theorem
- Graphic Matroid

4. Disjoint-set data structure

- Definitions and operations
- Linked list representation and forest representation

5. Minimum spanning trees

- Generic algorithm
- Kruskal algorithm
- Prim algorithm

6. Shortest path problems

- Dijkstra Algorithm
- Floyd-Warshall Algorithm

7. Hash Tables

- Direct-address tables
- Hash tables

8. Introduction to NP-completeness and reducibility

- The class P, NP and NP-complet problems
- Reducibility between problems: reducibility notion and various examples.

Prerequisites

Basic notions of programming, algorithms and data structures

Teaching form

Lectures, practice exercises, and classroom laboratory exercises all in presence.

Lectures (8 three-hour lectures and 4 two-hour lectures, for a total of 32 hours) will be carried out in unidirectional lecture mode.

Practice exercises activities (10 two-hour activities, for a total of 20 hours) and classroom laboratory exercises acivities (8 three-hour activities, for a total of 24 hours) will be carried out with an initial part in unidirectional mode and a second part in interactive mode.

At the end of each lesson, a self-assessment activity is provided for students, focusing on their understanding of the topics covered.

The course is in Italian.

Textbook and teaching resource

T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduzione agli Algoritmi e Strutture dati, Ed. Mc. Graw Hill

Further material and exercises are available through the e-learning website.

Semester

First semester

Assessment method

Written examination: It consitsts of two parts

- Part 1: exercises related to the main topics (maximum score of 31)

- Part 2: open questions on the theoretical aspects of the topics explained in the course (maximum score of 31)

The exam is passed if and only if each part has a score of at least 15 and the average of the two scores is at least 18.

This average will correspond to the final grade on a 30-point scale (with 30 cum laude awarded for scores above 30).

During the January and February exam sessions, the final grade may be increased by 0 to 3 points (based on the answer to an additional optional question/exercise), but only if the Part 2 score is not lower than 25.

The final grade may be further increased by 0 to 3 points based on the results of all self-assessment tests completed by students at the end of each lesson.

Partial written examinations:
The two parts of the written exam may be taken separately in two partial exams, held halfway through the course

(Part 1) and during the January exam session (Part 2).

Office hours

By appointment

Sustainable Development Goals