Module Description

Algorithms

General Information

Number of ECTS Credits

3

Module code

TSM_Alg

Responsible of module

Éric Taillard, HES-SO

Language

Explanations regarding the language definitions for each location:

- Instruction is given in the language defined below for each location/each time the module is held.
- Documentation is available in the languages defined below. Where documents are in several languages, the percentage distribution is shown (100% = all the documentation).
- The examination is available 100% in the languages shown for each location/each time it is held.

	Berne	Lausanne			Lugano	Zurich		
Instruction	☐ E 100%	☐ E 100%		☒ F 100%	☐ E 100%	☒ E 100%		☐ D 100%
Documentation	☐ E 100%	☐ E 100%	☒ E 50%	☒ F 50%	☐ E 100%	☒ E 100%	☐ E %	☐ D %
Examination	☐ E 100%	☐ E 100%	☐ E 100%	☒ F 100%	☐ E 100%	☒ E 100%	☐ E 100%	☐ D 100%

Module category

☐ FTP Fundamental theoretical principles

☒ TSM Technical/scientific specialization module

☐ CM Context module

Lessons

2 lecture periods and 1 tutorial period per week

Entry-level competencies

Prerequisites, previous knowledge

The student has working knowledge of:

- Geometry, linear algebra, algorithms (sorting, searching, hashing) and data structures (linear structures, trees)
- Basics of graph data structures and algorithms
- Algorithmic complexity, logic, probability theory.

These topics are generally contained in books introducing algorithms. For instance, chapters 1-12, 15-17, 22-26, 28-29, 34-35 of (Cormen 09) covers very well the prerequisites

Brief course description of module objectives and content

This module introduces students with different categories of advanced algorithms and typical application areas.
In the first part of the module, the students will have a sound understanding of data structures and algorithms for efficiently handling either very large, complex or dynamic data sets or combinations thereof. They will be able to evaluate suitable algorithms and to apply them to typical tasks such as efficiently indexing, searching, retrieving, inserting or updating data such as large volumes of hypertext or spatial data.
The students will be familiar with dynamic algorithms used, for example, in artificial intelligence or molecular sciences.
The second part of the module will present basic techniques for designing algorithms for hard combinatorial optimization problems. The combination of these basic components —problem modeling, problem decomposition, solution building, solution improvement, learning— lead to classical metaheuristics like genetic algorithms, ant colonies or tabu search. The students will be able to design new algorithms for hard combinatorial optimization problems and to apply them.

Aims, content, methods

Learning objectives and acquired competencies

This module gives the student an overview of frequently used algorithms classes. Based on this strong foundation, students can design and implement the most suitable and efficient algorithms for use in their own applications. The student:

- is familiar with different categories of advanced algorithms and with typical application areas;
- has a sound understanding of data structures and algorithms for efficiently handling either very large, complex or dynamic data sets or combinations thereof;
- is able to evaluate suitable algorithms and to apply them to typical tasks such as efficiently indexing, searching, retrieving, inserting or updating data, including data types such as large volumes of hypertext or spatial data;
- is familiar with dynamic algorithms used in robotics, artificial intelligence or molecular sciences.

Contents of module with emphasis on teaching content

Computational Geometry and Multi-dimensional Data Structures. Weight 50%

- geometric algorithms
- computational geometry algorithms
- multidimensional data structures and algorithms

Metaheuristic-based algorithms. Weight 50%

- Constructive methods
- Local searches
- Decomposition techniques
- Learning techniques
- Classical metaheuristics: GRASP, ant colonies, tabu search, simulated annealing, noising methods, genetic algorithms

Part I Computational geometry Reminder: • Asymptotical notation • Complexity of recursive algorithms • Basic data structures (table, list, stack, queue, binary tree, heap, hashing table) • Basic algorithms (sorting, 1D indexing and searching) • Graphs and networks, planar graphs, doubly-connected edge list (DCEL) • Linear algebra (point, vector, dot product, cross product)
Introduction to Computational Geometry • Introductory problem : visibility map, polygons and boolean operations, line intersection, numerical problems • Basic objects and primitives (points, segments, polygons, rays, lines) • Existing CG software and libraries.
Construction paradigms • Incremental construction (line arrangements, line segment intersection, overlay problem) • Divide and conquer (convex hull for points in the plane) • Plane sweep technique (closest pair problem, intersections detection in segments sets)
Range Searching • Kd and Range trees for orthogonal range searching (example of "multi-level structure") • Quadtrees
Windowing • Interval tree for horizontal line segments • Priority search tree
Voronoï Diagram, Delaunay triangulation • Computation of Voronoï Diagram • Terrain and randomized algorithm for Delaunay triangulation
Part II Metaheuristics Introduction and reminder: • Basic problems and algorithms for graphs and networks • Optimal trees, paths and flows, linear assignment • Combinatorial optimization, complexity theory, problem modelling and utility function • Hard problems: Travelling salesman, Steiner tree, Quadratic assignment, Graph colouring, Scheduling
Constructive methods • Random building • Greedy construction • Beam search, Pilot method
Local searches • Neighbourhood structure, moves, 2-opt and 3-opt for the travelling salesman problem • Neighbourhood limitation • Neighbourhood extension, Lin-Kernighan neighbourhood for the travelling salesman problem, fan and filter
Randomized methods • Threshold accepting, great deluge and demon algorithms • Simulated annealing • Noising methods • GRASP • Variable neighbourhood search
Decomposition methods • Large neighbourhood search, POPMUSIC
Learning methods for solution building • Artificial ant systems • Vocabulary building
Learning methods for iterative local search • Tabu search
Methods with a population of solutions • Genetic algorithms • Scatter Search • GRASP with path relinking

Teaching and learning methods

- Ex-cathedra teaching
- Presentation and discussion of case studies
- Theory and programming exercises

Literature

M. de Berg, G. Cheong, M. van Kreveld, M. Overmars. *Computational Geometry : Algorithms and Applications*, Springer, Third Edition 2008.
T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*, third edition, MIT Press, 2009.
P. Siarry (ed.), *Metaheuristics*, ISBN 978-3-319-45403-0, Springer, 2016.
H. H. Hoos, Th. Stützle *Stochastic Local Search: Foundations and Applications,* Morgan Kaufmann / Elsevier, 2004.
É. Taillard *Introductions aux méta-heuristiques,* WWW, 2015

Assessment

Certification requirements for final examinations (conditions for attestation)

Basic principle for exams:
All the standard final exams for modules are written exams.
The repetition exams can be either written or oral.

Standard final exam for a module and written repetition exam

Kind of Exam	written
Duration of exam	120 minutes
Permissible aids	☐ No aids
	☑ Permissible aids:
	☑ Electronical aids: None
	☑ Hardcopy form: Books, copy of slides (solutions to exercises excluded)
	☐ _____

Special case: Repetition exam as an oral exam

If an oral exam is set (only possible for ≤ 4 students), the following applies:

Kind of Exam	oral
Duration of exam	30 minutes
Permissible aids	No aids