Module Description, available in: EN, FR

Algorithms

General Information

Number of ECTS Credits

3

Module code

FTP_Alg

Valid for academic year

2020-2021

Last modification

2020-01-06

Responsible of module

Eric Taillard (HES-SO, eric.taillard@heig-vd.ch)

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			**X** F 100%			**X** E 100%		
Documentation			**X** F 50%	**X** E 50%		**X** E 100%		
Examination			**X** F 100%			**X** E 100%		

Module Category

FTP Fundamental theoretical principles

Lessons

2 lecture periods and 1 tutorial period per week

Entry level competences

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.
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.

Contents of module with emphasis on teaching content

Algorithms and data structures for selected practical problems. Weight 50%

- Reminder on linear structures, binary search tree, heap and algorithmic complexity
- Randomized algorithms (qsort, selection in linear time, kD tree, range tree, Delaunay triangulation)
- Deterministic algorithms(selection, kD tree)
- Sweep algorithms (closest pair of points, segment intersection)
- Greedy algorithms (Huffman code, Dijkstra, Fibonacci heap)

Design of heuristic algorithms. Weight 50%

- Reminder on complexity theory, combinatorial optimization problems, problem modelling
- Greedy heuristics
- Local search heuristics
- Decomposition methods
- Randomized heuristics
- Learning heuristics
- Classical metaheuristics: GRASP, ant colonies, tabu search, simulated annealing, noising methods, genetic algorithms

Teaching and learning methods

- Ex-cathedra teaching
- Presentation and discussion of case studies
- Problem-based learning
- 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, 2015-2020

Assessment

Certification requirements

Module does not use certification requirements

Basic principle for exams

As a rule, all the standard final exams for modules and also all repetition exams are to be in written form

Standard final exam for a module and written repetition exam

Kind of exam

written

Duration of exam

120 minutes

Permissible aids

Aids permitted as specified below:

Permissible electronic aids

No electronic aids permitted

Other permissible aids

Books, copy of slides (solutions to exercises excluded)

Special case: Repetition exam as oral exam

Kind of exam

oral

Duration of exam

30 minutes

Permissible aids

No aids permitted

Description du module, disponible en: EN, FR

Algorithmique

Informations générales

Nombre de crédits ECTS

3

Code du module

FTP_Alg

Valable pour l'année académique

2020-2021

Dernière modification

2020-01-06

Nom du/de la responsable de module

Eric Taillard (HES-SO, eric.taillard@heig-vd.ch)

Explication des définitions de langue par lieu :

- Les cours se dérouleront dans la langue définie ci-dessous par lieu/exécution.
- Les documents sont disponibles dans les langues définies ci-dessous. Pour le multilinguisme, voir la répartition en pourcentage (100% = documents complets)
- L'examen est disponible à 100% dans chaque langue sélectionnée pour chaque lieu/exécution.

	Berne	Lausanne			Lugano	Zurich		
Leçons			**X** F 100%			**X** E 100%		
Documentation			**X** F 50%	**X** E 50%		**X** E 100%		
Examen			**X** F 100%			**X** E 100%		

Catégorie de module

FTP bases théoriques élargies

Leçons

2 leçons et 1 leçon de pratique par semaine

Compétences préalables

Connaissances préalables, compétences initiales

L'étudiant dispose de connaissances pratiques en :

- géométrie, algèbre linéaire, algorithmes (classification, recherche, hachage) et structures de données (structures linéaires, structures arborescence)
- connaissances de base de la théorie des graphes (structures de données et algorithmes)
- complexité algorithmique, logique, théorie des probabilités

Ces connaissances font partie de tout bon livre d'introduction sur les algorithmes. Par exemple, les chapitres 1-12, 15-17, 22-26, 28-29, 34-35 de Cormen (2009) couvrent parfaitement les pré-requis de ce cours.

Brève description du contenu et des objectifs

Ce module présente différentes catégories d'algorithmes avancés ainsi que leurs domaines d'application typiques.
La première partie du module approfondira les connaissances sur les structures de données qui permettent de gérer efficacement des ensembles de données très grands, complexes ou dynamiques, voire combinant ces trois caractéristiques. À l'issue du module, les étudiants seront capables de sélectionner le meilleur algorithme et de l'appliquer à des tâches telles que l'indexation, la recherche, l'extraction, l'insertion ou la mise à jour de données volumineuses, comme celles que l'on rencontre dans les systèmes d'informations géographiques, les hypertextes ou en intelligence artificielle.
La seconde partie du module présentera des techniques de base pour la conception d'algorithmes appliqués à des problèmes d'optimisation difficiles. La combinaison de ces techniques de base ?modélisation, construction de solution, amélioration de solution, décomposition de problème, apprentissage? conduit à des métaheuristiques classiques comme les algorithmes génétiques, les colonies de fourmi artificielles ou la recherche avec tabous. À l'issue du module, les étudiants seront capables de concevoir et d'appliquer ces techniques à des problèmes d'optimisation difficiles.

Objectifs, contenus, méthodes

Objectifs d'apprentissage, compétences à acquérir

Ce module donne une vue d'ensemble diverses classes d'algorithmes fréquemment utilisés en pratique. Sur la base de connaissances solides, l'étudiant est capable de concevoir et d'implanter les algorithmes les plus efficaces et les plus appropriés pour ses propres applications. À l'issue du module, l'étudiant :

- connaît différentes catégories d'algorithmes avancés et leurs domaines d'application ;
- a une bonne connaissance en structures de données avancées et leur utilisation pour gérer efficacement des données volumineuses, complexes et/ou dynamiques ;
- est en mesure d'évaluer quels algorithmes sont appropriés pour réaliser efficacement certaines tâches telles que l'indexation, la recherche, l'extraction, l'insertion ou la mise à jour de données volumineuses ;
- connaît les algorithmes dynamiques utilisés en robotique ou en intelligence artificielle

Contenu des modules avec pondération du contenu des cours

Algorithmes et structures de données pour quelques problèmes spécifiques. Poids 50%

- Rappels sur les structures linéaires, les arbres de recherche binaires, les tas et la complexité algorithmique
- Algorithmes randomisés (tri rapide, sélection en temps linéaire, arbre kD, arbre de requête d'énumération, triangulation de Delaunay)
- Algorithmes déterministes (sélection en temps linéaire, arbre kD)
- Algorithmes de balayage (plus proche paire de points, intersection de segments)
- Algorithmes gloutons (codage de Huffman, Dijkstra, tas de Fibonacci)

Conception d'algorithmes heuristiques. Poids 50%

- Rappels de théorie de la complexité, de modélisation de problèmes, d'optimisation combinatoire
- Algorithmes gloutons
- Heuristiques de recherche locale
- Méthodes de décomposition
- Heuristiques randomisées
- Heuristiques d'apprentissage
- Métaheuristiques classiques: GRASP, colonies de fourmis artificielles, recherche tabou, recuit simulé, méthodes de bruitage, algorithmes génétiques

Méthodes d'enseignement et d'apprentissage

- Enseignement ex-cathedra
- Présentation et discussion d'études de cas
- Apprentissage par résolution de problème
- Exercices théoriques et de programmation

Bibliographie

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.

Traduction française : Introduction à l'algorithmique, 3e édition, Dunod, 2010.

P. Siarry (ed.), Métaheuristiques, EAN13 : 9782212139297, Eyrolles 2014.

H. H. Hoos, Th. Stützle Stochastic Local Search: Foundations and Applications, Morgan Kaufmann / Elsevier, 2004.

É. Taillard Introductions aux méta-heuristiques, WWW, 2015-2020

Evaluation

Conditions d'admission

Le module n'utilise pas de conditions d'admission.

Principe pour les examens

En règle générale, tous les examens de fin de module réguliers et les examens de rattrapage sont organisés sous la forme écrite

Examen de fin de module régulier et examen écrit de répétition

Type de l'examen

écrit

Durée de l'examen

120 minutes

Aides autorisées

Les aides suivantes sont autorisées:

Aides électroniques autorisées

Aucun

Autres aides autorisées

Livres, copie des transparents (sauf solutions aux exercices)

Cas spécial: examen de répétition oral

Type de l'examen

oral

Durée de l'examen

30 minutes

Aides autorisées

Sans aides