Algorithmique et complexité

ECTS
9 crédits

Composante
UFR
Mathématiques

Volume horaire
84h

En bref

Forme d'enseignement : Cours magistral, Travaux dirigés & Travaux pratiques

Ouvert aux étudiants en échange: Non

Présentation

DESCRIPTION

Consolidation des connaissances en algorithmique, connaissance des rudiments de la complexité et des approches algorithmiques classiques.

Le cours est en partie mutualisé avec le master Math-Info.

* les étudiants du M1 mathématiques suivent les deux parties du cours (sur 12 semaines)
* les étudiants du M1 mathématiques-informatique suivent la seconde partie du cours (sur les 8 dernières semaines). Les étudiants de cette filière peuvent néanmoins suivre la première partie s'ils le désirent.

Algorithmique

* Algorithmes (tris, ...) et méthodes algorithmiques (diviser pour régner, recherche exhaustive, programmation dynamique, algorithmes gloutons) de bases.
* Structures de données classiques : liste, file, pile, arbre, graphes, table de hachage.
* Graphes, algorithmes classiques de base : parcours, recherche de cycle, tri topologique, arbre de recouvrement, décomposition en composantes connexes, plus courts chemins
* Maîtrise d'un langage de programmation impératif

Volume horaire : 3h CM + 4h TD/TP pendant 4 semaines puis 1h TP pendant 8 semaines

Complexité

* Introduction : modèle de calcul, pseudo-langage
* Situer la difficulté algorithmique de certains problèmes classiques. Quelques exemples :
  * Temps polynomial : méthodes de tris, plus courts chemins
  * Problèmes nécessitant peu de mémoire : accessibilité de deux sommets dans un graphe
  * Problèmes nécessitant beaucoup de temps et/ou de mémoire
* Formalisation des mesures de complexité en temps et espace.
* Réductions entre problèmes : transitivité, notion de complétude, problème "représentatif" d'une difficulté algorithmique
* Problèmes réputés difficiles en temps
  * Problèmes difficiles en théorie et en pratique, exemples.
  * NP : définition (algorithmes non-déterministe, witness-checking), NP-complétude
  * Théorème de Cook-Levin : SAT est NP-complet
  * Exemples de problèmes NP-complets et réductions : cliques, ensembles indépendants maximaux, recouvrement des sommets d'un graphe (Vertex Cover), ordonnancement de tâches, contraintes, ...

Université Paris Cité

    * Résolution pratiques de problèmes difficiles : SAT-solver pour le problème de satisfaisabilité, approches pratiques type *kernelization* pour les problèmes paramétrés (exemple de Vertex Cover)
* Problèmes réputés difficiles en espace (et au delà)
    * Espace mémoire polynomial (PSPACE) : définition et exemple (recherche de stratégie gagnante dans des jeux)
    * Problèmes PSPACE-complets
* Algorithmique parallèle
    * Problèmes parallélisables efficacement. Exemples : implémentation des opérations arithmétiques usuelles, multiplication matricielle, matching maximal dans un graphe
    * Classes de complexité parallèle et circuits (NC et AC)
    * Problèmes non-parallélisables efficacement: P-complétude, exemples
* Perspective : comparaison des classes de complexité (efficacité comparée des approches algorithmiques), hiérarchie en temps et en espace
    * Ouvertures possibles : rudiments sur les algorithmes d'approximation, algorithmes probabilistes

Volume horaire: 3h CM + 3h TD sur 8 semaines

HEURES D'ENSEIGNEMENT

Algorithmique	Cours Magistral	24h
Algorithmique		16h
Algorithmique	Travaux Pratiques	8h
Complexité	Cours Magistral	24h
Complexité	Travaux Dirigés	24h

SYLLABUS

Bibliographie

* Arora, S., & Barak, B. (2009). *Computational complexity: a modern approach.* Cambridge University Press.
* Dasgupta, S., Papadimitriou, C.H., and Vazirani U.V. (2008). *Algorithms*. McGraw-Hill.
* Manber, U. (1989). *Introduction to algorithms: a creative approach*. Reading, MA: Addison-Wesley.