Algorithmique et Complexité
Présentation générale du cours

CentraleSupélec – Gif

2023, ST2

Plan

Objectifs du cours

Algorithmique. . .

Méthodes informatiques pour résoudre des problèmes

- Méthode informatique : traitement systématique, automatisable
- Problème informatique : représentation d'une question
  à laquelle un ordinateur doit être en mesure de répondre
  - ➜ Exemple : Calculer $n!$ pour n donné

Objectifs du cours

Algorithmique. . .

Méthodes informatiques pour résoudre des problèmes

- Méthode informatique : traitement systématique, automatisable
- Problème informatique : représentation d'une question à laquelle un ordinateur doit être en mesure de répondre
  - ➜ Exemple : Calculer $n!$ pour n donné

. . . et complexité

- Complexité des algorithmes
- Complexité des problèmes

Démarche

Problèmes

- Plus court chemin
- Arbre couvrant min
- Flot maximum
- Bin Packing
- *etc.*

Démarche

Problèmes

- Plus court chemin
- Arbre couvrant min
- Flot maximum
- Bin Packing
- *etc.*

Complexité

Classes P et NP, réduction polynomiale

Démarche

Problèmes

- Plus court chemin
- Arbre couvrant min
- Flot maximum
- Bin Packing
- *etc.*

Complexité

Classes P et NP, réduction polynomiale

Outils algorithmiques

Structures de données :

➜ Listes, Piles, Files, Arbres, Graphes, Dictionnaires. . .

Algorithmiques connues :

➜ Parcours de graphes, Algorithmes glouton, Programmation dynamique. . .

Démarche

Problèmes

- Plus court chemin
- Arbre couvrant min
- Flot maximum
- Bin Packing
- *etc.*

Outils algorithmiques

Structures de données :

➜ Listes, Piles, Files, Arbres, Graphes, Dictionnaires...

Algorithmiques connues :

➜ Parcours de graphes, Algorithmes glouton, Programmation dynamique...

Complexité

Classes P et NP, réduction polynomiale

Algorithmes

- Algorithmes célèbres : Dijkstra, Ford-Fulkerson, Ford-Bellman...
- Analyse de la complexité de l'algorithme
- Optimalité de la solution

Compétences validées à l'issue du cours — C1 + C6.1 + C6.2

- Modélisation informatique d'un problème d'ingénierie (*computational thinking*)
- Choix de structures de données et de techniques de résolution adaptées
  - ➜ Savoir se ramener à un problème connu
  - ➜ Le résoudre numériquement (programmation avancée)
- Recherche de solution optimale :
  - ➜ Méthodes exactes et méthodes approchées
  - ➜ Évaluation de la solution
- Calcul de la complexité d'un algorithme
  - ➜ Temps et mémoire consommés par la machine
- Déterminer la classe de complexité d'un problème

Plan

Plan détaillé des cours

- Parcours de graphe, recherche de chemin, arbre couvrant minimum (3 séances)
- Flot maximum (1 séance)
- Programmation dynamique (1 séance)
- Complexité des problèmes (1 séance)
- Méthodes exactes pour les problèmes NP-difficiles (1 séance)

Plan détaillé des cours et des TDs/TPs

- Parcours de graphe, recherche de chemin, arbre couvrant minimum (3 séances)
  - ➜ 2 séances de TD
- Structures de données
  - ➜ 1 séance de TP
- Flot maximum (1 séance)
  - ➜ 1 séance de TD
- Programmation dynamique (1 séance)
  - ➜ 1 séance double de TD-Pratique
- Résolution de problème
  - ➜ Une séance double de TP
- Complexité des problèmes (1 séance)
  - ➜ 1 séance de TD
- Méthodes exactes pour les problèmes NP-difficiles (1 séance)
  - ➜ 1 séance de TD sur les méthodes approchées
  - ➜ 1 séance double de TD-Pratique

Organisation

7 séances de cours

5 séances de TD (5 × 1h30) + 2 séances de TD-Pratique (2 × 3h)

Exercices « comme à l'examen »

2 TPs (1h30 + 3h)

Langage Python

Supports et examen identiques pour tous les groupes

(en français comme en anglais...)

Évaluation

Contrôle continu : 20%

- TP en présentiel uniquement (séance double du 22 décembre)
  - ➜ Livrable : code + réponses aux questions
- 20% si la note est supérieure à celle de l'examen écrit

  En cas d'absence (justifiée ou non) → note d'examen *(selon session)*

Examen écrit : 80%

- 3 heures sur papier, type TD avec questions de cours
- Pas de programmation (mais nécessité d'écrire des algorithmes)
- Tous documents **manuscrits** autorisés + dictionnaire

Plan

Outils numériques

Site web de l'UE : Accessible depuis EduNao

- Copie des transparents de cours
- Sujets de TD et de TP
- Matériel supplémentaire :
  - Pratique des TDs (sujets Python en ligne)
  - Exercices supplémentaires avec éléments de correction

Outils numériques

Site web de l'UE : Accessible depuis EduNao

- Copie des transparents de cours
- Sujets de TD et de TP
- Matériel supplémentaire :
  - Pratique des TDs (sujets Python en ligne)
  - Exercices supplémentaires avec éléments de correction

Environnement de développement pour TD-pratique et TP

Visual Studio Code, PyCharm, Spyder... à vous de gérer !

➜ IDE avec **Débogage**

➜ Fichier de test sur Eduano !

Vous devez le faire fonctionner avant le premier TP

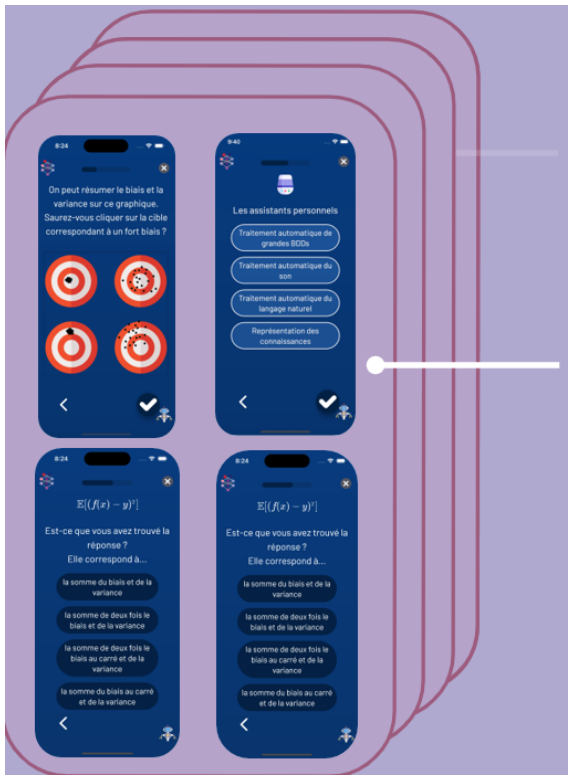
Micro-apprentissage : Ariago

Ariago

Outil de micro-apprentissage

- Support : site internet, application mobile, application tablette

➜ Vous recevrez un e-mail avec URL, login, mot de passe



Ariago

Outil de micro-apprentissage

- Support : site internet, application mobile, application tablette

➜ Vous recevrez un e-mail avec URL, login, mot de passe

Pourquoi Ariago

- Rappel des notions importantes du cours
- Exemples supplémentaires d'applications réelles pour les algorithmes classiques
- Petits exercices pour vous aider à mieux comprendre les notions de cours et à mieux préparer l'examen

Conseils pour bien réussir

Séances de cours

- Poser des questions en cours
- Réviser son cours avant les TDs

Conseils pour bien réussir

Séances de cours

- Poser des questions en cours
- Réviser son cours avant les TDs

Séances de TD

- Participer, aller au tableau (présentiel)
  Distanciel : proposer des réponses, même fausses !
- Refaire les exercices pour s'entraîner
- Programmer : Pratique des sujets de TD (en ligne)

Conseils pour bien réussir

Séances de cours

- Poser des questions en cours
- Réviser son cours avant les TDs

Séances de TD

- Participer, aller au tableau (présentiel)
  Distanciel : proposer des réponses, même fausses !
- Refaire les exercices pour s'entraîner
- Programmer : Pratique des sujets de TD (en ligne)

Après chaque TD de tous les groupes, une version avec des éléments de correction sera mise en ligne, mais cela ne vous dispense pas de prendre des notes !

Groupes de niveau

- Chaque voie/amphi comprend :
  - 1 groupe niveau “débutant” (assuré par le chargé d’amphi)
  - 2 groupes niveau “standard”
  - 1 groupe niveau “confirmé”

Groupes de niveau

- Chaque voie/amphi comprend :
  - 1 groupe niveau “débutant” (assuré par le chargé d’amphi)
  - 2 groupes niveau “standard”
  - 1 groupe niveau “confirmé”

- Le groupe débutant profitera de 5 séances de renforcement :
  - 1h chacune : allonger 5 séances particulières 1CM/2TDs/2TP
  - séances présentes sur vos agendas Géode
  - ➜ prendre le temps pour expliquer davantage

Groupes de niveau

- Chaque voie/amphi comprend :
  - 1 groupe niveau “débutant” (assuré par le chargé d’amphi)
  - 2 groupes niveau “standard”
  - 1 groupe niveau “confirmé”
- Le groupe débutant profitera de 5 séances de renforcement :
  - 1h chacune : allonger 5 séances particulières 1CM/2TDs/2TP
  - séances présentes sur vos agendas Géode
  - ➜ prendre le temps pour expliquer davantage
- Le groupe confirmé entamera en fin de TD :
  - les exercices supplémentaires
  - la partie pratique

Groupes de niveau

- Chaque voie/amphi comprend :
  - 1 groupe niveau “débutant” (assuré par le chargé d’amphi)
  - 2 groupes niveau “standard”
  - 1 groupe niveau “confirmé”
- Le groupe débutant profitera de 5 séances de renforcement :
  - 1h chacune : allonger 5 séances particulières 1CM/2TDs/2TP
  - séances présentes sur vos agendas Géode
  - ➜ prendre le temps pour expliquer davantage
- Le groupe confirmé entamera en fin de TD :
  - les exercices supplémentaires
  - la partie pratique
- Même examen pour tous, mêmes objectifs pédagogiques
  - ➜ Vous serez mieux préparés si vous choisissez un groupe adapté à votre niveau

Pour aller plus loin ...

Trois livres de référence

- *Introduction to Algorithms, Third Edition*. Par Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein. MIT Press, 2009.

  L'édition précédente (1996) est disponible en français chez Dunod sous le titre « Introduction à l'algorithmique ».

- *Algorithm Design*. Par Jon Kleinberg et Éva Tardos. Pearson Ed. (Addison-Wesley), 2006.

  Pas d'édition en français mais disponible en PDF sur Internet.

- *Programmation efficace : les 128 algorithmes qu'il faut avoir compris et codés en Python au cours de sa vie*. Par Christophe Dürr et Jill-Jênn Vie. Ellipse, 2016.

  français, chinois, anglais (bientôt)