Colle d'informatique 3

Programme

Révisions

- Langages formels et automates finis.
- Grammaires hors contextes.
- Décidabilité
- Graphes

Classes de complexité

- Problèmes de décision, de recherche, d'existence et de vérification, d'optimisation, de seuil.
- Modèle de calcul, opérations élémentaires, taille de l'entrée
- Classe P, problème du diviseur commun, problème du chemin dans un graphe
- Réduction polynomiale, appartenance à P par réduction polynomiale, 2SAT
- Classe NP, certificat, problème de vérification, problème SAT, inclusion de P dans NP
- Problèmes NP-difficiles, NP-complets, réduction polynomiale, problème SAT (théorème de Cook Levin admis)

Algorithmes d'apperentissage

- Apprentisagge supervisé, algorithme des k plus proches voisins, matrice de confusion, arbre k-d, arbre de décision, entropie de Shannon, gain.
- Apprentissage non-supervisé, algorithme des k moyennes, pseudo-matrice de confusion.

Commentaires

- Sur la partie *classes de complexité*, merci de guider les étudiants.
  La méthodologie proposée pour établir la NP-complétude d'un problème A est la suivante.
  - Prouver que A admet un *vérificateur polynomial* ; ce qui garantit $A \in NP$.
  - Prouver que pour un problème B NP-complet, on a $B \leqslant_p A$.

  Ont été traités en classe le problème du chemin hamiltonien (HamPath), le problème de la décomposition en facteurs premiers (Composite), le problème de la partition d'un multi-ensemble d'entiers positifs (Partition), le problème du sous-ensemble d'entiers de somme fixée (SubsetSum).
- Sur la partir *algorithmes d'apprentissage*, on peut faire analyser et interpréter des résultats.

Prévisionnel

- Cette semaine de reprise sera consacrée aux *algorithmes probabilistes*, thème au programme dès la semaine prochaine.
- Viendra ensuite le chapitre consacré aux *algorithmes d'approximations.*
- Puis celui sur les *jeux d'accessibilité à deux joueurs.*
- Le chapitre suivant permettra de réviser la *logique propositionnelle* et la *logique du premier ordre* pour aborder ensuite la *déduction naturelle.*
- Enfin, un dernier chapitre sera consacré à la *gestion de la concurrence et synchronisation* par l'étude des threads internes à un processus.

Extraits du programme officiel

Décidabilité et classes de complexité

On s'intéresse à la question de savoir ce qu'un algorithme peut ou ne peut pas faire, inconditionnellement ou sous condition de ressources en temps. Cette partie permet de justifier la construction, plus haut, d'algorithmes exhaustifs, approchés, probabilistes, etc. On s'appuie sur une compréhension pratique de ce qu'est un algorithme.

Notions	Commentaires
Problème de décision. Taille d'une instance. Complexité en ordre de grandeur en fonction de la taille d'une instance. Opération élémentaire. Complexité en temps d'un algorithme. Classe **P**.	Les opérations élémentaires sont les lectures et écritures en mémoire, les opérations arithmétiques, etc. La notion de machine de Turing est hors programme. On s'en tient à une présentation intuitive du modèle de calcul (code exécuté avec une machine à mémoire infinie). On insiste sur le fait que la classe **P** concerne des problèmes de décision.
Réduction polynomiale d'un problème de décision à un autre problème de décision.	On se limite à quelques exemples élémentaires.
Certificat. Classe **NP** comme la classe des problèmes que l'on peut vérifier en temps polynomial. Inclusion $\mathbf{P} \subseteq \mathbf{NP}$.	Les modèles de calcul non-déterministes sont hors programme.
NP-complétude. Théorème de Cook-Levin (admis) : SAT est NP-complet.	On présente des exemples de réduction de problèmes NP-complets à partir de SAT. La connaissance d'un catalogue de problèmes NP-complets n'est pas un objectif du programme.
Transformation d'un problème d'optimisation en un problème de décision à l'aide d'un seuil.	
Notion de machine universelle. Problème de l'arrêt.	
Mise en œuvre	
On prend soin de distinguer la notion de complexité d'un algorithme de la notion de classe de complexité d'un problème. Le modèle de calcul est une machine à mémoire infinie qui exécute un programme rédigé en OCaml ou en C. La maîtrise ou la technicité dans des formalismes avancés n'est pas un objectif du programme.	

Algorithmique pour l'intelligence artificielle

Notions	Commentaires
Apprentissage supervisé.	Algorithme des k plus proches voisins avec distance euclidienne. Arbres k dimensionnels. Apprentissage d'arbre de décision : algorithme ID3 restreint au cas d'arbres binaires. Matrice de confusion. On observe des situations de surapprentissage sur des exemples.
Apprentissage non-supervisé.	Algorithme de classification hiérarchique ascendante. Algorithme des k-moyennes. La démonstration de la convergence n'est pas au programme. On observe des convergences vers des minima locaux.