928. PROBLÈMES NP-COMPLETS : EXEMPLES ET RÉDUCTIONS

Prérequis : Les notions de machine de Turing (déterministe ou non) et de langage décidé par une machine de Turing.

I/ La classe NP

1) Complexité

[Car p 196]

def 1.1 La classe P est l'ensemble des langages (= problèmes de décision) décidés par une machine de Turing déterministe en temps $O(Q(n))$ où Q est un polynôme.

def 1.2 La classe NP est l'ensemble des langages décidés par une machine de Turing non déterministe en temps $O(Q(n))$ où Q est un polynôme.

rq 1.3 On a bien sûr $P \subset NP$. À ce jour, on ne sait pas si $P = NP$ ou non

[Car p200]

def 1.4 Un vérificateur en temps polynomial pour un langage L est une machine de Turing déterministe M qui accepte les entrées de la forme $<w, c>$ en temps polynomial en $|w|$ et telle que $L = \{w \mid \exists c, <w,c> \in L(M)\}$

prop 1.5 Un langage L appartient à NP ssi il existe un vérificateur en temps polynomial pour L

2) Réduction.

[Car p 202]

def 1.6 Soit $L_A \subset \Sigma_A^*$ et $L_B \subset \Sigma_B^*$ deux langages codant respectivement les instances positives du problème de décision A et celles de B.
Une réduction polynomiale de A à B est une fonction $f : \Sigma_A^* \longrightarrow \Sigma_B^*$ calculable en temps polynomial par une Machine de Turing déterministe et telle que $w \in L_A \iff f(w) \in L_B$. On note alors $A \leq_p B$

prop 1.7 Si $A \leq_p B$ et $B \in NP$ alors $A \in NP$

3) NP-complétude

[Car p 202-203]

def 1.8 Un problème A est dit NP-difficile si tout problème $B \in NP$ se réduit polynomialement à A

prop 1.9 Si A est NP-difficile et $A \leq_p B$ alors B est NP-difficile

def 1.10 Un problème A est dit NP-complet si $A \in NP$ et A est NP-difficile

4) Optimisation et décision

[Garey Johnson p 18-19]

def 1.11 Un problème de décision A est un problème pour lequel les seules réponses possibles sont "oui" et "non". On lui associe le langage $L_A = \{$instances positives de $A\}$

def 1.12 Un problème d'optimisation est un problème dans lequel on recherche une structure ayant un coût minimal.

prop 1.13 On peut associer à un problème d'optimisation un problème de décision.

ex 1.14 : Pb d'optimisation : minimiser le poids du chemin entre s et t dans un graphe complet pondéré G

→ Pb de décision : Entrée = G, s, t et $k \in \mathbb{N}$.

Question : Existe-t-il un chemin de s à t dans G de poids inférieur à k ?

rq 1.15 On dit que le problème d'optimisation A est NP-complet si le problème de décision qui lui est associé l'est.

II/ Problèmes autour de la satisfiabilité d'une formule

def 2.1 Le problème SAT est le suivant :

Entrée : Une formule φ du calcul propositionnel.

Question : φ est-elle satisfiable ?

[Car p203]

Th 2.2 (Théorème de Cook) SAT est NP-complet (DEV1)

déf 2.3 • Le problème CNF-SAT est le suivant :

Entrée : Une formule φ du calcul propositionnel sous forme normale conjonctive (CNF)

Question : φ est-elle satisfiable ?

• Le problème 3SAT est le suivant :

Entrée : Une formule φ du calcul propositionnel en CNF et telle que chaque clause contienne exactement 3 littéraux

Question : φ est-elle satisfiable ?

[Carp 205 p3SAT]

prop 2.4 $SAT \leq_p$ CNF-SAT et $SAT \leq_p 3SAT$ (Tseitin)

prop 2.5 CNF-SAT et 3SAT sont NP-complets

déf 2.6 • Le problème 2SAT est le suivant :

Entrée : Une formule φ du calcul propositionnel en CNF et telle que chaque clause contienne exactement 2 littéraux.

Question : φ est-elle satisfiable ?

• Le problème HORN-SAT est le suivant :

Entrée : Une formule φ du calcul propositionnel en CNF et telle que chaque clause contienne au plus un littéral positif

Question : φ est-elle satisfiable ?

prop 2.7 2SAT et HORN-SAT sont dans P

III / Séparation par automate [Floyd]

déf 3.1 Le problème de séparation par automate (PSA) est le suivant :

Entrée : $k \in \mathbb{N}$, S et T des langages finis sur Σ^*

Question : Existe-t-il un automate fini déterministe à k états tel que $S \subset L(A)$ et $T \cap L(A) = \emptyset$

prop 3.2 $SAT \leq_p PSA$

prop 3.3 PSA est NP-complet.

4 Problème du voyageur de commerce

[Carton p206]

déf 4.1 : Problème du chemin Hamiltonien :

Entrée : un graphe orienté G non pondéré, deux sommets A et B de G

Question : existe-t-il un chemin de A à B dans G qui passe une et une seule fois par chaque sommet de G ?

déf 4.2 : Problème du cycle Hamiltonien

Entrée : Un graphe orienté G non pondéré.

Question : existe-t-il un cycle qui passe une et une seule fois par chaque sommet dans G ?

prop 4.3 : CNF-SAT $\leq_p$ Chemin Hamiltonien et est NP-complet

prop 4.4 : CNF-SAT $\leq_p$ cycle Hamiltonien et est NP-complet

prop 4.5 : chemin hamiltonien $\leq_p$ cycle Hamiltonien.

[Cormen p1008]

déf 4.6 : Problème du voyageur de commerce

Entrée : Un graphe complet, pondéré non orienté G, un entier k

Question : existe-t-il un cycle Hamiltonien de poids $\leq k$.

remarque 4.7 : Le problème du voyageur de commerce est le problème de décision associé au problème d'optimisation du voyageur de commerce dont l'objectif est la recherche d'un cycle Hamiltonien de poids minimal)

proposition 4.8 : Cycle Hamiltonien $\leq_p$ voyage de commerce et est NP-complet

[Garey Johnson p131]

proposition 4.9 : Dans le cadre euclidien, (i.e si les chemins respectent l'inégalité triangulaire), il existe une 2-approximation polynomiale du problème du voyageur de commerce à l'aide de l'arbre couvrant minimal, c'est à dire un algorithme qui s'exécute en temps polynomial et renvoie des chemins de poids inférieur à 2x le chemin optimal. (DEV2)

Garey Johnson p147

prop 4.10 : Si $P \neq NP$ il n'existe aucune ρ-approximation polynomiale au problème du voyageur de commerce.

DEV2

5 Programmation linéaire en nombre entiers

Garey Johnson p245

def 5.1 Programmation linéaire en nombre entiers PLNE
Entrée : une matrice d'entiers C, un vecteur o, un vecteur b, un entier k. C est de taille n,k b est de taille k et o de taille n
Question : existe-t-il un vecteur d'entier x tel que $Cx \leq b$, $\langle o,x \rangle \leq k$?

remarque 5.2 PLNE est le problème de décision associé au problème d'optimisation $\min\limits_{\substack{Cx \leq b \\ x \in \mathbb{Z}^n}} (\langle o,x \rangle)$

prop 5.3 $\text{CNF-SAT} \leq_p \text{PLNE}$ et est NP-complet.

prop 5.4 : La relaxation de PLNE, c'est à dire $\min\limits_{\substack{Cx \leq b \\ x \in \mathbb{R}^n}} \langle o,x \rangle$ est polynomiale. [Admis].

Cette relaxation est appellée programation linéaire.

Références
- [Car] = Carton, Langages formels
- Cormen
- [Garey-Johnson] = Computers and intractability, A guide to the theory of NP-completeness
- Le langage des machines, Floyd & Biegel (pour PSA)
- Sipser, Introduction to the theory of computation (pour les réductions)