

Calculabilité et complexité

Cours n° 5

Nicolas (Miki) Hermann

LIX, École Polytechnique

hermann@lix.polytechnique.fr

Couplage triparti

Problème: 3 DIMENSIONAL MATCHING (3DM)

Entrée: Trois ensembles B (garçons), G (filles) et H (maisons), avec $|B| = |G| = |H| = n$, et une relation $T \subseteq B \times G \times H$.

Question: Existe-t-il un ensemble $S \subseteq T$ de n triples disjoints ?

Couplage triparti

Problème: 3 DIMENSIONAL MATCHING (3DM)

Entrée: Trois ensembles B (garçons), G (filles) et H (maisons), avec $|B| = |G| = |H| = n$, et une relation $T \subseteq B \times G \times H$.

Question: Existe-t-il un ensemble $S \subseteq T$ de n triples disjoints ?

Théorème

3DM est NP-complet.

Couplage triparti

Problème: 3 DIMENSIONAL MATCHING (3DM)

Entrée: Trois ensembles B (garçons), G (filles) et H (maisons), avec $|B| = |G| = |H| = n$, et une relation $T \subseteq B \times G \times H$.

Question: Existe-t-il un ensemble $S \subseteq T$ de n triples disjoints ?

Théorème

3DM est NP-complet.

Indication pour la preuve :

3DM \in NP : choix et test.

Borne inférieure : réduction à partir de 3SAT. □

Couplage biparti

Problème: PERFECT MATCHING

Entrée: Deux ensembles B (garçons) et G (filles), avec $|B| = |G| = n$,
et une relation $T \subseteq B \times G$.

Question: Existe-t-il un ensemble $S \subseteq T$ de n couples disjoints ?

Couplage biparti

Problème: PERFECT MATCHING

Entrée: Deux ensembles B (garçons) et G (filles), avec $|B| = |G| = n$, et une relation $T \subseteq B \times G$.

Question: Existe-t-il un ensemble $S \subseteq T$ de n couples disjoints ?

Théorème

PERFECT MATCHING est dans P pour un algorithme en temps $O(n^{2.5})$.

Problème: EXACT COVER BY 3-SETS (X3C)

Entrée: Une constante $m \in \mathbb{N}$, un ensemble U avec $|U| = 3m$ et une famille $F = \{S_1, \dots, S_n\}$ de sous-ensembles de U , tels que $|S_i| = 3$.

Question: Existe-t-il m sous-ensembles S_{i_1}, \dots, S_{i_m} disjoints parmi F tel que $S_{i_1} \cup \dots \cup S_{i_m} = U$?

Problème: EXACT COVER BY 3-SETS (X3C)

Entrée: Une constante $m \in \mathbb{N}$, un ensemble U avec $|U| = 3m$ et une famille $F = \{S_1, \dots, S_n\}$ de sous-ensembles de U , tels que $|S_i| = 3$.

Question: Existe-t-il m sous-ensembles S_{i_1}, \dots, S_{i_m} disjoints parmi F tel que $S_{i_1} \cup \dots \cup S_{i_m} = U$?

Théorème

X3C est NP-complet.

Problème: EXACT COVER BY 3-SETS (X3C)

Entrée: Une constante $m \in \mathbb{N}$, un ensemble U avec $|U| = 3m$ et une famille $F = \{S_1, \dots, S_n\}$ de sous-ensembles de U , tels que $|S_i| = 3$.

Question: Existe-t-il m sous-ensembles S_{i_1}, \dots, S_{i_m} disjoints parmi F tel que $S_{i_1} \cup \dots \cup S_{i_m} = U$?

Théorème

X3C est NP-complet.

Indication pour la preuve

Banalisation de 3DM. □

Problème: SET PACKING

Entrée: Une famille $F = \{S_1, \dots, S_n\}$ de sous-ensembles d'un ensemble fini U et une constante $K \in \mathbb{N}$.

Question: Existe-t-il K sous-ensembles disjoints parmi F ?

Théorème

SET PACKING est NP-complet.

Indication pour la preuve

Réduction à partir de x3c. □

Problème: SET COVERING

Entrée: Une famille $F = \{S_1, \dots, S_n\}$ de sous-ensembles d'un ensemble fini U et un budget $B \in \mathbb{N}$.

Question: Existe-t-il au plus B sous-ensembles parmi F dont l'union est égale à U ?

Théorème

SET COVERING est NP-complet.

Indication pour la preuve

Réduction à partir de SET PACKING. □

Problème: INTEGER PROGRAMMING

Entrée: Un système d'inéquations $Ax \leq b$ avec une matrice $k \times n$ entière A et un vecteur de longueur n entier b .

Question: Existe-t-il une solution $x \in \mathbb{Z}^k$ pour $Ax \leq b$?

Théorème

INTEGER PROGRAMMING est NP-complet.

Indication pour la preuve

INTEGER PROGRAMMING \in NP : Nontriviale ! S'appuie sur le Lemme de Farkas (s'il existe une solution pour $Ax \leq b$, il existe une solution dont la norme est bornée).

Borne inférieure : réduction à partir de 3SAT. Pour chaque variable x_i on ajoute $x_i \geq 0$ et $x_i \leq 1$. Le littéral y est codé par $c(y) = y$, le littéral $\neg y$ est codé par $c(\neg y) = 1 - y$. Pour chaque clause $c_i = l_1^i \vee l_2^i \vee l_3^i$, on écrit $c(l_1^i) + c(l_2^i) + c(l_3^i) \geq 1$. □

Exprimer SET COVERING en forme de INTEGER PROGRAMMING.

Problème: KNAPSACK

Entrée: Un ensemble d'éléments $A = \{a_1, \dots, a_n\}$ avec le poids $w(a_i) \in \mathbb{N}$ et la valeur $v(a_i) \in \mathbb{N}$, une limite de poids W et une valeur minimale V .

Question: Existe-t-il un sous-ensemble $S \subseteq A$ tel que $\sum_{a \in S} w(a) \leq W$ et $\sum_{a \in S} v(a) \geq V$?

Problème: KNAPSACK

Entrée: Un ensemble d'éléments $A = \{a_1, \dots, a_n\}$ avec le poids $w(a_i) \in \mathbb{N}$ et la valeur $v(a_i) \in \mathbb{N}$, une limite de poids W et une valeur minimale V .

Question: Existe-t-il un sous-ensemble $S \subseteq A$ tel que $\sum_{a \in S} w(a) \leq W$ et $\sum_{a \in S} v(a) \geq V$?

Cas spécial

Problème: SUBSET SUM

Entrée: Un ensemble fini d'entiers $A = \{a_1, \dots, a_n\}$ et une borne $B \in \mathbb{N}$.

Question: Existe-t-il un sous-ensemble $S \subseteq A$ tel que $\sum_{a \in S} a = B$?

Théorème

KNAPSACK et SUBSET SUM sont NP-complets.

Théorème

KNAPSACK et SUBSET SUM sont NP-complets.

Idée de la preuve

Réduction à partir de x3C avec $W = V = B = 2^n - 1$. Coder les ensembles par vecteurs caractéristiques, i.e., en nombre binaires. \square

Théorème

KNAPSACK peut être résolu en temps $O(nW)$ et SUBSET SUM peut être résolu en temps $O(nB)$.

Théorème

KNAPSACK peut être résolu en temps $O(nW)$ et SUBSET SUM peut être résolu en temps $O(nB)$.

Indication pour la preuve

Programmation dynamique. Calculer

$D(w, i)$ = valeur maximale atteignable par une sélection parmi i premiers éléments, tel que le poids total est égal à w .

Relation de récurrence

$$D(w, i + 1) = \max\{D(w, i), v(a_i) + D(w - w(a_{i+1}), i)\}$$

avec $D(w, 0) = 0$ pour chaque $w \leq W$.

La réponse est YES si et seulement si $D(W, n) \geq V$. □

Corollaire

KNAPSACK et SUBSET SUM sont **pseudo-polynomiaux**, i.e., polynomiaux en notation unaire.

Corollaire

KNAPSACK et SUBSET SUM sont **pseudo-polynomiaux**, i.e., polynomiaux en notation unaire.

La NP-complétude vient de nombres dans le problème.

NP-complétude au sens fort

Problème: BIN PACKING

Entrée: Un ensemble $A = \{a_1, \dots, a_n\}$ avec les capacités $c(a_i) \in \mathbb{N}$, la capacité limite C et le nombre de cases B .

Question: Existe-t-il un distribution de A en B cases, telle que la somme de capacités d'éléments dans chaque case ne dépasse pas C ?

Théorème

BIN PACKING est NP-complet au sens fort, i.e., même si les capacités sont donnés en notation unaire.

Indication pour la preuve

Réduction à partir de 3DM. □

Théorème

Un langage L est NP-complet si et seulement si son complément $\bar{L} = \Sigma^* \setminus L$ est coNP-complet.

Problème: TAUTOLOGY

Entrée: Une formule propositionnelle φ en forme normale **disjonctive**.

Question: La formule φ est-elle une tautologie ?

Théorème

TAUTOLOGY est coNP-complet.

Théorème

TAUTOLOGY est coNP-complet.

Indication pour la preuve

TAUTOLOGY \in coNP : choix et refus.

Borne inférieure : φ est une tautologie si et seulement si $\neg\varphi$ n'est pas satisfaisable. Complément de SAT. \square

Théorème

TAUTOLOGY est coNP-complet.

Indication pour la preuve

TAUTOLOGY \in coNP : choix et refus.

Borne inférieure : φ est une tautologie si et seulement si $\neg\varphi$ n'est pas satisfaisable. Complément de SAT. \square

Théorème

NP = coNP si et seulement si P = NP.

Structure de la classe NP

La carte de la classe NP : 3 possibilités

- 1 $P = A = \text{NP-complets}$
- 2 $P = \text{NP-complets}$
- 3 $P = \text{NP} = \text{NP-complets}$

Structure de la classe NP

La carte de la classe NP : 3 possibilités

- 1 $P = A = \text{NP-complets}$
- 2 $P = \text{NP-complets}$
- 3 $P = \text{NP} = \text{NP-complets}$

Théorème (Ladner)

Si $P \neq \text{NP}$, il existe un langage $L \in \text{NP} \setminus P$ qui n'est pas NP-complet.

Problèmes PSPACE-complets

Problème: QSAT

Entrée: Une formule $\psi = \exists x_1 \forall x_2 \exists x_3 \cdots Q x_n \varphi(x_1, \dots, x_n)$ où φ est une formule propositionnelle et les variables x_i peuvent prendre des valeurs booléennes.

Question: La formule ψ est-elle valide ?

Remarque : n n'est pas fixé

Théorème

QSAT est PSPACE-complet.

Indication pour la preuve

Réduction à partir d'une machine de Turing travaillant en espace polynomial, avec les idées similaires de la preuve de la NP-complétude de SAT. □

Problème: GEOGRAPHY

Entrée: Un graphe orienté $G = (V, A)$ et un sommet de départ s .

2 joueurs : Alice et Bob. Règles du jeu : Les joueurs doivent nommer alternativement des noms de villes (sommets) où le symbole à la fin de la ville précédente doit être le symbole initial de la ville suivante (flèches). Les noms des villes (sommets) ne peuvent pas se répéter. Le joueur qui ne peut plus continuer a perdu.

Question: Alice a-t-elle une stratégie gagnante ?

Théorème

GEOGRAPHY est PSPACE-complet.

Question : Y-a-t-il des classes entre NP et PSPACE fermées par réductions ?

Réponse : OUI, la hiérarchie polynomiale.

Considérons une Machine de Turing $M^?$ qui peut poser des questions à une boîte noire (**oracle**) si elle se trouve dans un état spécial appelé $q^?$ (interrogation). L'oracle répond YES ou NO. La machine $M^?$ continue de travailler selon la réponse de l'oracle. L'appel d'oracle coûte une unité de temps et aucun espace. Le travail d'oracle ne compte pas dans le travail de la machine $M^?$.

M^{SAT} : la machine M pose des questions à l'oracle qui répond aux questions SAT

M^{NP} : oracle NP

M^{coNP} : équivalent à M^{NP}

Nous pouvons définir de nouvelles classes en utilisant les oracles.

Hiérarchie polynomiale

$$\begin{aligned}P &= \Delta_0 P = \Sigma_0 P = \Pi_0 P \\ \Sigma_1 P &= \text{NP} \\ \Pi_1 P &= \text{coNP} \\ \Delta_{k+1} P &= P^{\Sigma_k P} \\ \Sigma_{k+1} P &= \text{NP}^{\Sigma_k P} \\ \Pi_{k+1} P &= \text{coNP}^{\Sigma_k P} \\ \\ \Pi_k P &= \text{co}\Sigma_k P \\ \text{PH} &= \bigcup_{i=1}^{\infty} \Sigma_i P\end{aligned}$$

Problème: $QSAT_k$

Entrée: Une formule $\psi = \exists X_1 \forall X_2 \exists X_3 \cdots QX_k \varphi(X_1, \dots, X_k)$ où φ est une formule propositionnelle (en CNF pour k impaire et en DNF pour k paire) et X_i sont des vecteurs de variables qui prennent des valeurs booléennes.

Question: La formule ψ est-elle valide ?

Remarque : k est fixé

Théorème

$QSAT_k$ est $\Sigma_k P$ -complet.

Il est évident que $PH \subseteq PSPACE$. Quid de $PH = PSPACE$?

Théorème

Si $PH = PSPACE$ alors il existe un k tel que

$$\Sigma_k P = \Pi_k P = \Delta_{k+1} P = \Sigma_{k+1} P = \dots = PSPACE$$

Autrement dit, la hiérarchie polynomiale s'écroule.

Les oracles peuvent nous aider de séparer les classes de complexité.

Théorème

Il existe des oracles A, B, A', B', A_k, B_k pour chaque $k \in \mathbb{N}$ pour lesquels

- 1 $P^A = NP^A$
- 2 $P^B \neq NP^B$
- 3 $NP^{A'} = PSPACE^{A'}$
- 4 $NP^{B'} \neq PSPACE^{B'}$
- 5 $\Sigma_k P^{A_k} = \Sigma_{k+1} P^{A_k}$
- 6 $\Sigma_k P^{B_k} \neq \Sigma_{k+1} P^{B_k}$

Problème: HORNSAT

Entrée: Une formule propositionnelle φ en CNF avec au plus un littéral positif par clause.

Question: La formule φ est-elle satisfaisable ?

Théorème

HORNSAT et P-complet via les réductions logarithmiques.

Problème: HORNSAT

Entrée: Une formule propositionnelle φ en CNF avec au plus un littéral positif par clause.

Question: La formule φ est-elle satisfaisable ?

Théorème

HORNSAT et P-complet via les réductions logarithmiques.

Idée de la preuve

Le travail d'une machine de Turing **déterministe** M travaillant en temps polynomial peut être décrite (en utilisant les idées de la preuve de Cook pour SAT) par les règles

$$x_1 \wedge \cdots \wedge x_k \rightarrow y$$

Théorème

REACHABILITY est NL-complet.

Littérature :

- 1 Ch. H. Papadimitriou. **Computational Complexity**. Addison-Wesley, 1994.
- 2 M. R. Garey et D. S. Johnson. **Computers and Intractability : A Guide to the Theory of NP-Completeness**. Freeman and co., New York, 1979.
- 3 R. Lassaigne et M. de Rougemont. **Logique et Complexité**. Hermes, 1996.
- 4 N. Hermann et P. Lescanne. **Est-ce que $P = NP$?** Les Dossiers de La Recherche, vol. 20, pp. 64–68, 2005.

Pages web sur $P = ? = NP$

- http://www.claymath.org/millennium/P_vs_NP/
- http://en.wikipedia.org/wiki/P=NP_problem
- <http://www.win.tue.nl/~gwoegi/P-versus-NP.htm>
- http://qwiki.caltech.edu/wiki/Complexity_Zoo