Problèmes NP-Complets

1. Modèles de calcul, complexité.
2. $\mathcal{NP}$-complétude
3. Problèmes logiques $\mathcal{NP}$ complets
4. Problèmes de graphes $\mathcal{NP}$ complets
5. Problèmes en nombres entiers $\mathcal{NP}$ complets

Qu'est-ce qu'un calcul ?

Problème : défini par les données d'entrée, et pour chaque donnée, le résultat désiré. (= spécification).

1. **Machine de Turing :** Modélise un programme. La mémoire consiste en un ruban sur lequel sont écrits des 0 et des 1, un curseur se déplace le long du ruban et, en fonction de l' “état” courant de la machine, modifie le contenu de la mémoire, l'état courant, et déplace éventuellement le curseur d'une case vers la droite ou vers la gauche. Donné d'entrée: écrite sur le ruban.
2. **Programme d'un mini-langage de base :** Types autorisés : caractères, chaînes de caractères. Opérations autorisées: comparaison au mot vide, concaténation d'un caractère, accès à ou suppression du premier caractère. Instructions autorisées : affectation de caractères, conditionnelle `if...then...else...` ou `case...`, itération `while... do....`

Processus : pour une entrée donnée, suite des instructions exécutées lors du calcul.

Problème des boucles infinies.

Indécidabilité : Il existe des problèmes pour lesquels il n'existe pas de programme qui termine toujours au bout d'un temps fini.

Exemple : Correspondance de Post.
Donnée : couples de mots

$$(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n).$$

Résultat : existe-t-il un mot w tel que

$$w = u_{i_1} u_{i_2} \ldots u_{i_q} = v_{i_1} v_{i_2} \ldots v_{i_q}?$$

Thèse de Church : Tous les modèles de calcul effectif ont la même puissance et sont traductibles l'un par l'autre.

Complexité

Dans ce cours, tous les problèmes étudiés sont décidables.

Complexité : Un algorithme a complexité polynomiale s'il existe un polynome p tel que le nombre d'instructions exécutées par le processus soit au plus $p(n)$, où le maximum est pris sur toutes les données d'entrée de taille n. Un problème a complexité polynomiale s'il existe un algorithme polynomial pour le résoudre.

Postulat : Tous les modèle de calcul raisonnables sont équivalents du point de vue de la complexité polynomiale.

Postulat : Un algorithme est efficace s'il a complexité polynomiale.

La grande question : identifier les problèmes de complexité polynomiale.

Classe $\mathcal{P}$

Problème de décision : Résultat binaire $\in \{$"oui","non"$\}$, réponse à la question : La donné d'entrée possède-t-elle la propriété suivante : … ?

L'ensemble des **problèmes de décision polynomiaux** constitue la classe $\mathcal{P}$.

Exemples : existence d'un arbre couvrant de poids $\leq p$, d'un chemin de longueur $\leq \ell$ entre deux sommets d'un graphe, d'un flot de valeur $\geq v$, d'une coupe de capacité $\leq c$, d'un couplage de taille $\geq t$, d'une solution faisable pour un programme linéaire de valeur $\leq v$, …

Classe $\mathcal{NP}$

Un problème de décision Π est dans la **classe** $\mathcal{NP}$ s'il existe une propriété Π' de $\mathcal{P}$ et un polynome p tel que : pour toute entrée w du problème Π, la réponse est "oui" si et seulement s'il existe un mot v de longueur au plus $p(|w|)$ tel que (w, v) satisfasse la propriété Π'.

$$w \in \mathbf{Oui}(\Pi) \iff \exists v \ \mathbf{tel\ que}\ (|v| \leq p(|w|)\ \mathbf{et}\ (w, v) \in \mathbf{Oui}(\Pi')).$$

Exemple : chemin hamiltonien

Problème Π. Entrée w : graphe non orienté. Propriété : existe-t-il un chemin qui passe une fois et une seule par tous les sommets ?

Problème Π'. Entrée (w, v): w graphe non orienté, v suite de n sommets du graphe. Résultat : v est-il un chemin de w qui passe une fois et une seule par tous les sommets ?

$\Pi' \in \mathcal{P}$ **:** évident. $|v| \leq p(|w|)$ **:** clair.

w possède un chemin hamiltonien si et seulement s'il existe un v tel que v soit un chemin hamiltonien de w.

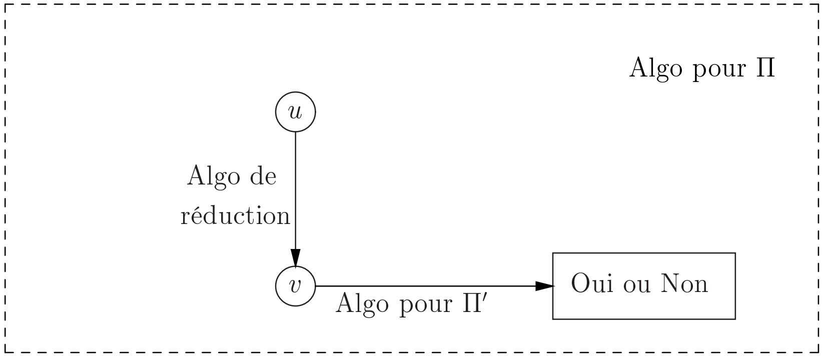
Remarque sur $\mathcal{NP}$: Oracle qui donne la réponse v, lire cette réponse et vérifier qu'elle est correcte prend un temps polynomial.

LA conjecture de l'informatique théorique

$$\mathcal{P} \neq \mathcal{NP}\ ?$$

Réduction polynomiale

Un problème Π **se réduit à** un problème Π' s'il existe un algorithme polynomial qui transforme une entrée u de Π en une entrée v de Π' telle que : $u \in \Pi' \iff v \in \Pi$.



$\mathcal{NP}$ complétude

- Si $\Pi' \in \mathcal{P}$ alors $\Pi \in \mathcal{P}$.
- Si $\Pi \notin \mathcal{P}$ alors $\Pi' \notin \mathcal{P}$.
- La réduction est transitive.

Définition : Un problème Π' de $\mathcal{NP}$ est $\mathcal{NP}$ **complet** si pour tout problème Π de $\mathcal{NP}$, il existe une réduction polynomiale de Π à Π'.

Propriété : si Π est $\mathcal{NP}$ complet et si $\Pi \in \mathcal{P}$, alors $\mathcal{P} = \mathcal{NP}$.

Propriété bis : si Π est $\mathcal{NP}$ complet et si $\mathcal{P} \neq \mathcal{NP}$, alors il n'existe pas d'algorithme polynomial pour Π.

SAT : un premier problème $\mathcal{NP}$ complet

- **SAT :** Entrée : formule CNF, $(x_1 \vee \bar{x_2}) \wedge (x_2 \vee x_4 \vee \bar{x_7}) \wedge \ldots$.
  Résultat : existe-t-il une assignation de valeurs de vérité aux variables x_i qui rende la formule vraie ?
- **SAT $\in \mathcal{NP}$:** on devine l'assignation et on vérifie que toutes les clauses sont satisfaites.
- **Théorème de Cook :** Soit $\Pi \in \mathcal{NP}$. Il existe un polynôme p et un algorithme qui pour chaque entrée w de Π, construit en temps $p(|w|)$ une expression booléenne f_w telle que

$$w \in \Pi \Leftrightarrow f_w \text{ satisfaisable}$$

- **Preuve :** voir cours Majeure 1.

Comment trouver d'autres problèmes $\mathcal{NP}$ complets ?

Soit $Q \in \mathcal{NP}$. Supposons qu'on connaisse déjà un problème $\mathcal{NP}$ complet Π. Si Π se réduit polynomialement à Q, alors :

$$Q \text{ est } \mathcal{NP} \text{ complet.}$$

3-SAT est $\mathcal{NP}$ complet

3-SAT : les clauses contiennent au plus trois littéraux

3-SAT $\in \mathcal{NP}$: Deviner l'assignation et vérifier qu'elle rend la formule vraie.

SAT se réduit à 3-SAT : On prend les clauses de la formule SAT une à une et on les remplace par des clauses à au plus 3 littéraux.

$$C = x \vee y \vee z \vee u \vee v \vee w \vee t.$$

On introduit les nouvelles variables a, b, c et on remplace C par

$$(x \vee y \vee a) \wedge (\bar{a} \vee u \vee b) \wedge (\bar{b} \vee v \vee c) \wedge (\bar{c} \vee w \vee t).$$

Remarque : 2-SAT $\in \mathcal{P}$

On réécrit le problème sous forme de recherche de chemins dans le graphe orienté G suivant :

- Pour chaque variable x_i, G a deux sommets étiquetés x_i et $\bar{x}_i$.
- Pour chaque clause $(y_i \vee y_j)$, G a un arc $\overline{y_i} \to y_j$ et un arc $\overline{y_j} \to y_i$, traduisant les implications "si y_i est faux alors y_j doit être vrai" et "si y_j est faux alors y_i doit être vrai".
- On montre que la formule est satisfaisable si et seulement si, dans le graphe G, pour aucun i il n'existe de circuit passant à la fois par y_i et par $\overline{y_i}$.

STABLE est $\mathcal{NP}$ complet

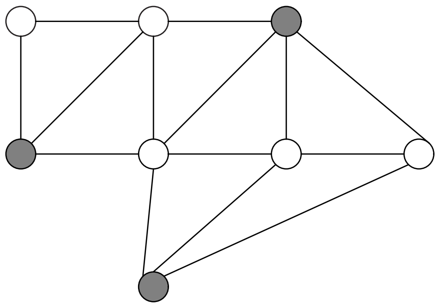


Figure 7: Un graphe et un de ses stables

Un **stable** est un sous-ensemble S de V tel que deux sommets de S ne sont pas joints par une arête.

Le problème STABLE : Entrée : (G, k). Existence d'un stable de taille $\geq k$? STABLE $\in \mathcal{NP}$: clair.

Réduction de 3-SAT à STABLE

Soit f une formule comportant n clauses chacune avec exactement trois littéraux. On construit le graphe G ayant $3n$ sommets. On relie par trois arêtes les sommets correspondants à une clause. On relie par une arête tous les couples de sommets qui correspondent à un x_i et un $\overline{x_i}$. Alors : f est satisfaisable si et seulement si G possède un indépendant de taille n.

COUVRANT-MIN est $\mathcal{NP}$ complet

Un **ensemble couvrant** Y de sommets est un sous-ensemble de V tel que toute arête du graphe possède au moins une extremité dans Y.

Idée : Un ensemble couvrant est le complémentaire d'un indépendant.

SUBSET-SUM est $\mathcal{NP}$ complet

Données d'entrée : Un ensemble fini d'entiers S, et un entier x.

Problème : Existe-t-il $F \subset S$ tel que $\sum_{f \in F} f = x$?

SUBSET-SUM $\in \mathcal{NP}$: clair.

Réduction : A partir de COUVRANT-MIN.

COUVRANT-MIN se réduit à SUBSET-SUM

Soit $G = (V, E)$ le graphe d'entrée de COUVRANT-MIN. Soient $x_1, \ldots, x_n$ ses sommets et $e_0, e_1, \ldots, e_{m-1}$ ses arêtes. On veut construire un ensemble S et un entier x :

- A chaque sommet x_i incident à $e_{i_1}, e_{i_2}, \ldots, e_{i_p}$ on associe le nombre

$$a_i = \sum_{j=1}^{p} 4^{i_j} \quad + \quad 4^m.$$

- $S = \{a_1, a_2, \ldots, a_n\} \cup \{1, 4, 4^2, \ldots, 4^{m-1}\}$
- $x = 2\left(\sum_{j=0}^{m-1} 4^j\right) \quad + \quad k4^m.$

Analyse

x est somme d'un sous-ensemble de S ssi il existe un ensemble couvrant de sommets de taille k pour G :

Tout entier x qui est somme des éléments de S s'écrit (en base 4) :

$$x = \sum_{j=0}^{m-1} u_i 4^j \quad + \quad y4^m \quad \text{avec} \quad 0 \leq u_i \leq 3.$$