I Notion de NP-complétude

1. Un premier exemple : le voyageur de commerce

On s'intéresse au problème suivant :

Problème 01 : Le voyageur de commerce

Entrées : n villes et les distances entre elles

Sortie : Un chemin de longueur totale minimale passant une fois par chaque ville.

Il possède :
- Une solution de complexité $O(n!)$: Tester tout les chemins
- Un algo. linéaire vérifiant si la longueur d'un chemin est inférieure à k fixé. Calculer la longueur du chemin.

2. La classe NP

Définition 01 : Problèmes NP ; Problème P

La classe NP (resp P) est la classe des problème de complexité non déterministe (resp. déterministe) polynomiale. C'est-à-dire pour lesquels il est possible de "deviner" puis de vérifier une solution en temps polynomial.

Remarque : Un problème de décision est dans NP ssi il peut être décidé par une Machine de Turing non déterministe en temps polynomial.

exemples : SAT & 3-SAT sont dans NP

Théorème 01 : Théorème de Cook

Si SAT $\in$ P, alors P = NP

Ce théorème motive la définition de la classe des problème qui vérifient cette propriété.

3. Problèmes NP-difficile ; Problèmes NP-complets

Définition 02 : Réduction Polynomiale.

Soient A et B, deux problèmes de décision. On dit que A se réduit polynomialement à B, ce que l'on note $A \leq_p B$ si il existe une fonction f calculable en temps polynomial telle que $x \in A \Leftrightarrow f(x) \in B$.

exemple : 3-SAT $\leq_p$ SAT

Définition 03 : Problèmes NP complets

Un problème de décision A est dit NP complet ssi :
- $\rightarrow A \in NP$
- $\rightarrow \forall P \in NP, P \leq_p A$.

Si la seconde condition seule est vérifiée, A est dit NP-difficile

Théorème 02 : Problème NP-complet ; preuve.

Pour montrer qu'un problème est NP-complet, il suffit de montrer qu'il est NP et qu'un problème NP complet peut se réduire à lui.

La tâche délicate est donc de trouver un premier problème NP-complet : Théorème de Cook [DVP].

Nous avons maintenant à disposition une méthode de détermination de la NP-complétude.

II Quelques Problèmes de Graphes NP-complets

Il est possible de réduire polynomialement 3-SAT à de nombreux problèmes de Graphes. Ces réductions se font à l'aide de Gadgets, structures se répétant dans les graphes construits et représentant les variables et clauses des formules de 3-SAT.

Définition & Proposition 1 : CLIQUE est NP complet
Le problème CLIQUE consiste à déterminer si un Graphe donné contient un sous-graphe complet de cardinal au moins égal à $k \in \mathbb{N}^*$ donné.

Définition & Proposition 2 : HAMPATH est NP-complet
Le problème consiste à déterminer si dans un graphe G donné, pour deux sommets s et t de G, il existe un chemin reliant s à t et passant une unique fois par chaque sommet de G.

Définition & Proposition 3 : VERTEX-COVER est NP-complet
Le problème consiste à déterminer si dans un graphe G donné, il existe un sous-graphe à k (donné) sommets contenant toutes les arêtes de G.

Une fois ces problèmes NP complets posés, nous pouvons introduire des réductions plus avancées.

Définition & Proposition 4 : HAMCYCLE est NP-complet
Le problème consiste à déterminer si dans un graphe donné il existe un cycle passant une unique fois par chaque sommet.

La réduction se fait depuis HAMPATH.

Proposition : Le Voyageur de commerce
Le Problème du voyageur de Commerce est NP-complet. [Problème 01]

La réduction se fait depuis HAMCYCLE

III Des problèmes NP-complets sur les entiers

Définition & Proposition 5 : SET-COVER
Le problème consiste à déterminer si étant donné un ensemble fini U, un ensemble S de parties de U et un entier k il existe un sous-ensemble de S à k éléments qui recouvre U. Le problème est NP-complet.

La Réduction se fait depuis VERTEX-COVER.

Définition & Proposition 6 : SUBSET-SUM
Étant donné un entier t et un ensemble S d'entiers, le problème consiste à déterminer s'il existe un sous-ensemble de S de somme t.
Le problème est NP-complet.

La réduction se fait depuis 3-SAT

IV La Programmation linéaire en entiers

La Programmation linéaire est à l'origine un problème d'optimisation sous la forme

$$\begin{cases} \text{Minimiser } c \cdot x \\ Ax \leq b \end{cases}$$

où c est un vecteur, A une Matrice, b un vecteur.

Si x est continue, ce problème est dans P, mais devient plus compliqué si on restreint x à des valeurs entières.

Définition - Proposition 7: PLE

Le Problème PLE consiste à chercher si étant donné $A \in \mathcal{M}_{m,n}(\mathbb{Z})$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$ et $k \in \mathbb{Z}$, il existe $x \in \mathbb{Z}^n$ tel que $\begin{cases} c \cdot x \leq k \\ Ax \leq b \end{cases}$.

PLE est NP complet

Reduction de Problemes NP-complets à PLE :

1. SET-COVER

On se place sur $\mathbb{Z}^S$ et on cherche $x \in \mathbb{Z}^S$ tel que :

- $\sum_S x_S \leq k$
- $\forall a = (s_1, s_2),\ x_{s_1} + x_{s_2} \geq 1$.

SET-COVER est équivalent à l'existence d'un tel x.

2. KNAPSACK

Le problème consiste à déterminer, étant donné un volume V, des couples $(v_1, s_1), \dots, (v_n, s_n)$ et $k \in \mathbb{N}$ si il existe un sous-ensemble S de $[\![1, n]\!]$ tel que

$$\begin{cases} \sum_{i \in S} s_i \geq k \\ \sum_{i \in S} v_i \leq V \end{cases}$$

Ce problème s'exprime à l'aide de PLE.

3. Le Voyageur de Commerce

Étant donné une instance $(\{v_i, i \in [\![1,n]\!]\}, d_{i,j})$ du problème du Voyageur de commerce, on considère le problème :

$\rightarrow \sum x_{i,j} d_{i,j} \leq k$

$\rightarrow \forall i \in [\![1, n]\!], \forall j \in [\![1, n]\!], 0 \leq x_{i,j} \leq 1$

$\rightarrow \forall i \in [\![1, n]\!], \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{i,j} = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{j,i} = 1$

$\rightarrow \forall S \in \mathcal{P}([\![1, n]\!]), \sum_{i \in S, j \notin S} x_{i,j} \geq 2$

Cela constitue une reduction à PLE. Cependant elle n'est pas polynomiale.

V Le Problème PSA [DEV]

Théorème 03 : Le Problème de séparation par automate est NP-complet.

Il s'agit pour des langages L_1 et L_2 et pour $k \in \mathbb{N}$ de chercher l'existence d'un automate fini déterministe à k états qui accepte tout mots de L_1 et refuse tout ceux de L_2.

[DICHORNOY]