Complexité - TD 01

Benjamin Bordais

19 Novembre 2020

Ici, on confond les notions de langage et de problème de décision. En effet, à partir d'un problème de décision, on peut considérer le langage associé qui est le langage des instances positives de ce problème de décision. De même, à partir d'un langage, on peut considérer le problème qui consiste à décider si une instance appartient à ce langage.

Les réductions considérées sont en espace logarithmique (sauf spécification contraire). On rappelle la définition du problème de décision SAT :

— ENTRÉE : une formule propositionnelle ϕ sous forme normale conjonctive
— SORTIE : ϕ est satisfiable

Ce problème est NP-complet.

On donne également la définition suivante d'une coloration d'un graphe. Une *k-coloration* d'un graphe non orienté $G = (V, E)$ est une fonction $c : V \to \{0, \dots k-1\}$ telle que si $\{u, v\} \in E$ alors $c(u) \neq c(v)$. Le problème $3 - \mathsf{COLORATION}$ est défini ainsi :

— ENTRÉE : un graphe non orienté G
— SORTIE : il existe une 3-coloration de G

Ce problème est NP-complet.

Échauffement

Réduction en temps polynomial

Soit $L \subseteq \Sigma^*$ un langage sur l'alphabet Σ. À quelles conditions L est PTIME-dur pour des réductions en temps polynomial ?

Solution :

Tout langage L tel qu'il existe $w_i \in L$ et $w_o \notin L$ est PTIME-dur pour des réductions en temps polynomial. En effet, considérons un langage $L' \in \mathsf{PTIME}$. On pose alors la réduction $f : \Sigma^* \to \Sigma^*$ qui, sur une entrée $x \in \Sigma^*$, décide en temps polynomial si $x \in L'$ (ce qui est possible car $L' \in \mathsf{PTIME}$) et qui renvoie w_i (resp. w_o) si $x \in L'$ (resp. si $x \notin L'$). On a bien $x \in L' \Leftrightarrow f(x) \in L$ et la transformation f calculable en temps polynomial. Ainsi, tout langage non-trivial (c'est-à-dire différent de $\emptyset$ et Σ^*) est PTIME-dur pour des réductions en temps polynomial.

C'est pour cela que lorsque l'on considère la dureté pour une classe, on considère des réductions qui sont moins coûteuses (en temps ou en espace) qu'un algorithme pour décider des problème de décision de cette classe. Ainsi, les langages PTIME-dur sont usuellement considérés pour des réductions en espace logarithmique.

Un simple problème NP-complet

Soit L le langage $\{(M, x, 1^t) \mid M \text{ accepte sur } x \text{ en temps au plus } t\}$ avec M le code d'une machine de Turing non-déterministe. Montrer que L est NP-complet.

Solution :

Il faut d'abord montrer que $L \in \mathsf{NP}$. On considère l'algorithme non déterministe qui, sur une entrée $(M, x, 1^t)$ simule M sur x (en simulant le non-déterminisme de M) en conservant un time-out qui compte le nombre d'étape effectué. Si ce time-out dépasse t, cet algorithme rejette. Sinon, on imite ce qu'aurait fait la machine M. Cet algorithme décide bien L et est en temps polynomial car le time-out t est donné en unaire dans l'entrée (ainsi, avoir un compteur allant jusqu'à t correspond bien à un temps polynomial en la taille de l'entrée). Ainsi, $L \in \mathsf{NP}$.

Montrons à présent de L est NP-dur. Soit $L' \in \mathsf{NP}$. Il existe alors une machine de Turing non déterministe M et un polynôme p tels que M décide L et termine en temps au plus $p(n)$ où n est la taille de l'entrée. On construit alors une réduction $f : \Sigma^* \to \Sigma^*$ ainsi : pour $x \in \Sigma^*$, on a $f(x) = (M, x, 1^{p(|x|)})$. Par définition de L et M, on a bien $x \in L' \Leftrightarrow f(x) \in L$. De plus f peut se calculer en espace logarithmique : la machine M est fixé on la recopie sur la bande de sortie en temps constant, on recopie ensuite l'entrée x et on a ensuite besoin d'un compteur (en binaire) de taille $\log(p(|x)) = O(\log(|x|))$. Ainsi, L est NP-dur pour des réduction en espace logarithmique.

Problème NP-complet sur des graphes

Ensemble indépendent

Un *ensemble indépendant* dans un graphe non orienté $G = (V, E)$ est un ensemble $C \subseteq V$ de sommets dont aucun n'est relié à aucun autre par une arête de G, c'est-à-dire tel que $u, v \in C$ implique $\{u, v\} \notin E$. Démontrer que le langage $\mathsf{INDEPENDENT-SET}$ défini comme suit est NP-complet.
ENTRÉE : un graphe non orienté $G = (V, E)$, un entier $m \in \mathbb{N}$;
SORTIE : G a-t-il un ensemble indépendant de cardinal au moins m

Solution :

Vérifions d'abord que $\mathsf{INDEPENDENT-SET} \in \mathsf{NP}$. On peut considérer un algorithme non-déterministe qui devine un ensemble de sommets de cardinal m, puis qui vérifie (en temps polynomial) que celui-ci est bien indépendant (à l'aide de deux boucles imbriqués sur les éléments de cet ensemble). Cet algorithme décide $\mathsf{INDEPENDENT-SET}$ en temps polynomial.

Montrons à présent que ce problème est NP-dur. Pour cela, on construit une réduction f depuis SAT. Considérons une instance $\phi = \wedge_{i=1}^{p} C_i$ avec $C_i = \vee_{j=1}^{a_i} l_{i,j}$ et $l_{i,j}$ un littéral, sur les variables $(x_k)_{1 \leq k \leq n}$. On construit alors le graphe $G_\phi = (V, E)$ tel que $V = \{l_{i,j} \mid 1 \leq i \leq p,\ 1 \leq j \leq a_i\} \cup \{x_k, \neg x_k \mid 1 \leq k \leq n\}$ et $E = \{(l_{i,j}, l_{i,j'}) \mid 1 \leq i \leq p,\ j, j' \leq a_i, j \neq j'\} \cup \{(x_k, \neg x_k) \mid 1 \leq k \leq p\} \cup \{(x_k, l_{i,j}), (\neg x_k, l_{i',j'}) \mid \text{if } l_{i,j} = \neg x_k \text{ and } l_{i',j'} = x_k\}$. Plus explicitement, à chaque clause on associe un ensemble de sommets (un par littéral) connectés deux à deux. Ainsi, un ensemble indépendant contiendra au plus un élément de chaque clause. De même, pour chaque variable, on a associé deux sommets qui sont connectés. De même, tout ensemble indépendant contiendra au plus l'un de ces deux sommets. On pose alors $f(\phi) = (G_\phi, m)$ avec $m = n + p$. C'est-à-dire, si ensemble indépendant de taille m existe, alors il contient exactement un littéral par clause et, pour chaque variable, son instance positive ou négative. On peut dès à présent se rendre compte que cette réduction peut s'effectuer en espace logarithmique, car il suffit d'avoir deux boucles imbriquées parcourant l'ensemble des clauses et variables de la formule pour construire les ensembles V et E. Montrons à présent que $\phi \in \mathsf{SAT} \Leftrightarrow f(\phi) \in \mathsf{INDEPENDENT-SET}$.

Supposons que $\phi \in \mathsf{SAT}$. Alors il existe une valuation des variables $(x_k)_{1\leq k\leq n}$ telle que chaque clause soit satisfaite – c'est-à-dire qu'il existe (au moins) un littéral parmi chaque clause qui est satisfait. On considère alors comme ensemble indépendant l'ensemble constitué des variables (et de leur négation) satisfaites par la valuation et d'un litéral par clause satisfait par cette valuation. Cet ensemble est bien de taille $m = n + p$. De plus, il est bien indépendant car si un littéral $l_{i,j} = x_k$, alors il y a une arête entre $l_{i,j}$ et $\neg x_k$. Mais si la valuation satisfait $l_{i,j}$, alors l'ensemble consdéré contiendra x_k et pas $\neg x_k$, et ainsi il n'y aura pas d'arête entre deux éléments de l'ensemble. De même si $l_{i,j} = \neg x_k$. Il vient que $f(\phi) \in \mathsf{INDEPENDENT} - \mathsf{SET}$.

De manière similaire, supposons $f(\phi) \in \mathsf{INDEPENDENT} - \mathsf{SET}$. D'après les remarques précédentes, un ensemble indépendant F de taille $m = n + p$ contient exactement une variable ou sa négation (ainsi qu'un littéral par clause). Considérons alors la valuation des variables qui correspond aux variables (ou à leur négation) qui est dans F. On peut vérifier que cette valuation satisfait la formule ϕ. En effet, pour chaque clause C_i, considérons le littéral $l_{i,j}$ qui est dans F (on sait qu'il y en a un par clause car $m = n + p$). Par exemple, $l_{i,j} = x_k$. Alors, comme F est indépendant, le nœud $\neg x_k$ n'est pas dans F (car $(l_{i,j}, \neg x_k) \in E$). Donc, $x_k \in F$. Ainsi, la valuation valus positivement la variable x_k, et satisfait donc $l_{i,j}$ et C_i. On en conclut que $\phi \in \mathsf{SAT}$.

Ensemble couvrant

Un *recouvrement* C d'un graphe non orienté $G = (V, E)$ est un ensemble $C \subseteq V$ de sommets tel que toute arête de E est incidente à C, c'est-à-dire à au moins un élément de C. Démontrer que le langage $\mathsf{NODE} - \mathsf{COVER}$ défini comme suit est NP-complet.
ENTRÉE : un graphe non orienté $G = (V, E)$, un entier $m \in \mathbb{N}$;
SORTIE : G a un recouvrement de cardinal au plus m

Solution :

Ce problème est bien dans NP : on devine un candidat potentiel (ici, un ensemble de taille m), et on vérifie qu'il vérifie bien la propriété souhaitée (en temps polynomial, ici avec une boucle sur les arêtes et une autre pour vérifier qu'un nœud de l'ensemble candidat est l'extrémité de l'arête considéré.)

Pour prouver la NP-dureté, on peut réduire directement depuis le problème précédent en remarquant que : C est un recouvrement de G (de cardinal au plus m) si et seulement si $V \setminus C$ est un ensemble indépendant de G (de cardinal au moins $n - m$) où $n = |V|$. Ainsi la réduction f considéré vérifie $f(G, m) = (G, n - m)$.

Clique

Une *clique* C d'un graphe non orienté $G = (V, E)$ est un sous-ensemble $C \subseteq V$ induisant un sous-graphe complet de G, c'est-à-dire tel que pour tous $u, v \in C$ avec $u \neq v$, on a $\{u, v\} \in E$. Montrer que le problème CLIQUE défini comme suit est NP-complet.
ENTRÉE : un graphe non orienté $G = (V, E)$, un entier $m \in \mathbb{N}$;
SORTIE : G a une clique de cardinal au moins m

Solution :

Comme pour les deux cas précédents, il est facile de prouver que ce problème est bien dans NP : on devine un candidat potentiel (un ensemble de taille m), et on vérifie qu'il vérifie bien la propriété souhaitée (en temps polynomial, ici avec deux boucles imbriquées sur les paires de sommets de l'ensemble candidat.)

On peut ensuite montrer une réduction depuis $\mathsf{INDEPENDENT-SET}$. Pour un graphe $G = (V, E)$, on consière le graphe $\overline{G} = (V, \overline{E})$, tel que $\overline{E} = \{(u, v) \mid u, v \in V, u \neq v, \{u, v\} \notin E\}$. Alors C est une clique de G si et seulement si C est un ensemble indépendant de $\overline{G}$. On pose donc $f(G, m) = (\overline{G}, m)$ qui est calculable en espace polynomial, avec une boucle sur les arêtes de G.

Homomorphisme de graphe

Un homomorphisme d'un graphe $G = (V, E)$ à un graphe $G' = (V', E')$ est une fonction $h : V \to V'$ telle que pour tout $\{v_1, v_2\} \in E$, on a $\{h(v_1), h(v_2)\} \in E'$. Montrer que le problème $\mathsf{GRAPH-HOMOMORPHISM}$ défini ci-après est NP-complet.
ENTRÉE : deux graphes non orientés, G_1 et G_2 ;
SORTIE : il existe un homomorphisme de G_1 à G_2

Solution :
On doit toujours vérifier que le problème est dans NP. Comme précédemment, on devine un candidat potentiel (une fonction h représenté par un ensemble de paires dans $V \times V'$), et on vérifie qu'elle vérifie bien la propriété souhaitée (en temps polynomial, ici avec une boucle sur les arêtes de E, puis en consultant la fonction h et l'ensemble E').

Pour ce qui est de la NP-dureté, on peut réduire depuis le problème de $3-\mathsf{COLORATION}$. En effet, un graphe G est 3-coloriable si et seulement si il existe un homomorphisme entre G et le graphe K_3 (où K_n est le graphe complet (i.e. où tous les sommets sont reliés deux à deux) avec n sommets). Dans ce cas, on a $f(G) = (G, K_3)$ (qui se calcule en espace constant).

Isomorphisme de graphe

Deux graphes $G = (V, E)$ et $G' = (V', E')$ sont isomorphes si $|V| = |V'|$ et $|E| = |E'|$ et il existe une fonction bijective $h : V \to V'$ telle que $\{v_1, v_2\} \in E$, si et seulement si $\{h(v_1), h(v_2)\} \in E'$. Montrer que le problème $\mathsf{GRAPH-ISOMORPHISM}$ défini ci-après est NP-complet.
ENTRÉE : Deux graphes G et H.
QUESTION : G contient un sous-graphe isomorphe à H

Solution :
Comme pour tous les autres problèmes, il faut vérifier que ce problème est dans NP. Dans le cas présent, on devine un candidat potentiel (deux sous-ensemble V' et E', et une fonction h représenté par un ensemble de paires dans $V \times V'$), et on s'assure qu'elle vérifie bien la propriété souhaitée (en temps polynomial, on vérifie que l'on a bien $|V| = |V'|$, $|E| = |E'|$ et que l'on a bien un isomorphisme).

Pour la NP-dureté, on peut réduire depuis $\mathsf{INDEPENDENT-SET}$. En effet, G a un ensemble indépendant de cardinal au moins m si et seulement si il existe un sous-graphe de G isomorphe à $\overline{K_m}$ (le graphe avec m sommets sans aucune arête). Ici, $f(G, m) = (G, K_m)$. Il faut remarquer un piège ici qui dépend de l'écriture de m. Si m est écrit en binaire, on ne pourra pas écrire K_m sur la bande de sortie en utilisant seulement un espace logarithmique – puisque K_m est de taille exponentiel en le nombre de bit nécessaire pour écrire m en binaire (on n'a pas de problème si m est écrit un unaire). On peut dans un premier temps comparer m et $n = |V|$. Si $m > n$, alors on renvoie une trivialement négative du problème $\mathsf{GRAPH-ISOMORPHISM}$ (par exemple une instance où H a plus de sommets que G).

Dans le cas contraire, on renvoie la réduction $f(G, m) = (G, K_m)$ à l'aide d'une boucle allant jusqu'à m, ce qui est plus petit que le nombre de sommets de G.

Bonus

Prouver que le problème $3-\mathsf{COLORATION}$ est NP-complet. Pour prouver la NP-dureté de ce problème, on pourra effectuer une réduction depuis le problème 3-SAT, également NP-complet, qui une restriction de SAT aux formules avec au plus trois literaux par clause.