IFT503/711 - Exercices sur P et NP
Solutions

Manuel Lafond

Hiver 2020

Exercice 1

Montrez que P $\subseteq$ NP $\cap$ co-NP.

Solution.

On sait que P $\subseteq$ NP, car s'il y a une MT déterministe M qui accepte un langage $L \in$ P, M peut être vue comme une MT non-déterministe qui n'a qu'un seul choix pour chaque transition. Le langage accepté par M est L, et donc $L \in$ NP.

On montre ensuite que $L \in P \iff \overline{L} \in P$ (dans le jargon, on dit que P est fermé sous le complément). Ceci est relativement facile à voir : une MT qui décide L peut servir directement à déterminer si un mot est dans $\overline{L}$ ou non. Voici les détails.

Soit $L \in$ P, et soit M une MT qui décide L en temps polynomial. Considérez $\overline{L}$. On a $w \in L \iff w \notin \overline{L}$. Donc, on peut prendre la machine M' qui décide $\overline{L}$ de la façon suivante : sur entrée w, simuler M sur entrée w et retourner le résultat inverse. Lorsque $w \in \overline{L}$, M rejettera w et donc M' l'acceptera. Inversément, lorsque $w \notin \overline{L}$, M acceptera w et M' le rejettera. Donc M' décide $\overline{L}$. Puisque simuler M prend un temps polynomial, ceci veut dire que $\overline{L} \in P$.

Puisque L a été choisi arbitrairement dans P, ceci montre que $L \in P \iff \overline{L} \in P$. En particulier, $\overline{L} \in NP$. Puisque le complément de L est dans NP, on a $L \in$ co-NP. Puisque ceci est vrai pour tout $L \in$ P, on a P $\subseteq$ co-NP.

Exercice 2

Un langage L est co-NP-complet si $L \in$ co-NP, et si, pour chaque langage $A \in$ co-NP, $A \leq_P L$.

Montrez qu'un langage L est NP-complet si et seulement si $\overline{L}$ est co-NP-complet.

Solution.

($\Rightarrow$) : supposons que L est NP-complet. Alors $L \in$ NP, et donc $\overline{L} \in$ co-NP. Maintenant, soit $A \in$ co-NP. On veut montrer que A se réduit à $\overline{L}$. Par définition, on a $\overline{A} \in$ NP, et donc $\overline{A} \leq_P L$ puisque L est NP-complet. Soit f une fonction de réduction polynomiale telle que $w \in \overline{A} \iff f(w) \in L$. Il s'avère qu'on peut aussi utiliser f pour réduire A à $\overline{L}$. C'est-à-dire, on montre que $w \in A \iff f(w) \in \overline{L}$.

Soit $w \in A$. Alors $w \notin \overline{A}$, et donc $f(w) \notin L$, ce qui implique que $f(w) \in \overline{L}$. Dans l'autre sens, soit $w \notin A$. Alors $w \in \overline{A}$, et donc $f(w) \in L$, ce qui implique que $f(w) \notin \overline{L}$. Ceci montre qu'un langage arbitraire dans co-NP se réduit à $\overline{L}$, et donc que $\overline{L}$ est co-NP-difficile.
($\Leftarrow$) : la preuve est presque identique à la précédente, donc nous donnons une version abrégée. Supposons que $\overline{L}$ est co-NP-complet. Alors $\overline{L} \in$ co-NP et $L \in$ NP. Maintenant, soit $A \in$ NP. On veut montrer que A se réduit à L. On a $\overline{A} \in$ co-NP, et donc il existe une réduction f telle que $w \in \overline{A} \iff f(w) \in \overline{L}$. Ceci est équivalent à $w \in A \iff f(w) \in L$. Il s'ensuit que f est une réduction de A vers L.

Exercice 3

Montrez que $A_{NTM} = \{\langle M, x\rangle : M$ encode une MT non-déterministe qui accepte $x\}$ est NP-difficile.

Solution.

Soit $L \in$ NP, et soit M l'encodage d'une MT non-déterministe dont le langage est L. Soit $x \in \Sigma^*$. Notre réduction f transforme x en $f(x) = \langle M, x\rangle$. Puisque M est de taille constante, f peut être exécutée en temps polynomial par rapport à x. On montre que $x \in L \iff f(x) \in A_{NTM}$.

Si $x \in L$, alors M accepte x. Donc, $f(x) = \langle M, x\rangle \in A_{NTM}$. Si $x \notin L$, alors M n'accepte pas x. Donc, $f(x) = \langle M, x\rangle \notin A_{NTM}$.

Exercice 4

Montrez que IND-SET $\in$ PSPACE.

Solution.

Soit $\langle G, k\rangle$ une instance de IND-SET. On peut vérifier si G contient un ensemble indépendant de taille k ou plus par force brute, come suit :

- Pour chaque sous-ensemble X de $V(G)$:
  - Vérifier si $|X| \geq k$ et si X est un ensemble indépendant.
  - Si oui, Accepter.
- Rejeter.

Notez que l'on pourrait se contenter d'énumérer les sous-ensembles de taille k, ce qui ne changerait rien au résultat. Il n'est pas difficile d'énumérer chaque sous-ensemble possible en espace polynomial. Par exemple, on peut stocker un vecteur de bits de taille n démarrant à 0, l'interpréter comme un entier entre 1 et $2^{n+1} - 1$, et l'incrémenter à chaque tour de boucle (ici, n est le nombre de sommets). Chaque vecteur de bits représente un sous-ensemble possible : si le i-ème bit est à 1, alors on inclut le i-ème sommet. L'espace requis pour faire cette énumération est polynomial, et chaque vérification se fait clairement en temps, et donc en espace, polynomial.

Exercice 5

Soit EXPTIME $= \bigcup_{k\in\mathbb{N}} DTIME(2^{n^k})$. Montrez que NP $\subseteq$ EXPTIME, possiblement en utilisant la notion de certificat.

Solution.

Soit $L \in$ NP, et soit V un vérificateur pour L, c'est-à-dire une MT déterministe telle que $w \in L \iff$ il existe un mot c de taille $O(|w|^k)$, $k \in \mathbb{N}$, tel que V accepte $\langle w, c\rangle$ en temps $O(|w|^d)$, $d \in \mathbb{N}$. Supposons que c est toujours de taille au plus $\alpha|w|^k$, où α est la constante cachée dans la notation O.

On veut montrer que $L \in$ EXPTIME. Soit la MT M qui, sur entrée w, simule V sur toutes les valeurs possibles de $\langle w, c\rangle$, où c est de taille au plus $\alpha|w|^k$, et accepte w si et seulement si V a accepté au moins une telle paire. Puisque V est un vérificateur, il s'ensuit que M décide L.

Le nombre de mots de taille au plus αn^k est borné par $O(|\Sigma|^{\beta n^k})$ pour une certaine constante β (si vous ne le voyez pas, faites-le en exercice). Ceci est $O(2^{\log|\Sigma|\cdot\beta n^k})$. Puisque $|\Sigma|$ et β sont des constantes, il existe $k' \geq k$ tel que ceci est $O(2^{n^{k'}})$. La MT M' décide donc L en un temps borné par $O(n^d \cdot 2^{n^{k'}}) = O(2^{d\log n} \cdot 2^{n^{k'}})$, ce qui est $O(2^{n^{k''}})$, $k'' \in \mathbb{N}$. Ceci démontre que $L \in$ EXPTIME.

Exercice 6

Dans le problème de la couverture minimum, on reçoit des ensembles, et on veut choisir un minimum de ces ensembles pour couvrir tous les éléments. Plus formellement, soit $\mathcal{S} = \{S_1, \ldots, S_n\}$ une collection d'ensembles, et soit U un ensemble qu'on appelle *l'univers*. Le langage correspondant au problème de la couverture minimum est le suivant :

$$\text{SET-COVER} = \{\langle \mathcal{S}, U, k\rangle : \text{il existe } \mathcal{S}' \subseteq \mathcal{S} \text{ tel que } |\mathcal{S}'| \leq k \text{ et } \bigcup_{S \in \mathcal{S}'} S = U\}$$

Montrez que SET-COVER est NP-complet.

Solution.

Il est clair que SET-COVER est dans NP, car une sous-collection $\mathcal{S}' \subseteq \mathcal{S}$ peut servir de certificat : il est facile de vérifier que $|\mathcal{S}'| \leq k$ et que $\mathcal{S}'$ couvre U.

On montre que SET-COVER est NP-difficile, par réduction de CNF-SAT. Soit ϕ une formule CNF-SAT, et soient $x_1, \ldots, x_n$ les variables utilisées par ϕ. De plus, soient $C_1, \ldots, C_m$ les clauses de ϕ.

On créé une instance correspondante $\langle \mathcal{S}, U, k\rangle$ de SET-COVER. Les éléments de l'univers sont $U = \{C'_1, \ldots, C'_m, x'_1, \ldots, x'_n\}$, chaque élément C'_i correspondant à une clause et chaque élément x'_i correspondant à une variable. Les ensembles sont $\mathcal{S} = \{S_1, \overline{S}_1, \ldots, S_n, \overline{S}_n\}$, où S_i correspond à x_i et $\overline{S}_i$ à $\overline{x}_i$. Pour chaque i, $1 \leq i \leq n$, le contenu de S_i et $\overline{S}_i$ sont

$$S_i = \{x'_i\} \cup \{C'_j : C_j \text{ est satisfaite par } x_i\}$$
$$\overline{S}_i = \{x'_i\} \cup \{C'_j : C_j \text{ est satisfaite par } \overline{x}_i\}$$

On pose $k = n$.

On montre maintenant que ϕ est satisfaisable si et seulement si U peut être couvert avec au plus n ensembles de $\mathcal{S}$.

($\Rightarrow$) : supposons qu'il existe une affectation A de $x_1, \ldots, x_n$ qui satisfait ϕ. On construit $\mathcal{S}' \subseteq \mathcal{S}$ de la façon suivante. Pour chaque i, où $1 \leq i \leq n$, on ajoute S_i à $\mathcal{S}'$ si x_i est positif dans A, et on ajoute plutôt $\overline{S}_i$ à $\mathcal{S}'$ si x_i est négatif dans A. Il est clair que $|\mathcal{S}| \leq n$. De plus, $\mathcal{S}$ couvre tous les éléments $\{x'_1, \ldots, x'_n\}$ car on ajouté ajouté un ensemble S_i ou $\overline{S}_i$ pour chaque variable x_i. Chaque élément-clause C'_i est aussi couvert : puisque A satisfait ϕ, une variable x_j est affectée de façon à satisfaire la clause C_i, ce qui implique que S_j ou $\overline{S}_j$, selon le cas, couvre C'_i.

($\Leftarrow$) : supposons qu'il existe $\mathcal{S}' \subseteq \mathcal{S}$ de taille au plus n qui couvre U. On remarque que pour couvrir $x'_1, \ldots, x'_n$, il faut ajouter au moins un de S_i ou $\overline{S}_i$ pour chaque $1 \leq i \leq n$. Puisque $|\mathcal{S}'| \leq n$, ceci veut dire qu'on a en fait dans $\mathcal{S}'$ exactement un de S_i ou $\overline{S}_i$, mais pas les deux.

Pour une solution à ϕ, on affecte x_i à vrai si $S_i \in \mathcal{S}$, et x_i à faux si $\overline{S}_i \in \mathcal{S}$, pour chaque $1 \leq i \leq n$. Notons que x_i n'est pas à la fois vrai et faux. De plus, chaque clause C_j est satisfaite par le x_i tel que S_i ou $\overline{S}_i$ couvre C'_j, selon le cas. Donc, ϕ est satisfaisable.

Exercice 7

Soit $G = (V, E)$ un graphe. Une *coupe* de G est une partition de V en deux parties $\{V_1, V_2\}$ non-vides. La *taille* d'une coupe $\{V_1, V_2\}$ est $|\{uv \in E : u \in V_1, v \in V_2\}|$, donc le nombre d'arêtes dont les extrémités sont dans V_1 et V_2. On définit

$$\text{MAX-CUT} = \{\langle G, k\rangle : G \text{ a une coupe de taille au moins } k\}$$

Montrez que MAX-CUT est NP-complet.

Solution.

On voit que MAXCUT $\in$ NP, car une coupe $\{V_1, V_2\}$ peut servir de certificat. En effet, étant donné $\{V_1, V_2\}$, il suffit de compter le nombre d'arêtes dont les extrémités sont d'un côté et de l'autre.

On montre maintenant que MAXCUT est NP-difficile. On applique une réduction à partir de INDSET. Soit $\langle G, k\rangle$ une instance de INDSET. On créé une instance correspondante $\langle G', k'\rangle$ de MAXCUT. Je vous invite à dessiner la réduction décrite ci-bas.

Pour obtenir G', on prend d'abord les sommets de V et on ajoute un sommet spécial x. Pour chaque arête $uv \in E(G)$, on ajoute aussi deux sommets appelés $[u, v]$ et $[v, u]$. Pour les arêtes, on rend d'abord x voisin de tous les autres sommets. Pour chaque $u \in V(G)$ et chaque voisin v de u dans G, on ajoute dans G' une arête entre u et $[u, v]$. On ajoute aussi l'arête entre $[u, v]$ et $[v, u]$ (ajoutée une seule fois). On définit $k' = k + 4|E(G)|$.

On montre que G a un ensemble indépendant avec k sommets ou plus si et seulement si G' a une coupe de taille k' ou plus.

($\Rightarrow$) : soit I un ensemble indépendant de G tel que $|I| \geq k$. On définit la coupe $\{V_1, V_2\}$ de G'. D'abord, on ajoute x à V_2. Pour chaque u dans I, on ajoute u dans V_1 et pour chaque v dans $V(G) \setminus I$, on ajoute v dans V_2. On note que les k arêtes $\{ux : u \in I\}$ sont dans la coupe.

Ensuite, pour chaque arête uv telle que $u \in I$, on ajoute $[v, u]$ à V_1 et $[u, v]$ à V_2 (on note que $v \notin I$). Ceci définit 4 arêtes dans la coupe qui correspondent à uv : on a $x[v, u], [u, v][v, u], v[v, u]$ et $u[u, v]$.

Puis, pour chaque arête uv telle que $u, v \notin I$, on ajoute $[u, v]$ et $[v, u]$ à V_1. Puisque $u, v \in V_2$, ceci définit 4 arêtes dans la coupe qui correspondent à uv : on a $x[u, v], x[v, u], u[u, v]$ et $v[v, u]$.

Au total, la coupe contient les k premières arêtes décrites, plus 4 arêtes de coupe pour chaque arête de G, totalisant $k + 4|E(G)|$, tel que désiré.

Il reste à vérifier que toutes les arêtes décrites sont différentes — ce que nous balayons du revers de la main en disant que c'est en exercice.

($\Leftarrow$) : soit $\{V_1, V_2\}$ une coupe de G' de taille au moins $k + 4|E(G)|$. Soit E_c les arêtes de la coupe. Considérez une arête $uv \in E(G)$ et les arêtes de G' suivantes : $u[u, v], v[v, u], [u, v][v, u], x[u, v], x[v, u]$. On appelle ces 5 arêtes le *gadget* de uv. Nous vous laissons vérifier (par force brute) que E_c peut contenir

au maximum 4 de ces arêtes. De plus, si u, v ne sont *pas* dans la même partie de coupe que x, E_c peut contenir seulement 3 de ces arêtes. Soit ℓ le nombre d'arêtes uv de G telles que E_c contient 3 arêtes ou moins du gadget de uv. Donc, le nombre d'arêtes de E_c dans les gadgets des arêtes de G est au maximum $4(|E(G)| - \ell) + 3\ell = 4|E(G)| - \ell$. Puisque $|E_c| \geq k + 4|E(G)|$, ceci veut dire que E_c doit prendre $k + \ell$ arêtes supplémentaires de ailleurs, la seule possibilité étant les arêtes de x vers V.

Soit $S = \{v \in V : xv \in E_c\}$, où $|S| \geq k + \ell$. On obtient $S' \subseteq S$ de la façon suivante. Pour chaque u, v dans S tel que $uv \in E(G)$, on retire u ou v de S de façon arbitraire. Puisque chaque paire u, v considérée correspond à un gadget avec 3 arêtes de coupe, il y a ℓ telles paire, et nous allons donc retirer de S au maximum ℓ sommets. Après ces retraits, S' contient k sommets ou plus, et aucun de ces sommets ne partagent un arête. Donc, S' est un ensemble indépendant de taille au moins k.

Exercice 8

Dans un graphe, un sous-ensemble de sommets $X \subseteq V(G)$ est *dominant* si chaque sommet de $V(G) \setminus X$ a au moins un voisin dans X. Considérez le langage

$$\text{DOM-SET} = \{\langle G, k\rangle : G \text{ est un graphe dans lequel il existe un ensemble dominant } X \subseteq V(G) \text{ de taille } k \text{ ou moins}\}$$

Montrez que DOM-SET est NP-complet.

Solution.

Cette réduction est très similaire à celle de SET-COVER, donc nous donnons une version abrégée et plus informelle.

On réduit de CNF-SAT. Sur instance ϕ, créez un graphe G dans lequel il y a les sommets $s_1, \overline{s}_1, \ldots, s_n, \overline{s}_n$. Ajoutez les sommets $c_1, \ldots, c_m$, et une arête entre s_i et c_j si x_i satisfait C_j, ou entre $\overline{s}_i$ et c_j si $\overline{x}_i$ satisfait C_j. Ajoutez ensuite les sommets $x'_1, \ldots, x'_n$ tels que x'_i est voisin de s_i et $\overline{s}_i$. On cherche à savoir si G a un ensemble dominant de taille au plus n.

Si ϕ est satisfaisable par une assignation A, prenez les éléments correspondants dans $s_1, \overline{s}_1, \ldots, s_n, \overline{s}_n$ pour dominer G. Si G a un ensemble dominant de taille n, il doit contenir un sommet entre s_i et $\overline{s}_i$ pour dominer x'_i. De plus, ces sommets doivent dominer les c_j, et donc l'assignation correspondante satisfait ϕ.

Exercice 9

Une collection d'ensembles $\{S_1, \ldots, S_\ell\}$ est dite *disjointe* si, pour tout i, j avec $1 \leq i < j \leq \ell$, on a $S_i \cap S_j = \emptyset$. Dans le problème SET-PACKING, on reçoit une collection d'ensembles $\mathcal{S}$ et on doit choisir une sous-collection disjointe de taille maximum. En terme de langage, on a :

SET-PACKING $= \{\langle \mathcal{S}, k\rangle$:il existe une sous-collection $\mathcal{S}' \subseteq \mathcal{S}$ disjointe telle que $|\mathcal{S}'| \geq k\}$

Montrez que SET-PACKING est NP-complet.

Solution.

SET-PACKING est dans NP car une sous-collection $\mathcal{S}$ peut servir de certificat : on peut vérifier en temps polynomial que $|\mathcal{S}| \geq k$ et que chaque paire d'ensembles de $\mathcal{S}$ ont une interesection vide.

On montre que SET-PACKING est NP-difficile par réduction via IND-SET. Soit $\langle G, k\rangle$ une instance de IND-SET. Notre collection $\mathcal{S}$ sera sur l'univers $U = E(G)$, i.e. on doit couvrir un élément correspondant à chaque arête. Pour chaque sommet $u \in V(G)$, on ajoute à $\mathcal{S}$ un ensemble

$$S_u = \{uv : v \text{ est un voisin de } u \text{ dans } G\}$$

i.e. S_u permet de couvrir chaque arête touchant à u. On montre que G a un ensemble indépendant de taille k si et seulement si il existe une sous-collection disjointe $\mathcal{S}' \subseteq \mathcal{S}$ telle que $|\mathcal{S}'| \geq k$ (donc ici, le k est le même pour les deux problèmes).

($\Rightarrow$) soit I un ensemble indépendant de G, avec $|I| \geq k$. Soit $\mathcal{S}' = \{S_u : u \in I\}$. Il est évident que $|\mathcal{S}'| \geq k$. De plus, puisque pour toute paire $u, v \in I$, il n'y a pas d'arête entre u et v, les arêtes incidentes à u sont disjointes des arêtes incidentes à v. Donc, $S_u \cap S_v = \emptyset$. On conclut que $\mathcal{S}'$ est une sous-collection disjointe de taille au moins k.

($\Leftarrow$) soit $\mathcal{S}'$ une sous-collection disjointe de $\mathcal{S}$ de taille au moins k. Soit $I = \{u : S_u \in \mathcal{S}'\}$. Bien sûr, $|I| \geq k$. Par construction de $\mathcal{S}$, si $S_u, S_v \in \mathcal{S}'$, alors u et v ne partagent pas d'arête. Puisque ceci est vrai pour tout $u, v \in I$, il s'ensuit que I est un ensemble indépendant.

Exercice 10

Soit G un graphe. Un *arbre couvrant* de G est un sous-graphe G' de G tel que G' est un arbre connectant tous les sommets de G. Dans le problème MINDEG-AC, on veut savoir s'il existe un arbre couvrant dont le degré maximum ne dépasse pas k :

$$\text{MINDEG-AC} = \{\langle G, k\rangle : G \text{ est un graphe dans lequel il existe un arbre couvrant tel que chaque sommet a un maximum de } k \text{ voisins}\}$$

Montrez que MINDEG-AC est NP-complet.
Suggestion : considérez un k très petit.

Solution.

Je ne donne que le sketch d'une preuve. MINDEG-AC est dans NP car un arbre couvrant peut servir de certificat.

Pour la NP-difficulté, on peut réduire de HAMPATH. La réduction conserve le même graphe, mais pose $k = 2$. Il suffit ensuite d'observer qu'un arbre couvrant est un chemin Hamiltonien si et seulement si son degrée maximum est 2.

Exercice 11

Dans le problème 2-SAT, on reçoit une formule ϕ en forme CNF dans laquelle chaque clause a deux variables. On veut savoir si ϕ est satisfaisable. Montrez que 2-SAT est dans P.

Solution

J'ai écrit assez de solutions à ce point-ci. Je redirige vers : `http://philippe.gambette.free.fr/SCOL/Graphes/Gambette%20-%20Un%20graphe%20pour%20resoudre%202SAT.pdf`.

Exercice 12

Dans le problème MAX-2-SAT, on reçoits un ensemble de clauses CNF avec chacune deux variables, et on veut en satisfaire un maximum. En terme de langage, on a

$$\text{MAX-2-SAT} = \{\langle C, k\rangle : C \text{ est un ensemble de clauses CNF, et il existe une assigntion qui en satisfait au moins } k\}$$

Montrez que MAX-2SAT est NP-complet.
Suggestion : ce n'est pas trivial ! Une réduction à partir de MAX-2-XOR est possible.

Solution

Je vous laisse naviguer le web pour trouver une réduction. Si vous réduisez de MAX-2-XOR, traduisez chaque $x_i \oplus x_j$ en clause 2CNF, et faites votre analyse.