Université de Strasbourg **UFR de Mathématiques et Informatique**
Master 1 ILC et ISI **décembre 2013**

Complexité et Calculabilité

Contrôle Continu n°3

Responsable : Prof. Christian RONSE

Corrigé

(1) Réduction polynomiale.
(A) Soient L_1 et L_2 deux langages sur l'alphabet Σ, et soient $\overline{L_1} = \Sigma^* \setminus L_1$ et $\overline{L_2} = \Sigma^* \setminus L_2$ leurs complémentaires.

(i) S'il existe une réduction polynomiale de L_1 vers L_2, qu'y a-t-il entre $\overline{L_1}$ et $\overline{L_2}$?

Soit τ la réduction polynomiale de L_1 vers L_2, c.-à-d. τ est une fonction calculable en temps polynomial telle que pour tout $w \in \Sigma^*$ on a $w \in L_1 \Leftrightarrow \tau(w) \in L_2$. Par complémentation, on a

$$w \in \overline{L_1} \Leftrightarrow w \notin L_1 \Leftrightarrow \tau(w) \notin L_2 \Leftrightarrow \tau(w) \in \overline{L_2} \ ,$$

ce qui signifie que τ est une réduction, toujours polynomiale, de $\overline{L_1}$ vers $\overline{L_2}$.

(ii) S'il existe une réduction polynomiale de L_1 vers L_2, et L_2 est de complexité NP, que peut-on dire de la complexité de L_1 ?

Alors L_1 est NP. La preuve est la même que pour le cas P vu en cours. Soient τ la réduction polynomiale de L_1 vers L_2, M_1 une Machine de Turing (déterministe) calculant τ en temps polynomial, et M_2 une Machine de Turing non déterministe décidant L_2 (au sens non déterministe) en temps polynomial. Considérons la Machine de Turing M_1M_2 (l'état d'arrêt de M_1 débouche sur l'état initial de M_2) ; elle est non déterministe. Pour une entrée $w \in \Sigma^*$, M_1 lit w, écrit $\tau(w)$ sur la bande puis s'arrête, après un nombre de transitions $\leq \mathbf{p}(|w|)$, où $\mathbf{p}$ est un polynôme ; alors $|\tau(w)| \leq \mathbf{p}(|w|)$. Ensuite M_2 lit $\tau(w)$ sur la bande, le décide puis s'arrête, après un nombre de transitions $\leq \mathbf{q}(|\tau(w)|)$, où $\mathbf{q}$ est un polynôme ; en enlevant les termes à coeeficients négatifs de $\mathbf{q}$, on obtient un polynôme $\mathbf{q}' \geq \mathbf{q}$ tel que $\mathbf{q}'$ est croissant (p.ex., si $\mathbf{q}(X) = X^4 - 3X^3 + X^2 - X$, on obtient $\mathbf{q}'(X) = X^4 + X^2$) ; alors $\mathbf{q}(|\tau(w)|) \leq \mathbf{q}'(|\tau(w)|) \leq \mathbf{q}'(\mathbf{p}(|w|))$. Donc M_1M_2 s'arrête en un temps $\leq \mathbf{p}(|w|) + \mathbf{q}'(\mathbf{p}(|w|))$, qui est polynomial. Si $w \in L_1$, alors $\tau(w) \in L_2$ et il y a au moins une dérivation (suite de choix de transitions) de M_2 aboutissant à l'arrêt sur y. Si $w \notin L_1$, alors $\tau(w) \notin L_2$ et toute dérivation de M_2 aboutit à l'arrêt sur n. Donc M_1M_2 décide L_1 (au sens non déterministe) en temps polynomial.

(B) Soit L_0 un langage NP-complet sur Σ. Supposons qu'on puisse démontrer que son complémentaire $\overline{L_0}$ est de complexité NP.

Il s'agit bien d'une supposition, je n'affirme pas qu'on ait montré une telle chose pour un certain langage. Cet exercice relève donc de la science-fiction, il faut donner les conséquences d'une hypothèse qui n'est pas nécessairement valide.

(*iii*) *Etant donné un langage L de complexité NP, prouver qu'alors son complémentaire $\overline{L}$ l'est aussi.*

Comme L_0 est NP-complet et L est NP, il y a une réduction polynomiale de L vers L_0. Par (i), elle est une réduction polynomiale de $\overline{L}$ vers $\overline{L_0}$. Comme $\overline{L_0}$ est NP, $\overline{L}$ est NP par (ii).

(*iv*) *Etant donné un langage L NP-complet, prouver qu'alors son complémentaire $\overline{L}$ l'est aussi.*

1ère preuve: Par (iii), $\overline{L}$ est NP ; comme L est NP-complet, il y a une réduction polynomiale de $\overline{L}$ vers L. Par (i), elle est une réduction polynomiale de L vers $\overline{L}$; comme L est NP-complet, cela signifie que $\overline{L}$ est aussi NP-complet.
[Expliqué en cours : tout langage H de complexité NP se réduit polynomialement vers L qui se réduit polynomialement vers $\overline{L}$, donc par transitivité H se réduit polynomialement vers $\overline{L}$.]
2ème preuve: Soit H un langage NP ; par (iii), $\overline{H}$ est NP. Comme L est NP-complet, il y a une réduction polynomiale de $\overline{H}$ vers L. Par (i), elle est une réduction polynomiale de $\overline{\overline{H}} = H$ vers $\overline{L}$; comme c'est vrai pour tout langage H dans NP, $\overline{L}$ est NP-complet.

(2) Homomorphisme non effaçant.
Soit Σ un alphabet (fini), et soit $f : \Sigma \to \Sigma^+$ une application associant à tout caractère $c \in \Sigma$ un mot non-vide $f(c)$. On en déduit alors l'homomorphisme $g : \Sigma^* \to \Sigma^*$ appliquant f à chaque caractère d'un mot, c.-à-d.

$$g(\varepsilon) = \varepsilon \qquad et \qquad g(c_1 \cdots c_n) = f(c_1) \cdots f(c_n) \quad (n > 0) \ .$$

Comme $f(c) \neq \varepsilon$ pour tout $c \in \Sigma$, cet homomorphisme est non effaçant.
Donner un algorithme non-déterministe polynomial qui décide l'ensemble des $g(w)$, $w \in \Sigma^*$. Plus précisément, à partir d'un mot $m \in \Sigma^*$, il peut produire, s'il existe, un mot $w \in \Sigma^*$ tel que $g(w) = m$ (mais il peut aussi échouer), et il échouera toujours s'il n'existe pas.

On définit les variables : *Entree*, *Sortie* : chaînes de caractères ; *Succes*, *Echec* : booléens.
$Succes := FAUX$; $Echec := FAUX$; *lire*(*Entree*) ;
RÉPÉTER
SI *Entree* $== \varepsilon$ ALORS { *ecrire*(*Sortie*) ; *Succes* $:= VRAI$ }
SINON { choisir un caractère c ;
SI $f(c)$ est un préfixe de *Entree*
ALORS { effacer $f(c)$ en tête de *Entree* ; ajouter c en queue de *Sortie* }
SINON *Echec* $:= VRAI$ }
JUSQU'À *Succes* ou *Echec*.

Comme f est non effaçant, $|m| \leq |w|$, et la boucle s'arrête après au plus $|w|$ étapes.

Exemple :
$\Sigma = \{a, b, c\}$, $f(a) = ab$, $f(b) = aba$, $f(c) = aac$, $w = abaabaac$.
Entree $= abaabaac$, *Sortie* $= \varepsilon$;
choix : b, $f(b) = aba$, *Entree* $= abaac$, *Sortie* $= b$;
choix : a, $f(a) = ab$, *Entree* $= aac$, *Sortie* $= ba$;
choix : c, $f(c) = aac$, *Entree* $= \varepsilon$, *Sortie* $= bac$;
Succes, $m = bac$.
NB. Toute autre suite de choix que b, a, c donne finalement *Echec*, qui provoque l'arrêt.