Corrections TD 03 – Réductions

Exercice 4. *Réductions Turing many-one*

Écrire chacune des réductions (Turing many-one) suivantes, et indiquer ce que l'on peut en déduire quant à la récursivité de ces langages.

Remarque : dans la correction des réductions nous allons passer sous silence l'étape de vérification que les instances sont bien formées. Par exemple si un langage comporte des mots qui sont des codes machines de Turing, alors notre transformation doit commencer par vérifier que le mot à transformer est bien le code d'une machine Turing. Cette vérification est calculable (car L_{enc} est décidable) : si elle échoue alors notre transformation doit renvoyer un mot qui n'est pas dans le langage vers lequel on réduit (n'importe lequel), et si elle réussit (l'entrée est bien formée) alors nous procédons à la réduction à proprement parler.

1. Réduire $L_u = \{\langle M\rangle\#w \mid M \text{ accepte le mot } w\}$ à $B = \{\langle M\rangle \mid a \in L(M)\}$.
   Nous proposons la transformation des instances de L_u en des instances de B suivante, $f : \langle M\rangle\#w \mapsto \langle M'\rangle$ telle que M' est la machine de Turing qui :
   - **(a)** efface un symbole a (M' s'arrête si le mot en entrée ne commence pas par a),
   - **(b)** écrit w sur le ruban et revient au début du mot w (pour cela nous ajoutons $|w| + 1$ nouveaux états),
   - **(c)** simule M (pour cela nous recopions la fonction de transition de M à l'intérieur de M'),

   avec l'état final de M' qui est l'état final de sa copie de M.

   Cette transformation f est calculable, car pour construire $\langle M'\rangle = f(\langle M\rangle\#w)$ nous avons connaissance du code de la machine M et du mot w. Notre construction de M' ajoute simplement des états et des transitions en préambule de la machine M.

   Remarquon que pour savoir si $\langle M'\rangle \in B$, il suffit de s'intéresser à son comportement sur l'entrée a. Nous montrons que $\langle M\rangle\#w \in L_u \iff f(\langle M\rangle\#w) = \langle M'\rangle \in B$:
   — Si $\langle M\rangle\#w \in L_u$ alors par définition de L_u, la machine M accepte le mot w. Lorsque M' est lancée sur l'entrée a, elle efface a (le ruban ne contient alors que des blancs B), écrit w, puis simule M sur l'entrée w. Puisque dans ce cas M accepte le mot w et que l'état final de M' correspond à celui de M, la machine M' accepte. Donc M' accepte a, c'est-à-dire $a \in L(M')$, et nous concluons que $\langle M'\rangle \in B$.
   — Si $\langle M\rangle\#w \notin L_u$ alors par définition de L_u, la machine M n'accepte pas le mot w. Lorsque M' est lancée sur l'entrée a, elle efface a (le ruban ne contient alors que des blancs B), écrit w, puis simule M sur l'entrée w. Puisque dans ce cas M n'accepte pas le mot w et que l'état final de M' correspond à celui de M, la machine M' n'accepte pas (soit M ne s'arrête jamais et alors M' ne s'arrête jamais, soit M s'arrête dans un état non final et il en est de même pour M'). Donc M' n'accepte pas a, c'est-à-dire $a \notin L(M')$, et nous concluons que $\langle M'\rangle \notin B$.

   Puisque L_u n'est pas décidable et $L_u \leq_m^T B$, nous pouvons en déduire que B n'est pas décidable.

2. Réduire L à $aL = \{aw \mid w \in L\}$ pour tout langage L.
   Soit f la fonction qui au mot w associe le mot aw, c'est-à-dire $f : w \mapsto aw$. Cette transformation est bien calculable car elle consiste simplement à ajouter un symbole a au début du mot w. Par définition de aL nous avons l'équivalence $w \in L \iff f(w) = aw \in aL$.

3. Réduire aL à L pour tout langage $L \subsetneq \{a,b\}^*$.
Si un mot $w \in \{a,b\}^*$ appartient à $a\{a,b\}^*$, c'est-à-dire si w s'écrit sous la forme av, avec $v \in \{a,b\}^*$, on note w/a le suffixe v de w. Par exemple, $aabba/a = abba$. Soit $u_0 \notin L$ (on sait qu'un tel u_0 existe car $L \neq \{a,b\}^*$). On considère l'application $r : \{a,b\}^* \to \{a,b\}^*$ définie par :

$$r(w) = \begin{cases} w/a & \text{si } w \in a\{a,b\}^*; \\ u_0 & \text{sinon.} \end{cases}$$

Alors r est clairement calculable et on a :

(a) $\big(w \in aL\big) \Rightarrow \big(\exists u \in L : w = au\big) \Rightarrow \big(r(w) = u \text{ et } r(w) \in L\big)$;

(b) $\big(r(w) \in L\big) \Rightarrow \big(r(w) \neq u_0\big) \Rightarrow \big(\exists v : w = av \text{ et } v \in L\big) \Rightarrow \big(w \in aL\big)$.

Des questions 2 et 3 nous pouvons déduire que L est décidable (respectivement semi-décidable) si et seulement si aL est décidable (respectivement semi-décidable).

4. Réduire $L_{\bar{u}} = \{\langle M\rangle\#w \mid M \text{ n'accepte pas } w\}$
à $C = \{\langle M\rangle\#w \mid M \text{ n'accepte pas } w \text{ mais accepte } bbw\}$.
Nous définissons la transformation $f : \langle M\rangle\#w \mapsto \langle M'\rangle\#w$, des instances de $L_{\bar{u}}$ en des instances de C, avec M' la machine de Turing qui :

(a) si on la lance sur bbw alors M' accepte,

(b) sinon, M' retourne au début du mot d'entrée et simule M (en identifiant l'état final de M' à celui de M).

On peut bien calculer le code de M' à partir du code de M et de w, car nous avons connaissance de la machine M et du mot w quand nous construisons M'. À l'étape (a) il faut $|w|+3$ états dans M' pour vérifier si son entrée est bbw en lisant chacun des $|w|+2$ symboles du mot bbw puis le symbole blanc B (sans effacer les symboles, pour éventuellement passer à l'étape (b)).
Argumentons l'équivalence $\langle M\rangle\#w \in L_{\bar{u}} \iff f(\langle M\rangle\#w) = \langle M'\rangle \in C$.
— Si $\langle M\rangle\#w \in L_{\bar{u}}$ alors par définition de $L_{\bar{u}}$ la machine M rejette le mot w, donc :
  — si on lance M' sur w, elle arrivera à l'étape (b) et rejettera de la même façon que M,
  — si on lance M' sur bbw, à l'étape (a) la machine M' acceptera.
  Il s'ensuit que M' rejettera w mais acceptera bbw, c'est-à-dire $\langle M'\rangle \in C$.
— Si $\langle M\rangle\#w \notin L_{\bar{u}}$ alors par définition de $L_{\bar{u}}$ la machine M accepte le mot w, donc :
  — si on lance M' sur w, on arrivera à l'étape (b) et M' acceptera comme M,
  — si on lance M' sur bbw, à l'étape (a) la machine M' acceptera.
  Il s'ensuit que M' acceptera w et acceptera bbw, c'est-à-dire $\langle M'\rangle \notin C$.

Puisque $L_{\bar{u}}$ n'est pas semi-décidable et $L_{\bar{u}} \leq_m^T C$, nous en déduisons que C n'est pas semi-décidable.

5. Réduire $L_{stupide} = \{a\}$ à L_u.
Attention, ici le langage $L_{stupide}$ est décidable ! Soient $\langle M^+\rangle\#v^+ \in L_u$ et $\langle M^-\rangle\#v^- \notin L_u$ arbitraires, nous pouvons alors définir la transformation calculable

$$f(w) = \begin{cases} \langle M^+\rangle\#w^+ & \text{si } w = a, \\ \langle M^-\rangle\#w^- & \text{sinon.} \end{cases}$$

Il s'ensuit que $w \in L_{stupide} \iff f(w) \in L_u$. Nous ne pouvons en revanche faire aucune déduction à partir de cette réduction d'un langage décidable vers un langage non décidable.