Théorème de Rice

Référence : Introduction à la calculabilité. Pierre Wolper

2011-2012

On définit dans la suite deux langages :

Définition 1
On note (w_i) *et* (M_j) *des énumérations des mots et des machines de Turing, et on pose*
– $L_0 := \{w \mid w = w_i \text{ et } M_i \text{ n'accepte pas } w_i\}$.
– $LU := \{< M, w > \mid M \text{ accepte } w\}$
On note $\overline{L_0}$ *et* $\overline{LU}$ *les complémentaires.*

Théorème 2
Tout propriété non triviale des langages récursivement énumérables est indécidable.

Démonstration. On va montrer que les langages L_0, $\overline{L_0}$ et LU sont indécidables, puis, pour une propriété non triviale sur les langages récursivement énumérables P, on montrera que P est indécidable par une réduction à partir de LU.

Lemme 3
L_0 *est indécidable.*

Démonstration. Supposons L_0 décidable : il existe une machine de Turing qui l'accepte, soit M_k. On a alors :
– si M_k accepte w_k, alors $w_k \notin L_0$ par définition de L_0 $\rightarrow$ contradiction.
– si M_k n'accepte pas w_k, alors $w_k \in L_0$ $\rightarrow$ contradiction.

$\Diamond$

Lemme 4
$\overline{L_0}$ *est indécidable.*

Démonstration. Si $\overline{L_0}$ était décidable, alors L_0 aussi. $\Diamond$

Lemme 5
LU *est indécidable.*

Démonstration. On fait une réduction à partir de $\overline{L_0}$.

Supposons donc LU décidable. Considérons l'algorithme suivant, prenant en entrée un mot w :

– on détermine i tel que $w = w_i$;
– on détermine M_i ;
– on applique la procédure de décision pour LU à $< M_i, w_i >$: si le résultat est positif, on accepte w, sinon on le rejette.
Alors cet algorithme décide $\overline{L_0} \to$ contradiction. $\Diamond$

Soit maintenant P une propriété non triviale sur les langages récursivement énumérables.

On peut supposer que le langage vide ne vérifie pas P (sinon, on considère $\overline{P}$).

Comme P est non triviale, il existe une machine de Turing M_p qui accepte un langage vérifiant P.

Pour une instance $< M, w >$ de LU, on construit une machine M' qui a le comportement suivant :
– M' simule l'exécution de M sur w, sans tenir compte du mot d'entrée x ;
– si M accepte w, elle simule M_p sur x ;
– si M n'accepte pas w (rejet ou exécution infinie), M' n'accepte aucun mot.
On a alors : $\mathscr{L}(M')$ vérifie P si et seulement si $< M, w >\in LU$. $\square$