Université de Limoges
Faculté des Sciences et Techniques

Année universitaire 2015-2016

Master 1 Informatique et Mathématiques

Complexité et calculabilité - Feuille d'exercices n°3

Problèmes indécidables

On a vu dans le cours :

Théorème 1. *Soit A un alphabet non vide. Il existe une machine de Turing U (dite « machine universelle ») sur A telle que,*

- *pour toute machine de Turing M sur A, il existe un encodage de M dans A^*, noté $< M >$, et un encodage des couples (M, w) avec $w \in A^*$, noté $< M, w >$*
- *U accepte $< M, w >$ si et seulement si M accepte w, et rejette $< M, w >$ si et seulement si M rejette w.*

En particulier, U s'arrête sur $< M, w >$ si et seulement si M s'arrête sur w.

Exercice 1. On considère le langage $\mathcal{L}_{acc} = \{< M, w >| \ M \text{ accepte } w\}$.

1. Le langage $\mathcal{L}_{acc}$ est-il dans RE ?
2. On cherche à montrer que ce langage n'est pas dans R. Supposons qu'il existe une machine D qui décide le langage $\mathcal{L}_{acc}$. Il s'agit alors de montrer que cette hypothèse engendre une contradiction et qu'un tel D ne peut exister : $\mathcal{L}_{acc}$ ne sera donc pas décidable. Pour cela, on construit la machine C, qui pour toute entrée de la forme $< M >$ (M machine de Turing quelconque) fonctionne comme suit :
   (a) C exécute D sur $< M, < M >>$,
   (b) C renvoie le contraire de ce que D renvoie.

   Si l'on exécute C sur $< C >$, que se passe-t-il ? Conclure.

Exercice 2. Problème de l'arrêt. On a considéré dans le cours le langage :

$$\mathcal{L}_{stop} = \{< M, w >| \ M \text{ s'arrête sur l'entrée } w\}.$$

1. Supposons qu'il existe une machine de Turing D qui décide $\mathcal{L}_{stop}$. Construire une machine de Turing qui décide $\mathcal{L}_{acc}$.
2. En déduire que $\mathcal{L}_{stop}$ n'est pas décidable. Le langage $\mathcal{L}_{stop}$ est-il dans RE ?

Une telle preuve est appelée « preuve par réduction de $\mathcal{L}_{acc}$ à $\mathcal{L}_{stop}$ » : la décidabilité de $\mathcal{L}_{stop}$ entrainerait la décidabilité de $\mathcal{L}_{acc}$, ce qui n'est pas possible.

Exercice 3. Problème du langage vide. On considère le langage :

$$\mathcal{L}_{vide} = \{< M >| \mathcal{L}(M) = \emptyset\}.$$

1. Soit M une machine de Turing et w un mot. Montrer que l'on peut construire une machine M_w telle que $\mathcal{L}(M_w) = \{w\}$ si M accepte w et $\mathcal{L}(M_w) = \emptyset$ sinon.

2. Montrer par réduction de $\mathcal{L}_{acc}$ que $\mathcal{L}_{vide}$ est indécidable.

3. Le langage $\mathcal{L}_{vide}$ est-il dans RE ?

Exercice 4. Problème de l'égalité des langages. On considère le langage :

$$\mathcal{L}_{eg} = \{(< M_1 >, < M_2 >) \mid \mathcal{L}(M_1) = \mathcal{L}(M_2)\},$$

où M_1 et M_2 sont des machines de Turing. Montrer que $\mathcal{L}_{eg}$ est indécidable. On procèdera par réduction de $\mathcal{L}_{vide}$. Le langage $\mathcal{L}_{eg}$ est-il dans RE ?

Caractérisation de R et de RE

Exercice 5. Soit $A = \{a, b\}$ un alphabet à deux symboles[1]. On ordonne les mots de A^* d'abord par longueur, puis par ordre alphabétique pour départager les mots de même longueur. On note $\prec$ cette relation d'ordre totale sur A^*.

1. Construire une machine de Turing, nommée $SUCC$ qui étant donné un mot de A^* calcule son successeur pour $\prec$.

2. Soit $\mathcal{L}$ un langage sur A. Un énumérateur de $\mathcal{L}$ est une machine de Turing (ne s'arrêtant pas forcément) qui prend en entrée une (ou des) bande vide et qui écrit l'un après l'autre tous les mots (non vides) de $\mathcal{L}$, dans un ordre quelconque.

   (a) On suppose $\mathcal{L}$ infini. Montrer que $\mathcal{L}$ est dans R si et seulement si il existe un énumérateur de $\mathcal{L}$ capable d'écrire les mots de $\mathcal{L}$ dans l'ordre $\prec$. Quel est le problème si $\mathcal{L}$ n'est pas supposé infini ?

   (b) Montrer que $\mathcal{L}$ est dans RE si et seulement si il existe un énumérateur pour $\mathcal{L}$.
   Indication : on peut construire des machines de Turing dont le nombre de transitions effectuées lors d'un calcul est limité par une valeur entière spécifiée en entrée.

   (c) Donner un exemple de langage pour lequel il existe un énumérateur respectant l'ordre $\prec$. Donner un exemple de langage pour lequel il existe un énumérateur, mais pour lequel aucun énumérateur ne peut respecter l'ordre $\prec$. Donner un exemple de langage pour lequel il n'existe aucun énumérateur.

[1]Cet exercice pourrait se généraliser à un alphabet avec un nombre fini quelconque de symboles.