TD (In)décidabilité

9 novembre 2013

`Exercice 1` `Rice ou pas Rice ?`

On considère ces problèmes de décision :

entrée : machine de Turing M
sortie : oui si M a un nombre pair d'états

entrée : machine de Turing M
sortie : oui ssi $L(M)$ est le langage des images miroirs des mots de... $L(M)$

entrée : machine de Turing M
sortie : oui ssi M s'arrête pour au moins un mot d'entrée

entrée : machine de Turing M
sortie : oui ssi M s'arrête pour tous les mots d'entrée

entrée : machine de Turing M
sortie : oui ssi $L(M)$ est indécidable

entrée : deux machines de Turing M_1, M_2
sortie : oui ssi $L(M_1) \cap L(M_2) = \emptyset$

1. L'un de ces problèmes est décidable. Pour les autres, prouver qu'ils sont indécidables (pour cela, appliquer le théorème de Rice si cela est approprié).

2. Montrer que le problème suivant n'est ni dans RE ni dans co-RE :

entrée : deux machines de Turing M_1, M_2
sortie : oui ssi $L(M_1) = L(M_2)$

`Exercice 2` `Correspondance de Post`

Réf : [Sipser, p. 183-189]. On s'intéresse d'abord au problème de Post défini de la manière suivante :

entrée : un alphabet fini Σ, une famille finie de couples de mots sur Σ, c'est à dire $\left(\left(\begin{array}{c} h_i \\ b_i \end{array}\right)\right)_{i=1..n}$
sortie : oui ssi il existe $i_1, \ldots, i_p \in \{1, \ldots, n\}$ tels que $h_{i_1} \ldots h_{i_p} = b_{i_1} \ldots b_{i_p}$.

C'est une variante où l'alphabet est en entrée du problème. On s'intéresse ensuite au cas où l'alphabet n'est plus en entrée.

1. Est-ce que cette instance est positive ?

a	ab	bba
baa	aa	bb

2. Et cette instance ?

ab	b	aba	aa
$abab$	a	b	a

3. Est-ce que le problème est récursivement énumérable ?

Pour démontrer que le problème de correspondance de Post est indécidable, on peut effectuer une réduction à partir de l'arrêt et représenter les exécutions par des séquences du type

$$\#q_I aab\#bqab\#cq_f ab\#.$$

4. Prouver que le problème de correspondance de Post est indécidable.

A présent, on ne met plus l'alphabet en entrée du problème. Autrement dit, on considère un alphabet fini Σ et on considère le problème de Post suivant :

entrée : une famille finie de couples de mots sur Σ, c'est à dire $\left(\left(\begin{array}{c} h_i \\ b_i \end{array}\right)\right)_{i=1..n}$
sortie : oui ssi il existe $i_1, \ldots, i_p \in \{1, \ldots, n\}$ tels que $h_{i_1} \ldots h_{i_p} = b_{i_1} \ldots b_{i_p}$.

5. Si Σ n'a qu'une lettre, est-ce que ce problème est décidable ?

6. Et si Σ n'a que deux lettres ?

`Exercice 3` `Grammaire algébrique ambigüe`

Réf : [Sipser, exo. 5.19]

Soit Σ à deux lettres. On suppose connu le fait que le problème de Post c'est à dire est indécidable :

entrée : une famille de couples de mots sur Σ, c'est à dire $\left(\begin{pmatrix} h_i \\ b_i \end{pmatrix}\right)_{i=1..n}$
sortie : oui ssi il existe $i_1, \ldots, i_p \in \{1, \ldots, n\}$ tels que $h_{i_1} \ldots h_{i_p} = b_{i_1} \ldots b_{i_p}$.

1. Montrer que le problème suivant est indécidable :

entrée : une grammaire G
sortie : oui ssi G est ambigüe

(l'ambiguïté se situe sur le fait que l'on reconnaisse un mot avec les h_i ou avec les b_i)

`Exercice 4` `Pavage du plan`

On appelle type de tuiles un 4-uplet

$$t = \langle gauche(t), droite(t), haut(t), bas(t) \rangle \in \mathbb{N}^4.$$

On peut dessiner une tuile comme ceci :

1. Montrer que le problème suivant est indécidable :

entrée : un ensemble non vide fini T de type de tuiles et $t_0 \in T$
sortie : oui ssi on peut paver le quart de plan $\mathbb{N} \times \mathbb{N}$ en mettant t_0 en $(0, 0)$

2. Est-il récursivement énumérable ?

3. (optionnel) Que pensez-vous des deux problèmes suivants ?

entrée : un ensemble non vide T de type de tuiles et $t_0 \in T$, un entier n écrit en représentation *unaire*
sortie : oui ssi on peut paver le carré $\{0, \ldots, n-1\} \times \{0, \ldots, n-1\}$ en mettant t_0 en $(0, 0)$

entrée : un ensemble non vide T de type de tuiles et $t_0 \in T$, un entier n écrit en représentation binaire
sortie : oui ssi on peut paver le carré $\{0, \ldots, n-1\} \times \{0, \ldots, n-1\}$ en mettant t_0 en $(0, 0)$

Exercice 5 — Grammaire algébrique

1. Montrer que le problème suivant, dit de la vacuité, est décidable [Sipser, Th. 4.7] :

entrée : une grammaire algébrique G
sortie : oui ssi $L(G) = \emptyset$

2. Quelle est la complexité du problème de vacuité pour les grammaires algébriques ?

On s'intéresse maintenant au problème de l'universalité :

entrée : une grammaire algébrique G
sortie : oui ssi $L(G) = \Sigma^*$

On va démontrer que ce problème indécidable. Pour cela, nous allons réduire le problème de l'arrêt au problème de l'universalité.

3. Soit M une machine de Turing et w un mot. Donner un automate à pile non-déterministe qui reconnaît les chaînes de caractères qui ne sont pas des exécutions acceptantes de M sur w. On représente les exécutions de la manière suivante :

$$\#C_1\#C_2^R\#\ldots\#$$

où les C_i représentent des configurations (par exemple la configuration initiale pourrait être représentée par $q_I aab$) et C_i^R représente l'image miroir d'une configuration.

4. En déduire que le problème de l'universalité est indécidable.

`Exercice 6` `Castor affairé`

On considère des machines de Turing déterministe où l'alphabet du ruban ne contient qu'un symbole '1' (et le symbole blanc). Considérons l'exécution de M sur le mot vide. Si la machine s'arrête, alors $score(M) = 0$ sinon $score(M)$ est le nombre de 1 écrit sur le ruban à la fin de l'exécution.

Soit la fonction

$$\begin{array}{rccl} f: & \mathbb{N} & \to & \mathbb{N} \\ & n & \mapsto & max_{\text{machine } M \text{ à } n \text{ états}} score(M). \end{array}$$

Le but est de montrer que f n'est pas calculable. Pour le montrer, on montre que pour toute fonction c calculable, il existe n_0, tq pour tout $n \geq n_0$, $c(n) < f(n)$.

1. Montrer que f n'est pas calculable.

`Exercice 7` `caractérisation de` RE

Soit M est une machine de Turing. Soit $f_M(w)$ le mot écrit à la fin de l'exécution partant de $(s, 0, w)$. Si M boucle w, alors $f_M(w) = \perp$. f_M s'appelle la fonction calculée par M.

Le langage calculé par M est

$$L_c(M) = \{u \mid \text{il existe un } w \text{ tq } M \text{ s'arrête pour } w \text{ et } u = f_M(w)\}.$$

1. Montrer que un langage est RE ssi il est le langage calculé par une MT.

2. Montrer que un langage est RE ssi il est engendré par une grammaire (*avec* contexte).