Examen du cours de L3: Calculabilité

12 novembre 2015, durée: 3 heures

All documents are allowed. The electronic devices are forbidden. The results that have been proved in the exercise class cannot be used unless they are proved again.

1

Which of the following problems are decidable ? Justify.

1. **Data:** three Turing machine codes: M_1, M_2, M_3
   Question: $L(M_1) \cup L(M_2) = L(M_3)$
2. **Data:** A Turing machine M, a word w and non negative integer n
   Question: M accepts w after at most n computation steps.
3. **Data:** A Turing machine M
   Question: There exists a Turing machine M' such that $L(M') = \Sigma^* \setminus L(M)$
4. **Data:** `no data`
   Question: There exists a universal Turing machine M on the alphabet $\Sigma = \{0, 1, \$, B\}$ with only 8 states.
5. **Data:** A Turing machine M and a word w
   Question: The computation of M on w goes infinitily often through the initial state q_0.
6. **Data:** Two Turing machines M_1, M_2 that halt on every input.
   Question: For all x, $M_2(M_1(x)) = x$
7. **Data:** Two linearly bounded machines M_1, M_2
   Question: $L(M_1) = L(M_2)$
   A Turing machine is linearly bounded if every transition $\delta(q, B)$ is of the form $\delta(q, B) = (q', B, \leftarrow)$. (In other words, the machine computes within the space of the data).
8. **Data:** A recursive primitive function with one argument f, and a non negative integer n.
   Question: $\forall m.\ f(m) \leq \psi_n(m)$
   where ψ_n is the nth function of the Grzegorczyk hierarchy.

2

We consider a computation model $\mathcal{M}$, in which a machine is defined by a finite alphabet Σ, a distinguished symbol $f \in \Sigma$ and a finite set of rules δ of one of the following forms:

- $x \to axa$ where $a \in \Sigma$
- $uxv \to xb$ where $u, v \in \Sigma^*, b \in \Sigma$

A configuration of such a machine is a word $w \in \Sigma^*$. The machine may move from w to w' (written $w \to w'$) if there is a $z \in \Sigma^*$ and a rule $u \to v \in \delta$ such that $u\{x \mapsto z\} = w$ and $v\{x \mapsto z\} = w'$. ($\{x \mapsto z\}$ is the replacement of x with z).

The machine *accepts* w if there is a sequence of moves $w \to \cdots \to f$.

Show that the following problem"

Data: a machine $M \in \mathcal{M}$ and a word w

Question: M accepts w

is undecidable.

Solution

1

1. Ce problème est indécidable. Par le théorème de Rice, comme vu en cours, le probème de savoir si, étant donnée M, $L(M) = \emptyset$ est indécidable. On réduit ce problème en choisissant $M_1 = M$, M_2, M_3 des machines qui s'arrêtent immédiatement en reject. $L(M_1) \cup L(M_2) = L(M)$ et donc $L(M_1) \cup L(M_2) = L(M_3)$ ssi $L(M) = \emptyset$.

2. Ce problème est décidable: Il suffit de simuler n étapes de M sur w

3. Ce problème est indécidable d'après le théorème de Rice. En effet, il s'agit d'une propriété des langages récursivement énumérables. Elle est non triviale puisqu'il existe des langages récursivement énumérables et non co-récursivement énumérables (par exemple L_u).

4. Ce probème est décidable, par une machine qui renvoie toujours accept ou une machine qui renvoie toujours reject.

5. Ce problème est indécidable: on réduit le problème de l'arrêt. Étant donnés M_1, w_1 des données du problème de l'arrêt, on construit M à deux bandes en ajoutant un état q_0 (initial) et les transitions: si $\delta_1(q_1, a_1) = (q_2, a_2, d)$ alors dans $\delta(q_1, a_1)$ écrit (q_2, a_2, d) sur le deuxième ruban et passe dans létat q_0. depuis l'état q_0, si on lit \$ sur le deuxième ruban, alors on passe dans l'état initial de M_1 (sans bouger), sinon, on lit (q_2, a_2, d_2) sur le deuxième ruban, on l'efface et on passe dans létat q_2 après avoir écrit a_2 sur le premier ruban et effectué le déplacement d_2 sur le premier ruban.

   La machine M passe infiniment souvent par q_0 ssi M_1 ne s'arrête pas.

6. C'est indécidable. Par réduction, il suffit de montrer que le problème de savoir si une machine qui s'arrête toujours calcule l'identité est indécidable. Pour cela, on réduit le complémentaire du problème de l'arrêt: si M_1, w_1 est une donnée du problème de l'arrêt, alors M est la machine à 3 bandes, qui, sur la donnée x commence par recopier x sur la troisième bande, puis simule $|x|$ étapes de M_1 sur w_1. Si M_1 est sur le point de s'arrêter, alors M boucle. Sinon, si M_1 ne s'arrête pas en moins de $|x|$ étapes, alors M s'arrête et retourne x.

   M_1 ne s'arrête pas sur w_1 ssi M calcule l'identité.

7. C'est indécidable. Par réduction, on se ramène au problème de savoir si le langage d'un automate linéairement borné est vide. (Il suffit de choisir M_2 qui accepte un langage vide).

   On réduit alors le problème du non-arrêt: étant donné M, w, on construit l'automate linéairement borné M_1 qui, sur la donnée x, simule le calcul de M sur w, tant que cette simulation ne demande pas d'écrire sur un blanc. Autrement dit, M_1 ignore x, recopie w sur son ruban et simule M, jusqu'à ce que ou bien M lit un B, et dans ce cas M_1 s'arrête en rejet, ou bien M s'arrête, et dans ce cas M_1 accepte.

   Si M s'arrête sur w, alors soit n l'espace utilisé par M lors de ce calcul (i.e., la longueur maximale d'une configuration). M_1 accepte les mots de longueur au moins n et donc

$L(M_1) \neq \emptyset$. Si M ne s'arrête pas sur w, alors, pour tout x, ou bien M_1 rejette x (espace insuffisant) ou bien M_1 ne s'arrête pas sur x (cas où M boucle sur w).

Dans tous les cas, $L(M_1) = \emptyset$ ssi M ne s'arrête pas sur w.

8. C'est indécidable. On réduit le problème de l'arrêt. Comme dans la preuve du cours de l'équivalence entre MT et fonctions récursives, il existe une fonction primitive récursive f_M à un argument telle que $f_M(n) = m$ ssi m code le mot γ_n tel que $\gamma_0 \vdash_M^n \gamma_n$. De même, il existe une fonction primitive récursive g qui associe 1 aux codes des configurations finales de M et 0 aux autres entiers. Soit $g \circ f_M$ est la fonction nulle si et seulement si M ne s'arrête pas sur w. On considère enfin $f(n) = g(f_M(n)) \times (n+2)$. $f(m) \leq m+1$ pour tout m si et seulement si $g \circ f_M$ est la fonction nulle, ssi M ne s'arrête pas sur w. Il suffit alors de choisir la fonction ψ_0 de la hiérarchie pour obtenir une réduction du problème de non-arrêt au problème de l'énoncé.

2

On réduit le problème de correspondance de Post sur l'alphabet Σ_0. On considère $\Sigma = \Sigma_0 \uplus \{f, c\}$.

La machine M contient les règles $x \to axa$ pour tout $a \in \Sigma \cup \{c\}$ (Appelons R_1 ces règles)/ et les règles

$$\widetilde{u_i} x v_i f \to xf$$

pour les paires (u_i, v_i) du problème de correspondance de Post (appelons R_2 ces ègles)/

Et enfin la règle $cxc \to xf$, que nous mettons dans R)2

Si PCP a une solution $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$, alors

$$\epsilon \to^* c\widetilde{u_{i_k}} \cdots \widetilde{u_{i_1}} \cdot v_{i_1} \cdots v_{i_k} c$$

en utilisant seulement les règles de R_1.

$$c\widetilde{u_{i_k}} \cdots \widetilde{u_{i_1}} \cdot v_{i_1} \cdots v_{i_k} c \to \widetilde{u_{i_k}} \cdots \widetilde{u_{i_1}} \cdot v_{i_1} \cdots v_{i_k} f \to^* f$$

en utilisant les règles de R_2.

Réciproquement, si $\epsilon \to^* f$, la première règle utilisée est dans R_1 (puisque les autres ne peuvent pas s'appliquer à un mot vide). Lors de la séquence de réductions, il y a aussi au moins une règle de R_2 utilisée puisque le premier mot est de longueur 2 et le mot final est de longueur 1 et seules les règles de R_2 e font décroitre la longueur. Considérons la première occurrence d'application d'une règle de R_2.

$$\epsilon \to_{R_1}^* \widetilde{w} \cdot w \to_{R_2} \to^* f$$

w ne peut pas se terminer par f car dans ce cas $\widetilde{w}$ commence par f et aucune règle de R_2 ne s'applique à un mot commençant par f. Donc $w = w_0 \cdot c$.

$$\epsilon \to_{R_1}^* \widetilde{w} \cdot w \to_{R_2} \widetilde{w_0} \cdot w_0 \cdot f \to^* f$$

On remarque de plus que le nombre de f dans une configuration ne peut qu'augmenter par les transitions de la machine. De plus, si $w \to_{R_1} awa \to_{R_2} w'$, alors $a = c$ et $w' = wf$.

Donc, chaque fois qu'une règle de R_2 est appliquée après une règle de R_1 le nombre de f croit strictement.

Il en résulte que

$$\epsilon \to^*_{R_1} \widetilde{w \cdot c} \cdot w \cdot c \to_{R_2} \widetilde{w} \cdot w \cdot f \to^*_{R_2} f$$

Et donc $w = u_{i_1} \cdots u_{i_k} = v_{i_1} \cdot v_{i_k}$ (par récurrence sur le nombre k de règles de R_2 utilisées dans la réduction).