Université de Bordeaux — UF d'informatique
Master-1 d'Informatique — 2015/2016
Modèles de Calcul (J1N7W11) — Semestre 1

Modèles de Calcul : feuille 4

Problème de correspondance de Post.
Modèles alterntatifs aux machines de Turing.

I. Problème de correspondance de Post

Une instance I du *problème de correspondance de Post* (ou PCP en raccourci) est un ensemble fini de paires de mots

$$I = \{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$$

où $u_i, v_i \in A^+$, $i = 1, \ldots, n$ (A étant un alphabet). Un mot $w \in A^+$ est une *solution* du problème s'il existe une suites d'indices $i_1, i_2, \ldots, i_N$ telle que

$$w = u_{i_1} u_{i_2} \ldots u_{i_N} = v_{i_1} v_{i_2} \ldots v_{i_N} .$$

Pour une instance I, on dénote $S(I)$ l'ensemble de ses solution. La question principale que l'on se pose est la suivante : *existe-t-il une solution* (c'est-à-dire, l'ensemble $S(I)$ est-il vide) ?

On verra (voir l'Exercice 3.4) qu'en général il est impossible de donner une réponse à cette question : le problème de correspondance de Post est indécidable.

Exercice 4.1

1. Trouver une solution pour l'instance suivante : $I = \{(bbb, bb), (abb, babbb)\}$.
2. Montrer que pour l'instance $I = \{(ba, bab), (abb, bb), (bab, abb)\}$ il n'y a pas de solutions.
3. Montrer que si l'alphabet A consiste d'une seule lettre alors PCP est décidable.
   Indication : Considérer deux cas de figure : (A) pour tout $i = 1, 2, \ldots, n$ on a $|u_i| < |v_i|$; (B) il existe des indices i et j tels que $|u_i| < |v_i|$ et $|u_j| > |v_j|$.

Exercice 4.2

Réduire le PCP aux problèmes suivants de la théorie des langages algébriques :

1. Une grammaire algébrique donnée est-elle ambiguë ?
2. Un langage algébrique donné contient-il un palindrome ?
3. Un langage algébrique donné contient-il un carré (un mot de la forme uu, $u \in A^+$) ?
4. Pour deux langages algébriques donnés, leur intersection est-elle vide ?

Ainsi, si on admet l'indécidabilité du PCP, cela entraîne que les quatre problèmes ci-dessus sont indécidables.

Exercice 4.3

Dans le PCP *modifié*, on démande que la construction d'une solution commence toujours par une paire de mots fixée, par exemple par (u_1, v_1). Réduire le PCP modifié au PCP général.

Indication. On ajoute à l'alphabet un symbole supplémentaire $*$. Pour un mot $w = x_1 x_2 \dots x_m$, on dénote $\overline{w} = x_1 * x_2 * \dots * x_m$. Alors, pour une instance $I = \{(u_1, v_1), (u_2, v_2), \dots, (u_n, v_n)\}$ du PCP modifié on fabrique l'instance suivante du PCP habituel :

$$I^* = \{(*\overline{u}_1, *\overline{v}_1);\ (*\overline{u}_i, \overline{v}_i*) \text{ pour tout } i = 1, 2, \dots, n;\ (*\#, \#)\}\,.$$

(Continuer et finir la solution de l'exercice.)

Exercice 4.4
Réduire le problème de l'arrêt d'une machine de Turing aux problème de correspondance de Post modifié.

Les exercices 3.3 et 3.4 donnent une série de réductions

$$\text{Problème de l'arrêt pour MT} \quad \leq \quad \text{PCP modifié} \quad \leq \quad \text{PCP}\,,$$

ce qui entraîne l'indécidabilité du PCP.

Exercice 4.5
Le professeur Cosinus a une preuve de décidabilité du problème de correspondance de Post :

> Soit I une instance du PCP. Considérons l'ensemble $M(I)$ des solutions de longueur minimale de I. Comme $M(I)$ est fini (éventuellement vide), il est rationnel. Soit $\mathcal{A}(I)$ un automate fini reconnaissant $M(I)$. On teste si le langage reconnu est vide ou non. S'il est vide alors I n'a pas de solutions, sinon I a une solution.

Que pensez-vous des affirmations de Cosinus ?

II. Modèles alterntatifs aux machines de Turing

Exercice 4.6
Montrer que toute machine de Turing peut être simulée par une *machine à deux piles*.

Machine à compteurs. Une *machine à k compteurs* est une machine qui utilise k registres $r_1, r_2, \dots, r_k$ qui contiennent chacun un entier. Un programme pour une telle machine est une série d'instructions numérotées de la forme :

1. n : inc (r_i) ; goto m
2. n : dec (r_i) ; goto m
3. n : if $r_i = 0$ then goto m_1 else goto m_2
4. stop

Ici n, m, m_1 et m_2 dénotent des numéros d'instruction. Aussi, on utilise les conventions suivantes :

1. On se donne la liberté d'appeler les registes x, y, etc. au lieu de r_1, r_2, etc.
2. Si dec (x) est exécutée alors que $x = 0$, la valeur de x reste zéro.
3. On se permet de ne pas écrire explicitement, dans l'instruction numéro n, "goto $n+1$" (c'est-à-dire, aller à la ligne suivante).

Exercice 4.7
Écrire les programmes pour la machine à compteurs qui font les opérations suivantes :

1. if $x > 0$ then goto m_1 else goto m_2 ;
2. affectation $x = 0$;
3. affectation $y = x$ (attention : avant cette commande, le contenu de y n'est pas forcément nul ; après l'exécution de la commande, le contenu de x doit être le même qu'avant l'exécution) ;
4. $y = 2x$;
5. $z = \mathrm{div}\,(x, y)$ (le quotient entier de la division de x par y) ;
6. $t = \mathrm{rem}\,(x, y)$ (le reste de la division entière de x par y).

Exercice 4.8
Écrire un programme qui calcule la fonction

$$x \mapsto (x', t) \quad \text{où} \quad x' = \mathrm{div}\,(x, 2), \quad t = \mathrm{rem}\,(x, 2)$$

en n'utilisant que les deux registres x et t.

Exercice 4.9

1. Montrer qu'une machine à deux compteurs peut simuler une machine à une pile.
2. En déduire qu'une machine à quatre compteurs peut simuler une machine à deux piles.
3. Montrer qu'une machine à k compteurs peut être simulée par une machine à deux compteurs.
   Indication : Coder le contenu de k registres $r_1, r_2, \ldots, r_k$ par un entier $x = 2^{r_1} 3^{r_2} \ldots p_k^{r_k}$.

Systèmes de réécriture. Étant donné un alphabet A, on appelle *système de réécriture* sur A un ensemble fini de règles de type $u \to v$ où $u \in A^+$ et $v \in A^*$. On interprète ces règles comme une autorisation de réécrire un mot $\alpha u \beta$ en mot $\alpha v \beta$ où $\alpha, \beta \in A^*$; cette opération est notée $\alpha u \beta \to \alpha v \beta$. Quand pour un même mot plusieurs règles sont applicables on en choisit une de manière arbitraire. Si un mot w' est obtenu à partir d'un mot w par une application successive de plusieurs réécritures on dénote cela $w \xrightarrow{*} w'$.

Exercice 4.10

1. Simuler une machine de Turing à l'aide d'un système de réécriture.
2. Montrer que les deux problèmes suivants sont indécidables :

   Instance : un système de réécriture et deux mots $w, w' \in A^*$;
   Question 1 : en appliquant les règles du système, peut-on réécrire w en w' (c'est-à-dire, obtenir $w \xrightarrow{*} w'$) ?
   Question 2 : peut-on réécrire $w \xrightarrow{*} \varepsilon$?
3. Montrer que le problème suivant est indécidable :

   Instance : un système de réécriture, un mot $w \in A^*$, et un langage rationnel $L \subseteq A^*$;
   Question : peut-on réécrire w en un mot $w' \in L$?

Exercice 4.11

1. Simuler une machine à deux piles par une machine de Turing non-déterministe.
2. Simuler un système de réécriture à l'aide d'une machine de Turing non-déterministe à deux bandes.

Donner une description informelle (mais exacte) des machines en question.