Modèles de calcul (HAI402I)

Université de Montpellier

Correction de l'exercice 2 du TD 4

Exercice 1 Ensembles Rationnels vs Récursifs Primitifs

Étant donné l'alphabet $\Sigma = \{0, 1\}$, les mots de Σ^* n'ont pas de bijection complètement triviale vers $\mathbb{N}$. On ne peut pas mettre à la fois les mots 010 et 10 en bijection avec l'entier 2. Pour représenter un mot w de Σ^* on utilisera donc deux entiers, sa longueur $|w|$ et l'entier qu'il represente en binaire $entier(w)$. Inversement, on notera $bin(x, b)$ le mot binaire à b bits tel que $x = entier(bin(x, b))$. Notons que le nombre de bits doit vérifier $b \geq \lceil \log_2(x + 1) \rceil$.

1. Quels sont les entiers $entier(110)$, $entier(001)$ et $entier(100)$? Et quels sont les mots $bin(5, 3)$, $bin(5, 6)$ et $bin(0, 2)$?

   ▶ $entier(110) = 6$, $entier(001) = 1$ et $entier(100) = 4$.
   $bin(5, 3) = 101$, $bin(5, 6) = 000101$ et $bin(0, 2) = 00$.

   On peut maintenant définir les langages récursifs primitifs. Un langage $\mathcal{L} \subseteq \Sigma^*$ est *récursif primitif* si il existe un programme de Rosza $\boxed{\mathcal{L}}$ tel que $\boxed{\mathcal{L}}(x, b) = \begin{cases} 1 & \text{si } bin(x, b) \in \mathcal{L} \\ 0 & \text{sinon} \end{cases}$.

   Le but de cet exercice est de démontrer que **tout langage rationnel est également récursif primitif**. Pour ce faire on va montrer que pour toute expression rationnelle E on peut fournir un programme, noté $\boxed{E}$, calculant le prédicat suivant $\boxed{E}(x, b) = \begin{cases} 1 & \text{si } bin(x, b) \in \mathcal{L}(E) \\ 0 & \text{sinon} \end{cases}$.

   ▶ Une autre façon de coder les mots de de Σ^* consisterait à rajouter un 1 devant, et à considérer l'entier ainsi représenté en binaire. On pourrait faire cet exercice en considérant ce codage !

2. Pour toute constante $c \in \mathbb{N}$, on note $\boxed{=c}$ le jeton de Rosza tel que $\boxed{=c}(x) = \begin{cases} 1 & \text{si } x = c \\ 0 & \text{sinon} \end{cases}$.

   Étant donné une expression rationnelle E qui est un ensemble fini de mots $E \subsetneq \{0, 1\}^*$, décrire le programme $\boxed{E}$, c'est à dire le prédicat tel que $\boxed{E}(x, b) = \begin{cases} 1 & \text{si } bin(x, b) \in E \\ 0 & \text{sinon} \end{cases}$.

   ▶ Si $E = \{\}$ alors $\boxed{E} = \boxed{\blacktriangleright}\boxed{\blacktriangleright}\boxed{0}$.
   Sinon on considère un mot $u \in E$, et on définit $c = entier(u)$, $b = |u|$, et $E' = E \setminus \{u\}$. Dans ce cas, on suppose que l'on dispose du programme $\boxed{E'}$, et on pose :
   $\boxed{E} = \boxed{\circledcirc}\boxed{|}\boxed{E'}\ \boxed{\circledcirc}\boxed{\&}\ \boxed{\blacktriangleright}\boxed{=c}\ \boxed{\blacktriangleleft}\boxed{=b}$.

3. Étant donnée une expression rationnelle de la forme $E = E_1 + E_2$, écrire le programme $\boxed{E}$ en se basant sur $\boxed{E_1}$ et $\boxed{E_2}$.

   ▶ $\boxed{E} = \boxed{\circledcirc}\boxed{|}\boxed{E_1}\boxed{E_2}$.

4. Écrire un programme, noté $\boxed{\mathbf{Su}}$, qui pour un triplet b', x, b calcul l'entier codé sur les b' derniers bits de $bin(x, b)$.

   ▶ L'entier renvoyé vaut $x \bmod 2^{b'}$. On utilise les programmes $\boxed{2^x}$ et prédécesseur $\boxed{\mathbf{P}}$ de la fiche de TD 2. On utilise aussi le programme $\boxed{\mathrm{mod}}$ de la fiche de TD 3, tel que $\boxed{\mathrm{mod}} : x, y \longrightarrow x \bmod (y + 1)$.
   $\boxed{\mathbf{Su}} = \boxed{\circledcirc}\boxed{\mathrm{mod}}\ \boxed{\blacktriangleleft}\boxed{\blacktriangleright}\boxed{\mathrm{I}}\boxed{\blacktriangleright}\boxed{\blacktriangleright}\boxed{\circledcirc}\boxed{\mathbf{P}}\boxed{2^x}$.

5. Écrire un programme, noté [**Pr**], qui pour un triplet b', x, b calcul l'entier codé sur les $b-b'$ premiers bits de $bin(x, b)$.

▶ On utilise le programme [$/2$] de la fiche de TD 3, tel que [$/2$] : $x \longrightarrow \lfloor x/2 \rfloor$.
[**Pr**] = [R][▶][I][◀][▶][▶][$/2$].

6. Étant donnée une expression rationnelle de la forme $E = E_1.E_2$, écrire le programme [E] en se basant sur [E_1] et [E_2].

▶ On cherche à savoir si il existe un découpage de $bin(x, b)$ en 2 qui fonctionne bien. Pour cela on va définir un programme calculant $b', x, b \longrightarrow \begin{cases} 1 & \text{si } \exists i \leq b' \text{ tel que couper au } (b-i)\text{ème bit fonctionne} \\ 0 & \text{sinon} \end{cases}$

Et ensuite on lui fournit les bon arguments. Pour revenir à ce programme qui commence par [R] la condition d'arret ($f(x, b)$) c'est $bin(x, b) \in E_1$ ET $\epsilon \in E_2$. Pour g c'est "ça marche pour $i \leq b' - 1$" OU "ça marche pour $i = b'$".

[**E**] = [ⓞ] [R] [ⓞ][&][E_1][◀][◀][ⓞ][E_2][0][0]
[ⓞ][|][◀][▶][▶][I] [ⓞ][&]
[ⓞ][E_1][ⓞ][**Pr**][▶][▶][▶][I][◀][◀][▶][I][◀][◀][◀][I][ⓞ][−][◀][◀][◀][I][▶][▶][▶][I]
[ⓞ][E_2][ⓞ][**Su**][▶][▶][▶][I][◀][◀][▶][I][◀][◀][◀][I][▶][▶][▶][I]
[◀][I] [▶][I] [◀][I]

7. Étant donnée une expression rationnelle de la forme $E = E_1^*$, écrire le programme [E] en se basant sur [E_1^*].

▶ Cette fonction est complexe et permet notamment de constater qu'il n'est pas possible de faire certaines récursions...

Soit le programme [f_0], tel que [f_0]$(b', x, b) = \begin{cases} 1 & \text{si le suffixe de } bin(x, b) \text{ de lg } b' \text{ est dans } \mathcal{L}(E_1) \\ 0 & \text{sinon} \end{cases}$

[f_0] = [ⓞ][E_1][**Su**][▶][▶][I]

Soit le programme [bit], tel que [bit](i, y) renvoie le $(i+1)$ème bit de $bin(y, \infty)$ en partant de la droite :
[bit] = [ⓞ][mod] [R] [I] [◀][▶][$/2$] [▶][▶][ⓞ][S][0]

Le programme [h_1](b', m, x, b) calcule si il existe un $1 \leq i \leq b'$, tel que le $i$ème bit de $bin(m, b)$ en partant de la droite est 1, et tel que le suffixe de $bin(x, b)$ de longueur i appartient à $\mathcal{L}(E_1)$:
[h_1] = [R][f_1][g_1]
$h_1(0, m, x, b) = f_1(m, x, b) = 0$ donc [f_1] = [◀][◀][◀][0]
$h_1(b'+1, m, x, b) = g_1(b', h_1(b', x, b), m, x, b)$ donc
[g_1]= [ⓞ][|][◀][▶][▶][▶][I][ⓞ][&] [bit] [▶][▶][▶][▶][I] [▶][▶][◀][◀][I] [◀][◀][◀][◀][I]
[ⓞ][f_0] [▶][▶][▶][▶][S] [▶][◀][◀][◀][I] [◀][◀][◀][◀][I]

Enfin, pour décider si $bin(x, b) \in \mathcal{L}(E_1^*)$, on va calculer l'entier m tel que le mot $bin(m, b)$ indique pour quelles longueurs le préfixes de $bin(x, b)$ est un mot de $\mathcal{L}(E_1^*)$. Si le $i$ème bit est un 1 alors le préfixe de longueur $i-1$ appartient à $\mathcal{L}(E_1^*)$. Par exemple, comme $\epsilon \in \mathcal{L}(E_1^*)$, le 1er bit de m est toujours un 1. Puis on retourne le $(b+1)$ème bit de $bin(m, b+1)$. Les paramètres de cette fonction sont x, b :

[ⓞ][mod] [ⓞ][R][f_2][g_2] [◀][I] [▶][I] [◀][I] [▶][▶][ⓞ][S][0]

On initialise le premier bit de m (c'est aussi le $(b+1)$ ème en partant de la droite).
$h_2(0,x,b) = f_2(x,b) = 1$

f_2 = ◀ ◀ @ S 0

On poursuit la construction de m en prenant ses $b'+1$ premiers bits et en les décalant d'un cran à gauche (on multiplie m par 2) et en ajoutant 0 ou 1 suivant que le préfixe de longueur $b'+1$ soit dans $\mathcal{L}(E_1^*)$ ou pas.
$h_2(b'+1,x,b) = g_2(b', h_2(b',x,b), x, b) = 2 \times h_2(b',x,b) + h_1(b'+1, h_2(b',x,b), \mathbf{Pr}(b-(b'+1),x,b), b'+1)$

g_2 = @ + @ × ◀ ◀ ◀ ◀ @ S @ S 0 ◀ ▶ ▶ I
@ h_1 ▶ ▶ ▶ S
◀ ▶ ▶ I
@ **Pr** @ − ◀ ◀ ◀ I ▶ ▶ ▶ S ◀ ◀ ▶ I ◀ ◀ ◀ I
▶ ▶ ▶ S

8. Que peut-on en déduire ?

► Tout langage rationnel est récursif primitif.

9. Montrer que le langage des mots équilibrés est récursif primitif. Qu'en déduit-on ?

► On définit f tel que $f(x,b) = 1$ si $bin(x,b)$ contient exactement $b/2$ 1, sinon $f(x,b) = 0$.

f = @ = ◀ I @ × ◀ ◀ @ S @ S 0
@ R ◀ 0 @ + ◀ ▶ I bit ▶ ▶ I ◀ ◀ I ◀ I ▶ I

On a vu à l'exercice 2 que ce langage n'est pas rationnel. L'ensemble des langages rationnels est donc **strictement** inclu dans celui des langages récursifs primitifs.