Feuille 6 - Langages Rationnels

Informatique Théorique 2 - Unité J1INPW11
Licence 3 - Université de Bordeaux

Exercice 1

Un mot de u de $\{a, b\}^*$ est dit *bien équilibré* si, pour tout préfixe v de u, la différence $|v|_a - |v|_b$ vaut 0, 1 ou -1. Par exemple, *abbab* est bien équilibré, mais *abaab* ne l'est pas (le préfixe *abaa* comporte trop de a).

L'ensemble des mots bien équilibrés est-il un langage reconnaissable ?

Exercice 2

On considère ici les expressions arithmétiques qu'on peut construire à l'aide des lettres x, y, z et des symboles arithmétiques $+$, $-$, $*$, $/$.

1. Donner un automate reconnaissant l'ensemble des expressions arithmétiques correctes sans parenthèse. Par exemple, $x + y * z - x$ est une expression correcte sans parenthèse.
2. On désigne par A_1 l'ensemble des expressions arithmétiques correctes contenant au plus un niveau de parenthèses. Par exemple, $x+y*z-x$, $x*(y-z)*y+x$ et $x*(y-z)*(y+x)$ sont des expressions arithmétiques correctes avec au plus un niveau de parenthèses.
   Donner un automate reconnaissant le langage A_1.
3. On désigne par A_2 l'ensemble des expressions arithmétiques correctes contenant au plus deux niveaux de parenthèses. Par exemple, les expressions $x+y*z-x$, $x*(y-z)*y+x$ et $x * ((y - z) * y + x)$ sont des mots de A_2.
   Donner un automate reconnaissant A_2.
4. De même pour A_n, n quelconque.
5. Peut-on en déduire que le langage $\cup_{n=0}^{\infty} A_n = A_0 \cup A_1 \cdots \cup A_n \ldots$ est rationnel ?

Exercice 3

Que pensez-vous du raisonnement ci-dessous ?

1. Le langage $L_1 = a^*b^*$ est rationnel.
2. Le langage $L_2 = \{a^n b^p | n + p \text{ est pair}\}$ est inclus dans L_1.
3. Or tout sous-ensemble d'un langage rationnel est rationnel.
4. L_2 est donc rationnel.

Exercice 4

Parmi les langages suivants sur l'alphabet $\{a, b, c\}$, lesquels sont rationnels (justifiez dans chaque cas) ? Si le langage est algébrique donnez également une grammaire (on ne demande pas pour l'instant de donner une preuve si le langage n'est pas algébrique)

1. Les mots comportant exactement 42 occurrences de a et 42 occurrences de b.
2. Les mots comportant au moins autant de a que de b.
3. Les mots de la forme $a^{p+q}b^pc^q$ pour p, $q \in \mathbb{N}$.
4. Les mots de la forme $a^{2n}b^{2n}$ pour $n \in \mathbb{N}$.
5. Les mots comportant deux fois autant de a que de b.
6. Les mots dont la longueur est une puissance de 2.
7. Les mots dont la longueur est un multiple de 2^{64}.
8. Les mots carrés, c'est-à-dire les mots pouvant s'écrire sous la forme uu, où u est un mot de $\{a, b, c\}^*$.

Exercice 5

Le but de l'exercice est d'utiliser les automates pour tester la satisfiabilité de certaines formules d'arithmétique.

Considérons par exemple la formule à deux variables : $x+x < y$. Le but est de construire un automate qui accepte les couples d'entiers (x, y) (codés sous la forme d'un mot) qui satisfont la formule.

On commence par expliquer comment coder des couples. On note $A = \{0, 1\}$, il est bien connu qu'on peut coder un entier sous la forme d'un mot de A^* par son écriture en binaire. Prenons maintenant l'alphabet suivant :

$$A_2 = A^2 = \left\{ \binom{0}{0}, \binom{0}{1}, \binom{1}{0}, \binom{1}{1} \right\}$$

Un mot de A_2^* code un couple d'entiers, par exemple, le couple $(101, 11100)$ est codé par le mot suivant :

$$\begin{array}{ll} \binom{0}{1}\binom{0}{1}\binom{1}{1}\binom{0}{0}\binom{1}{0} & \leftarrow \text{ Composante 1} \\ & \leftarrow \text{ Composante 2} \end{array}$$

1. Construire un automate qui accepte les couples (x, y) tels que $x = y$.
2. Construire un automate qui accepte les couples (x, y) tels que $x < y$.
3. Construire un automate qui accepte les couples (x, y) tels que $x + x = y$.
4. Construire un automate qui accepte les couples (x, y) tels que $x + x < y$. On pourra décomposer la formule en la conjonction de formules construites précédemment en augmentant le nombre de variables (il faudra donc généraliser le codage à un plus grand nombre de variables).

Exercice 6: Exercice plus difficile et recommandé en travail personnel

Dans cet exercice, on cherche à généraliser la méthode de l'exercice précédent à une classe complète : l'arithmétique de Presburger.

Une formule de l'arithmétique de Presburger est construite par combinaisons booléennes et quantifications de formules atomiques. Une formule atomique est une égalitée ($=$) ou une inégalitée ($<$) utilisant des variables ($x, y, z, \dots$) des constantes ($0, 1, 2, 3$) ainsi que l'addition ($+$). Par exemple la formule suivante est une formule atomique :

$$x + y + 2 < z$$

Une formule de Presburger s'obtient en appliquant combinaisons booléennes ($\wedge, \vee, \neg$) et quantifications ($\forall, \exists$). On a par exemple la formule suivante :

$$\varphi(x) = \exists z(x = z + z) \wedge \neg(x > 4))$$

La sémantique se définit de manière naturelle. Les *solutions* d'une formule $\Psi(x_1, \dots, x_n)$ à n variables libres $x_1, \dots, x_n$ sont les vecteurs d'entiers $(i_1, \dots, i_n)$, tels que la formule obtenue en substituant $(x_1, \dots, x_n)$ par $(i_1, \dots, i_n)$ est valide. Une formule est dite satisfiable si et seulement il existe une solution.

Par exemple la formule $\varphi(x)$ ci-dessus a pour solutions 0, 2 et 4. Inversement, la formule $\forall y\ y < x$ n'est pas satisfiable (aucun entier n'est supérieur à tous les autres).

Dans l'exercice, on cherche à montrer que pour toute formule de Presburger, il est possible de contruire un automate qui accepte le langage de ses solutions (codées sous la forme de l'exercice précédent).

1. Soit $n \in \mathbb{N}$. Montrer que pour tous les entiers $i, j, k \leq n$ on peut construire un automate qui accepte les vecteurs $(x_1, \dots, x_n)$ tels que :
   (a) $x_i = c$ (où c est une constante).
   (b) $x_i = x_j$.
   (c) $x_i < x_j$.
   (d) $x_i = x_j + x_k$.
2. Montrer que pour toute formule atomique de Presburger, on peut construire une formule équivalente de la forme :

   $$\exists x_1 \exists y_1 \exists z_1 \cdots \exists x_n \exists y_n \exists z_n\ \varphi_1(x_1, y_1, z_1) \wedge \cdots \wedge \varphi_n(x_n, y_n, z_n)$$

   Où les formules φ_i sont toutes de l'une des quatres formes suivantes : $x_i = c$ (où c est une constante), $x_i = y_i$, $x_i < y_i$ ou $x_i = y_i + z_i$. (Autrement dit on peut supposer que toutes les formules atomiques sont de l'une de ces formes).
3. Montrer que les langages réguliers sont clos par projection. C'est à dire que si L est un langage régulier sur un alphabet $A \times B$ alors le langage $L' = \{w = a_1 \cdots a_m \in A^* \mid \exists b_1, \dots, b_m\ (a_1, b_1) \cdots (a_m, b_m) \in L\}$ est régulier.
4. Conclure qu'on peut construire un automate pour tout formule de Presburger.