Langages Formels

Licence Informatique – ENS Cachan

Examen du 27 mai 2013

durée 3 heures

Document autorisé : polycopié du cours.
Toutes les réponses devront être correctement justifiées.
Le chiffre en regard d'une question est une indication sur sa difficulté ou sa longueur.

1 Logique et automates d'arbres

On fixe $A_p = \{d_1, \ldots, d_p\}$ un alphabet de directions et Σ un alphabet. On considère la logique du premier ordre $\mathrm{FO}(\Sigma, <, \downarrow_1, \ldots, \downarrow_p)$ sur les arbres ($x < y$ signifie que le nœud x est un ancêtre du nœud y).

[2] **a)** Donner des formules du premier ordre ayant exactement une variable libre pour les propriétés suivantes :

$\psi_1(x)$ tous les fils de x sont étiquetés par la lettre a,

$\psi_2(x)$ il existe une branche issue de x dont la suite des étiquettes (à partir de x) est dans $a^*b\Sigma^*$,

$\psi_3(x)$ toutes les branches issues de x comportent au moins un nœud étiqueté b,

$\psi_4(x)$ il existe une branche issue de x telle que tous les nœuds de cette branche qui sont étiquetés a satisfont ψ_3.

On considère l'alphabet $\Sigma' = \Sigma \times \{0,1\}$. Pour un arbre $t \in T_p(\Sigma')$ on notera $t_1 \in T_p(\Sigma)$ sa première projection et $t_2 \in T_p(\{0,1\})$ sa deuxième projection. On remarque que $\mathrm{dom}(t) = \mathrm{dom}(t_1) = \mathrm{dom}(t_2)$.

Étant donné une formule $\varphi(x) \in \mathrm{FO}(\Sigma, <, \downarrow_1, \ldots, \downarrow_p)$ ayant exactement une variable libre x, on note

$$\mathcal{L}(\varphi) = \{t \in T_p(\Sigma') \mid \forall u \in \mathrm{dom}(t),\ t_2(u) = 1 \text{ ssi } t_1, x \mapsto u \models \varphi\}\,.$$

[4] **b)** On considère la formule

$$\varphi_1(x) = \exists y \left(P_a(y) \wedge \bigvee_{n=1}^{p} x \downarrow_n y \right).$$

Donner un automate d'arbres $\mathcal{A}_1$ déterministe et complet (ascendant) qui reconnaît $\mathcal{L}(\varphi_1)$.
Il faut bien sûr prouver que $\mathcal{A}_1$ reconnaît $\mathcal{L}(\varphi_1)$.

[4] **c)** On considère la formule

$$\varphi_2(x) = \forall z \left[(x \leq z \wedge \mathrm{feuille}(z)) \rightarrow \exists y \left(x \leq y \leq z \wedge P_b(y) \wedge \forall z\, (x \leq z < y \rightarrow P_a(z)) \right) \right]$$

dans laquelle $\mathrm{feuille}(z)$ est une macro pour $\neg \exists y\, z < y$.
Donner un automate d'arbres $\mathcal{A}_2$ déterministe et complet (ascendant) qui reconnaît $\mathcal{L}(\varphi_2)$.
Il faut bien sûr prouver que $\mathcal{A}_2$ reconnaît $\mathcal{L}(\varphi_2)$.

2 Fonctions séquentielles

Le filtre à rebonds est l'application $f : \{0,1\}^* \to \{0,1\}^*$ définie par $f(u_1 \cdots u_n) = v_1 \cdots v_n$ avec

$$v_i = \begin{cases} 0 & \text{si } u_{i-1} = u_{i+1} = 0 \\ 1 & \text{si } u_{i-1} = u_{i+1} = 1 \\ u_i & \text{sinon} \end{cases}$$

en posant $u_0 = u_{n+1} = 0$. Par exemple, $f(0101101) = 0011110$.

[4] **a)** Donner (dessiner) un automate séquentiel ayant au plus 5 états qui réalise la fonction f.
Normaliser cet automate et dessiner l'automate obtenu.
Minimiser l'automate précédent et dessiner l'automate minimal du filtre à rebonds.

Remarque : Un filtre à rebonds peut être utilisé pour " lisser " une suite de 0 et de 1 par exemple pour réduire les imperfections d'une image.

3 Grammaires

On considère la grammaire G des expressions postfixes définie par

$$S \to a \mid SS+ \mid SS\times$$

sur l'alphabet terminal $\Sigma = \{a, +, \times\}$.

Sur le même alphabet, on considère aussi la grammaire G' définie par les règles :

$$\begin{aligned} S &\to aS_1 \\ S_1 &\to \varepsilon \mid aS_1S_2 \\ S_2 &\to +S_1 \mid \times S_1 \end{aligned}$$

[2] **a)** Montrer que la grammaire G n'est pas LL.
Calculer First_1 et Follow_1 pour toutes les variables de la grammaire G'.
En déduire que G' est LL(1).
Construire la table d'analyse fortement LL(1) de la grammaire G'.

[2] **b)** Construire une grammaire G'' en forme normale de Greibach qui soit équivalente à G (si vous n'utilisez pas la construction du cours, il faudra prouver que G'' est équivalente à G).
Montrer que les grammaires G' et G'' sont équivalentes.

[4] **c)** Montrer que la grammaire G engendre le langage

$L = \{w \in \Sigma^* \mid |w|_a = |w|_+ + |w|_\times + 1 \text{ et } |u|_a > |u|_+ + |u|_\times \text{ pour tout préfixe propre } \varepsilon < u < w\}$

[2] **d)** Le langage engendré par la grammaire G est-il rationnel ?
La grammaire G est-elle ambiguë ?

[3] **e)** Donner un automate à pile déterministe, temps-réel, simple qui reconnaît par sommet de pile le langage engendré par la grammaire G.
Il faut bien sûr prouver que l'automate à pile reconnaît L.