Langages Formels

Licence Informatique – ENS Cachan

Examen du 27 mai 2010

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 Indices et exposants

On s'intéresse dans cette partie aux expressions LaTeX utilisant des indices et des exposants comme par exemple :

`x_i`	x_i	`x_{i+1}`	x_{i+1}	`x^n`	x^n	`x^{n+1}`	x^{n+1}
`x_i^2`	x_i^2	`x^2_i`	x_i^2	`x^{n^2}`	x^{n^2}	`x^{n_i}`	x^{n_i}
`x_{i_j}`	x_{i_j}	`x_{i^2}`	x_{i^2}	`{x^2}^2`	x^{2^2}	`x^{2^2}`	x^{2^2}
`x_{i_i}`	x_{i_i}	`{x_i}_i`	x_{ii}	`x_i_j`	erreur	`x_i^2_j`	erreur

On considère la grammaire G_1 définie par

$$S \to a \mid \{S\} \mid S \hat{} S \mid S \_ S$$

ainsi que la grammaire G_2 définie par

$$1 :\ S \to T \qquad 2 :\ S \to T \hat{} T \qquad 4 :\ S \to T \hat{} T \_ T \qquad 6 :\ T \to a$$
$$3 :\ S \to T \_ T \qquad 5 :\ S \to T \_ T \hat{} T \qquad 7 :\ T \to \{S\}$$

On note G_2' la grammaire G_2 augmentée par la règle $0 :\ S' \to S$.

[2] **a)** Montrer que la grammaire G_1 est ambiguë et montrer qu'elle engendre des expressions non autorisées en LaTeX.
Montrer que G_2 engendre `a^{a_a}_a` et dessiner un arbre de dérivation pour ce mot.
Montrer que G_2 n'engendre pas le mot `a^a_a_a`.

[2] **b)** Montrer que la grammaire G_2 n'est pas LL.
Calculer First_1 et Follow_1 pour les variables S', S et T. Détailler les calculs.

[6] **c)** Calculer l'automate $\mathcal{C}_0$ des contextes pour les 0-items.
On se restreindra bien sûr aux états accessibles. On pourra représenter un état de façon concise sous la forme clot(W) en donnant seulement les éléments de W et pas tous les éléments de la clôture.
Donner la table d'analyse SLR de la grammaire G_2'.
Y a-t-il des conflits ? La grammaire est-elle ambiguë ?

On note Σ l'alphabet terminal des grammaires ci-dessus. Soit A l'alphabet réduit aux deux accolades { et } et π_A la projection de Σ sur A.

[2] **d)** Donner une grammaire G_3 qui engendre $\pi_A(\mathcal{L}_{G_2}(S))$.
Il faut bien sûr prouver que G_3 engendre bien ce langage.

[2] **e)** Montrer que tout mot engendré par G_2 est bien "parenthésé". Formellement, il faut prouver que $\pi_A(\mathcal{L}_{G_2}(S))$ est inclus dans le langage de Dyck D_1^* sur l'alphabet A.
Montrer que l'inclusion est stricte.

2 Distance d'édition

On définit 3 *opérations d'édition* sur les mots : insertion d'une lettre, suppression d'une lettre, modification d'une lettre. Par exemple, on peut passer du mot $abacacb$ au mot $aabaab$ au moyen de 3 opérations : insertion d'un a au début et suppression des deux c, ou encore suppression du premier b, modification des deux c en b et a respectivement.

La distance d'édition $d(u,v)$ entre deux mots u et v est le nombre minimal d'opérations d'édition qui permettent de passer de u à v. Par exemple, $d((abc)^3,(bca)^3) = 2$.

Si $L \subseteq \Sigma^*$ est un langage et $k \in \mathbb{N}$ un entier, on note $L_k = \{v \in \Sigma^* \mid \exists u \in L,\ d(u,v) \leq k\}$.

[4] **a)** Soit $\Sigma = \{a,b,c\}$ et $L = \Sigma^* ab \Sigma^* ab \Sigma^*$.
Montrer que $L_1 = \Sigma^* ab \Sigma^* (a+b) \Sigma^* \cup \Sigma^* (a+b) \Sigma^* ab \Sigma^*$.
Calculer l'automate minimal de L_1.

[2] **b)** Montrer que si $L \subseteq \Sigma^*$ est reconnu par un automate (déterministe ou non) ayant n états, alors le langage L_1 peut être reconnu par un automate non déterministe ayant $\mathcal{O}(n)$ états.
En déduire que l'on peut construire un automate déterministe pour L_1 ayant au plus $2^{\mathcal{O}(n)}$ états à partir d'un automate (déterministe ou non) pour L ayant n états.

[3] **c)** Soit $\Sigma = \{a,b\}$ et $L = \Sigma^* aa \Sigma^n$.
Donner un automate non déterministe reconnaissant L avec $\mathcal{O}(n)$ états.
Montrer que $L_1 = \Sigma^* a \Sigma^n \cup \Sigma^* a \Sigma^{n+1}$.
Montrer que l'automate minimal de L_1 a au moins $2^{\frac{n}{2}}$ états.

[3] **d)** Soit $L \subseteq \Sigma^*$ un langage rationnel, on définit la fonction partielle $f : \Sigma^* \to \Sigma^*$ dont le domaine est L_1 et qui à un mot $v \in L_1$ associe le plus petit mot dans l'ordre lexicographique de $\{u \in L \mid d(u,v) \leq 1\}$. Montrer que la fonction f n'est pas forcément séquentielle.