Licence ST Informatique – S4 – 2008/2009

Automates et langages

mai 2009

Automate minimal déterministe

Nous allons voir comment, à partir d'un automate, construire l'automate déterministe complet minimal équivalent. Tout d'abord, rappelons que l'on peut construire à partir d'un automate quelconque, un automate $M = (X, Q, I, F, \delta)$ déterministe complet équivalent, en appliquant l'algorithme de déterminisation. Tous les états de M sont accessibles.

Définition 1 : Soit $M = (X, Q, I, F, \delta)$ un automate déterministe complet dont tous les états sont accessibles. On définit la relation $\simeq_M$ sur $Q \times Q$ par :

$$q \simeq_M q' \text{ ssi } \forall w \in X^*, (\delta^*(q, w) \cap F = \emptyset) \leftrightarrow (\delta^*(q', w) \cap F = \emptyset)$$

rappelons que comme M déterministe complet, $\delta^*(q, w)$ est toujours un singleton

Remarque :

- $\simeq_M$ est une relation d'équivalence.
- Intuitivement, deux états sont équivalents si on peut reconnaître les même mots à partir de ces deux états.
- Si une classe d'équivalence contient un état de sortie, alors tous ses éléments sont des états de sortie.

Définition 2 : Soit $M = (X, Q, \{q_0\}, F, \delta)$ un automate déterministe complet dont tous les états sont accessibles. Soit $M_{\text{min}} = (X, Q_{\text{min}}, I_{\text{min}}, F_{\text{min}}, \delta_{\text{min}})$ l'automate défini par :

- $Q_{\text{min}} = Q/\simeq_M$, ensemble des classes d'équivalence de Q pour $\simeq_M$.
- $I_{\text{min}} = [q_0]$, la classe de l'état initial de M.
- $F_{\text{min}} = \{[q] \mid q \in F\}$
- $\forall q, \forall x, \delta_{\text{min}}([q], x) = [\delta(q, x)]$

Proposition 1 : M_{min} est l'automate minimal de $L(M)$, c'est-à-dire qu'il n'existe pas d'automate déterministe complet équivalent avec strictement moins d'états.

Pour construire M_{min} à partir de M, on construit les classes d'équivalence de Q pour $\simeq_M$. Voici, schématiquement, l'algorithme utilisé :

- On démarre avec une partition de Q en deux ensembles : F et $Q \backslash F$.
- Pour chaque paire d'états q_1 et q_2 appartenant au même ensemble E, et pour chaque lettre x, si $\delta(q_1, x)$ et $\delta(q_2, x)$ n'appartiennent pas au même ensemble, alors on "sépare" q_1 et q_2 en partitionnant E.

- Lorsqu'on ne peut plus séparer d'états, alors la partition obtenue est l'ensemble des classes d'équivalence.

cf l'exemple traité en cours

L'automate minimal reconnaissant un langage L est unique et donne une **représentation canonique** du langage L : on peut donc décider de l'équivalence de deux automates, i.e. on peut écrire un programme qui accepte en entrée deux automates et qui, en sortie, renvoie TRUE s'ils sont équivalents, FALSE sinon. Comme on sait construire un automate à partir d'une expression rationnelle (cf le cours sur la construction de Glushkov), on sait aussi décider de l'équivalence de deux langages rationnels.
Nous avons également vu en cours le lien entre les états de l'automate minimal reconnaissant L et les résiduels du langage L, mais ça ne fait pas partie du programme de l'examen