Année 2007-2008

Examen de Maths Discrètes
9 Janvier 2008
éléments de corrections

/!\ : corrigé partiel et non officiel,
à lire d'un œil critique

Exercice 1 :

Question 1 :

Rappel de cours : le nombre de solution d'un système

Définition : On dit qu'un langage L est solution d'un système algébrique S (ou grammaire) si S engendre exactement les mots appartenants à L.

Définition : Un système propre, est un système dans lequel on ne trouve aucune transition de la forme $S \rightarrow \varepsilon$ (autre notation : $S \rightarrow 1$), et aucune transition de la forme $S \rightarrow S'$ (où S' est un non-terminal isolé différent de S).

Théorème : Un système propre admet une unique solution (ne contenant pas ε).

Définition : Un système strict, est un système dans lequel on ne trouve que des transitions de la forme $S \rightarrow \varepsilon$ ou $S \rightarrow m$, avec m contenant au moins un terminal.

Théorème : Un système strict admet une unique solution.

Correction :

Ce système est strict, car il ne contient que des transitions de la forme $S \rightarrow \varepsilon$ ou $S \rightarrow m$, avec m contenant au moins un terminal.
D'après le théorème du cours, il y admet donc une unique solution.

Solution intuitive (non demandée) :

$L_x = \{ w : \exists k \in \mathbb{N} : |w|_a = |w|_b = 2k \text{ et } \forall u \leq_{pref} w, |u|_a \geq |u|_b \}$
$L_y = \{ avb \cup bva : v \in L_x \}$

Question 2 :

Rappel de cours : les approximants

Définition : Un approximant de rang *n* d'un non-terminal S est un mot qu'on peut former à partir de S avec *n* dérivations ou moins.

En d'autres termes :

- Les approximants de rang 0 sont ∅ (car on ne génère aucun mot à partir d'un non-terminal sans effectuer au moins une dérivation).
- Les approximants de rang 1 sont les mots que l'on peut obtenir en supposant que les non-terminaux dans la partie droite de l'équation peuvent générer les approximants de rang 0 (c'est à dire ∅, autrement dit les approximants de rang 1 ne sont que les terminaux qu'on voit à droite dans les règles).
- Les approximants de rang 2 sont les mots que l'on peut obtenir en supposant que les non-terminaux dans la partie droite de l'équation peuvent générer les approximants de rang 1.

et ainsi de suite...

Correction :

Calcul des approximants :

A_0 : ∅
∅

A_1 : { ε } car si on suppose x = ∅ et y = ∅ à droite de l'équation, *a*x*b*y = ∅ (et ∅ + ε = ε)
∅ car si on suppose x = ∅ et y = ∅ à droite de l'équation, *a*x*b* = ∅ et *b*x*a* = ∅

A_2 : { ε } car si on suppose y = ∅ à droite de l'équation, *a*x*b*y = ∅
{ *ab*, *ba* } car si on suppose x = ε à droite de l'équation, *a*x*b* = *ab*, et *b*x*a* = *ba*

A_3 : { ε, *aabb*, *abab* } etc...
{ *ab*, *ba* }

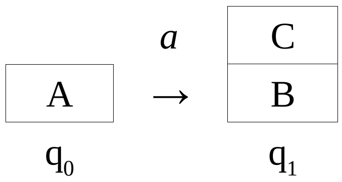
Rang 4 (non demandé) :

A_4 : { ε, *aabb*, *abab*, *aabbaabb*, *aabbabab*, *ababaabb*, *abababab* }
{ *ab*, *ba*, *aaabbb*, *baabba*, *aababb*, *bababa* }

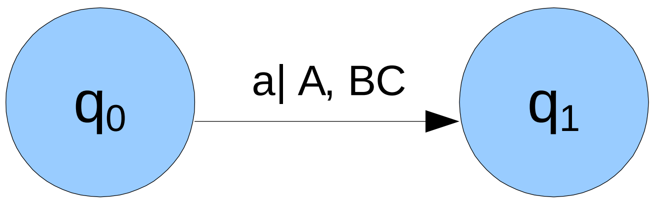
Exercice 2 :

Rappel de cours : les transitions d'un automate à pile

Une transition que l'on note par exemple : t (a, q_0, A) = (q_1, BC)
signifie : "Depuis l'état q_0, en ayant A au sommet de la pile, et en lisant la lettre a, on peut passer dans l'état q_1 et remplacer le A en sommet de pile par BC" (B en dessous, C au dessus) :



Sur un dessin d'automate, où l'on indique déjà à l'aide d'une flèche qui sont l'état de départ et l'état cible, on peut simplement noter la transition "a | A, BC" :



On lit toujours une et une seule lettre (ici : a).
On dépile toujours un et un seul symbole de sommet de pile (ici : A).
Mais on peut empiler autant de symbole de pile qu'on veut (ici, on en empile 2 : BC), y compris aucun, et dans ce cas on note ce rien « ε » (exemple : a | A, ε).

NB : Il existe 2 façons de noter ce que l'on empile dans les transitions d'un automates à pile. Ici, dans un souci de cohérence avec la notation du professeur de Paris 7 et des anciens examens, nous choisirons la façon qui consiste à mettre le sommet de la pile à droite (a | A, BC), mais on peut aussi trouver dans les corrections de certains professeurs de TD ou dans certains ouvrages une notation qui met le sommet de la pile à gauche (a | A, CB).

Méthode :

Simplifions l'écriture de L :

$L = \{ a^n c a^p c \mid p = n+1 \} = a^n c a^n a c$ (car $\{a^p \mid p = n+1\} = a^{n+1} = a^n a$)

Mettre en évidence les membres de même longueur (ici, les 2 occurrences de a^n) permet de mieux visualiser ce qu'il faudra empiler et dépiler si on veut construire un automate à pile.

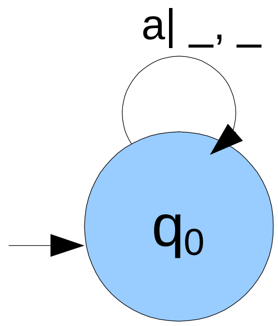
Question 1 :

On a 2 façons de construire un automate à pile :

1) La façon intuitive (de loin ma préférée) :

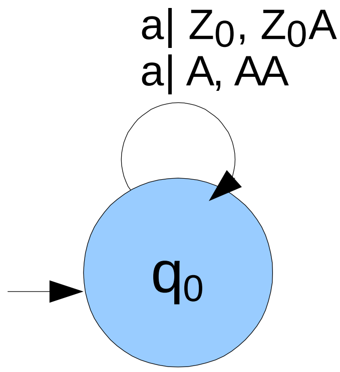
Tout automate qui se respecte doit avoir un symbole de fond de pile initial, appelons le Z_0.

On veut pouvoir lire un nombre quelconque de *a* au début du mot, on crée donc un premier état initial q_0 et une transition qui reste sur q_0 tant qu'elle lit un *a :*



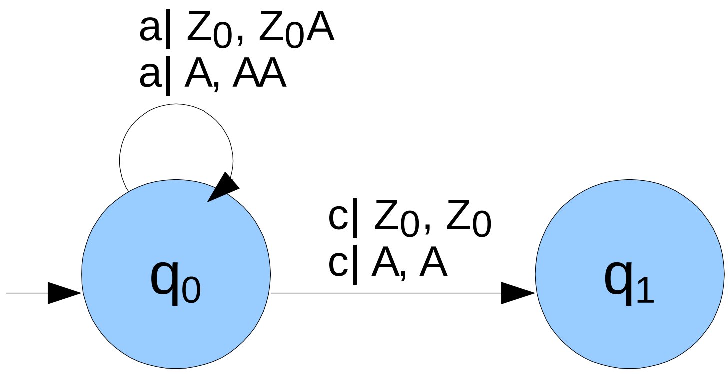
On veut compter combien de *a* sont ainsi lu (puisque plus loin dans le mot, on devra s'assurer qu'une autre chaîne est bien de même longueur), on utilise donc la pile de l'automate, en empilant un marqueur (appelons le "A") pour chaque *a* lu de cette façon, ce qui nous donne un état initial q_0 et 2 transitions :

- t (a, q_0, Z_0) = (q_0, Z_0A) (pour lire le 1er *a*, on aura un Z_0 en sommet de pile)
- t (a, q_0, A) = (q_0, AA) (pour lire les suivants, on aura un A en sommet de pile)



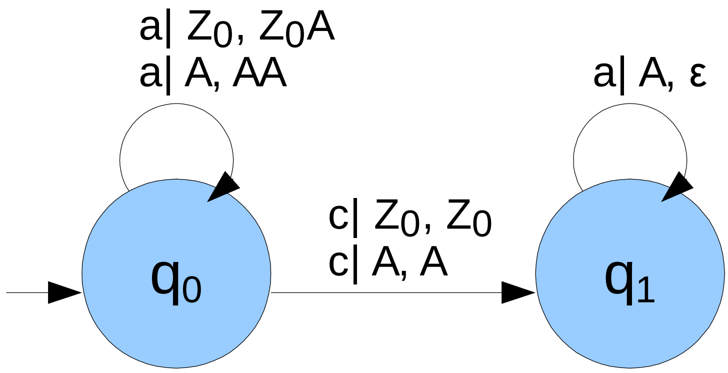
On veut ensuite lire un *c* (pas besoin de compter puisqu'on sait précisément combien on en veut, on ne touche donc pas à la pile), on crée donc un état q_1 vers lequel renvoie *c* (le changement d'état est important, puisque le comptage des *a* du premier membre ne devra pas être repris ensuite), ce qui donne les transitions :

- t (c, q_0, Z_0) = (q_1, Z_0) (au cas où on n'aurait pas lu de *a* avant)
- t (c, q_0, A) = (q_1, A) (au cas où on en aurait lu)



On veut à présent lire des *a*, le même nombre que la première fois, on donne donc à q_1 une transition permettant de dépiler un compteur A pour chaque *a* lu :

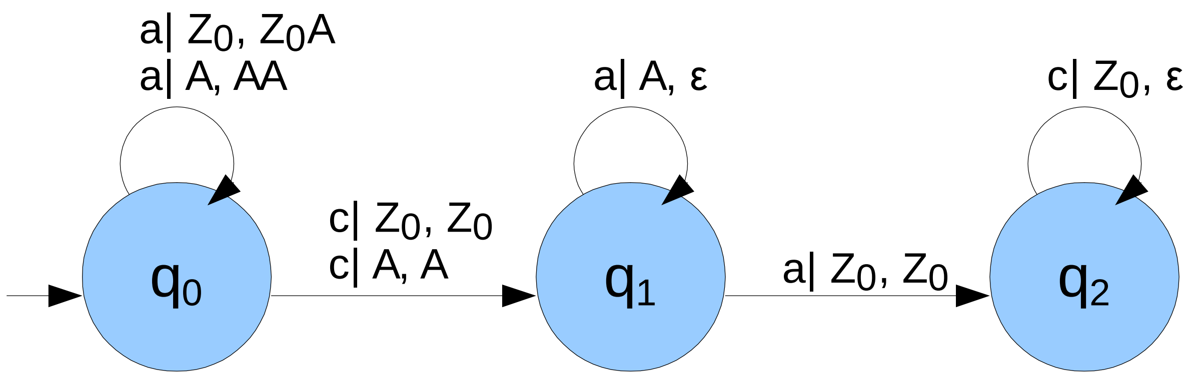
- t (*a*, q_1, A) = (q_1, ε)



On ajoute enfin une transition permettant de lire le dernier *a* une fois tous les marqueurs dépilés, et renvoyant dans un état q_2 où on videra la pile en lisant la dernière lettre voulue, un *c* (puisque l'automate voulu est par pile vide) :

- t (*a*, q_1, Z_0) = (q_2, Z_0)
- t (*c*, q_2, Z_0) = (q_2, ε)

On obtient ainsi l'automate suivant :



Symbole initial de fond de pile : Z_0
Acceptation par pile vide

2) La façon méthodique (à n'utiliser qu'en dernier recours) :

À partir de la version simplifiée de L, construisons une grammaire qui l'engendre :

$S_0 \to aS_0a + c$ $\qquad$ $(S_0 = a^nca^n)$

$S_1 \to S_0ac$ $\qquad$ $(S_1 = a^nca^nac = L)$

Mettons la sous forme normale de Greibach :
Cette grammaire est déjà propre et réduite, rendons là directement "presque Greibach" :

$S_0 \to aS_0a + c$

$S_1 \to aS_0aac + cac$ (on remplace S_0 par son contenu)

et maintenant sous forme normale de Greibach :

$S_0 \to aS_0A + c$

$S_1 \to aS_0AAC + cAC$

$A \to a$

$C \to c$

À présent, on crée un automate à pile avec un unique état initial Q (parfois aussi appelé "automate sans état"), et pour chaque $X \to aY$ dans la grammaire, on ajoute une transition de la forme t (a, Q, X) = (Q, Z), avec Z = “Y écrit à l'envers” (puisqu'on a choisi plus haut de noter les transitions avec le sommet de pile à droite) :

$t(a, Q, S_0) = (Q, AS_0)$

$t(c, Q, S_0) = (Q, \varepsilon)$

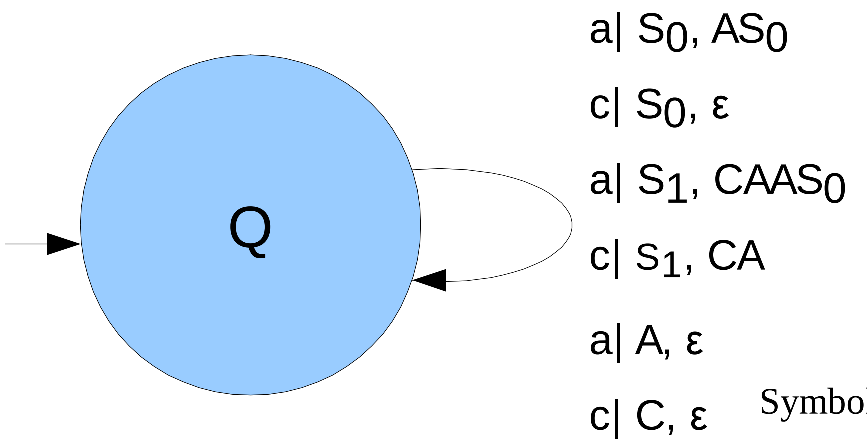
$t(a, Q, S_1) = (Q, CAAS_0)$

$t(c, Q, S_1) = (Q, CA)$

$t(a, Q, A) = (Q, \varepsilon)$

$t(c, Q, C) = (Q, \varepsilon)$

NB : N'oublions pas que le non terminal engendrant L est S_1, il devra donc être le fond de pile initial de l'automate.
On obtient ainsi cet automate (parfaitement équivalent au précédent) :



Symbole initial de fond de pile : S_1
Acceptation par pile vide

Question 2 :

(Suite traitée prochainement)

Questions, commentaires, critiques, remarques, etc : phosphore85@gmail.com