

TD N° 2

1. Les langages finis sont réguliers

a. Soit le langage : $L = \{ba, aba, abca\}$. Classer ses mots dans l'ordre alphabétique, comme dans un dictionnaire.

b. En déduire un AFD qui reconnaît le langage (fini) L .

c. Quel est le langage engendré quand on rend finals tous les états de l'automate précédent ?

d. Soit L un langage fini. On note K l'ensemble de tous les préfixes des mots de L . On construit un AFD de la manière suivante :

- les états sont les éléments de K ,
- l'état initial est ϵ , l'ensemble des états finals est L (inclus dans K),
- $\delta(w, a) = wa$ si $wa \in K$.

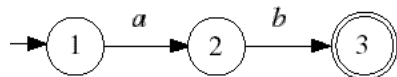
Construire l'AFD correspondant au langage $L' = \{aba, ba, bba, bbb\}$.

Facultatif : En faisant une induction sur la longueur de w , montrer que :

$\delta(\epsilon, w) = w$ si $w \in K$.

NB. Cette technique de représentation de lexiques, où les préfixes communs sont mis en facteur, est appelée « l'arbre des lettres » (ou arbre digital), et « trie » en anglais (jeu de mot sur *tree* et *retrieval*).

e. Une autre manière particulièrement simple de représenter un ensemble fini de mots est de construire un AFN. Pour chaque mot w , on construit un automate appelé *écorché* de w obtenu en insérant des états entre les lettres de w : l'écorché de w a donc $|w| + 1$ états et autant de transitions que de lettres dans w . Par exemple, l'écorché de ab :



Construire les trois automates écorchés des mots du langage $L = \{ba, aba, abca\}$ de la question **a.** En faisant l'union de ces trois automates, on obtient un AFN pour L .

Appliquer l'algorithme de déterminisation à l'AFN ainsi obtenu. Que remarque-t-on ?

2. Nombres

Ecrire un automate reconnaissant les nombres sans signe. Exemples : 5280, 39.37, 1.849E-4. Le premier chiffre peut être zéro (voir Aho, Sethi, Ullman, p. 124). Avez-vous obtenu un AFN ou un AFD ?

3. AFN

a. Dessiner l'automate non déterministe AFN défini par :

$\Sigma = \{a, b\}$, $Q = \{1, 2, \dots, 6\}$, δ , $\text{Init} = \{1, 3, 5\}$, $\text{Fin} = \{4, 6\}$

Avec δ donnée par la table :

	1	2	3	4	5	6
a	{2}	\emptyset	{4}	{4}	{5}	\emptyset
b	\emptyset	{3}	\emptyset	\emptyset	{5, 6}	\emptyset

Décrire le langage reconnu par cet automate.

On peut utiliser deux méthodes pour reconnaître un mot par un AFN.

b. Parcours en profondeur d'abord. On gère une pile de couples (état courant, position dans le mot). Le premier choix est celui de l'état de départ. Un choix également chaque fois qu'il y a deux transitions possibles. Quand on est bloqué (pas de transition possible) on dépile jusqu'à trouver (éventuellement) une autre transition.

Simuler sur le mot *abaab* en indiquant les couples successifs. Les premières valeurs sont :

(1, *abaab*),

(12, *baab*),...

c. Parcours en largeur d'abord (version dynamique de la détermination). On gère une suite de couples (ensemble d'états, position dans le mot). Au départ (Init, début de mot) = ($\{1,3,5\}$, *abaab*). On passe de ($\{q_1, q_2, \dots\}$, xw) à ($\{q'_1, q'_2, \dots\}$, w) si $\{q'_1, q'_2, \dots\}$ est l'ensemble d'états que l'on peut obtenir à partir de $\{q_1, q_2, \dots\}$ par une transition étiquetée par la lettre x .

Même question qu'en **b**. Les premières valeurs sont :

($\{1, 3, 5\}$, *abaab*),

($\{2, 4, 5\}$, *baab*),...

d. Calculer un AFD équivalent à l'AFN précédent (par l'algo du cours).

4. (Examen juin 2006)

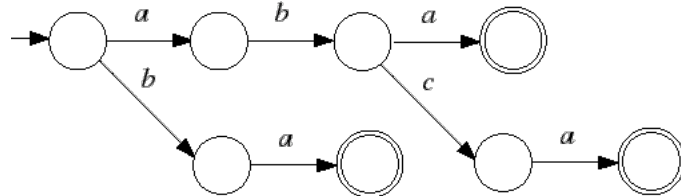
Construire un AFD reconnaissant les nombres divisibles par 3. Indication :

TD N° 2 - CORRECTION

1.

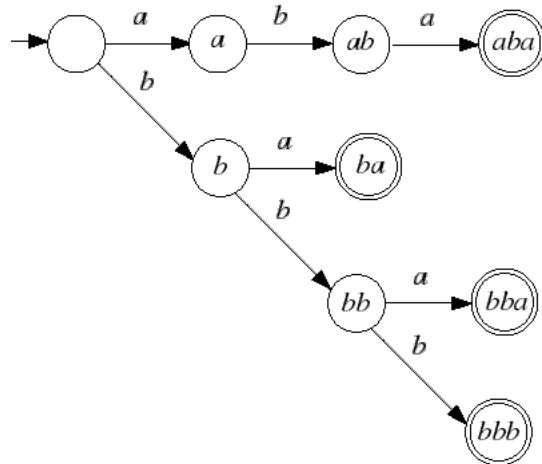
a. Ordre alphabétique : *aba, abca, ba*

b.

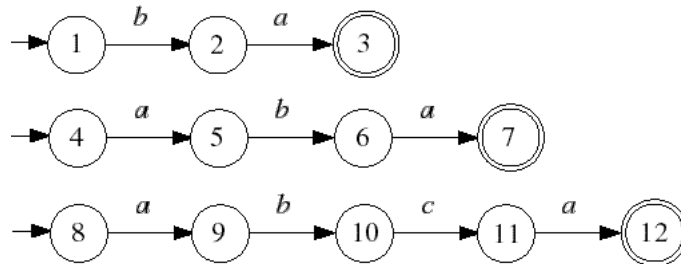


c. L'ensemble des préfixes des mots de *L*.

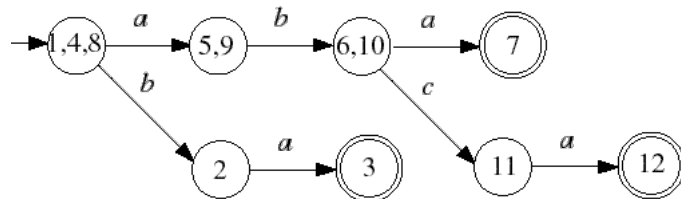
d.

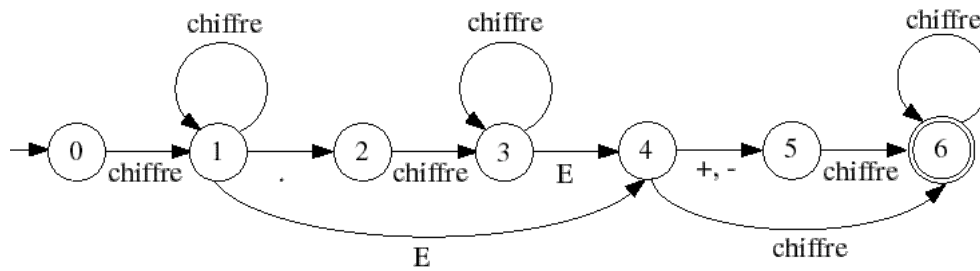


e.



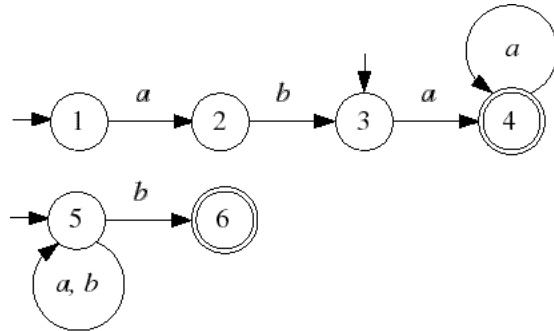
Après détermination, on retrouve l'AFD du a.





3

a.



b. Valeurs de la pile :

- (1, *abaab*)
- (12, *baab*)
- (123, *aab*)
- (1234, *ab*)
- (12344, *b*) bloqué, tout dépiler
- (3, *abaab*)
- (34, *baa*) bloqué, tout dépiler
- (5, *abaab*)
- (55, *baab*)
- (555, *aab*)
- (5555, *ab*)
- (55555, *b*)
- (555555, ϵ) pas terminal, dépiler un élément
- (555556, ϵ) 6=terminal, donc reconnu

c. Valeurs de la suite de couples :

- ({1, 3, 5}, *abaab*)
- ({2, 4, 5}, *baab*)
- ({3, 5, 6}, *aab*)
- ({4, 5}, *ab*)
- ({4, 5}, *b*)
- ({5, 6}, ϵ) 6=terminal, donc reconnu