



## THÉORIES DES LANGAGES

Mr,HEMIOUD  
hemourad@yahoo,fr  
Université de Jijel  
Département d'informatique



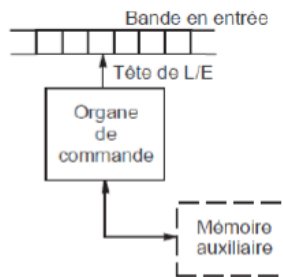
## LES AUTOMATES FINIS



2

## LES AUTOMATES À ÉTATS FINI

- Un **automate** est une machine abstraite qui permet de lire un mot et de répondre à la question : "**un mot  $w$  appartient-il à un langage  $L$  ?**" par **oui** ou **non**.
- Un automate est composé de :



3

## Formellement

- Un **alphabet** pour les mots en entrée noté  $A$  ;
- Un ensemble non vide **d'états** noté  $Q$ ;
- Un **état initial** noté  $q_0 \in Q$ ;
- Un ensemble *non vide* d'états fin  $Q_f \in Q$ ;
- Une **fonction de transition** (permettant de changer d'état) notée  $\delta$  (est une fonction totale de  $Q \times A$  dans  $Q$ ,

$$\delta(p, a) = q \text{ / } p, q \in Q$$

4

### Configuration d'un automate

- Le fonctionnement d'un automate sur un mot se fait à travers un ***ensemble de configurations***.
- On appelle ***configuration d'un automate*** en fonctionnement les valeurs de ses différents composants, à savoir la ***position*** de la tête L/E, ***l'état*** de l'automate et éventuellement le ***contenu*** de la mémoire auxiliaire (lorsqu'elle existe).

5

Il existe deux configurations spéciales :

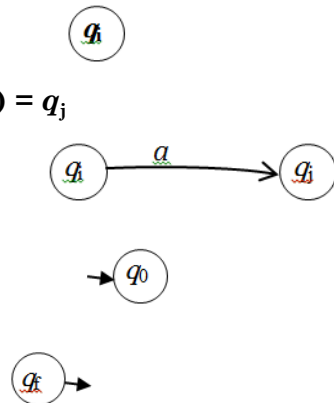
- La configuration ***initiale*** est celle qui correspond à l'état initial  $q_0$  et où la **tête** de L/E est *positionnée* sur le *premier* symbole du mot à lire.
- Une configuration ***finale*** est celle qui correspond à un des états finaux  $q_f$  et où le mot a été *entièrement* lu.

6

## Représentation graphique d'un automate

Un automate fini correspond à un graphe orienté

- L'état  $q_i$
- la transition  $\delta(q_i, a) = q_j$
- Etat initiale ( $q_0$ ):
- Etats finaux ( $q_f$ )

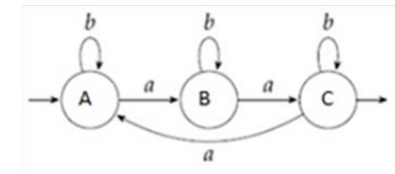


7

**Exemple :** Représentation graphique de l'automate

$X = (A = \{a, b\}, Q = \{A, B, C\}, q_0 = A, q_F = \{C\}, \delta)$  tel que :

$\delta(A, a) = B$   
 $\delta(A, b) = A$   
 $\delta(B, a) = C$   
 $\delta(B, b) = B$   
 $\delta(C, b) = C$



8

### Mot reconnu par un automate

Un mot  $w$  est **reconnu par l'automate** A s'il existe une *configuration successive*

Configuration-initiale  $(w) \vdash^* \text{configuration-final}(w)$

$$(q_0, w) \vdash^* (q_f, \varepsilon)$$

- La relation  $\vdash$  permet de formaliser la notion d'étape élémentaire de calcul d'un automate. Ainsi on écrira, pour  $a$  dans  $A$  et  $v$  dans  $A^*$  :

$$(q, \underline{a}v) \vdash (\delta(q, a); v)$$

9

### Langage reconnu par un automate

On dit qu'un **langage est reconnu par un automate** X lorsque tous les mots de ce langage sont reconnus par l'automate on note  $L(X)$

$$L(X) = \{w \in A^* / \text{Configuration-initiale}(w) \vdash^* \text{configuration-final}(w)\}$$

$$L(X) = \{ w \in A^* / (q_0, w) \vdash^* (q, \varepsilon), \text{ avec } q \in Q_F \}$$

10

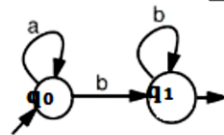
- **Exemple** : L'automate qui reconnaît les mots de la forme  $a^n b^m$  ( $n \geq 0, m > 0$ ) est le suivant :

( $\{a, b\}, \{q_0, q_1\}, q_0, \{q_1\}, \delta$ ) tel que  $\delta$  est donnée par :

- 1)  $\delta(q_0, a) = q_0$
- 2)  $\delta(q_0, b) = q_1$
- 3)  $\delta(q_1, b) = q_1$

ou par la table :

Etat	a	b
$q_0$	$q_0$	$q_1$
$q_1$	-	$q_1$

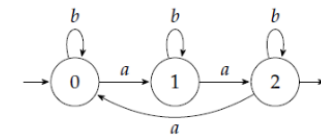


11

### Exemple 2 :

La fonction de transition correspondant à ce graphe s'exprime matriciellement par

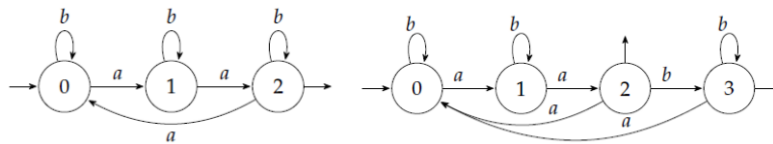
$\delta$	a	b
0	1	0
1	2	1
2	0	2



- Donnez 3 mots qui sont reconnu par cet automate

12

- **Langage reconnaissable** : Un langage est reconnaissable s'il existe un automate fini qui le reconnaît.
- **Automates équivalents** : Deux automates finis  $X_1$  et  $X_2$  sont équivalents si et seulement s'ils reconnaissent le même langage ( $L(X_1)=L(X_2)$ )

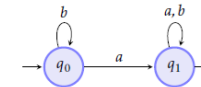


Les deux automates reconnaissent le langage de tous les mots qui contiennent un nombre de  $a$  congru à 2 modulo 3.

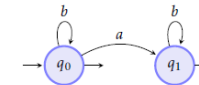
13

### Exemples

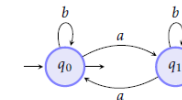
1. Le langage des mots contenant au moins une fois la lettre  $a$  :



1. Le langage des mots contenant au plus une fois la lettre  $a$  :



2. Le langage des mots contenant un nombre pair de fois la lettre  $a$  :



14

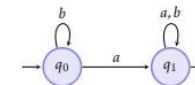
### Exercice

Pour chacun des langages suivants, construire un automate d'états finis qui l'accepte :

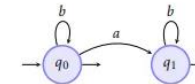
1. le langage dénoté par  $aba + bab$ .
2. Le langage des mots admettant  $aba$  pour facteur
3. le langage dénoté par  $(aba)^* + (bab)^*$ .
4.  $L = \{ w \in \{a, b\}^* / w = a^n b^m a \text{ ou } w = b a^n ; n, m \geq 1 \}$  ;
5.  $L = \{ w \in \{0, 1\}^* / w = 1(101)^n 00 \text{ ou } w = 0(010)^n 11, n \geq 0 \}$

15

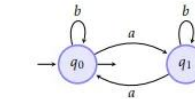
Le langage des mots contenant au moins une fois la lettre  $a$  :



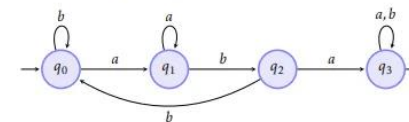
Le langage des mots contenant au plus une fois la lettre  $a$  :



Le langage des mots contenant un nombre pair de fois la lettre  $a$  :



Le langage des mots admettant  $aba$  pour facteur :



16



## Passage de l'expression régulière vers l'automate

Il existe trois méthodes permettant de réaliser cette tâche.

1. Algorithme de Glushkov
2. Algorithme de Thompson
3. La méthode des dérivées

17

## Passage de l'expression régulière vers l'automate (algorithme de Glushkov)

- Exemple:  $(ab + c)^*ab$  est linéarisée en  $(12 + 3)^*45$ .

	a	b	c
0	{1, 4}	$\emptyset$	{3}
1	$\emptyset$	{2}	$\emptyset$
2	{1, 4}	$\emptyset$	{3}
3	{1, 4}	$\emptyset$	{3}
4	$\emptyset$	{5}	$\emptyset$
5	$\emptyset$	$\emptyset$	$\emptyset$

18

### Passage de l'automate vers l'expression régulière

Soit  $X = (A, Q, q_0, Q_F, \delta)$  un automate à états fini quelconque.

On note par  $L_i$  le langage reconnu par l'automate si son état initial était  $q_i$ .

Par conséquent, trouver le langage reconnu par l'automate revient à trouver  $L_0$  étant donné que la reconnaissance commence à partir de l'état initial  $q_0$ . L'automate permet d'établir un système d'équations aux langages de la manière suivante :

- si  $\delta(q_i, a) = q_j$  alors on écrit :  $L_i = aL_j$  ;
- si  $q_i \in Q_F$ , alors on écrit :  $L_i = \epsilon$
- si  $L_i = \alpha$  et  $L_j = \beta$  alors on écrit :  $L_i = \alpha | \beta$  ;

19

- Il suffit ensuite de résoudre le système précédant à des substitutions et en utilisant la règle suivante :

la solution de l'équation  $L = \alpha L | \beta$  ( $\epsilon \notin \alpha$ ) est le langage  $L = \alpha^* \beta$  (*Le lemme d'Arden*)

20

### Le lemme d'Arden

Ce résultat dit que l'équation réursive suivante

$$X = LX + M$$

où  $X$  est le langage cherché et  $L$  et  $M$  sont deux langages connus, a une unique solution qui est :

$$X = L^*M:$$

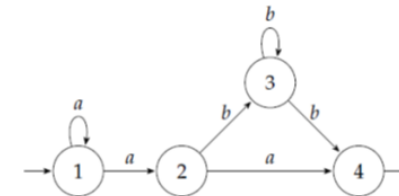
**Attention** : ici, on suppose que  $L$  ne contient pas le mot vide ( $\epsilon \notin L$ ).

21

### Automate fini déterministe AFD

Un AEF  $(A, Q, q_0, Q_F, \delta)$  est dit **déterministe** si les deux conditions sont vérifiées :

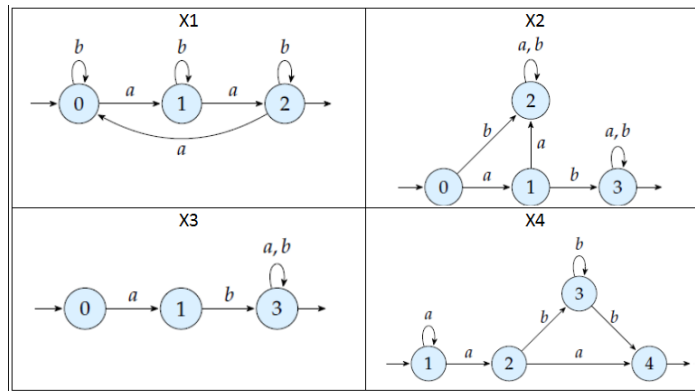
- $\forall q_i \in Q, \forall a \in X$ , il existe au plus un état  $q_j$  tel que  $\delta(q_i, a) = q_j$  ;
- L'automate ne comporte pas de  $\epsilon$ -transitions.



Deux transitions sortantes de 1 sont étiquetées par  $a$  :

$$\delta(1, a) = \{1, 2\}$$

22



23

### **Théorème (Déterminisation).**

Pour tout AFN  $X$ , on peut construire un AFD  $X'$  équivalent à  $X$ .

De plus, si  $X$  a  $n$  états, alors  $X'$  a au plus  $2^n$  états.

24

### Algorithme : Déterminiser un AEF sans les $\varepsilon$ -transitions

**Principe** : considérer des ensembles d'états plutôt que des états (dans l'algorithme suivant, chaque ensemble d'états représente un état du futur automate).

- 1- Partir de l'état initial  $E^{(0)} = \{q_0\}$  (c'est l'état initial du nouvel automate) ;
- 2- Construire  $E^{(1)}$  l'ensemble des états obtenus à partir de  $E^{(0)}$  par la transition a :

$$E^{(1)} = \bigcup_{q' \in E^{(0)}} \delta(q', a)$$

- 3- Recommencer l'étape 2 pour toutes les transitions possibles et pour chaque nouvel ensemble  $E^{(i)}$  ;

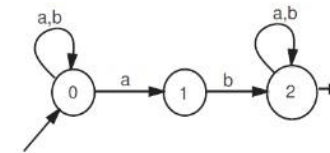
$$E^{(i)} = \bigcup_{q' \in E^{(i-1)}} \delta(q', a)$$

- 4- Tous les ensembles contenant au moins un état final du premier automate deviennent finaux ;
- 5- Renommer les états en tant qu'états simples.

25

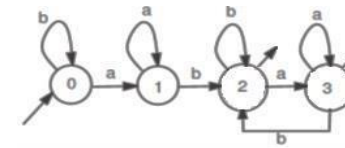
### Déterminisation d'un AEF sans $\varepsilon$ -transition

Exemple 1:



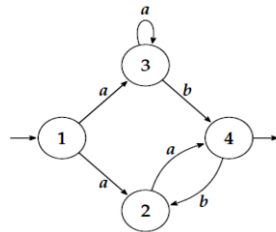
État	a	b	État	a	b
0	0,1	0	0	1	0
0,1	0,1	0,2	1	1	2
0,2	0,1,2	0,2	2	3	2
0,1,2	0,1,2	0,2	3	3	2

L'automate déterministe est

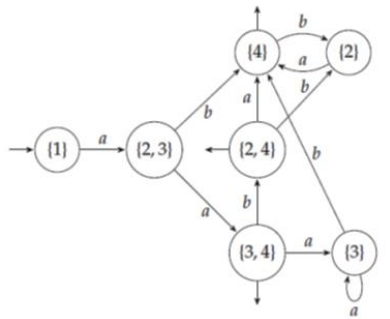


26

Exemple 2



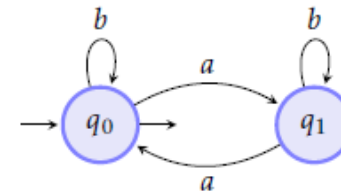
Le résultat de la détermination



27

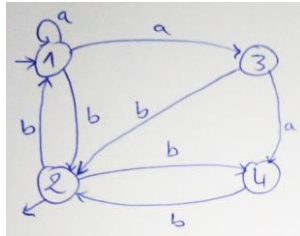
Exemple

Determiner le langage reconnu par l'automate suivant.

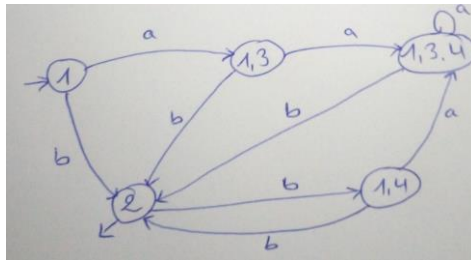


28

Exemple 3:



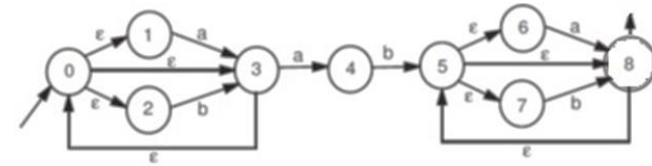
Le résultat de la déterminisation



29

### Déterminisation avec les $\epsilon$ -transitions

- On appelle  **$\epsilon$ -fermeture** de E l'ensemble des états incluant, en plus de ceux de E, tous les états accessibles depuis les états de E par un chemin étiqueté par le mot  $\epsilon$ .



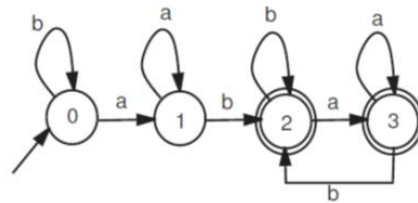
- $\epsilon$ -fermeture  $(\{0\}) = \{0, 1, 2, 3\}$
- $\epsilon$ -fermeture  $(\{1, 2\}) = \{1, 2\}$
- $\epsilon$ -fermeture  $(\{3\}) = \{0, 1, 2, 3\}$

30

**Exemple :** Appliquons l'algorithme de détermination à l'automate précédent

État	a	b	État	a	b
0,1,2,3	0,1,2,3,4	0,1,2,3	0	1	0
0,1,2,3,4	0,1,2,3,4	0,1,2,3,5,6,7,8	1	1	2
0,1,2,3,5,6,7,8	0,1,2,3,4,5,6,7,8	0,1,2,3,5,6,7,8	2	3	2
0,1,2,3,4,5,6,7,8	0,1,2,3,4,5,6,7,8	0,1,2,3,5,6,7,8	3	3	2

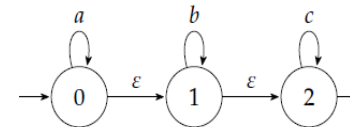
=>



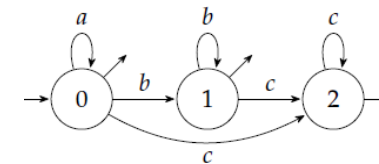
31

### Exemple 2

- Un automate avec  $\varepsilon$ -transitions correspondant à  $a^*b^*c^*$  ( $a^n b^m c^l$  avec  $n, m, l \geq 0$ )



- L'automate déterministe équivalent



32



### Minimisation d'un AEF déterministe

La minimisation s'effectue en éliminant les états dits **inaccessibles** et en *confondant* (ou fusionnant) les états reconnaissant le même langage.

- **Définition 1** : Un état est dit **inaccessible** s'il n'existe aucun chemin permettant de l'atteindre à partir de l'état initial.
  - → Donc les états inaccessibles sont improductifs, c'est-à-dire qu'ils ne participeront jamais à la reconnaissance d'un mot.
- **Définition 2** : Deux états  $q_i$  et  $q_j$  sont dits  **$\beta$ -équivalents** s'ils permettent d'atteindre les états finaux en utilisant les mêmes mots. On écrit alors :  $q_i \beta q_j$ .
  - La relation  $\beta$ -équivalence est donc dite une relation de **congruence**

33

### Minimiser un AEF

La méthode de réduction d'un AEF est la suivante :

- Nettoyer l'automate en éliminant les états **inaccessibles** ;
- Regrouper les états **congruents** (appartenant à la même classe d'équivalence).

« Un état est dit **inaccessible** s'il n'existe aucun chemin permettant de l'atteindre à partir de l'état initial. »

34

- **Exemple 13** : Soit à minimiser l'automate suivant (les états finaux sont les états 1 et 2 tandis que l'état 1 est initial) :

État	a	b
1	2	5
2	2	4
3	3	2
4	5	3
5	4	6
6	6	1
7	5	7

35

1. La première étape consiste à éliminer les états **inaccessibles**, il s'agit juste de l'état 7.

Les étapes de détermination des classes de congruences sont les suivantes :  $A = \{1, 2\}$ ,  $B = \{3, 4, 5, 6\}$  ;

2.  $\delta(3, b) = 2 \in A$ ,  $\delta(4, b) = 3 \in B$  ainsi il faut séparer 4 du reste de la classe B. Alors, on crée une classe C contenant l'état 4 ;
3.  $\delta(3, b) = 2 \in A$ ,  $\delta(5, b) = 6 \in A$  ainsi il faut séparer 5 du reste de la classe B. Mais inutile de créer une autre classe puisque  
 $\delta(4, a) = 5 \in B$ ,  
 $\delta(5, a) = 4 \in B$   
et  $\delta(4, b) = 3 \in B$   
et  $\delta(5, b) = 6 \in B$ ,  
il faut donc mettre 5 dans la classe C.  $C = \{4, 5\}$   
et  $B = \{3, 6\}$  ;
4. Aucun autre changement n'est possible, alors on arrête l'algorithme.

36

- Le nouvel automate est donc le suivant (l'état initial est A) :

Etat	a	b
A	A	C
B	B	A
C	C	A

- L'automate obtenu est **minimal** et est **unique**,

37

### Opérations sur les automates

#### Le complément

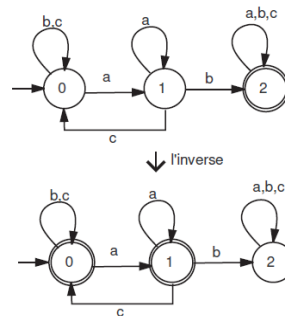
- Soit  $A = (A, Q, q_0, Q_F, \delta)$  un automate déterministe reconnaissant le langage  $L$ .
- L'automate reconnaissant le **langage inverse** (c'est-à-dire  $L$  ou  $A^* - L$ ) est défini par le quintuplet  $(A, Q, q_0, Q - Q_F, \delta)$  (*en d'autres termes, les états finaux deviennent non finaux et vice-versa*).
- cette propriété ne fonctionne que si l'automate est **complet**.

38

### Exemple

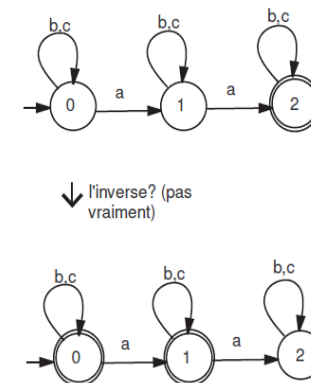
:

- Soit le langage des mots définis sur l'alphabet  $\{a, b, c\}$  **contenant le facteur ab**.
- appliquer la propriété précédente pour trouver l'automate des mots qui **ne contiennent** pas le facteur **ab**



39

### Exemple

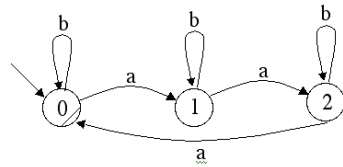


- L'automate obtenu reconnaît les mots contenant au plus 1 a

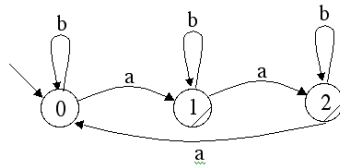
40

- On commence par L1 ayant un nombre de a multiple de 3. L1' est le complémentaire de L1 (le nombre de a n'est pas multiple de 3).

- L1



- L1'

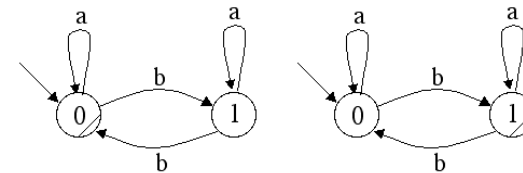


41

### Exemple

- Soit L2, le langage ou le nombre de b est pair et L2' son complémentaire. Nous avons les deux automates pour L2 et L2'

- 



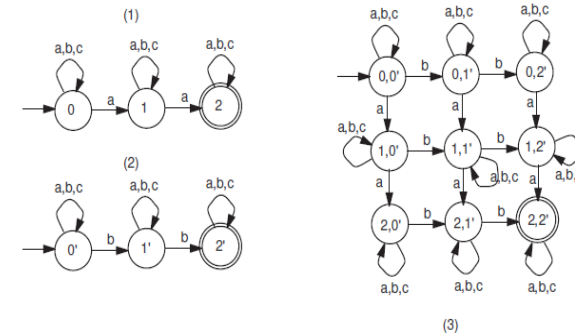
42

### Produit d'automates

- Soit  $X = (A, Q, q_0, Q_F, \delta)$  et  $X' = (A', Q', q_0', Q'_F, \delta')$  deux automates à états finis.
- On appelle produit des deux automates  $X$  et  $X'$  l'automate  $X''(A'', Q'', q_0'', Q''_F, \delta'')$  défini comme suit :
  - $A'' = A \cup A'$  ;
  - $Q'' = Q \times Q'$  ;
  - $q_0'' = (q_0, q_0')$  ;
  - $Q''_F = Q_F \times Q'_F$  ;
  - $\delta((q, q'), a) = (\delta(q, a), \delta(q', a))$
- $L(X * X') = L(X) \cap L(X')$

43

### Exemple



44

### Le langage miroir

- Soit  $X = (A, Q, q_0, Q_F, \delta)$  un automate reconnaissant le langage  $L(X)$ .
- L'automate qui reconnaît le langage  $(L(X))^R$  est reconnu par l'automate  $X^R = (A, Q, Q_F, \{q_0\}, \delta^R)$  tel que :  $\delta^R(q', a) = q$  si  $\delta(q, a) = q'$ .
- En d'autres termes, il suffit juste d'*inverser* les sens des arcs de l'automate et le statut initial/final des états initiaux et finaux

45

### Langages réguliers

Les langages *réguliers* sont les langages générés par des grammaires de type 3 (ou encore grammaires régulières). Ils sont reconnus grâce aux automates à états finis.

Le terme régulier vient du fait que les mots de tels langages possèdent une forme particulière pouvant être décrite par des expressions dites régulières.

46

### Passage de la grammaire vers l'automate

soit  $G = (V, N, S, R)$  une grammaire régulière à droite, si toutes les règles de production sont de la forme :  $A \rightarrow aB$  ou  $A \rightarrow B$  ( $A, B \in N, a \in V \cup \{\varepsilon\}$ ) alors il suffit d'appliquer l'algorithme suivant :

1. Associer un état à chaque non terminal de  $N$ ;
2. L'état initial est associé à l'axiome ;
3. Pour chaque règle de production de la forme  $A \rightarrow \varepsilon$ , l'état  $q_A$  est final ;
4. Pour chaque règle de production de la forme  $A \rightarrow a$  ( $a \in V$ ), alors créer un nouvel état final  $q_f$  et une transition partant de l'état  $q_A$  vers l'état  $q_f$  avec l'entrée  $a$  ;
5. Pour chaque règle  $A \rightarrow aB$  alors créer une transition partant de  $q_A$  vers l'état  $q_B$  en utilisant l'entrée  $a$  ;
6. Pour chaque règle  $A \rightarrow B$  alors créer une  $\varepsilon$ -transition partant de  $q_A$  vers l'état  $q_B$  ;