

# Voca / Déf

## AEL

cour ① du 10/09

Alphabet : - composé de lettres  
- fini / non vide

Mot : - définit sur un alphabet  $\Sigma$   
- composé de lettres de  $\Sigma$

Long. d'un mot :  $|v| \rightarrow$  Longueur (nombre de lettres)

$|v|_x \rightarrow$  Occurrence de  $x$  dans  $v$

$\epsilon$  : le mot vide  $\rightarrow |v|=0$  si  $v=\epsilon$

Concaténation : associative :  $(u.v).w = u.(v.w)$

si  $u = x_1 x_2 \dots x_n$  et  $v = y_1 y_2 \dots y_m$  alors

$u.v = z_1 z_2 \dots z_{n+m}$  où  $z_i = x_i$  pour  $i \leq n$   
et  $z_i = y_{i-n}$  pour  $i > n$

itérat° de concat :

$\forall v^0 = \epsilon$  ;  $v^{i+1} = v.v^i = v^i.v$  pr  $i \geq 0$

facteur :

•  $v$  facteur de  $w$  ssi il existe des mots  $p, s$  t.q. :

$$w = p.v.s$$

• il existe facteur gche (préfixe) et droit (suffixe)

•  $v$  facteur propre si  $v \neq \epsilon$  et  $v \neq w$

Langage - ensemble de mots

- non nécessairement fini/vide

concat de langage:  $L, L' \text{ Pang. sur } X$

associative

$$L \cdot L' = \{ u v \mid u \in L, v \in L' \}$$

itération des Pang. :  $L^0 = \{\epsilon\}$

### Closure de Kleene

- $\bigcup_{i \geq 0} L^i \rightarrow$  déf : l'ensemble des mots que l'on peut assembler avec les mots de  $L$ .
- On note  $L^* = \bigcup_{i \geq 0} L^i$  (notation n'étoile de Kleene)

$$\{a\}^* = \{\epsilon, a, aa, \dots\}$$

n'étoile stricte de  $\tilde{z}$ :

$$L^+ = \bigcup_{i \geq 1} L^i \Rightarrow \text{on retire } \epsilon \text{ et les mots de } L$$

### monoïde

Si  $X$  un alph.  $* X^* = \{\text{alphab. contenant les comb. } X^n\}$

Expr rationnelle: défini induitivement par:

- $\emptyset$ ,  $\epsilon$  et  $x$  (lettres  $x \in X$ ) en sont
- $e_1$  et  $e_2$  en sont alors:
  - $(e_1)$  en est une
  - $e_1 + e_2$  aussi
  - $e_1 \cdot e_2$  aussi
  - $e_1^*$  —
  - $e^+$  équivaut à  $e^* \cdot e$  ( $^+ \neq +$ )

plus prio |  $a *$  désigne  $\{a\}^*$

a.b  $\rightarrow \{a\} \cdot \{b\} = \{ab\}$

prio ↓  $a+b \rightarrow \{a\} \cup \{b\} = \{a, b\}$

union

note TD

- prendre l'habitude de noter ses langages afin de mieux percevoir

Cour ① du 20/09

Soit  $e$  une expr. rat.,  $e$  est associée à un langage  $\mathcal{L}(e)$  par:

- $\mathcal{L}(\emptyset) = \emptyset$ ,  $\mathcal{L}(\epsilon) = \{\epsilon\}$   $\mathcal{L}(x) = \{x\}$   $\forall x \in X$
- $\mathcal{L}(e_1) = \cancel{\mathcal{L}} \mathcal{L}(e_1)$
- $\mathcal{L}(e_1 + e_2) = \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$
- $\mathcal{L}(e_1 \cdot e_2) = \mathcal{L}(e_1) \cdot \mathcal{L}(e_2)$  si  $\not\exists e_1'$  et  $e_2'$

$$\text{tq } e_1 = e' + e'' \text{ ou } e_2 = e' + e''$$

$$\cdot \mathcal{L}(e_1*) = \mathcal{L}(e_1)*$$

$$\text{si } & e' \text{ et } e'' \text{ tq } e_1 = e' + e'' \text{ ou } e_2 = e' \cdot e''$$

Langage rationnel:

- peut être dénoté par une expression rat.
- On appelle RAT la famille de ses lang.

RAT est la petite famille de langage:

- contient  $\emptyset, \Sigma, \alpha \in \Sigma^*$
- closé par  $\cup, \text{concat.}, \text{et } *$

$T \in \text{Langage fini} \subseteq \text{RAT}$

Automate fini déterministe def par

- un alph  $\Sigma$
- un ensemble fini  $Q$  appelé ensemble d'états
- une fonction  $\delta: Q \times \Sigma \rightarrow Q$ : fonction de transition
- un état initial  $q_{ini} \in Q$
- un ss-ensemble  $F \subseteq Q$  d'états dit acceptants / finals / terminaux

Notation:  $\delta(q_d, x) = q_a$  pourra être noté

$$q_d \xrightarrow{x} q_a$$

$q_d$ : état de départ  $q_a$ : état d'arrivée

AEL

cour # du 20/09

$$X = \{a, b\}$$

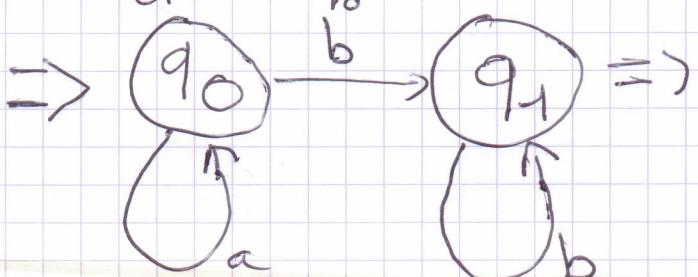
$$Q = \{q_0, q_1\}$$

$$q_0, a \mapsto q_0$$

$$q_0, b \mapsto q_1$$

$$q_1, b \mapsto q_1$$

on le traduit graphiquement

par étatini =  $q_0$  état d'arrivée

$\epsilon \notin L(A)$  :  $q_0$  non acceptant

$b \in L(A)$  : on arrive à  $q_1$ , qui est accepté

$a \notin L(A)$  : \_\_\_\_\_  $q_0$  \_\_\_\_\_ non accept

$ab \in L(A)$  : \_\_\_\_\_  $q_1$  \_\_\_\_\_

$bb \in \text{_____}$  : \_\_\_\_\_

$aa \notin L(A)$  : \_\_\_\_\_  $q_0$  \_\_\_\_\_

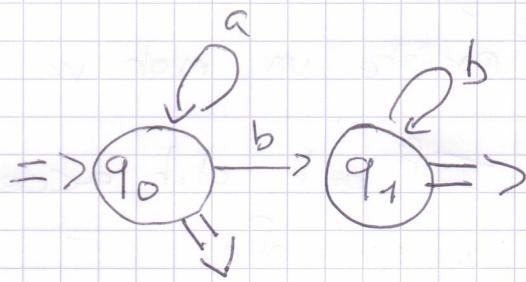
$ba \in L(A)$  : \_\_\_\_\_  $q_1$  mais le mot n'est pas  $q_0$  en entier

En gros, Pour qu'un automate lit un mot, il faut que le lit pas entièrement alors le mot n'est pas accepté. Si le lit en entier alors le mot est accepté.

pensée

On peut traduire  $\equiv A$  par l'expression rationnelle  $(a)* (b)*$

$E/F$	a	b
$\Leftarrow$	oui	$q_0$
$\Leftarrow$		$q_0 \quad q_1$
$\Leftarrow$	oui	$q_1$
		$q_1$



$\hat{S}$  (état, mot gey)

$\hat{S}$  (état,  $\epsilon$ ) = état

$\hat{S}$  (état, lettre) =  $G(\text{état, lettre})$

Langage reconnu :

c'est un langage reconnu par un automate déterministe

$$A = (X, Q, \delta, q_{ini}, F)$$

$$\mathcal{L}(A) = \{ w \in X^* \mid \hat{\delta}(q_{ini}) \in F \}$$

Réfrop

Automate complet : valide

automate déterministe A est complet si sa fonction de transit° est définie  $\forall (q, x) \in Q \times X$

S'il ne l'a pas, un algo simple permet d'en construire un qui reconnaît le même langage.

## automate accessible :

Soit  $A$  un automate. Un état  $q$  est dit accessible si il existe un mot  $w$  tel que  $\hat{\delta}(q_{ini}, w) = q$

$I^*$  est dit accessible si tous ses états sont accessibles

## Etat co-accessible :

Un automate, un état  $q$  est dit co-accessible

$$\exists w \quad t_q \quad \hat{\delta}(q, w) \in F$$

langage reconnu par un automate accessible

## Automate co-accessible :

les états sont co-accessibles

## Automate émondé :

si accessible et co-accessible

## Automate finis non-déterministe :

defini par:

- ensemble non vide d'état initiaux

$$I_{ini} \in 2^Q, I_{ini} \neq \emptyset$$

- fonct<sup>o</sup>  $\delta$ :  $Q \times X \rightarrow 2^Q \mapsto$  fonct de transition

Pour déterminiser :

$\delta$	a	b
$\rightarrow \{1,2\}$	$\{1\}$	$\{1,2,4\}$
$\{1,2,4\}$		

le but est de voir  
les tuples d'état  
afin de faire un  
nouveau automate  
qui sera celui-ci  
déterministe.

De plus, il sera complet  
et complètement accessible  
mais pas minimal

### Stabilité de REC par $\cup, \cdot, *$

- Soit  $L_1 \in \text{REC}$  par un automate  $A_1$
- et  $L_2 \in \text{REC}$  par un automate  $A_2$
- On suppose  $Q_{A_1} \cap Q_{A_2} = \emptyset$  (sinon, on renomme les états)

L'automate non déterministe A :

- $Q_A = Q_{A_1} \cup Q_{A_2}$
- $Init_A = Init_{A_1} \cup Init_{A_2}$
- $F_A = F_{A_1} \cup F_{A_2}$
- $\forall q \in Q_{A_1}, \delta_A(q, x) = \delta_{A_1}(q, x)$
- $\forall q \in Q_{A_2}, \delta_A(q, x) = \delta_{A_2}(q, x)$

reconnait le lang.  $L_1 \cup L_2$

dc  $L_1 \cup L_2 \in \text{REC}$

$P(n)$  { Si  $L$  est rationnel alors  $L$  est REC  
 $n \geq 1$  } Si  $e$  est une exp rationnelle de longueur  $\leq n$   
alors  $\mathcal{L}(e)$  est reconnaissable.

Pour  $P(1) \rightarrow$  longueur 1 :

- $e = \emptyset$  -  $e = \infty$
- $e = \epsilon$

On ne peut avoir une exp rat.  
de longeur 0

$a+b$	3
$a*$	2
$(a)*$	4
$\emptyset$	1
$\epsilon$	1

Démonstration : par récurrence

donc  $\{\mathcal{L}(e)\} = \emptyset$

$\xrightarrow{\text{cas ①}}$   $= \{\epsilon\}$

REC  $\left| \begin{array}{l} P(1) \text{ vrai car} \\ \text{REC} \end{array} \right.$

$\left| \begin{array}{l} \text{pour } n=1, \text{ on reconnaît} \\ \text{que des langages REC} \end{array} \right.$

Supposons  $P(n)$  vrai pour un certain  $n$ :

$e$  de longeur  $n+1$   $\xrightarrow{\text{cas ②}}$

$\bullet \exists e' \text{ tq } e = (e')$   
 $f(e') = n-1$

$\mathcal{L}(e) = \mathcal{L}(e')$  et  $\mathcal{L}(e')$  est REC

Alors  $\mathcal{L}(e)$  REC

$\bullet \exists e_1, e_2 \text{ tq } e = \overbrace{[e_1 + e_2]}^{n+1} \rightarrow \text{cas ③}$

$\mathcal{L}(e_1)$  REC,  $\mathcal{L}(e_2)$  REC

$\mathcal{L}(e) = \mathcal{L}(e_1 \cup e_2) = \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$

Alors  $\mathcal{L}(e)$  est rec.

idem

$\bullet \exists e_1, e_2 \text{ tq } e = e_1 \cdot e_2 \rightarrow \text{cas ④}$

idem

$\bullet \exists e' \text{ tq } e = e'^*$   $\rightarrow \text{cas ⑤}$

Donc notre propriété est vérifiée

$P(n)$   $\forall n \geq 1$  si  $e$  est une exp rat. alors  $\mathcal{L}(e)$  est REC

# AEL

cour ① du 11/10

Résoudre une équation :  $x \in \mathbb{R}$

- $x \in \mathbb{R}$

$$x = ax + b$$

$x$  inconnue  
 $a, b$  connus

On a alors :

$$a=1, b \neq 0 \quad \textcircled{*} \quad \text{aucune solution} \quad 0=b$$

$$a \neq 1$$

$$\textcircled{*} \quad \text{une solution} \quad x = \frac{b}{1-a}$$

$$a=1, b=0 \quad \textcircled{*} \quad \text{plusieurs } \underline{\hspace{1cm}}$$

$$x = x$$

n inconnu n équation:

exemple :  $\begin{cases} ax = a'x + by + c & x, y \text{ inconnus} \\ y = a''x + b'y + c' & a, b, c, a', b', c' \text{ connus} \end{cases}$

On applique cela au langage

lemme d'arden

$$L = A \cdot L \cup B$$

L inconnu  
A et B 2 lang. "communs"

Pour quelques valeurs de A et B avons nous :

- $\textcircled{*}$  aucune solut°
- $\textcircled{*}$  une solut°
- $\textcircled{*}$  plusieurs solut°

- $A \star B$  est une solution

- $A \star B$  est minimal

- si  $E \not\in A$ , alors  $A \star B$  est l'unique solution

$$L = A \star B$$

$$A \cdot L \cup B = A \cdot (A \star B) \cup B = \underbrace{(A \star \cup \underline{\hspace{1cm}})}_{A \star B} \cdot B$$

Si  $L$  est une solution alors  $L$  contient  $A \cup B$

Soit une solut<sup>o</sup>  $\star L = A L \cup B \Rightarrow B \subseteq L$

$$A \cdot B \subseteq A \cdot L \subseteq L$$

Dc  $A \cdot B \subseteq L$  et

$$A \cdot (A \cdot B) \subseteq A \cdot L \subseteq L$$

$$\text{dc } AA \cdot B = A^2 \cdot B \subseteq L$$

par ~~ester~~ récurrence :  $A^i \cdot B \subseteq L \forall i \geq 0$

Si  $\epsilon \notin A$  alors  $A \cdot B$  est l'unique solut<sup>o</sup>

Soit  $L$  une solut<sup>o</sup>  $A \cdot B \subseteq L$  faut

Reste  $L \subseteq A \cdot B$

démonstration par Recurrence

P(n) Soit  $v$  un mot de  $L$  tq  $|v| \leq n$

\* P(0)  $v = \epsilon$   $\epsilon \in L = A L \cup B$   $\epsilon \in A L$

tout  $\epsilon \in L$  mots de  $A$  str de  $p_0 \geq 1$

$$|AL| \geq 1$$

dc  $\epsilon \notin AL$

$\epsilon \in B \rightarrow v \in A \cdot B$

donc P(0) vraie

\* Supposons P(n) vraie tq  $|v| = n+1$

si  $v \in A L \cup B$  on a soit

①  $v \in B$  ds  $v \in A \cdot B$

②  $v \in A L \mid \exists v_1 \in A \text{ tq } v = v_1 \cdot u_2$   
 $\exists u_2 \in L$

$|v_1| \geq 1$  et dc  $|v_2| \leq n$

$\epsilon \notin A$

P(n) supp vraie dc  $u_2 \in A \cdot B$

AEL

Cov  $\hat{Q}_{\text{dLHf}}$

$P(n)$  supp wai

$$v_2 \in A * B$$

$$v \in A. A * B \subseteq A * B \quad e_A \in A * B$$

donc  $P(n+1)$  est clairement vraie

dc le principe de récurrence nous donne

$P(n)$  vrai si  $n$  de:

$$\forall v \in L \quad \underline{v} \in A \times B$$

$$L \subseteq A * B$$

Incomes  $L_1$   $L_2$

Languages  $\Rightarrow$   $A_{11} A_{12} A_{21} A_{22} B_1 B_2$

$$(1) \quad \sum L_1 = A_{T_1} L_1 \cup A_{\frac{T_1}{2}} L_2 \cup B_1$$

$$(2) \quad L_2 = A_{z_1} L_1 \cup A_{z_2} L_2 \cup B_2$$

$$\text{Supp } \varepsilon \notin A_i, \forall i$$

$$L_2 = A_{2_2} \cup (A_{2_2} \cap B_2)$$

dc par le permis d'arden:

$$d_C L_2 = A_{22} * (A_{21} L_1 \cup B_2)$$

$$L_2 = A_{2_2} * A_{2_1} L_1 \cup A_{2_2} * B_{2_2} \quad (f2)$$

$$(1) \Rightarrow L_1 = A_{11}L_1 \cup A_{12} \left( \overbrace{A_{22} * A_{21}L_1 \cup A_{22} * B_2}^{\text{Redacted}} \right)$$

$$L_1 = \left( A_{11} \cup A_{12} A_{22} * A_{21} \right) L_1 \cup \left( A_{12} A_{22} * B_2 \cup B_1 \right)$$

$L_1 = (A) * \circ (B) \rightarrow$  bin me d'orden

## Grammaire algébrique

- alph finis
- un alphabet "terminal"  $\Sigma \neq \emptyset$  minuscule
- définit des langages sur  $\Sigma^*$
- un alph "non-terminal" (des variables)  $V \neq \emptyset$  majuscule
- une variable est appelée Axiome
- un ensemble de règles

$$\Sigma = \{a, e, b\}$$

les règles:

$$X \rightarrow aXa$$

$$X \rightarrow bXb$$

$$X \rightarrow cY$$

$$V = \{X, Y\}$$

Axiome X

$$Y \rightarrow cY$$

$$Y \rightarrow E$$

## Dérivation

$$X \rightarrow bXb \rightarrow bacYab \rightarrow bacYab \rightarrow bacab$$

Dérivation à une longueur: le nbr de "flèches" qui apparaissent  
ici on a 4.

On peut rajouter la Pg sur la flèche

$$X \xrightarrow{1} bXb \Rightarrow X \rightarrow bXb$$

$$X \xrightarrow{4} bacab$$

$$bXb \xrightarrow{\text{nbr q'c de dérivat' on précise}} bacYab$$

Langage engendré par  $G$ :

$$\mathcal{L}(G) = \{ w \in \Sigma^* \mid \text{Axione } \xrightarrow{*} w \}$$

$$\epsilon \in \mathcal{L}(G)$$

$$\epsilon \notin \mathcal{L}(G)$$

$$bacab \in \mathcal{L}(G)$$

$$aca \in \mathcal{L}(G)$$