

# Fichez revisions

mercredi 24 octobre 2018 11:23

## Définition (Alphabet)

Un **alphabet** est un ensemble fini dont les éléments sont appelés **symboles**.

## Définition (Mot)

Un **mot de longueur**  $n \in \mathbb{N}$  est une *application* de  $\{0, \dots, n - 1\}$  vers  $\Sigma$ .

Remarque C'est une application, et non pas une fonction quelconque.

### Longueur d'un mot

On note  $|u|$  la *longueur* du mot  $u$ .

### Définition (Mot vide)

- Le **mot vide** est la fonction de l'ensemble vide ( $\emptyset$ ) vers  $\Sigma$ .
- Le mot vide est noté  $\epsilon_\Sigma$  ou  $\epsilon$ , lorsque le contexte est clair.

L'ensemble de tous les mots sur l'alphabet  $\Sigma$  est noté  $\Sigma^*$ .

### Définition (Langage)

Un langage sur l'alphabet  $\Sigma$  est un ensemble de mots sur  $\Sigma$  ;

- c-à-d un sous-ensemble de  $\Sigma^*$  ;
- c-à-d un élément de l'ensemble  $\mathcal{P}(\Sigma^*)$ , où  $\mathcal{P}(\Sigma^*)$  dénote l'ensemble des sous-ensembles de  $\Sigma^*$ .

### Exemple (Langage)

- $\emptyset$  est un langage sur  $\Sigma$  : langage vide,
- $\Sigma^*$  est un langage sur  $\Sigma$  : langage universel,

### Définition (Concaténation)

- La concaténation est une application de  $\Sigma^* \times \Sigma^*$  vers  $\Sigma^*$ ,
- La concaténation de deux mots  $u$  et  $v$  dans  $\Sigma^*$  est le mot  $u \cdot v : \{0, \dots, |u| + |v| - 1\} \rightarrow \Sigma$  tel que :

$$(u \cdot v)(i) \stackrel{\text{def}}{=} \begin{cases} u(i) & \text{si } i \in \{0, \dots, |u| - 1\} \\ v(i - |u|) & \text{si } i \in \{|u|, \dots, |u| + |v| - 1\} \end{cases}$$

## Définition (Préfixe, suffixe et facteur)

Considérons deux mots  $u$  et  $v$  sur un alphabet  $\Sigma$ .

- $v$  est un **préfixe** de  $u$ , noté  $v \preceq u$ , s'il existe un mot  $v'$  (sur  $\Sigma$ ) tel que  $v \cdot v' = u$ .
- $v$  est un **suffixe** de  $u$ , s'il existe un mot  $v'$  (sur  $\Sigma$ ) tel que  $v' \cdot v = u$ .
- $v$  est un **facteur** de  $u$ , s'il existe deux mots  $v'$  et  $v''$  (sur  $\Sigma$ ) tels que  $v' \cdot v \cdot v'' = u$ .
- $v$  est une **extension** de  $u$ , si  $u$  est un préfixe de  $v$ .

## Définition (Concaténation de langages)

$$\cdot : \mathcal{P}(\Sigma^*) \times \mathcal{P}(\Sigma^*) \rightarrow \mathcal{P}(\Sigma^*)$$
$$L_1 \cdot L_2 \stackrel{\text{def}}{=} \{u_1 \cdot u_2 \mid u_1 \in L_1 \wedge u_2 \in L_2\}$$

Remarque Pas de commutativité :

$$L_1 \cdot L_2 \neq L_2 \cdot L_1 \quad (\text{en général})$$

□

## Exemple (Concaténation de langages)

Considérons les langages  $\{a, aa\}$  et  $\{b, bb\}$ .

$$\{a, aa\} \cdot \{b, bb\} = \{ab, abb, aab, aabb\}$$

## Exemple (Concaténation de langages)

Soit un langage  $L$  sur un alphabet  $\Sigma$  :

- $L \cdot \emptyset = \emptyset (= \{u_1 \cdot u_2 \mid u_1 \in L \wedge u_2 \in \emptyset\})$ ,
- si  $L \neq \Sigma^*$  et  $\epsilon \notin L$ , alors  $L \cdot \Sigma^* \subset \Sigma^*$ ,
- Si  $\epsilon \in L$  alors  $L \cdot \Sigma^* = \Sigma^*$ ,
- $L \cdot \{\epsilon\} = \{\epsilon\} \cdot L = L$   
 $(= \{u_1 \cdot u_2 \mid u_1 \in L \wedge u_2 \in \{\epsilon\}\})$ .

## Définition (Fermeture par préfixe et par suffixe)

- La fermeture par préfixe de  $L$ , noté  $\text{Pref}(L)$ , est le langage formé par l'ensemble préfixes des mots de  $L$ , défini comme :

$$\text{Pref}(L) = \{w \in \Sigma^* \mid \exists w' \in L, \exists w'' \in \Sigma^* : w' = w \cdot w''\}$$

## Chapitre 2

### Définition (Automate à états fini déterministe)

Un **automate à états fini déterministe** (abrégé AEFD) est donné par un 5-tuple

$$(Q, \Sigma, q_{\text{init}}, \delta, F)$$

tel que :

- $Q$  est un ensemble non-vide dont les éléments sont appelés **états** ;
- $\Sigma$  est l'alphabet de l'automate ;
- $q_{\text{init}} \in Q$  est l'**état initial** ;
- $\delta : Q \times \Sigma \rightarrow Q$  est la **fonction de transition** de l'automate ; elle peut être partielle ;
- $F \subseteq Q$  est l'ensemble des états **états accepteurs (terminaux)**.

Un AEFD est dit **complet**, si sa fonction de transition est *totale*.

## Définition (Relation de dérivation)

La relation  $\rightarrow$  de **dérivation** entre configurations est définie comme suit :

$$\forall q \in Q, \forall a \in \Sigma, \forall u \in \Sigma^* : (q, a \cdot u) \rightarrow (q', u) \text{ ssi } \delta(q, a) = q'.$$

## Définition (Acceptation d'un mot par un automate)

Un mot  $u \in \Sigma^*$  est **accepté** par  $A$ , s'il existe une exécution de  $u$  sur  $A$

$$(q_0, u_0) \cdots (q_n, u_n)$$

de  $A$  telle que

- $u_0 = u,$
- $u_n = \epsilon,$
- $q_n \in F.$

## Définition (Langage reconnu par un automate)

Le **langage reconnu par  $A$** , qu'on note par  $L(A)$ , est l'ensemble

$$\{u \in \Sigma^* \mid u \text{ est accepté par } A\}.$$

## Définition (Fonction de transition étendue aux mots)

À partir de  $\delta$ , on définit la **fonction de transition étendue aux mots**  $\delta^*$  :

$$\forall w \in \Sigma^*, \forall q \in Q : w = a_1 \cdot a_2 \cdots a_n \Rightarrow \delta^*(q, w) = \delta\left(\dots \delta\left(\delta(q, a_1), a_2\right) \dots, a_n\right).$$

En utilisant la définition inductive des mots.

## Définition (Fonction de transition étendue aux mots - définition inductive)

À partir de  $\delta$ , on définit la **fonction de transition étendue aux mots**  $\delta^*$  :

- $\delta^*(q, \epsilon)$ , pour tout état  $q \in Q$ ,
- $\delta^*(q, w \cdot a) = \delta(\delta^*(q, w), a)$ , pour tout état  $q \in Q$ , mot  $w \in \Sigma^*$ ,  $a \in \Sigma$ .

## Définition (Complétion d'automates)

L'automate complété de  $A$  est  $C(A) = (Q \cup \{q_p\}, \Sigma, q_{\text{init}}, C(\delta), F)$  tel que

- $q_p \notin Q$  et
- $C(\delta) : Q \cup \{q_p\} \times \Sigma \rightarrow Q \cup \{q_p\}$  est une *application* définie par :

$$C(\delta)(q, a) \stackrel{\text{def}}{=} \begin{cases} \delta(q, a) & \text{pour tout } (q, a) \in \text{dom}(\delta) \\ q_p & \text{sinon} \end{cases}$$

↔ Ajouter un état "puis"

Correction de la procédure de complétion

$$L(A) = L(C(A))$$

## Définition (Complémentation d'un AEDF complet)



Le complémentaire de  $A$  est l'automate  $A^c = (Q, \Sigma, q_{\text{init}}, \delta, Q \setminus F)$ .

## Définition (Produit d'automates)

L'automate produit de  $A$  et de  $B$  est  $A \times B = (Q, \Sigma, q_{\text{init}}, \delta, F)$  où :

- $Q = Q^A \times Q^B$
- $q_{\text{init}} = (q_{\text{init}}^A, q_{\text{init}}^B)$
- $\delta : (Q^A \times Q^B) \times \Sigma \rightarrow (Q^A \times Q^B)$  est telle que

$$\delta((q^A, q^B), a) = (\delta^A(q^A, a), \delta^B(q^B, a))$$

- $F = F^A \times F^B$ .

## Théorème

Soient  $A = (Q^A, \Sigma, q_0^A, \delta^A, F^A)$  et  $B = (Q^B, \Sigma, q_0^B, \delta^B, F^B)$  deux AEFDS.

- $L(A \times B) = L(A) \cap L(B)$ .
- La classe EF des langages à états est fermée par intersection.

## Successseurs d'un état

L'ensemble des états **successeurs** d'un état  $q \in Q$ , selon la fonction de transition  $\delta$ , est :

$$\text{Succ}(q) = \{ q' \in Q \mid \exists a \in \Sigma : \delta(q, a) = q' \}$$

En itérant sur  $\text{Succ}(q)$ , nous obtenons les états selon la priorité donnée par les symboles.

## Prédécesseurs d'un état

L'ensemble des états **prédécesseurs** d'un état  $q \in Q$ , selon la fonction de transition  $\delta$ , est :

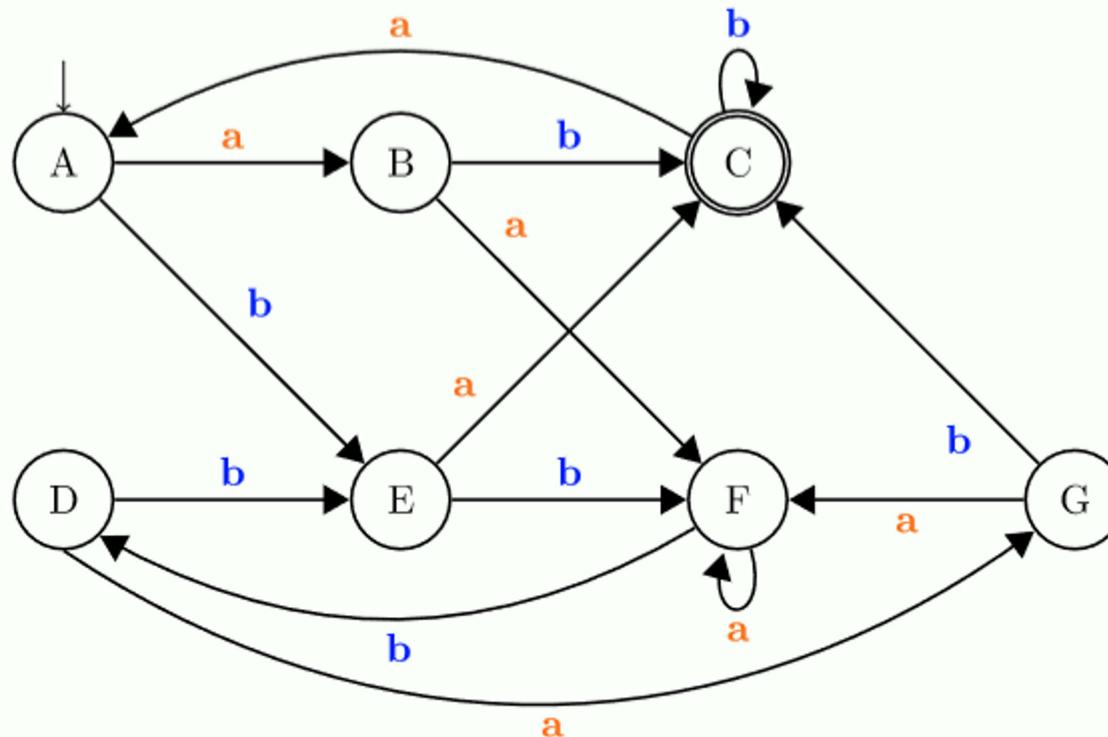
$$\text{Pré}(q) = \{ q' \in Q \mid \exists a \in \Sigma : \delta(q', a) = q \}$$

(En itérant sur  $\text{Pré}(q)$ , nous obtenons les états selon la priorité donnée par les symboles.)

Remarque Un état  $q$  est successeur d'un état  $q'$  ssi  $q'$  est un prédecesseur de  $q$



## Exemple (Parcours en profondeur vs largeur d'abord)



▫ Parcours en prof. d'abord

▫ Parcours en largeur d'abord

Priorité de **a** sur **b**

- prof. d'abord : A, B, F, D, G, C, E
- largeur d'abord : A, B, E, F, C, D, G

Priorité de **b** sur **a**

- prof. d'abord : A, E, F, D, G, C, B
- largeur d'abord : A, E, B, F, C, D, G

## Définition (Cycle - deux définitions équivalentes)

- **séquence (non-vide) de transitions consécutives** (l'état d'arrivée d'une transition est l'état de départ de la prochaine transition) t.q. le 1<sup>er</sup> et le dernier états soient identiques.
- automate avec une **transition arrière** :
  - soit une transition d'un état sur lui même,
  - soit une transition d'un état vers l'un de ces ancêtres dans l'arbre produit par le parcours en profondeur d'abord.

## Accessibilité et co-accessibilité : définition informelle

Propriétés s'appliquant aux états d'un AEFD :

- état accessible : peut être atteint à en suivant la fonction de transition ;
- état co-accessible : mène à un état accepteur en suivant la fonction de transition.

## Définition (Accessibilité d'un état dans un AEFD)

$q \in Q$  est accessible s'il existe un mot  $u \in \Sigma^*$  tel que  $\delta^*(q_{\text{init}}, u) = q$ .

Un automate accessible et co-accessible est dit **émondé**.

## Définition (Problème de décision)

- Question que l'on peut *exprimer mathématiquement* (formellement).
- Question à un certain nombre de paramètres que l'on souhaite pouvoir passer à un programme informatique.
- La réponse à la question est *oui ou non*.

Deux sortes de problèmes de décision existent :

- **problèmes décidables** : on peut trouver un algorithme ou un programme qui sait répondre à la question (avec une mémoire et un temps non-borné).
- **problèmes indécidables** : on ne peut pas trouver un algorithme ou un programme qui sait répondre à la question (de manière générale).

## Chapitre 5

### Théorème : à propos de la relation de distinguabilité

La relation de distinguabilité  $\neq$  entre états de  $Q$  est :

- anti-réflexive :  $\forall q \in Q : \neg(q \neq q)$ ,
- symétrique :  $\forall p, q \in Q : p \neq q \implies q \neq p$ .

## Définition (Distinguabilité à $k$ pas)

Pour chaque  $k \in \mathbb{N}$ , on introduit la relation  $\neq_k$  sur  $Q$  :

①  $q \neq_0 q'$  ssi  $q \in F \Leftrightarrow q' \in F$ .

② Pour  $k \in \mathbb{N}$ ,  $p \neq_{k+1} q$  ssi

$$(p \neq_k q) \vee (\exists a \in \Sigma : \delta(p, a) \neq_k \delta(q, a)).$$

## Lemme

Pour tout  $k \in \mathbb{N}$ ,  $q \neq_k q'$  ssi

$$\exists u \in \Sigma^* : |u| \leq k \wedge (\delta^*(q, u) \in F \Leftrightarrow \delta^*(q', u) \in F).$$

## Définition (Relation d'équivalence entre états)

La relation d'équivalence  $\approx$  entre états sur  $Q$  est définie par :

$$\forall p, q \in Q : p \approx q \text{ ssi } \forall u \in \Sigma^* : (\delta^*(p, u) \in F \Leftrightarrow \delta^*(q, u) \in F)$$

## Définition (Minimisé d'un automate)

Le minimisé de  $A$  est l'automate  $A_{/\approx} = (Q_{/\approx}, \Sigma, [q_{\text{init}}], \delta_{/\approx}, F_{/\approx})$  où :

- $\delta_{/\approx}$  est l'application de transition t.q. :

$$\begin{aligned}\delta_{/\approx} & : Q_{/\approx} \times \Sigma \rightarrow Q_{/\approx} \\ \delta_{/\approx}([q], a) & \stackrel{\text{def}}{=} [\delta(q, a)]\end{aligned}$$

- $F_{/\approx} = \{[q] \mid q \in F\}$ .