



## INF 302 : LANGAGES & AUTOMATES

### Chapitre 8 : Automates à états finis non-déterministes avec $\epsilon$ -transitions

Yliès Falcone

[yliès.falcone@univ-grenoble-alpes.fr](mailto:yliès.falcone@univ-grenoble-alpes.fr) — [www.yliès.fr](http://www.yliès.fr)

Univ. Grenoble-Alpes, Inria

Laboratoire d'Informatique de Grenoble - [www.liglab.fr](http://www.liglab.fr)  
Équipe de recherche LIG-Inria, CORSE - [team.inria.fr/corse/](http://team.inria.fr/corse/)

Année Académique 2021 - 2022

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

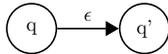
- 1 Motivations
  - Utilisation pratique des automates avec  $\epsilon$ -transitions
  - Fermeture de Kleene d'un langage
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
  - Utilisation pratique des automates avec  $\epsilon$ -transitions
  - Fermeture de Kleene d'un langage
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### Utilisation des $\epsilon$ -transitions

On autorise  $\epsilon$  comme étiquette des transitions ; on parle d' $\epsilon$ -transition.



Un mot est accepté si on peut lire le mot jusqu'à un état accepteur :  
 $\hookrightarrow$  les  $\epsilon$ -transitions ne "consomment" pas de symbole pendant la lecture du mot d'entrée.

Propriété sous-jacente :  
 $\forall w \in \Sigma^* : w \cdot \epsilon = \epsilon \cdot w = w$ .  
 ( $\epsilon$  est l'élément neutre de la concaténation entre mots)

### Utilisation des $\epsilon$ -transitions

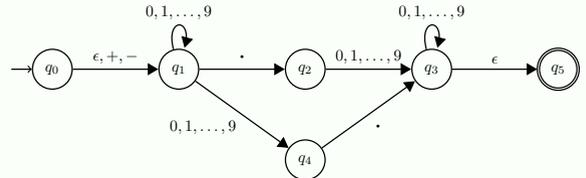
Nombres décimaux

#### Exemple (Nombres décimaux)

Un nombre écrit en notation décimale consiste en :

- un signe + ou - optionnel,
- un mot de numéros  $0, 1, 2, \dots, 9$ ,
- un point pour marquer la décimale,
- un mot de numéros  $0, 1, 2, \dots, 9$ .

L'un des deux mots de numéros peut être vide, mais ils ne peuvent pas être tous les deux vides.



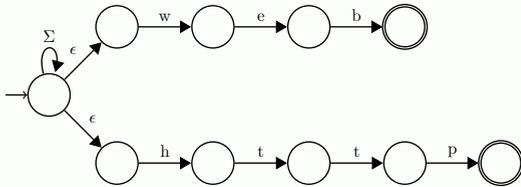
L'utilisation d' $\epsilon$ -transitions facilite la définition de l'automate (notamment concernant les choix).

### Utilisation des $\epsilon$ -transitions

Reconnaissance de mots clés et transformation d'automates

#### Exemple (Reconnaissance de mots clés - compositionnalité)

Reconnaissance de deux mots clés :  
 web et http



L'ajout d'un mot clé se fait de *manière compositionnelle* :

- écrire un automate reconnaissant uniquement ce mot clé,
- ajouter une  $\epsilon$ -transition depuis l'état initial de l'automate général vers l'état initial de l'automate reconnaissant,
- l'unique état initial est celui de l'automate général.

Nous verrons en TD que les  $\epsilon$ -transitions facilitent la transformation d'automates, p. ex. transformer un automate pour garder que certains préfixes/suffixes du langage.

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
  - Utilisation pratique des automates avec  $\epsilon$ -transitions
  - Fermeture de Kleene d'un langage
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### Fermeture de Kleene d'un langage

Définition - vision langage

Soit  $L$  un langage (quelconque, pas forcément à états) sur  $\Sigma$ .

Définition (Fermeture de Kleene)

La Fermeture de Kleene de  $L$ , notée  $L^*$ , est l'ensemble défini inductivement comme le plus petit ensemble généré par les deux règles suivantes :

- $\epsilon \in L^*$ , et
- si  $u \in L, v \in L^*$ , alors  $u \cdot v \in L^*$ ,
- (de manière équivalente à la précédente règle : si  $u \in L^*, v \in L$ , alors  $u \cdot v \in L^*$ ).

Remarque Autrement dit, la fermeture de Kleene de  $L$  est l'ensemble des mots formés par un nombre fini de concaténations de mots de  $L$  :

$$L^* = \{\epsilon\} \cup \{a_0 \dots a_n \mid n \in \mathbb{N}, \forall i \leq n : a_i \in L\}$$

Exemple (Fermeture de Kleene)

- $L_1 = \{b \cdot a, c \cdot d\}$
- $L_1^* = \{\epsilon, ba, cd, baba, bacd, cdcd, cdba, \dots\}$
- $L_2 = \{a \cdot a, b\}$
- $L_2^*$  est l'ensemble des mots contenant un nombre pair de  $a$  et des  $b$  de manière non contrainte.

Pour un langage constitué de mots de longueur 1 (qui peut être vu comme un alphabet), la fermeture de Kleene de ce langage est le langage universel (sur cet alphabet).

### Fermeture de Kleene d'un langage

Vision automate

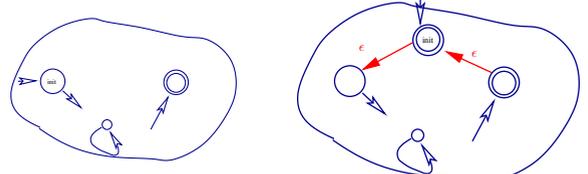
On veut montrer que si  $L$  est dans EF alors  $L^*$  est dans EF :

$$\forall L \subseteq \Sigma^* : L \in EF \implies L^* \in EF$$

Étant donné un automate qui reconnaît  $L$ .

Peut-on construire un automate qui reconnaît  $L^*$  ?

On obtient facilement un automate pour  $L^*$  à partir de n'importe quel automate de  $L$  en utilisant les  $\epsilon$ -transitions :



Automate pour  $L$

Automate pour  $L^*$

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
  - Définition
  - Langage accepté
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
  - Définition
  - Langage accepté
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

### ANDEF avec $\epsilon$ -transitions

Soit  $\Sigma$  un alphabet où le symbole  $\epsilon \notin \Sigma$ .

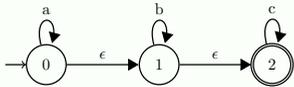
#### Définition (ANDEF avec $\epsilon$ -transitions)

Un *automate non-déterministe avec  $\epsilon$ -transitions* ( $\epsilon$ -ANDEF) est donné par un quintuplet  $(Q, \Sigma, q_0, \Delta, F)$  où

- $Q$  est un ensemble fini d'états,
- $\Sigma$  est l'alphabet de l'automate,
- $q_0 \in Q$  est l'état initial,
- $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$  est la relation de transition,
- $F \subseteq Q$  est l'ensemble des états terminaux/finaux.

#### Exemple (ANDEF avec $\epsilon$ -transitions)

Soit  $\Sigma = \{a, b, c\}$ .



### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

#### Motivations

- Automates à états finis non-déterministes avec  $\epsilon$ -transitions
  - Définition
  - Langage accepté

#### Élimination des $\epsilon$ -transitions

#### Retour sur la fermeture de la classe des langages à états

#### Résumé

### Configuration

Soit  $A = (Q, \Sigma, q_0, \Delta, F)$  un  $\epsilon$ -ANDEF.

#### Définition (Configuration)

Une *configuration* de l'automate  $A$  est un couple  $(q, u)$  où  $q \in Q$  et  $u \in \Sigma^*$ .

#### Définition (Relation de dérivation (entre configurations))

On définit la relation  $\rightarrow_{\Delta}$  de *dérivation* entre configurations :

$$(q, a \cdot u) \rightarrow_{\Delta} (q', u') \quad \text{ssi} \quad ((q, a, q') \in \Delta \text{ et } u' = u) \quad \text{ou} \quad (a \cdot u = u' \text{ et } (q, \epsilon, q') \in \Delta)$$

#### Notation

- On note  $q \xrightarrow{a_1 \dots a_n}^*_{\Delta} q'$  lorsqu'ils existent  $q_1, \dots, q_{n-1}$  tels que :

$$(q, a_1, q_1) \in \Delta, (q_1, a_2, q_2) \in \Delta, \dots, (q_{n-1}, a_n, q') \in \Delta.$$

- On note  $q \xrightarrow{*}_{\Delta} q'$  lorsqu'ils existent  $a_1, \dots, a_n$  tels que  $q \xrightarrow{a_1 \dots a_n}^*_{\Delta} q'$ .

### Exécution

#### Définition (Exécution)

Une *exécution de l'automate  $A$*  est une séquence de configurations  $(q_0, u_0) \dots (q_n, u_n)$  telle que

$$(q_i, u_i) \rightarrow_{\Delta} (q_{i+1}, u_{i+1}), \text{ pour } i = 0, \dots, n - 1$$

Les notions

- d'acceptation d'un mot, et
- de langage reconnu

sont définies comme dans le cas des ANDEF mutatis mutandis.

Plan Chap. 8 - Automates à états finis non-déterministes avec  $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
  - Traduction vers ANDEF
  - Traduction (directe) vers ADEF
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

Plan Chap. 8 - Automates à états finis non-déterministes avec  $\epsilon$ -transitions

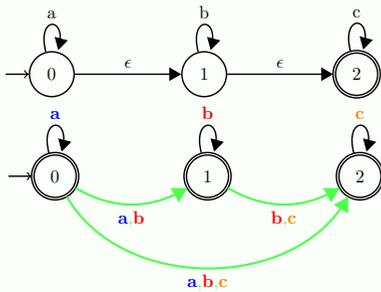
- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
  - Traduction vers ANDEF
  - Traduction (directe) vers ADEF
- 4 Retour sur la fermeture de la classe des langages à états
- 5 Résumé

Élimination des  $\epsilon$ -transitions

L'idée sur un exemple

Exemple (ANDEF avec  $\epsilon$ -transitions)

Soit  $\Sigma = \{a, b, c\}$



Élimination des  $\epsilon$ -transitions : traduction vers ANDEF

Définition

Soit  $A = (Q, \Sigma, q_0, \Delta, F)$  un  $\epsilon$ -ANDEF

Définition (Élimination des  $\epsilon$ -transitions)

On construit un ANDEF

$$\epsilon\ell(A) = (Q, \Sigma, q_0, \epsilon\ell(\Delta), \epsilon\ell(F))$$

qui reconnaît  $L(A)$  tel que :

- La relation de transition  $\epsilon\ell(\Delta)$  est définie par :  $(q, a, q') \in \epsilon\ell(\Delta)$  ssi ils existent  $q_1, q_2 \in Q$  tels que :
  - 1  $q \xrightarrow{\epsilon}^* q_1$
  - 2  $(q_1, a, q_2) \in \Delta$
  - 3  $q_2 \xrightarrow{\epsilon}^* q'$
- L'ensemble des états accepteurs  $\epsilon\ell(F)$  est défini par :

$$\epsilon\ell(F) = \{q \in Q \mid \exists q' \in F : q \xrightarrow{\epsilon}^* q'\}$$

### Correction de la procédure d'élimination des $\epsilon$ -transitions

Soit  $A = (Q, \Sigma, q_0, \Delta, F)$  un  $\epsilon$ -ANDEF.

**Théorème : Correction de la procédure d'élimination des  $\epsilon$ -transitions**

$$L(A) = L(\epsilon\ell(A)).$$

**Preuve (par induction)**

Pour tout  $u, u' \in \Sigma^*$  (et pour tout  $q, q' \in Q$ ),

$$(q, u) \xrightarrow{\epsilon\ell(\Delta)}^* (q', u') \text{ si et seulement si } (q, u) \xrightarrow{\Delta}^* (q', u').$$

- $\epsilon \in L(A)$  si et seulement si  $\epsilon \in L(\epsilon\ell(A))$

- Soit  $u \in \Sigma^*$ ,

Supposons que pour tout  $u' \in \Sigma^*$ ,

$$(q, u) \xrightarrow{\Delta}^* (q', u') \text{ si et seulement si } (q, u) \xrightarrow{\epsilon\ell(\Delta)}^* (q', u')$$

Soit  $a \in \Sigma$ , il faut montrer que pour tout  $u' \in \Sigma^*$

$$(q, u \cdot a) \xrightarrow{\Delta}^* (q', u') \text{ si et seulement si } (q, u \cdot a) \xrightarrow{\epsilon\ell(\Delta)}^* (q', u')$$

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

1 Motivations

2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions

3 Élimination des  $\epsilon$ -transitions

- Traduction vers ANDEF

- Traduction (directe) vers ADEF

4 Retour sur la fermeture de la classe des langages à états

5 Résumé

### Fermeture par $\epsilon$ ( $\epsilon$ -fermeture)

Définition

L' $\epsilon$ -fermeture d'un état  $q$  consiste à regrouper tous les états qu'on peut atteindre en suivant toutes les transitions sortantes de  $q$  et étiquetées par  $\epsilon$

**Définition ( $\epsilon$ -Fermeture d'un état)**

Soit  $q \in Q$  un état, on définit  $ECLOSE(q)$  de façon récursive :

- *Case de base* :  $q \in ECLOSE(q)$

- *Induction* : Si  $p \in ECLOSE(q)$  et s'il existe une transition de  $p$  vers  $r \in Q$  étiquetée par  $\epsilon$ , alors  $r \in ECLOSE(q)$

De manière équivalente :  $ECLOSE(q) = \delta^*(q, \epsilon)$

**Définition ( $\epsilon$ -Fermeture d'un ensemble d'états)**

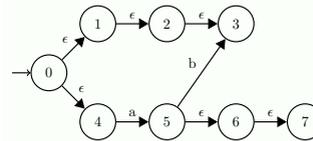
Pour  $S \subseteq Q$  :

$$ECLOSE(S) = \bigcup_{q \in S} ECLOSE(q)$$

### Fermeture par $\epsilon$ ( $\epsilon$ -fermeture)

Exemples

**Exemple ( $\epsilon$ -Fermeture)**



$\epsilon$ -fermeture d'états :

- $ECLOSE(0) = \{0, 1, 2, 3, 4\}$

- $ECLOSE(3) = \{3\}$

- $ECLOSE(5) = \{5, 6, 7\}$

$\epsilon$ -fermeture d'ensembles d'états :

- $ECLOSE(\{0, 3\}) = \{0, 1, 2, 3, 4\}$

- $ECLOSE(\{3, 5\}) = \{3, 5, 6, 7\}$

### Relation de transition étendue

#### Définition

On définit la relation de transition étendue  $\hat{\delta}$  qui permet de lire en entrée des symboles de l'alphabet et  $\epsilon$ ;  $\epsilon$  est vu comme un symbole ne consommant pas de symbole d'entrée.

Intuitivement,  $\hat{\delta}(q, w)$  est l'ensemble d'états atteints en suivant un chemin dont les étiquettes concaténées forment  $w$  (et  $\epsilon$  ne "contribue" pas à  $w$ ).

#### Définition (Relation de transition étendue)

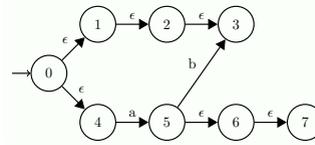
Étant donné  $q \in Q$  et  $w \in (\Sigma \cup \{\epsilon\})^*$  :

- Cas de base :  $\hat{\delta}(q, \epsilon) = ECLOSE(q)$
- Induction : Pour  $w = x \cdot a$ , avec  $a \in \Sigma$ ,  $\hat{\delta}(q, x \cdot a)$  est défini par :
  - soit  $\{p_1, p_2, \dots, p_k\} = \hat{\delta}(q, x)$ ,
  - soit  $\{r_1, r_2, \dots, r_m\} = \bigcup_{i=1}^k \delta(p_i, a)$ ,
  - alors  $\hat{\delta}(q, w) = ECLOSE(\{r_1, r_2, \dots, r_m\})$ .

### Relation de transition étendue

#### Exemple

#### Exemple (Relation de transition étendue)



- $\hat{\delta}(0, \epsilon) = ECLOSE(0) = \{0, 1, 2, 3, 4\}$
- $\hat{\delta}(0, a) = \{5, 6, 7\}$
- $\hat{\delta}(0, b) = \emptyset$
- $\hat{\delta}(0, ab) = \{3\}$

### Élimination des $\epsilon$ -transitions et déterminisation "à la volée"

Traduction vers ADEF - définition du déterminisé

Soit  $A = (Q, \Sigma, q_0, \Delta, F)$  un  $\epsilon$ -ANDEF

#### Définition (Déterminisation et élimination des $\epsilon$ -transitions, à la volée)

Le déterminisé de  $A$  est l'ADEF

$$(Q_D, \Sigma, q_D, \delta, F_D)$$

tel que :

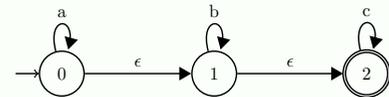
- $Q_D = \mathcal{P}(Q)$
- $q_D = ECLOSE(q_0)$
- $\delta$  est définie comme suit : pour tout  $S \in Q_D, a \in \Sigma$  :
  - soit  $\{p_1, p_2, \dots, p_k\} = S$ ,
  - soit  $\{r_1, r_2, \dots, r_m\} = \bigcup_{i=1}^k \Delta(p_i, a)$ ,
  - alors  $\delta(S, a) = ECLOSE(\{r_1, r_2, \dots, r_m\})$ ,
- $F_D = \{S \in \mathcal{P}(Q) \mid S \cap F \neq \emptyset\}$ .

Remarque Chaque état de l'automate déterminisé (atteint avec  $\delta$ ) correspond à un ensemble d'états de l'automate non-déterministe avec  $\epsilon$ -transitions qui est  $\epsilon$ -fermé.  $\square$

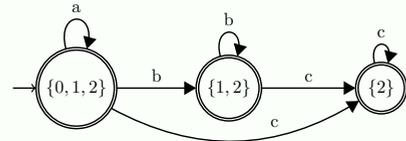
### Élimination des $\epsilon$ -transitions et déterminisation "à la volée"

Traduction vers ADEF : exemple

#### Exemple (Élimination des $\epsilon$ -transitions - Traduction vers ADEF)



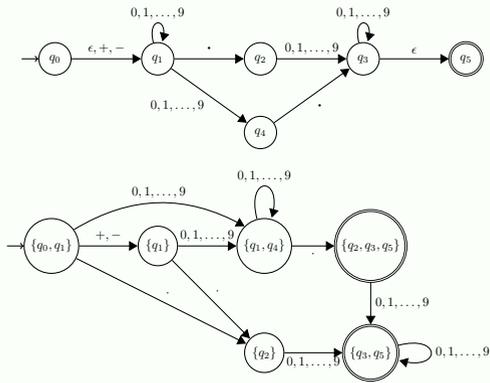
On combine élimination des  $\epsilon$ -transitions et déterminisation. On fait les deux opérations "à la volée".



## Élimination des $\epsilon$ -transitions et déterminisation "à la volée"

Traduction vers ADEF : un autre exemple

### Exemple (Élimination des $\epsilon$ -transitions - Traduction vers ADEF)



## Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

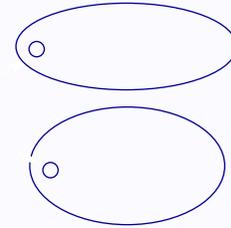
- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
  - Fermeture par union et concaténation
  - Fermeture par opération miroir
  - Fermeture par morphisme
- 5 Résumé

## Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

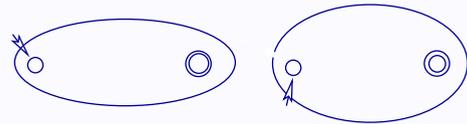
- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
  - Fermeture par union et concaténation
    - Fermeture par opération miroir
    - Fermeture par morphisme
- 5 Résumé

## Fermeture des langages à états par union et concaténation

### Union

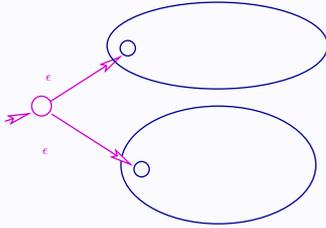


### Concaténation

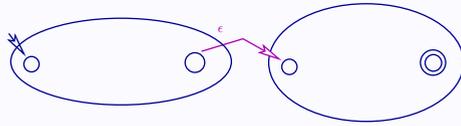


## Fermeture des langages à états par union et concaténation

### Union



### Concaténation



## Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
  - Fermeture par union et concaténation
  - Fermeture par opération miroir
  - Fermeture par morphisme
- 5 Résumé

## Opération miroir

Le miroir d'un mot est le mot écrit en lisant de droite à gauche.

### Définition (Opération miroir – mot et langage)

- Pour  $w = a_1 \cdot a_2 \cdots a_n$ , le *miroir* de  $w$  est le mot dénoté  $w^R$  et défini par :

$$w^R = a_n \cdot a_{n-1} \cdots a_1$$

- Pour  $L \subseteq \Sigma^*$ , le *miroir* de  $L$  est le langage, dénoté  $L^R$ , des mots miroirs de  $L$  :

$$L^R = \{w^R \mid w \in L\}$$

### Exemple (Opération miroir)

Pour  $L = \{001, 10, 111\}$ , on a  $L^R = \{100, 01, 111\}$ .

## Fermeture des langages à états par opération miroir

### Fermeture de EF par l'opération miroir

- Si  $L \subseteq \Sigma^*$  est un langage à états, alors ainsi est  $L^R$ .
- Donc EF est fermé par l'opération miroir.

### Preuve informelle basée sur les automates

Étant donné un langage  $L$  à états et son automate reconnaisseur  $A$  :

- 1 Inverser toutes les transitions de  $A$ .
- 2 Faire de l'état initial de  $A$  l'unique état accepteur.
- 3 Créer un nouvel état initial  $q_0$  (si l'ancien état initial était accepteur, rendre ce nouvel état initial accepteur).
- 4 Ajouter une transition étiquetée par  $\epsilon$  depuis  $q_0$  vers chaque état accepteur de l'automate initial.

(La preuve est laissée sous forme d'exercice en TD.)

## Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- 1 Motivations
- 2 Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- 3 Élimination des  $\epsilon$ -transitions
- 4 Retour sur la fermeture de la classe des langages à états
  - Fermeture par union et concaténation
  - Fermeture par opération miroir
  - Fermeture par morphisme
- 5 Résumé

## Rappel : morphismes (de groupes)

### Définition (Groupe)

Un groupe est un couple  $(G, *)$  où  $G$  est un ensemble et  $*$  une opération entre éléments de  $G$  tels que pour tout  $g_1, g_2, g_3 \in G$  :

- $g_1 * g_2 \in G$ ,
- $g_1 * (g_2 * g_3) = (g_1 * g_2) * g_3$
- il existe un élément neutre  $e_G$   
( $g_1 * e_G = e_G * g_1 = g_1$ )
- chaque élément a un symétrique

Soient  $(G, \bullet)$  et  $(G', *)$  2 groupes dont les éléments neutres sont  $e_G$  et  $e_{G'}$ , respectivement.

### Définition (Morphisme)

Une application  $f : G \rightarrow G'$  est un morphisme de groupes si

$$\forall x, y \in G : f(x \bullet y) = f(x) * f(y)$$

### Exemple (Morphisme)

L'application  $f : (Z, +) \rightarrow (R, \times)$  définie par  $f(n) = 2^n$  est un morphisme de groupes.

Dans la suite, pour chaque alphabet  $\Sigma$ , nous considérons le groupe  $(\Sigma^*, \cdot)$  où  $\cdot$  est l'opération de concaténation entre mots de  $\Sigma^*$  et des morphismes pour traduire des mots sur un alphabet vers des mots sur un autre alphabet.

## Morphisme sur les mots

Soit  $\Sigma$  et  $\Sigma'$  deux alphabets.

### Définition (Morphisme de mots)

Une application  $h : \Sigma \rightarrow \Sigma'^*$  induit un morphisme  $\hat{h}$ , de  $\Sigma^* \rightarrow \Sigma'^*$  défini par :

- $\hat{h}(\epsilon) = \epsilon$ , et
- $\hat{h}(u \cdot a) = \hat{h}(u) \cdot h(a)$ .

Remarque On montre que  $\hat{h}$  est un morphisme en montrant  $\forall x, y \in \Sigma^* : \hat{h}(x \cdot y) = \hat{h}(x) \cdot \hat{h}(y)$  par induction sur  $y$  ou récurrence sur  $|y|$ . □

### Exemple (Morphisme de mots)

Considérons  $\Sigma = \{a, b\}$ ,  $\Sigma' = \{0, 1\}$  et l'application  $h : \Sigma \rightarrow \Sigma'^*$  telle que  $h(a) = 0$  et  $h(b) = 1 \cdot 1$ .

L'application  $h$  induit bien un morphisme  $\hat{h} : \Sigma^* \rightarrow \Sigma'^*$ .

En effet on a, par exemple,  $\hat{h}(b \cdot a \cdot a) = 1 \cdot 1 \cdot 0 \cdot 0 = \hat{h}(b) \cdot \hat{h}(a \cdot a)$ .

À partir de maintenant, on écrit  $h$  au lieu de  $\hat{h}$ .

## Fermeture des langages à états par morphisme

Soit  $h$  un morphisme.

### Théorème : Fermeture de EF par morphisme

Si  $L \subseteq \Sigma^*$  est un langage à états alors ainsi est son image par  $h$ , notée  $h(L)$  et définie par

$$h(L) = \{h(u) \mid u \in L\}.$$

Donc EF est fermé par morphisme.

### Preuve

Basée sur les automates.  
Laissée en exercice.

### Exemple (Fermeture de EF par morphisme)

Considérons  $\Sigma = \{a, b\}$ ,  $\Sigma' = \{0, 1\}$  et le morphisme  $\hat{h}$  (noté  $h$  ci-dessous) comme décrit précédemment et induit par l'application  $h : \Sigma \rightarrow \Sigma'^*$  telle que  $h(a) = 0$  et  $h(b) = 1 \cdot 1$ .

- Le langage  $L_1 \subseteq \Sigma^*$  contenant l'ensemble des mots avec un nombre impair de  $a$  est un langage à états.
- Le langage  $h(L_1) \subseteq \Sigma'^*$  contenant l'ensemble des mots avec un nombre impair de 0 et un nombre pair de 1 est un langage à états.

### Fermeture de EF par morphisme inverse

Soit  $h$  un morphisme et  $h^{-1}$  le morphisme inverse (application inverse).

#### Théorème : Fermeture de EF par morphisme inverse

Si  $L \subseteq \Sigma'^*$  est un langage à états alors ainsi est son image  $h^{-1}(L)$  par  $h^{-1}$  définie par

$$h^{-1}(L) = \{u \in \Sigma^* \mid \exists u' \in L : h(u) = u'\}.$$

Donc EF est fermé par morphisme inverse.

#### Preuve

Basée sur les automates.  
Laisée en exercice.

#### Exemple (Fermeture de EF par morphisme inverse)

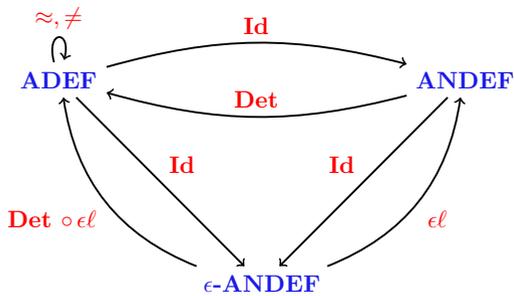
Considérons  $\Sigma = \{a, b, c, d\}$ ,  $\Sigma' = \{0, 1, 2\}$  et le morphisme  $\hat{h}$  (noté  $h$  ci-dessous) induit par l'application  $h : \Sigma \rightarrow \Sigma'$  telle que  $h(a) = 0$ ,  $h(b) = 1$ ,  $h(c) = \epsilon$  et  $h(d) = 2$ .

- Le langage  $L_1 \subseteq \Sigma'^*$  contenant l'ensemble des mots avec un nombre pair de 0 et pas de 2 est un langage à états.
- Le langage  $h^{-1}(L_1) \subseteq \Sigma^*$  contenant l'ensemble des mots avec
  - un nombre pair de 0,
  - pas de  $d$ , et
  - des  $b$  et des  $c$  de manière non-contrainte
 est un langage à états.

### Plan Chap. 8 - Automates à états finis non-déterministes avec $\epsilon$ -transitions

- Motivations
- Automates à états finis non-déterministes avec  $\epsilon$ -transitions
- Élimination des  $\epsilon$ -transitions
- Retour sur la fermeture de la classe des langages à états
- Résumé

### Résumé 1 : transformations entre automates



### Résumé 2 : fermeture de la classe des langages à états, problèmes et procédures de décision

#### Propriétés de fermeture

Les langages d'états finis sont fermés par les opérations suivantes :

- |                        |                                    |
|------------------------|------------------------------------|
| 1 union, intersection, | 2 l'étoile/la fermeture de Kleene, |
| 3 complément,          | 4 opération miroir,                |
| 5 concaténation,       | 6 morphisme, morphisme inverse.    |

Nous avons associé ces opérations à des transformations d'automates.

#### Problèmes et procédures de décision

Les problèmes de décision suivants sont décidables :

- |                     |                   |                          |
|---------------------|-------------------|--------------------------|
| 1 accessibilité,    | 2 langage vide,   | 3 inclusion de langages, |
| 4 co-accessibilité, | 5 langage infini, | 6 égalité de langages.   |

Nous avons donné une **procédure de décision** pour chaque problème.