



# INF 302 : LANGAGES & AUTOMATES

## Chapitre 6 : Automates à États Finis Non Déterministes

Yliès Falcone

[yliès.falcone@univ-grenoble-alpes.fr](mailto:yliès.falcone@univ-grenoble-alpes.fr) — [www.ylies.fr](http://www.ylies.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 2018 - 2019

# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
- 3 Applications en informatique
- 4 Résumé

# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
  - Idée et motivation
  - Définition et langage reconnu
- 2 Détermination des automates non-déterministes
- 3 Applications en informatique
- 4 Résumé

## Idée, motivation

### Idée :

- **Déterminisme** : Pour chaque état et pour chaque symbole de l'alphabet, il existe au plus un état successeur (c'est-à-dire 0 ou 1).
- **Non-déterminisme** : Pour un état et un symbole, on peut avoir 0, 1 ou plusieurs états successeurs.

### Motivations

- Il est souvent plus facile de trouver un automate non-déterministe qui reconnaît un langage  $L$  qu'un automate déterministe.
- Pour certains langages, on peut trouver un automate non-déterministe reconnaisseur qui est plus petit que tout automate déterministe reconnaisseur.

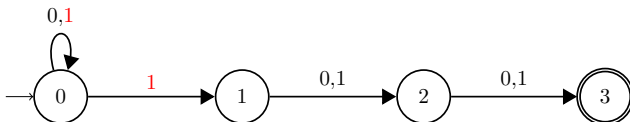
### Mais

Nous verrons qu'on ne pourra pas se passer des automates déterministes.

## Automates à états finis non-déterministes : exemple

Soit  $\Sigma = \{0, 1\}$ .

Soit  $L_3$  le langage constitué des mots de longueur  $\geq 3$  et dont le 3<sup>ième</sup> symbole en partant de la droite est 1.



Le plus petit automate déterministe qui reconnaît  $L_3$  a 8 états.

### Plus généralement

Soit  $L_k$  le langage constitué des mots de longueur  $\geq k$  et dont le  $k^{\text{ème}}$  symbole de droite est 1.

Aucun automate déterministe avec moins de  $2^k$  états ne reconnaît  $L_k$ .

# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
  - Idée et motivation
  - Définition et langage reconnu
- 2 Déterminisation des automates non-déterministes
- 3 Applications en informatique
- 4 Résumé

# Automates à états finis non-déterministes

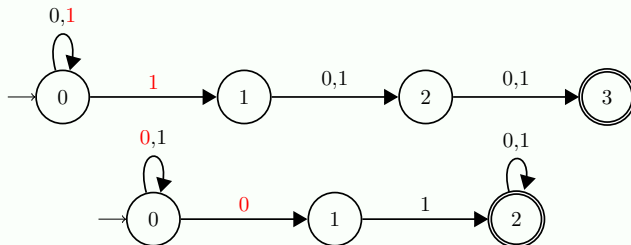
## Définition

### Définition (Automate à états finis non-déterministes (AEFND))

Un AEFND est donné par un quintuplet  $(Q, \Sigma, q_{\text{init}}, \Delta, F)$  où :

- $Q$  est un ensemble fini d'**états**,
- $\Sigma$  est l'alphabet de l'automate,
- $q_{\text{init}} \in Q$  est l'**état initial**,
- $\Delta \subseteq Q \times \Sigma \times Q$  est la **relation de transition**,
- $F \subseteq Q$  est l'ensemble des **états accepteurs**.

### Exemple (AEFND)



# Automate à états Finis Non-déterministes

Configuration, dérivation, exécution

Soit  $A = (Q, \Sigma, q_{\text{init}}, \Delta, F)$  un AEFND.

## 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)

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

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

## Définition (Exécution)

Une **exécution de l'automate  $A$**  sur le mot  $u$  est une séquence de configurations  $(q_0, u_0) \cdots (q_n, u_n)$  telle que

$$\forall i \in \{0, \dots, n-1\} : (q_i, u_i) \rightarrow_{\Delta} (q_{i+1}, u_{i+1}).$$

$$\bullet u_0 = u, \quad \bullet u_{n-1} = \epsilon, \quad \bullet q_0 = q_{\text{init}}.$$

On note par  $\xrightarrow{\star}_{\Delta}$  la fermeture réflexive et transitive de  $\rightarrow_{\Delta}$ .



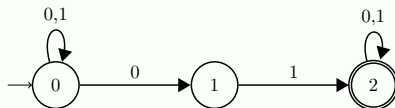
# Acceptation d'un mot par un AEFND

Soit  $A = (Q, \Sigma, q_{\text{init}}, \Delta, F)$  un AEFND

## Définition (Acceptation d'un mot)

Un mot  $u \in \Sigma^*$  est **accepté** par  $A$ , s'il existe une **exécution** de  $A$  sur  $u$  telle que l'état de sa dernière configuration soit accepteur.

## Exemple (Acceptation d'un mot par un AEFND)



Mots acceptés :

- 01 car exécution  $(0, 01) \cdot (1, 1) \cdot (2, \epsilon)$
- 001 car exécution  $(0, 001) \cdot (0, 01) \cdot (1, 1) \cdot (2, \epsilon)$

# Langage reconnu par un AEFND

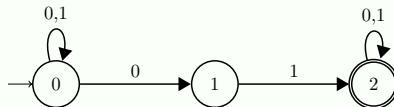
## Définition (Langage reconnu)

Le **langage reconnu par**  $A$ , noté  $L(A)$ , est l'ensemble

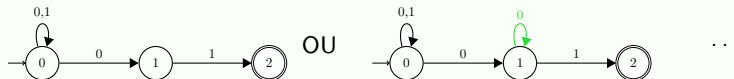
$$\{u \in \Sigma^* \mid u \text{ est accepté par } A\}.$$

## Exemple (Langage reconnu)

- Mots sur  $\Sigma = \{0, 1\}$  qui contiennent un 0 suivi d'un 1



- Mots sur  $\Sigma = \{0, 1\}$  qui terminent par un 0 suivi d'un 1



## AEFNDs vs AEFDs

Utiliser les AEFNDs facilite la conception d'un automate reconnaissant/définissant un langage.

Nous avons évidemment :

Tout AEFD est un AEFND



Par définition.

Nous allons montrer :

Tout AEFND a un AEFD équivalent

Par déterminisation (calcul des sous-ensembles).

# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
  - Procédure de détermination
  - Correction de la procédure de détermination
  - À propos de la complexité de la détermination
- 3 Applications en informatique
- 4 Résumé

# Procédure de détermination (subset construction)

## L'idée

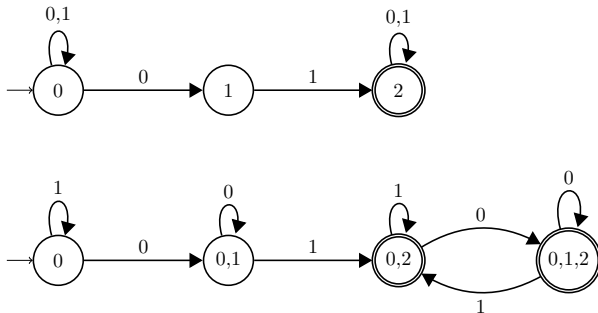
Objectif de la procédure :

- entrée : un AEFND,
- sortie : un AEFD qui reconnaît le même langage que l'automate d'entrée.

## Rabin & Scott (1959)

On va coder dans un état accessible par un mot  $u$  dans l'automate déterministe tous les états qu'on peut atteindre avec  $u$  dans l'automate non-déterministe.

Soit  $\Sigma = \{0, 1\}$ .



## Procédure de détermination

Soit  $A = (Q, \Sigma, q_{\text{init}}, \Delta, F)$  un AEFND.

### Définition (Déterminisé d'un automate)

Le **déterminisé** de  $A$ , noté  $\text{Det}(A)$ , est l'automate à états fini déterministe :

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

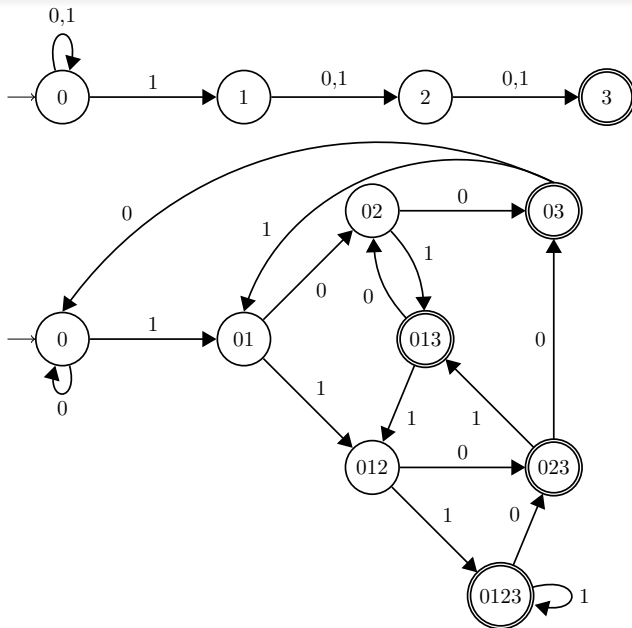
où :

- $\delta(X, a) = \{q' \mid \exists q \in X : (q, a, q') \in \Delta\}$ ,
- $\mathcal{F} = \{X \in \mathcal{P}(Q) \mid X \cap F \neq \emptyset\}$  (cad.  $X \in \mathcal{F}$  ssi  $X \cap F \neq \emptyset$ ).

### Intuition

- L'ensemble à états du déterminisé ( $\mathcal{P}(Q)$ ) est l'ensemble des sous-ensembles d'états.
- À partir d'un ensemble d'états  $X \subseteq Q$ , l'ensemble des états sur un symbole est l'ensemble des états que l'on peut atteindre à partir des états de  $X$  avec ce symbole.
- Les états accepteurs du déterminisé sont ceux qui contiennent au moins un état accepteur.

## Procédure de détermination : exemple



# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
  - Procédure de détermination
  - Correction de la procédure de détermination
  - À propos de la complexité de la détermination
- 3 Applications en informatique
- 4 Résumé



# Procédure de Détermination

## Correction de la procédure

### Théorème : correction de la procédure de détermination

$$L(\text{Det}(A)) = L(A)$$

Rappel : pour  $\delta \subseteq Q \times \Sigma \times Q$  une relation de transition (qui peut être une fonction),  $\delta^*$  dénote la fermeture réflexive et transitive de  $\delta$ .

### Extension des fonctions de transition

Représenter l'ensemble d'états atteint depuis un ensemble d'états :

- à partir d'un symbole, i.e.,  $\delta : \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$ ,
- à partir d'un mot, i.e.,  $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ .

### Preuve

Soit  $D = (Q^D, \Sigma, \{q_{\text{init}}\}, \delta_D, F^D)$  un AEFD construit par détermination de  $N = (Q^N, \Sigma, q_{\text{init}}, \delta_N, F^N)$

- Preuve de  $\delta_D^*(\{q_{\text{init}}\}, w) = \delta_N^*(q_{\text{init}}, w)$  par induction sur  $|w|$ .
- $D$  et  $N$  acceptent tous deux  $w \in \Sigma^*$  ssi  $\delta_D^*(\{q_{\text{init}}\}, w) \cap F_N \neq \emptyset$  et  $\delta_N^*(q_{\text{init}}, w) \cap F_N \neq \emptyset$ , respectivement.

# Procédure de déterminisation

## Preuve de la correction de la procédure

### Preuve de $\delta_D^*(\{q_{\text{init}}\}, w) = \delta_N^*(q_{\text{init}}, w)$ par induction sur $|w|$

**Base**  $|w| = 0$ , i.e.,  $w = \epsilon$ . D'après les définitions des fonctions de transitions pour les AEFDs et les AEFNDs, on a :  $\delta_D^*(\{q_{\text{init}}\}, \epsilon) = \delta_N^*(q_{\text{init}}, \epsilon) = \{q_{\text{init}}\}$

**Induct.** Soit  $w = x \cdot a$  de longueur  $n + 1$  et supposons que l'hypothèse vérifiée pour les séquences de longueur  $n$ .

- D'après l'hypothèse d'induction, on a  $\delta_D^*(\{q_{\text{init}}\}, x) = \delta_N^*(q_{\text{init}}, x) \subseteq Q^N$
- Soit  $\{p_1, p_2, \dots, p_k\}$  cet état
- D'après la définition inductive de  $\delta_N^*$ , on a :

$$\delta_N^*(q_{\text{init}}, w) = \bigcup_{i=1}^k \delta_N(p_i, a) \quad (\text{Eq.1})$$

- D'après la procédure de déterminisation, on a :

$$\delta_D(\{p_1, p_2, \dots, p_k\}, a) = \bigcup_{i=1}^k \delta_N(p_i, a). \quad (\text{Eq.2})$$

- En utilisant (Eq.2) et  $\delta_D^*(q_{\text{init}}, x) = \{p_1, p_2, \dots, p_k\}$  et la définition inductive de  $\delta_D^*$  pour les AEFDs :

$$\begin{aligned} \delta_D^*(\{q_{\text{init}}\}, w) &= \delta_D(\delta_D^*(\{q_{\text{init}}\}, x), a) = \delta_D(\{p_1, p_2, \dots, p_k\}, a) \\ &= \bigcup_{i=1}^k \delta_N(p_i, a). \quad (\text{Eq.3}) \end{aligned}$$

- En utilisant (Eq.1) et (Eq.3), on a  $\delta_D^*(\{q_{\text{init}}\}, w) = \delta_N^*(q_{\text{init}}, w)$ .

# Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
  - Procédure de détermination
  - Correction de la procédure de détermination
  - À propos de la complexité de la détermination
- 3 Applications en informatique
- 4 Résumé

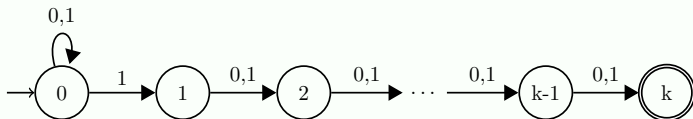
## À propos de la complexité de la détermination

En pratique, la taille de l'AEFD généré est sensiblement la même que celle de l'AEFND initial.

### Exemple (Un exemple où les choses se passent mal)

Soit  $\Sigma = \{0, 1\}$  et soit  $L_k$  le langage constitué des mots de longueur  $\geq k$  et dont le  $k$ -ième symbole de droite est 1 :

$$L_k = \{a_1 \cdots a_n \mid n \geq k \wedge a_{n-k+1} = 1\}$$



# À propos de la complexité de la détermination

Sur cet exemple

## Lemme

Aucun automate déterministe avec moins de  $2^k$  états ne reconnaît  $L_k$ .

## Preuve par contraposition

- Soit  $A = (Q, \Sigma, q_{\text{init}}, \delta, F)$  un automate déterministe tel que  $|Q| < 2^k$  et  $L(A) = L_k$ .
- Soient  $u = a_1 \cdots a_k$  et  $v = b_1 \cdots b_k$  deux mots différents de longueur  $k$  tels que  $\delta^*(q_{\text{init}}, u) = \delta^*(q_{\text{init}}, v)$ .

De tels mots doivent exister car il existe  $2^k$  différents mots de longueur  $k$  et seulement  $|Q| < 2^k$  états.

Soit  $q = \delta^*(q_{\text{init}}, u)$ .

- Comme  $u$  et  $v$  sont différents, il existe  $i$  tel que  $a_i \neq b_i$ .  
Sans perte de généralité (symétrie de  $\neq$ ), supposons  $a_i = 1$  et  $b_i = 0$ .
- Soient  $u' = u0^{i-1}$  et  $v' = v0^{i-1}$ . Alors,
  - $u'(|u'| - k + 1) = u'(k + i - 1 - k + 1) = u'(i) = a_i = 1$ , et
  - $v'(|v'| - k + 1) = b_i = 0$ .

Donc  $u' \in L_k$  et  $v' \notin L_k$ . Ceci implique  $\delta^*(q, 0^{i-1}) \in F$  et  $\delta^*(q, 0^{i-1}) \notin F$ .

Contradiction.

- Ainsi,  $\delta^*(q_{\text{init}}, u') = \delta^*(q_{\text{init}}, v')$  n'est pas possible.

## Plan Chap. 6 - Automates à États Finis Non Déterministes

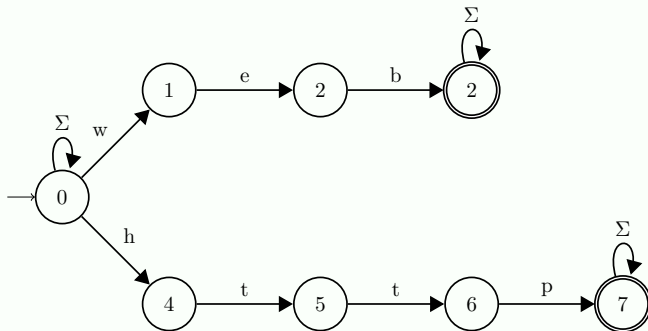
- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
- 3 Applications en informatique
- 4 Résumé

# Applications en informatique

Plusieurs applications en informatique :

- Reconnaissance de texte (web-browser).
- Compilation : analyse lexicale (reconnaissance ces mots clés d'un langage de programmation).
- Spécification de systèmes : non-déterminisme utilisé pour modéliser l'inconnu (environnement).

## Exemple (Reconnaissance d'un ensemble de mots clés)



## Plan Chap. 6 - Automates à États Finis Non Déterministes

- 1 Automates à états finis non déterministes
- 2 Détermination des automates non-déterministes
- 3 Applications en informatique
- 4 Résumé**



# Plan Chap. 6 - Automates à États Finis Non Déterministes

## Résumé

### Automates d'États-Finis Non-Déterministes

- Définition
- Critère d'acceptation, langage reconnu
- Concision des AEFND vs AEFD
- Procédure de déterminisation
  - algorithme
  - correction
  - idée sur la complexité
- Applications des AEFNDs en informatique