

IN 406 – Théorie des Langages

Cours 2 : Automate fini non-déterministe, déterministe

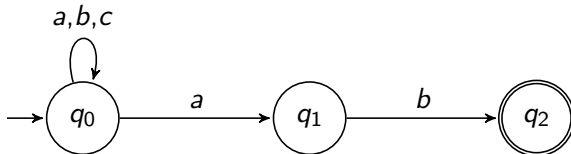
Franck Quessette – Franck.Quessette@uvsq.fr

Université de Versailles – Saint-Quentin

V4 2020–2021

Non déterminisme

Soit l'automate qui reconnaît les mots se terminant par ab sur l'alphabet $\Sigma = \{a, b, c\}$:



La reconnaissance du mot $acab$ se fait avec la séquence d'états : q_0, q_0, q_0, q_1, q_2 . La séquence q_0, q_0, q_0, q_0, q_0 se termine dans un état non final.

Cet automate est **non-déterministe** : pour le couple (q_0, a) il existe deux transitions différentes : (q_0, a, q_0) et (q_0, a, q_1) .

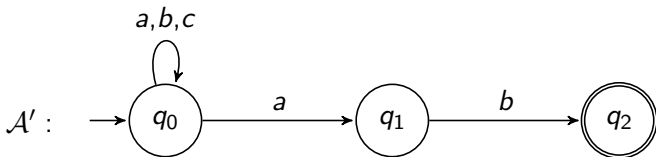
Non déterminisme

Définition

Pour un automate $\mathcal{A} = (\Sigma, Q, q_0, F, T)$, pour tout état q et toute lettre a , on définit l'**ensemble des successeurs** de (q, a) par :

$$S(q, a) = \{q' \in Q, (q, a, q') \in T\}$$

Exemple



$$S(q_0, a) = \{q_0, q_1\}$$

$$S(q_1, a) = \emptyset$$

$$S(q_2, a) = \emptyset$$

$$S(q_0, b) = \{q_0\}$$

$$S(q_1, b) = \{q_2\}$$

$$S(q_2, b) = \emptyset$$

$$S(q_0, c) = \{q_0\}$$

$$S(q_1, c) = \emptyset$$

$$S(q_2, c) = \emptyset$$

AFN – AFD

Définition

Soit un automate fini $\mathcal{A} = (\Sigma, Q, q_0, F, T)$,

- ▶ si $\exists q \in Q, \exists a \in \Sigma, |S(q, a)| > 1$
 $\Rightarrow \mathcal{A}$ est **non-déterministe** ;
- ▶ si $\forall q \in Q, \forall a \in \Sigma, |S(q, a)| \leq 1$
 $\Rightarrow \mathcal{A}$ est **déterministe** ;
- ▶ si $\forall q \in Q, \forall a \in \Sigma, |S(q, a)| = 1$
 $\Rightarrow \mathcal{A}$ est **déterministe complet** .

Notations

On note :

- ▶ **AFN** : **A**utomate **F**ini **N**on-déterministe ;
- ▶ **AFD** : **A**utomate **F**ini **D**éterministe.

ε -transition

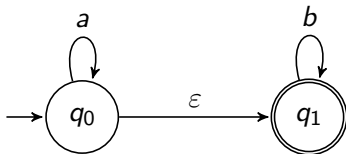
Définition

Une ε -transition dans un automate est une transition (q, ε, q') .

Un ε -chemin de q à q' existe s'il existe une suite d' ε -transitions permettant d'aller de q à q' .

Exemple

Automate reconnaissant le langage $L = \{a^n b^m, n \geq 0, m \geq 0\}$:



Les ε -transitions simplifient la construction d'automates.

Un AFD ne contient pas d' ε -transition.

AFN avec ε -transition \rightarrow AFN \rightarrow AFD

Théorème

Pour tout \mathcal{A}_1 **AFN avec ε -transitions**,
il existe \mathcal{A}_2 **AFN sans ε -transition**,
tel que $L(\mathcal{A}_2) = L(\mathcal{A}_1)$.

Théorème

Pour tout \mathcal{A}_2 **AFN sans ε -transitions**, il existe \mathcal{A}_3 **AFD**,
tel que $L(\mathcal{A}_3) = L(\mathcal{A}_2)$.

Théorème

Pour tout \mathcal{A}_3 **AFD**, il existe \mathcal{A}_4 **AFD complet**,
tel que $L(\mathcal{A}_4) = L(\mathcal{A}_3)$.

Théorème

Pour tout \mathcal{A}_4 **AFD**, il existe \mathcal{A}_5 **AFD minimal complet**,
tel que $L(\mathcal{A}_5) = L(\mathcal{A}_4)$.

Suppression des ε -transitions

Algorithme par fermeture arrière

Pour tout ε -chemin de q_1 à q_2 et pour toute transition (q_2, a, q_3) avec $a \neq \varepsilon$

Ajouter la transition (q_1, a, q_3)

Si q_2 est un état final, **Ajouter** q_1 dans les états finaux

Supprimer toutes les ε -transitions.

Exemple

Automate reconnaissant le langage $L = \{a^n b^m, n \geq 0, m \geq 0\}$:



La suppression des ε -transitions n'augmente pas le nombre d'états de l'automate.

Parties d'un ensemble

Notation

Pour tout ensemble E , l'ensemble des parties (sous-ensembles) de E est noté : 2^E .

Exemple

Si $E = \{x, y, z\}$ alors,

$$2^E = \{\emptyset, \{x\}, \{y\}, \{z\}, \{x, y\}, \{x, z\}, \{y, z\}, \{x, y, z\}\}.$$

Remarque

Pour tout ensemble E , $|2^E| = 2^{|E|}$. Dans l'exemple $|E| = 3$, $|2^E| = 8$ et $8 = 2^3$.

Déterminisation d'automate 1

Théorème

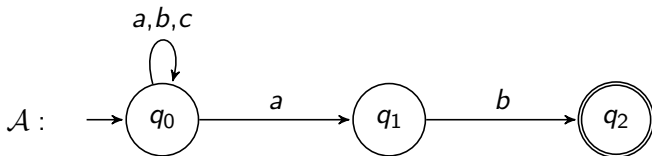
Soit $\mathcal{A} = (\Sigma, Q, q_0, F, T)$ un AFN et soit $\mathcal{A}' = (\Sigma', Q', q'_0, F', T')$ un automate fini défini par :

- ▶ $\Sigma' = \Sigma$;
- ▶ $Q' = 2^Q$ (chaque élément de Q' est un sous-ensemble de Q) ;
- ▶ $q'_0 = \{q_0\}$;
- ▶ $\forall q' \in Q'$, si $\exists q \in q'$ tel que $q \in F$, alors $q' \in F'$;
- ▶ $T' = \{(q', a, \bigcup_{q \in q'} S(q, a)), q' \in Q', a \in \Sigma'\}$.

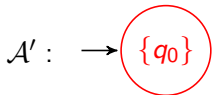
alors \mathcal{A}' est déterministe et $L(\mathcal{A}') = L(\mathcal{A})$.

Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

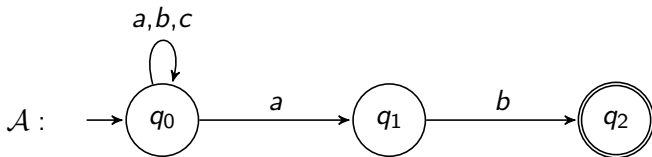


Étape 1 : État initial $q'_0 = \{q_0\}$

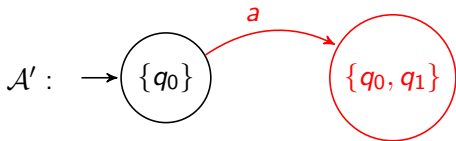


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

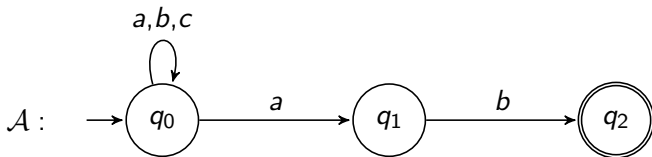


Étape 2 : $S(q_0, a) = \{q_0, q_1\}$

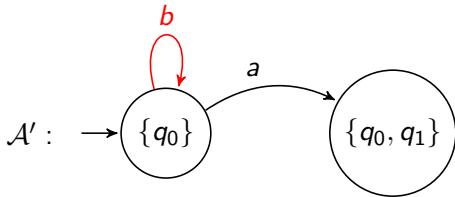


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

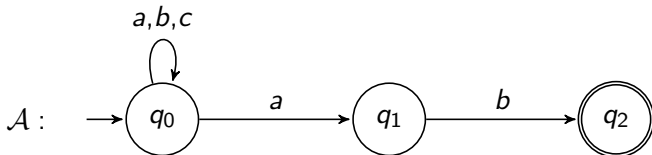


Étape 3 : $S(q_0, b) = \{q_0\}$

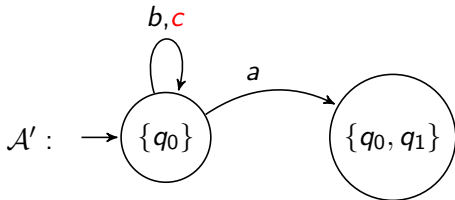


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

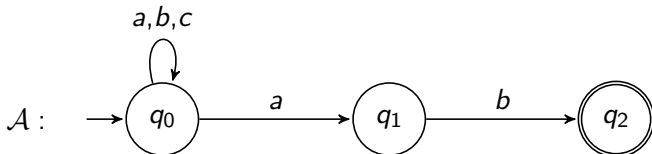


Étape 4 : $S(q_0, c) = \{q_0\}$

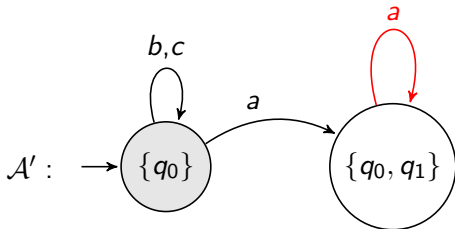


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

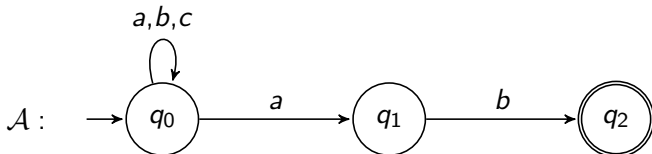


Étape 5 : $S(q_0, a) \cup S(q_1, a) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

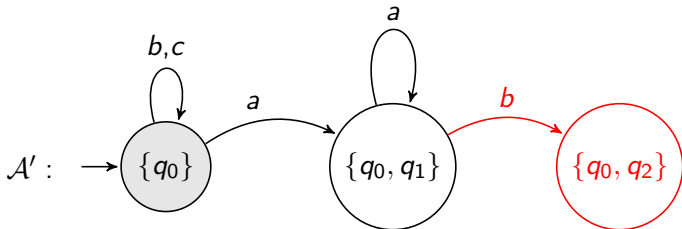


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

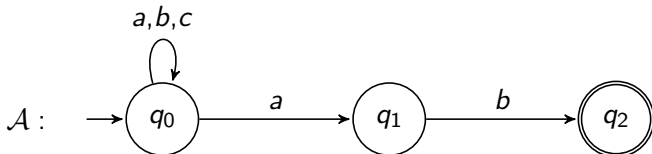


Étape 6 : $S(q_0, b) \cup S(q_1, b) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

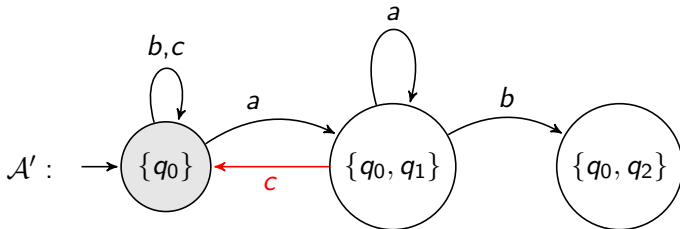


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

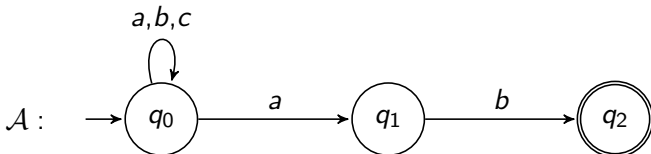


Étape 7 : $S(q_0, c) \cup S(q_1, c) = \{q_0\} \cup \emptyset = \{q_0\}$

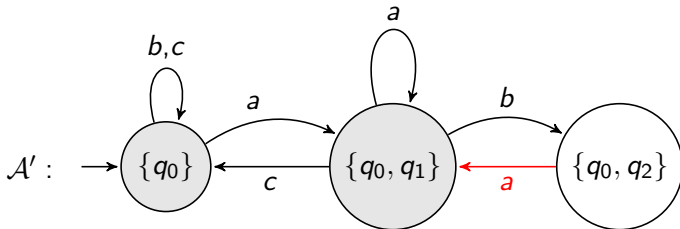


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

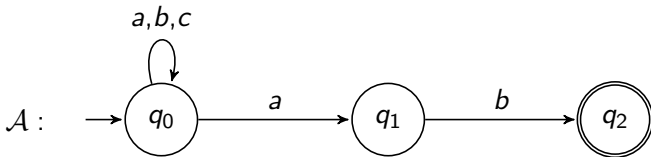


Étape 8 : $S(q_0, a) \cup S(q_2, a) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

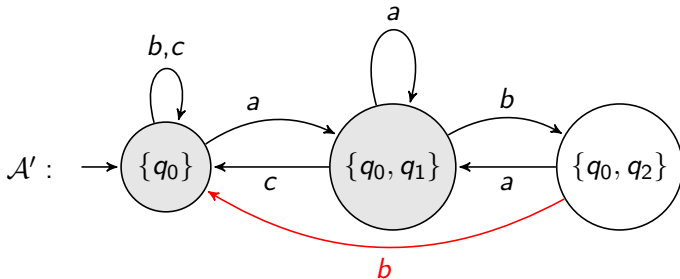


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

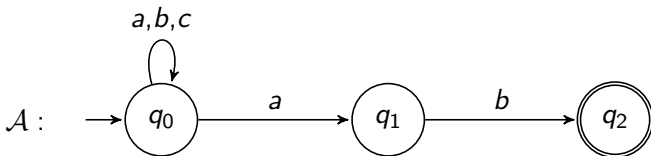


Étape 9 : $S(q_0, b) \cup S(q_2, b) = \{q_0\} \cup \emptyset = \{q_0\}$

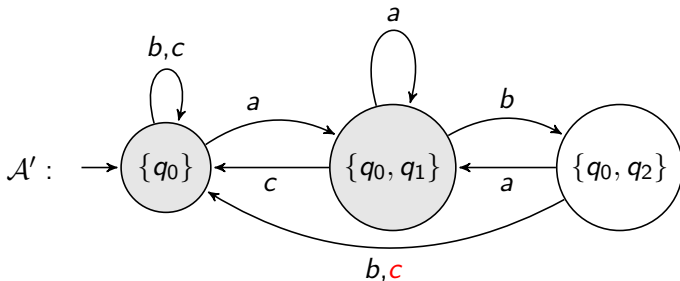


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

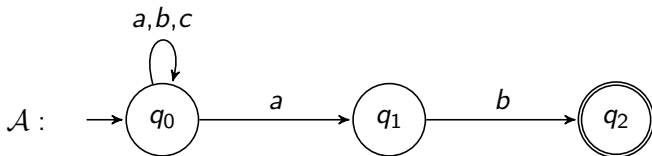


Étape 10 : $S(q_0, c) \cup S(q_2, c) = \{q_0\} \cup \emptyset = \{q_0\}$

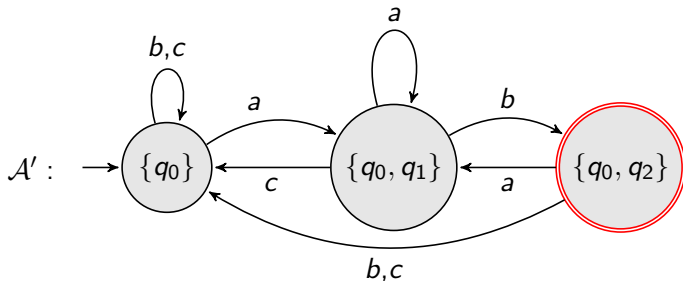


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$

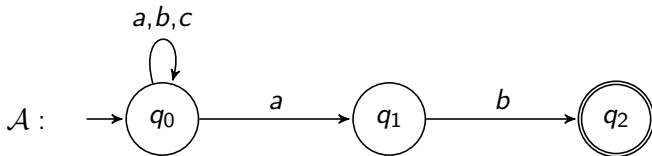


Étape 11 : États finaux $F = \{\{q_0, q_2\}\}$

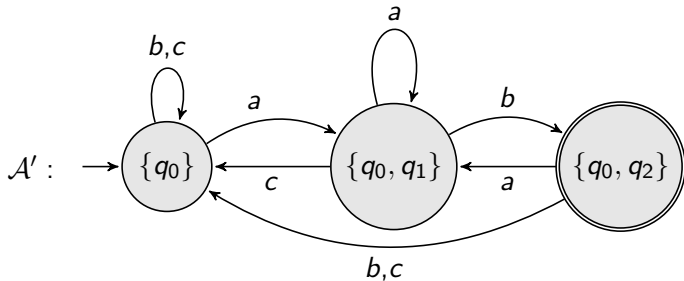


Déterminisation d'automate 2

$$\mathcal{A} = (\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, q_0, F = \{q_2\}, T)$$



Fin et $L(\mathcal{A}') = L(\mathcal{A})$



Automate fini déterministe complet

Théorème

Soit $\mathcal{A} = (\Sigma, Q, q_0, F, T)$ un AFD et soit $\mathcal{A}' = (\Sigma', Q', q'_0, F', T')$ un automate fini défini par :

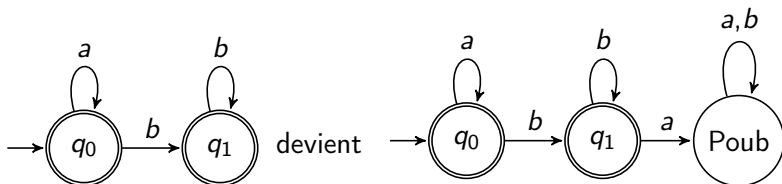
- ▶ $\Sigma' = \Sigma$;
- ▶ $Q' = Q \cup \{\text{Poubelle}\}$;
- ▶ $q'_0 = q_0$;
- ▶ $F' = F$
- ▶ $T' = T \cup \{(q, a, \text{Poubelle}) \text{ pour } (q, a) \text{ tel que } |S(q, a)| = 0\}$
 $\cup_{a \in \Sigma} \{(\text{Poubelle}, a, \text{Poubelle})\}.$

alors \mathcal{A}' est déterministe et complet et $L(\mathcal{A}') = L(\mathcal{A})$.

Automate fini déterministe complet 2

Exemple

Automate reconnaissant le langage $L = \{a^n b^m, n \geq 0, m \geq 0\}$:



Déterminisme vs Non-déterminisme 1

AFN

Les caractéristiques du **non-déterminisme** sont :

- ▶ facilité de construction de l'automate ;
- ▶ nombre d'états généralement polynomial dans la taille du problème ;
- ▶ reconnaissance d'un mot moins aisée, il faut potentiellement essayer tous les chemins possibles. Peut néanmoins se faire en temps polynomial ;
- ▶ certificat polynomial : si on donne la séquence des états qui reconnaît un mot, la vérification est rapide.

Déterminisme vs Non-déterminisme 2

AFD

Les caractéristiques du **déterminisme** sont :

- ▶ construction de l'automate difficile ;
- ▶ nombre d'états peut être exponentiel dans la taille du problème ;
- ▶ reconnaissance d'un mot polynomiale ;
- ▶ certificat de reconnaissance et de non reconnaissance d'un mot polynomial ;
- ▶ automate minimum canonique.

Déterminisme vs Non-déterminisme 3

Comparaison

Notations :

- ▶ Σ l'alphabet ;
- ▶ $ED(L)$ nombre minimum d'état dans un AFD reconnaissant L ;
- ▶ $EN(L)$ nombre minimum d'état dans un AFN reconnaissant L ;
- ▶ $TD(L)$ nombre minimum de transitions dans un AFD reconnaissant L ;
- ▶ $TN(L)$ nombre minimum de transitions dans un AFN reconnaissant L .

On a :

$$EN(L) \leq ED(L) \leq 2^{EN(L)}$$

$$TD(L) = |\Sigma| \times ED(L)$$

$$EN(L) - 1 \leq TN(L) \leq |\Sigma| \times (EN(L))^2$$

Définition

Un problème est dans **classe de complexité P** s'il existe une machine de Turing **déterministe** qui décide (problème de décision) en temps polynomial dans la taille de l'entrée.

Définition

Un problème est dans **classe de complexité NP**

- ▶ s'il existe une machine de Turing **non-déterministe** qui décide (problème de décision) en temps polynomial dans la taille de l'entrée ; **ou bien**
- ▶ s'il existe une machine de Turing **déterministe** qui est capable de décider positivement en temps polynomial dans la taille de l'entrée. C'est la notion de **certificat**.

On a $P \subseteq NP$, mais on ne sait pas si $P \neq NP$.