Théorie des Langages 1

Cours 2 : Opérations sur les langages, automates finis

L. Rieg (*thanks* M. Echenim)

Grenoble INP - Ensimag, 1re année

Année 2020-2021

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \\
L.M &\stackrel{\text{def}}{=} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} \\
L^0 &\stackrel{\text{def}}{=} \\
L^* &\stackrel{\text{def}}{=} \qquad\qquad\qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$L \cup M \stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\}$$

$$L.M \stackrel{\text{def}}{=}$$

$$\forall i > 0,\ L^i \stackrel{\text{def}}{=}$$

$$L^0 \stackrel{\text{def}}{=}$$

$$L^* \stackrel{\text{def}}{=} \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} \\
L^0 &\stackrel{\text{def}}{=} \\
L^* &\stackrel{\text{def}}{=} \qquad\qquad\qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{array}{rcll}
L \cup M & \stackrel{\text{def}}{=} & \{w \in V^* \mid w \in L \text{ ou } w \in M\} & \\
L.M & \stackrel{\text{def}}{=} & \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} & \\
\forall i > 0,\ L^i & \stackrel{\text{def}}{=} & L.L^{i-1} & \text{concaténation de } i \text{ mots de } L \\
L^0 & \stackrel{\text{def}}{=} & & \\
L^* & \stackrel{\text{def}}{=} & \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=} &
\end{array}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{aligned}
L \cup M &\overset{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\overset{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\overset{\text{def}}{=} L.L^{i-1} \qquad \text{concaténation de } i \text{ mots de } L \\
L^0 &\overset{\text{def}}{=} \{\varepsilon\} \\
L^* &\overset{\text{def}}{=} \qquad\qquad \text{et} \qquad L^+ \overset{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{array}{rcll}
L \cup M & \stackrel{\text{def}}{=} & \{w \in V^* \mid w \in L \text{ ou } w \in M\} & \\
L.M & \stackrel{\text{def}}{=} & \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} & \\
\forall i > 0,\ L^i & \stackrel{\text{def}}{=} & L.L^{i-1} & \text{concaténation de } i \text{ mots de } L \\
L^0 & \stackrel{\text{def}}{=} & \{\varepsilon\} & \\
L^* & \stackrel{\text{def}}{=} & \bigcup_{i \geq 0} L^i \quad \text{et} \quad L^+ \stackrel{\text{def}}{=} &
\end{array}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{array}{rcll}
L \cup M & \stackrel{\text{def}}{=} & \{w \in V^* \mid w \in L \text{ ou } w \in M\} & \\
L.M & \stackrel{\text{def}}{=} & \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} & \\
\forall i > 0,\ L^i & \stackrel{\text{def}}{=} & L.L^{i-1} & \text{concaténation de } i \text{ mots de } L \\
L^0 & \stackrel{\text{def}}{=} & \{\varepsilon\} & \\
L^* & \stackrel{\text{def}}{=} & \displaystyle\bigcup_{i \geq 0} L^i \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=} \bigcup_{i > 0} L^i &
\end{array}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$). On définit :

$$
\begin{array}{rcl}
L \cup M & \stackrel{\text{def}}{=} & \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M & \stackrel{\text{def}}{=} & \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i & \stackrel{\text{def}}{=} & L.L^{i-1} \qquad\qquad \text{concaténation de } i \text{ mots de } L \\
L^0 & \stackrel{\text{def}}{=} & \{\varepsilon\} \\
L^* & \stackrel{\text{def}}{=} & \displaystyle\bigcup_{i \geq 0} L^i \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=} \bigcup_{i > 0} L^i
\end{array}
$$

Notation : on pourra noter LM au lieu de $L.M$.

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \;\middle|\; n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \\ L^* &= \end{aligned}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \end{aligned}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{array}{rcl} LM & = & \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* & = & \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{array}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{aligned}$$

Question

Si L est un langage, peut-on avoir $\varepsilon \in L^+$?

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

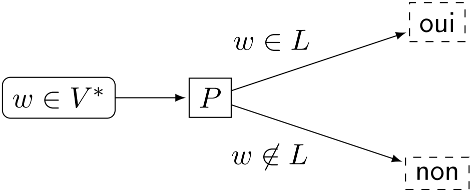
$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{aligned}$$

Question

Si L est un langage, peut-on avoir $\varepsilon \in L^+$? Oui, ssi $\varepsilon \in L$
Exemple : $L = \{\varepsilon, a\}$, $L^2 = \{\varepsilon, a, aa\}$, $L^+ = \{a^n \mid n \geq 0\} = L^*$

Les automates finis

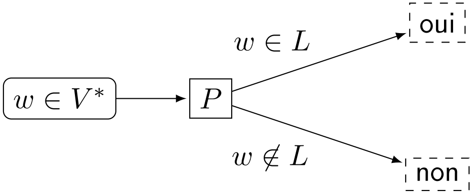
On s'intéresse à définir des « programmes » qui reconnaissent des langages.



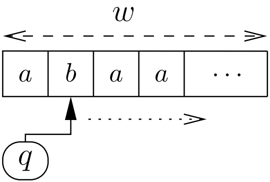
Les programmes les plus « simples » sont les automates finis.

Les automates finis

On s'intéresse à définir des « programmes » qui reconnaissent des langages.



Les programmes les plus « simples » sont les automates finis.



À chaque pas d'exécution, l'automate peut changer d'état et/ou lire un symbole et se positionner sur le symbole suivant.

Définition

Un automate fini (AF) est un quintuplet $\langle Q, V, \delta, I, F \rangle$, où :

- Q est un ensemble fini d'états
- V est le vocabulaire d'entrée
- $\delta \subseteq Q \times (V \cup \{\varepsilon\}) \times Q$ est la relation de transition
- $I \subseteq Q$ est l'ensemble des états initiaux
- $F \subseteq Q$ est l'ensemble des états acceptants (ou finaux ou finals)

Définition

Un automate fini (AF) est un quintuplet $\langle Q, V, \delta, I, F \rangle$, où :

- Q est un ensemble fini d'états
- V est le vocabulaire d'entrée
- $\delta \subseteq Q \times (V \cup \{\varepsilon\}) \times Q$ est la relation de transition
- $I \subseteq Q$ est l'ensemble des états initiaux
- $F \subseteq Q$ est l'ensemble des états acceptants (ou finaux ou finals)

Relation de transition

- Pour $a \in V$, si $(p, a, q) \in \delta$, alors étant dans l'état p et lisant un a, l'automate peut passer dans l'état q et avancer.
- Si $(p, \varepsilon, q) \in \delta$, alors étant dans l'état p, l'automate peut passer à l'état q sans avancer.

Exemple et représentation graphique

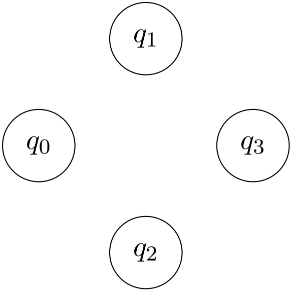
Soit A l'automate défini par :

$$\begin{array}{rcl} Q & = & \{q_0, q_1, q_2, q_3\}\,,\ V\ =\ \{a, b\}\,, \\ \delta & = & \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\} \\ I & = & \{q_0\}\,,\ F\ =\ \{q_3\} \end{array}$$

Exemple et représentation graphique

Soit A l'automate défini par :

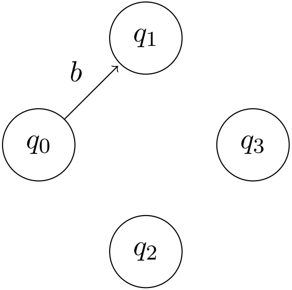
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\} \\
I &= \{q_0\}\,,\ F = \{q_3\}
\end{aligned}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

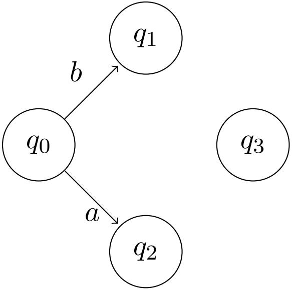
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\} \\ I &= \{q_0\}\,,\ F = \{q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

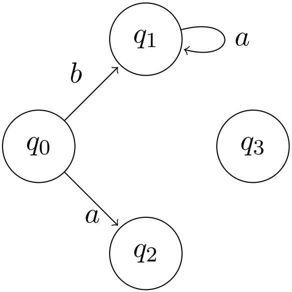
$$\begin{array}{rcl} Q & = & \{q_0, q_1, q_2, q_3\}\,,\ V \ = \ \{a, b\}\,, \\ \delta & = & \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\} \\ I & = & \{q_0\}\,,\ F \ = \ \{q_3\} \end{array}$$



Exemple et représentation graphique

Soit A l'automate défini par :

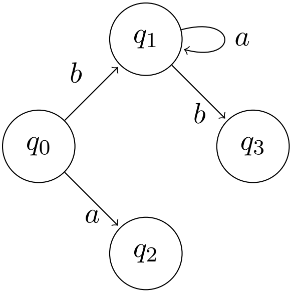
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), \color{red}{(q_1, a, q_1)}, (q_1, b, q_3), (q_2, b, q_1)\} \\ I &= \{q_0\}\,,\ F = \{q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

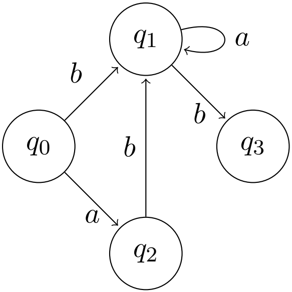
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,,\\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\}\\
I &= \{q_0\}\,,\ F = \{q_3\}
\end{aligned}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

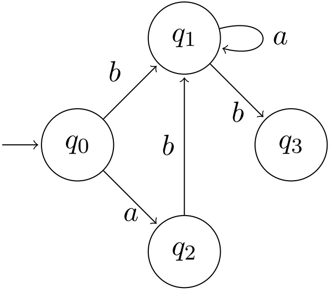
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), {\color{red}(q_2, b, q_1)}\} \\ I &= \{q_0\}\,,\ F = \{q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

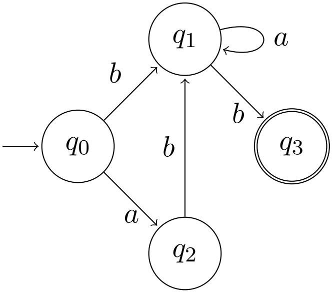
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,,\\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\}\\
I &= \{q_0\}\,,\ F = \{q_3\}
\end{aligned}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

$$
\begin{array}{rcl}
Q & = & \{q_0, q_1, q_2, q_3\}\,,\ V \ = \ \{a, b\}\,, \\
\delta & = & \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, b, q_3), (q_2, b, q_1)\} \\
I & = & \{q_0\}\,,\ F \ = \ \{q_3\}
\end{array}
$$



Langage reconnu par un automate

« Ens. des mots permettant de passer d'un état initial à un état final »

Langage reconnu par un automate

« Ens. des mots permettant de passer d'un état initial à un état final »

Définition

Soit un automate $A = \langle Q, V, \delta, I, F \rangle$.
Une configuration de A est un couple $(q, w) \in Q \times V^*$.

Soit $x \in V \cup \{\varepsilon\}$.
On dit qu'on peut passer de la configuration (q, xw)
à la configuration (q', w) dans A si $(q, x, q') \in \delta$.
On note alors $(q, xw) \vdash (q', w)$.

Langage reconnu par un automate

« Ens. des mots permettant de passer d'un état initial à un état final »

Définition

Soit un automate $A = \langle Q, V, \delta, I, F \rangle$.
Une configuration de A est un couple $(q, w) \in Q \times V^*$.

Soit $x \in V \cup \{\varepsilon\}$.
On dit qu'on peut passer de la configuration (q, xw)
à la configuration (q', w) dans A si $(q, x, q') \in \delta$.
On note alors $(q, xw) \vdash (q', w)$.

Un mot w est reconnu par A s'il existe $q_0 \in I$ et $q_n \in F$ tels que

$$(q_0, w) \vdash (q_1, w') \vdash \cdots \vdash (q_n, \varepsilon)$$

Langage reconnu par un automate

« Ens. des mots permettant de passer d'un état initial à un état final »

Définition

Soit un automate $A = \langle Q, V, \delta, I, F \rangle$.
Une configuration de A est un couple $(q, w) \in Q \times V^*$.

Soit $x \in V \cup \{\varepsilon\}$.
On dit qu'on peut passer de la configuration (q, xw)
à la configuration (q', w) dans A si $(q, x, q') \in \delta$.
On note alors $(q, xw) \vdash (q', w)$.

Un mot w est reconnu par A s'il existe $q_0 \in I$ et $q_n \in F$ tels que

$$(q_0, w) \vdash (q_1, w') \vdash \cdots \vdash (q_n, \varepsilon)$$

On note $\mathcal{L}(A)$ l'ensemble des mots reconnus par A.

Non-déterminisme

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. Card$(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Non-déterminisme

Un AF $\langle Q, V, \delta, I, F\rangle$ est dit non-déterministe si

1. $\mathrm{Card}(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Non-déterminisme

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. $\mathrm{Card}(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Non-déterminisme :

- ne donne pas immédiatement un « programme » reconnaisseur

Non-déterminisme

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. Card$(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Non-déterminisme :

- ne donne pas immédiatement un « programme » reconnaisseur
- mais facilite la définition des automates !

Déterminisme

À l'inverse, un AF $\langle Q, V, \delta, I, F\rangle$ est dit déterministe si

1. Card$(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Déterminisme

À l'inverse, un AF $\langle Q, V, \delta, I, F\rangle$ est dit déterministe si

1. $\mathrm{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\,(q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques. Mais elles n'existent pas toujours !

Déterminisme

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. $\text{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques. Mais elles n'existent pas toujours !

Définition (Automate complet)

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction : $Q \times V \to Q$.

Déterminisme :

- Donne directement un « programme » reconnaisseur
- Sera en particulier utilisé en architecture/CEP

Déterminisme (cf. cours 4)

À l'inverse, un AF $\langle Q, V, \delta, I, F\rangle$ est dit déterministe si

1. $\mathrm{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques. Mais elles n'existent pas toujours !

Définition (Automate complet)

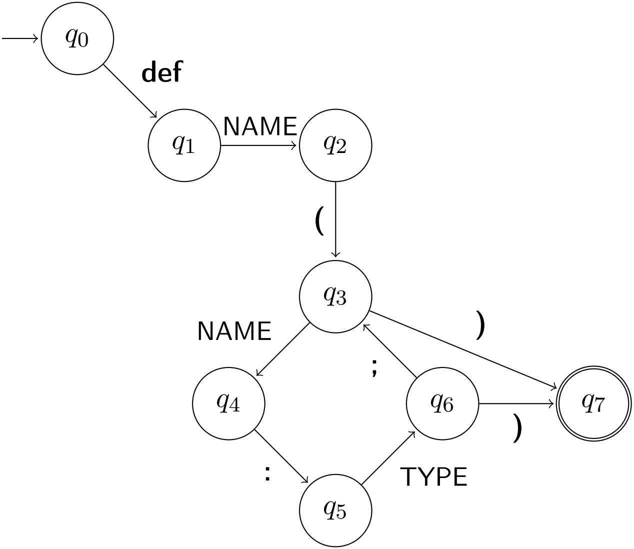
Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction : $Q \times V \to Q$.

Déterminisme :

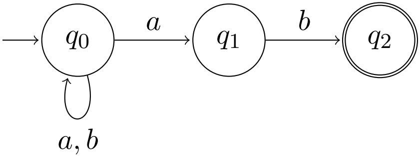
- Donne directement un « programme » reconnaisseur
- Sera en particulier utilisé en architecture/CEP

Exemples (2)

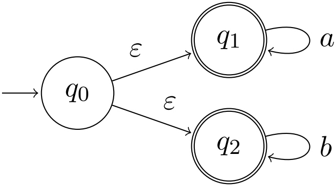


Exemples

1. Non-déterministe, sans ε-transition



2. Non-déterministe, avec ε-transition



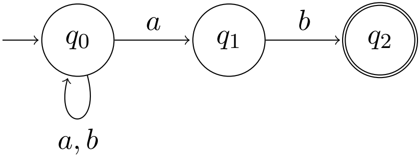
Automates équivalents

Deux automates A et A' sont équivalents ssi $\mathcal{L}(A) = \mathcal{L}(A')$.

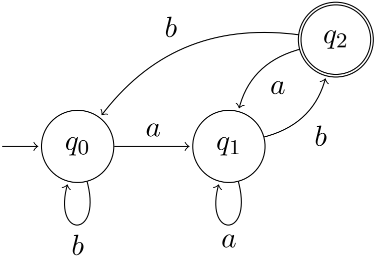
Automates équivalents

Deux automates A et A' sont équivalents ssi $\mathcal{L}(A) = \mathcal{L}(A')$.

- Automate A :



- Automate A' :



Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.

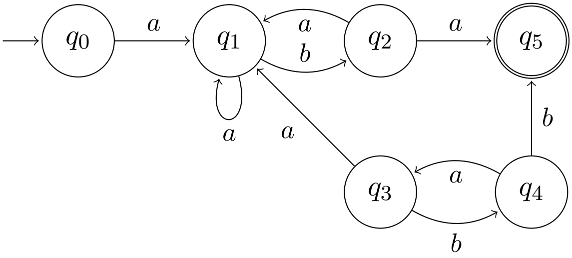
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



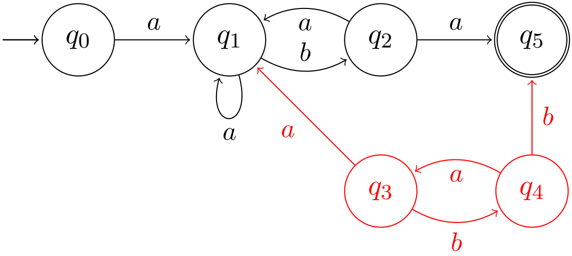
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



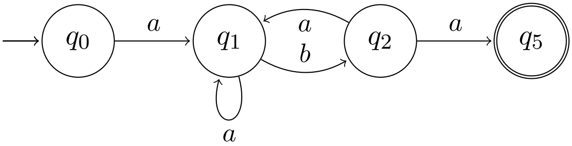
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



Exercices

Exercice 1

Construire un automate fini qui reconnaît le langage

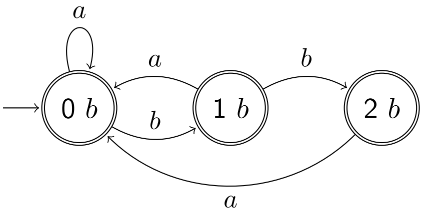
$$L = \{w \in \{a,b\}^* \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs}\}$$

Exercices

Exercice 1

Construire un automate fini qui reconnaît le langage

$$L = \{w \in \{a, b\}^* \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs}\}$$



Exercices

Exercice 2

Construire un automate fini qui reconnaît le langage

$$L = \left\{ w \in \{a, b\}^+ \;\middle|\; w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs} \right\}$$

Exercices

Exercice 2

Construire un automate fini qui reconnaît le langage

$$L = \left\{ w \in \{a, b\}^+ \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs} \right\}$$